

《综合创新-数字通信》

# 基于 EELAB 的 ASK 传输系统

17241072 孙启楚

18373543 尹航

18373533 陈明澍

18373429 张驰宇

18373038 钱思远

电子信息工程学院

2020 年 12 月 10 日

# 目 录

1	作品简介.....	3
1.1	作品组成及系统外观（接口）说明.....	4
1.2	作品功能.....	4
1.3	实物作品操作使用方法.....	4
2	制作概况.....	6
3	设计总结.....	7
3.1	方案设计.....	7
3.1.1	ASK 调制解调仿真.....	8
3.1.2	PSK 调制解调仿真.....	10
3.1.3	FSK 调制解调仿真.....	13
3.2	硬件设计.....	17
3.2.1	EELAB-FPGACORE2 硬件实验平台.....	17
3.2.2	DAC/ADC.....	3
3.3	软件设计.....	5
3.3.1	程序流程.....	5
3.3.2	各程序模块详细介绍.....	5
3.4	其他设计.....	10
3.4.1	码元宽度误差和抖动.....	10
3.4.2	帧头.....	10
4	调试过程.....	11
5	测试结果.....	12
6	遇到的问题与解决方法.....	16
6.1	软件设计过程.....	16
6.2	制作过程.....	16
6.3	调试过程.....	16
7	结论.....	16
	参考文献.....	17
	附录.....	17

# 1 作品简介

本小组的实物作品基于 EELAB 开发板进行制作，使用 Verilog HDL 语言进行编程并通过 ISE 进行综合、烧录。最终的作品实现了信号的 AD 转换、DA 转换以及完整的调制与解调。最终的作品实物如下图所示：

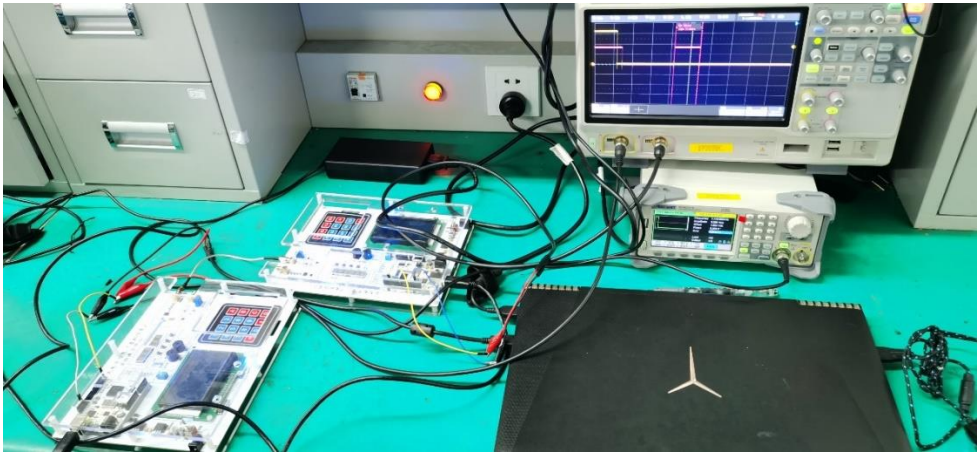


图1 实物作品实物

本作品硬件部分采用实验教学中心提供的 EELAB-DIGIEXP，一款具有多种电路模块的扩展开发板，用于为小型化设计的处理器核心板提供更多的扩展功能。该开发板采用专用扩展接口与处理器核心板连接，与位于 EELAB-FPGACORE2 背面的扩展接口兼容。开发板上具有电子类课程实验及科技创新开发所需的基本电路模块，包括数码管、按键开关、LED、蜂鸣器、矩阵键盘、液晶屏、ADC、DAC 等，通过与 EELAB-FPGACORE2 等处理器核心板相配合，能够实现显示、控制、信号发生和采集等功能。

EELAB-DIGIEXP 板可支持的数字电路实验包括流水灯实验、计数器实验、AD 转换实验、DA 转换实验；可支持的通信原理实验包括 PCM 编译码器系统实验、帧成形及其传输实验、AMI/HDB3 码型变换实验、ASK 传输系统实验、FSK 传输系统实验、PSK 传输系统实验；可支持的数字信号处理实验包括 FIR 数字滤波器设计实验、IIR 数字滤波器设计实验、FFT 信号谱分析实验。

硬件仿真及实物制作过程中小组成员分工如下表所示。

表1 组员信息与分工

学号	姓名	负责内容	备注
17241072	孙启楚	代码编写、系统调试	组长
18373533	陈明澍	硬件仿真、系统调试	
18373543	尹航	硬件仿真、系统调试	
18373429	张驰宇	硬件仿真、报告撰写	
18373038	钱思远	报告撰写、PPT 制作	

### 1.1 作品组成及系统外观（接口）说明

最终的作品包括两个 EELAB 开发板、连接线及杜邦线若干。

其中开发板 1（图 1 左）用来调制基带信号。开发板 1 中使用到的端口包括 FPGA 上的 IO 口 D0、GND 接口以及 DA\_OUT 模拟输出端口。各个端口功能及电气说明如下：IO 口 D0 作为输入端口输入波形发生器产生的方波信号（基带信号），该接口使用 3.3V 标准的 LVTTTL 电平，在作为输入接口使用时要注意使用高电平为 3.3V，低电平为 0V 的方波信号；GND 端口为整个系统提供一个基准电压；DA\_OUT 端口则用于输出经过 DA 转换之后的模拟调制信号。

开发板 2（图 1 右）用来解调调制信号。开发板 2 使用到的端口包括 FPGA 上的 IO 口 D3、GND 以及 AD\_IN 模拟输入端口。D3 作为输出端口输出解调之后的信号，其端口电气特性与作为输入端口的 D0 相同；GND 端口为整个系统提供一个基准电压；AD\_IN 模拟输入端口用于输入模拟调制信号。

### 1.2 作品功能

本实物作品可以实现对基带信号（波形发生器产生的方波信号）完整的 ASK 调制与解调。基带信号由开发板的 D0 口输入，使用 ASK 进行调制（载波频率 200kHz），调制之后的信号经过 DA 模块转换为模拟信号。该模拟信号输入开发板 2，经过解调之后又 D3 口输出。另外，我们设计了相应的帧格式(宽度小于最小码元宽度的 10 码)，能够在理论上对有效数据进行区分。下表列出实物作品调制解调过程中的关键指标。

表2 实物作品调制解调过程的关键指标

系统参数指标	设计值	要求数值
基带码元宽度	90 μ s	不大于 1ms
步进	90 μ s	不大于 0.1ms
有效比特传输速率	10.44 kbps	大于等于 10 kbps
载波频率	200kHz	大于等于 10 倍符号速率
符号速率	10.44 kbps	
码宽误差	<5%	不大于 10%
误码率	0	0

经过实际测试，各项指标都能够符合要求，本次实物作品较好的完成了基础实验要求，误差均在范围要求之内。

### 1.3 实物作品操作使用方法

整个系统的连接方式如下：

- 1.波形发生器的输出信号正极连接到开发板 1 的 D0 口，负极连接到开发板 1 的 GND 接口以输入基带信号；
- 2.同时将波形发生器产生的信号（基带信号）接到示波器上以便于后面观察解调性能；
- 3.使用同轴线将开发板 1 的 DA\_OUT 模拟输出端口与开发板 2 的 AD\_IN 模拟输入端口进行连接，将调制好的信号传输到另一块开发板，并转换为数字信号；

4.将开发板 2 的 D3 口通过连接线接到示波器上，以便于观察解调效果；

5.使用数据线将两块开发板分别连接到电脑的 USB 端口上。

将整个系统按照上述方式进行连接，将程序烧录进开发板之后即可观察系统运行情况。

程序加载方式如下所示：

点击"Implement top module"进行编译，之后点击"Generate Target PROM/ACE File"，先后双击"Boundary Scan"和"Initialize Chain"，初始化菊花链。双击"Create PROM File (PROM File Formatter)"选项，设置 PROM File。弹出选择窗口，选择源.bit 文件，双击工程窗口左侧的"Generate File..."等待创建成功。工程文件夹中就会出现设置好的 .mcs 文件。双击"Boundary Scan" 按键，新窗口下单击工具栏的连接图标，弹出窗口，选择"yes"。之后会弹出文件选择窗口，该窗口用来选择烧录文件为对应的 device，第一个窗口选"Bypass"，第二个步选.bit 文件。最后左键选择 xc3s500e 芯片，点击"program"完成烧录。这样我们就完成了 FPGA 开发板上程序的烧录。

加载完程序之后调整波形发生器，模式选择方波信号、调节频率、使高电平为 3.3V、低电平为 0V，按下 output 输出方波信号（基带信号）。



图2 波形发生器参数设置界面

此时整个系统开始工作，可以通过示波器检测实物作品的功能与性能。将所需要观察的信号连接到示波器的任一通道，打开相应通道即可观察相应的波形。将开发板 1 的 DA\_OUT 输出端口接到示波器上可以观察模拟调制信号。

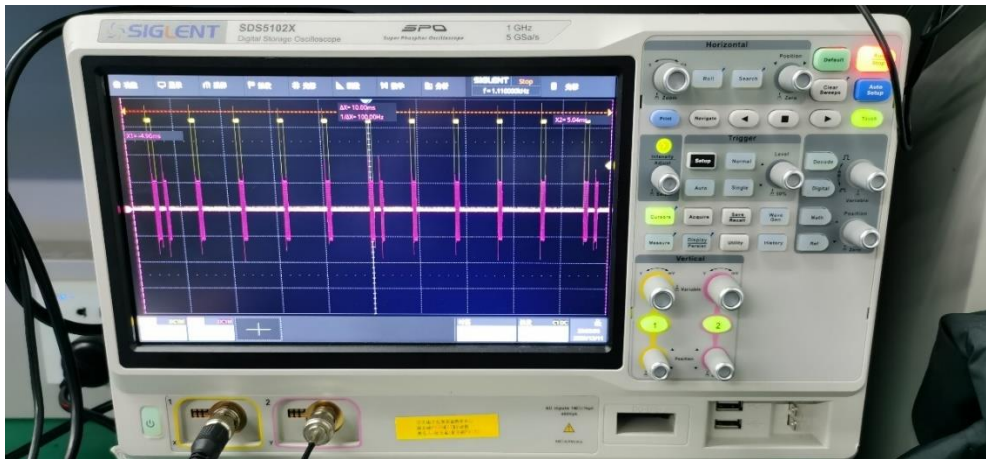


图3 观察调制信号

将开发板 2 的 D3 口接到示波器上可以观察解调之后输出的信号，此时可以在示波器上观察误码率、码元宽度误差、帧格式等内容。

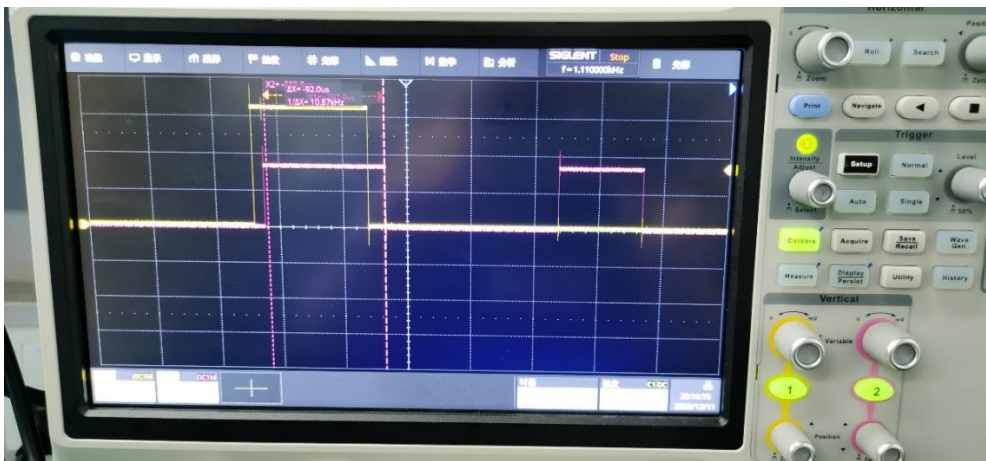


图4 解调输出信号与基带信号对比及帧格式

## 2 制作概况

制作过程主要分为硬件仿真与实物制作两部分进行。硬件仿真于第六周到第九周进行，实物制作于第十周到第十四周完成。首先明确实验任务，硬件仿真部分通过 ISE 编写 Verilog 代码并用 ModelSim 仿真实现 ASK、PSK 和 FSK 的调制解调，实物制作部分使用实验教学中心的 EELAB\_DIGIEXP 开发板实现数字通信，具体内容包括完成 AD/DA 转换、ASK 的调制与解调。接下来以周为单位，分阶段推进，在此期间，大家一边学习新知识一边将其切实运用到实验项目中。

本组作品的制作过程与时间记录如下表所示。

表3 制作过程记录

周次	制作进度
第六周	安装 ModelSim 和 ISE 软件并配置好环境
第七周	学习 Verilog 语言和软件使用
第八周	编写代码，分工完成三种调制方式部分
第九周	完成输入输出数据的串并转换部分代码， 对调制解调再次调试
第十周	确定实物选题，学习 ISE 烧录程序到 FPGA
第十一周	学习实验板，熟悉实验板各模块功能
第十二周	烧录例程（流水灯）、学习相关资料代码
第十三周	基本完成 ASK 的调制解调
第十四周	完善帧格式，改进程序达到要求，撰写报告

3 设计总结

实验整体流程设计如图所示，通过信号发生器输出一个周期性的高电平（3.3V）和低电平（0V）到一块实验板上，作为基带信号，对基带信号加入帧格式并进行 ASK 调制后，再将调制信号经过 DAC 完成 DA 转换后成为模拟信号通过同轴线传输至另一块实验板，经过 ADC 完成 AD 转换成为数字信号，而后进行 ASK 的解调，最终将解调后的信号输出。

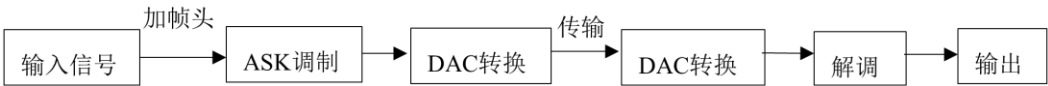


图5 设计流程图

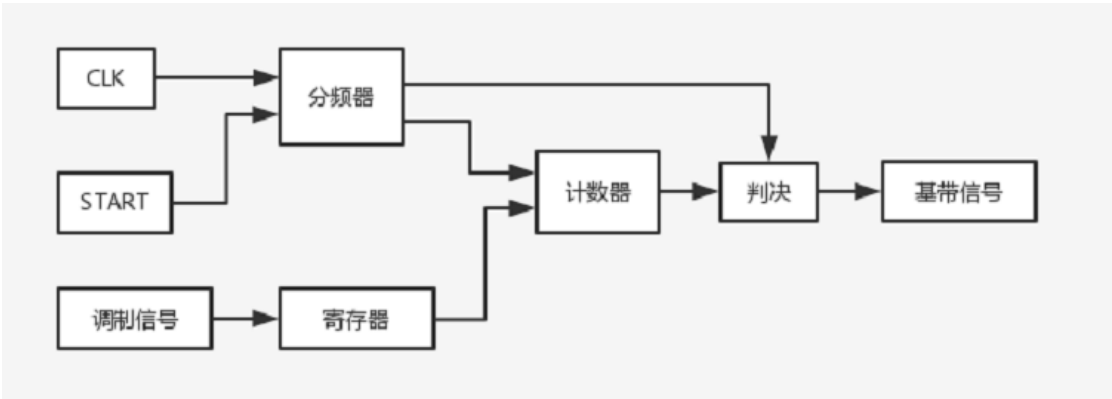


图6 解调流程框图

3.1 方案设计

本实验在方案设计时主要通过 ISE 编写程序进行 ASK、PSK 和 FSK 的调制解调并通过 ModelSim 软件仿真观察波形。

本系统硬件仿真部分采用的方案为通过 Verilog HDL 硬件语言编写数据串并转换模块，数据并串转换模块，通过调用 IP 核完成对 ASK、PSK、FSK 调制与解调各模块的编写，并



编写 testbench 测试文件对各个调制解调方式进行仿真。ASK 和 FSK 的调制通过数字基带信号与正弦载波相乘实现，PSK 中信号码元“0”和信号码元“1”对应正弦波相位相差 90 度。解调方式均为非相干解调中的过零检测法。

本系统未采用方案为使用包络解调法进行解调。该解调方案要求在 MATLAB 中利用滤波器设计工具设计滤波器，并生成滤波器的参数文件 coe 文件，供 ISE 中滤波 IP 核调用以生成 IP 核。该方案相对复杂，且对滤波器的设计有一定的要求，耗时较长，且在实际调试过程中，波形通过滤波器后有较大失真，输出波形并不理想，故并未采用该方案。

### 3.1.1 ASK 调制解调仿真

#### (一)理论分析

##### (1) 数据读入和并串转换模块：

首先，将待输入的四组学号存入 txt 文档中，图下图所示。



图7 输入信息

我们调用 Verilog 读取函数对 txt 文档内容循环读取，直到读空，并将读入数据以串行二进制形式储存，实现对输入数据的并串转换。

##### (2) 调制模块：

首先，通过 IP 核生成频率为 500KHz 的正弦载波，当基带信号为零时，调制信号为零；当基带信号为 1 时，调制信号为 500KHz 的正弦载波，得到调制信号。

##### (3) 解调模块：

解调采用过零检测法，即对调制信号的每个码元周期进行采样，统计每个码元周期内的下降零点个数，若大于 10，则该码元周期的解调结果为 1，反之为 0，得到解调信号。

##### (4) 串并转换和数据输出模块：

对串行二进制形式的解调结果进行遍历，每四位二进制数转换为一位十进制数，调用 Verilog 的写入函数将其写入 stunum\_out 文件，并在学号之间输出换行符，实现对解调信号的串并转换和输出。

#### (二)仿真过程

##### (1) 运行程序

ASK 文件夹下包含的文件如下图所示，点击 ASK.xise 在 ISE 中打开工程。



名称	修改日期	类型	大小
_xmsgs	2020/11/7 14:07	文件夹	
ipcore_dir	2020/11/7 14:07	文件夹	
iseconfig	2020/11/7 14:07	文件夹	
work	2020/11/7 14:07	文件夹	
xst	2020/11/7 14:07	文件夹	
ask.cmd_log	2020/11/7 13:19	CMD_LOG 文件	1 KB
ASK.gise	2020/11/7 14:06	GISE 文件	8 KB
ask.lso	2020/11/7 13:19	LSO 文件	1 KB
ask.ngc	2020/11/7 13:19	NGC 文件	23 KB
ask.ngr	2020/11/7 13:19	NGR 文件	12 KB
ask.prj	2020/11/7 13:19	PRJ 文件	1 KB
ask.stx	2020/11/7 13:19	STX 文件	0 KB
ask.syr	2020/11/7 13:19	SYR 文件	20 KB
ask.v	2020/11/7 13:34	V 文件	1 KB
ASK.xise	2020/11/7 13:34	Xilinx ISE Project	36 KB
ask.xst	2020/11/7 13:19	XST 文件	2 KB
ask_demodulation.v	2020/11/7 14:04	V 文件	1 KB
ask_envsettings.html	2020/11/7 13:47	Chrome HTML D...	10 KB
ask_module.v	2020/11/7 13:58	V 文件	1 KB
ask_summary.html	2020/11/7 13:47	Chrome HTML D...	6 KB
ask_Test.fdo	2020/11/7 13:51	FDO 文件	2 KB
ask_Test.udo	2020/11/7 13:19	UDO 文件	1 KB
ask_Test.v	2020/11/7 13:19	V 文件	2 KB
ask_Test_wave.fdo	2020/11/7 13:19	FDO 文件	1 KB
ask_xst.xrpt	2020/11/7 13:19	XRPT 文件	15 KB
stunum.txt	2020/11/6 15:58	TXT 文件	1 KB
stunum_in.v	2020/11/7 13:09	V 文件	2 KB

图8 ASK程序包含的文件

在下图界面中点击 Simulate Behavior 进行 ISE，ModelSim 联合仿真，并在弹出的 ModelSim 界面中观察仿真结果。

## (2) 程序说明

程序中所涉及各模块及调用关系如下图所示

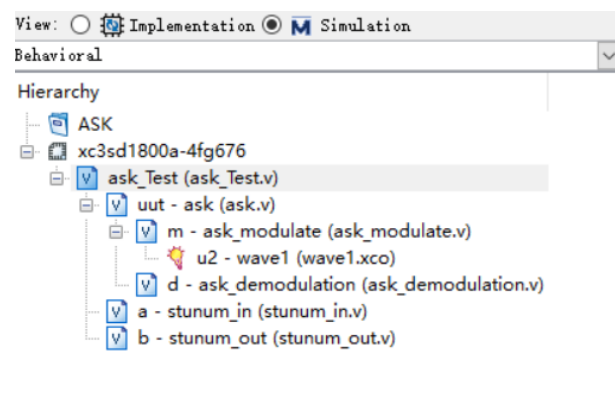


图9 ASK程序涉及模块及其关系

其中 ask 为顶层模块，实现对输入信号的调制与解调，该顶层模块当中调用了两个子模块 ask\_module（调制模块）和 ask\_demodulation（解调模块）。stunum\_in 模块将 stunum.txt 文件中的学号数据读入并进行并串转换，生成调制模块的输入数据；stunum\_out 模块将解调之后的数据重新转换为学号信息，并将其输出到 stunum.txt 文件当中。

## (一) 仿真结果

### (1) 输入

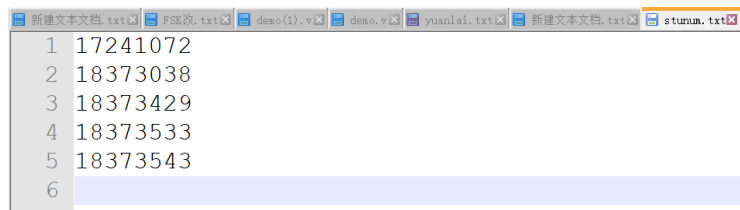


图10 输入信息

## (2) 输出



图11 输出信息

### (3) ModelSim 仿真波形

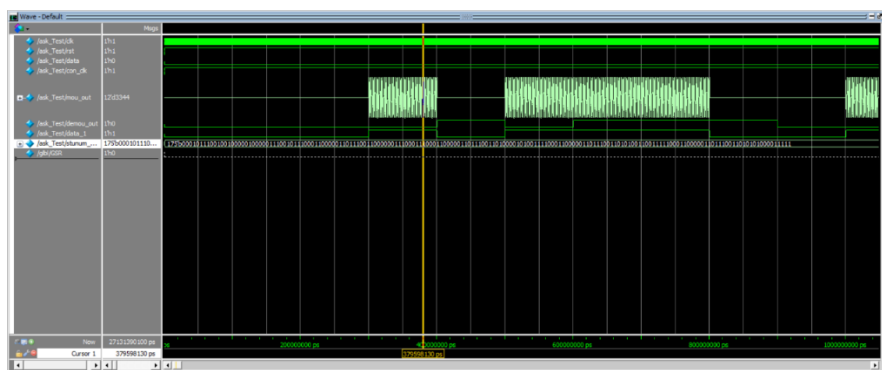


图12 ModelSim仿真波形

其中，clk 是输入的时钟信号，频率为 50MHz；rst 是复位信号；con clk 是开始信号。

`mou_out` 是调制后的输出，正弦载波对应基带信号的 1，0 对应基带信号的 0；`demou_out` 是解调之后的输出，比基带信号延时一个码元周期；`stu_in` 是并串转换之后的二进制序列。在 ASK 文件夹下找到生成的 `stunum_out` 文件可以看到输出到 `txt` 中的学号数据。

### 3.1.2 PSK 调制解调仿真

### (一)理论分析

串并、并串转换与输入输出模块与 ASK 部分相同，不再赘述。

(1) 调制模块:

首先，通过 IP 核生成频率为 100KHz 的正弦载波，当基带信号为 1 时，调制信号为 100KHz 的正弦波；当基带信号为 0 时，调制信号为相位相差  $180^\circ$  的 100KHz 的正弦波，得到调制信号。

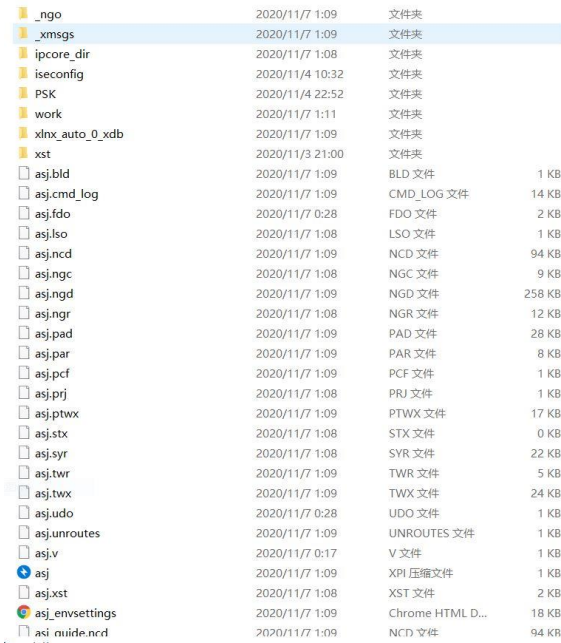
(2) 解调模块:

解调采用相干解调法，即利用乘法器，将调制信号与一路与载波同频同相的信号相乘得到一个相干载波，再对调制信号的每个码元周期进行采样，统计每个码元周期内的大于零的点的个数，由于硬件仿真时为理想情况，故只要存在大于零的点即可认为是 1，否则为 0。

## (二)仿真过程

### (1) 运行程序

PSK 文件夹下包含的文件如下图所示，点击 PSK.xise 在 ISE 中打开工程。



._ngo	2020/11/7 1:09	文件夹	
._xmsgs	2020/11/7 1:09	文件夹	
._ipcore_dir	2020/11/7 1:08	文件夹	
._iseconfig	2020/11/4 10:32	文件夹	
PSK	2020/11/4 22:52	文件夹	
work	2020/11/7 1:11	文件夹	
._xlnx_auto_0_xdb	2020/11/7 1:09	文件夹	
._xst	2020/11/3 21:00	文件夹	
asj.bld	2020/11/7 1:09	BLD 文件	1 KB
asj.cmd_log	2020/11/7 1:09	CMD_LOG 文件	14 KB
asj.fdo	2020/11/7 0:28	FDO 文件	2 KB
asj.iso	2020/11/7 1:08	LSO 文件	1 KB
asj.ncd	2020/11/7 1:09	NCD 文件	94 KB
asj.ngc	2020/11/7 1:08	NGC 文件	9 KB
asj.ngd	2020/11/7 1:09	NGD 文件	258 KB
asj.ngr	2020/11/7 1:08	NGR 文件	12 KB
asj.pad	2020/11/7 1:09	PAD 文件	28 KB
asj.par	2020/11/7 1:09	PAR 文件	8 KB
asj.pcf	2020/11/7 1:09	PCF 文件	1 KB
asj.prj	2020/11/7 1:08	PRJ 文件	1 KB
asj.ptwx	2020/11/7 1:09	PTWX 文件	17 KB
asj.stx	2020/11/7 1:08	STX 文件	0 KB
asj.syr	2020/11/7 1:08	SYR 文件	22 KB
asj.twr	2020/11/7 1:09	TWR 文件	5 KB
asj.twx	2020/11/7 1:09	TWX 文件	24 KB
asj.udo	2020/11/7 0:28	UDO 文件	1 KB
asj.unroutes	2020/11/7 1:09	UNROUTES 文件	1 KB
asj.v	2020/11/7 0:17	V 文件	1 KB
asj	2020/11/7 1:09	XPI 压缩文件	1 KB
asj.xst	2020/11/7 1:08	XST 文件	2 KB
asj_envsettings	2020/11/7 1:09	Chrome HTML D...	18 KB
asj_nuide.ncd	2020/11/7 1:09	NCD 文件	94 KB

图13 PSK程序包含的文件

在下图界面中点击 Simulate Behavior 进行 ISE，ModelSim 联合仿真，并在弹出的 ModelSim 界面中观察仿真结果。

## (2) 程序说明

程序中所涉及各模块及调用关系如下图所示

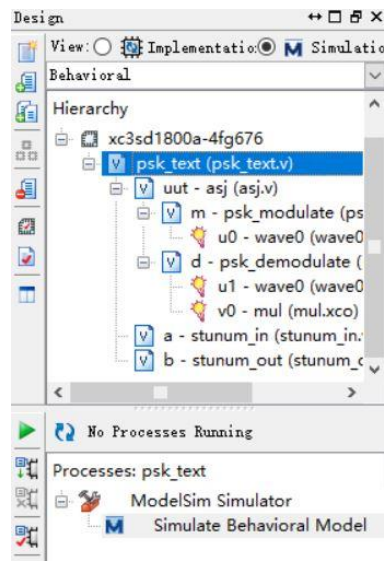


图14 ASK程序涉及模块及其关系

其中 `psk` 为顶层模块，实现对输入信号的调制与解调，该顶层模块当中调用了两个子模块 `psk_modulate`（调制模块）和 `psk_demodulation`（解调模块）。`stunum_in` 模块将 `stunum.txt` 文件中的学号数据读入并进行并串转换，生成调制模块的输入数据；`stunum_out` 模块将解调之后的数据重新转换为学号信息，并将其输出到 `stunum.txt` 文件当中。

## (三) 仿真结果

### (1) 输入

```
stunum - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
17241072
18373038
18373429
18373533
18373543
```

图15 输入信息

### (2) 输出

```
stunum_out - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
17241072
18373038
18373429
18373533
18373543
```

图16 输出信息

### (3) ModelSim 仿真波形

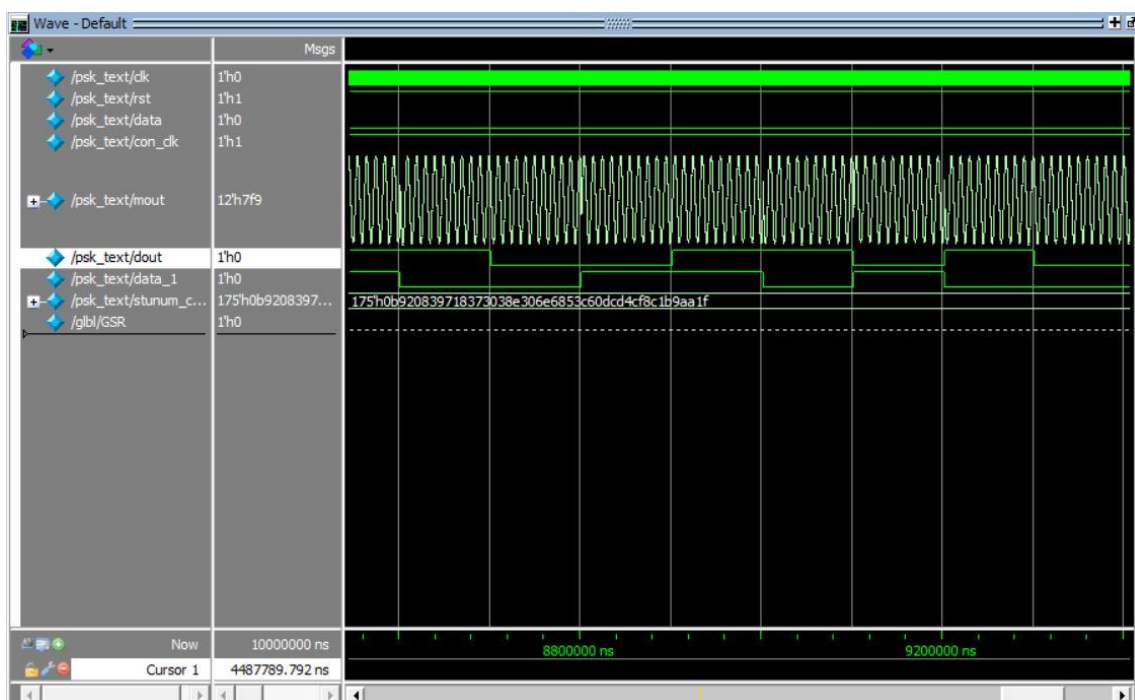


图17 ModelSim仿真波形

clk 是输入的时钟信号，频率为 50MHz；rst 是复位信号；con\_clk 是开始信号。

mout 是调制后的输出；dout 是解调之后的输出，比基带信号延时一个码元周期；data\_1 是并串转换之后的二进制序列。在 PSK 文件夹下找到生成的 stunum\_out 文件可以看到输出到 txt 中的学号数据。

#### 3.1.3 FSK 调制解调仿真

##### (一)理论分析

###### (1) 数据读入和并串转换模块：

由于 FSK 的解调方式与 ASK 一样都使用了过零检测，所以 FSK 仿真当中使用了与 ASK 仿真当中一样的数据并串、串并转换模块。其具体的原理及实现功能已经在 ASK 仿真部分进行了介绍。

###### (2) 调制模块：

由于 FSK 调制需要使用两个不同频率的载波，所以在程序设计时首先调用两个 DDS 核，分别生成频率为 100k 和 500k 的两个载波，通过调整 DDS 核中的参数使得载波每一时刻输出的值保存在一个 12 位的变量中（其中第 12 位是符号位）。然后使用连续赋值语句

```
assign mou_out=(data==1'b1) ? wave1_reg : wave0_reg;
```

图18 连续赋值语句

即如果数据为 1 则调制信号的输出为载波 1（500kHz），如果数据为 0 则调制信号的输出为载波 2（100kHz）。此时便完成了信号的调制。

### (3) 解调模块:

解调采用过零检测法, 首先对调制信号进行整形, 将调制信号大于 0 的部分记为 1, 将载波小于 0 的部分记为 0。由于调制模块的输出就是解调模块的输入, 所以解调模块每一时刻的输入仍是一个 12 位的变量 (调制信号), 且该变量的第一位是符号位。取出每一时刻调制信号的符号位即完成了对调制信号的整形。然后使用 50MHz 的时钟对整形后的信号进行检测, 检测到由 1 到 0 跳变说明出现了一个零点, 零点数加 1 (没有检测由 0 到 1 的跳变, 所以一个周期内只记录一个零点)。经过一个码元周期之后进行判决, 如果零点数大于 30 则解调为 1, 否则解调为 0。此时即完成了对调制信号的解调。

### (4) 顶层模块:

顶层模块的输入为时钟信号、复位信号、输入的基带信号, 输出为解调输出信号。在顶层模块中调用调制解调两个模块, 将调制模块的输出作为解调模块的输入即完成了顶层模块设计。

### (5) 测试文件:

在测试文件中调用顶层模块以及并串、串并模块, 将从 txt 中读取的数据作为顶层模块的基带数据, 再将模块的输出的解调信号进行串并转换之后输出到另一个 txt 文件中。设计好程序之后既可以使用 ModelSim 仿真观察仿真结果。

## (二) 仿真过程

### (1) 运行程序

fsk 文件夹下包含的文件如下图所示, 点击 fsk.xise 在 ISE 中打开工程。

._xmsgs	2020/11/6 22:29	文件夹	
ipcore_dir	2020/11/6 22:25	文件夹	
iseconfig	2020/11/6 22:43	文件夹	
work	2020/11/6 22:47	文件夹	
fsk.bld	2020/11/6 17:49	BLD 文件	2 KB
fsk.cmd_log	2020/11/6 17:49	CMD_LOG 文件	12 KB
fsk.gise	2020/11/6 22:48	GISE 文件	18 KB
fsk.lso	2020/11/6 17:43	LSO 文件	1 KB
fsk.ncd	2020/11/6 17:49	NCD 文件	64 KB
fsk.ngc	2020/11/6 17:43	NGC 文件	25 KB
fsk.ngd	2020/11/6 17:49	NGD 文件	182 KB
fsk.ngr	2020/11/6 17:43	NGR 文件	17 KB
fsk.pad	2020/11/6 17:49	PAD 文件	28 KB
fsk	2020/11/6 17:49	Solid Edge Part ...	8 KB
fsk.pcf	2020/11/6 17:49	PCF 文件	1 KB
fsk.prj	2020/11/6 17:43	PRJ 文件	1 KB
fsk.ptwx	2020/11/6 17:49	PTWX 文件	17 KB
fsk.stx	2020/11/6 17:43	STX 文件	0 KB
fsk.syr	2020/11/6 17:43	SYR 文件	20 KB
fsk.twr	2020/11/6 17:49	TWR 文件	5 KB
fsk.twx	2020/11/6 17:49	TWX 文件	24 KB
fsk.unroutes	2020/11/6 17:49	UNROUTES 文件	1 KB
fsk.v	2020/11/6 22:35	V 文件	1 KB
fsk	2020/11/6 22:48	Xilinx ISE Project	36 KB
fsk	2020/11/6 17:49	好压 XPI 压缩文件	1 KB
fsk.xst	2020/11/6 17:43	XST 文件	2 KB
fsk_demodulation.fdo	2020/11/2 11:21	FDO 文件	2 KB
fsk_demodulation.udo	2020/11/2 11:21	UDO 文件	1 KB
fsk_demodulation.v	2020/11/6 21:04	V 文件	1 KB
fsk_demodulation_wave.fdo	2020/11/2 11:21	FDO 文件	1 KB
fsk_envsettings	2020/11/6 22:59	HTML 文档	20 KB

图19 FSK程序包含的文件

在下图界面中点击 Simulate Behavior 进行 ISE, ModelSim 联合仿真, 并在弹出的 ModelSim 界面中观察仿真结果。

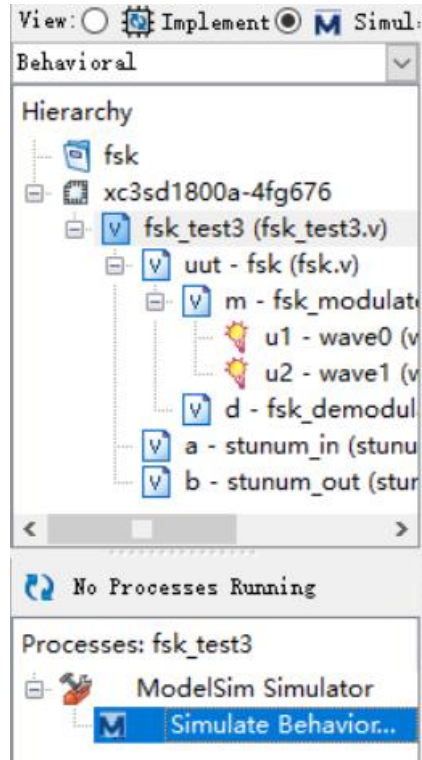


图20 联合仿真操作说明

(2) 程序说明

程序中所涉及各模块及调用关系如下图所示

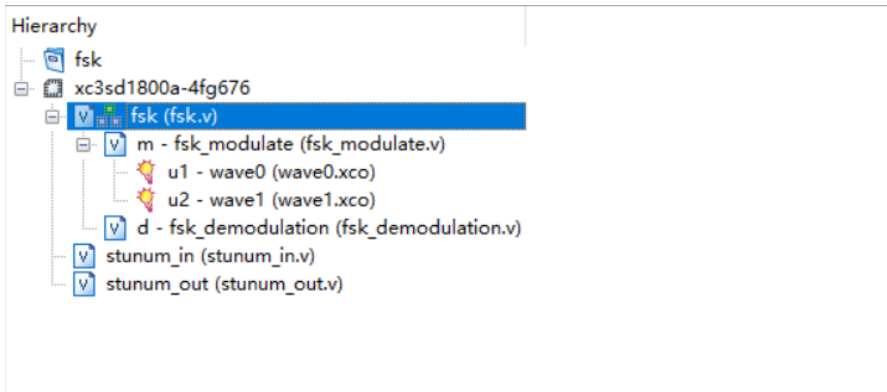


图21 FSK程序涉及模块及其关系

其中 fsk 为顶层模块, 实现对输入信号的调制与解调, 该顶层模块当中调用了两个子模块 fsk\_modulate (调制模块) 和 fsk\_demodulation (解调模块)。stunum\_in 模块将 stunum.txt 文件中的学号数据读入并进行并串转换, 生成调制模块的输入数据; stunum\_out 模块将解调之后的数据重新转换为学号信息, 并将其输出到 stunum.txt 文件当中。



(三)仿真结果

(1) txt 输入

```
stunum - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
17241072
18373038
18373429
18373533
18373543
```

图22 输入信息

(2) txt 输出

```
stunum_out - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
17241072
18373038
18373429
18373533
18373543
```

图23 输出信息

ModelSim 界面各输入输出信号波形如下图所示

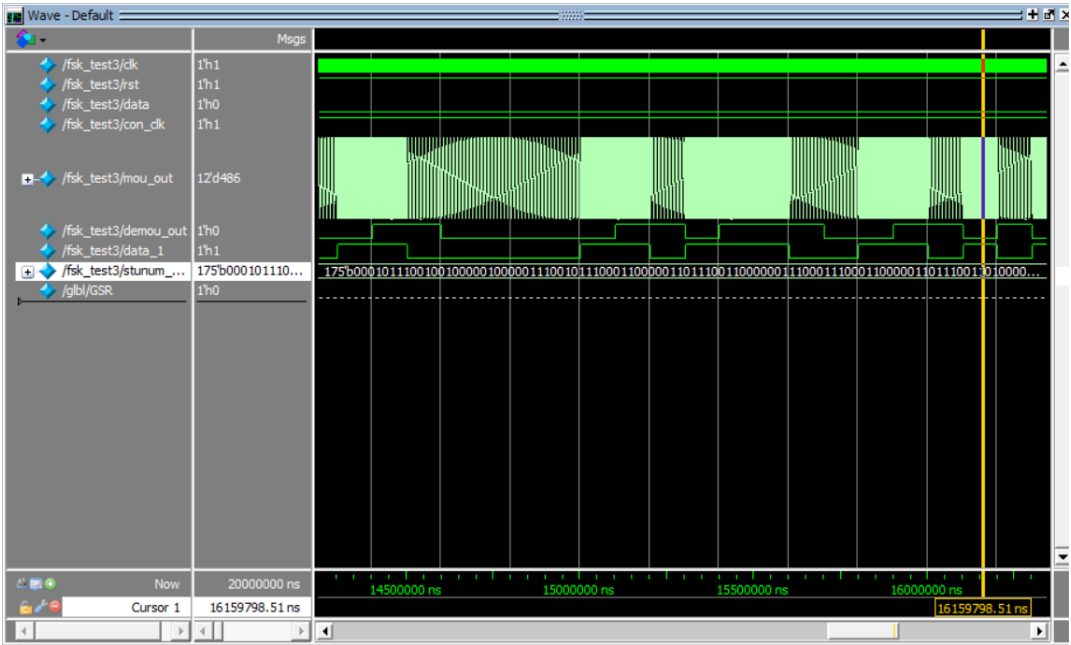


图24 ModelSim仿真波形

clk 是输入的时钟信号，频率为 50MHz；rst 是复位信号；con\_clk 是开始信号；mou\_out 是调制后的输出；demou\_out 是解调之后的输出；stu\_in 是并串转换之后的二进制序列。在 fsk 文件夹下找到生成的 stunum\_out 文件可以看到输出到 txt 中的学号数据。

## 3.2 硬件设计

### 3.2.1 EELAB-FPGACORE2 硬件实验平台

硬件实验平台 EELAB-FPGACORE2 电路采用 XILINX 公司 Spartan™-3E 系列的芯片 XC3S500E-VQG100 作为核心芯片，程序配置 PROM 芯片为 XCF04SV020。EELAB-FPGACORE2 本身集成了编程器，不需要外置的编程模块就可以直接使用。EELAB-FPGACORE2 的输入输出模块主要有 3 路按键、2 路拨码开关、4 路 LED、2 位七段数码管、6 路 ADC、1 路 USB 串口和 17 个用户可扩展 I/O，EELAB-FPGACORE2 外观形状及接口位置等采用的是和 Arduino-UNO 模块兼容的形式,可以与 Arduino 常见扩展模块直接连接。该硬件实验平台完全可以满足高校数字电路基础教学实验要求，为了满足综合实验要求，EELAB-FPGACORE2 背面还具有专用扩展接口（包含 28 个用户可扩展 IO），可以无缝连接到 EELAB-FBOARD1 上，实现更多的扩展功能。

Spartan™-3E FPGA 是数字视频、工业、医疗、通信与数字消费类电子应用中的大容量、以门电路为核心的可编程逻辑设计的理想选择。EELAB-FPGACORE2 主芯片 XC3S500-VQG100E 将大量的可编程逻辑、IP 核、高级时钟电路和嵌入式存储器与多种快速互连结构整合在一起。其具有 50 万门规模，10476 个可配置的逻辑单元，66 个 I/O 引脚，4 个 DCM 数字时钟管理模块，360Kb 的 Block RAM，73Kb 的分布式 Distributed RAM，20 个用于高性能 DSP 应用的嵌入式 18×18 乘法器。

EELAB-FPGACORE2 硬设计电路图与模块框图如下。

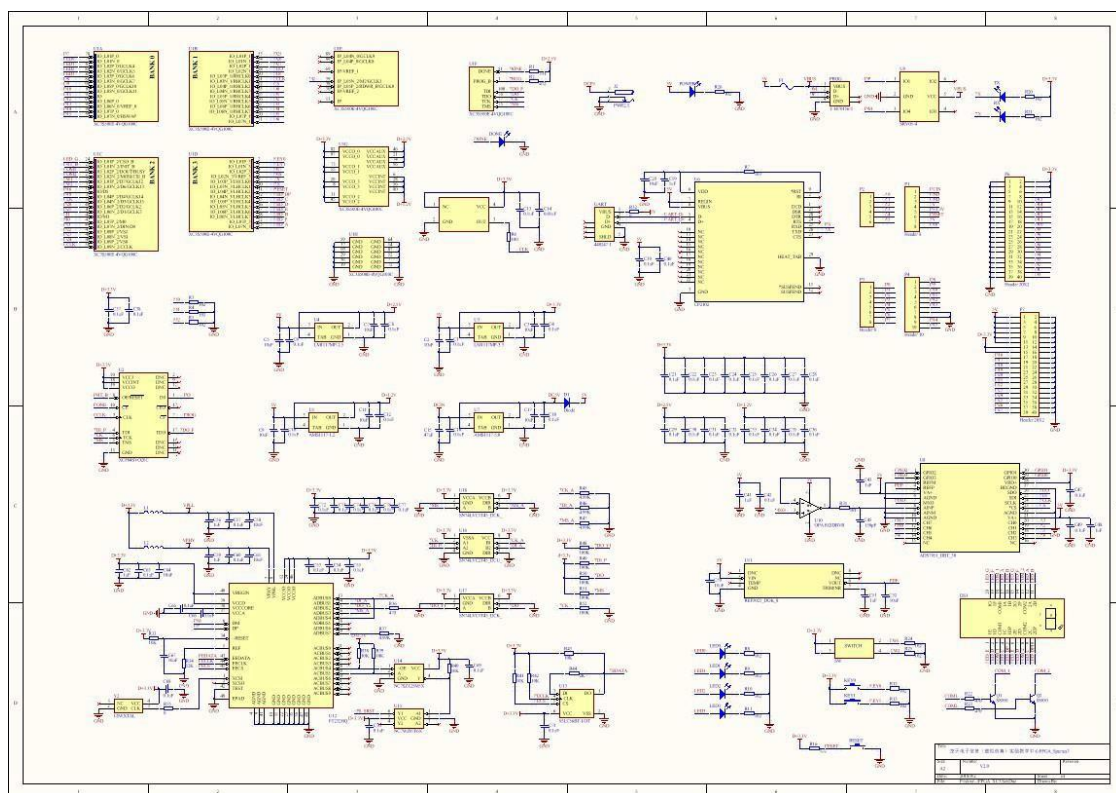


图25 硬件原理图

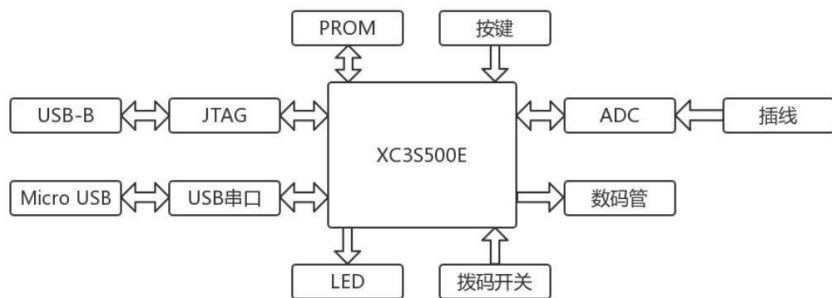


图26 EELAB-FPGACORE2 连接框图

根据以上系统框图的功能划分，对各模块原理进行分别介绍。

### (一)JTAG 接口

JTAG(Joint Test Action Group, 联合测试工作组)是一种国际标准测试协议（IEEE 1149.1 兼容），主要用于芯片内部测试。其基本原理是在器件内部定义一个 TAP（Test Access Port，测试访问口）通过专用的 JTAG 测试工具对内部节点进行测试。JTAG 测试允许多个器件通过 JTAG 接口串联在一起，形成一个 JTAG 链，能实现对各个器件分别测试。

EELAB-FPGACORE2 配有 JTAG 接口，支持 JTAG（边界扫描模式）配置模式。这种配置模式不需要 PROM，操作方便快捷。JTAG 口包含 TCK（Test Clock，测试时钟输入）、TDI（Test Data In，测试数据输入）、TDO（Test Data Out，测试数据输出）和 TMS（Test Mode Select，测试模式选择）4 个引脚。为了方便电路调试，EELAB-FPGACORE2 集成了编程器，不需要外置的编程模块就可以直接通过 Micro USB 形式的 JTAG 接口进行编程测试。

JTAG 接口原理图如图所示。

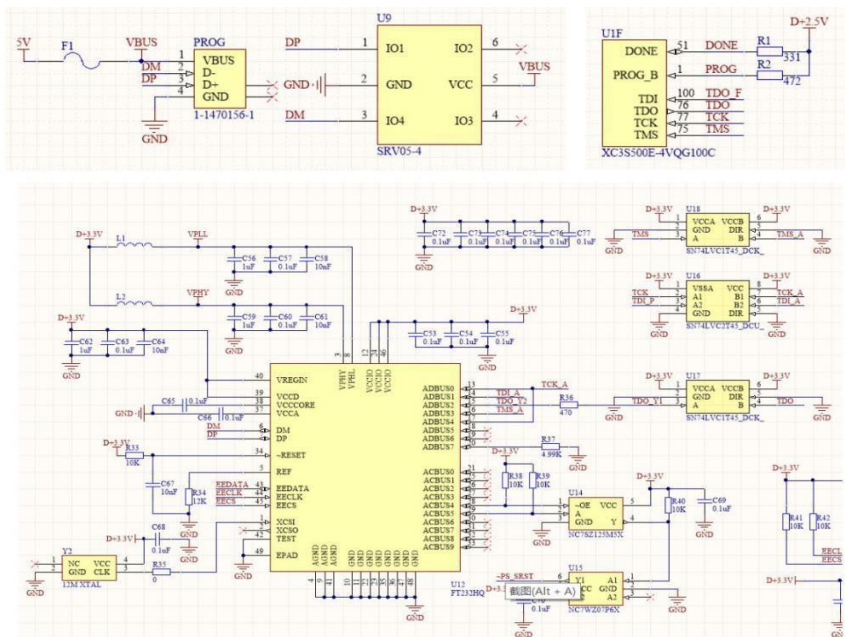


图27 JTAG接口电路

(二)PROM

EELAB-FPGACORE2 采用了 Xilinx Platform Flash 配置解决方案,选用的 PROM 芯片为 XCF04SV020,供电电压 VCCINT、VCCO 和 VCCJ 均为 3.3V,密度为 4Mbit。该用于配置 XilinxFPGA 的 PROM 采用低功耗高级 CMOS NOR FLASH 工艺,具有 20000 个程序/擦除周期的耐久性,使用 Xilinx Alliance ISE 和 Foundation ISE 系列软件包进行设计支持。

PROM 电路原理图如图所示。

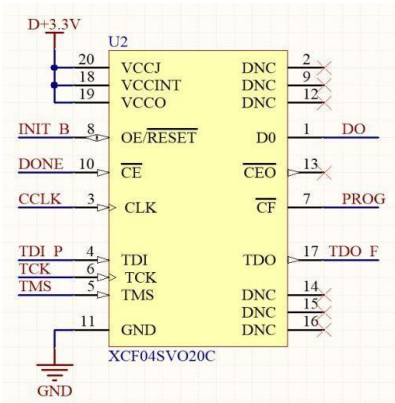


图28 PROM电路原理图

(三)按键

EELAB-FPGACORE2 配有 3 路独立按键,采用非锁死按键形式,其中 1 路(定义为 RESET)一般作为复位键使用,默认输入状态为高电平,按下按键输入为低电平。其余 2 路默认输入状态为低电平,按下按键输入为高电平。

按键电路原理图如图所示。

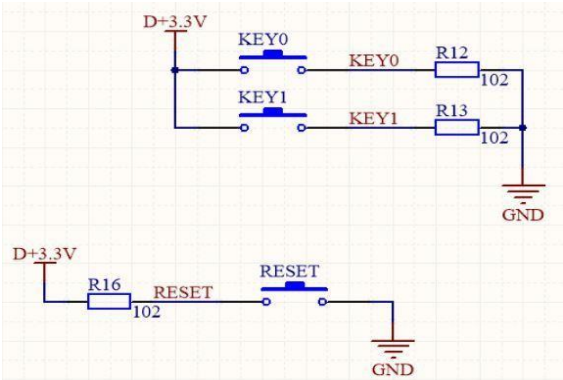


图29 按键电路原理图

下表是按键的引脚分配。

表4 按键引脚分配

FPGA 引脚序号	引脚定义	引脚说明
P11	RESET	复位按键

(四)ADC

EELAB-FPGACORE2 的 ADC 电路采用 ADS7951 芯片，ADS7951 是一个 12 位引脚兼容型 8 通道模数转换器，采样率为 1MHz，串行接口速率达到 20MHz。ADS7951 包含一个基于电容器的 SAR 模数转换器，具有固有的采样保持；接受从 2.7V 到 5.25V 的宽模拟电源范围；极低功耗使其适用于电池供电和隔离电源应用；零等待时间。

ADC 电路原理图如图所示。

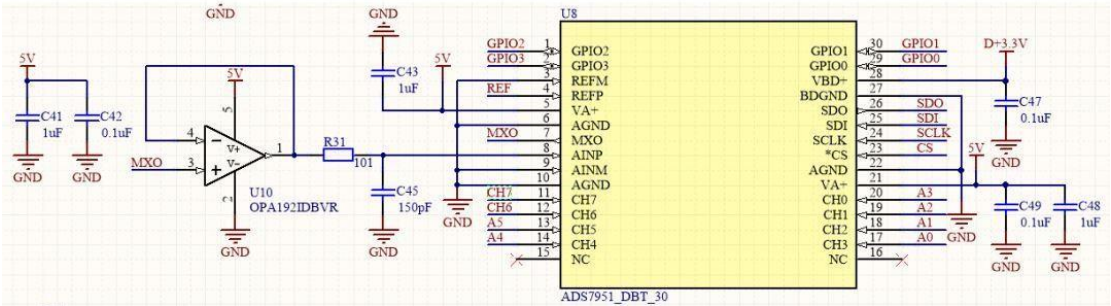


图30 ADC电路原理图

下表是 ADC 的引脚分配。

表5 ADC 引脚分配

FPGA 引脚序号	引脚定义	引脚说明
P58	SDO	串行数据输出（以 ADS7951 为判断方向）
P60	SDI	串行数据输入（以 ADS7951 为判断方向）
P61	SCLK	串行时钟输入（以 ADS7951 为判断方向）
P62	CS	片选输入（以 ADS7951 为判断方向）

(五)USB 串口

为了方便 EELAB-FPGACORE2 与计算机之间进行进行数据交互，EELAB-FPGACORE2 扩展了 USB 串口，采用 Micro USB 接口形式，可实现 RS232 通信。



USB 串口电路原理图如图所示。

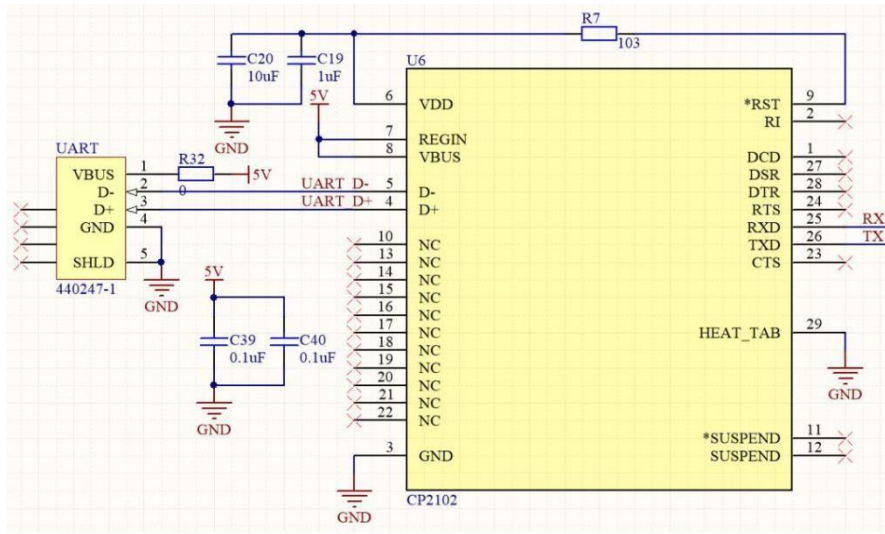


图31 USB串口电路原理图

### USB 串口的引脚分配。

### 表6 USB 串口引脚分配

FPGA 引脚序号	引脚定义	引脚说明
P9	TX	串口发送端（以 EELAB-FPGACORE2 为判断方向）
P10	RX	串口接收端（以 EELAB-FPGACORE2 为判断方向）

## (六)可扩展 IO

为了方便扩展，EELAB-FPGACORE2 正面配置了 16 个间距 2.54mm 的用户可扩展 IO，背面还具有专用扩展接口（包含 28 个用户可扩展 IO）。

下表是 EELAB-FPGACORE2 正面可扩展 IO 的引脚分配。

表7 可扩展 IO 的引脚分配

FPGA 引脚序号	引脚定义	引脚说明
P63	D0	可扩展 IO-0
P65	D1	可扩展 IO-1
P66	D2	可扩展 IO-2
P67	D3	可扩展 IO-3
P68	D4	可扩展 IO-4
P70	D5	可扩展 IO-5
P71	D6	可扩展 IO-6
P78	D7	可扩展 IO-7
P86	D8	可扩展 IO-8
P90	D9	可扩展 IO-9
P91	D10	可扩展 IO-10
P92	D11	可扩展 IO-11
P94	D12	可扩展 IO-12
P95	D13	可扩展 IO-13
P98	D14	可扩展 IO-14
P99	D15	可扩展 IO-15
P34	D16	可扩展 IO-16

### 3.2.2 DAC/ADC

本实验采用的 EELAB-DIGIEXP 实验板配有一路 DAC 输出通道和 ADC 输入通道，其中 DAC 选用 TI 公司的 12 位串行输入数模转换器芯片 DAC7811，ADC 选用 TI 公司的双路 14 位同步采样模数转换芯片 ADS7853。

#### (一)DAC7811

DAC7811 采用 SPI 通信接口，串行时钟速率最高可达 50MHz，具有回读数据功能，用户可以通过 SDO 引脚读取 DAC 寄存器的内容；工作于 2.7V 至 5.5V 的电源供应，输出电压范围为-2.5V 至 2.5V；在 D/A 转换电路与输出的 SMA 接口之间连接了一个 5 阶



低通滤波器，用于滤除 DAC 输出的模拟信号的谐波噪声；可提供出色的四象限乘法特性以及 10MHz 的大信号乘法带宽。由施加的外部基准输入电压( $V_{REF}$ )确定满标量程输出电流。当其与外部电流到电压精密放大器结合使用时，集成式反馈电阻器( $R_{FB}$ )可提供温度跟踪和满标量程电压输出。

其电路结构为经典 R-2R 倒 T 型电阻网络，如下图所示。

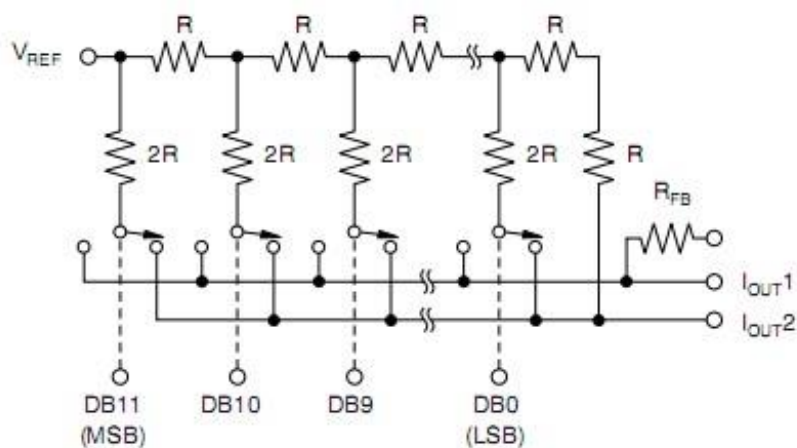


图32 倒T型电阻网络

DAC7811 电路原理如下图所示。

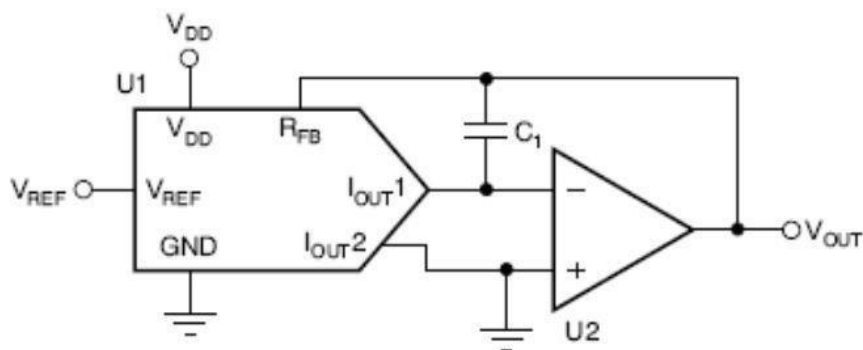


图33 DAC7811电路原理图

## (二)ADS7853

ADS7853 是一款 14 位单端伪差分单级逐次逼近型寄存器(SAR) ADC，其最大吞吐量为 1MSPS(即数据传输速率，对于多通道同步采样 ADC，其吞吐量=最大采样率\*同步采样通道数)；包含两个可独立编程的基准电压源，可用于系统级的增益校准；配有一个可在宽

电源供电范围内运行的灵活串行接口，从而轻松实现与多种主机控制器的通信；支持两种低功耗模式，可针对给定输出优化功耗。

ADS7853 功能框图如下图所示。

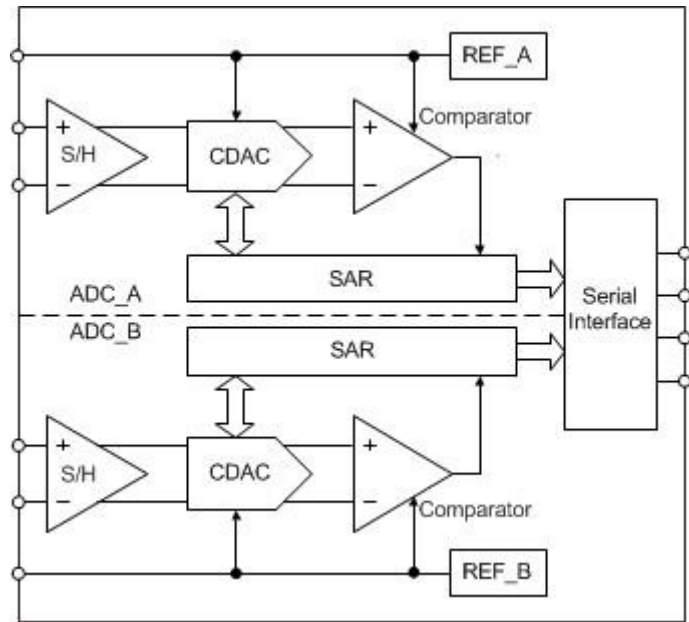


图34 ADS7853功能框图

3.3 软件设计

3.3.1 程序流程

软件设计原理框图如图所示。

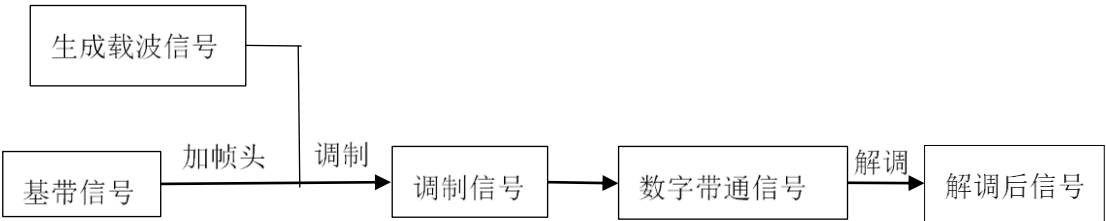


图35 软件设计原理框图

3.3.2 各程序模块详细介绍

(一)DCM 数字时钟管理模块

DCM (digital clock manager) 是较高级 FPGA 产品中集成的专门用于时钟综合、消除时钟偏斜和进行时钟相位调整的固件资源，利用 DCM 完成时钟倍频、分频、相移十分方

便，给 FPGA 的系统时钟设计带来了方便。DCM 动态重配置设计利用一个常有的时钟对 DCM 的工作状态标识进行监测，当 DCM 由于输入时钟的瞬时抖动或突然变化而失锁后，自动产生一个脉冲将 DCM 复位，使其重新锁定并恢复正常工作。应用 FPGA 中内嵌的数字时钟管理（DCM）模块可以建立可靠的系统时钟。

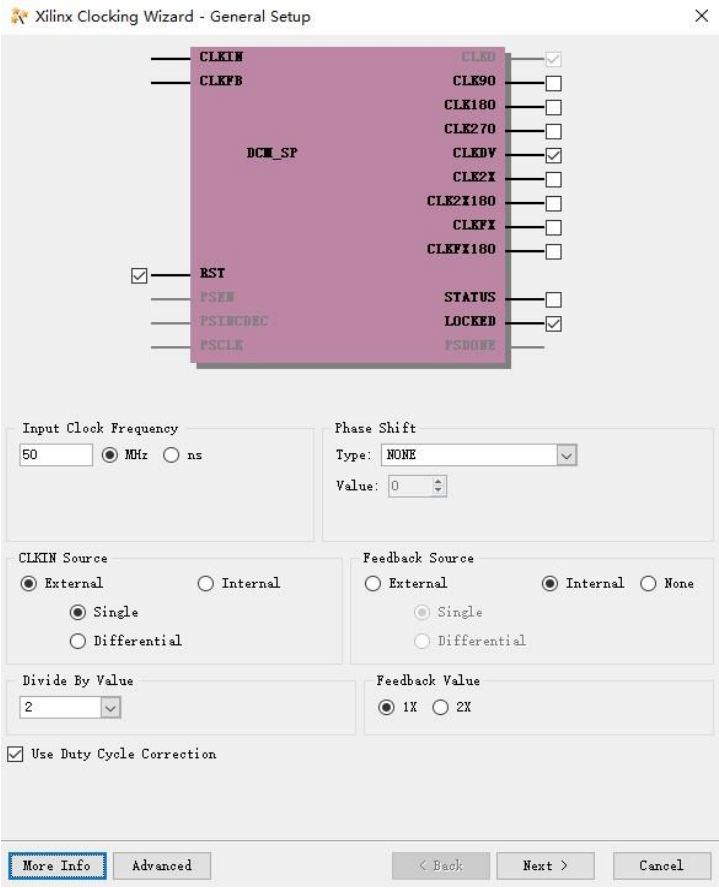


图36 DCM数字时钟管理模块

上图为 DCM 参数设定，其中输入时钟频率为系统时钟，设定 50MHz 且为单端外部信号，无相移，设定分频倍数为 2 及内部反馈倍数为 1。模块的 Verilog 程序调用如下。

```
DCM_25 instance_name ( //DCM25模块
    .CLKIN_IN(sys_clk),
    .RST_IN(1'b0),
    .CLKDV_OUT(AD_SCLK),
    .CLKIN_IBUFG_OUT(CLKIN_IBUFG_OUT),
    .CLK0_OUT(clk),
    .LOCKED_OUT(LOCKED_OUT)
);
```

图37 DCM25模块程序

通过调用 DCM 模块得到了稳定后的可靠的系统时钟。

## (二)载波产生

EELAB-FPGACORE2 实验平台的系统时钟为 50MHz，利用系统时钟可以通过查表法产生单频正弦波，查表步长由以下公式得出

$$Freqstep = \frac{2^{48}}{CLK \times Freq} \quad (1)$$

其中 CLK 为系统时钟 50MHz，Freq 为所需单频正弦波的频率，Freqstep 即为产生对应频率载波的查表步长。

模块的 Verilog 程序调用如下。

```
single_sintable single_sintable (                                     //单频正弦波发生函数
    .clka(clk), // input clka
    .ena(1'b1), // input ena
    .addra(Address), // input [11 : 0] addra
    .douta(DA_Rom1_Data) // output [11 : 0] douta
);
```

图38 单频正弦波发生函数

## (三)DAC 串行接口 SPI 模块

12 位 DAC 通过串行外设接口 SPI 控制和数据传输。传入 DAC 为 16 位信号，其中前 4 位为控制信号 (ctrl)，后 12 位为需要 DA 转换的数字信号，SPI 模块工作流程为初始计数时置 0，之后经过 16 个时钟周期即 32 个计数将一组 16 位信号上传到 DAC7811，再经过 16 个计数的空闲期最终将数据初始化完成一个周期的上传。同时，上传时帧同步置 0，其他时间置 1，此时每个轮回刚好为 50 个系统时钟周期，由系统时钟频率 50MHz，得转换速率 SR 为 1MSPS，与 ADC 最大转换速率相匹配，通过公式便可得到 DAC 采样率。

$$SampleRate = \frac{1000}{SR - 1} \times 50 \quad (2)$$

其中 SR 为转换速率，单位 KSPS，SampleRate 即为 DAC 模块输入参数中采样率。

模块的 Verilog 程序调用如下。

```
//DAC7811_SPI模块
DAC7811_SPI DAC7811_SPI (
    .CLK_IN(clk),
    .RESET(sys_rst),
    .DAC_EN(1'b1),
    .DAC_SDO(DAC_SDO),
    .SampleRate(SampleRate),
    .DA_Data({Ctrl, DA_Data}),
    .DAC_SCLK(DAC_SCLK),
    .DAC_SYNC(DAC_SYNC),
    .DAC_SDI(DAC_SDI)
);
```

图39 DAC7811\_SPI模块

#### (四)ADC 串行接口 SPI 模块

ADC 芯片串行时序完全遵从 SPI 协议，当片选信号处于低电平有效时，在串行时钟 SCLK 的上升沿，数据从高位到低位依串行写入时钟芯片内部寄存器，每个寄存器写入顺序不能随意改变，每个寄存器数据的值由芯片内部特性决定，可根据芯片的 Data\_Out 确定每一位的数值,本实验 AD 模块的芯片为 ADS7853，为 14 位双通道串行 ADC。

ADC\_SPI 的程序主要包含这几个部分：时钟芯片内部存储器数据赋初值：本部分由 AD\_top 进行完成实现。外部激励：即 RST 复位信号。数据串行转换：为 AD\_SPI 模块主要内容，根据 CS 信号，当 CS 信号表示低电平时进行有效转换，转换时是一位一位传输的，转换完后将相关状态位拉至低位。

模块的 Verilog 程序调用如图 39 所示。

#### (五)帧格式

加入帧头是为了能够有效的辨别有效数据，设计帧格式可以使用信号的幅度、频率、调制形式等特征保证帧头不会与后面的数据重复。我们利用频率特征设计帧格式以保证帧头不会与有效数据出现重复，具体的帧格式设计如下：由于我们设置的一个码元的宽度为 90us，所以只要解调没有出现错误就不可能出现宽度小于 90us 的信号，所以我们在每一帧数据之前加入长度为 65us 的一个 0 和一个 1 作为帧头用于区分有效数据。我们规定波形发生器发出产生的五个周期的信号为一帧数据。在每一帧数据的固定位置加入帧头。

#### (六)2ASK 调制

根据 2ASK 调制的定义，即高电平时有信号，低电平时无信号，由于 DAC7811 为 12 位无符号输入，中间值 2048 对应为低电平,4095 对应为波峰，0 对应为波谷。即当有

信号时输出载波 1，载频为 200kHz，当无信号时输入到 DAC 为固定键值 2048，对应输出电压为 0。

Verilog 代码如下：

```
assign DA_Data = data_output ? DA_Rom1_Data :2048; //ASK
```

图40 2ASK调制代码

## (七)2ASK 解调

本实验中我们采用的解调方式如下：首先，根据调制信号无载波区域和有载波区域幅度的不同设定判决门限，并用 5MHz 时钟进行采样，每 5us 内若幅值高于此门限的个数超过 30，则判定该时段解调结果为 1，否则为 0。

具体而言，调制信号通过传输线输入到负责解调的 eclab 板的 AD\_in 口，在程序中，我们调用 ADtop 模块，将输入信号赋值给十四位寄存器变量 Data\_InB，并通过 ISE 软件的 ChipScope 调试软件观测 Data\_InB 的值。

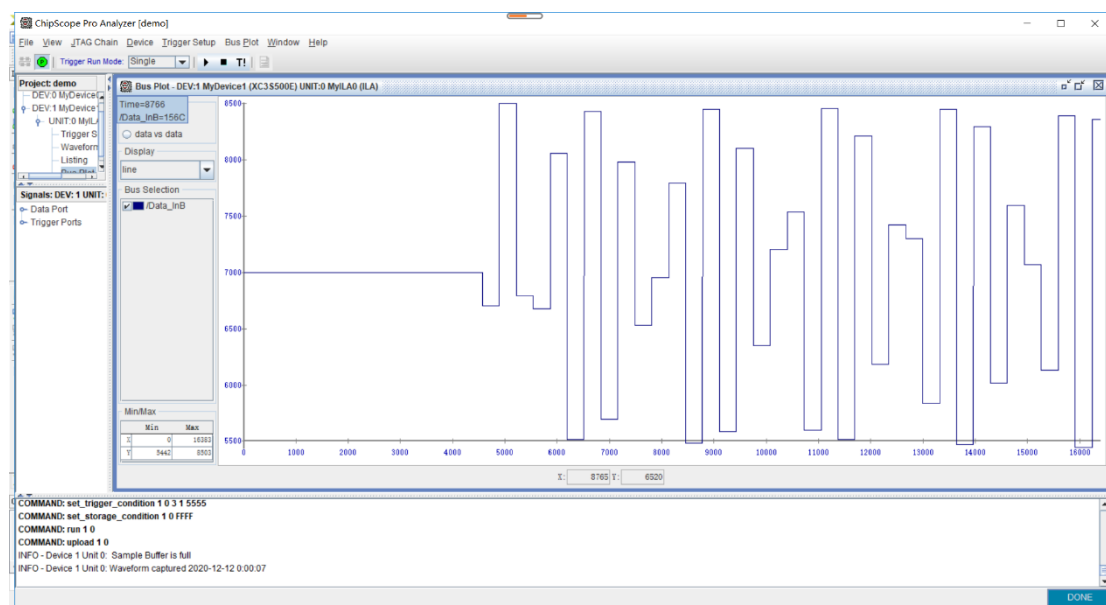


图41 ChipScope调试软件中Data\_InB曲线

完成 ChipScope 调试软件配置后运行程序，打开 Bus Plot 窗口，可以看到 Data\_InB 曲线呈周期性变化，当调制信号为正弦载波时，Data\_InB 曲线由于采样率较低呈现近似正弦波动的波形；当调制信号为 0 时，Data\_InB 曲线近似恒定不变。

当调制信号为零时，由于信号传输过程中存在噪声，Data\_InB 并不是恒定不变，而是在一个范围内波动，如下图所示。

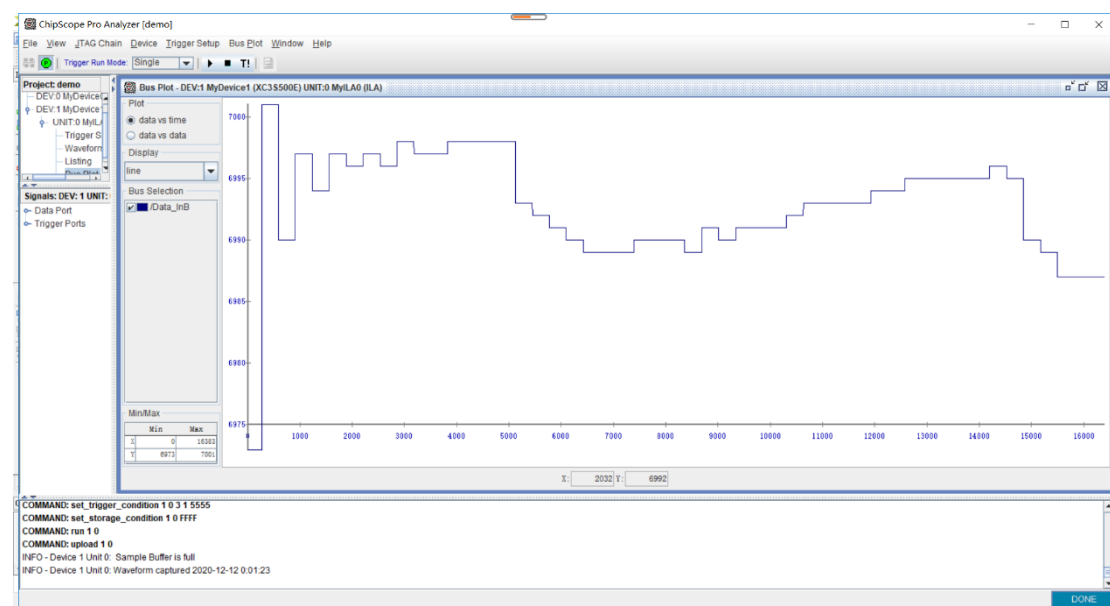


图42 ChipScope调试软件中噪声的Data\_InB曲线

通过多次运行观测，可以确定当调制信号为零时，噪声的 Data\_InB 上限为 7025，同时，经过测试，当调制信号为正弦载波时，用 5MHz 时钟进行采样，每 5us 内 Data\_InB 高于 7025 的个数超过 30，因此，判决门限设定为 7025。

## 3.4 其他设计

### 3.4.1 码元宽度误差和抖动

由于调制信号 0 和 1 边缘电压的过渡和抖动以及解调方式未经过整流滤波，受噪声影响较大，故解调信号码元宽度的变化，也就是连续运行时的抖动不可避免。

消除码宽抖动、减小码宽误差可以采用整流滤波取平均值的方法进行解调。

### 3.4.2 帧头

本作品的帧头设计为：五个方波周期为一帧，每一帧在开始的位置上固定位置产生一段 65us 的高脉冲，由于帧头宽度小于码元宽度 90us，因此，在理想情况下，帧头可以和有效数据分开。

但是在实际工程中，由于传输过程中的损耗，有效数据两侧会被消耗，导致有效数据宽度变低，此时帧头和有效数据的宽度变得难以区分，帧头便不能和有效数据分开。



因此，我们可以采用对帧头进行与有效数据不同的调制方式进行调制，改变其调制信号的幅度等参数，保证其能在实际工程应用与有效数据分开。

## 4 调试过程

本系统的调试步骤包括程序测试、程序加载、功能测试、性能测试、电路连接方式等。

程序测试部分我们主要利用 ChipScope 相关功能，生成.cdc 文件，选择触发时钟和所要观察的变量。连接硬件后通过在 ChipScope 中观察各个步骤中所进行的变量赋值，即可判断该段程序是否功能正常。

调试中程序加载过程如下所示。点击"Implement top module"进行编译，之后点击"Generate Target PROM/ACE File"，先后双击"Boundary Scan"和"Initialize Chain"，初始化菊花链。双击" Create PROM File (PROM File Formatter) " 选项，设置 PROM File。弹出选择窗口，选择源.bit 文件，双击工程窗口左侧的" Generate File..."等待创建成功。工程文件夹中就会出现设置好的 .mcs 文件。双击"Boundary Scan" 按键，新窗口下单击工具栏的连接图标，弹出窗口，选择"yes"。之后会弹出文件选择窗口，该窗口用来选择烧录文件为对应的 device，第一个窗口选"Bypass"，第二个步选.bit 文件。最后左键选择 xc3s500e 芯片，点击"program"完成烧录。这样我们就完成了 FPGA 开发板上程序的烧录。

本系统电路连接方式为：由同轴线自信号源引出的方波信号，一端接 FPGA 板上 D0 号 I/O 口，一端接 GND 口。一根同轴线一端接用于调制的开发板上的 DA\_OUT 输出口，另一端接用于解调的开发板上的 AD\_IN 输入口，以完成调制信号的传输。由于解调后的信号通过 D3 号 I/O 输出，故此处将传输线铁夹一端分别通过两根杜邦线接 D3 口和 GND 口，另一端接示波器以观察解调后得到的信号。

分部分测试时，受实验室条件限制，我们用一块 EELAB 板完成分部分调试工作。首先将开发板 DA\_OUT 输出口通过同轴线连接到示波器上，以观察调制后的信号。此部分功能测试为观察调制部分是否可以正常通过 2ASK 方式进行调试，示波器上显示的调制波形是否正常，是否可以同基带信号完全对应，帧格式是否可以明显看出以及和有效信号相区别。功能测试为测量基带码元宽度及其步进，计算系统有效比特传输速率和符号速率，以及测量调制信号中正弦波形周期，计算载波频率，观察正弦波波形有无明显失真。调制模块测试通过后，即将同轴线两端分别连接 DA\_OUT 输出口和 AD\_IN 输入口，以进行解调模块测试。解调模块测试时将 D3 号 I/O 输出的解调后信号接到示波器上以进行显示。此

部分功能测试为观察解调部分是否可以正常对调制信号进行解调，以及解调得到的信号是否与输入的基带信号一致，并观察帧头部分是否也被正常解调出来，是否可以和有效信号进行明显的区分。性能测试为观察并测量解调后的码元宽度，并计算其相对于基带码元宽度的误差，以及系统传输的误码率。以上性能测试指标均可通过示波器上“Measure”功能测量得到。

最终测试时我们采用两块 EELAB 开发板，一块用于调制，另一块用于解调。同轴线一端接在用于调制的开发板的 DA\_OUT 输出口，另一端接在用于解调的开发板的 AD\_IN 输入口，以进行信号的传输。通过示波器，观察基带信号和解调后得到的解调信号，再一次比较两者最小码元宽度、步进、有效传输码速率、符号速率、码宽误差、误码率等数值，以及观察帧格式是否可以和有效信号明显区分。

## 5 测试结果

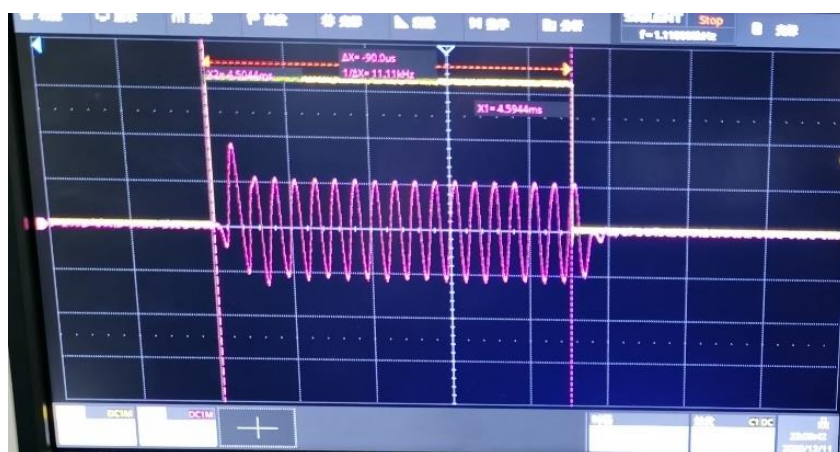


图43 调制信号

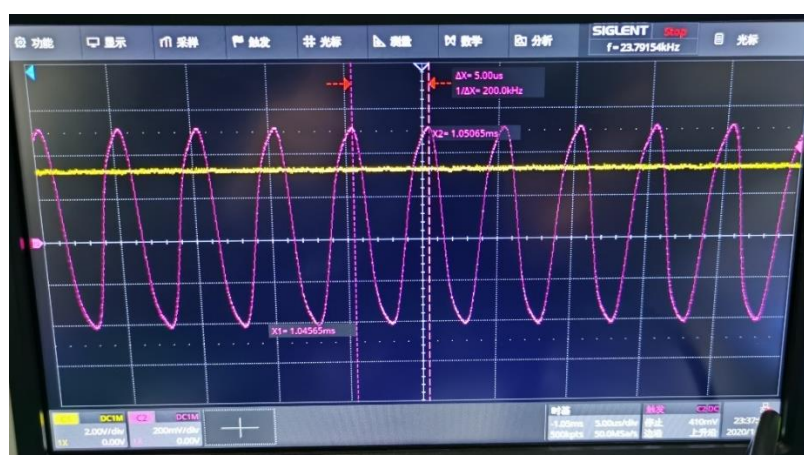


图44 测量载波周期



图45 调制信号码元宽度



图46 调制信号码元宽度



图47 帧格式示意

上图中，左侧为正常信号码元，右侧为帧头，可看出帧头码元宽度明显小于正常信号。



图48 调制信号波形



图49 解调后的信号波形

测试结果表明，系统调制模块可将由信号发生器输入的基带信号通过 2ASK 方式进行调制，调制信号波形满足要求。基带信号为 0 时，调制信号正弦波幅度为 0；基带信号为 1 时，调制信号正弦波幅度峰峰值为 1V。利用示波器“Measure”功能可以测量得到，作为载波的正弦波信号周期为 5  $\mu$ s，频率为 200kHz，正弦波波形无明显失真。基带信号码元宽度为 90  $\mu$ s，步进也为 90  $\mu$ s。由调制波形也可看到帧格式。帧格式即为每周期前出现的宽度为 65  $\mu$ s（小于正常码元宽度）的 10 信号。解调模块读取到帧头时，即开始一周期的有效信号的解调。有效码元传输速率则可由下式得到。

$$f = f_0 * \frac{N_{valid}}{N_{total}} \quad (3)$$

其中  $f$  为有效码元传输速率， $f_0$  为总码元传输速率，可由波形发生器读得，此处为 11.11kbps。 $N_{total}$  为帧头引领的一个周期内的总码元数，此处为 50。 $N_{valid}$  为该周期内除去

帧头有效的码元数目，由于帧头共占据了 3 个有效码元宽度，故此处为 47。经计算得有效码元传输速率为 10.44kbps/s。由于本系统采取 2ASK 的数字信号调制方式，故符号速率与有效比特传输速率相等，也为 10.44kbps。前测载波频率为 200kHz，大于了十倍符号速率，满足要求。

系统解调模块可将通过同轴线输入的调制信号进行解调，调制信号正弦波幅度为 0 时，信号解调为 0；调制信号正弦波幅度大于 0 时，信号解调为 1。调制与解调模块均能正常工作，且通过解调得到的信号与由信号源输入的基带信号相比，误码率为 0。基带信号码元宽度为 90  $\mu$  s，解调所得信号码元宽度为 92  $\mu$  s，计算得到码元宽度误差为 2.22%。帧格式在解调后的信号中也可看到，为码元宽度较正常码元短的 10 数据，可有效将帧头与有效信号分开。

我们将全系统整体进行测试，由一块 EELAB 板进行调制，一块 EELAB 板进行解调，观察由信号源输入的基带信号与解调出的信号。测试结果表明，本系统可有效完成信号的调制解调及传输工作，且各项指标均满足要求。由于受解调方式限制，码元边缘处的解调存在一定误差，故解调后得到的各个码元长度不严格相等，在示波器上显示则为码元边缘存在抖动现象。但解调后得到的码元宽度与基带信号码元宽度对比所得的码元宽度误差仍在 2%左右，小于系统要求的 10%误差，因此仍可较好地满足要求。

综上，本系统与数字通信相关的参数测量结果及结论为：基带码元宽度为 90  $\mu$  s，步进为 90  $\mu$  s，有效比特传输速率为 10.44kbps，符号速率与之相等，也为 10.44kbps。载波频率为 200kHz，大于了十倍符号速率。误码率为 0，码元宽度误差为 2.22%。本系统各项参数指标均满足要求。

表8 测试数据记录表

系统参数指标	实际测量值	要求数值
基带码元宽度	90 $\mu$ s	不大于 1ms
步进	90 $\mu$ s	不大于 0.1ms
有效比特传输速率	10.44 kbps	大于等于 10 kbps
载波频率	200kHz	大于等于 10 倍符号速率
符号速率	10.44 kbps	
码宽误差	2.22%	不大于 10%
误码率	0	0



## 6 遇到的问题与解决方法

### 6.1 软件设计过程

在统计 ASK 的 1 与 0 的过程中，根据查阅的资料进行“&”判断时，由于 ASK 的最低点也被默认为 0，因此一开始统计出来的波形在最低点时也会跑到中线那里形成断电，因此我们放弃了“&”的使用，改成了“?:”的选择方式，从而对这一问题予以了解决。

### 6.2 制作过程

在制作过程中，我们遇到了示波器上所显示波形只有频率约为 50Hz 的噪声波形，并不存在我们所希望看到的调制后的波形的问题。

经排查我们发现是因为所使用的用以连接开发板上的 DA\_OUT 输出端口和示波器的同轴线质量较差，导致信号传播过程中混入了较大噪声，掩盖住了本应显示的调制后的 2ASK 信号。解决方案为换用质量较好的同轴线连接开发板上的 DA\_OUT 输出端口和示波器，这样即可在示波器上看到正常的调制信号波形，无明显的噪声，解决了这一问题。

### 6.3 调试过程

在调试过程中遇到的难点问题有：解调后的波形误码率不为 0，多处基带信号为 1 处解调为了 0。

经分析我们认为这是代码解调部分阈值选择不当造成的。我们设置的一个采样周期内采样得到的大于 0 的次数可使此段信号判断为 1 的阈值过高，导致一些本应判断为 1 的片段判断为了 0。

我们解决这一测试过程中遇到的问题的方案是下调上述阈值，通过 ChipScope 观察由 AD 模块得到的数字化后的调制波形，观察噪声的上限值，并多次修改阈值进行尝试，以确定一个最佳的阈值。最终我们实现了解调后的波形误码率为 0，解决了这一问题。

## 7 结论

本次实验主要是利用 EELAB-DIGIEXP 完成 ASK 的调制与解调过程。ASK 过程即为移幅键控，又称为振幅键控，此过程中，当 1 出现时接通振幅为 A 的载波，0 出现时关断载波，相当于将原基带信号（脉冲列）频谱搬到了载波的两侧。

我们小组在完成此过程中，学习了 ISE 14.7 软件的使用方法，学习了利用 ChipScope 进行波形仿真查看，通过分频的相关原理进行了频率计算，了解了 SPI 通信协议中有关 SPI 时序部分的变换过程。小组成员在本学期的工作中共同配合、协同合作，逐步攻克了

Matlab 仿真、硬件仿真以及实物制作等诸多任务。从仿真到实际制作之间的跨度同样给予了我们启示，即理论可行的事情在实际工程中可能存在很多难以预料的难题，这些问题既包括代码书写上的小疏漏，也包括结构设计与电路搭建过程中未能避免的原理性错误，而最终，通过小组成员对于资料的广泛查阅、对于知识的认真学习，与对于问题的共同讨论，实物制作最终以优秀的成绩完成验收。尽管最终仍旧存在一些小问题，但就整个项目而言，我们的解决方法可能不是最优，但最后形成的结果已经非常贴近理论结果，而更重要的收获则是，我们在此次工程实践中学习到了团队协作的精神与工作方式，同时也培养了工程思维与实践能力。通过本次实践，小组成员收益颇丰，也立志在今后的学习与课余生活中将本学期所学习到的实践知识与工程精神贯彻到底。

## 参考文献

- [1] B. Williams. 电子滤波器设计[M]. 科学出版社, 2008.
- [2] 宋寿鹏. 数字滤波器设计及工程应用[M]. 江苏大学出版社, 2009.
- [3] 徐文波 田耘. Xilinx FPGA 开发实用教程（第 2 版）[M]. 清华大学出版社, 2012.
- [4] 赵瑄. 大数据背景下数字基带信号传输常用码型的专业分析与比较[J]. 营销界, 2020(47):78-79.
- [5] 通信原理[M]. 高等教育出版社 , 张会生, 2011
- [6] Soumya P. Dash, Ranjan K. Mallik, Saif K. Mohammed. Performance analysis of non-coherent PLC with multi-level ASK in impulsive noise environment. 2018, 12(7):816-823.

## 附录

电路图所在目录： SunQichu\_AllFiles\PIC

程序工程文件夹所在目录： SunQichu\_AllFiles\Final\_Program