Design of the IIP Radiometer Digital IF Section (Rev. 1)

Steven W. Ellingson*

May 6, 2002

Contents

*The Ohio State University, ElectroScience Laboratory, 1320 Kinnear Road, Columbus, OH 43210, USA. Email: ellingson.1@osu.edu.

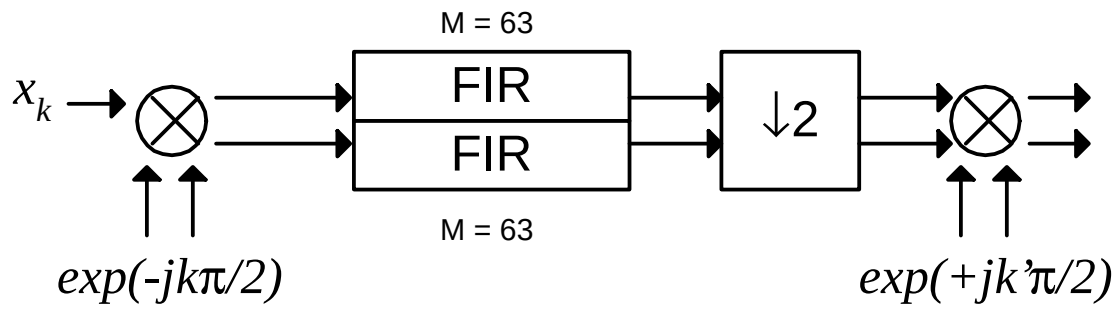

Figure 1: Conceptual block diagram of the digital IF path following one of the A/Ds.

1 Introduction

This document describes a proposed design for the digital intermediate frequency (IF) section of the IIP radiometer. The architecture of the radiometer was previously described in [1]. Here, we elaborate on the implementation of the signal path beginning at the output of one of the A/Ds and ending at the output of the final $F_S/4$ spectral shift. A simple conceptual description of the required processing is shown in Figure 1. The sequence of operations is:

1. Downconvert the real-valued output of the A/D by $F_S/4$, where F_S is the sample rate of the A/D. In this case, F_S=200 MSPS. This results in a complex baseband signal.

2. Lowpass filter to limit the bandwidth to 50 MHz and to remove the undesired sideband created in Step 1.

3. Decimate by 2, resulting in a new sample rate F_S'=100 MSPS.

4. Spectral shift by $\pm F_S'/4$. This is either an upconversion (+) or downconversion (−) depending on which A/D we are working with. For the remainder of this document, we will assume upconversion. The process for downconversion involves only a trivial modification, as will be discussed below.

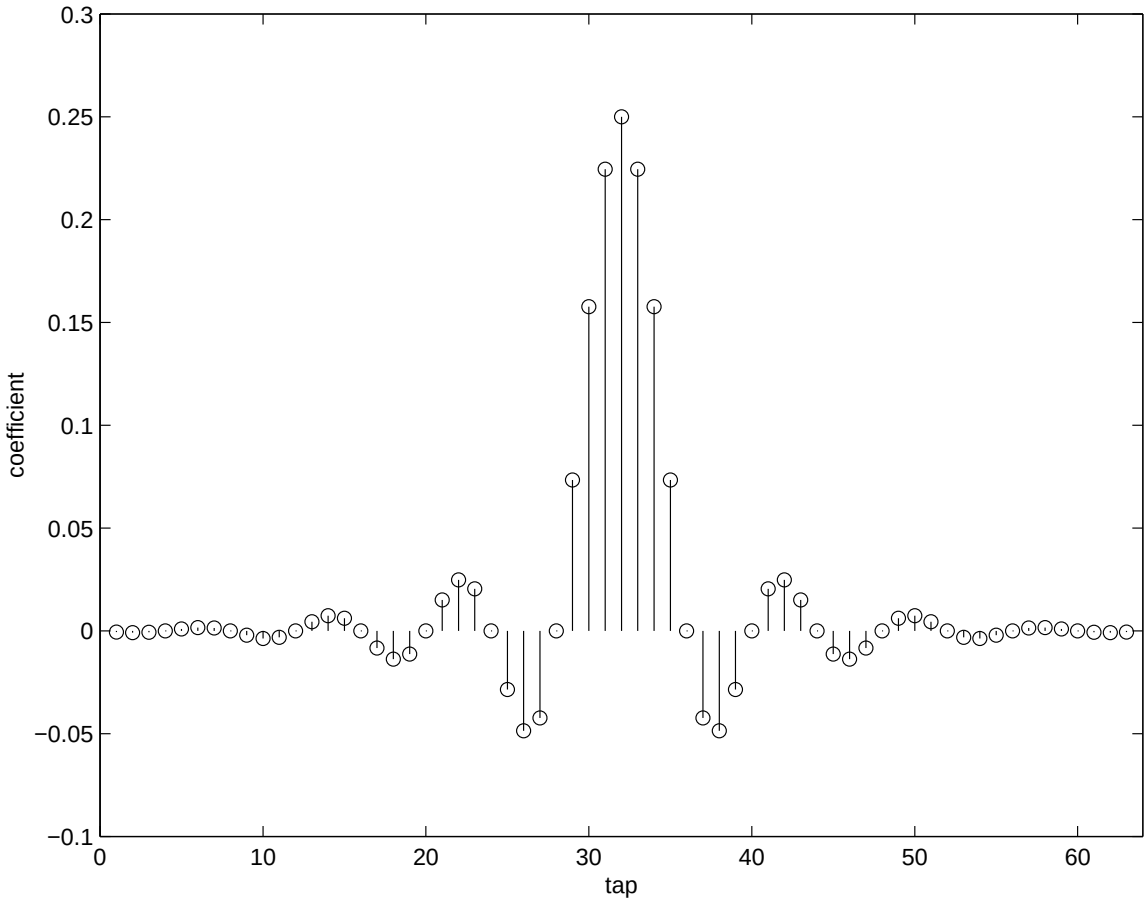

Figure 2: Impulse response of lowpass FIR filter (also, the values of the filter coefficients).

2 Lowpass FIR Filter Design

As a first step, let us design a suitable lowpass FIR filter. The design presented here will be adequate, but by no means optimum. In the following sections, a simple FIR implementation of the filter is used. This architecture is independent of the values of the coefficients defining the filter, so the process of substituting other filters at a later time is trivial.

This filter must accept a sample stream at $F_S = 200$ MSPS and impose a lowpass cutoff frequency around 25 MHz. We also desire aggressive attenuation at 30 MHz and beyond. The impulse and frequency responses of a suitable filter are shown in Figures 2 and 3 respectively. The impulse response also gives the coefficients of the filter; thus a filter of length $M = 63$ taps is proposed.

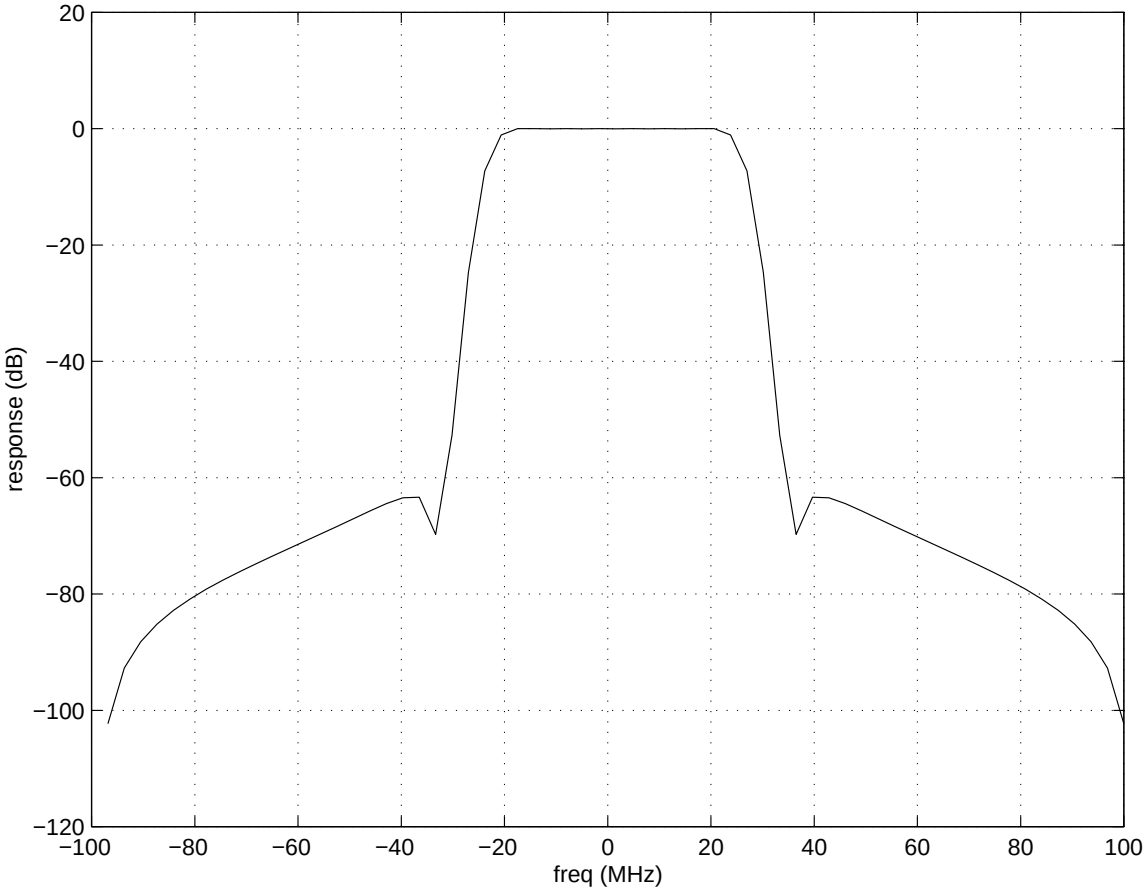

Figure 3: Frequency response of the lowpass FIR filter.

This filter was designed using the MATLAB code presented in Appendix A. The strategy was as follows:

1. A "model" frequency response is defined using $L = 256$ points. The model frequency response is chosen to be a perfect rectangular window with 25 MHz cutoff.

2. This frequency response is transformed into the time domain using the inverse FFT.

3. The resulting impulse response is truncated to the $M = 63$ largest contiguous values.

4. A Hamming window is applied to the result to smooth the variation in the passband.

For the remainder of this document, this filter is referred to by the values of it's coefficients h_m, where $m = 1, 2, ..M$.

3 Recommended Implementation

A literal implementation of the design as shown in Figure 1 would be very inefficient. A better design which is functionally equivalent is derived below.

First, note that the $F_S/4$ downconversion amounts to multiplication of the real-valued input sample stream x_k by $\exp(-j\omega t)$. In this case, $\omega = 2\pi(F_S/4)$ and $t = k/F_S$; thus $\exp(-j\omega t) = \exp(-j\frac{\pi}{2}k)$, which is simply the repeating sequence $\{+1, -j, -1, +j, ...\}$. Let u_k be the output of the $F_S/4$ downconversion. We find:

k	0	1	2	3	4	...
$\mathrm{Re}\{u_k\}$	x_0	0	$-x_2$	0	x_4	...
$\mathrm{Im}\{u_k\}$	0	$-x_1$	0	x_3	0	...

Next, the real and imaginary components of u_k are independently processed through the lowpass FIR filter. Let us consider the processing of the real component of u_k first. The table below shows the state of the tapped delay line for a few contiguous time steps k:

k	h_0	h_1	h_2	h_3	h_4	...	h_M
0	x_0	0	$-x_2$	0	x_4	...	$-x_M$
1	0	$-x_2$	0	x_4	0	...	0
2	$-x_2$	0	x_4	0	$-x_6$	...	$-x_{M+2}$
3	0	x_4	0	$-x_6$	0	...	0
4	x_4	0	$-x_6$	0	x_8	...	$-x_{M+4}$

The top row shows the values of the filter coefficients and the remaining rows show the samples that are being multiplied by these coefficients at each time step. The process implied by this table can be simplified in at least three ways. First, looking ahead, note that the output of this filter is decimated by two. In other words, only every other output from the filter is needed. Thus, we can simply omit the processing associated with $k = 1, 3, ...$ If we do that, then note that the odd-indexed coefficients $(h_1, h_3, ...)$ are multiplying only zeros; therefore, these filter taps can be omitted. Finally, note we can absorb the alternating sign changes in the input samples x_k into the filter coefficients h_m, *if* we remember to alternate the sign of the filter output at each time step as well. The resulting table of operations now looks like this:

k'	h_0	$-h_2$	h_4	...	$-h_M$	multiply output by...
0	x_0	x_2	x_4	...	x_M	$+1$
1	x_2	x_4	x_6	...	x_{M+2}	-1
2	x_4	x_6	x_8	...	x_{M+4}	$+1$

Above, k' refers to the decimated sample index such that k'_0 is associated with k_0, k'_1 is associated with k_2, and so on. Note that we have now consolidated the $F_S/4$ downconversion, $M = 63$ filter, and decimation into a single operation involving only $(M + 1)/2 = 32$ real multiplies and a sign change.

Applying the same analysis to the *imaginary* component of u_k, we obtain:

k'	$-h_1$	h_3	$-h_5$	...	h_{M-1}	multiply output by...
0	x_1	x_3	x_5	...	x_{M-1}	$+1$
1	x_3	x_5	x_7	...	x_{M+1}	-1
2	x_5	x_7	x_9	...	x_{M+3}	$+1$

In this case, we have $(M-1)/2 = 31$ multiplies (one fewer than for the "real" branch of the processor). Finally, note that the sign alternation is synchronized between the real and imaginary branches. This will prove useful in a moment.

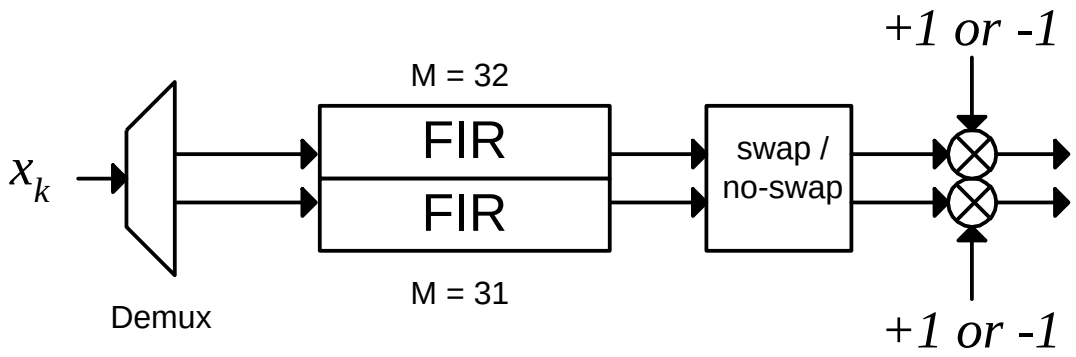

Figure 4: Recommended implementation of the digital IF path following one of the A/Ds.

The final step is to spectrally shift the output of the filters, taken as complex-valued samples, by $+F'_S/4$. Taking the output of the filters at time k' as $a_{k'} + jb_{k'}$, the desired result is:

k'	Output
0	$+a_0 + jb_0$
1	$-b_1 + ja_1$
2	$-a_2 - jb_2$
3	$+b_3 - ja_3$

In other words, we can implement the spectral shift simply by following a pattern of sign changes and swapping of the real and imaginary components. In fact, we can consolidate the sign alternation specified for the output of the polyphase filters derived above as well. If we do this, the final resulting process is:

k'	Swap real and imag.?	Multiply real by:	Multiply imag. by:
0	no	$+1$	$+1$
1	yes	$+1$	-1
2	no	-1	-1
3	yes	-1	$+1$

Note that if we had wanted to downconvert at this step (as opposed to upconverting), this would be acheived simply by sign changes in the rightmost two columns of the above table.

Figure 4 shows the completed, recommended implementation of the digital IF processor for a single A/D.

A MATLAB Code for Filter Generation

```
clear all;

FS = 200e+6;  % samples per second
FCO = 25e+6;  % nominal lowpass cutoff freq (Hz)
L = 256;
M = 63; % size of filter - must be odd

% hamming window
k = [0:1:M-1];
w = 0.54-0.46*cos(2*pi*k/(M-1));

H = zeros(L,1); % frequency response
bco = floor((L/2)*FCO/(FS/2));
H(1:bco) = ones(bco,1);
H(L-bco+1:L) = ones(bco,1);

h = ifft(H);

% truncate to length M
N=M+1;
ht(    1:N/2) = real(h(L-N/2+1:L  ));
ht(N/2+1:N  ) = real(h(      1:N/2));
ht = ht(2:N).*w;

% show response of new filter
f = [-M/2+1:M/2]*FS/M;
HT = fft(ht);
plot(f/(1e6),20*log10(abs(fftshift(HT))));
```

```
xlabel('freq (MHz)');
ylabel('response (dB)');
grid on;

stem(ht);
xlabel('tap');
ylabel('coefficient');
axis([0 64 -.1 .3]);

save ht.mat ht;
```

References

[1] S.W. Ellingson, "Design Concept for the IIP Radiometer", informal memo, January 11, 2002.