1. **I2C原理介绍**

IIC(Inter-Integrated Circuit)总线是一种由PHILIPS公司开发的两线式串行总线，用于连接微控制器及其外围设备。

两条双向串行线，一条数据线SDA，一条时钟线SCL。

SDA传输数据是大端传输，每次传输8bit，即1byte。

   支持多主控(multi-master)，任何时间点只能有一个主控。

总线上每个设备都有自己的一个addr，共7个bit，广播地址全0。

扩展的10bit地址兼容7bit地址的，不过在使用10bit模式时，总线上只能挂载一个device，并且不需要A2，A1，A0地址线，因为根本就没用，但切换至7bit地址模式时需要。

系统中可能有多个同种芯片，为此addr分为固定部分和可编程部份，细节视芯片而定，看datasheet。

* 1. **I2C位传输**

数据传输：SCL为高电平时，SDA线若保持稳定，那么SDA上是在传输数据bit。

数据改变：SCL为低电平时，SDA线才能改变传输的bit。

若SDA发生跳变，则用来表示一个会话的开始或结束。

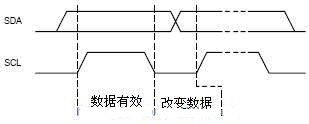


Figure 1 DATA DEFINE

* 1. **I2C开始信号与结束信号**

开始信号：SCL为高电平时，SDA由高电平向低电平跳变，开始传送数据。

结束信号：SCL为高电平时，SDA由低电平向高电平跳变，结束传送数据。

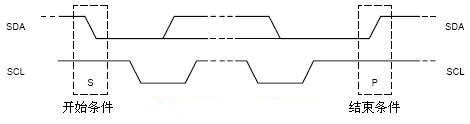


Figure 2 START&STOP

* 1. **I2C应答信号**

Master每发送完8bit数据后等待Slave的ACK。

即在第9个clock，若从IC发ACK，SDA会被拉低。

若没有ACK，SDA会被置高，这会引起Master发生RESTART或STOP流程。

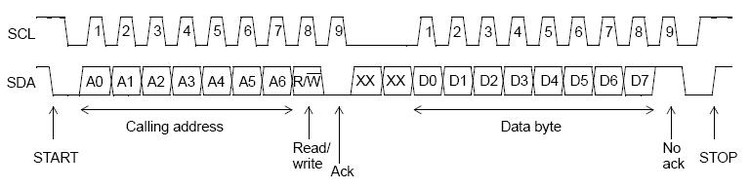


Figure 3 ACK

* 1. **I2C写操作流程**

写寄存器的标准流程为：

1.    Master发起START

2.    Master发送I2C addr（7bit）和w操作0（1bit），等待ACK

3.    Slave发送ACK

4.    Master发送reg addr（8bit），等待ACK

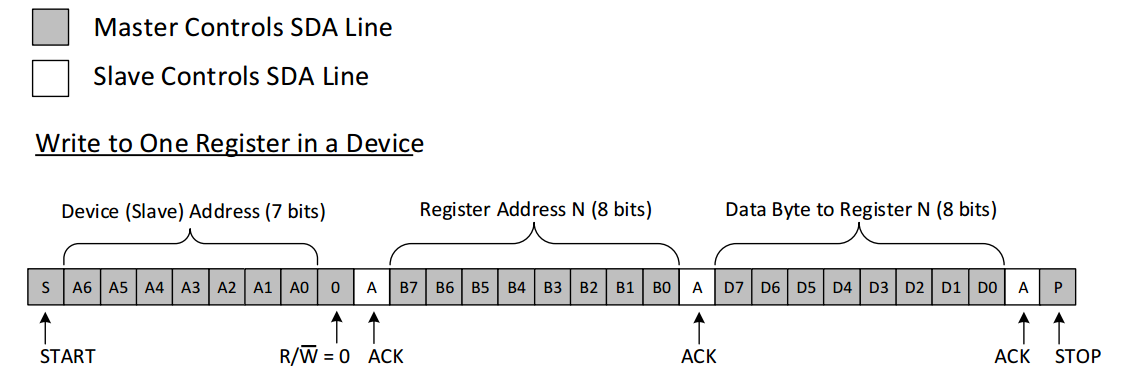
5.    Slave发送ACK

6.    Master发送data（8bit），即要写入寄存器中的数据，等待ACK

7.    Slave发送ACK

8.    第6步和第7步可以重复多次，即顺序写多个寄存器

9.    Master发起STOP



主机通过IIC总线往从机中写数据的时候，主机首先会发送一个起始信号，接着把IIC从机的7位设备地址后面添一个0(设备地址后面的0表示主机向从机写数据，1表示主机从从机中读数据)组成一个8位的数据，把这个8位的数据发给从机，发完这8位的数据以后主机马上释放SDA信号线等待从机的应答，如果从机正确收到这个数据，从机就会发送一个有效应答位0给主机告诉主机自己已经收到了数据，主机收到从机的有效应答位以后 ，接下来主机会发送想要写入的寄存器地址，寄存器发送完毕以后主机同样会释放SDA信号线等待从机的应答，从机如果正确收到了主机发过来的寄存器地址，从机会再次发送一个有效应答位给主机，主机收到从机的有效应答位0以后，接下来主机就会给从机发送想要写入从机的数据，从机正确收到这个数据以后仍然像之前两次一样会给主机发送一个有效应答位，主机收到这个有效应答位以后给从机发送一个停止信号，整个传输过程就结束了。

写一个byte

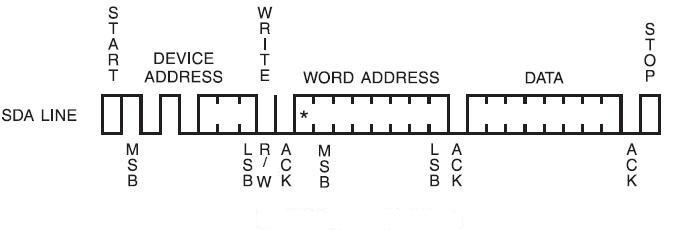


Figure 4 WRITE

写多个byte

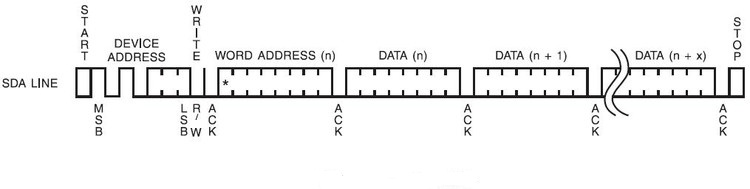


Figure 5 WRITE

* 1. **I2C读操作流程**

读寄存器的标准流程为：

1.    Master发送I2C addr（7bit）和w操作1（1bit），等待ACK

2.    Slave发送ACK

3.    Master发送reg addr（8bit），等待ACK

4.    Slave发送ACK

5.    Master发起START

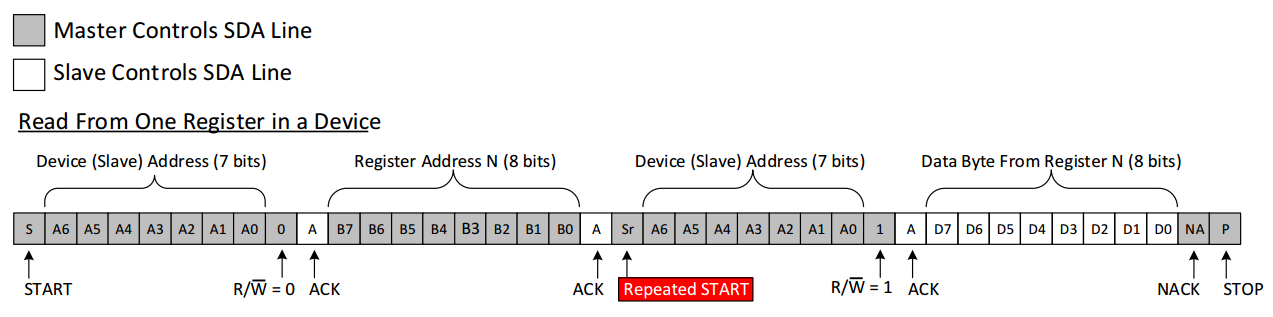
6.    Master发送I2C addr（7bit）和r操作1（1bit），等待ACK

7.    Slave发送ACK

8.    Slave发送data（8bit），即寄存器里的值

9.    Master发送ACK

10.   第8步和第9步可以重复多次，即顺序读多个byte



主机通过IIC总线从从机中读数据的过程与写数据的过程有相似之处，但是读数据的过程还多了一些额外的步骤。主机从从机读数据时主机首先会发送一个起始信号，接着把IIC从机的7位设备地址后面添一个0(设备地址后面的0表示主机向从机写数据，1表示主机从从机中读数据)，把这个8位的数据发给从机，发完这8位的数据以后主机马上释放SDA信号线等待从机的应答，如果从机正确收到这个数据，从机就会发送一个有效应答位0给主机告诉主机自己已经收到了数据，主机收到从机的有效应答位以后 ，接下来主机会发送想要读的寄存器地址，寄存器发送完毕以后主机同样会释放SDA信号线等待从机的应答，从机如果正确收到了主机发过来的寄存器地址，从机会再次发送一个有效应答位给主机，主机收到从机的有效应答位0以后，主机会给从机再次发送一次起始信号，接着把IIC从机的7位设备地址后面添一个1(设备地址后面的0表示主机向从机写数据，1表示主机从从机中读数据)，注意，第一次是在设备地址后面添0，这一次是在设备地址后面添1，把这个8位的数据发给从机，发完这8位的数据以后主机马上释放SDA信号线等待从机的应答，如果从机正确收到这个数据，从机就会发送一个有效应答位0给主机告诉主机自己已经收到了数据，接着从机继续占用SDA信号线给主机发送寄存器中的数据，发送完毕以后，主机再次占用SDA信号线发送一个非应答信号1给从机，主机发送一个停止信号给从机结束整个读数据的过程。

读一个byte

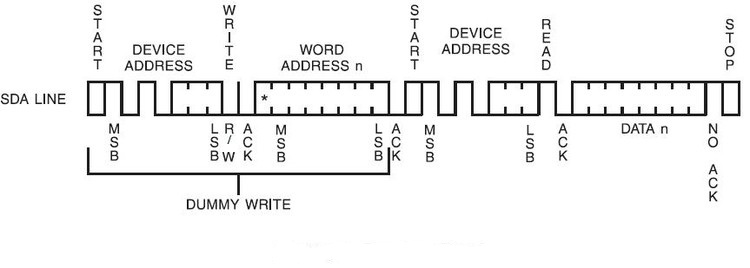


Figure 6 READ

读多个byte

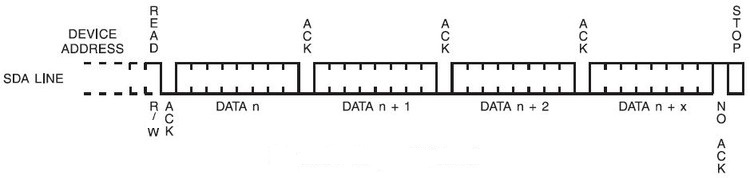
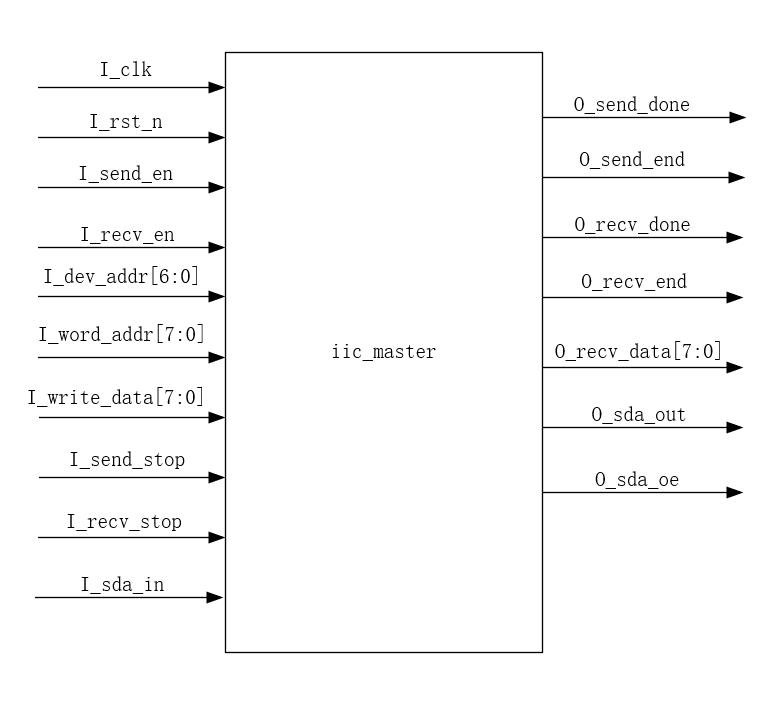


Figure 7 READ

1. **模块划分及端口说明**

**2.1 I2C主机模块**

|  |  |  |  |
| --- | --- | --- | --- |
| 端口名 | 位宽 | 方向 | 描述 |
| I\_clk | 1 | 输入 | 系统时钟 |
| I\_rst\_n | 1 | 输入 | 系统复位，低电平有效 |
| I\_send\_en | 1 | 输入 | 发送使能，高电平有效 |
| I\_recv\_en | 1 | 输入 | 接收使能，高电平有效 |
| I\_dev\_addr | 7 | 输入 | 设备地址 |
| I \_word\_addr | 8 | 输入 | 字地址 |
| I \_write\_data | 8 | 输入 | 写入的数据 |
| I\_send\_stop | 1 | 输入 | 发送操作结束信号，高电平有效 |
| I\_recv\_stop | 1 | 输入 | 接收操作结束信号，高电平有效 |
| O\_send\_done | 1 | 输出 | 发送字节完成标志，高电平有效 |
| O\_send\_end | 1 | 输出 | 发送操作完成标志，高电平有效 |
| O \_recv\_done | 1 | 输出 | 接收字节完成标志，高电平有效 |
| O\_recv\_end | 1 | 输出 | 接收操作完成标志，高电平有效 |
| O \_recv\_data | 8 | 输出 | 读出的数据 |
| I\_sda\_in | 1 | 输入 | SDA输入信号线 |
| O\_sda\_out | 1 | 输出 | SDA输出信号线 |
| O\_sda\_oe | 1 | 输出 | 设置SDA线（方向）模式（1为输出，0为输入） |
| O\_scl | 1 | 输出 | 串行时钟线 |

表2.1 I2C主机模块端口说明

注：

I\_send\_en：发送使能输入端，在写操作时要置位打开该使能端。

I\_recv\_en：接收使能输入端，在读操作时要置位打开该使能端。

I\_send\_stop：发送停止输入端，在一次写操作的最后一个字节数据发送完成之后，也即最后一次的数据发送完成标志O\_send\_done变为高电平输出时，此时要将该发送停止输入端置位，此时所做的操作是主机接口向IIC总线上发送停止信号。

I\_recv\_stop：接收停止输入端，在一次读操作的最后一个字节数据接收完成之后，也即最后一次的数据接收完成标志O\_ recv \_done变为高电平输出时，此时要将该接收停止输入端置位，此时所做的操作是主机接口向IIC总线上发送停止信号。

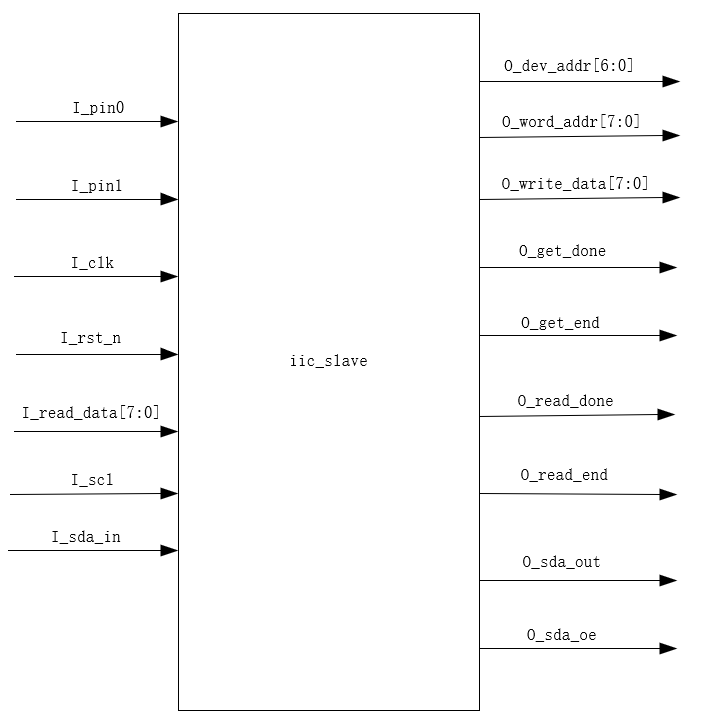
O\_send\_done：字节数据发送完成完成标志。在写操作中写入一个字节数据后，该标志位会被置1。可用来配合主机对下一次要写入数据的加载。

O\_send\_end：发送操作完成标志。在停止信号发送完成之后，该位会被置1，此时可以用来配合主机接口进行下一次的IIC通信，或者关闭写操作使能I\_send\_en。

O \_recv\_done: 字节数据接收完成完成标志。在读操作中读取一个字节数据后，该标志位会被置1。可用来配合主机对接收到的数据的正确读取。

O\_recv\_end：接收操作完成标志。在停止信号发送完成之后，该位会被置1，此时可以用来配合主机接口进行下一次的IIC通信，或者关闭读操作使能I\_recv\_en。

**2.2 I2C从机模块**



|  |  |  |  |
| --- | --- | --- | --- |
| 端口名 | 位宽 | 方向 | 描述 |
| I\_clk | 1 | 输入 | 系统时钟 |
| I\_rst\_n | 1 | 输入 | 系统复位，低电平有效 |
| I\_pin1 | 1 | 输入 | 从机7位地址的低两位 |
| I\_pin0 | 1 | 输入 | 从机7位地址的低两位 |
| I \_read\_data | 8 | 输入 | 读出的数据 |
| O\_dev\_addr | 7 | 输出 | 设备地址 |
| O \_word\_addr | 8 | 输出 | 寄存器地址 |
| O \_write\_data | 8 | 输出 | 写入数据 |
| O \_get\_done | 1 | 输出 | 写字节完成标志，高电平有效 |
| O \_get\_end | 1 | 输出 | 写操作完成标志，高电平有效 |
| O \_read\_done | 1 | 输出 | 读字节完成标志，高电平有效 |
| O \_read\_end | 1 | 输出 | 读操作完成标志，高电平有效 |
| I\_sda\_in | 1 | 输入 | SDA输入信号线 |
| O\_sda\_out | 1 | 输出 | SDA输出信号线 |
| O\_sda\_oe | 1 | 输出 | SDA线（方向）模式（1为输出，0为输入） |
| I \_scl | 1 | 输入 | 串行时钟线 |

表2.2 I2C从机模块端口说明

注：

O \_get\_done：字节数据接收完成标志。写操作中从机接口接收到一字节数据后该位被置1输出，可以用来配合从机将主机写入的该字节数据正确处理。

O \_get\_end：写操作完成标志。写操作中，接收到主机发送的停止信号后，该位会被置1，此时从机进入空闲态。

O \_read\_done：字节数据发送完成标志。读操作中从机接口发送一字节数据后该位被置1输出，可以用来配合从机将主机要读的下一字节数据正确载入。

O \_read\_end：读操作完成标志。读操作中，接收到主机发送的停止信号后，该位会被置1，此时从机进入空闲态。

1. **仿真及结果分析**

为进行仿真，先把iic\_master接口模块和iic\_slave接口模块在顶层模块中例化进去，并将相应模块的标准IIC总线相关信号线进行封装成三态门，用于两个独立模块之间的接口信号连接。如图3.1和图3.2所示。

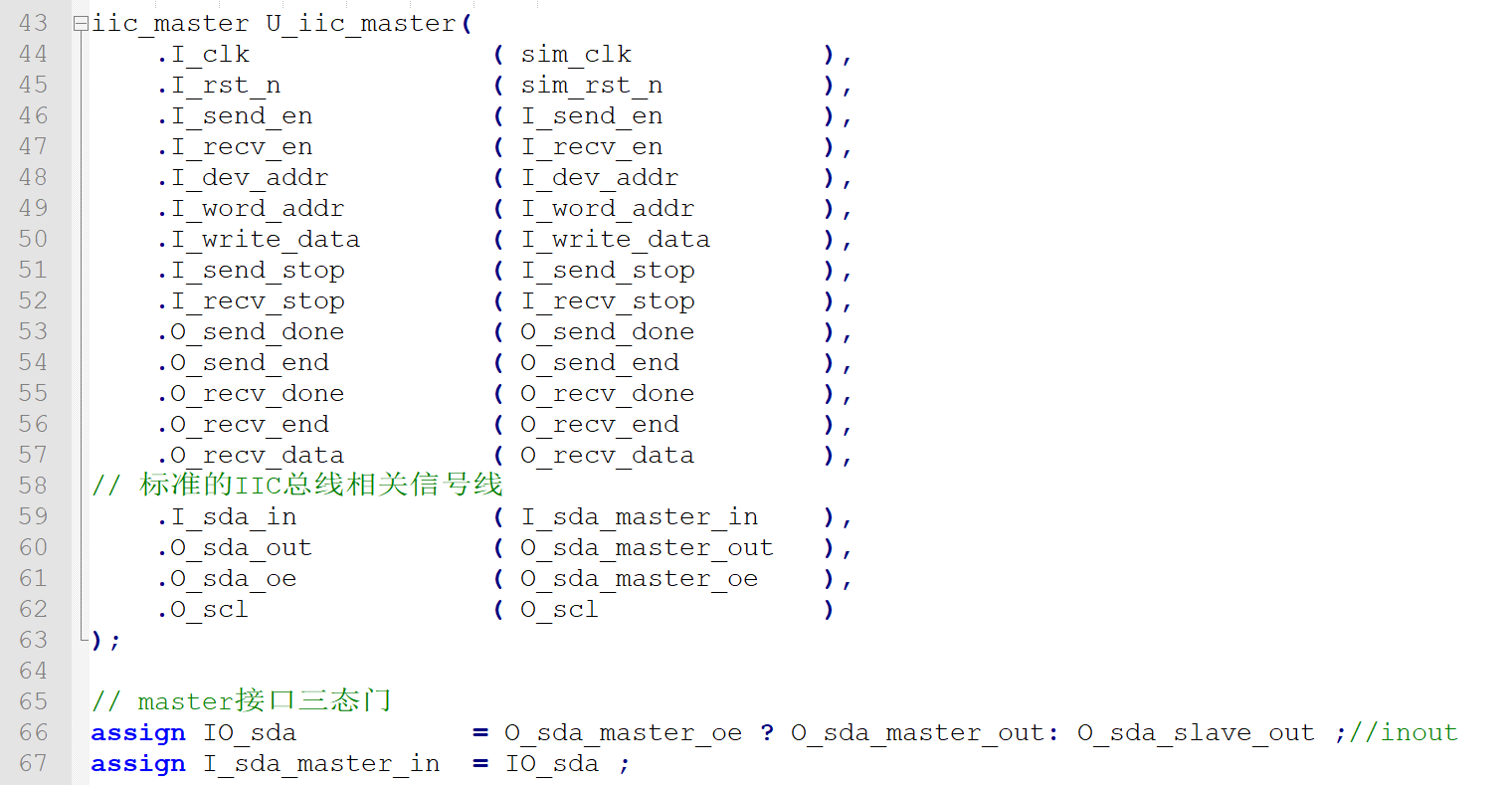
****

图3.1 IIC主机模块三态门封装

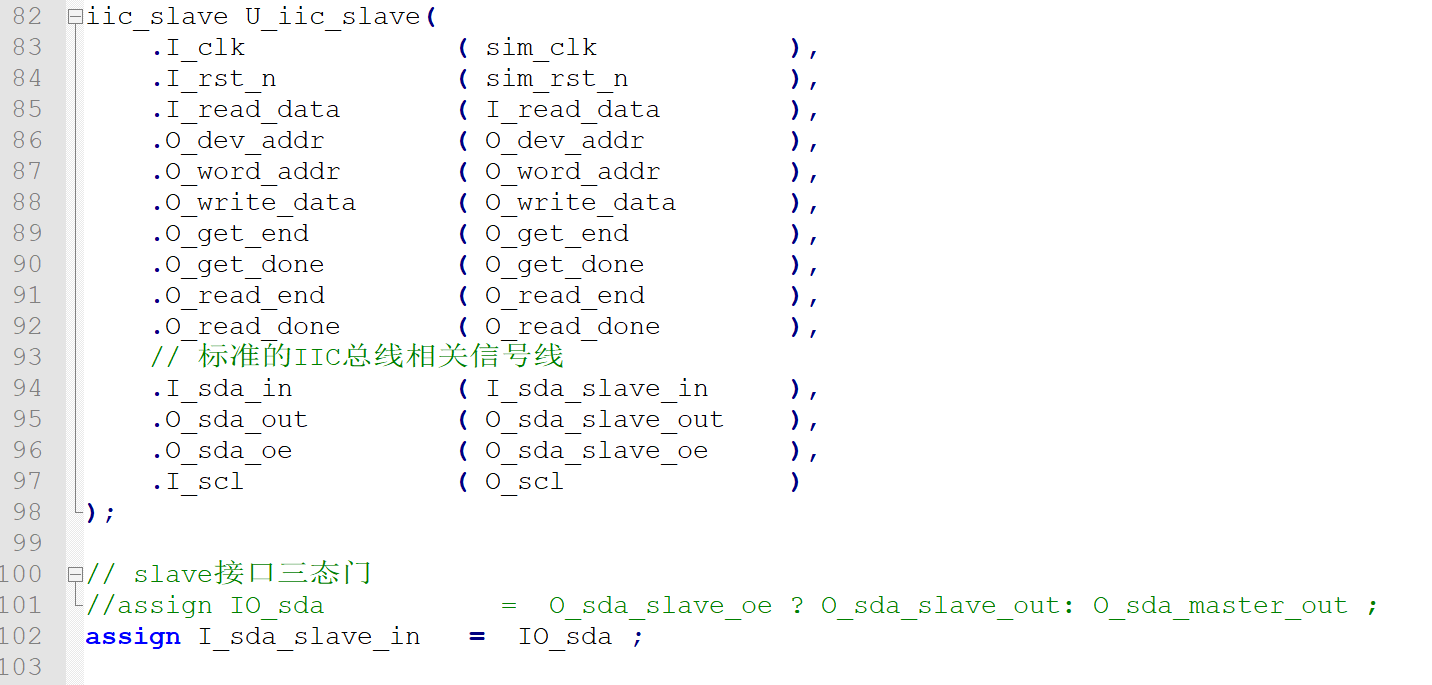


图3.2 IIC从机模块三态门封装

* 1. **写操作仿真**

**1、8字节数据写操作仿真**

根据IIC协议，IIC接口应能完成多字节数据连续写操作。在顶层模块中完成写操作激励如下图3.3所示，本次写入的数据是0x01~0x08。

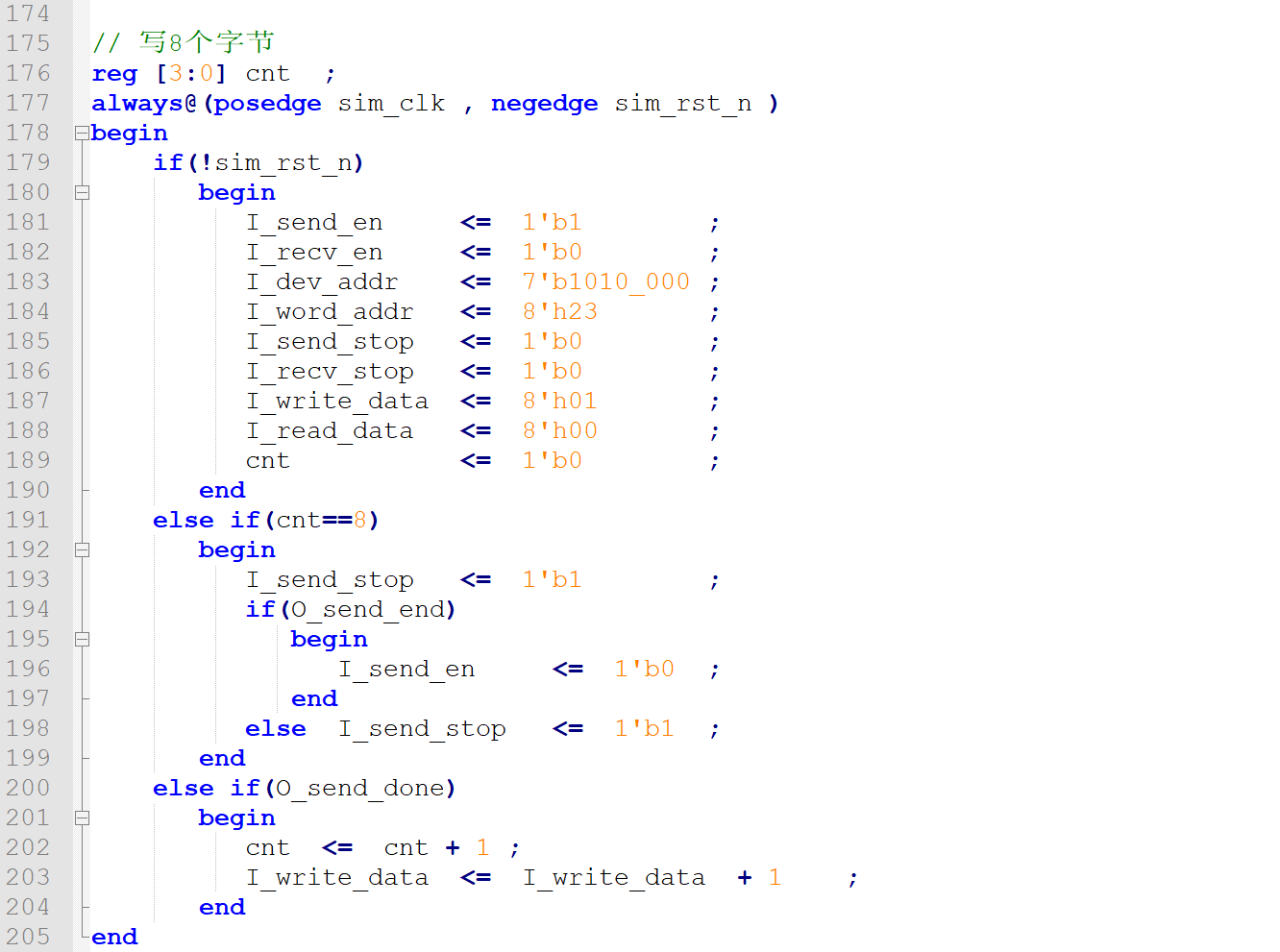


图3.3 连续写8字节仿真激励

利用Modlesim仿真，得到的仿真波形图如图3.4所示。由仿真波形图容易观察到，在主机输出的SCL信号线连续高电平期间，主机SDA信号线出现下降沿，为本次写操作的起始信号。随后主机向从机发送7位器件地址0x50，第8位发送0，表示向从机写入器件地址。从机接收到器件地址之后确认是自己的器件地址，然后在SCL低电平期间将SDA信号线拉低，在SCL高电平期间向主机输出应答位0。随后主机向从机写入字地址0x23，同样从机接收到8位的字节地址后给主机发送应答位0。接着主机就开始向从机连续写入字节数据0x01~0x08，从机每接受到一字节数据都会向主机发送一次应答位0，同时产生一个字节数据接收完成标志O\_get\_done高电平，用于配合从机设备将该数据正确读取。在第8个字节数据写入完成后，主机在SCL高电平期间将SDA释放，SCL和SDA都转向空闲态高电平，等待下一次起始信号产生。如下图3.4所示，I\_write\_data为主机接口写入的数据，O\_write\_data为从机接口接收到的主机接口写入的数据，可以看到O\_write\_data可以正确接收到0x01~0x08的字节数据。

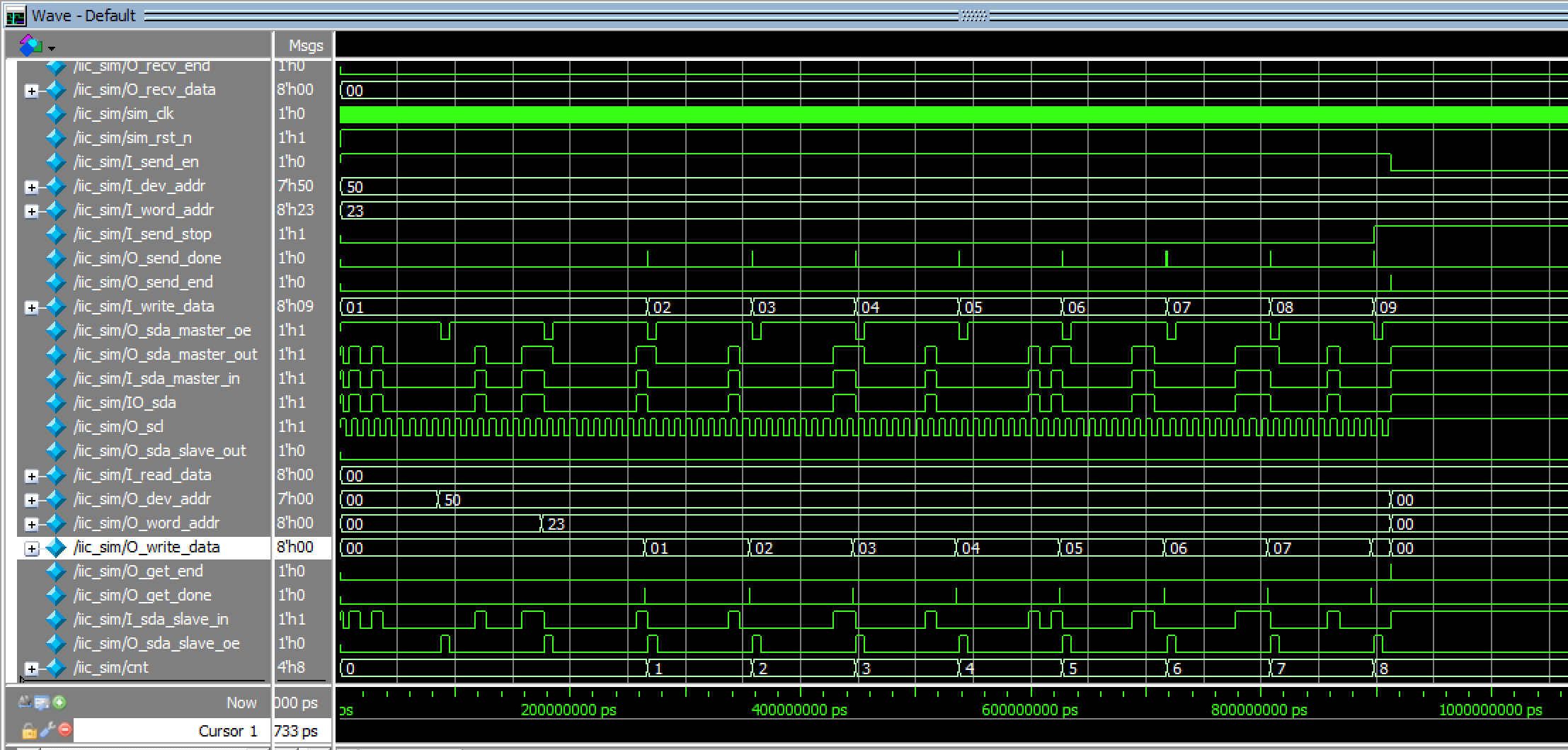


图3.4 连续写8字节操作仿真波形

**2、长字节数据写操作仿真**

为了仿真所设计的IIC接口能正确完成0x00~0xff的任意字节数据的正确写操作，进行长字节数据0x00~0xff的连续写操作仿真。仿真激励如图3.5所示。得到的仿真波形如图3.6所示。

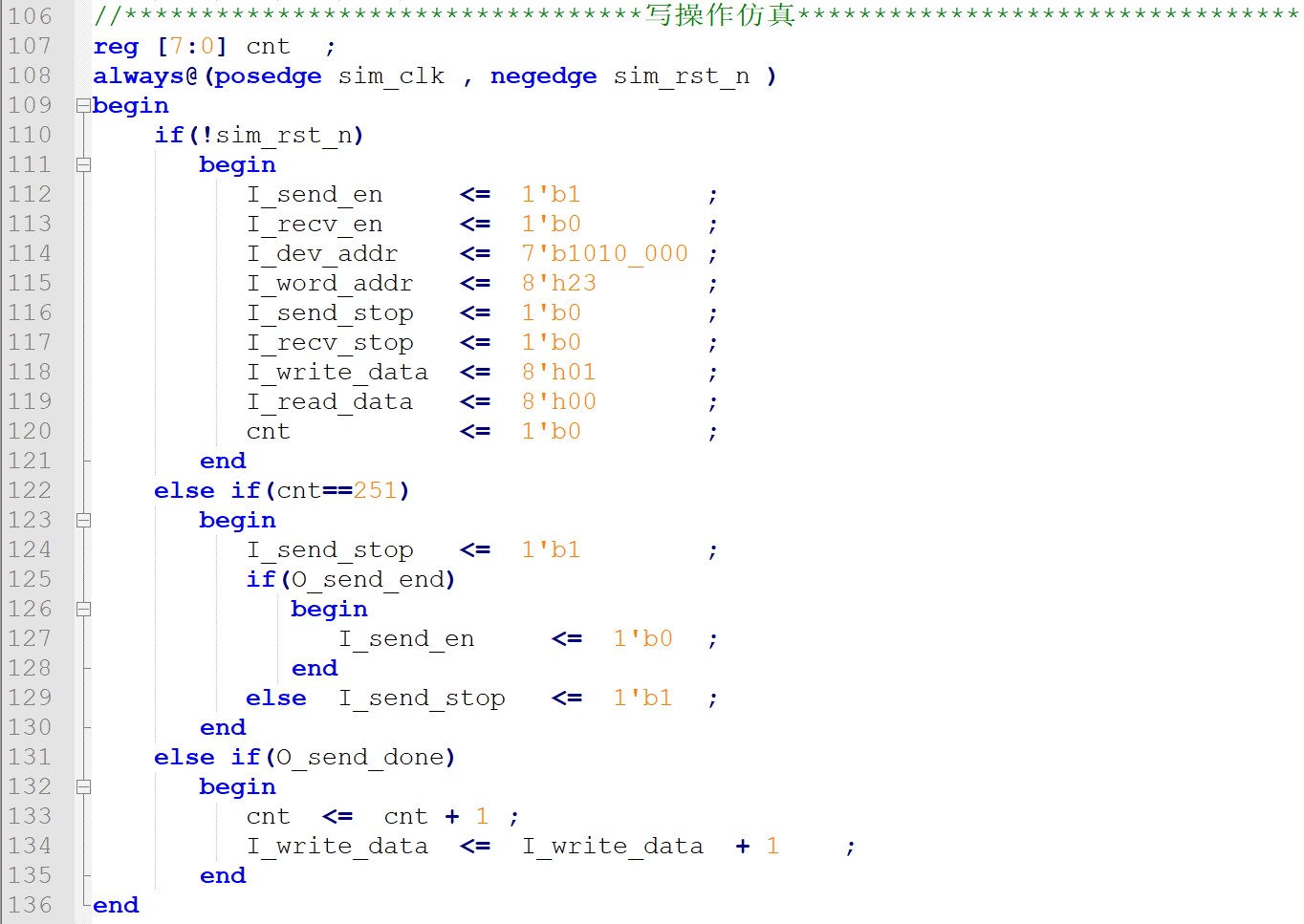


图3.5 连续写多字节仿真激励

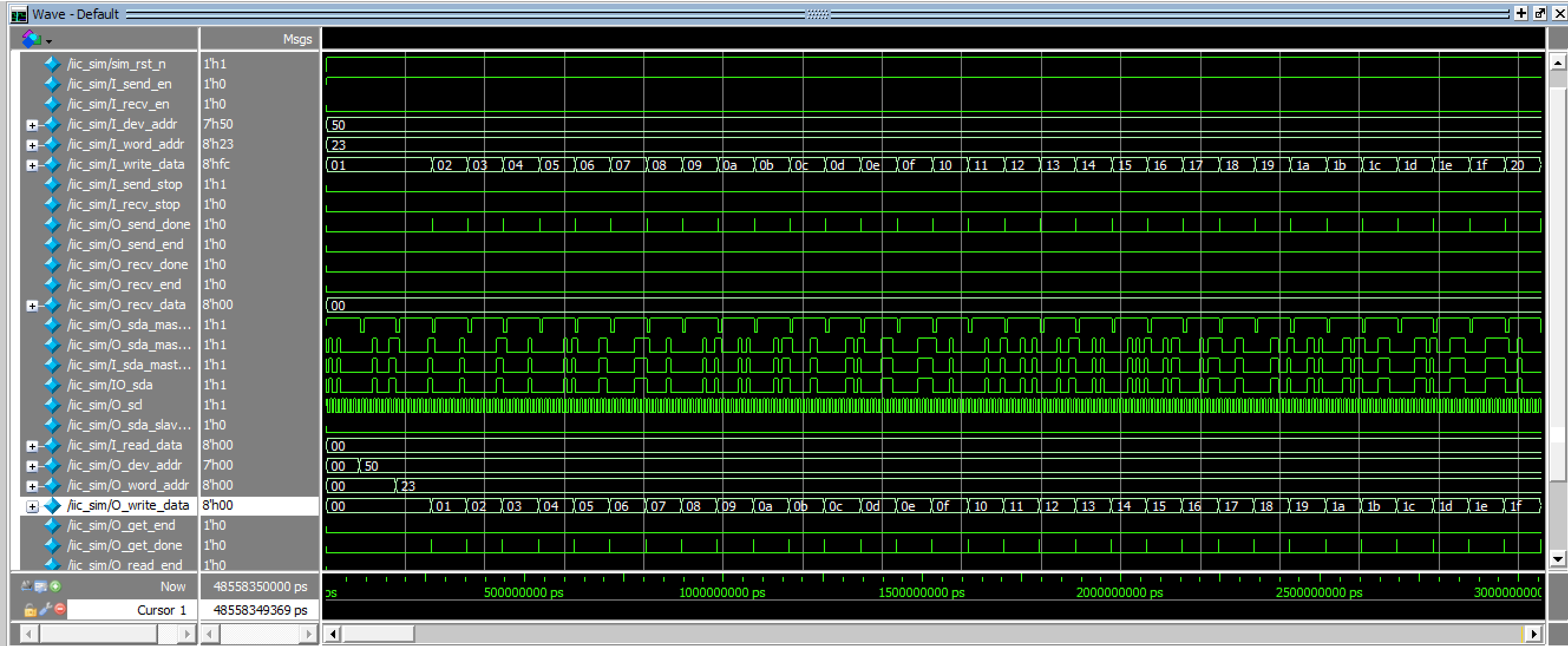


图3.6 连续写多字节数据0x00~0xff仿真波形

* 1. **读操作仿真**

**1、8字节数据读操作仿真**

根据IIC协议，IIC接口应能完成多字节数据连续读操作。在顶层模块中完成读操作激励如下图3.3所示，本次读操作的数据是0x01~0x08。



图3.7 连续读8字节仿真激励

利用Modlesim仿真，得到的仿真波形图如图3.8所示。由仿真波形图容易观察到，在主机输出的SCL信号线连续高电平期间，主机SDA信号线出现下降沿，为本次写操作的起始信号。随后主机向从机发送7位器件地址0x50，第8位发送0，表示向从机写入器件地址。从机接收到器件地址之后确认是自己的器件地址，然后在SCL低电平期间将SDA信号线拉低，在SCL高电平期间向主机输出应答位0。随后主机向从机写入字地址0x23，同样从机接收到8位的字节地址后给主机发送应答位0。与写操作不同的时，主机接收到此次应答位后，会再次发起一次起始信号，然后第二次向从机发送器件地址，但是第八位为高电平1表示读操作。在接收到从机应答位之后，主机便开始从从机读数据。

主机就开始从从机接口连续读出字节数据0x01~0x08，主机每读出一字节数据都会向主机发送一次应答位0，同时产生一个字节数据接收完成标志O\_recv\_done高电平，用于配合主机设备将该数据正确读取。在第8个字节数据读出完成后，主机在SCL高电平期间将SDA释放，SCL和SDA都转向空闲态高电平，等待下一次起始信号产生。如下图3.8所示，I\_read\_data为从机接口待读出的数据，O\_recv\_data为主机接口读到的来自从机接口的数据，可以看到O\_recv\_data可以正确接收到0x01~0x08的字节数据。

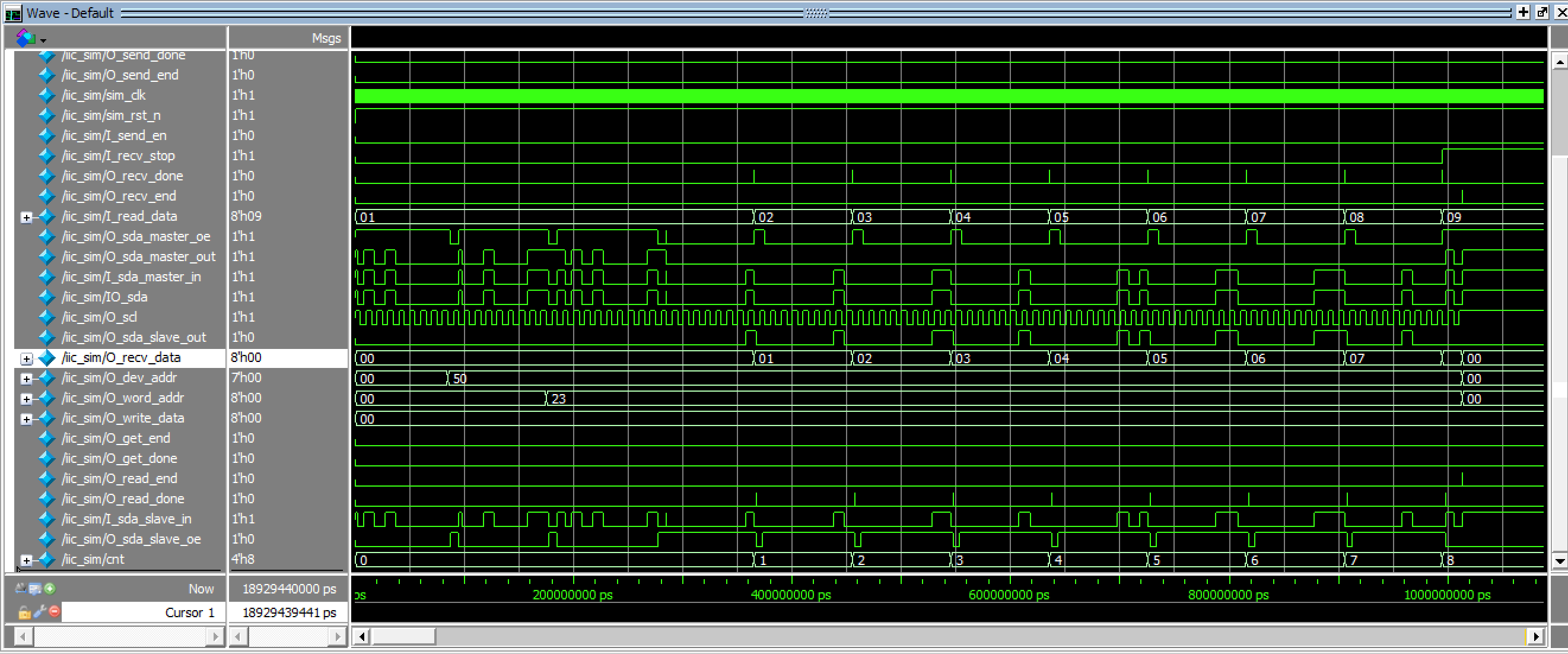


图3.8 连续读8字节操作仿真波形

**2、长字节数据读操作仿真**

为了仿真所设计的IIC接口能正确完成0x00~0xff的任意字节数据的正确读操作，进行长字节数据0x00~0xff的连续读操作仿真。仿真激励如图3.9所示。得到的仿真波形如图3.10所示。

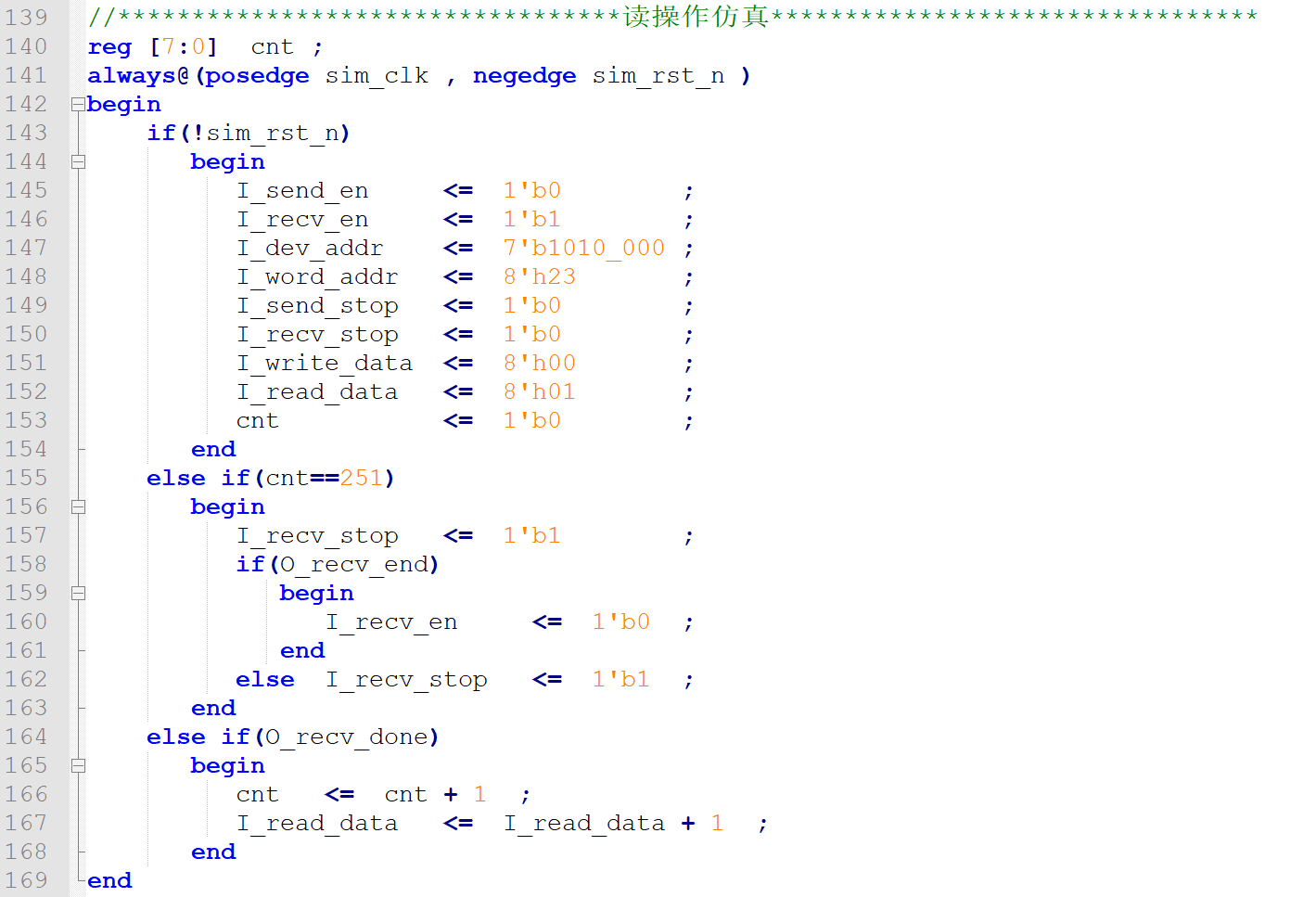


图3.9 连续读多字节仿真激励

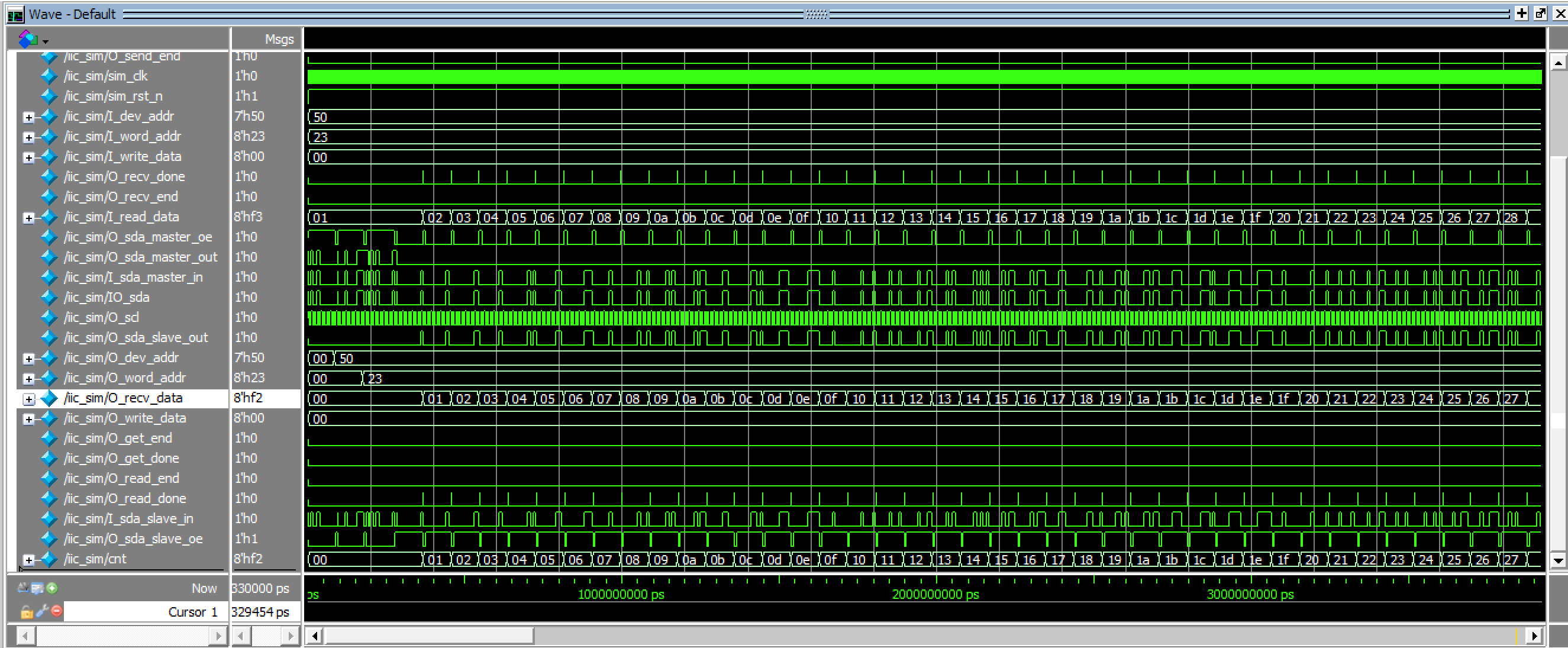


图3.10 连续读多字节数据0x00~0xff仿真波形

1. **FPGA验证**
   1. **主机iic\_master接口调用板载的24C02 EEPROM进行FPGA验证**

验证示意图如图4.1所示。FPGA作为主机，板载的24C02芯片作为从机，且该芯片自带有标准的IIC从机接口。因此只需配置好管脚约束，并将所设计的IIC主机iic\_master接口烧写比特流文件后下载到FPGA开发板中，即可验证主机的读、写操作的正确性。

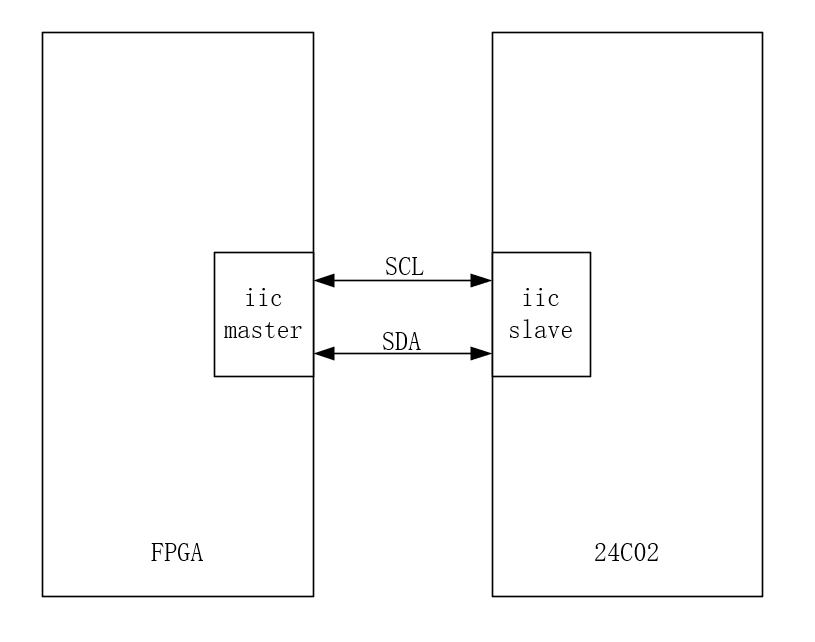


图4.1 iic\_master接口验证结构图

**4.1.1 写操作**

首先，利用24C02芯片验证所设计的iic\_master接口的写操作的功能正确性。进行FPGA验证中利用ILA捕捉到的波形如图4.2所示，可以看到I\_write\_data[7:0]为0x45时表示成功将0x45这一字节数据写入了器件地址为1010000的24C02芯片的0x23这一存储单元中。表示所设计的iic\_master接口的写操作的功能正确性得到验证。

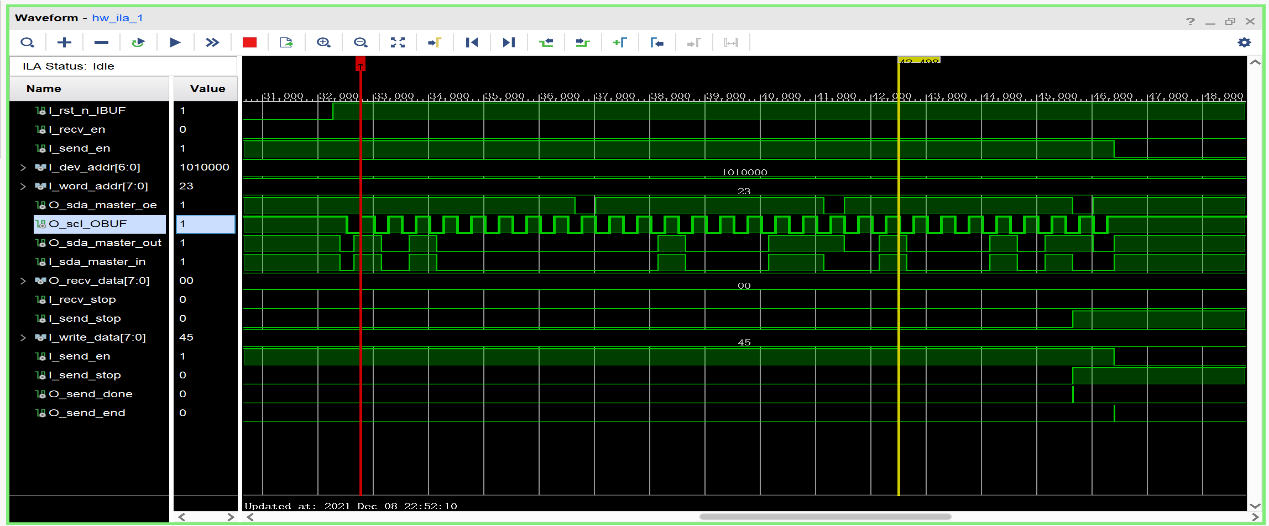


图4.2 iic\_master接口写操作验证波形图

**4.1.2 读操作**

再利用24C02芯片验证所设计的iic\_master接口的读操作的功能正确性。

这里将上述写入24C02芯片的0x23存储单元的0x45这一自己数据若能正确读出来，则可验证验证所设计的iic\_master接口的读操作的功能正确性。

进行FPGA验证中利用ILA捕捉到的波形如图4.3所示。可见当O\_recv\_data[7:0]为0x45时表示将上述所写入到器件地址为1010000的24C02芯片的0x23这一存储单元中的一字节数据0x45正确读出。表示所设计的iic\_master接口的读操作的功能正确性得到验证。

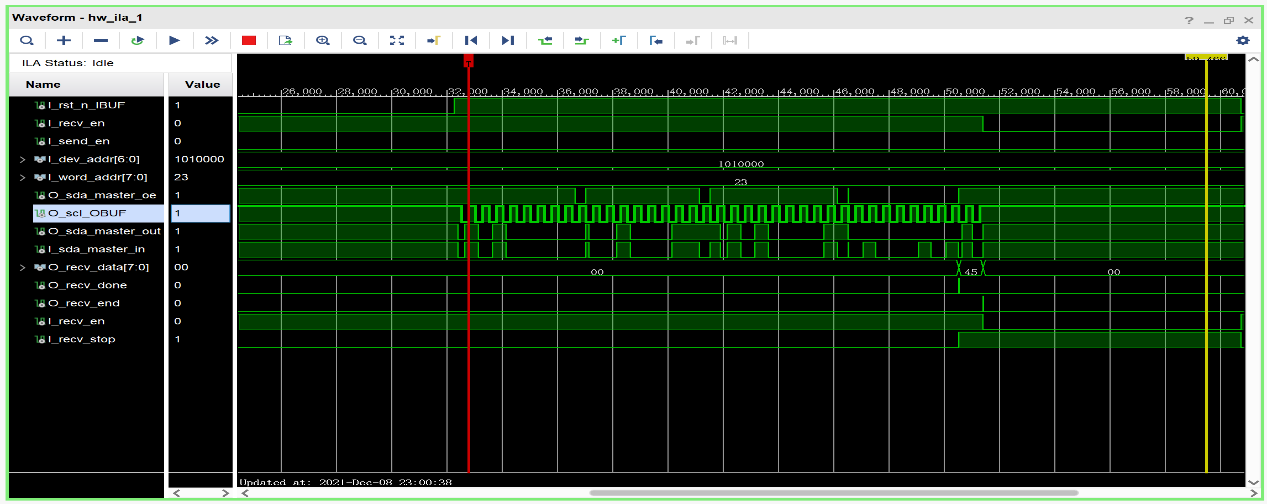


图4.3 iic\_master接口读操作验证波形图

* 1. **从机iic\_slave接口利用主机iic\_master接口进行FPGA验证**

在4.1中已经验证了主机iic\_master接口读、写操作的功能正确性。现在要验证从机iic\_slave接口的功能正确性，可以利用前面已通过验证的iic\_master接口配合验证。

**4.2.1 写操作**

首先验证该iic\_slave接口的写操作的功能正确性。

验证的结构示意图如图4.4所示。这里，FPGA控制I\_write\_data[7:0]端口数据从iic\_master主机接口输入，及从从机接口的O\_write\_data[7:0]端口接收数据。若接收到的数据就是输入的数据，则表明在该主、从机接口的配合下，数据能正确写入。而iic\_master主机接口的写操作功能正确性之前已验证通过，因而可表明所设计的从机iic\_slave接口写操作的功能正确性得到验证。

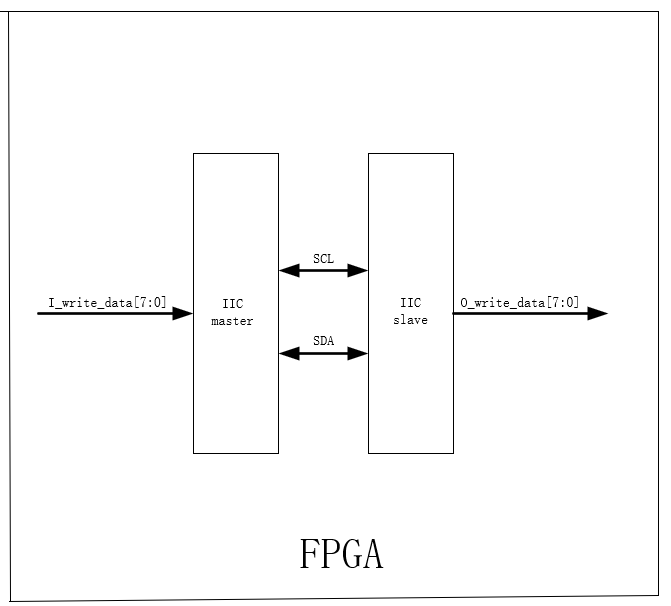


图4.4 iic\_slave接口写操作FPGA验证结构示意图

利用上述验证思路，在FPGA验证中利用ILA捕捉到的iic\_slave接口写操作验证波形图如图4.5和图4.6中所示。可以看到从I\_write\_data[7:0]端口写入的数据0x01~0xf6可以正确在O\_write\_data[7:0]端口接收到，表明iic\_slave接口写操作的功能正确性得到验证。

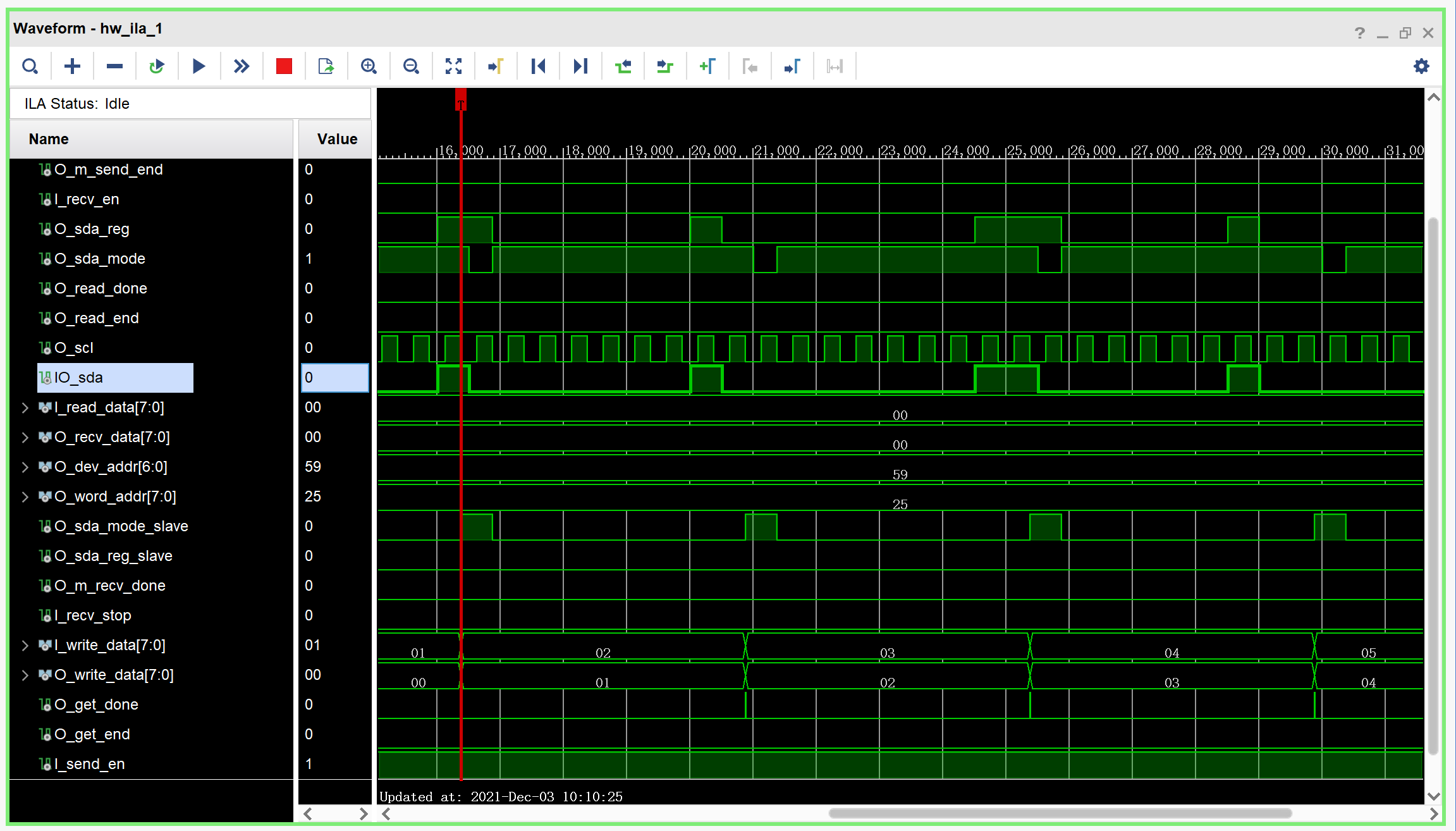


图4.5 iic\_slave接口写操作FPGA验证波形图1

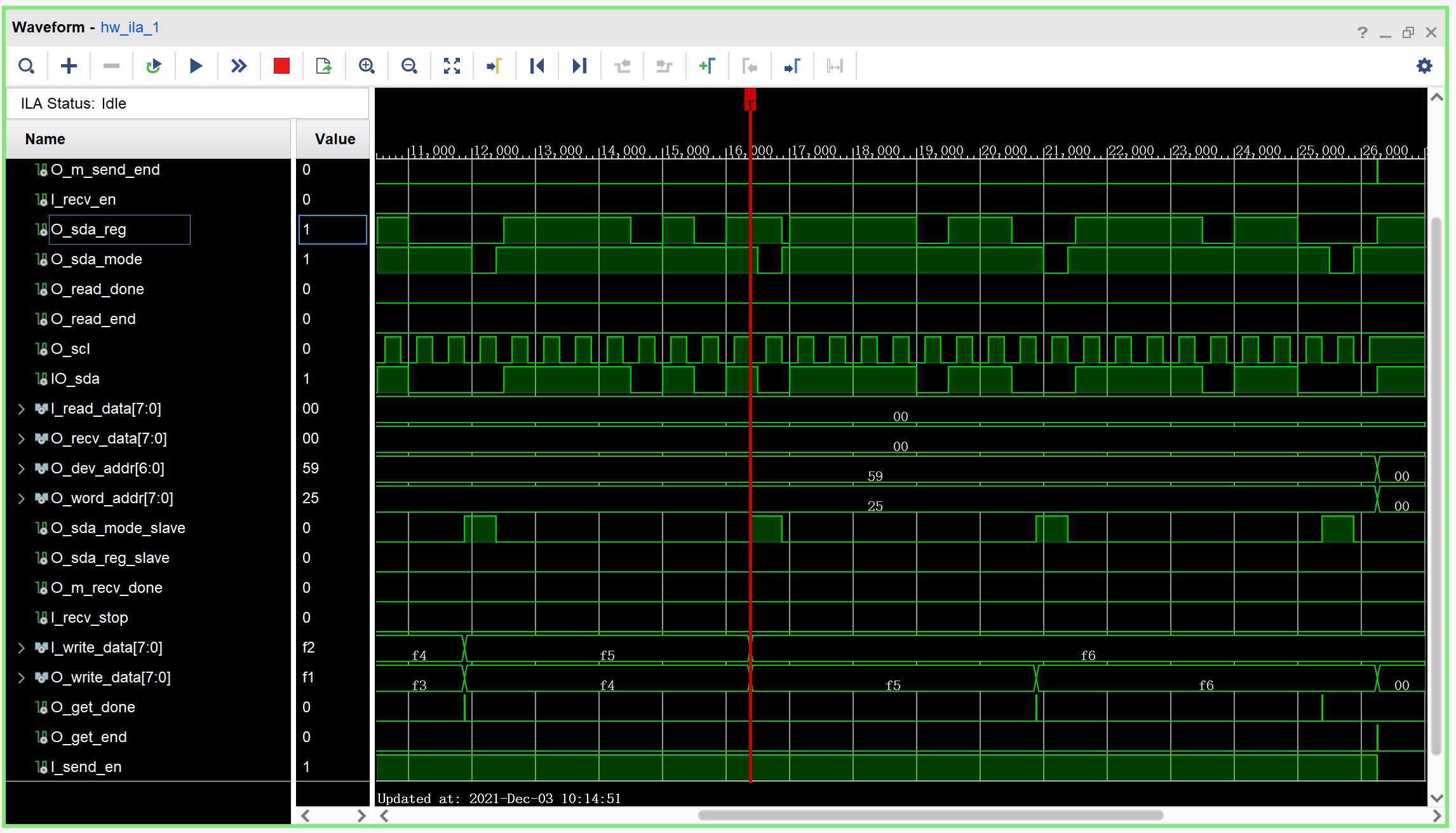


图4.6 iic\_slave接口写操作FPGA验证波形图2

**4.2.2 读操作**

再验证该iic\_slave接口的读操作的功能正确性。

FPGA验证的结构示意图如图4.7所示。这里，FPGA控制I\_read\_data[7:0]端口数据从iic\_slave从机接口输入，及从主机iic\_master接口的O\_recv\_data[7:0]端口接收数据。若接收到的数据就是输入的数据，则表明在该主、从机接口的配合下，数据能正确读出。而iic\_master主机接口的读操作功能正确性之前已验证通过，因而可表明所设计的从机iic\_slave接口读操作的功能正确性得到验证。

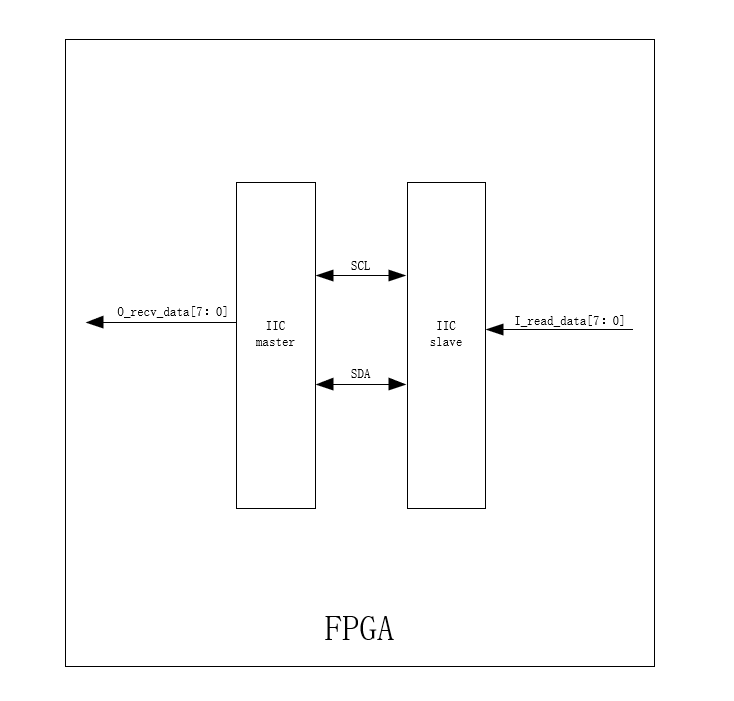


图4.7 iic\_slave接口读操作FPGA验证结构示意图

利用上述验证思路，在FPGA验证中利用ILA捕捉到的iic\_slave接口读操作验证波形图如图4.5和图4.6中所示。可以看到从I\_read\_data[7:0]端口读到的数据0x01~0xf6可以正确在O\_write\_data[7:0]端口输出，表明iic\_slave接口读操作的功能正确性得到验证。

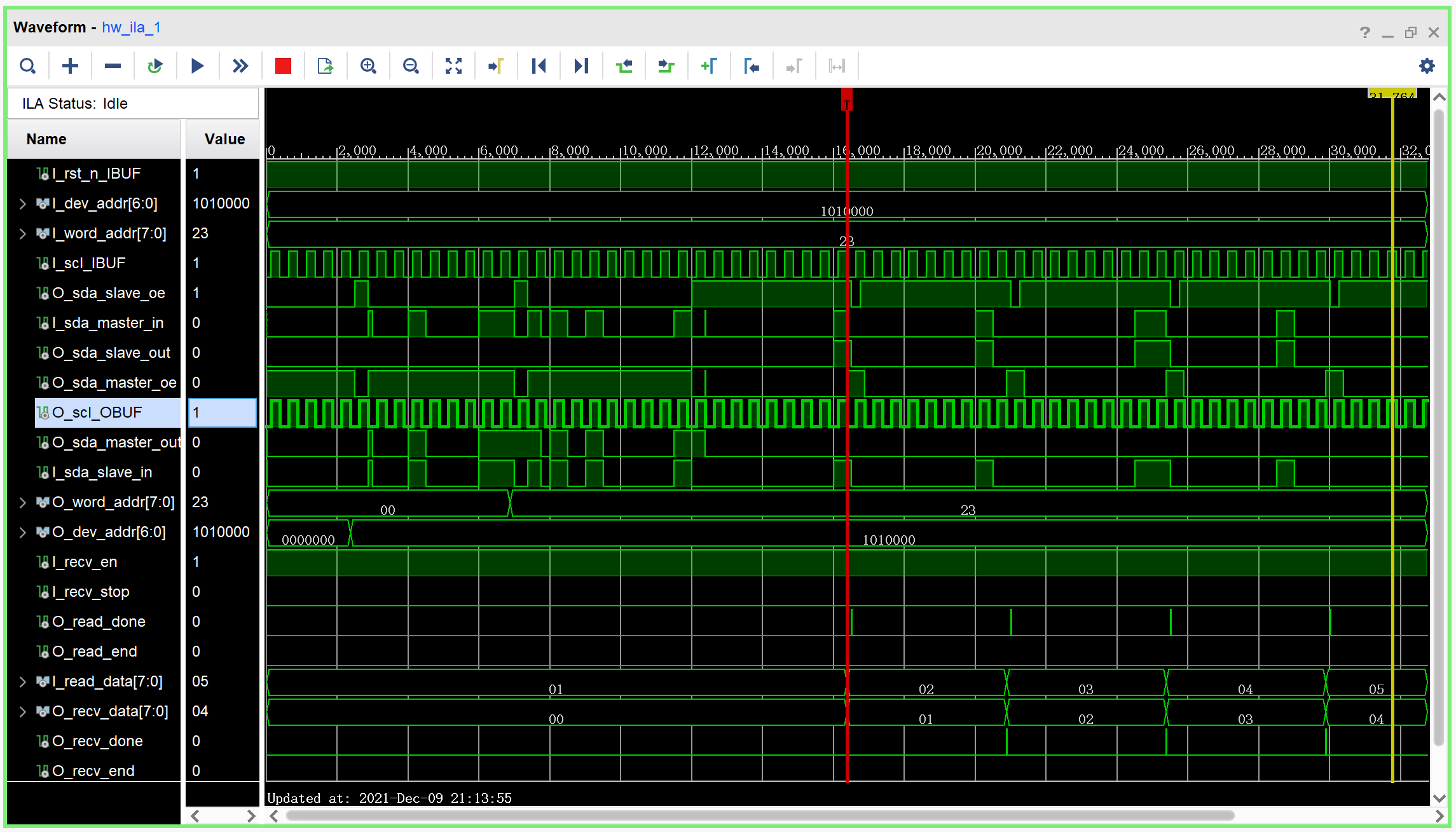


图4.8 iic\_slave接口读操作FPGA验证波形图1

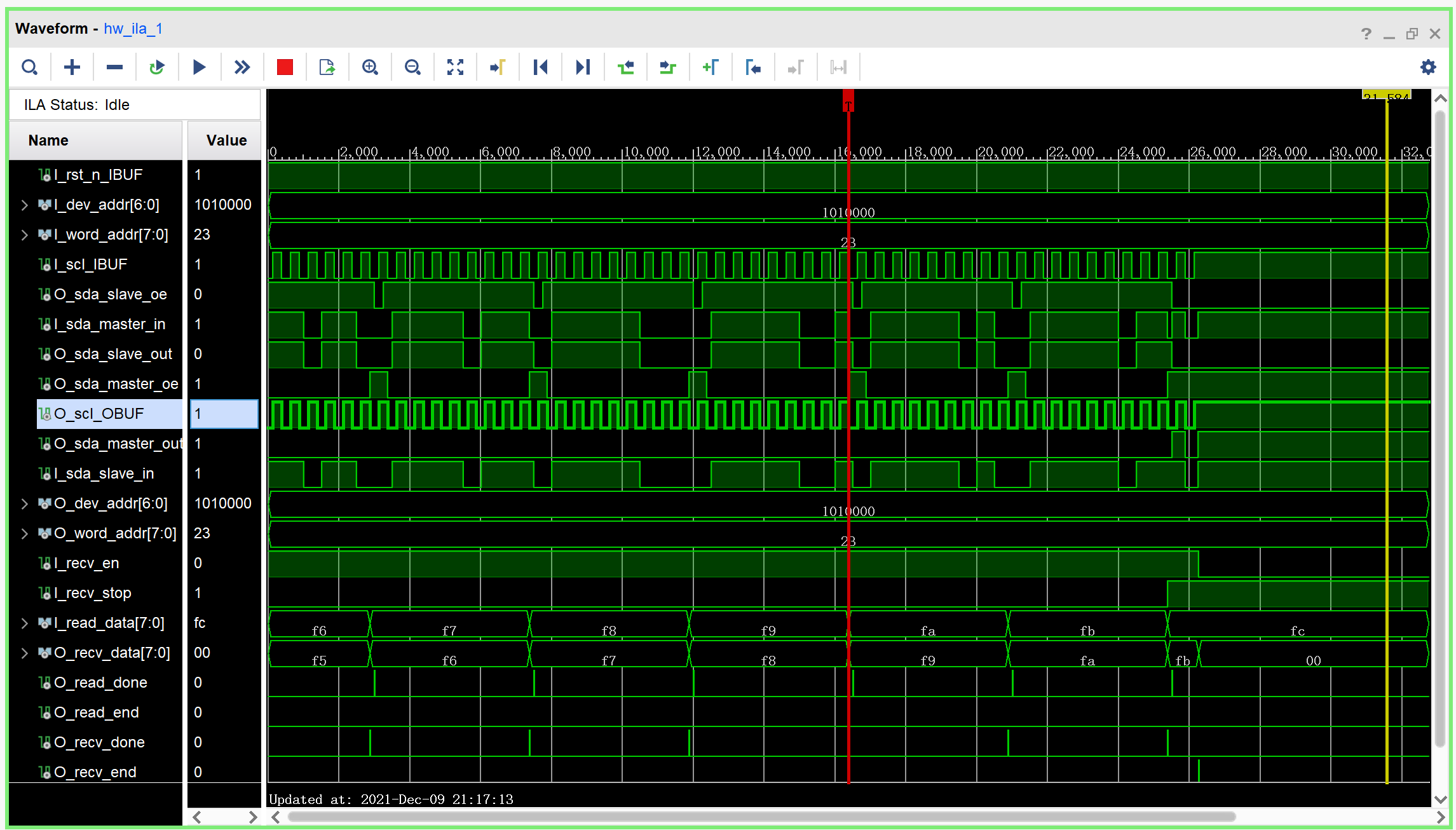


图4.9 iic\_slave接口读操作FPGA验证波形图2

* 1. **从机iic\_slave接口在一主多从（4个从机）模式下工作的FPGA验证**

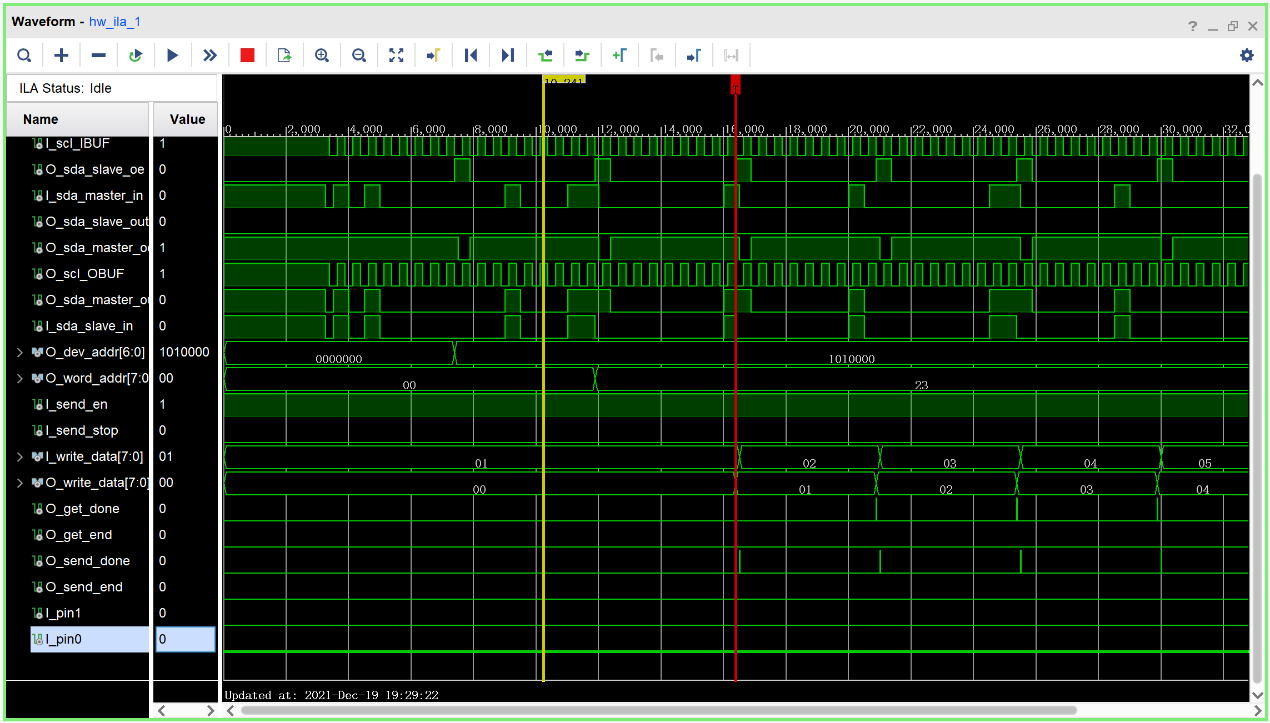
根据iic\_slave接口设计要求，7 位器件地址I\_dev\_addr[6:0]高五位I\_dev\_addr[6:2]设置为固定地址，这里设置为10100，在代码中采用参量形式，可以根据需求改变。7 位器件地址I\_dev\_addr[6:0]低两位I\_dev\_addr[1:0]采用Pin脚输入，代码中对应变量名I\_pin0 、I\_pin1，因而系统可支持搭载四个从机工作。

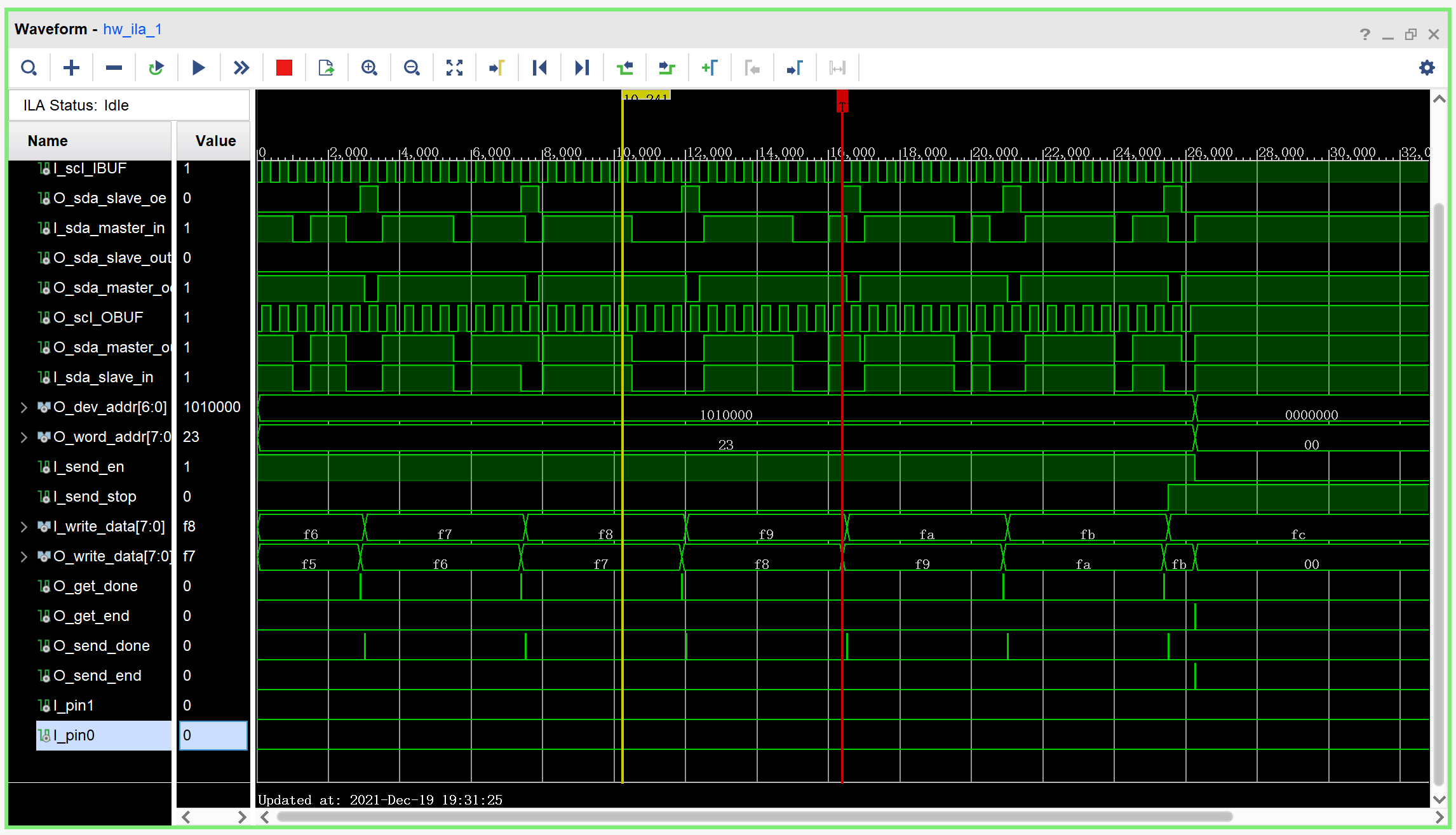
这里对四种Pin脚输入模式{ I\_pin1 ,I\_pin0}={0，0}、{ I\_pin1 ,I\_pin0}={0，1}、{ I\_pin1 ,I\_pin0}={1，0}、{ I\_pin1 ,I\_pin0}={1，1}分别作读、写操作的FPGA验证，并利用ILA进行波形捕捉。

**4.3.1 写操作**

1. Pin脚输入模式{ I\_pin1 ,I\_pin0}={0，0}

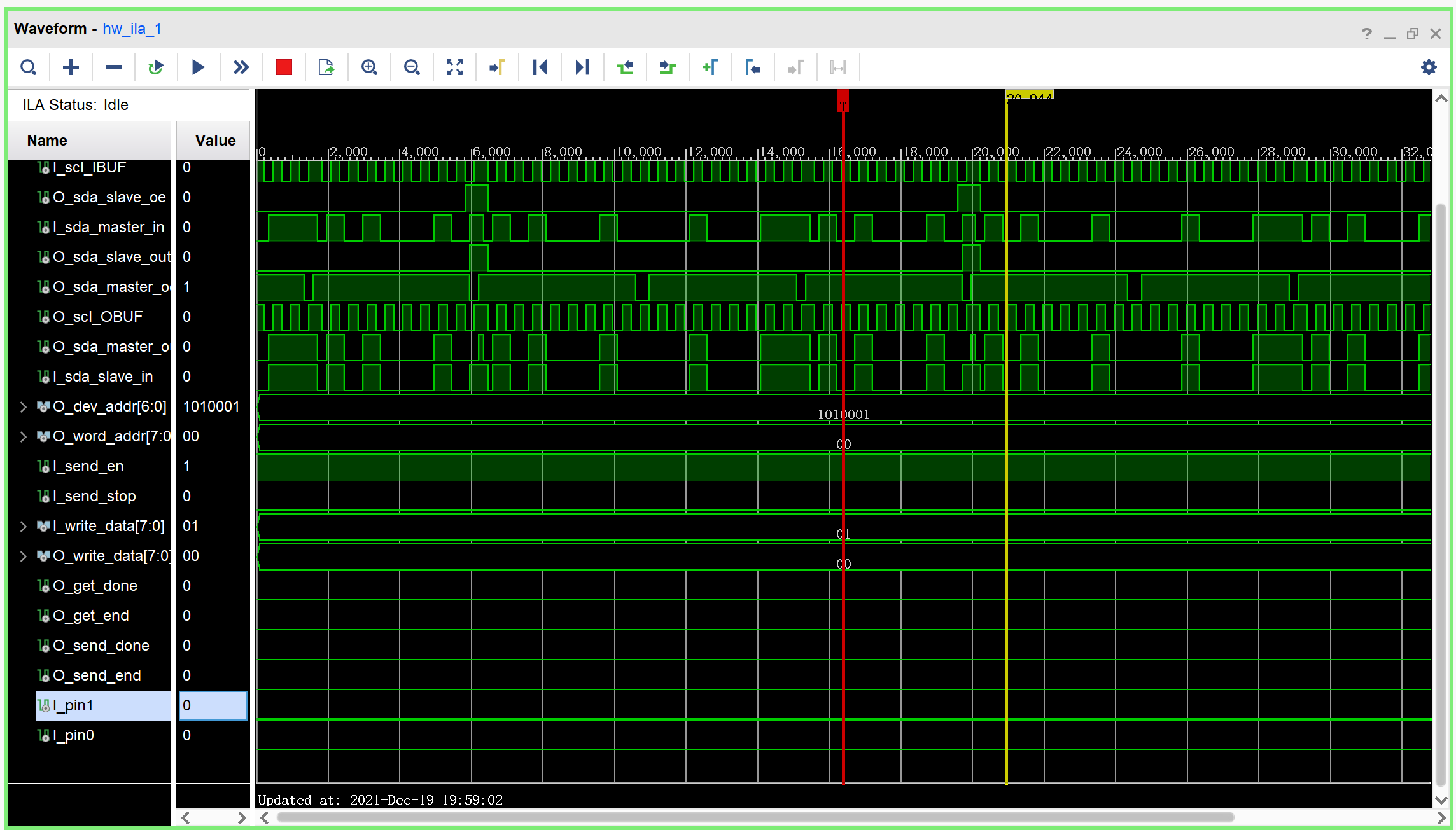
验证波形图如下图所示，可见当输入的7为器件地址I\_dev\_addr[6:0]为1010000时，此时寻址的器件地址正好与该从机接口的{ I\_pin1 ,I\_pin0}={0，0}相对应。





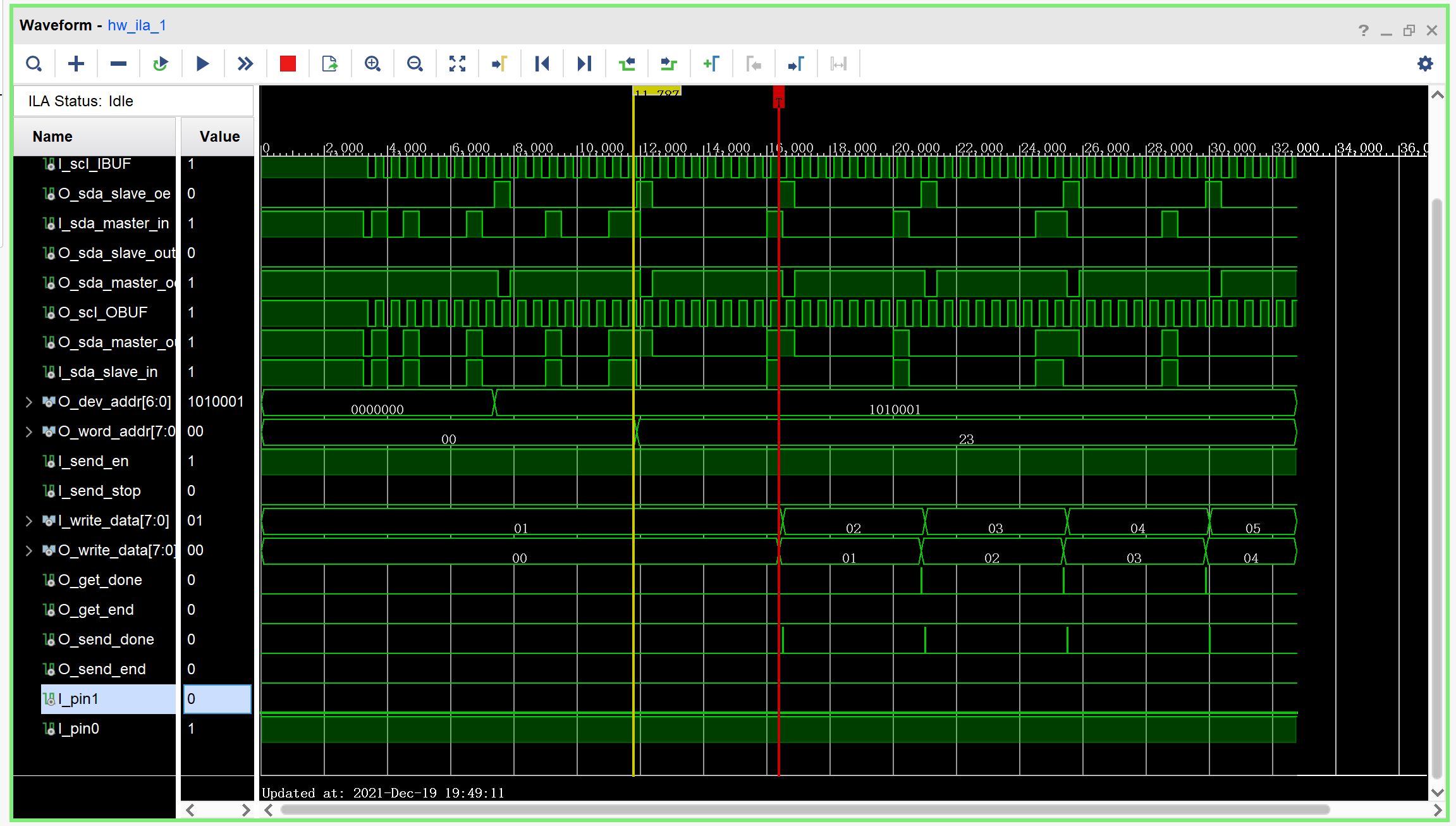
特别的，当主机想要建立通信的从机不是本从机，也即主机寻址的器件地址不是从机自己的器件地址的情况，这里以当该从机接口的{ I\_pin1 ,I\_pin0}={0，0}，接口寻址的7位地址为1010001为例进行验证，其他的情况与此类。

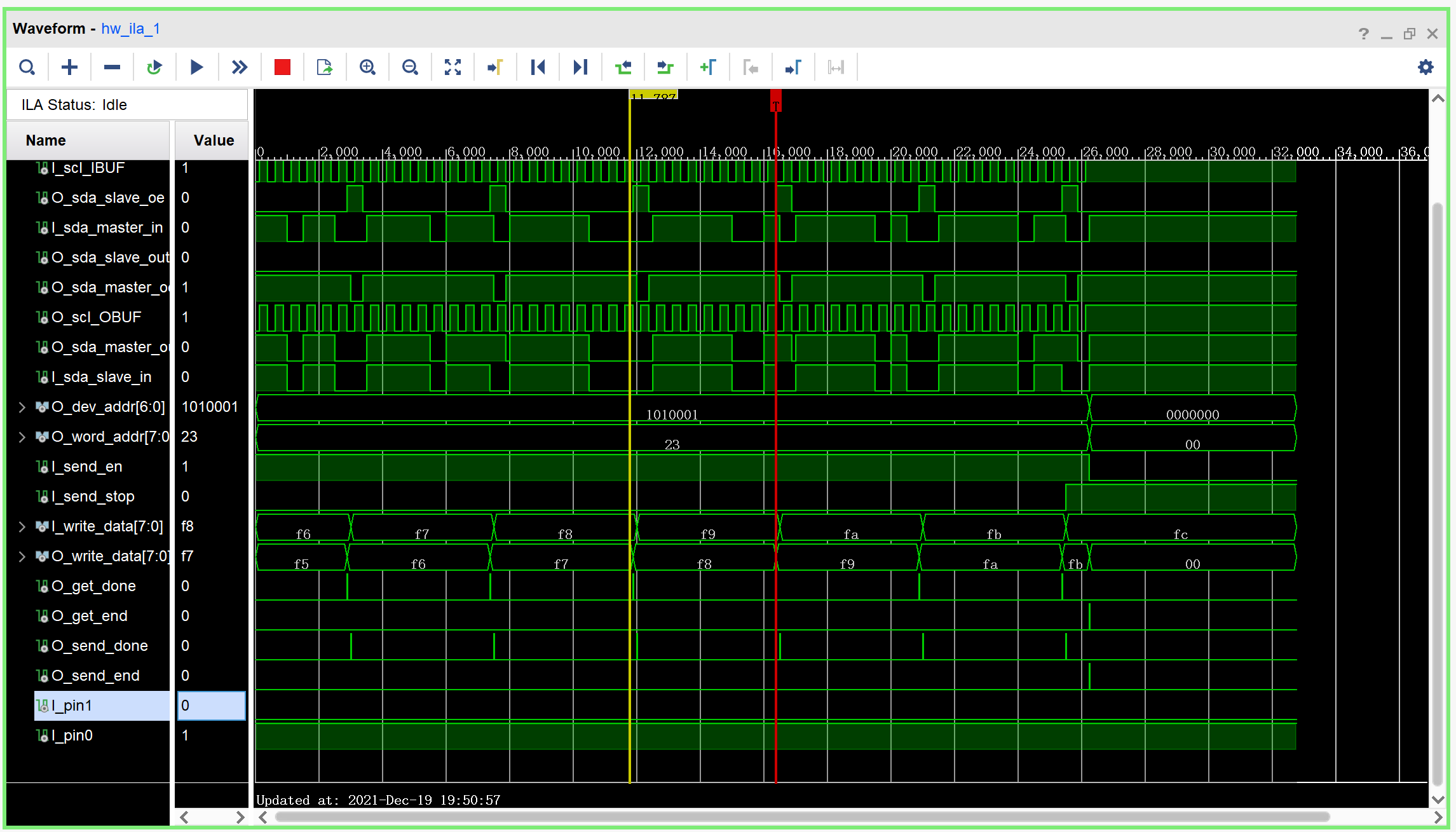
可知此时的从机地址I\_dev\_addr[6:0]为1010000，而若此时的接口寻址的7位地址为1010001时，即可知此时主机寻址的从机器件地址与本从机接口的器件地址不对应，也即主机本次想要进行数据通信的从机不是本从机。因而此时的从机接收到IIC信号线上的地址进行核对后，发现主机想要通信的不是自己，故而在接受到该器件地址之后会发送NOACK信号后，会进入IDLE状态继续等待下一次IIC信号线上的起始信号的到来，故而不会接受到本次传送的后面的数据信号。FPGA验证波形图入下图所示，O\_write\_data[7:0]始终接收不到I\_ write\_data[7:0]的数据，同时也可从从机的O\_get\_done、O\_get\_end一直为低电平，没有产生字节数据接收完成标志1的情况分析出来。



1. Pin脚输入模式{ I\_pin1 ,I\_pin0}={0，1}

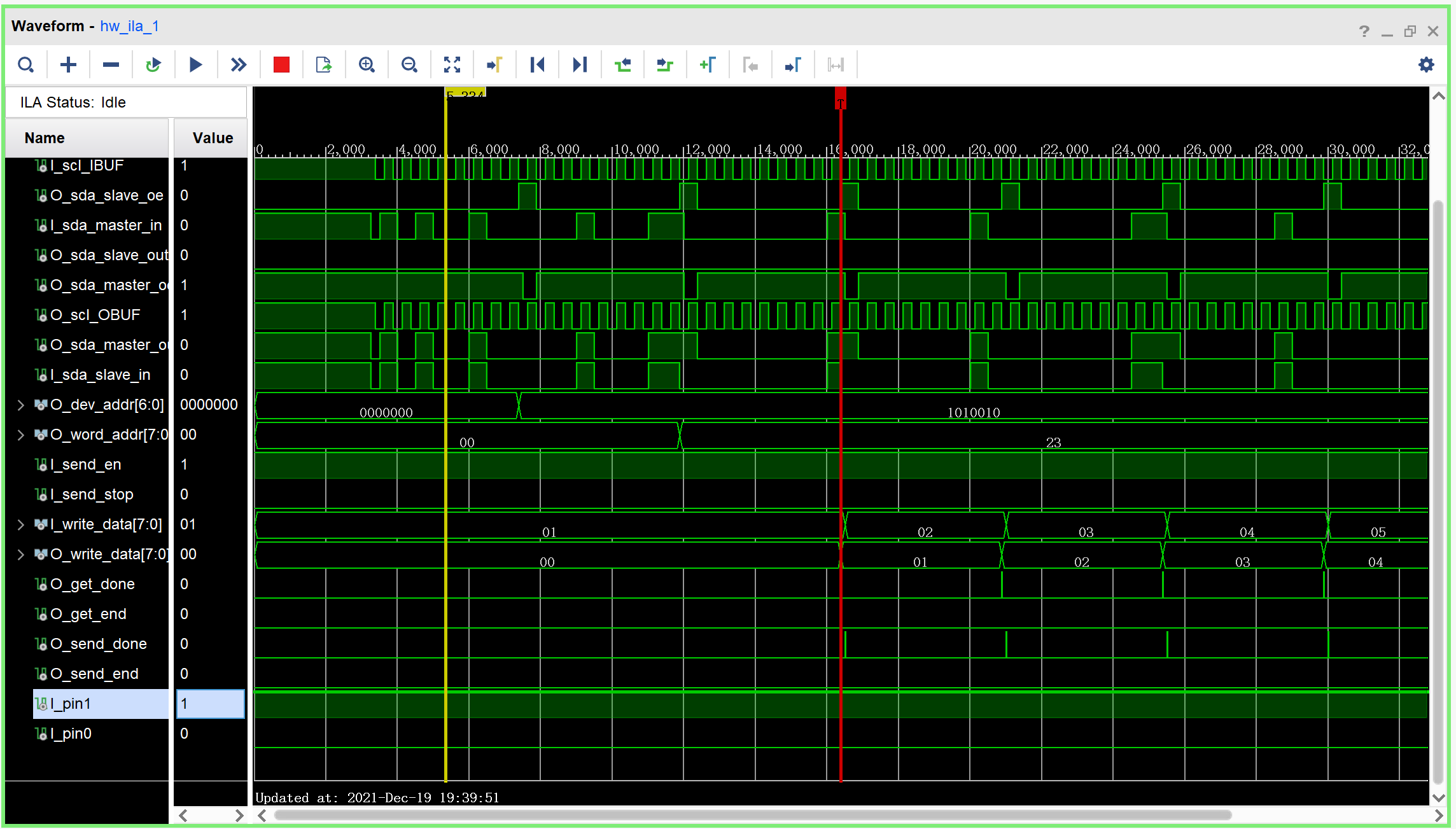
验证波形图如下图所示，可见当输入的7为器件地址I\_dev\_addr[6:0]为1010001时，此时寻址的器件地址正好与该从机接口的{ I\_pin1 ,I\_pin0}={0，1}相对应。

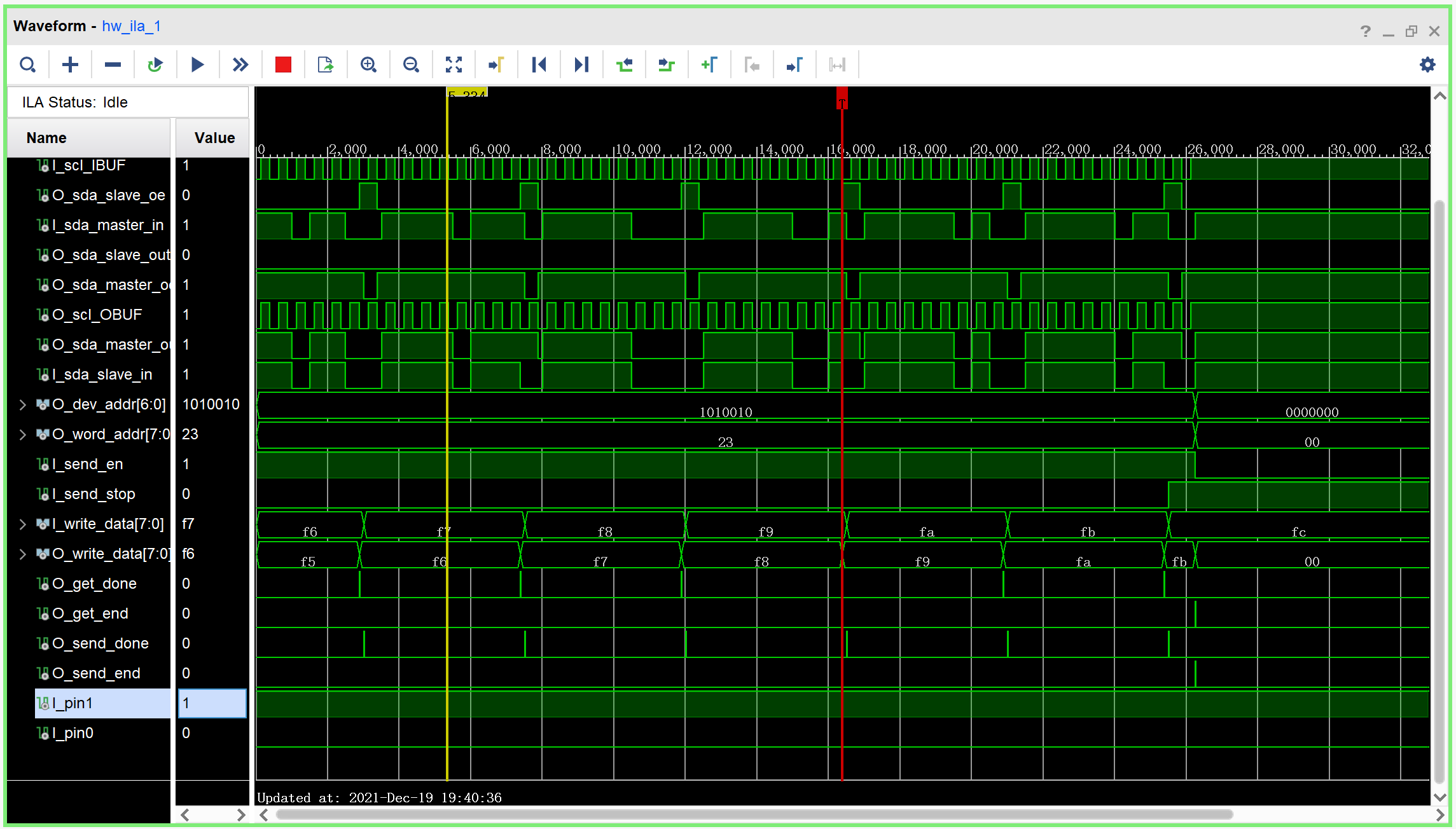




1. Pin脚输入模式{ I\_pin1 ,I\_pin0}={1，0}

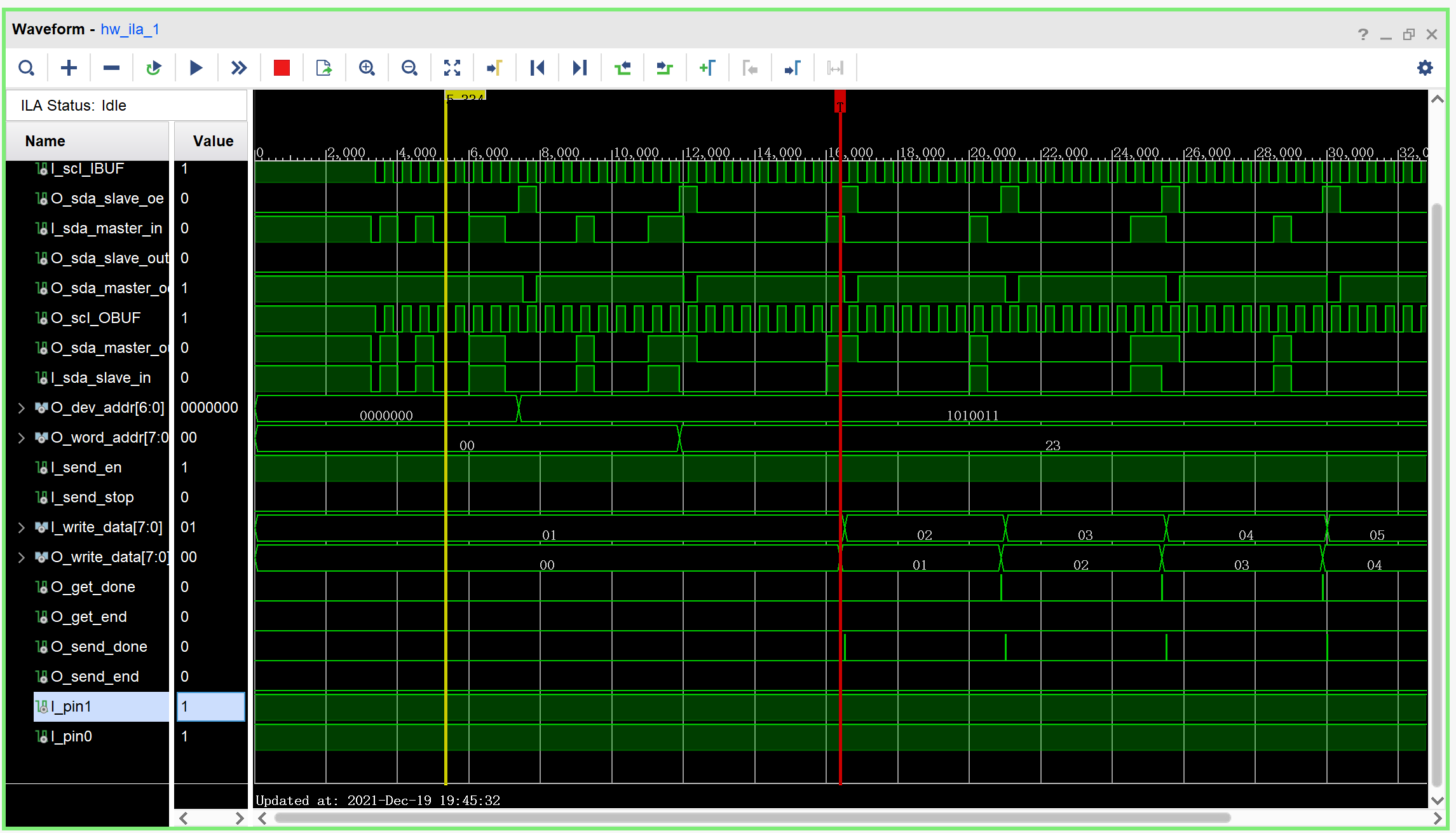
验证波形图如下图所示，可见当输入的7为器件地址I\_dev\_addr[6:0]为1010010时，此时寻址的器件地址正好与该从机接口的{ I\_pin1 ,I\_pin0}={1，0}相对应。

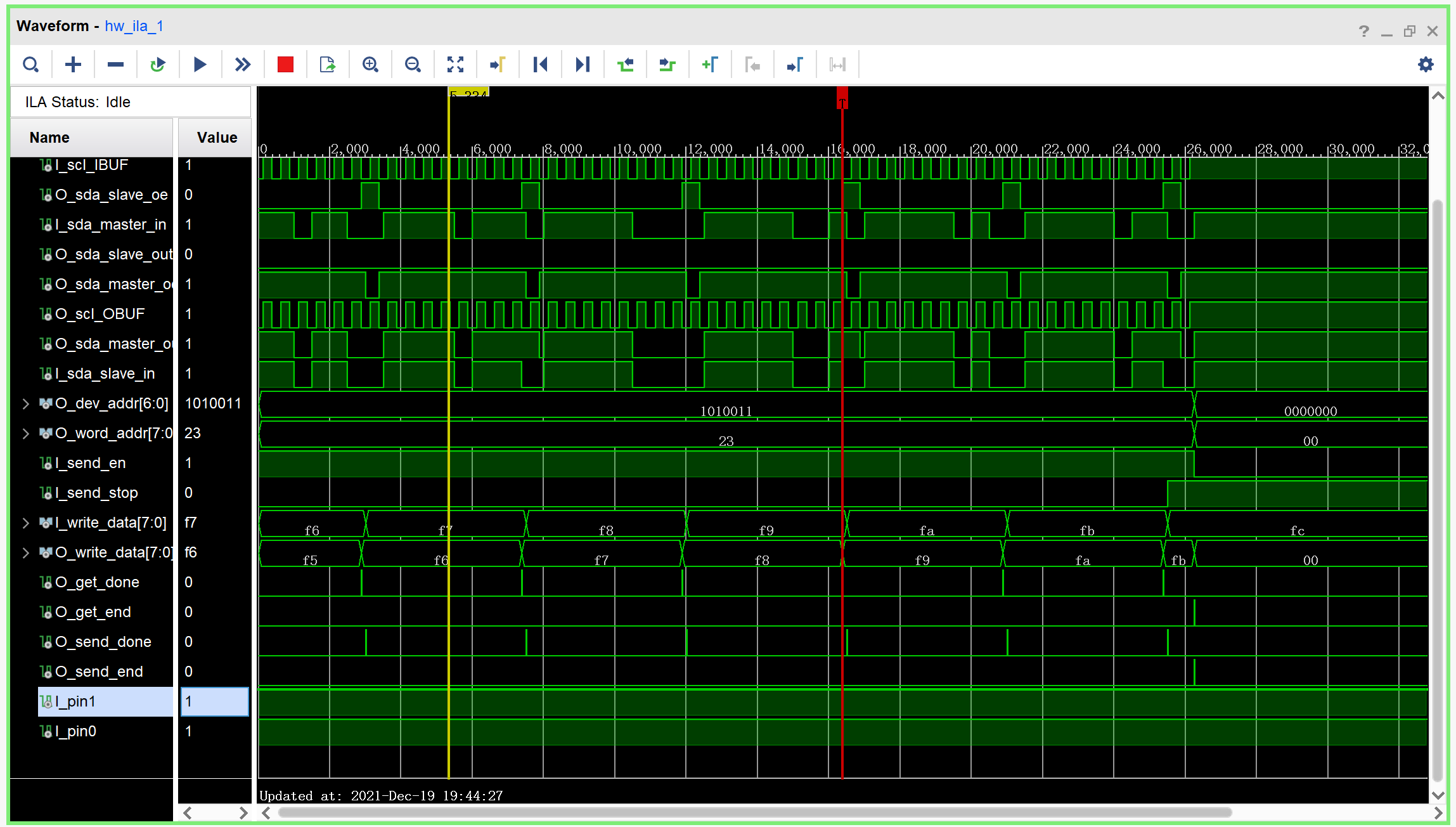




1. Pin脚输入模式{ I\_pin1 ,I\_pin0}={1，1}

验证波形图如下图所示，可见当输入的7为器件地址I\_dev\_addr[6:0]为1010011时，此时寻址的器件地址正好与该从机接口的{ I\_pin1 ,I\_pin0}={1，1}相对应。

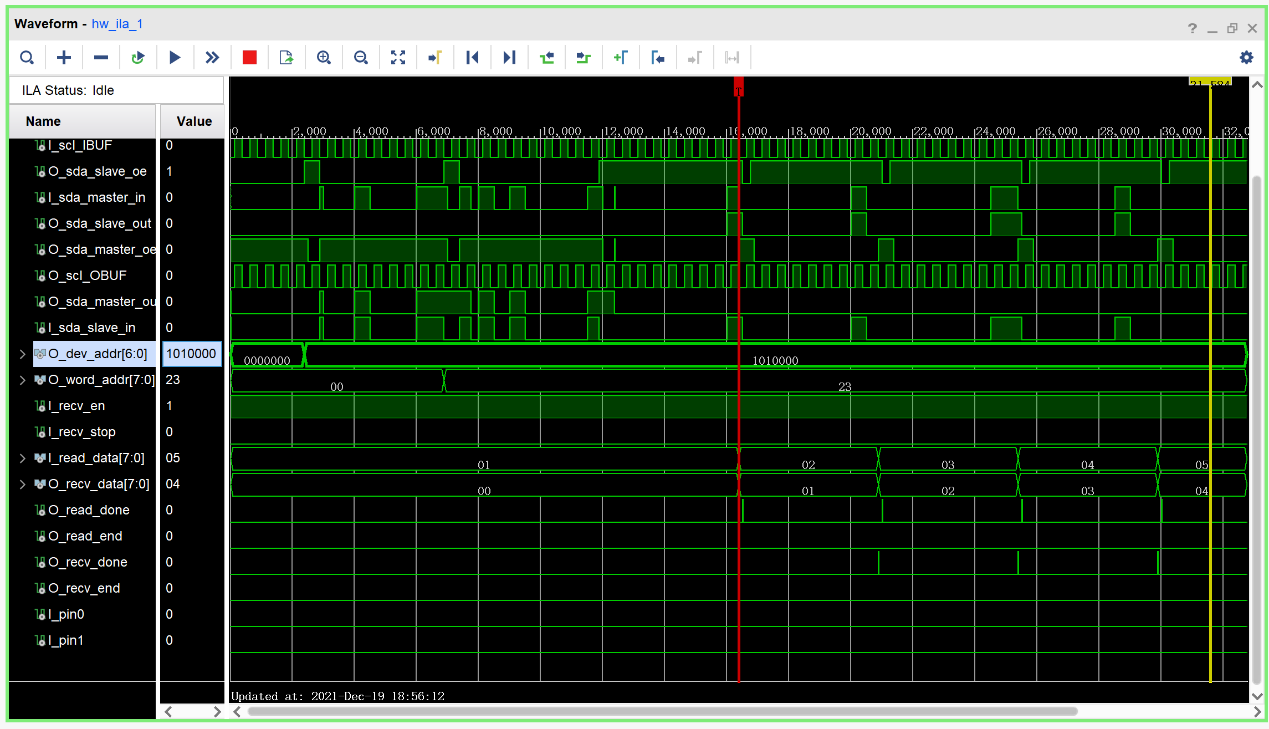


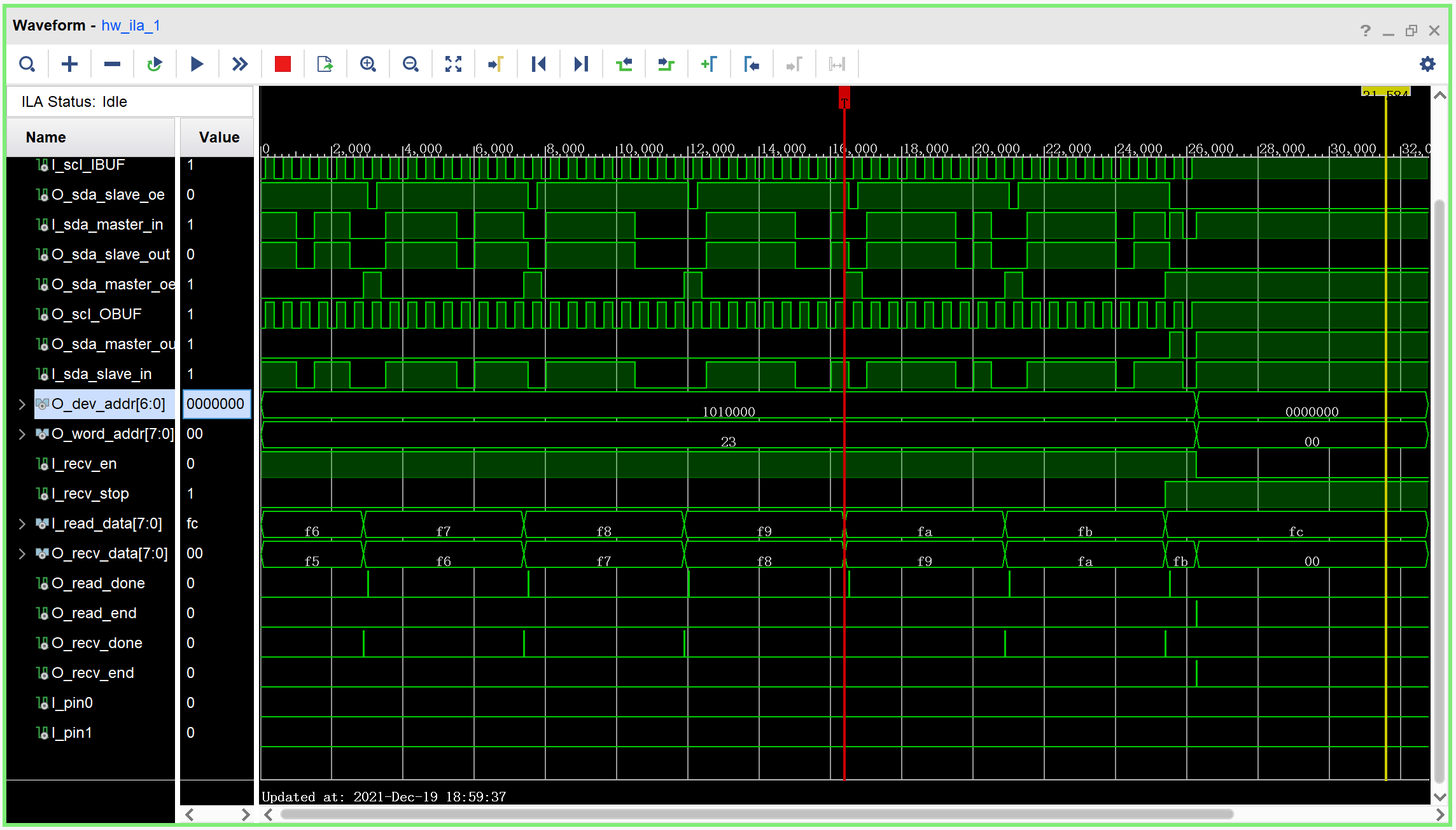


**4.3.2 读操作**

1. Pin脚输入模式{ I\_pin1 ,I\_pin0}={0，0}

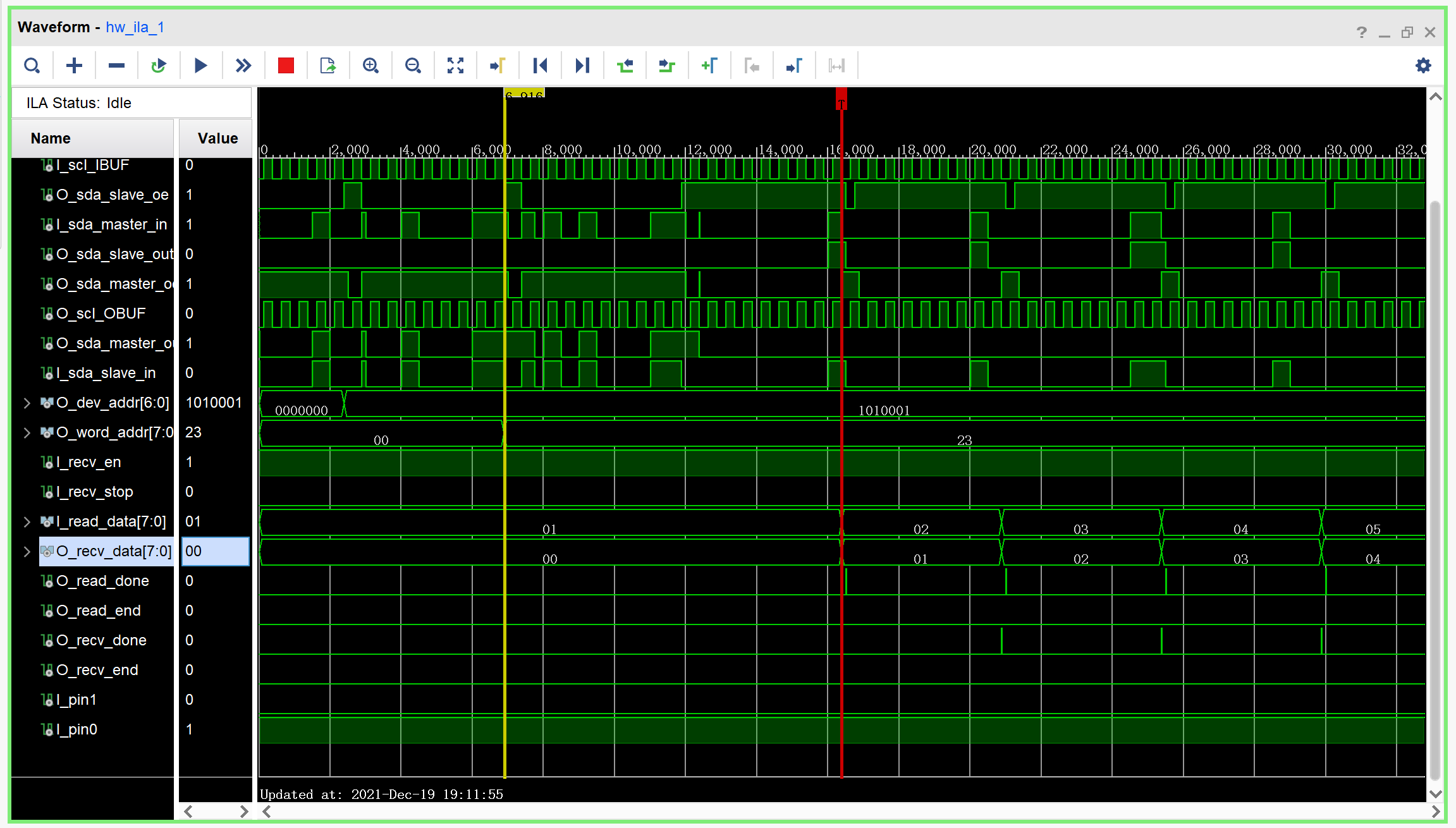
验证波形图如下图所示，可见当输入的7为器件地址I\_dev\_addr[6:0]为1010000时，此时寻址的器件地址正好与该从机接口的{ I\_pin1 ,I\_pin0}={0，0}相对应。

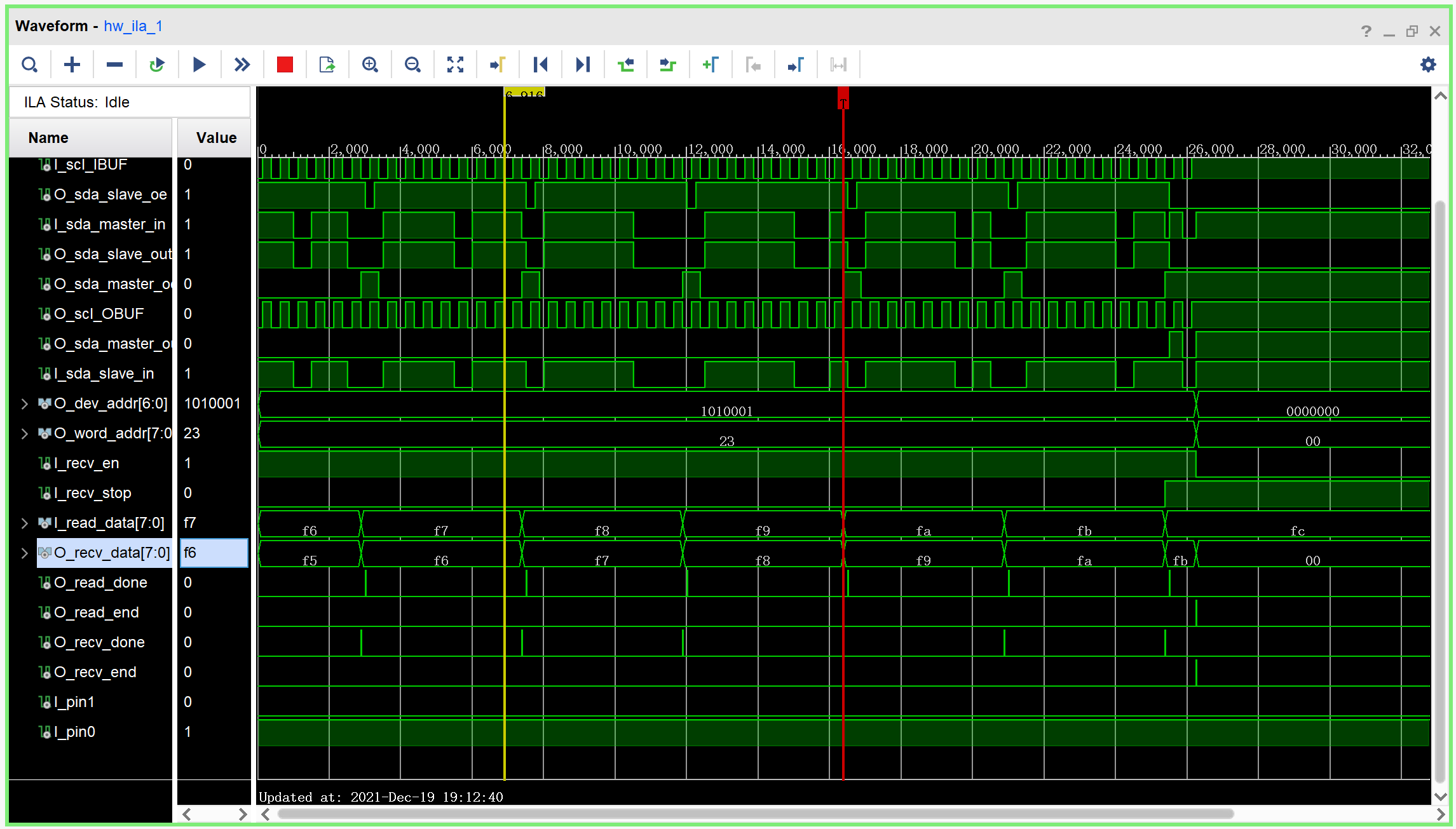




2、Pin脚输入模式{ I\_pin1 ,I\_pin0}={0，1}

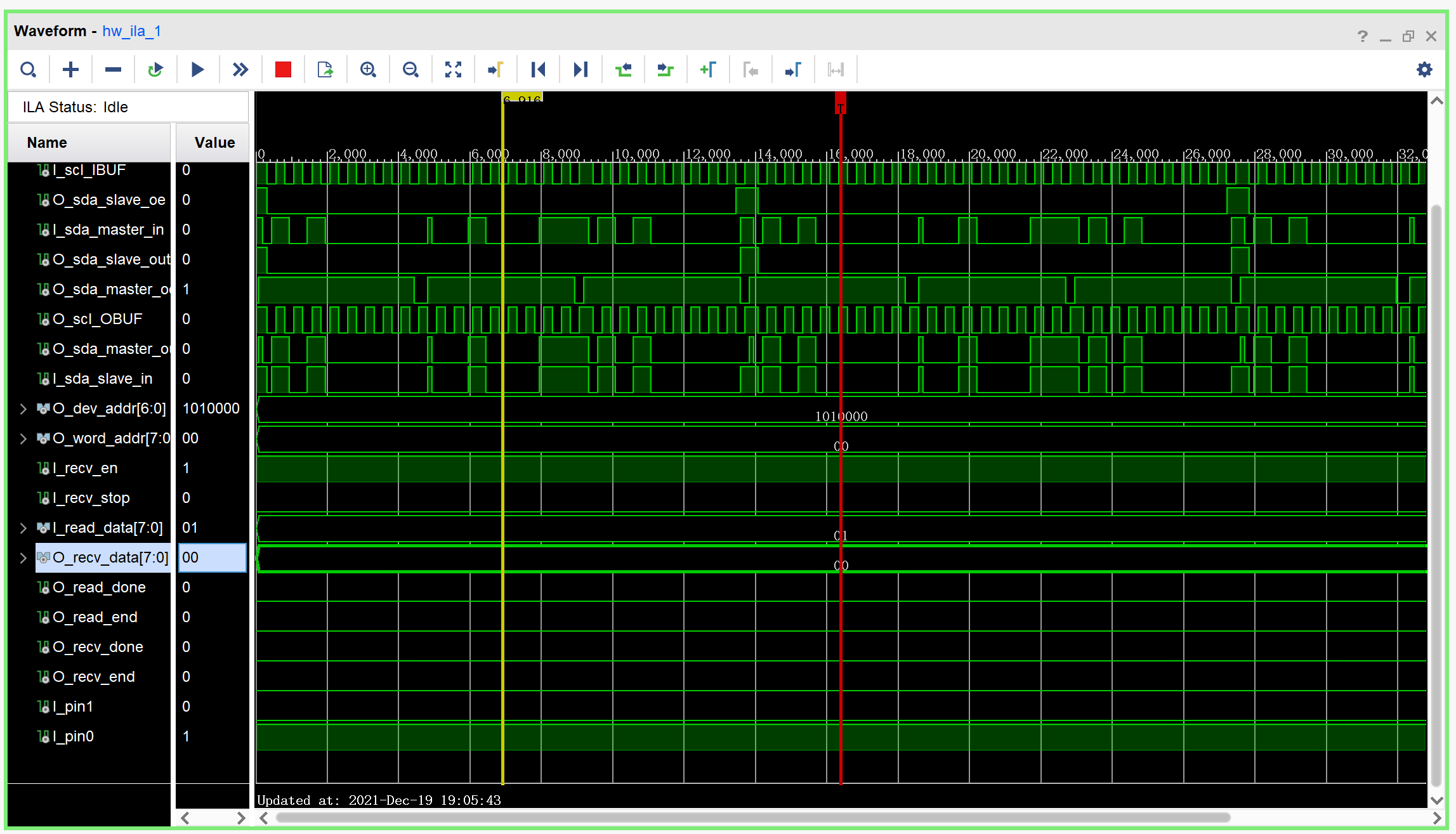
验证波形图如下图所示，可见当输入的7为器件地址I\_dev\_addr[6:0]为1010000时，此时寻址的器件地址正好与该从机接口的{ I\_pin1 ,I\_pin0}={0，0}相对应。





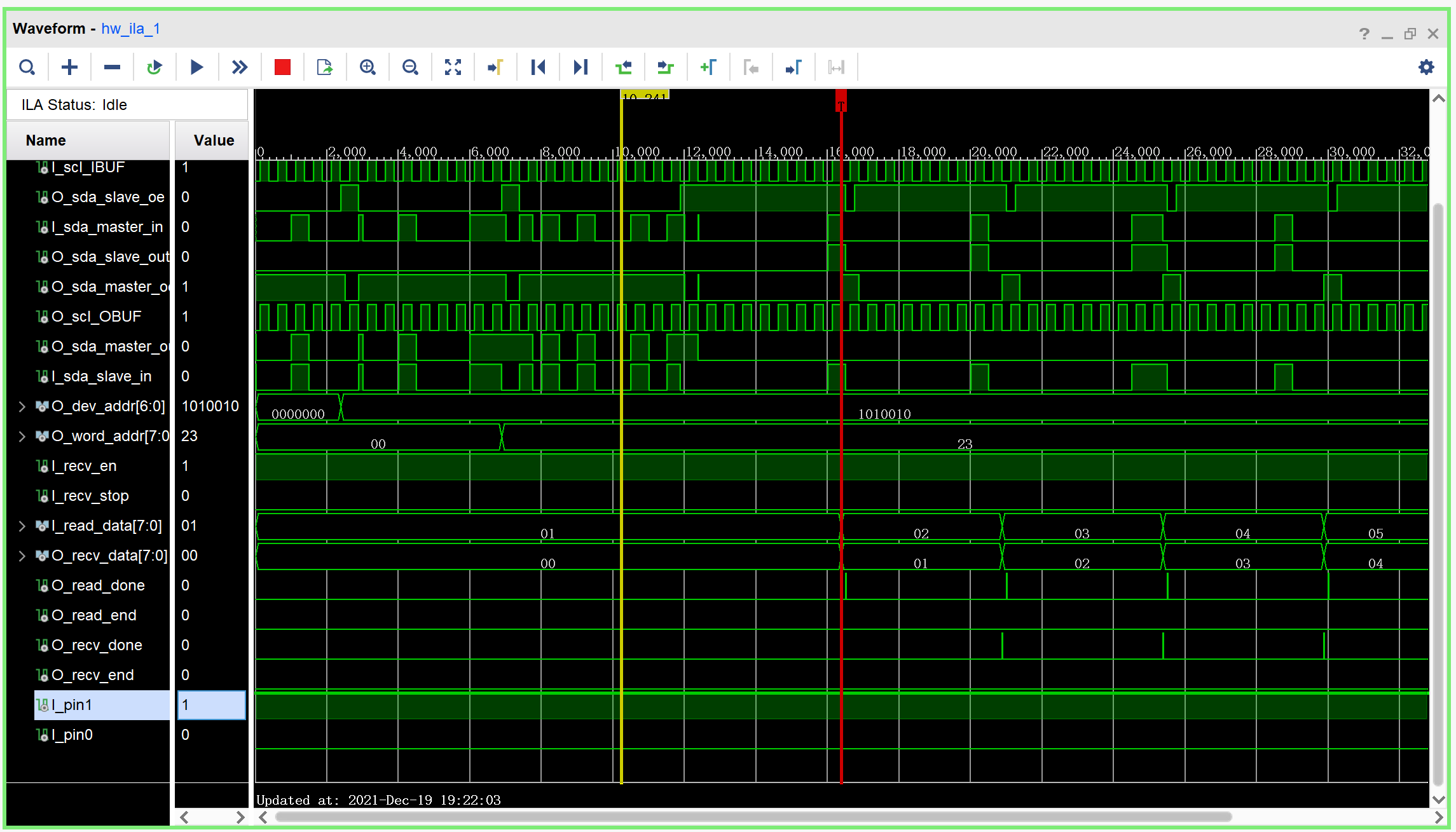
特别的，当主机想要建立通信的从机不是本从机，也即主机寻址的器件地址不是从机自己的器件地址的情况。这里以当该从机接口的{ I\_pin1 ,I\_pin0}={0，1}，接口寻址的7位地址为1010000为例进行验证，其他的情况与此类。

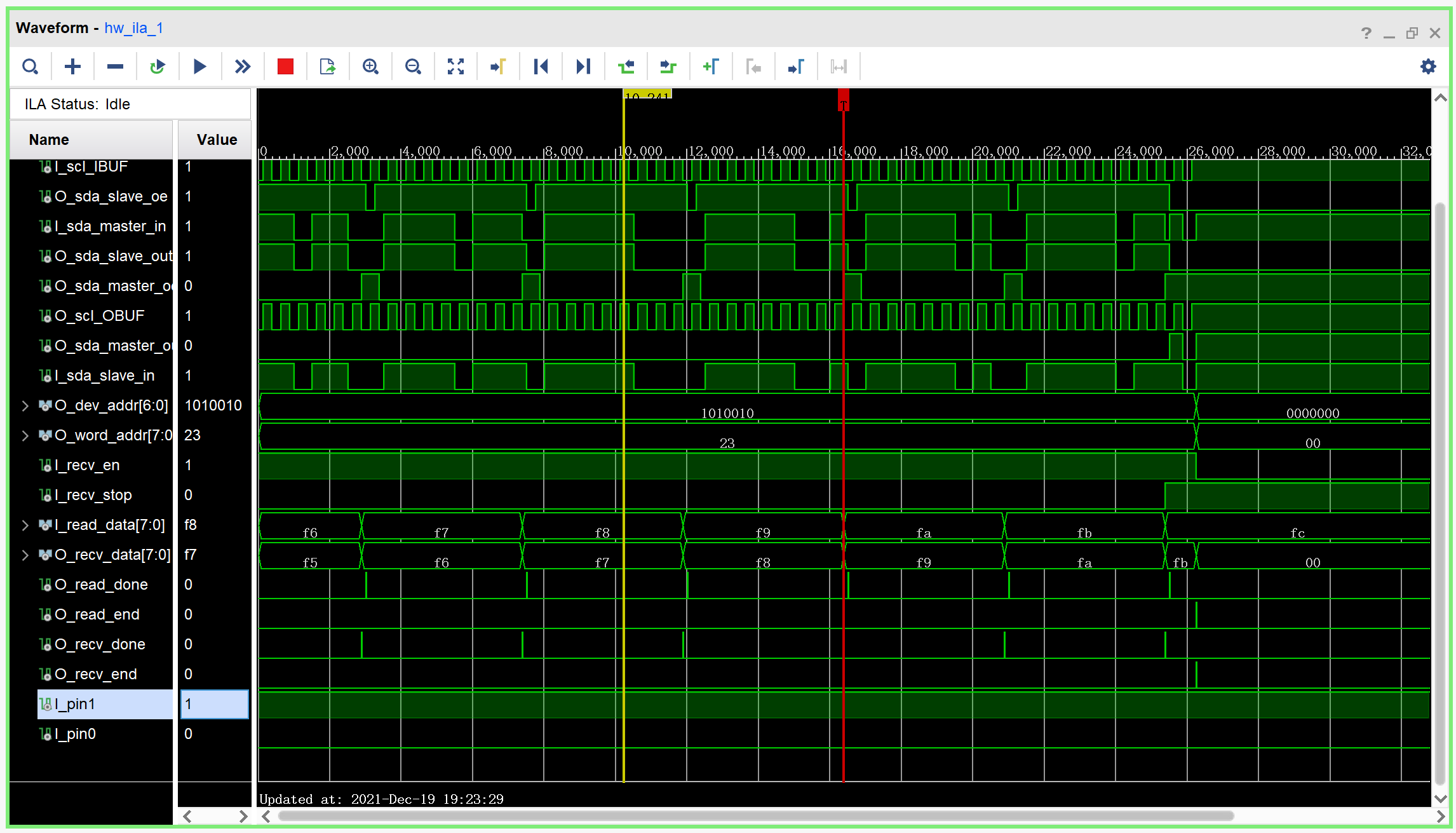
可知此时的从机地址I\_dev\_addr[6:0]为1010001，而若此时的接口寻址的7位地址为1010000时，即可知此时主机寻址的从机器件地址与本从机接口的器件地址不对应，也即主机本次想要进行数据通信的从机不是本从机。因而此时的从机接收到IIC信号线上的地址进行核对后，发现主机想要通信的不是自己，故而在接受到该器件地址之后会发送NOACK信号后，会进入IDLE状态继续等待下一次IIC信号线上的起始信号的到来，故而不会向接口中传送本次想要读取的后面的数据信号。FPGA验证波形图入下图所示，O\_recv\_data[7:0]始终读取不到I\_read\_data[7:0]的数据，同时也可从从机的O\_read\_done、O\_read\_end一直为低电平，没有产生字节数据读取完成标志1（高电平）的情况分析出来。



1. Pin脚输入模式{ I\_pin1 ,I\_pin0}={1，0}

验证波形图如下图所示，可见当输入的7为器件地址I\_dev\_addr[6:0]为1010010时，此时寻址的器件地址正好与该从机接口的{ I\_pin1 ,I\_pin0}={1，0}相对应。





4、Pin脚输入模式{ I\_pin1 ,I\_pin0}={1，1}

验证波形图如下图所示，可见当输入的7为器件地址I\_dev\_addr[6:0]为1010011时，此时寻址的器件地址正好与该从机接口的{ I\_pin1 ,I\_pin0}={1，1}相对应。

