**实验四 实时 FIR 滤波处理**

**王亭午**

**无210**

**2012011018**

1. **实验目的**

**1.1 学习实时数字信号处理系统中串口数据的输入/出缓存方法、系统实时性分析方法**

**1.2学习使用二维 DMA 的双自动缓存设置和立体声 FIR 滤波程序设计**

1. **实验材料**

**2.1 集成开发和调试系统软件版本 VisualDSP + + 4.5**

**2.2 评估板 ADSP − BF533 EZ − kit**

**2.3 实验用源程序**

(1) Talk\_Through\_533\_I2S \_FIR\_2MAC.rar (2) 第三次实验使用的 Talk\_Through\_533\_I2S.rar (3)自己编写的几种 FIR 滤波程序

**2.4 实验用音乐序列**

本次实验提供的噪声文件 noise.wav 和双音多频信号音 DTMF\_12345 6.wav

**2.5 实验用配件**

(1) 耳机 (2) 音乐播放器 (3) 双 3.5mm 立体声插头音频线

**2.6 阅读参考内容**

(1) 课程课件第五讲——《实时数字信号处理系统》(2) 第三次实验指导书及其附件 (3) VisualDSP + + 4.5 IDDE 帮助系统

1. **实验内容**

**3.1 任务一：执行并理解实时立体声 FIR 滤波例程**

**3.1.1实验步骤**

1. 正确建立实验环境(参看第一次和第三次实验指导书的相关部分)

a) 连接评估板和 PC 、电源、音频输入/出等接口 b)启动 IDDE VDSP++4.5 环境,新建一个工程 TalkThrough\_I2S\_FIR\_2MAC\_BF533 ,添加对应 BF533 的源文件,设置工程选项( project options ),选择对应评估板的 session。[注意] 请在 project options 的 project 配置窗口,选择自动检测处理器芯片版本号

(2) 在执行 Build project 成功后,直接运行程序, 播放随实验指导书给定的 noise.wav文件,用耳机可以听到滤波处理后的声音。

(3) 用View/Debug Windows/plot 直接观察输入/出 RxBUF/TxBUF 和 RxBuffer/TxBuffer 波形,观察滤波结果。

**3.1.2 实验问题**

**问题 1.1 如果不选择external SDRAM会出现什么问题?为什么?**

**答案：**这个时候出现了错误，如下：

|  |
| --- |
| **[Error li1040] "C:\Program Files (x86)\Analog Devices\VisualDSP 4.5\Blackfin\ldf\ADSP-BF561.ldf":507**  **Out of memory in output section 'bsz\_L1\_data\_b' in processor 'p0'**  **Total of 0xa80b0 word(s) were not mapped.**  **For more details, see 'linker\_log.xml' in the output directory.** |

可见这个时候因为没有指定所需要的数据长度，我们的有限硬件空间溢出了，无法分配。

**问题 1.2 你所观察到的 RxBUF/TxBUF 和 RxBuffer/TxBuffer 分别是什么?各保存在哪里?**

**答案：**我们组得到的结果图如下（为了显示漂亮我进行了反色）。在其中，RxBUF 表示接收整体波形段数据缓存器，TxBUF 表示发送的当前整体波形段数据缓存器。而对应了两个RxBuffer，TxBuffer 表示发送或者接受的一个 pingpong的数据。通过频域效果图，我们可以知道这是一个低通滤波器。

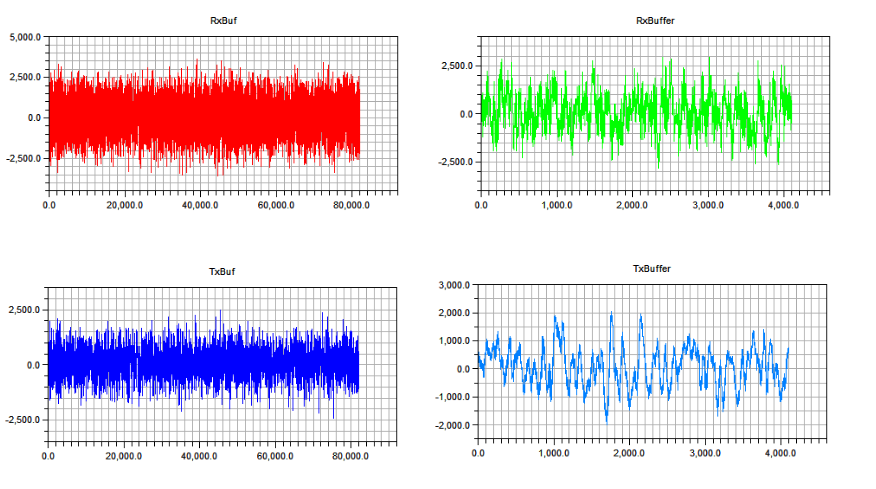


图1，RxBUF/TxBUF 和 RxBuffer/TxBuffer 的波形

通过查找，我们组发现：0x1004018 0x102c018 0x1000018 0x1002018分别为我们的RxBUF TxBUF RxBuffer TxBuffer的地址。通过查找 BF533 的 memory map 确定他们的位置，这四个地址均位于external memory中。

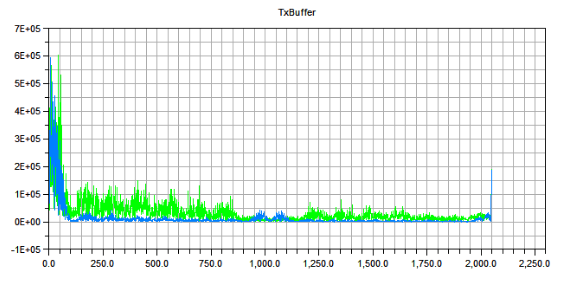


图2，频域效果图

**问题 1.3 使用 windows 系统自带的录音软件,配合实验盒中配备的 3.5mm 立体声音频线采录评估板输出,并保存为.wav文件,对比分析评估板的输入/出信号。**

**答案：**这里我们为了节省时间，交给了同组的桑煜同学使用matlab处理。结果图如下：

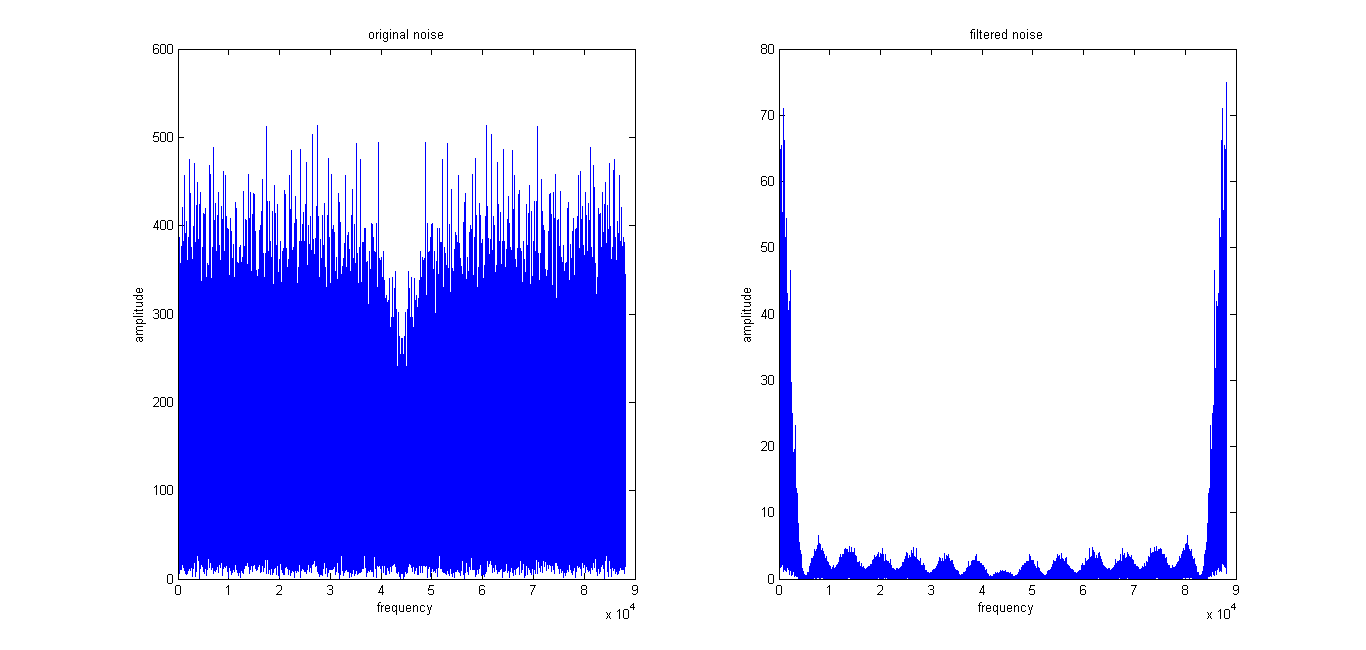


图3，输入输出信号结果图

从上面的图，我们可以看出：原文件的音频信号为噪声，各个频率分量的幅度都基本相同，趋近于白噪声。输出结果通过了一个低通的滤波器，不过因为滤波器的原因有许多旁瓣。用耳机听滤波处理后的声音噪声的幅度变小,声音变小，高频分量被滤除，声音不再刺耳。

**3.2 任务二：实时性分析**

根据 autobuffering 的原理,缓存器填满发送中断,执行该数据块的处理。如果数据处理复杂度太高,在一个数据块的时钟周期内无法完成,就会出现数据错误。利用 VDSP ++ 中Cycles 寄存器,可以统计程序执行的时钟周期,依据此数据,可以估算算法的实时性和autobuffer 的合适长度。

**3.2.1实验步骤**

(1) 实际测量时钟开销

a) 在ISR.c程序中调用process\_data.c语句设置一个断点,打开 Register\core\Cycles 窗口。b) 运行程序,停在断点处记录 Cycles 寄存器中的数值 c) 多次重复上一步骤( 3 次左右就可以) d) 计算两次之间的 Cycles 计数差,再求平均值,就是中断服务子程序的时间开销估计值

**答案：**我们得到的结果图如下：

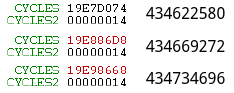


图4，时间周期结果图

经过同组同学的计算，我们得到，平均的时间cycles为631137。接下来的计算是这样的。我们已经知道数据的采样率为 fs = 48kHz，同时autobuffer 的长度为1024 × 2因此每个数据块的时间为Δt1=1024\*2/48e3=42.7ms。接下来计算总的cycle开销，计算如下：C1=41, C2= Ni/2\*(4+2(Nc/2-1))=9216。因此总的开销C=C1+C2=9257。最后考虑到BF561的主频率为600MHz。

最后我们得到处理一个数据块需要用时：Δt2=9257/600e6=**15.4283μs**。这个时间是远远的小于Δt1的，所以程序是没有问题的。

**3.3 任务三 分析本实验例程修改的目的**

本次实验的例程是在 fir. asm 和实验三的 Talk\_Through\_533\_I2S 的基础上改写而成。主要是针对实时处理,在 DMA 初始化程序中设置了双缓存器,针对立体声处理,在 fir. asm 中增加了一个参数 step(step = 2) ,针对 16 − bit 处理,AD1836 的控制寄存器和 SPORT 的接收和发送寄存器的控制字配置都要作相应修改。

**3.3.1 实验步骤**

1. 对比 Talk\_Through\_533\_I2S 和 Talk\_Through\_533\_I2S \_FIR\_2MAC 两个工程文件,找到相应修改的语句,分析其改动的理由。 a) 初始化程序对比分析。 b) 中断服务子程序的对比分析。Talk\_Through\_533\_I2S 的中断服务子程序中,在调用 Process\_data(∙) 之前/后使用数据搬移指令,将输入/出缓存器中的数据复制到变量(数据)存储空间:

**答案：**初始化的改动在下面分析，中断子程序的对比为：

|  |
| --- |
| **// confirm interrupt handling**  \*pDMA2\_0\_IRQ\_STATUS = 0x0001;  **// call function that contains user code**  Process\_Data(); |

而原来的程序为：

|  |
| --- |
| **// confirm interrupt handling**  \*pDMA1\_IRQ\_STATUS = 0x0001;  **// copy input data from dma input buffer into variables**  iChannel0LeftIn = iRxBuffer1[INTERNAL\_ADC\_L0];  iChannel0RightIn = iRxBuffer1[INTERNAL\_ADC\_R0];  iChannel1LeftIn = iRxBuffer1[INTERNAL\_ADC\_L1];  iChannel1RightIn = iRxBuffer1[INTERNAL\_ADC\_R1];  **// call function that contains user code**  Process\_Data();  **// copy processed data from variables into dma output buffer**  iTxBuffer1[INTERNAL\_DAC\_L0] = iChannel0LeftOut;  iTxBuffer1[INTERNAL\_DAC\_R0] = iChannel0RightOut;  iTxBuffer1[INTERNAL\_DAC\_L1] = iChannel1LeftOut;  iTxBuffer1[INTERNAL\_DAC\_R1] = iChannel1RightOut; |

**3.3.2 实验问题**

**问题 3.1 根据你对信号通路和数据传递模式的理解,你认为应该修改那几个初始化函数?阅读例程,确认你的理解。**

**答案：**主要的更改如下

|  |
| --- |
| \*pSPORT0\_RCR1 = RFSR | RCKFE;  \*pSPORT0\_RCR2 = SLEN\_16 | RSFSE;  **// Sport0 transmit configuration**  **// External CLK, External Frame sync, MSB first, Active Low**  **// 16-bit data, Secondary side disabled, Stereo frame sync enabl**e  \*pSPORT0\_TCR1 = TFSR | TCKFE;  \*pSPORT0\_TCR2 = SLEN\_16 | TSFSE; |

以及：

|  |
| --- |
| **// Configure DMA2 channel0**  **// Ping-Pong Buffer, the first half of RxBuffer is ping, the second half is pong**  **// 16-bit transfer, Interrupt on inner loop completion, 2 dimensional, Autobuffer mode**  \*pDMA2\_0\_CONFIG = WNR | WDSIZE\_16 | DI\_EN | DI\_SEL| DMA2D | FLOW\_1;  **// Start address of data buffer**  \*pDMA2\_0\_START\_ADDR = (void \*)RxBuffer;  **// DMA inner loop count**  \*pDMA2\_0\_X\_COUNT = 2\*FRAMESIZE;  **// Inner loop address increment**  \*pDMA2\_0\_X\_MODIFY = 2;  **// DMA outer loop count**  \*pDMA2\_0\_Y\_COUNT = 2;  **// Inner outer address increment**  \*pDMA2\_0\_Y\_MODIFY = 2; |

在上面的代码中，主要是启用了2D DMA，所以做了相应的修改，修改了bit为16。

**问题 3.2 在 Talk\_Through\_533\_I2S\_FIR\_2MAC 的中断服务子程序中有这样的操作吗?这是为什么?**

**答案：**没有，原因很简单。因为在本程序中,数据传递是通过指针直接读写memcopy完成的。这是一个优化程序的小trick。

**问题 3.3 process\_data. c 调用 fir. asm 新增加的参数 step 有什么意义?程序中如何使用它?**

**答案：**因为我们输入的音频信号为双声道左右声道相互交替进入输入流。所以在fir.asm中step是指明了这一点。当我们希望从输入流中逻辑上的位置乘以步长step就是它在物理中实际位置。

**3.4 任务四 实现自己设计的滤波器**

利用本实验提供的例程框架,将自己设计或已有的滤波器设计结果(滤波器系数)写成头文

件的形式,替换 fir\_coeff. h 文件,就可以实现自选滤波器的实时滤波处理。

**3.4.1 实验步骤**

(1) 设计并生成滤波器系数文件,注意数据类型转换。设计一个 FIR 低通滤波器,并将滤波器系数转换成有符号 16 − bit 定点小数形式,并参考fir\_coeff.h 的格式编写滤波器系数文件 myfircoeff.h。[建议] 使用 Matlab 的滤波器设计分析工具 fdatool ,用 C−header 形式导出滤波器系数文件，注意数据类型的选择和数组长度定义。

(2) 用 myfircoeff. h 替换例程中的 fir\_coeff.h 文件。(3) 重复任务一的步骤,用 noise. wav 文件验证滤波器设计效果。[提示] 可以利用采录的信号进行滤波器设计效果分析

**3.4.2 实验问题**

**问题 4.1 为什么要进行数据类型转换?这是必须的吗?**

**答案：**因为我们dsp中使用的frac16，所以我们必须进行数据转换。

**问题 4.2 请标明你设计的低通滤波器的设计指标,如通带和阻带截至频率等。**

**答案：**这里我们使用matlab进行了分析，如图：

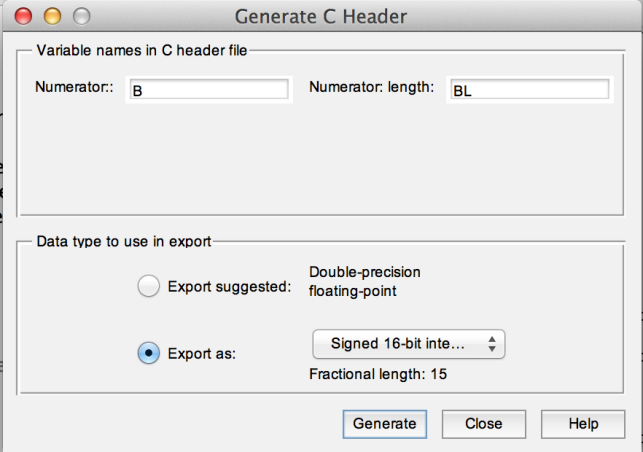
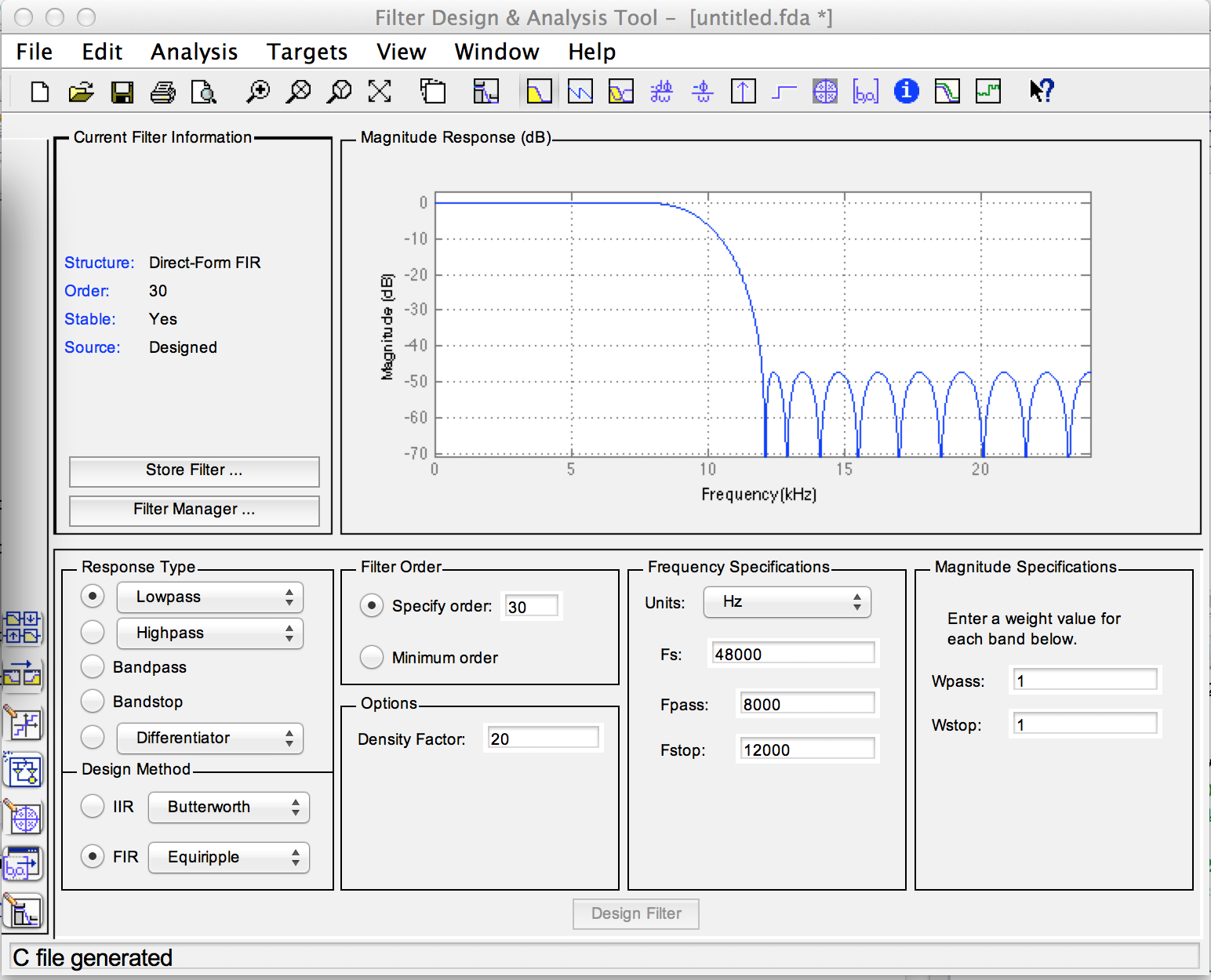


图5，滤波器设计图

其中采样率 48kHz 、截止频率 8kHz 、过渡带 8kHz~12kHz 。将滤波器系数通过 C − head 的形式进行导出。我们进行同样的滤波操作，发现效果也是非常不错的。

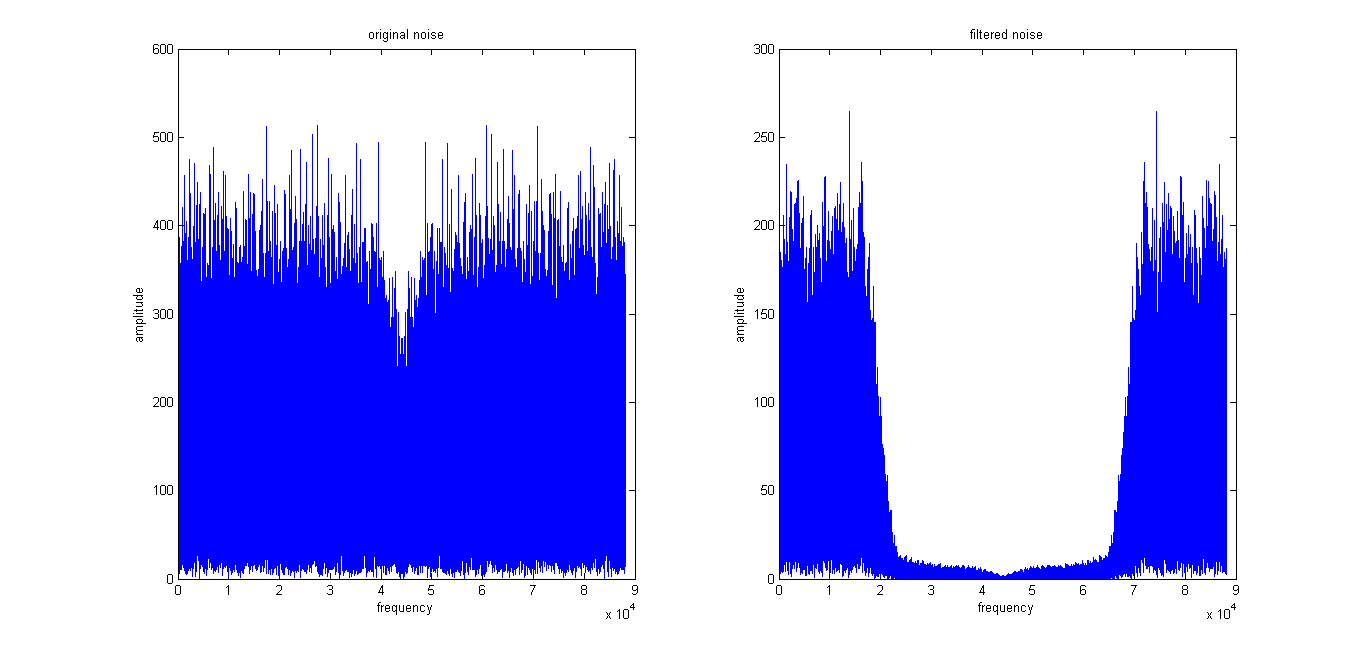


图6，自己设计滤波器效果图

头文件生成如下：

|  |
| --- |
| const int BL = 31;  const int16\_T B[31] = {  74, -94, -208, -13, 354, 251, -443, -713, 260,  1345, 464, -2012, -2318, 2525, 10009, 13667, 10009, 2525,  -2318, -2012, 464, 1345, 260, -713, -443, 251, 354,  -13, -208, -94, 74  }; |

**3.5 任务五 实现自己编写的实时 FIR 滤波系统**

利用本实验提供的例程框架,将自己设计或已有的滤波器设计结果(滤波器系数)写成头文

件的形式,替换 fir\_coeff. h 文件,就可以实现自选滤波器的实时滤波处理。

**3.5.1 实验问题**

**问题5.1 用实验四中自己写的 C 语言 FIR 滤波程序 myfir.c 代替例程中的 FIR.asm ,对比调用 C 程序和调用汇编程序这两种运行时间开销( Cycles )。**

**答案：**结果图如下：

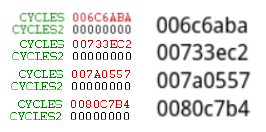


图7，C程序cycle图

根据上面的十进制时间，我们可以得到C=444840。

**问题5.2 用自己的汇编语言 FIR 滤波程序 myfir\_Smac\_asm.asm 和 myfir\_Dmac\_asm.asm代替例程中的 fir.asm ,对比这两个子程序运行时间开销( Cycles )。**

**答案：**同理我们得到几个cycle图，对于myfir\_Smac\_asm.asm，我们有：

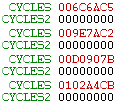


图8，汇编程序cycle图

得到的cycle数为C=3281751。

对于myfir\_Dmac\_asm.asm，我们有：

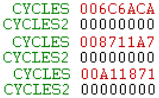


图9，汇编程序cycle图

得到的cycle数为C=1746653。通过上面的计算，我们可以知道，综合以上可知,运行的时间开销的大小关系为：单MAC开销大于双MAC开销大于原始asm开销。

1. **实验总结**

**4.1 总结 autobuffering 和双缓存器的作用,以及为满足实时性必须满足的条件。**

**答案：**从课件里，可以知道：autobuffering 使处理器的 sport 端口直接将接收到的数据存放到存储器中,而不需要产生中断，这是它最大的有点。在数据量填满 buffer 时,再产生中断信号,由 DMA 通道取走数据进行运算，这样可以节省很多时间。

而双缓存器，则是可以在一个 buffer 读取数据的同时,可以在另一个 buffer 处理数据。这样并行的方式提高了程序运行的效率。在我们的DSP实时要求下，这个更加重要。

**4.2 估算利用评估板实时处理 FIR 滤波的滤波器阶数的极限值,即该评估板可以实现最高多少阶的实时 FIR 滤波?**

**答案：**这个我们可以套用公式，我们的561板子的最大频率为600MHz，于是得到：C=Δt1\*f=25602000个cycle，于是Nc=2(C-C1)/Ni-2=50001。

**4.3 总结实验中的问题和解决办法**

**答案：**本次实验是和桑煜同学，杨思怡同学一起完成的，个人感觉是比较顺利的。中间唯一一个出现了卡顿的地方是因为板子有问题，而不是我们的问题。我们使用的那个板子出现了一组输入输出耳机接口有问题的情况，我们更换了我们板子，调整了程序，调整了驱动，最后发现问题来自于我们的开发板。于是我们使用了另外的一对输入输出口，问题得到了完美的解决。有一个小问题来自于我们当时找不到我们的BUFFER的大小。这个数据并没有在实验报告中指出来，因此我们一开始不太清楚该怎么弄。最后我们在头文件中才终于找到了数据的长度，在这里我们的反应有点迟钝了。

本次实验很多问题我们都是三个人合作完成的，因此还有一些是队友出现的问题，这里一并整理。在matlab设计的时候，我们需要考虑到采样率的不同，需要进行resample，然后才能开始进行频域的对比。

**4.4 本次实验体会与建议**

**答案：**最后一次实验非常完美的收场了。非常感谢这期间助教老师和老师对我们的指导和帮助。本次实验是合作实验，我发现合作的话能够极大的节省时间，前几次实验都是自己一个人单干，耗费了很长的时间，最后才发现，有点可惜。其次是感觉课程确实是一个非常完整的体系，围绕着几个核心问题，每次实验都对一个方面进行讲解，循序渐进，最后一次课完美收官。通过这几个实验的学习，我们成功的完成了对整个dsp能力的构建，感觉非常高兴。最后，作为一个即将毕业离开清华的学生，突然有种很悲伤的感觉，这样的课不会再有，这样的生活也只有最后一年。感谢清华对我们的教导。