Example 4.1: FIR Filter Implementation: Bandstop and Bandpass (FIR)

Figure 4.5 shows a listing of the C source program `FIR.c`, which implements an FIR filter. It is a generic FIR program, since the coefficient file included, `bs2700.cof` (Figure 4.6), specifies the filter's characteristics. This coefficient file, which contains 89 coefficients, represents an FIR bandstop (notch) filter centered at 2700 Hz. The number of coefficients N is defined in the coefficient file. This filter was designed using MATLAB's graphical user interface (GUI) filter designer SPTool, described in Appendix D [50]. Figure 4.7 shows the filter's characteristics (MATLAB's order of 88 corresponds to 89 coefficients). MATLAB's FDATool can be used in the place of SPTool (see Appendix D).

```
//Fir.c  FIR filter. Include coefficient file with length N

#include "bs2700.cof"                    //coefficient file
#include "dsk6713_aic23.h"               //codec-dsk support file
Uint32 fs=DSK6713_AIC23_FREQ_8KHZ;       //set sampling rate
int yn = 0;                              //initialize filter's output
short dly[N];                            //delay samples

interrupt void c_int11()                 //ISR
{
      short i;

      dly[0]=input_sample();             //input newest sample
      yn = 0;                            //initialize filter's output
      for (i = 0; i< N; i++)
         yn += (h[i] * dly[i]);          //y(n) += h(i)* x(n-i)
      for (i = N-1; i > 0; i--)          //starting @ end of buffer
         dly[i] = dly[i-1];              //update delays with data move
      output_sample(yn >> 15);           //scale output filter sample
      return;
}

void main()
   {
      comm_intr();                       //init DSK, codec, McBSP
      while(1);                          //infinite loop
   }
```

FIGURE 4.5. Generic FIR filter program (`FIR.c`).

```
//bs2700.cof FIR bandstop coefficients designed with MATLAB

#define N 89                             //number of coefficients

short h[N]= {-14,23,-9,-6,0,8,16,-58,50,44,-147,119,67,-245,
             200,72,-312,257,53,-299,239,20,-165,88,0,105,
            -236,33,490,-740,158,932,-1380,392,1348,-2070,
             724,1650,-2690,1104,1776,-3122,1458,1704,29491,
             1704,1458,-3122,1776,1104,-2690,1650,724,-2070,
             1348,392,-1380,932,158,-740,490,33,-236,105,0,
             88,-165,20,239,-299,53,257,-312,72,200,-245,67,
             119,-147,44,50,-58,16,8,0,-6,-9,23,-14};
```

FIGURE 4.6. Coefficients for a FIR bandstop filter (`bs2700.cof`).

A buffer `dly[N]` is created for the delay samples. The newest input sample, $x(n)$, is acquired through `dly[0]` and stored at the beginning of the buffer. The coefficients are stored in another buffer, $h[N]$, with $h[0]$ at the beginning of the coefficients' buffer. The samples and coefficients are then arranged in their respective buffer, as shown in Table 4.1.

Two "for" loops are used within the interrupt service routine (we will also implement an FIR filter using one loop). The first loop implements the convolution equa-

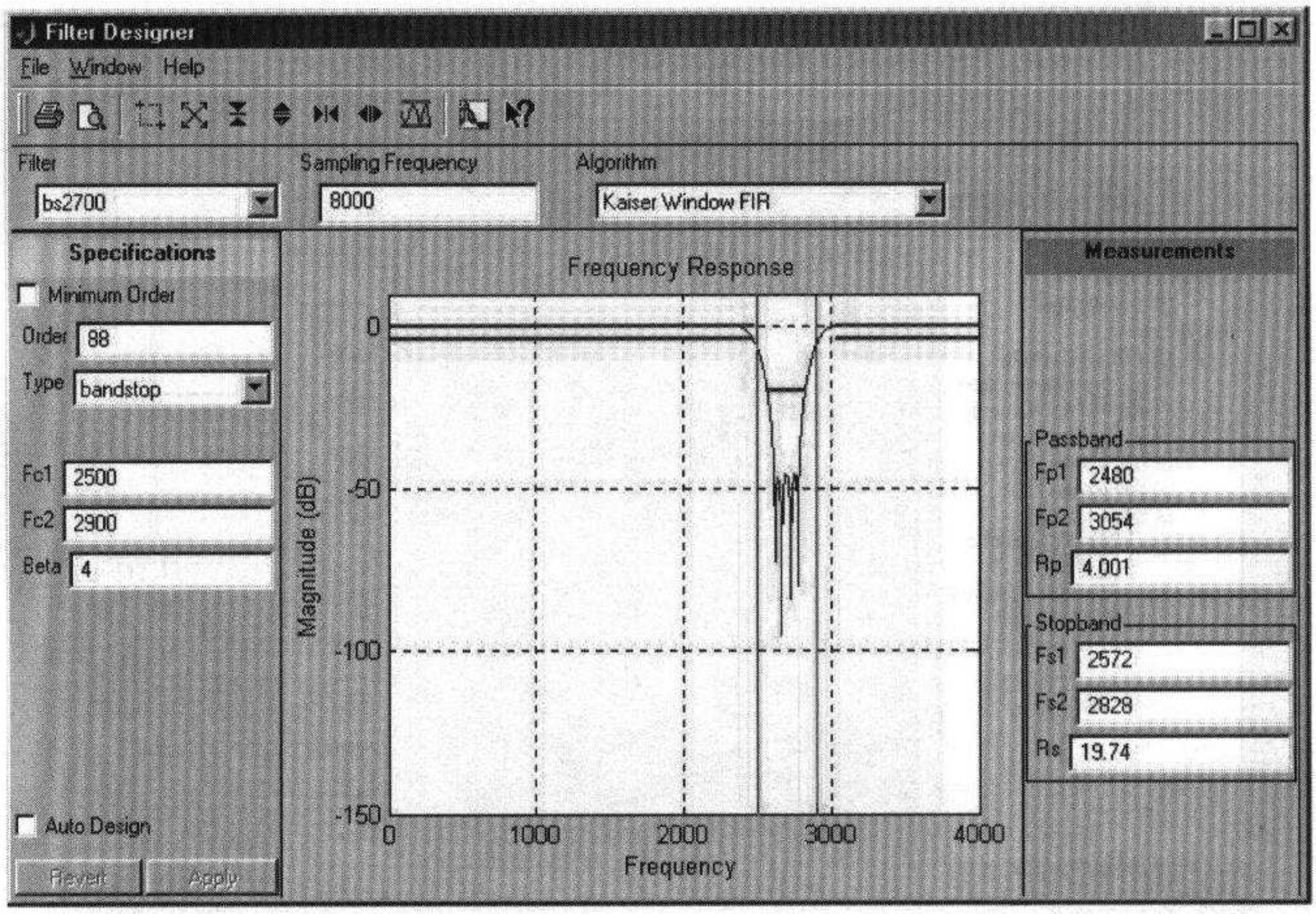

FIGURE 4.7. MATLAB's filter designer SPTool, showing the characteristics of a FIR band-stop filter centered at 2700 Hz.

tion with N coefficients and N delay samples for a specific time n. At time n the output is

$$y(n) = h(0)x(n) + h(1)x(n-1) + \cdots + h(N-1)x(n-(N-1))$$

The delay samples are then updated within the second loop to be used for calculating $y(n)$ at time $n+1$, or $y(n+1)$. The newly acquired input sample always resides at the beginning of the samples buffer (in this example). The memory location that contained the sample $x(n)$ now contains the newly acquired sample $x(n+1)$. The output $y(n+1)$ at time $n+1$ is then calculated. This scheme uses a data move to update the delay samples.

Example 4.7 illustrates how various memory organizations can be used for both the delay samples and the filter coefficients, as well as for updating the delay samples within the same loop as the convolution equation. We also illustrate the use of a circular buffer with a pointer to update the delay samples in lieu of moving the data in memory. The output is scaled (right-shifted by 15) before it is sent to the codec's DAC. This allows for a fixed-point implementation as well.

Bandstop, Centered at 2700 Hz (`bs2700.cof`)

Build and run this project as **`FIR`**. Input a sinusoidal signal and vary the input frequency slightly below and above 2700 Hz. Verify that the output is a minimum at 2700 Hz.

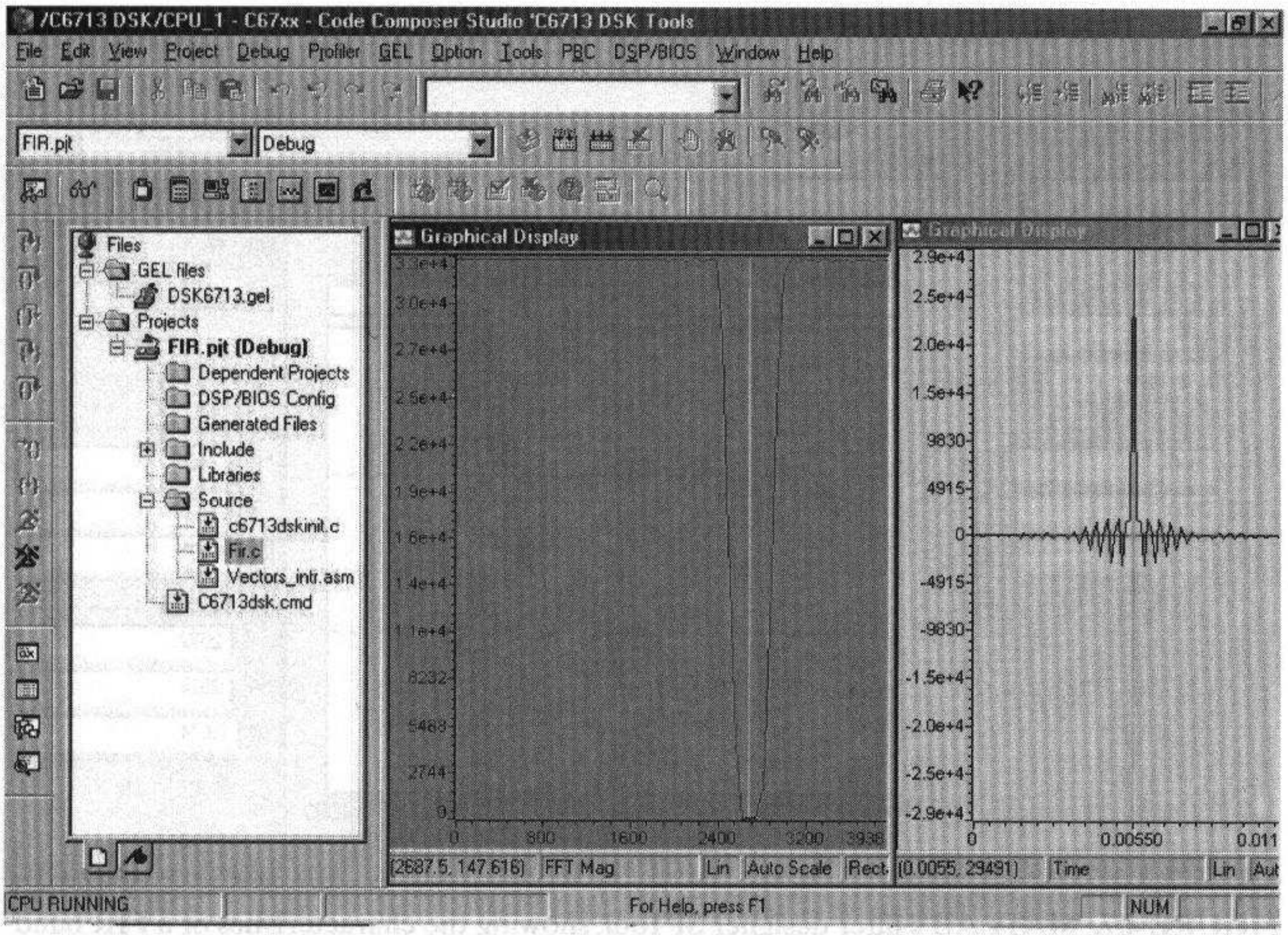

FIGURE 4.8. CCS plots displaying the FFT magnitude of the bandstop filter's coefficients and its impulse response.

Figure 4.8 shows a plot of CCS project windows. It shows the FFT magnitude of the filter's coefficients h (see Example 1.2, with a starting address of h) using a 128-point FFT. The characteristics of the FIR bandstop filter, centered at 2700 Hz, are displayed. Figure 4.8 also shows a CCS time-domain plot, or the impulse response of the filter.

With noise as input, the output frequency response of the bandstop filter can also be verified. The pseudorandom noise sequence developed in Chapter 2, or another noise source, can be used as input to the FIR filter, as illustrated later. Figure 4.9 shows a plot of the frequency response of the filter with a notch at 2700 Hz implemented in real time. This plot is obtained using an Hewlett-Packard (HP) 3561A dynamic signal analyzer with an input noise source from the analyzer. The roll-off at approximately 3850 Hz is due to the antialiasing lowpass filter on the codec.

Bandpass, Centered at 1750 Hz (`bp1750.cof`)

Within CCS, edit the program `FIR.c` to include the coefficient file *`bp1750.cof`* in lieu of *`bs2700.cof`*. The file *`bp1750.cof`* represents an FIR bandpass filter (81 coefficients) centered at 1750 Hz, as shown in Figure 4.10. This filter was designed

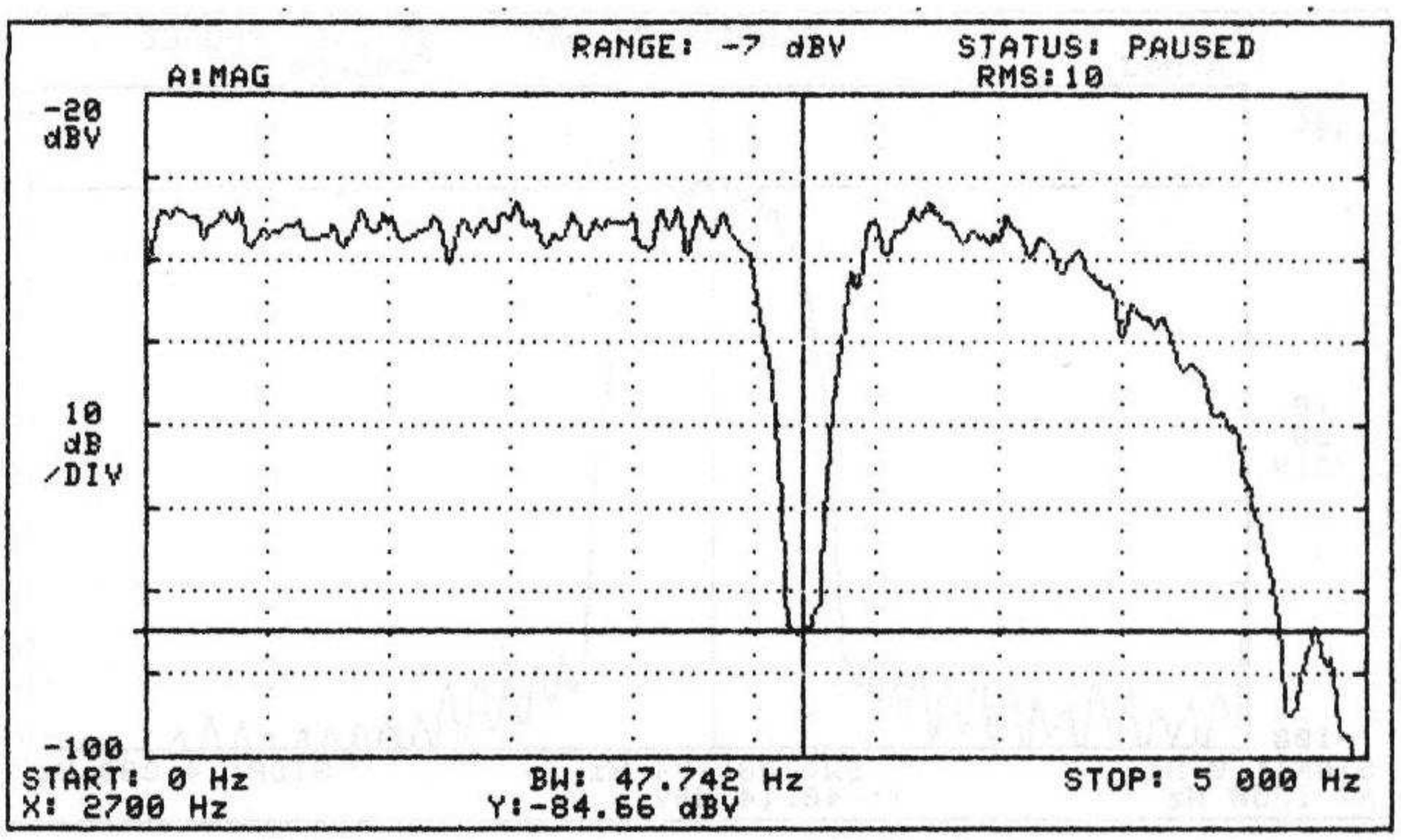

FIGURE 4.9. Output frequency response of a FIR bandstop filter centered at 2700 Hz, obtained with a signal analyzer.

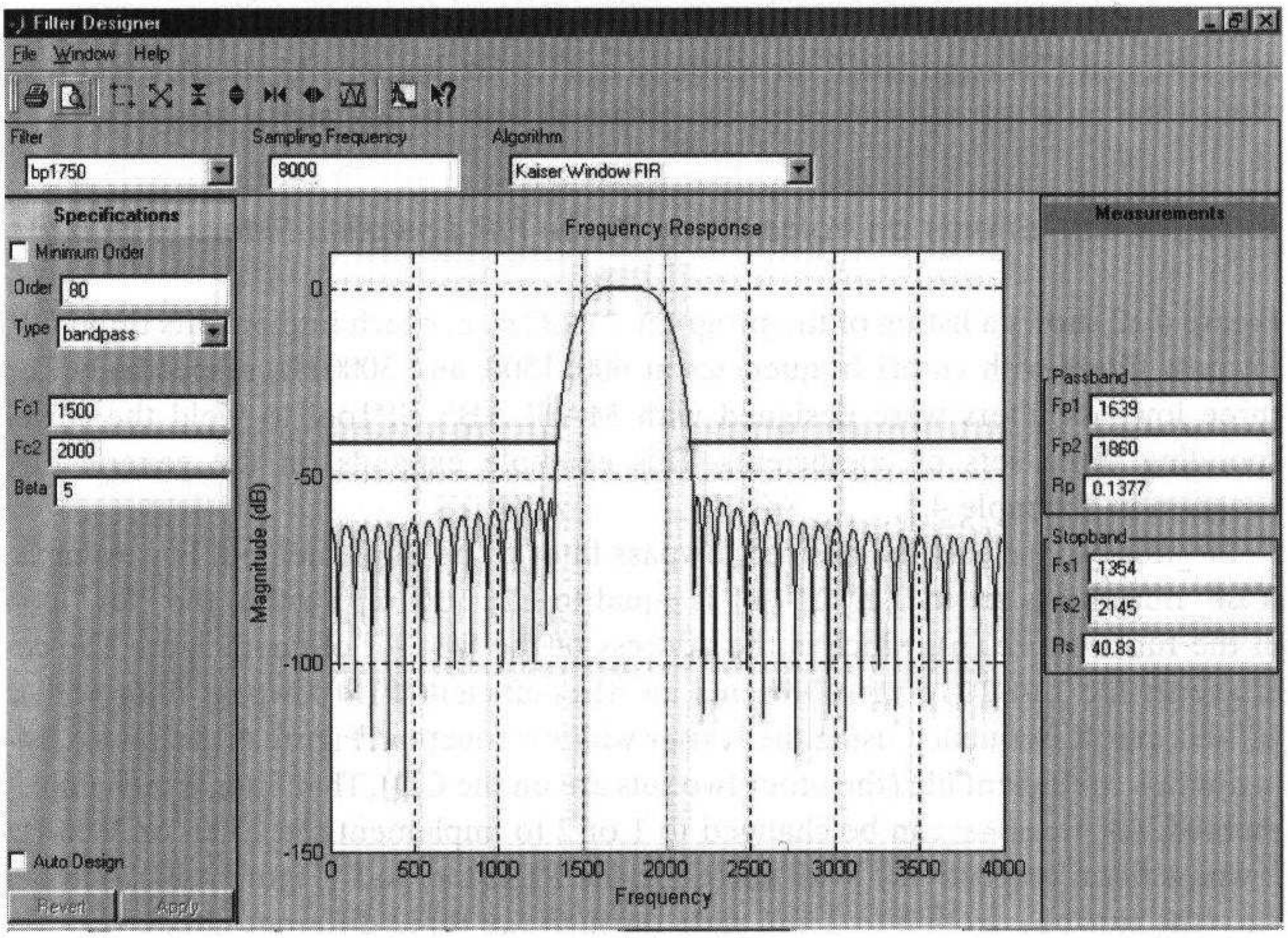

FIGURE 4.10. MATLAB's filter designer SPTool, showing the characteristics of a FIR bandpass filter centered at 1750 Hz.

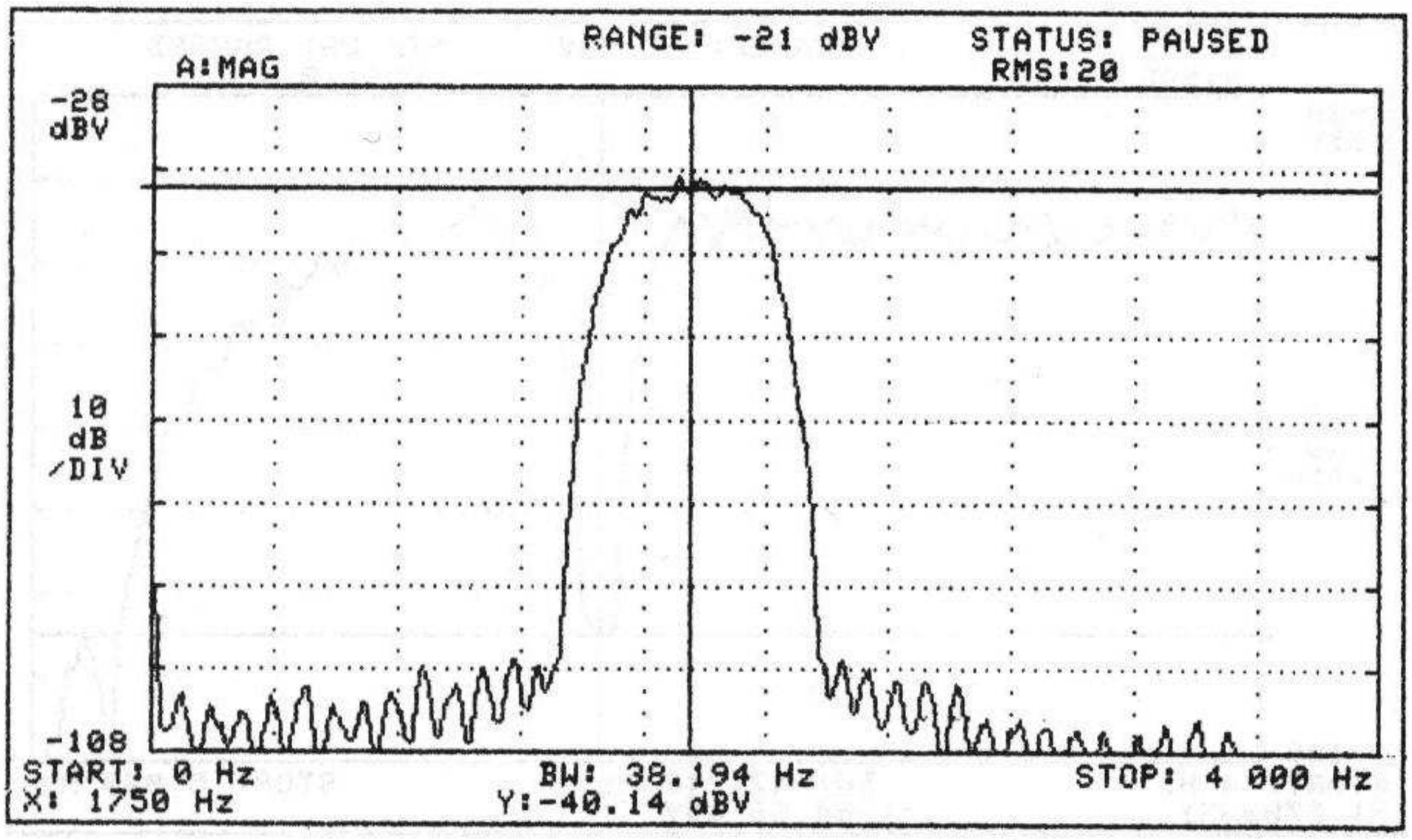

FIGURE 4.11. Output frequency response of a FIR bandpass filter centered at 1750 Hz, obtained with a signal analyzer.

with MATLAB's SPTool (Appendix D). Select the incremental Build, and the new coefficient file *bp1750.cof* will automatically be included in the project. Run again and verify an FIR bandpass filter centered at 1750 Hz. Figure 4.11 shows a real-time plot of the output frequency response obtained with the HP signal analyzer.

Example 4.2: Effects on Voice Using Three FIR Lowpass Filters (FIR3LP)

Figure 4.12 shows a listing of the program *FIR3lp.c*, which implements three FIR lowpass filters with cutoff frequencies at 600, 1500, and 3000 Hz, respectively. The three lowpass filters were designed with MATLAB's SPTool to yield the corresponding three sets of coefficients. This example expands on the generic FIR program in Example 4.1.

`LP_number` selects the desired lowpass filter to be implemented. For example, if `LP_number` is set to 0, `h[0][i]` is equal to `hlp600[i]` (within the "for" loop in the function *main*), which is the address of the first set of coefficients. The coefficients file `LP600.cof` represents an 81-coefficient FIR lowpass filter with a 600-Hz cutoff frequency, using the Kaiser window function. Figure 4.13 shows a listing of this coefficient file (the other two sets are on the CD). That filter is then implemented. `LP_number` can be changed to 1 or 2 to implement the 1500- or 3000-Hz lowpass filter, respectively. With the GEL file `FIR3LP.gel` (Figure 4.14), one can vary `LP_number` from 0 to 2 and slide through the three different filters.

Build this project as **FIR3LP.** Use the `.wav` file *TheForce.wav* (on the CD) as input and observe the effects of the three lowpass filters on the input voice. With

```
//Fir3LP.c FIR using 3 lowpass coefficients with three different BW

#include "lp600.cof"                          //coeff file LP @ 600 Hz
#include "lp1500.cof"                         //coeff file LP @ 1500 Hz
#include "lp3000.cof"                         //coeff file LP @ 3000 Hz
#include "dsk6713_aic23.h"                    //codec-dsk support file
Uint32 fs=DSK6713_AIC23_FREQ_8KHZ;            //set sampling rate
short LP_number = 0;                          //start with 1st LP filter
int yn = 0;                                   //initialize filter's output
short dly[N];                                 //delay samples
short h[3][N];                                //filter characteristics 3xN

interrupt void c_int11()                      //ISR
{
 short i;
 dly[0] = input_sample();                     //newest input @ top of buffer
 yn = 0;                                      //initialize filter output
 for (i = 0; i< N; i++)
 yn +=(h[LP_number][i]*dly[i]);               //y(n) += h(LP#,i)*x(n-i)
 for (i = N-1; i > 0; i--)                    //starting @ bottom of buffer
      dly[i] = dly[i-1];                      //update delays with data move
 output_sample(yn >> 15);                     //output filter
 return;                                      //return from interrupt
}

void main()
{
 short i;
 for (i=0; i<N; i++)
  {
      dly[i] = 0;                             //init buffer
      h[0][i] = hlp600[i];                    //start addr of LP600 coeff
      h[1][i] = hlp1500[i];                   //start addr of LP1500 coeff
      h[2][i] = hlp3000[i];                   //start addr of LP3000 coeff
  }
 comm_intr();                                 //init DSK, codec, McBSP
 while(1);                                    //infinite loop
}
```

FIGURE 4.12. FIR program to implement three different lowpass filters using a slider for selection (`FIR3LP.c`).

```
//LP600.cof FIR lowpass filter coefficients using Kaiser window

#define N 81                                  //length of filter

short hlp600[N]={0,-6,-14,-22,-26,-24,-13,8,34,61,80,83,63,19,-43,-113,
-171,-201,-185,-117,0,146,292,398,428,355,174,-99,-416,-712,-905,-921,
-700,-218,511,1424,2425,3391,4196,4729,4915,4729,4196,3391,2425,1424,
511,-218,-700,-921,-905,-712,-416,-99,174,355,428,398,292,146,0,-117,
-185,-201,-171,-113,-43,19,63,83,80,61,34,8,-13,-24,-26,-22,-14,-6,0};
```

FIGURE 4.13. Coefficient file for a FIR lowpass filter with a 600-Hz cutoff frequency (`LP600.cof`).

```
/*FIR3LP.gel Gel file to step through three different LP filters*/

menuitem "Filter Characteristics"

slider Filter(0,2,1,1,filterparameter) /*from 0 to 2,incr by 1*/
{
      LP_number = filterparameter;      /*for 3 LP filters*/
}
```

FIGURE 4.14. GEL file for selecting one of three FIR lowpass filter coefficients (`FIR3LP.gel`).

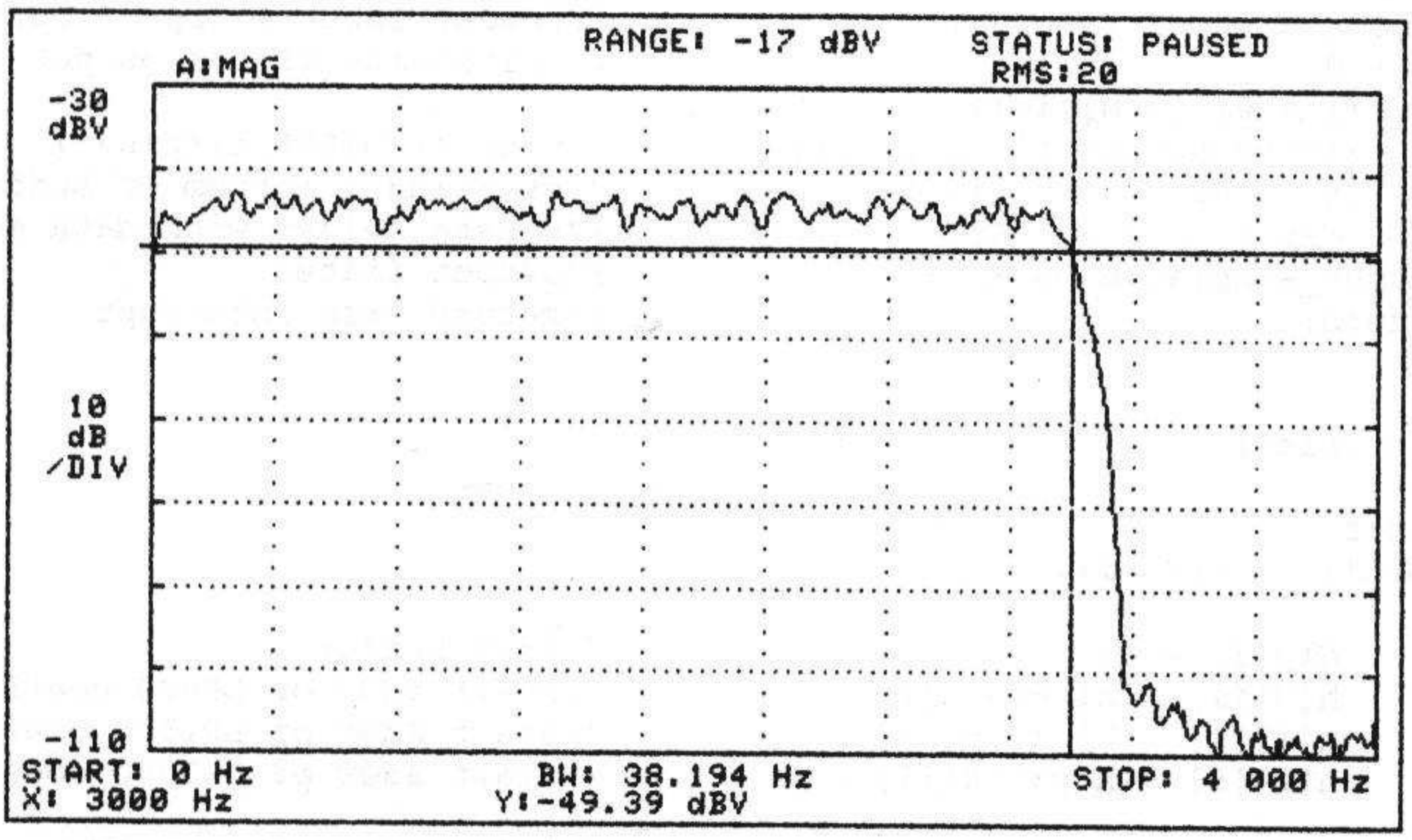

FIGURE 4.15. Frequency response of a FIR lowpass filter with a bandwidth of 3000 Hz using `LP3000.cof`, obtained with a signal analyzer.

the lower bandwidth of 600 Hz, using the first set of coefficients, the frequency components of the speech signal above 600 Hz are suppressed. Connect the output to a speaker or a spectrum analyzer to verify such results, and observe the different bandwidths of the three FIR lowpass filters. The shareware utility Goldwave generates different signals, including noise, using a sound card (see Appendix E). The output from the sound card with the noise generated by Goldwave can be used as the input to the DSK. Connecting the output from the DSK as the input to the sound card, Goldwave can also be used as a virtual spectrum analyzer. The frequency responses of these three lowpass filters can be obtained readily in real time. Figure 4.15 shows the frequency response of the 3000-Hz lowpass FIR filter, obtained with an HP signal analyzer.

Example 4.3: Implementation of Four Different Filters: Lowpass, Highpass, Bandpass, and Bandstop (`FIR4types`)

This example is similar to Example 4.2 and illustrates the GEL (slider) file to step through four different types of FIR filters. Each filter has 81 coefficients, designed with MATLAB's SPTool. The four coefficient files (on the accompanying CD) are:

1. *`lp1500.cof`*: lowpass with bandwidth of 1500 Hz
2. *`hp2200.cof`*: highpass with bandwidth of 2200 Hz
3. *`bp1750.cof`*: bandpass with center frequency at 1750 Hz
4. *`bs790.cof`*: bandstop with center frequency at 790 Hz

The program *`FIR4types.c`* (on the CD) implements this project. The program `FIR3LP.c` (Example 4.2) is modified slightly to incorporate the implementation of a fourth filter.

Build and run this project as **`FIR4types`**. Load the GEL file *`FIR4types.gel`* (on the CD) and verify the implementation of the four different FIR filters. This example can readily be expanded to implement more FIR filters.

Figure 4.16 shows the frequency response of the FIR bandstop filter centered at 790 Hz, using the coefficient file `bs790.cof`.

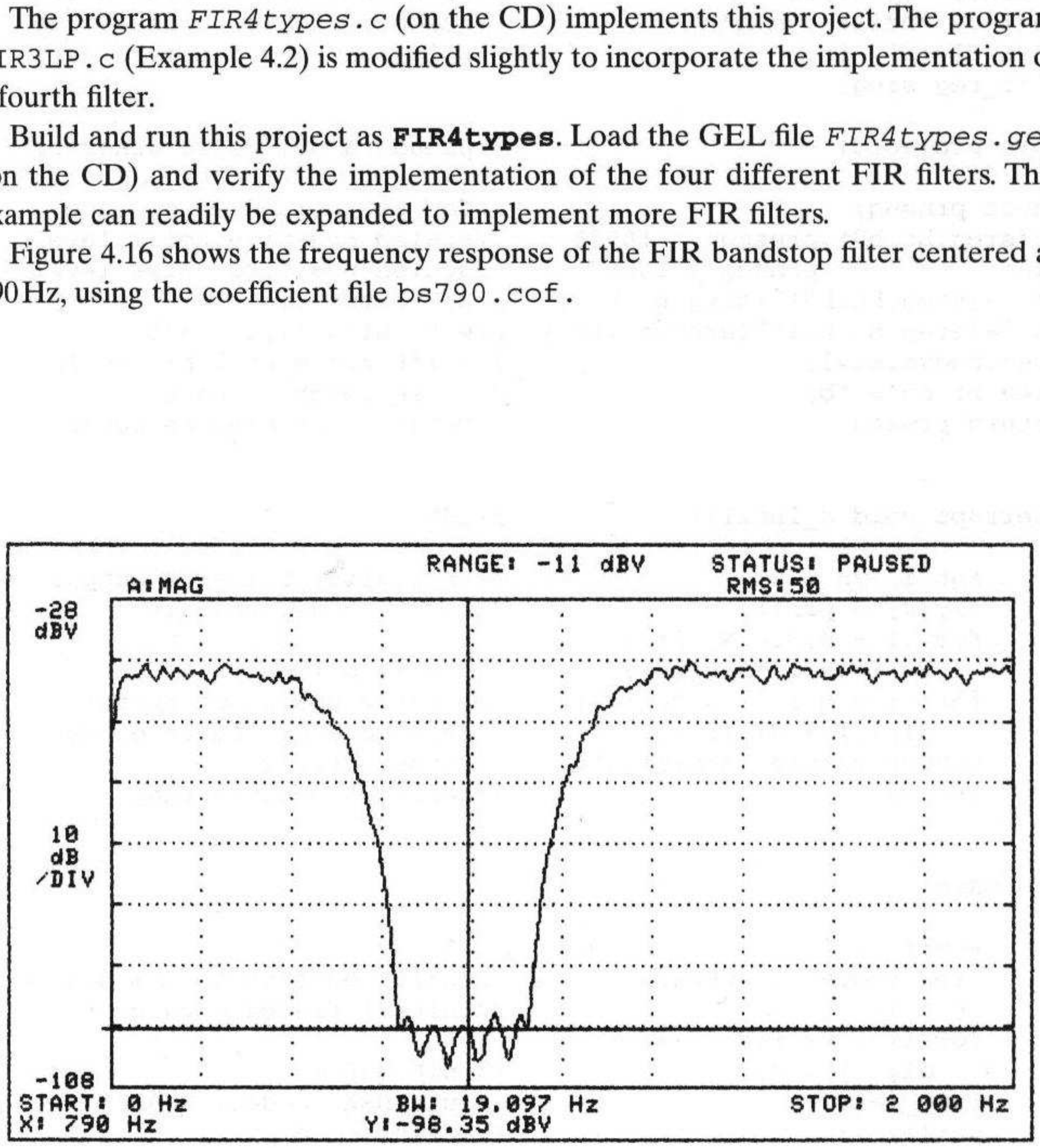

FIGURE 4.16. Frequency response of a FIR bandstop filter centered at 790 Hz using `bs790.cof`, obtained with a signal analyzer.