

# Interim Project Report



**Project Name:** Efficient Implementation of SSB demodulation,  
using multirate signal processing

**Team Name:** Tema Aliasing

**Team Members:** Martin Lindberg

**Email Adress:** mlch03@kom.aau.dk

**Contact No:** +45 24 45 17 19

**Instructor:** Peter Koch - pk@es.aau.dk



# Innovative nordic

## 0.1 Introduction

The main goal for introducing Software Defined Radio (SDR) is the reduction of the analog hardware in the system. Software radios define an emerging technology, thought to provide a flexible radio system, reconfigurable and reprogrammable by software [1]. The advantages of SDR can be stated as follows.

- Flexibility: Changing the functionality in the receiver can easily be done by a software updating, instead of replacing or modifying existing physical hardware like analog electronics.
- Cost: The total cost are together with power one of the biggest technical issues facing developers of SDR.
- Time-to-market: The prototype development of a SDR can be done with implementation of FGPA's, which makes the time-to-marked fast.
- Size: Analog components will be replaced by digital hardware, i.e. the physical size of the receiver will be reduced.
- ADC: The ADC must have a high dynamic range and sampling rate, this are limiting factors that determining the maximum achievable data rate of the receiver.

In software radio it is desirable to move the analog-to-digital conversion as close to the antenna as possible, as in the ideal software radio [6]. This is shown in figure 1, but is unfortunately not possible yet due to limitations in todays hardware technology [4].

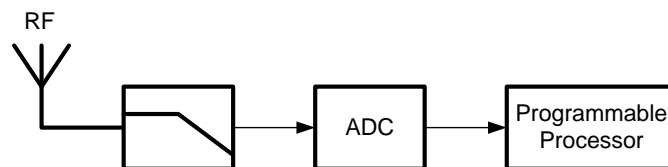


Figure 1: The ideal software radio architecture, where the digital processing starts at the RF range [4].

A part digitalization of a radio receiver will cause incorporation of software in the system and new hardware as well. The rapid development of high-speed analog to digital (ADC) converters

and large field programmable gate arrays (FPGA) allows designers to design compact solutions that were unthinkable a few years ago. Digital signal processing (DSP) with affordable high performance makes the possibility for mapping of the analogue part of the receiver in to the digital domain reachable.

The basis application point in this report is to redesign the receiver part of a current transceiver, which is depicted in the simplified block diagram in figure 1.2.

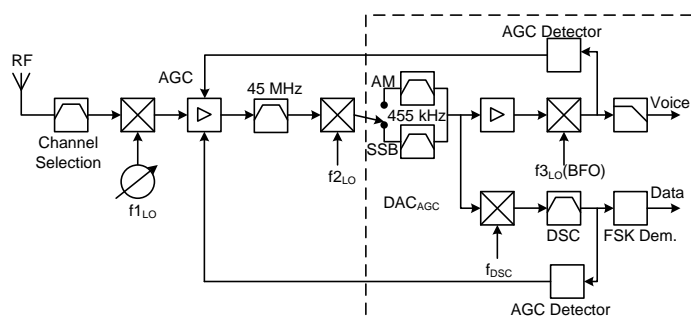


Figure 2: Simplification of a receiver system. The blocks inside the square are the parts of focus in the thesis.

## 0.2 Problem description

In the preceding, the implementation of software radio into an existing system has outlined that there is a need for investigation. Since the tasks of transforming a part of the receiver from the analog domain into the digital hardware is not a trivial task, there is a need for analysing these tasks. These are two coherent tasks, the first is the mapping of the analog circuitry into the digital domain, and the second is the design of a system architecture where the algorithm can be implemented.

Since this work is an existent system, the function basis is a priori information from the analog domain. This implies that there is a need for functions to be synthesized into the digital domain. Knowing about these existing approaches, considerations can be done in order to optimize the algorithms instead of developing new ones.

The receiver should perform a real-time SSB demodulation using the traditional heterodyne approach, and still be able to keep the specification within the limitations set by ETSI. A software radio for a Super-Heterodyne receiver should be able to handle a large dynamic range, for which reason an AGC is needed. Since the aim of this work is to consider a receiver design with the ADC placed after the second mixer, the second IF must be chosen carefully. Before the final demodulation re-produces the baseband output signal, the modulated signal is filtered to make the last selection in the chain of the receivers. This is done in several steps through the chain to avoid adjacent channel, and image response.

### 0.3 Design considerations

This section will state the design considerations as an intro to the algorithm chapter. We have the possibility to lower the second intermediate frequency, from 455 kHz to a new wanted frequency. There are mainly two things to be aware of when lowering this frequency, the bandwidth of the spectrum which is assumed to be band limited (attenuated by 70 dB) to  $\pm 40$  kHz with  $f_c$  as the center frequency of the spectrum. The other important consideration is the down-sampling factor which is wanted, to be an integer multiplum of the samplings frequency. In the design considerations, we consider the block diagram shown in figure 3.

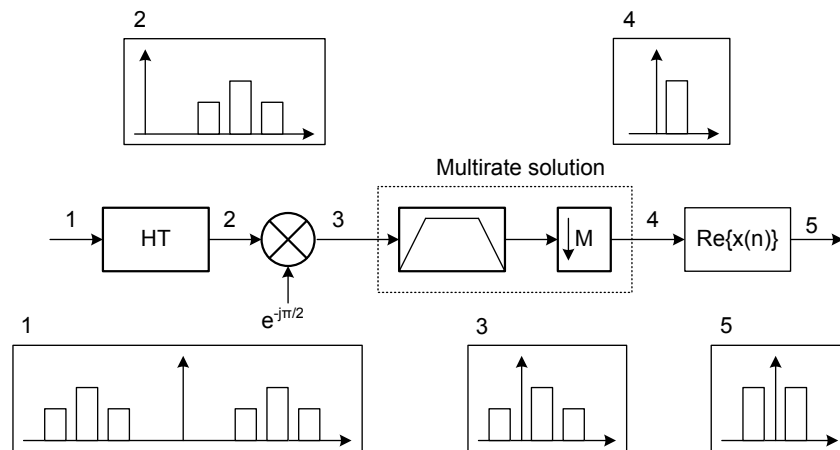


Figure 3: Block diagram showing the simulated scenario. The solution space is surrounded by the (a)dotted box, indicating that there are several possibilities to decrease the computation requirements, within this box.

#### Hilbert Transform

The Hilbert transform is applied to the sampled incoming signal, by using a two path half-band filter that we modulate to the quarter-sample rate, as shown in figure 4. When applying the Hilbert transform we zero out all negative frequency. Since the half of coefficients in a half band filter are zero, we can cost free utilize the inherent properties of down-sample by a factor of two by skipping the zero multiplications. This results in a throughput of four times the intermediate frequency.

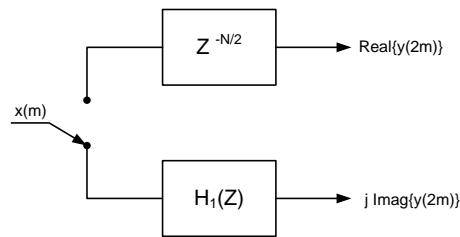


Figure 4: Two-path Hilbert transform filter.

Figure 5 shows the impulse response of the two-path Hilbert transform filter, where the upper figure is the real part of the filter. Since the upper part is a delay of  $N/2$ , where  $N$  is the length of the filter, the real path impulse response is simply a delayed version of the impuls. The impulse response in the lower part of figure 5, will shift all the negative frequency components by  $+90^\circ$  and the positive by  $-90^\circ$ .

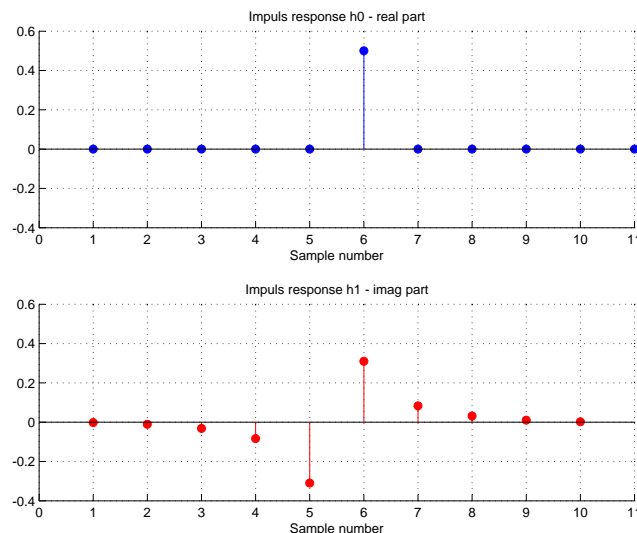


Figure 5: Impulse response of the upper and lower path of the HT. We see that the lower impulse response correspond to the properties of the HT.

Another recursive method to perform the Hilbert transform, is the a recursive method based on the two-path all-pass filters [3]. This recursive structure can be performed with a side band attenuation of 80 dB using only four coefficients.

### Down-conversion

Moving the wanted channel to base-band with the effect of the quadrature is cost-free in a hardware solution, and makes is less computational demanding . The output of the quadrature is shown in equation 1 to 4, where  $n$  is the sample number.

$$\begin{aligned}
I & \text{ for } n = 0, 4, \dots & (1) \\
Q & \text{ for } n = 1, 5, \dots & (2) \\
-I & \text{ for } n = 2, 6, \dots & (3) \\
-Q & \text{ for } n = 3, 7, \dots & (4)
\end{aligned}$$

In hardware this corresponds to only choose the right path and change sign this solution should be compared to a CORDIC solution, which is computational expensive.

### FIR Filter Design

The complex band-pass filter requirements to obtain the wanted selectivity are stated in table 1, where the transition band ( $\Delta f$ ) is the guard band that separates the channels.

$F_s$	=	4 fc
$\Delta f$	=	600 Hz
Stopband Attenuation	=	80 dB

Table 1: Filter requirements in order to obtain the wanted selectivity.

The required order of the filter is highly depended on the samplings frequency, and since we have chosen to use the properties of quadrature sampling the sampling frequency is dependent of the second intermediate frequency. Figure 6 shows an estimate of the filter order, when using the window and the Parks-McClellan (PM) method for designing the filters.

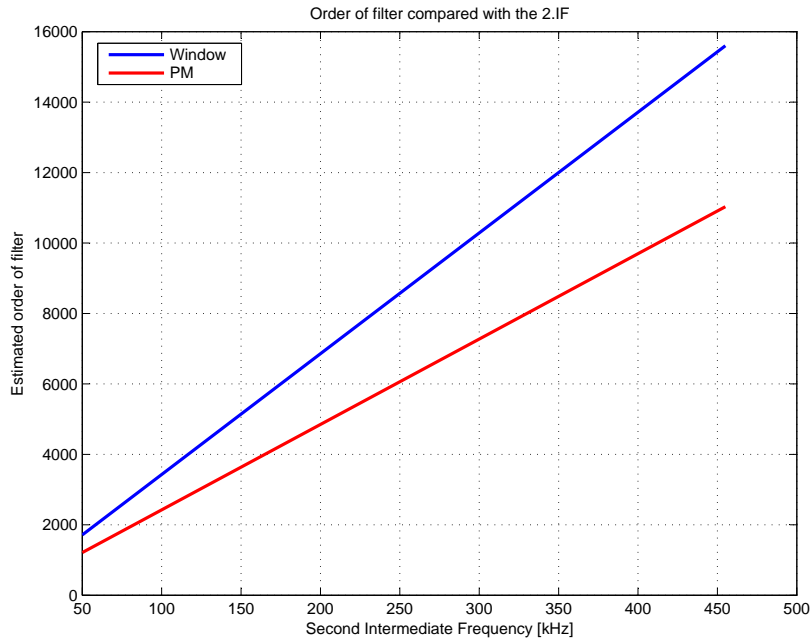


Figure 6: The estimated order of the filter, in order to obtain the wanted filter characteristic compared to the intermediate frequency, where the red line is based on Parks-McClellan filter design, and the blue line is base on the window method.

### FIR Polyphase decomposition

We can take advantage of decomposition the FIR filter, when using down-sampling, by utilizing the properties of the noble identity. Before decomposition the filter, we need to know the down-sampling factor  $M$ , since the output rate need to be 12 kHz, we use equation 5 to calculate  $M$ .

$$M = \frac{4fc}{12k} \quad (5)$$

Figure 7 shows how the down-sampling factor increases when increasing the second intermediate frequency. The required number of sub-filters in a poly-phase decomposition of a FIR filter are equal to the down-sampling factor  $M$ .

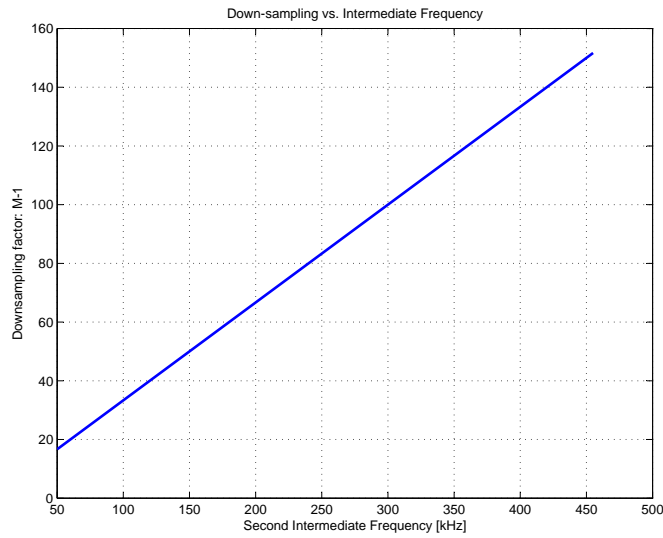


Figure 7: The down-sampling factor is linear dependent of the second intermediate frequency, which means that when lowering the frequency a lower down-sampling factor is required.

If we choose to lower the second intermediate frequency the down-sampling factor will decrease as well, like the sampling frequency and the order of the filter. Since we only need to calculate each sub-filter at each incoming samples, we can calculate the required number of operations in each sub-filter independent of the sampling frequency.

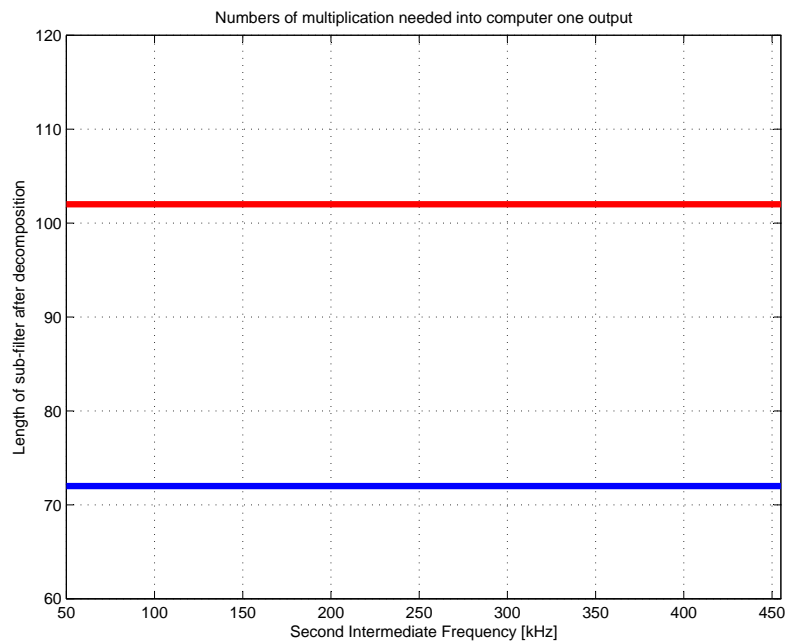


Figure 8: The red line is the window function that would require 102 operations in each sub-filter. The blue line is the PM, which only requires 72 operations in each sub-filter.

In figure 8 we see that when using the window function we need 102 computations in each sub-filter is needed, but only 72 when using the PM filter design method. It can be argued why this is the case, and that PM has the disadvantage of having more pass-band ripple, but we will not look into this now. Since the poly-phase filter still run at full-rate, we want to see if it is possible to make several down-sampling filter that runs at a lower rate, and still obtaining the wanted selectivity.

## Two-path recursive All-pass half-band filter

The two-path recursive all-pass filter is able to down-sample by a factor of two in each section [2]. This will cause the down-samplings factor to be a power of two. When down-sampling by a factor of two in each section, the down-sampling factor need to be a power of two as shown in equation 6. We know that the spectrum is band limited within  $\pm 40$  kHz. The output at audio has as sample frequency of 12 kHz, which corresponds to that the second intermediate frequency is not allowed to be lower than 96 kHz, and five section of down-sampling.

$$\frac{4 \cdot fc}{2^N} = F_{s_{out}} \quad (6)$$

$$\frac{4 \cdot 96 \text{ kHz}}{2^N} = 12 \text{ kHz} \quad (7)$$

However, we can not just cascade couple all five sections, this would cause aliasing in the wanted spectrum. After two down-samplings sections the spectrum need to be band limited. This is done by a recursive all-pass filter, which is a more efficient filter than the regular IIR filters [3]. The spectrum of the first half-band filter can be seen in figure 9, while the coefficients are shown table 2.

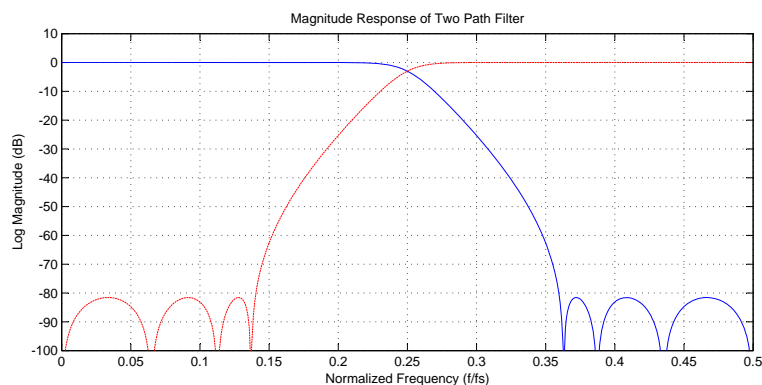


Figure 9: The first and second half-band filter, when down-sampling the signal by a factor two. The red line is the filter plotted as high pass filter, and is only a matter of sign changing [3]. This is included in order to show the possibility in this filter types.

In table 2 we see that only three multiplications needed to calculated the filter, two multipliers in the first path and one in the second path.

<b>Path-0</b>	<b>Polynomial Coefficient</b>
Filter-0	[1 0 0.07638269328882]
Filter-2	[1 0 0.69878248805530]

<b>Path-1</b>	<b>Polynomial Coefficient</b>
Filter-1	[1 0 0.30208200300176]

Table 2: Coefficients for the first half-band filter.

The last filter in the chain, need to be a complex filter, since the spectrum is complex and we want to obtain the final selectivity. This is done by up-converting a FIR filter. To obtain the wanted selectivity we need 201 taps.

The last recursive half-band filters are not plotted in this section, but will be present in the simulation results. The required number of coefficients is shown in table 3. Notice that we use five coefficient in block-3 instead of three, to shown how the number of coefficient change the steepness of the filter. This is further elaborated in the next chapter.

<b>Block-1</b>	<b>Block-2</b>	<b>Block-All</b>	<b>Filter-3</b>	<b>Bblock-4</b>	<b>Block-5</b>	<b>Block-FIR</b>	<b>Total</b>
3	3	13	5	3	3	201	231
48	24	104	20	6	3	201	406

Table 3: The second row shows the number of coefficients used in each section. While the third row shows the required number of multiplications needed in order to produce one output sample.

In table 3 the required number of coefficients is stated. To produce one output a total sum of 231 coefficients and 406 multiplications are required, as stated in table 3. This should be compared to the poly-phase implementation, where the entire filter length  $N$  need to run in order to deliver one output sample. Figure 6 indicates that the filter length is about 2000.

## 0.4 Simulation results

With the design consideration in mind we are able to design and simulate the system function to see if the requirements stated can be fulfilled. The calculation of the filter coefficient needed in the All-pass sections, is based on [7]. In this algorithm several different filter types has been used:

- Half-band Hilbert Transform
- Two-path Half-band filter
- Recursive All-pass filter
- Complex FIR filter

There are several possible constellations for combining the filters into this application, but in this algorithm most of the multirate techniques are used, in order to see the advantage in each filter design procedure, as well as the reduction in the computational complexity.

## 0.5 Simulation setup

To get a quick overview of the algorithm, figure 10 shows the different blocks, as designed in section 0.3. Since we are working on a multirate system the sampling frequency is varying from block to block. At the block diagram each incoming and outgoing frequency is stated to give an overview of the systems computational load.

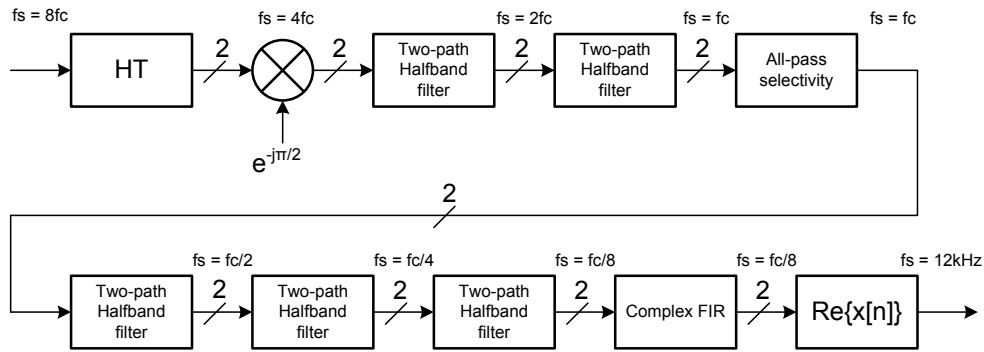


Figure 10: Block diagram of the algorithm showing the different filter blocks use. The number two on the arrow connections indicates that the signal is complex.

In order to fulfil the specification to the system specified by ETSI, we use the frequency component located relative to the carrier as the specification stats. Another important property to be aware of, when using multirate system is unwanted aliasing components that could rise in the wanted spectrum. In order to take this problem into account when simulating the system, we locate two frequency components at  $fc \pm 13$  kHz. To see whether the wanted channel is maintained and leaved unchanged though the system, we simulate the wanted channel with four modulated frequencies from the boundary of 300 to 2700 Hz.

Fs in:	8 fc
Fs out:	12 kHz
Test signal:	$fc(0.3,2.7)$ kHz
Test speech:	BW: 300-2700 Hz
Upper adj. signal:	$fc(+4,+5,+8)$ kHz
Lower adj. signal:	$fc(-1,-2,-5)$ kHz
Aliasing signal:	$fc+13$ kHz
Aliasing signal:	$fc-13$ kHz

Table 4: Test signals used in the simulation.

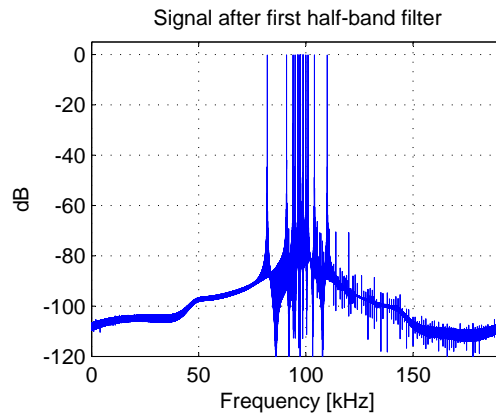


Figure 11: The SSB spectrum where the wanted signal is modulated with a carrier of 96 kHz, the other frequency components are unwanted

### Hilbert transform

The Hilbert transform is designed by quadrature modulation of the low-pass coefficient, and then implemented as a two-path half-band filter. The implementation of a two-path Hilbert transform makes it possible to obtain a cost free down-conversion by 2-1, since half of the coefficient is zero. The upper path of the Hilbert transform will be a delay of  $-N/2$ , and the lower path will be the coefficients of the Hilbert transform, by multiplying the following by  $j$  we shift the signal by  $90^\circ$ , doing so will present the signal analytic, given by  $x_a = x_{upper} + jx_{lower}$ . This can be seen as a one side spectrum, where the negative frequencies are zeroed out, as the right figure in 12. It should be noticed that the spectrum now is complex and asymmetric.

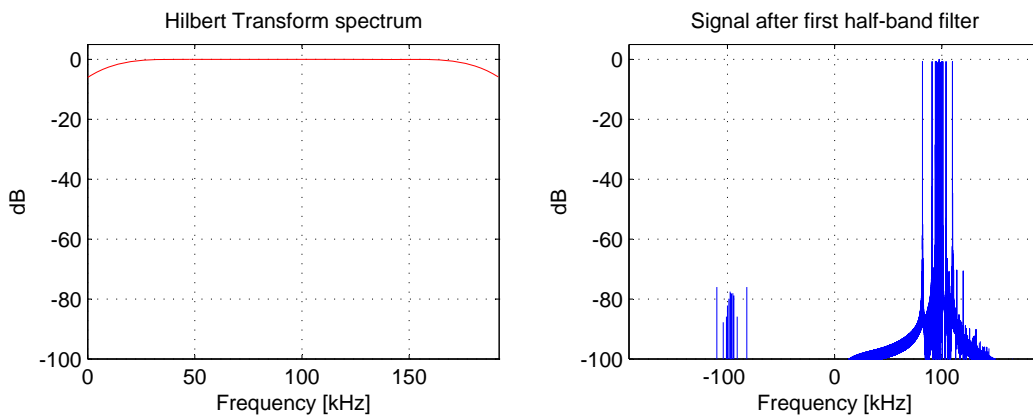


Figure 12: Left figure show the spectrum of the Hilbert transform, and the right the output since the Hilbert transform, where the negative frequency are not completely zeroed out but attenuated 80 dB.

## Down-conversion

To avoid an expensive solution, we take advantage of using the quadrature effect when doing the down-conversion. This will move the wanted channel down to base-band, as shown in figure 13.

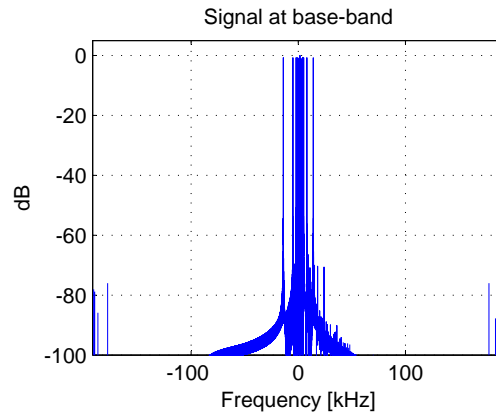


Figure 13: The quadrature down-conversion locate the wanted channel at baseband

## Down-sampling

From section 0.3 we use the property of the half-band filter by down-sampling by a factor of two in each section. Since the incoming sample rate is four times the intermediate frequency, we need to down-sample by a factor of 32 to obtain an output of 12 kHz. This can be obtained by using a cascade of five half-band filters. When designing these half-band filters we need to pay attention to the fact that aliasing can occur if the signal not are band limited when doing the down-sampling. It is possible to down-sample by a factor of four before problem with aliasing occurs. A down-sampling by a factor of four gives an output frequency of 96 kHz which will force a band limitation before doing another 2-1 down-sampling, otherwise aliasing will occur. The recursive all-pass filter is applied to band limit the signal, so the following half-band filters can down-sample by a factor of 8.

### First Half-band filter - ( $F_{s_{in}}$ :384 kHz $F_{s_{out}}$ :192 kHz)

After the second IF the spectrum is band-limited from the analog domain, and since the sampling frequency is equal to the quadrature, no aliasing problem occurs when doing a 2-1 down-sampling with a half-band filter. The spikes seen in the positive frequency spectrum in the right figure 14 are aliasing from negative frequency spectrum that the Hilbert transform did not remove entirely.

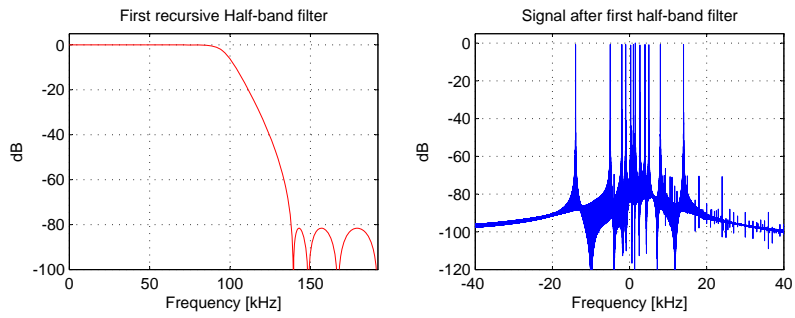


Figure 14: First half-band filter spectrum, and the output from the filter.

### Second Half-band filter - ( $F_{s_{in}}$ :192 kHz $F_{s_{out}}$ :96 kHz)

We can simply apply another half-band filter without care about aliasing since the half incoming frequency is still higher than the spectrum. Figure 15 shows the output spectrum from the filter as well as the filter spectrum.

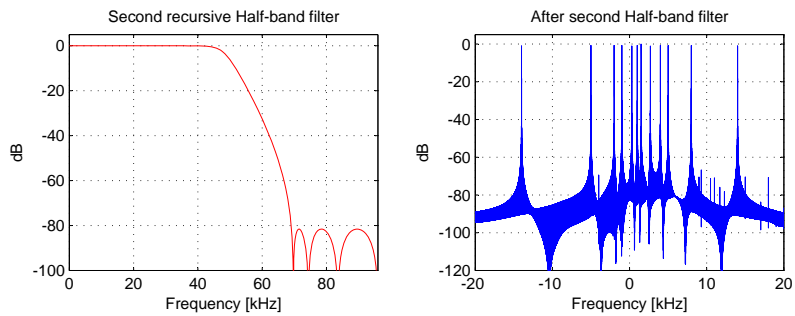


Figure 15: The left figure shows the spectrum of the half-band filter, and the right the output.

### Recursive All-pass filter - ( $F_{s_{in}}$ :96 kHz $F_{s_{out}}$ :96 kHz)

To lowering the sampling frequency by another factor of 2, we need to band-limit the spectrum to avoid aliasing problems. This is done with the all-pass filter that has a low complexity and is still able to have a high out of band attenuation. In figure 16, the spectrum shows the low pass-band and large stop band attenuation for the recursive all-pass filter. We also see that the channels located above the wanted is attenuated, and the only frequency components present are located at  $f_c-1$  kHz and  $f_c-2$  kHz. The small spikes shown are our unwanted frequency components that are attenuated below 80 dB.

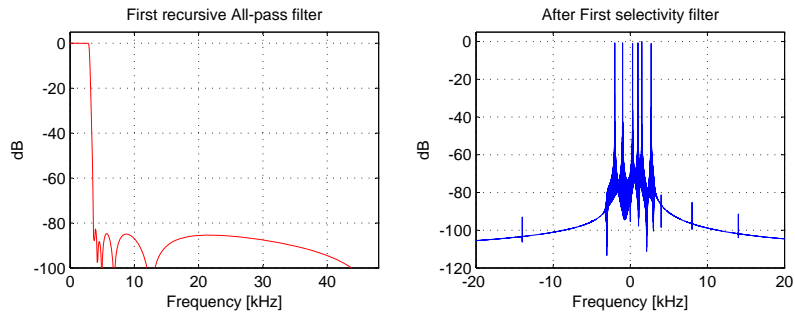


Figure 16: The left figure show the spectrum of the recursive All-pass selectivity filter, and the right the output.

### Third Half-band filter - ( $F_{s_{in}}$ :96 kHz $F_{s_{out}}$ :48 kHz)

The spectrum is now band-limited, so the requirements to the half-band filters are less demanding. The filter spectrum and output from the third half-band filter can be seen in figure 17.

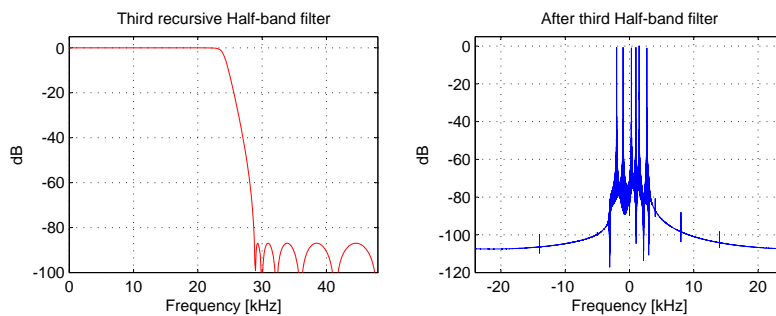


Figure 17: The left figure show the spectrum of the recursive All-pass selectivity filter, and the right the output.

### Fourth Half-band filter - ( $F_{s_{in}}$ :48 kHz $F_{s_{out}}$ :24 kHz)

Figure 18 shows the spectrum and the output of the filter. We see that the unwanted frequency components are almost invisible at this point in the chain, but the two frequency components below the wanted channel are still present.

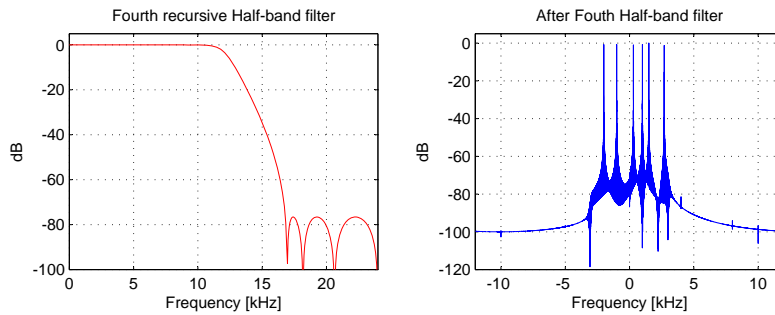


Figure 18: The left figure show the spectrum of the half-band filter, and the right the output.

### Fifth Half-band filter - ( $F_{s_{in}}$ :24 kHz $F_{s_{out}}$ :12 kHz)

The fifth filter is doing the last down-sampling, the output frequency is at this stage 12 kHz, which was the desired. However looking at the spectrum in the right figure 19 we see that two unwanted frequency component is still present, and need to be removed.

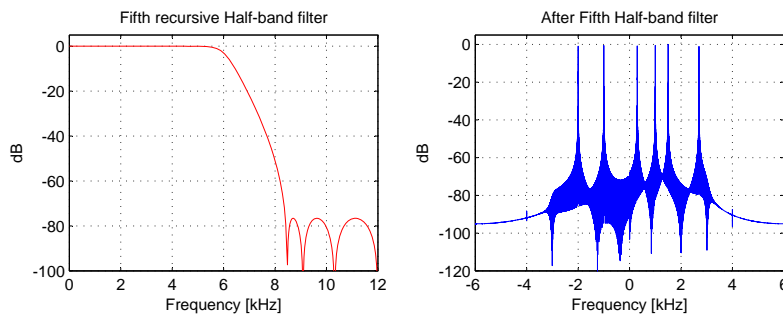


Figure 19: The left figure show the spectrum of the half-band filter, and the right the output.

### Selectivity filter - ( $F_{s_{in}}$ :12 kHz $F_{s_{out}}$ :12 kHz)

The last filter is applied to obtain the final selectivity in the chain, we have complex down-converted the signal, follow by down-sampling. The complex down-conversion leaves us with an asymmetric frequency spectrum, where the wanted channel is located from 300 to 2700 Hz, all that is located below and above this band is unwanted and needs to be attenuated. We need to design a filter that has an asymmetric frequency spectrum, the easiest way to do so is to design a low-pass filter with the cut-off frequency at half the bandwidth of the wanted signal. The low-pass filter is real, but we simply modulate the filter coefficients by a complex phase rotation so the center is located  $(f_1-f_2)/2$ , this leaves us with a complex band-pass filter as shown in figure 20. It can be argued that a complex IIR filter is more efficient. However, the sample rate is reduced to only 12 kHz, we don't consider time as a time constraint at this stage.

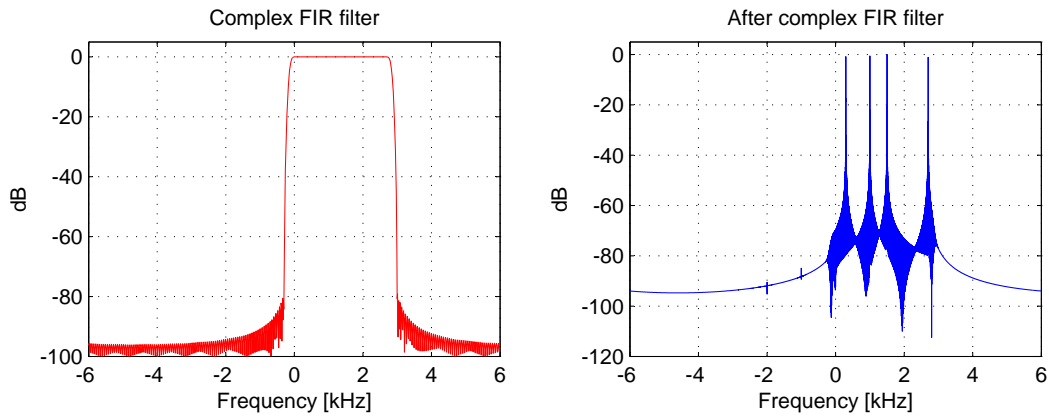


Figure 20: The left show the asymmetric spectrum of the complex band-pass filter, at the right only the wanted frequency component is present now.

## 0.6 Output result

The real output from the algorithm is shown in figure 21, where all the four frequency components still are present and the out of band attenuation below 80 dB. The final performance test is done by the company xxxx in order to verify the quality of the speech signal though the chain.

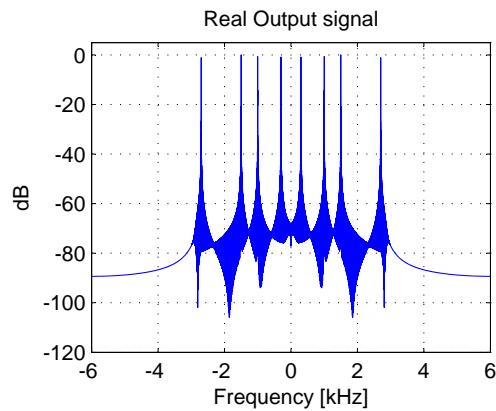


Figure 21: Real output spectrum.

## Quality Test

A quality test on the system has been made at algorithm stage. The three SINAD measurement was performed in order to verify the specification from ETSI. A 1 kHz sinus was modulated with the carrier ( $f_c$ ) as the wanted channel, the adjacent channel was located as stated in table 5. The adjacent channel was applied with an amplitude 50 dB above the wanted channel.

Test nr.	Test signal	SINAD	Distortion
Test-1	$((f_c-1)+(f_c+4))$ kHz	40 dB	N/A
Test-2	$f_c-2$ kHz	40.8 dB	0.9%
Test-3	$f_c+4$ kHz	45.3 dB	0.54%

Table 5: Results from quality test performed by xxxx

The specification stated that the SINAD measurement should be above 20 dB, and with a distortion of maximum 1%. The performance test shows that the adjacent channel located above the wanted in the frequency spectrum performed best with 45.3 dB SINAD. This was expected as shown in figure 19 and 20. The test verified the algorithm, and we can now consider the implementations possibilities in the design space exploration.

## 0.7 Sub-conclusion

The simulation shows that the algorithm is able to demodulate a SSB signal and still fulfil the requirements from ETSI. There are several solutions to demodulate the signal with multirate signal processing, this solution has tried to combine different filter type and obtaining a low complexity compare to the known legacy solution. In this solution the company is able to adapt the algorithm to fulfill their requirements, if it turns out that the sampling frequency is too high it can be lowered by a factor of two, without any lost performance since a factor 2 down-sampling is inherent in the Hilbert transformation, if it turns out the Hilbert transform not is efficient enough and the down-sampling is not needed, the company should look into the design of a recursive Hilbert transform where an out of band attenuation of 80 dB, can be achieved with only 4 coefficients, which will lower the complexity sufficiently. The quadrature solution for down-conversion was the obvious choice for down-conversion, since this is a matter of shift between the I-channel and Q-channel, and changing the sign of the channels. The recursive two-path half band filter makes it possible to lower the complexity since half the coefficients is zero, however the filter should be able to obtain a high out of band attenuation, and by utilizing the noble identity we split it into a poly-phase decomposition that makes the 2-1 down-sampling cost free. If these recursive half band filter is unwanted they can be replaced by a FIR filter decomposed into a poly-phase structure, this increase the complexity but avoid the feed back loop inherent in the recursive structure. One of the interesting filters in this algorithm is the recursive all-pass filter. This filter should be compared to IIR filter like the Elliptic, Chebyshev e.g., the complexity in this filter is lower than in the traditional IIR filter which makes it suitable when we have a time constraint.

In the current algorithm we have purely chosen to do the down-sampling with half band filters, which would require a final filtering to obtain the wanted selectivity, the filter needs to be complex since the spectrum is asymmetric. We have done the complex filtering with a modulated low-pass filter centred at half the band-width of the wanted spectrum.

The SINAD quality test of the algorithm was performed and the result is stated in table 5. The results showed that the overhead was above 20 dB SINAD, which indicates how effective the multirate rate filters can be.

## 0.8 Wordlength analysis

The disadvantage by using a DSP based implementation in a multirate base system is due to the need for FIFO buffers within between each block. For this reason an FPGA will be an obvious choice for prototyping development, as stated below.

- Variable clock rate
- Different word length
- No, control calculations

The specification were fulfilled, and we will start to optimize the algorithm to fit into a suitable architecture. In this algorithm the selectivity and sensitivity are the two important parameters, which have the highest priority when we compare it to the wanted performance, this mean that we accept a larger area in cost of performance.

There are two parameters that influence the area: the number of functional units and the word-length of the calculations used in each intermediate operation. Since the system is based on several recursive blocks a low number of coefficient, it would be an interesting parameter to see the performance of the blocks at different word length.

In an FPGA implementation, each variable can be customised to produce the best tradeoffs in numerical accuracy, design size, speed and power consumption, which is useful in a multirate system.

By using the word length analysis we will express the cost function purely dependent of the number of bits used in the total system. The new cost function can be seen in equation (8), and are the sum of bits used to present each filter block.

$$\text{Cost}_{\text{bit}} = \sum_{i=1}^M b_i \quad (8)$$

To simulate a fixed point hardware environment with different word-length in each block we need a high level league like SystemC.

## 0.9 SystemC

When simulating the algorithm in Matlab, the precision is approximate infinite. Unfortunately the hardware cost follow the precision trend toward infinity. In order to minimize cost in a FPGA based system, we want to see the performance of the algorithm, in a fixed point hardware environment. To model the behaviour of fixed point hardware, the tool SystemC[5] is used, which is a hardware description language like VHDL and Verilog, but is more aptly described as a system description language.

In SystemC it is possible to model the fixed point bit accurately for each intermediate variable in the functions used within the algorithm. Furthermore, SystemC supports features like modelling difference quantization mode, and overflow behaviour at a high level. This tool makes it possible to simulate a hardware environment, that enable us to see the performance of the algorithm at different word length.

Two data types are used to model the fixed point hardware architecture, one signed and one unsigned. The presentation of the fixed point word length is defined as expressed in equation (9) for signed and (10) for unsigned. These arguments are static and must be known at compile time.

$$wl_{\text{signed}} = [-2^{(iwl-1)}, 2^{(iwl-1)} - 2^{-(wl-iwl)}] \quad (9)$$

$$wl_{\text{unsigned}} = [0, 2^{(iwl)} - 2^{-(wl-iwl)}] \quad (10)$$

There are four parameters associated with these data class types, these are listed below:

- *wl* - Total word length, used for fixed point representation. Equivalent to the total number of bits used in the type.
- *iwl* - Integer word length - specifies the number of bits that are to the left of the binary point(.) in a fixed point number.
- *q\_mode* - quantization mode, this parameter determines the behavior of the fixed point type when the result of an operation generates more precision in the least significant bits than is available as specified by the word length and integer word length parameters.
- *o\_mode* - overflow mode, this parameter determines the behavior of the fixed point most significant bits when an operation generates more precision than the most significant bits available.

## 0.10 System setup

In order to measure performance in the filter blocks, with different word length, we have to come up with a performance estimate, where comparing the high level Matlab output with the word length reduced output signal. Since we are working at a high block level, we choose to express the performance as the error or total sum of squares as expressed in equation (11). We want to minimize the error, but to set value on an acceptable error is difficult, but we would take a look at the error influence of the filter characteristic, specially with focus on the slope of the filter and attenuation in the stop band.

$$e = \sum_{i=0}^n (q(i) - h(i))^2 \quad (11)$$

The integer word length (*iwl*) is determined by applying a steep response to each block, and to avoid overflow we set a watch flag in SystemC to observe if this occurs. The number of *iwl* bits for each section can be seen in table 6. This parameter is fixed and we vary the *wl* from thirty-two to ten bits, with a interval of two.

The hardware accumulator used in the simulation is set to infinity, but this can of cause be simulated with a fixed point bit number if wanted.

## 0.11 Simulation results

The total sum of squared error depend on the different word length as shown in figure 22. As expected the FIR filter blocks (HT & Complex filter) error are low compared to the recursive

<b>Value</b>	<b>HT</b>	<b>Block1-2</b>	<b>ALL-pass</b>	<b>Block3</b>	<b>Block4-5</b>	<b>Complex FIR</b>
wl	32:-2:10	32:-2:10	32:-2:10	32:-2:10	32:-2:10	32:-2:10
iwl	2	2	3	2	2	2

Table 6: The parameter used for each block simulation in SystemC.

filters. The half-band filters 1 and 2 using only three coefficients, and have equal coefficient which implies the same squared error, as well as 4 and 5. Comparing half-band filter 1,2 with 4,5 the error is almost equal, even when the coefficient are different, but the number of coefficient used are equal, which implies almost an equal error. The longest half band filter (3) have five coefficients, here we seen and increased error compared three coefficient filters. The last filter tested is the recursive all-pass filter, in this filter block the signal is band limited, and some of the selectivity is obtain here. The filter has thirteen coefficients and the error variation is the biggest in the filter chain.

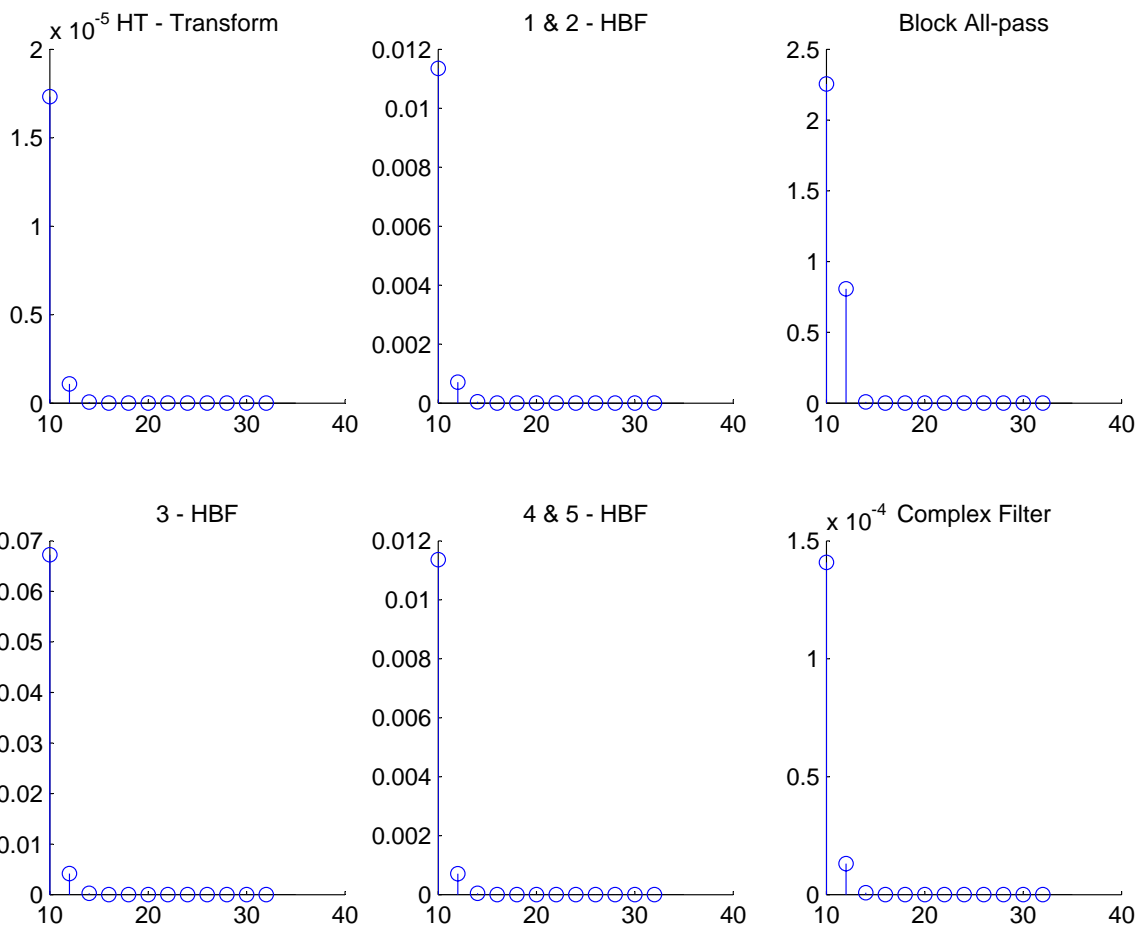


Figure 22: The graphs show the total sum off squared error, for each block. The x-axis represents the number of bits and the y-axis the error. It should be noticed that the filter coefficient in block 1 and 2 are the same, as well as in block 4 and 5.

To see a correlation between the error and the filter characteristic, the response has been plotted for each section. Figure 23 show that the transfer function is presented as wanted from 32 to 10 bit, which was expected since there is only ten coefficient, and the total sum of squared errors was small.

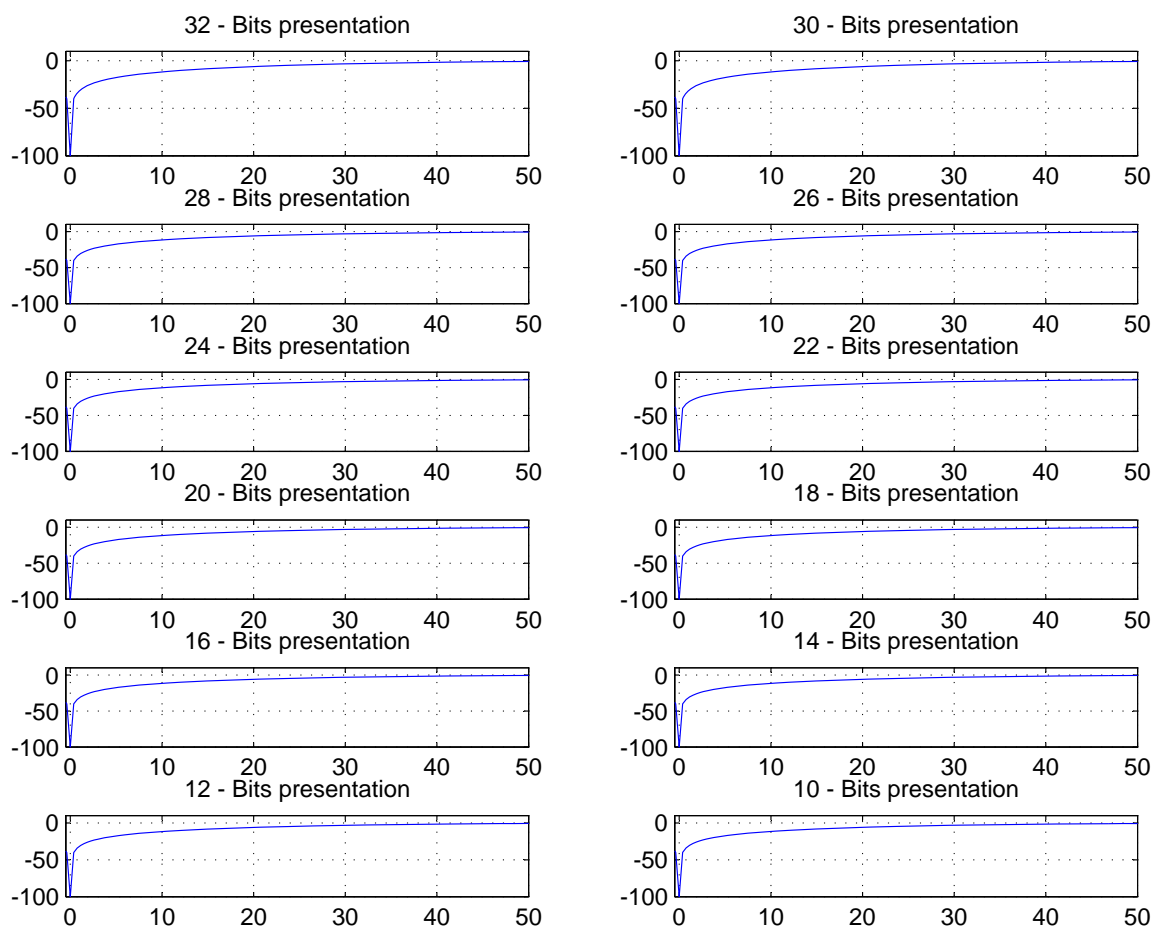


Figure 23: Filter characteristic Hilbert Transform for varying bit representation. The y-axis is in dB and the x-axis in kHz.

Figure 24 show the transfer characteristics for half-band blocks 1 and 2. The red line show the 80 dB criteria, and at least 20 bits is needed to maintain the stopband attenuation.

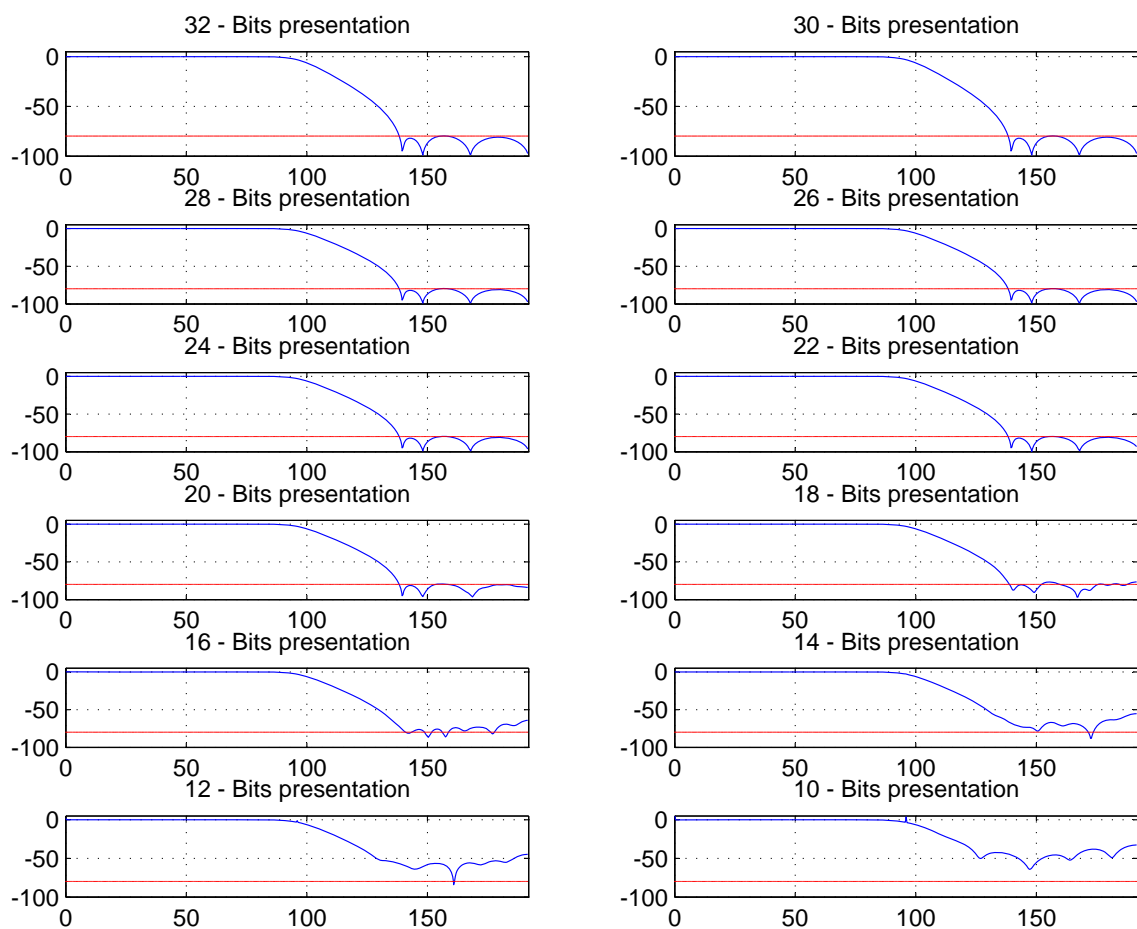


Figure 24: Filter characteristic for half pass filter block 1 and 2 for varying bit representation. The y-axis is in dB and the x-axis in kHz.

The recursive all-pass filter is the first selectivity filter used in the filter chain, and from figure 25, we see that due to the length of the filter it is more sensitive to quantization noise, than the short half-band filters. To maintain the 80 dB limit, at least 26 dB is needed.

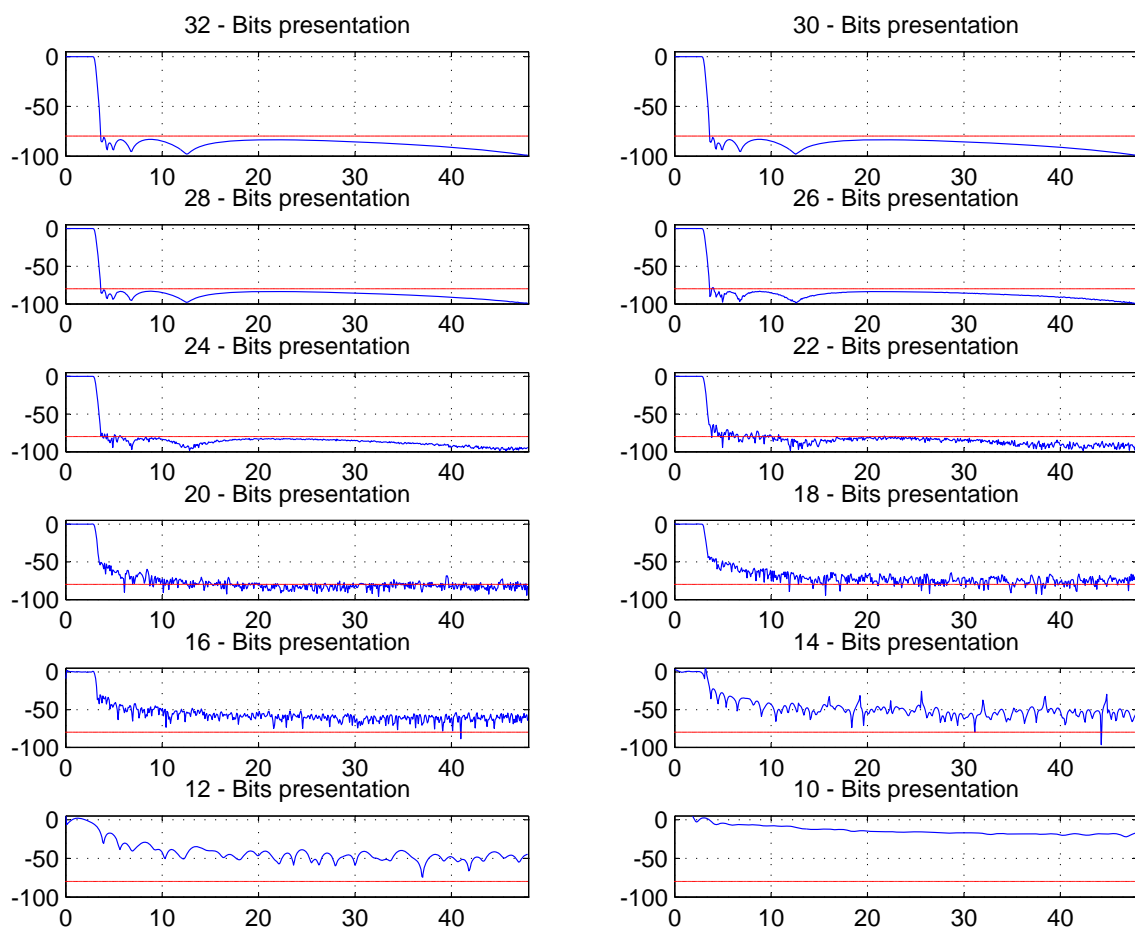


Figure 25: Filter characteristic for the recursive all-pass filter for varying bit representation. The y-axis is in dB and the x-axis in kHz.

The five coefficient half band filter need at least 20 bit to maintain the stop-band attenuation as shown in figure 26. This is equal to the representation of three coefficient half-band filters, and since the total sum of squared error only differed by a small factor, this was expected.

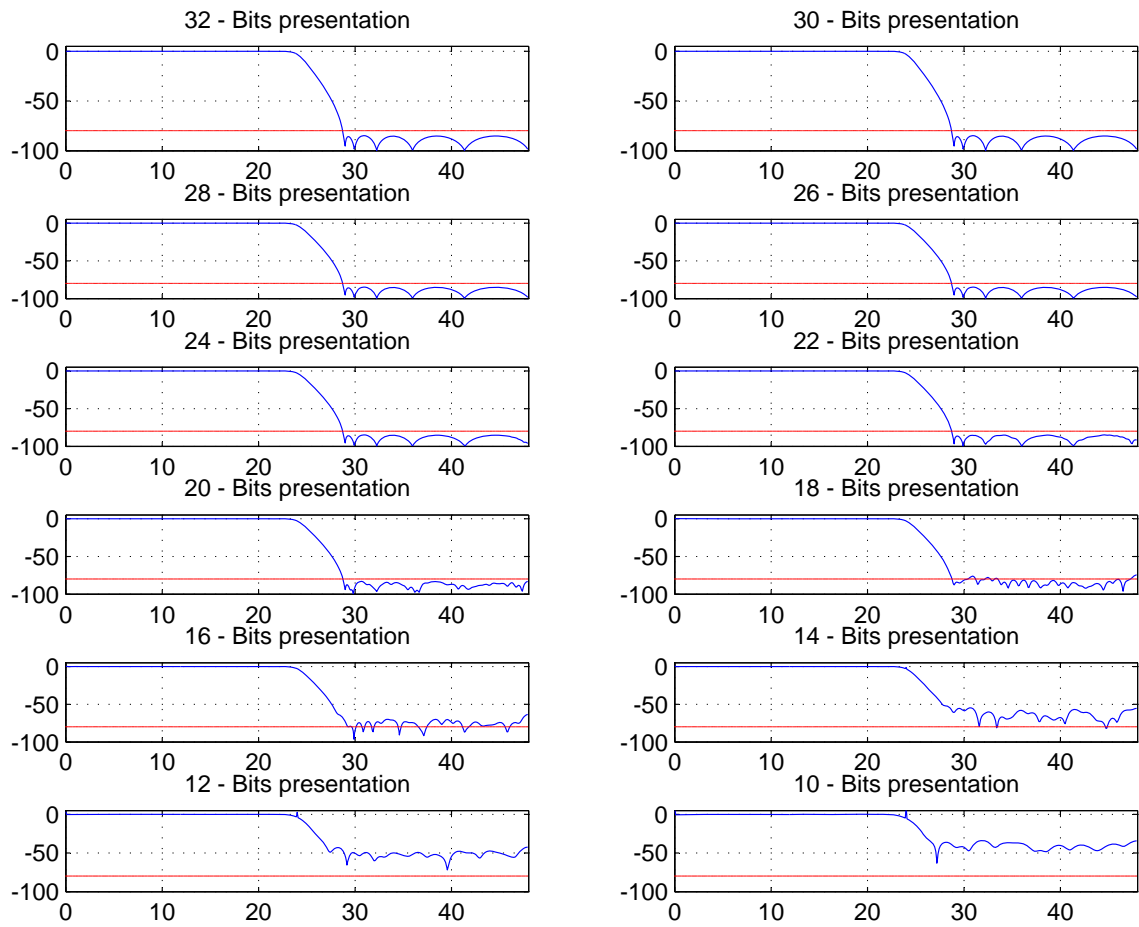


Figure 26: Filter characteristic for half pass filter block 3 for varying bit representation. The y-axis is in dB and the x-axis in kHz.

The last filter in the chain is the complex FIR, here the last selectivity is obtained. The slope on the filter has to maintain the steepness, in order to maintain the wanted selectivity. In figure 27, the simulation of the complex FIR filter shown, a word length would require 20 bit to maintain the slope of the filter, as well as the out of band attenuation.

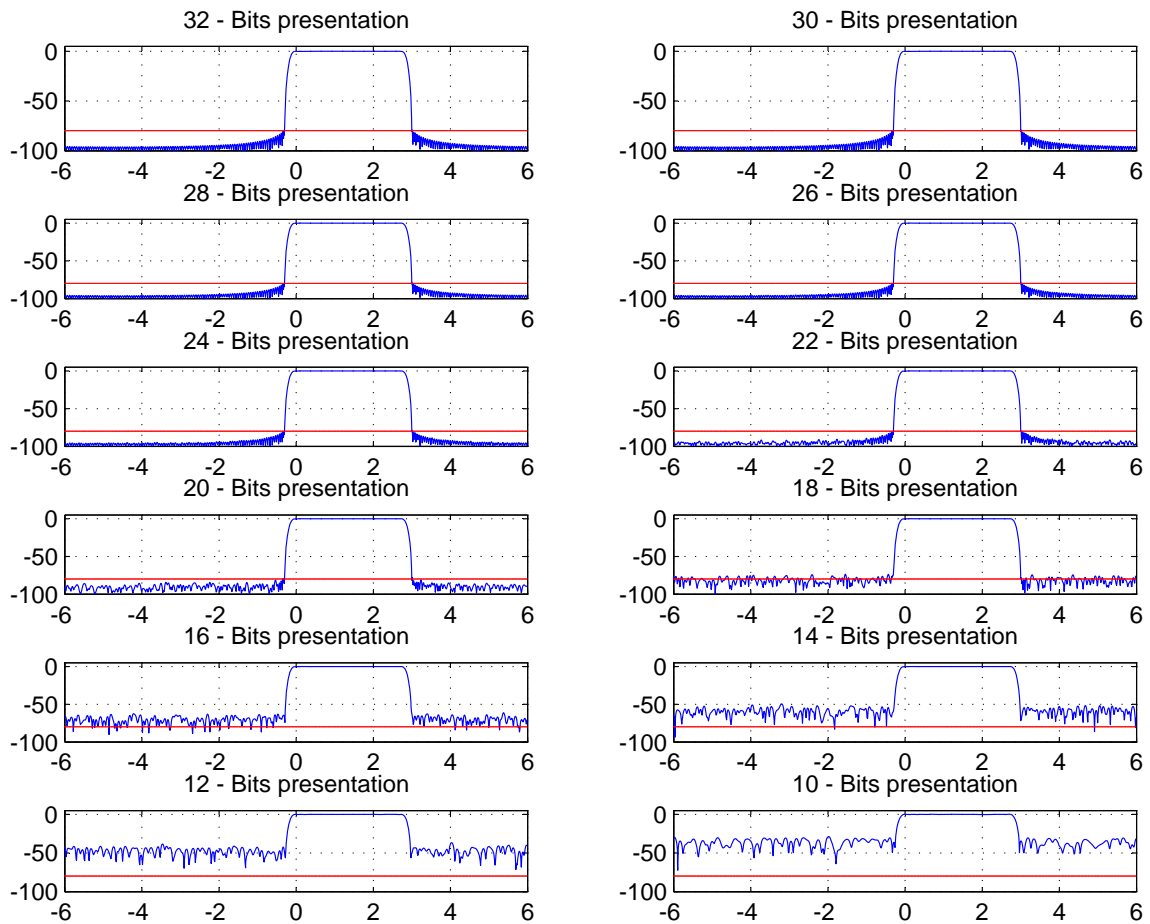


Figure 27: Filter characteristic for complex FIR filter block for varying bit representation. The y-axis is in dB and the x-axis in kHz.

## 0.12 Sub-conclusion

The simulation of the word length shows the performance of the filters at different bit representations. The recursive all-pass filter is most sensitive to quantization, which was expected since it was the longest of the recursive filters. The complex FIR filter is the most critical filter in order to obtain a high selectivity, as well as sensitivity due to the complex structure of the filter. It has been show that the number of bits can be reduced in each block, and this will reduce the cost in a hardware solution. However it should be notice that the simulation do not take care of scaling within each block, and in a final implementation of this should be done to avoid overflow. A final solution with reduced word length could be a Hilbert Transform presented with 10 bit, all the half-band section presented with 20 bit, the all-pass section with 24 bit and the complex FIR filter with 20 bit. With this combination the algorithm should still be able to fulfil the criteria specified, but this should be tested and verified before a final implementation.

# Implementation

The implementation in this work was done using the DSP Builder, which allows implementing of the system modelled in simulink. The SignalCompiler block generates the VHDL files for synthesis, hardware implementation, and simulation. To verify the fixed-point simulink simulation the output is compared with the Matlab result.

## 0.13 Block implementation

The implemented blocks design by as listed below.

- Hilbert transform: Two-path half band filter
- Down-convert: NCO MegaCore Function
- Recursive HBF1: Two-path half band filter
- Recursive HBF2: Two-path half band filter
- Recursive All-Pass: Two-path filter
- Recursive HBF3: Two-path half band filter
- Recursive HBF4: Two-path half band filter
- Recursive HBF5: Two-path half band filter
- Complex Filter: FIR filter

Each block was designed as a sub-system in Simulink and Mask in order to generate the VHDL code with the SignalCompiler. Each block was test independenly, as shown in figure 28.

### First Recursive Half Band Filter

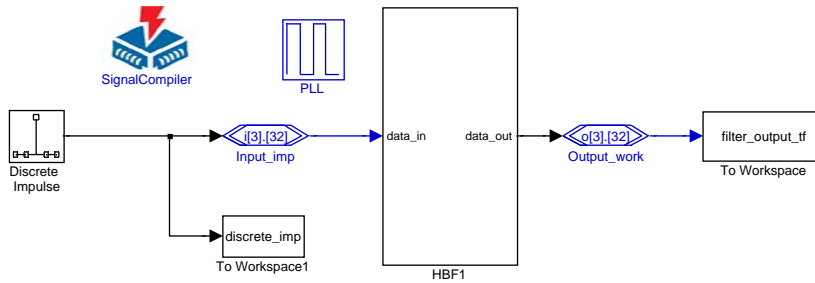


Figure 28: Black box test of the first half band filter.

Figure 29, show the sub-system for the first half band filter, and within the sub-system three All-pass sub-system is implemented. The output from each intermediate variable was save in Matlab's workspace in order to verify the result.

### Sub-system of the first recursive half band filter

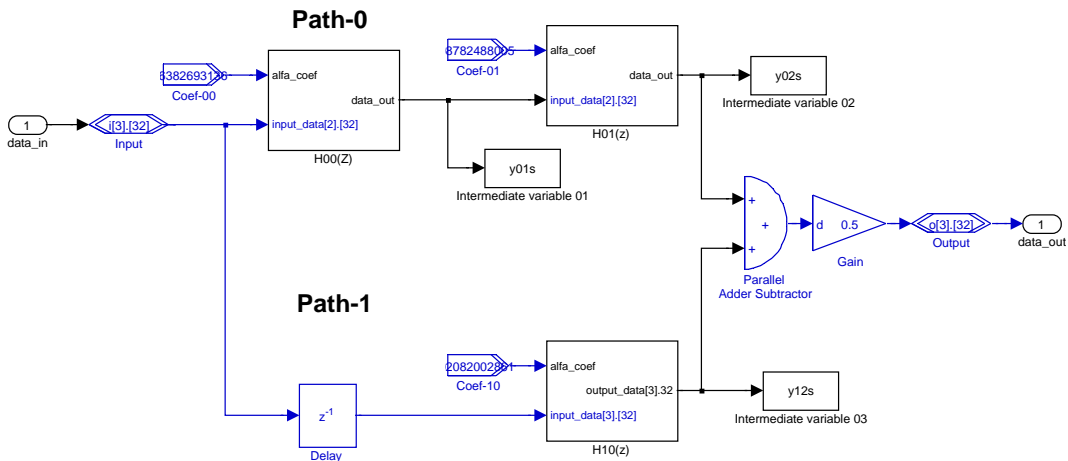


Figure 29: The sub-system for the first half band filter, notice that the down-samplings part is omitted in this test implementation.

The entire block system is implemented in simulink/DSPBuilder as shown in figure 10 at different word length, the result has then been compared with the SystemC simulation as shown in section 0.11.

## **0.14 Sub-conclusion**

The DSPBuilder has shown to be a quick development tools that in co-operation with simulink is capable of developing fast prototypes for a communication. The interaction with Matlab makes it easy to verify the result obtained from DSPBuilder and simulink with the simulated results in Matlab.

The next step in this work is to test the system in a real-time communication system in co-operation with the company xxxx, and give them an introduction to multirate filtering system together with the implementing tools used within this project. The outcome from this should be, to show the possibility to develop software defined radios in less time than earlier with the efficient hardware component and software tools available today.

# Bibliography

- [1] M.W. Chamberlain.  
A software defined hf radio.  
In *Military Communications Conference, 2005. MILCOM 2005. IEEE*, pages 2448–2453 Vol.4, 17-20 Oct. 2005.
- [2] fred harris.  
*An Efficient Constant-Q Spectral Analyzer Architecture Using All-pass Recursive Filters.*  
PhD thesis, Electrical and Computer Engineering Department San Diego State University, 2000.
- [3] fredric j harris.  
*Multirate Signal Processing For Communication Systems.*  
Prentice Hall, 1st edition, 2004.
- [4] Jung Ko, V.C. Gaudet, and R. Hang.  
Tier 3 software defined am radio.  
In *System-on-Chip for Real-Time Applications, 2005. Proceedings. Fifth International Workshop on*, pages 257–261, 20-24 July 2005.
- [5] C++ open source language.  
Systemc community.  
[www.systemc.org](http://www.systemc.org), 2007.
- [6] Walter HW Tuttlebee.  
Advances in software-defined radio.  
*Electronics Systems and Software*, pages 26–31, 2003.
- [7] R.A. Valenzuela and A.G. Constantinides.  
Digital signal processing schemes for efficient interpolation and decimation.  
*IEE Proc. Part G*, 130(6):225–235, Dec. 1983.