**《数字逻辑》实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **姓名** | **魏永森** | | **年级** | | **19** |
| **学号** | **20192242** | | **专业、班级** | | **计算机科学与技术(卓越)01班** |
| **实验名称** | **实验十四 存储器实验** | | | | |
| **实验时间** | **11.27** | **实验地点** | | **DS 1410** | |
| **实验成绩** |  | **实验性质** | | **□验证性 □设计性 □综合性** | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  评语：  评价教师签名（电子签名）： | | | | | |
| 一、实验目的  1.学习并掌握寄存器的特点。  2.学习并使用时钟分频的方法。 | | | | | |
| 二、实验项目内容   1. 利用BASYS3片内存储器单元实现单端口RAM设计（带异步读和同步读两种模式），在时钟（clk）上升沿，采集地址（addr）、输入数据（data\_in）、执行相关控制信息。当写使能（we）有效，则执行写操作，否则执行读取操作。同步与异步设计仅针对读操作：对于异步RAM而言，读操作为异步，即地址信号有效时，控制器直接读取RAM阵列；对于同步RAM而言，地址信号在时钟上升沿被采集。并保存在寄存器中，然后使用该地址信号读取RAM阵列。 2. 实现双端口（同步与异步）RAM设计，相对于单端口RAM而言，双端口RAM存在两个存取端口，并且可独立进行读写操作，具有自己的地址（addr\_a、addr\_b）、数据输入（din\_a、din\_b）/输出端口（dout\_a、dout\_b）。 3. 实现FIFO设计，FIFO由存储单元队列或阵列构成，和RAM不同的是FIFO没有地址，第一个被写入队列的数据也是第一个从队列中读出的数据。FIFO可以在输入输出速率不匹配时，作为临时存储单元；可用于不同时钟域中间的同步；输入数据路径和输出数据路径之间数据宽度不匹配时，可用于数据宽度调整电路。 | | | | | |
| 三、实验设计  1）单端口 RAM 设计（带异步读和同步读两种模式），在时钟上升沿，采集地址（addr）、输入数据（data\_in）、执行相关控制信息。当写使能（we）有效，则执行写操作，否则执行读取操作。同步与异步设计仅针对读操作：对于异步 RAM 而言，读操作为异步，即地址信号有效时，控制器直接读取 RAM 阵列；对于同步 RAM 而言，地址信号在时钟上升沿被采集，并保存在寄存器中，然后使用该地址信号读取 RAM 阵列。    2）双端口（同步与异步）RAM，相对于单端口 RAM 而言，双端口 RAM 存在两个存取端口，并且可独立进行读写操作，具有自己的地（addr\_a、addr\_b）、数据输入（din\_a、din\_b）/输出端口（dout\_a、dout\_b）以及控制信号。    3）  FIFO： 设计长度为16的queue，数据位宽为8；    利用wr，rd来控制入队和出队。然后用wp(图中为w\_addr)，rp(图中为r\_addr)来记录此时 写入和读出的位置在哪里，并用七段数码管显示出来。  并利用，wp，rp的相对位置来判断队列是否为空和是否为满。 | | | | | |
| 四、实验过程或算法(关键步骤、核心代码注解等）  1）  首先给时钟降频，能够在开发板上验证  //变频  always@(posedge clk)  begin  if(cnt==100\_000\_000)  begin  cnt<=0;  Hz=~Hz;  end  else  cnt<=cnt+1;  end  在根据同步和异步的区别设计相应的程序  //读操作,同步  always@(posedge clk)  begin  if(!isyi&&!we)  begin  memaddr<=addr;  // medata1<=;  end  end    //异步  always@(addr,isyi)  begin  if(isyi && !we)  begin  medata2<=memery[addr];  end  end  2）  设计两个端口，每个端口有自己的地址输入和输出，但是都是对于同类数据进行处理。  为了能够避免同时写入的情况，根据当前操作的状态判断是否构成矛盾的情况。  assign wr\_en\_a= en\_a & wr\_a ;  assign wr\_en\_b= en\_b & wr\_b ;  assign rd\_en\_a= en\_a & !wr\_a;  assign rd\_en\_b= en\_b & !wr\_b;  assign conflict=wr\_en\_a & wr\_en\_b & (addr\_a==addr\_b);//同时写冲突  下面是根据相应的操作数进行读写操作的相应代码  //mem wdata\_a wdata\_b  always@(posedge clk or negedge rst\_n)  if(rst\_n)  for(i=0; i<16; i=i+1)  mem[i] <=0;  else begin  if(wr\_en\_a & !conflict)  mem[addr\_a]<=wdata\_a;  if(wr\_en\_b & !conflict)  mem[addr\_b]<=wdata\_b;  end  // rdata\_a rdara\_b  always@(posedge clk or negedge rst\_n)  if(rst\_n)  begin rdata\_a<=0; rdata\_b<=0; end  else begin  if(rd\_en\_a) rdata\_a<=mem[addr\_a]; //两个if必须并列，否则就是只能执行一个  if(rd\_en\_b) rdata\_b<=mem[addr\_b];  end  3）**FIFO：**为了在开发板上可以看出队列效果。我们需要给时钟降频，到1Hz(人眼可辨识).  降频代码：  integer cnt=0; //时钟分频  always @(posedge clk)begin //开发板自带100MHz的时钟信号。  if(cnt==100000000) //我们将其改为跳100M次是产生hz信号  begin  cnt<=0; //cnt重置.  hz=~hz; //hz信号的反转  end  else begin  cnt<=cnt+1;  end  end  这样我们就可以使用hz信号来代替时钟信号，它1s反转一次。  **判断FIFO的满与空：**  其中wr，rd如原理所述，代表此时执行的是写入(入队)还是读出(出队)操作。Rst为复位信号。Hz为分频后的时钟信号。  重要的是wp入队指针和rp出队指针的关系。  full\_in和empty\_in代队列的满与空。  // 产生队列已满信号  always@(posedge hz or posedge rst) begin  if(rst) full\_in <= 1'b0;  else begin // &&符号之前表示只有在wr(入队)操作的时候会产生队列满信号。&&符号之后有两种情况，一种是wp转过一圈又遇到了rp；另一种是特殊情况，wp为15，rp为0。都代表了此时队列为满。  if( (~rd && wr)&&((wp==rp-1)||(rp==4'h0&&wp==4'hf)))  full\_in <= 1'b1;  else if(full\_in && rd)  full\_in <= 1'b0;  end  end  // 产生队列为空信号  always@(posedge hz or posedge rst) begin  if(rst) empty\_in <= 1'b1;  else begin // &&符号之前表示只有在rd(出队)操作的时候会产生队列满信号。&&符号之后有种情况，一种是rp转过一圈又遇到了wp；另一种是特殊情况，wp为0，rp为15。都代表了此时队列为空。  if((rd&&~wr)&&((rp==wp-1) || (rp==4'hf&&wp==4'h0)))  empty\_in<=1'b1;  else if(empty\_in && wr) empty\_in<=1'b0;  end  end | | | | | |
| 五、实验过程中遇到的问题及解决情况(主要问题及解决情况)  1）最开始不知道如何设计同步异步，后来用了一个信号表示是否是异步，如果是同步的话，先将地址保存在寄存器中，再根据寄存器中的地址读出数据，如果是异步的话，就直接读取地址对应的数据。  2）最开始主要的问题在于不知道ram不能同时写入，所以出现了错误，后来添加了conflict变量作为判断的依据，当两个端口同时在写入的时候，显示对应的指示灯，表示矛盾，同时写入的东西无效。  3）FIFO，刚开始遇到了仿真不出结果的问题，经过代码的各种更改验证，发现需要初始化变量的值(wr,rd,wp,rp).之后就可以产生仿真结果。在这之后又产生了问题：在板子上出不来结果，经过分析得知，时钟信号频率太高，人眼观察不到。之后对时钟信号进行降频操作，并通过七段数码管显示出wp和rp还有读出的数据data\_out，使程序更易懂。 | | | | | |
| 六、实验结果及分析和（或）源程序调试过程  1）RTL分析图    2）RTL分析图    下图为两个端口同时写入的情况（左边的灯表示矛盾）    3）RTL分析图    七、小组分工情况说明  魏永森：设计并编程完成单端口异步 RAM和FIFO,并编写激励程序进行仿真验证，进行开发板的调试。  张鑫：设计并编程完成单端口同步RAM和双端口同步异步RAM,并编写激励程序进行仿真验证。 | | | | | |