**FIR低通滤波器设计报告**

1. **设计内容**

本设计是基于FPGA的一个FIR低通滤波器设计，要求使用Verilog语言编写滤波器模块，通过编译和综合，并通过Matlab和modelsim联合仿真验证设计结果。

1. **设计原理**

FIR滤波器响应（简称FIR）系统的单位脉冲响应为有限长序列，系统函数 在有限z平面上不存在极点，其运算结构中不存在反馈支路，即没有环路。如果的长度为N，则它的系统函数和差分方程一般具有如下形式：





根据差分方程直接画出FIR滤波器的结构，称为直接型结构。如图所示：

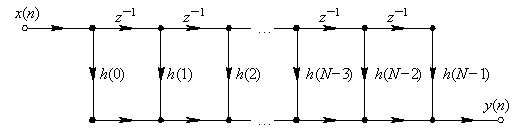


图2.1 FIR滤波器直接结构

FIR滤波器的特点：单位脉冲响应序列为有限个；可快速实现；可得到线性相位；滤波器阶数较高。对线性时不变系统保持线性相位的条件是：单位脉冲响应为偶对称或奇对称。即：



为设计线性滤波器，应保证h(n)为对称的。

1）若N为偶数，其线性相位FIR滤波器的对称结构流图：

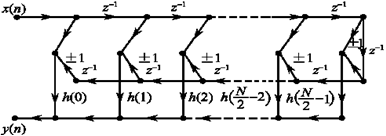


图2.2 若N为偶数线性相位FIR滤波器的对称结构流图

图中：“ +1 ” 对应偶对称情况，“ -1 ” 对应奇对称情况。当n为奇数时，最后一个支路断开。

2）若N为奇数，其线性相位FIR滤波器的对称结构流图：

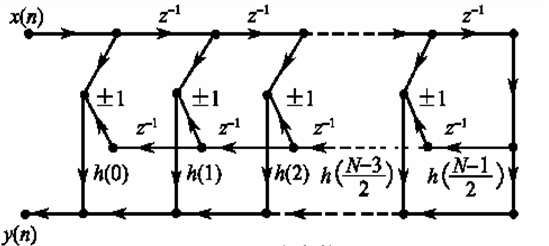


图2.3 N为奇数线性相位FIR滤波器的对称结构流图

在本设计中，我们采用线性FIR低通滤波器，所采用的阶数N=8，所以是偶对称的，估采取图2.2的结构，其中“±1“取“＋1”。

1. **设计思路**

要在FPGA上实现FIR滤波器，首先要确定滤波器的抽头系数。其系数的确定，我们可以通过两种办法来实现：第一种就是通过matlab编写FIR滤波器程序，然后直接导出抽头系数“h（n）”，另外一种办法就是使用matlab自带的FDATOOL简便地设计一个FIR滤波器，然后导出系数。考虑到要更直观地描述FIR滤波器的设计，我采用了第一种方法，用编写matlab代码的方式设计一个FIR低通滤波器。

设计好滤波器后，接着要用matlab产生一个高频率和低频率的信号，通过滤波器的滤波，能把高频的信号滤除，只剩下低频的信号，从而验证滤波器的性能。然后再把产生的输入信号进行采样，保存，量化，以供FPGA的FIR滤波器模块使用。

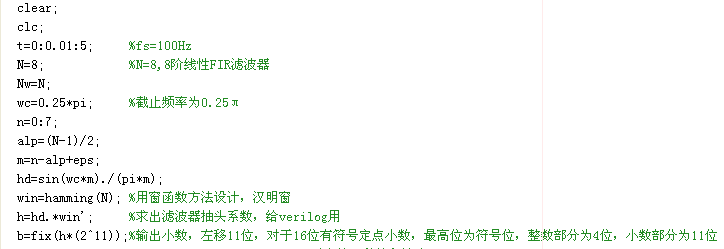
做好准备工作后，就可以在QUARTUSII里面编写Verilog代码了。由于是硬件描述语言，所以设计的思路很简单，就是通过把输入序列移位，然后首位对称相加再乘以抽头系数，然后把相乘结果再相加最后给输出。其中涉及的难点是FPGA对有符号小数的乘法处理部分。

在QUARTUSII编写好模块之后，就要用到modelsim来对设计进行仿真。对于仿真信号的输入，我们可以添加一个altera的romIP核来存放之前matlab产生的信号采样值，然后通过时钟的驱动一个一个输入到FIR滤波器模块，最后在modelsim中显示滤波器的输出结果。

验证的最后一步，就是要把modelsim输出的仿真结果用matlab进行绘图，和之前matlab程序的输出结果进行比较，从而验证滤波器的设计是否成功。

1. **设计过程**
2. **使用matlab设计FIR滤波器**

用matlab设计线性FIR滤波器，首先要确定其指标，在本设计里，我们规定滤波器的指标如下：阶数N=8，抽样频率fs=100Hz，截止频率为12.5Hz用窗函数设计FIR滤波器，窗函数选择汉明窗。根据此可以编得matlab代码如下：



可以求得滤波器抽头系数h（n）为：

h =

0.0028 0.0298 0.1259 0.2325 0.2325 0.1259 0.0298 0.0028

由于FPGA不支持浮点数的运算，所以我们采用定点数的格式来量化抽头系数。对于16位的输入数据，我们这里采用Q11的定点小数格式，即11位小数位，4位整数位，1位符号位。将抽头系数乘以2^11可得：

b =

5 60 257 476 476 257 60 5

这样我们就得到了8阶FIR滤波器的量化系数了。

确定好FIR滤波器的系数了，我们还需要用matlab产生仿真用的输入信号，在这里，我们采用了以下两组信号：

X1=0.01\*sin(2\*pi.\*t)+0.01\*sin(2\*pi\*40.\*t)，f1=0.02π，f2=0.8π（截止频率为0.25π）

X2=0.01\*sin(2\*pi.\*t)+0.01\*sin(2\*pi\*4.\*t)，f1=0.02π，f2=0.08π（截止频率为0.25π）

然后用这两组结果分别与滤波器系数进行卷积运算，可以得到滤波器后的结果，如下图所示：

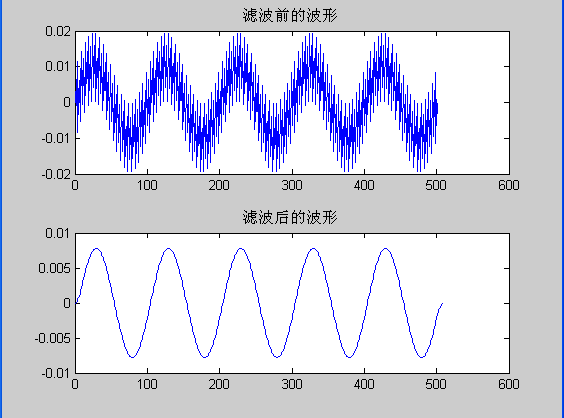


图4.1 当输入信号为x1时，滤波器的输入波形以及输出波形示意图

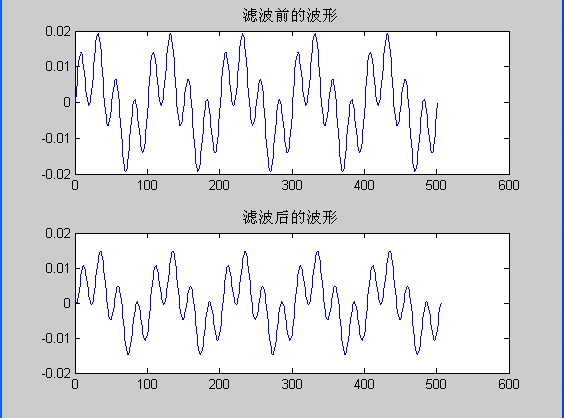
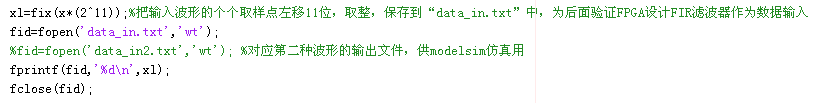


图4.2 当输入信号为x2时，滤波器的输入波形以及输出波形示意图

通过简单分析上述两图的结果，可以看出，当输入信号的频率大于0.25π时，信号得到很好的滤除，而当输入信号少于0.25π时，其信号会无失真地通过滤波器。

接下来，我们就先把这两种信号的采样值进行移位量化处理，然后保存到txt文件，供后续仿真使用，具体matlab代码如下：



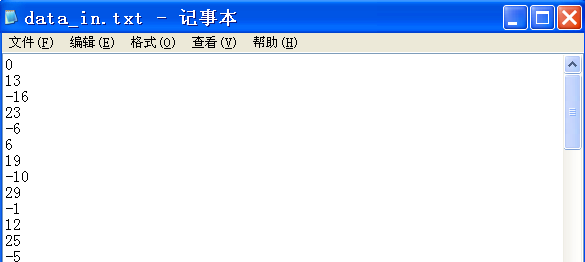


图4.3 matlba程序对x1波形进行采样，量化，然后保存到txt文件

1. **用Verilog编写滤波器模块**

由于Verilog是一种行为描述语言，所以用来描述FIR的结构是很方便的，其关键程序就是有符号小数的乘法处理模块。为了方便后续的仿真，整一个FIR滤波器程序可以分为三个部分：用来存放输入数据的rom模块，FIR滤波器主程序以及有符号小数的乘法模块。

**（1）Rom模块：**

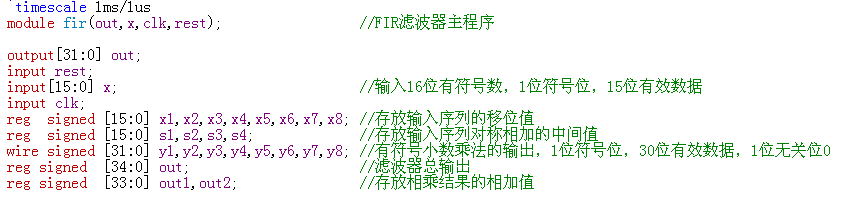
Rom模块使用QUARTUSII的IP核直接调用，然后把matlab采样的数据，直接保存到mif或hex文件，供rom调用。在本设计，一开始我们也是保存在mif文件中，但发现modelsim不支持mif文件的仿真，所以我们改为保存在hex文件中。

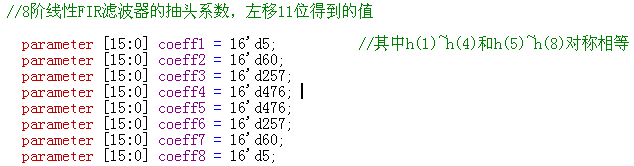


图4.3 hex文件存放的数据，节选

**（2）FIR滤波器主程序：**

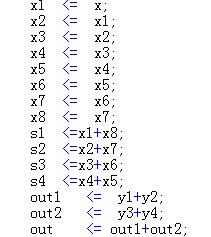
其主程序的变量定义如下：





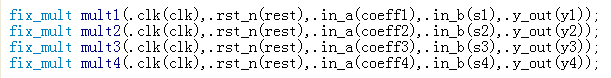
其中输入数据和滤波器的抽头系数，都是16位，由于输入序列最大值不超过40，所以相加结果仍然是16位。相乘的结果用32位表示。为了防止相乘再相加后数据的溢出，我们在这里把存放相乘再相加的结果扩大了2位，用到34位，然后最后的输出，在原来的基础上再扩大1位，35位的输出。在最后的输出中，可以把低位去掉，只保留若干高位，但这里只需要仿真出结果便可，所以没有进行截断处理，然而在实际工程中，由于数位的限制，还是要注意这点。

FIR主程序就是描述图2.2的FIR滤波器结构，其行为描述，具体可以用以下代码实现：



其中x1~x8通过前一个值对后一个值的赋值，可以实现移位操作；而s1~s4是用来存放首位对称相加的结果；然后y1~y4是乘法输出的结果，通过out1与out2的相加，最后输出到结果out。

其中乘法的运算，我们调用4次乘法模块（因为线性FIR滤波器是对称结构，所以可以只进行8/2=4次的乘法运算。



**（3）乘法模块**

对于有符号的小数运算，在FPGA里面，其实是和有符号整数的运算是一样的。只是定点数的小数点的位置，我们需要牢记，在最后输出结果的时候，要适当的进行移位。

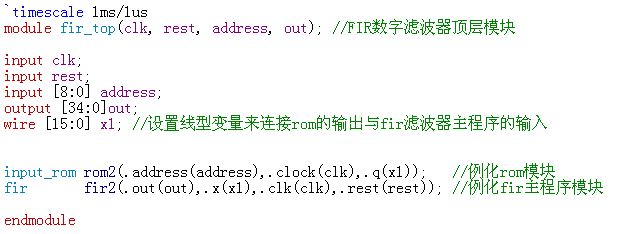
由于负数在Verilog中是以补码形式保存的，所以在输入16位有符号数的时候，在进行相乘之前，要把负数变为原码再相乘，而正数的补码就是其本身，所以不用转换。然后把符号位提出来，进行异或运算，得到输出的符号位，再把有效数位的原码进行相乘，最后得到1位符号位和30位相乘结果。为了补全32位，可以在最低位加上一个无关位0。最终输出如果是负数，还需要把它变成补码的形式输出。至此，一个有符号的小数乘法运算就完成了。

根据以上思路，我们可以写出出乘法模块的代码：



1. **综合设计模块**

为了把rom模块和FIR主程序连接起来，我们需要编写一个顶层文件。



1. **分析验证**

设计好以上模块后，可以进行编译综合了。最后的编译综合结果如下图所示：

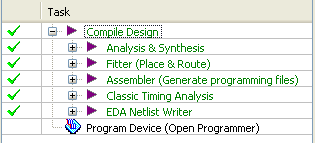
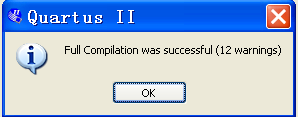


图5.1 编译结果

从上图可以看出，最后编译成功，程序设计没有语法错误。但具体要测试其工作是否正常，我们接下来，就需要用到modelsim对其进行仿真。

首先我们看看其RTL视图：

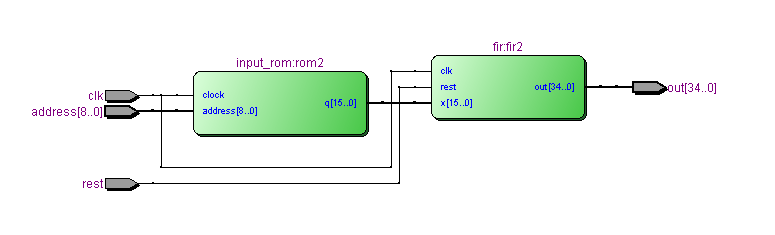


图5.2 FIR滤波器RTL视图

从上图的结构可以看出，我们要写testbench文件，就需要三个输入，一个输出。其中一个输入是时钟，一个复位信号，还有随时钟变化的地址输入和最后的输出。根据此可以写出测试文件：



由于最后输出的数是补码，为了能在modelsim的下方报告栏能显示出正常的十进制负数，我们定义了一个有符号寄存器out1，然后通过$display命令把结果按照时钟一个个输出到报告栏里。

仿真得到的波形和结果如下图所示：

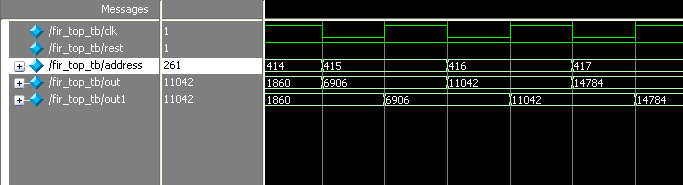


图5.3 modelsim仿真波形

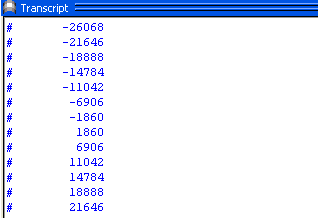


图5.4 modelsim输出结果显示

最后把报告栏里显示的结果，复制一部分到txt文件里，然后供matlab绘图用。

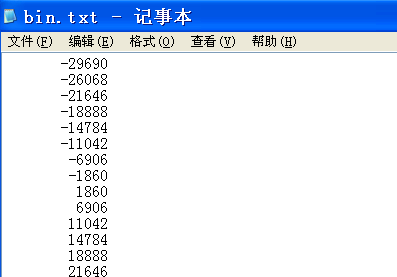
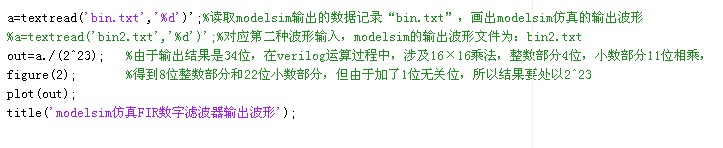


图5.5 把modelsim输出结果保存到txt中

最后执行以下代码，画出FIR滤波器输出的仿真波形：



由于我们没有对输出的35位数据进行截断处理，所以在最后的输出结果，我们要处以2^23。这是因为两个Q11的小数相乘，其小数位扩大了11位，变成了22位；然后加上1位的无关位，所以最后输出要处以2的23次方。

最后得出滤波器仿真的波形和matlab产生的结果进行比较：

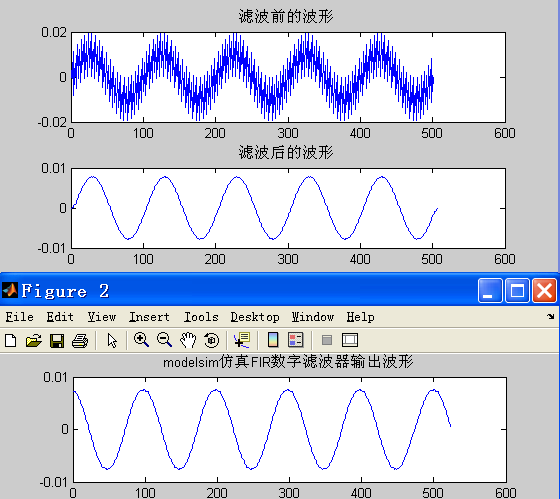


图5.6 当输入信号为x1时，modelsim输出的波形和matlab仿真的波形比较

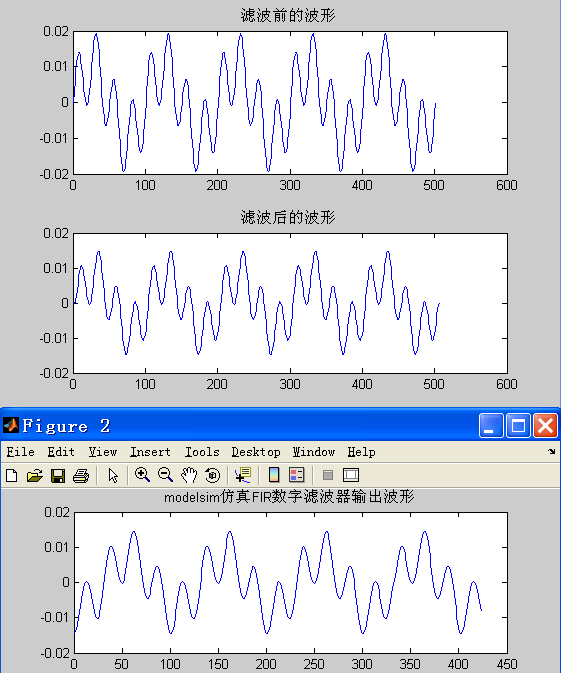


图5.7当输入信号为x2时，modelsim输出的波形和matlab仿真的波形比较

由于输出结果是随机选取，所以相位上会有偏差。但可以看得出，modelsim的输出波形是和matlab的输出波形一致的，可以确定滤波器正常工作，达到了预期的效果。

1. **设计总结**

通过本次FIR低通滤波器的设计，我对Verilog这门语言有了更深入的理解和认识，掌握了FPGA对有符号小数乘法的处理技巧。另外通过对设计进行modelsim和matlab的联合仿真，使我对这两个工具的操作水平有了进一步的提高。

刚开始做这个设计的时候，下了很多的源代码，参考了人家不少的程序，发现其设计的方法都不一样，但是这些程序都没有涉及到有符号小数的乘法处理问题。所以我决定参照别人的程序写一个有符号小数的乘法运算模块应用到FIR滤波器中，取得了很好的效果。

在使用modelsim对设计进行仿真的时候，由于这个设计使用了altera的romIP核，所以在编译和仿真的时候遇到了很多的问题。首先要仿真IP核，要添加220model.v和altera\_mf.v这两个文件到modelsim的project里面，然后还要下载一个“conver\_hex2ver.dll”的控件，到modelsim的win32aloem文件夹里，再在modelsim.ini中的List of dynamic…下面添加上：

“Veriuser = C:\altera\10.0\modelsim\_ase\win32aloem\convert\_hex2ver.dll”

这一行，才能够让rom正常读出hex文件的内容，从而实现对系统的正常仿真。虽然过程很曲折，但收获很大，这是我首次对这样一个完整的模块进行仿真，能解决这些问题，为今后的设计和应用积累了大量的经验。

通过分析这次设计，其实也存在不足的地方。在FIR系统模块里，其实我可以加上一个COUNT模块，用来产生地址信号，这样用modelsim仿真的时候只需要时钟和复位的输入即可，可以使系统看上去更完整。但由于本设计是基于modelsim仿真为目的，所以就没添加上去，而是在测试模块里补充；另外，对输出数据的数位处理，还不够恰当。输出的35位数据，其实可以通过截断，变成更低数位的输出。当然对于仿真的结果是没影响，但如果应用到实际的工程里，由于实际数位的限制，还是需要进行截断处理的。

总的来说，本次设计还是很成功的，希望以后能继续深入学习，积累更多的经验。