Laboratory exercise 3
Decimator Filter

1 Goal:

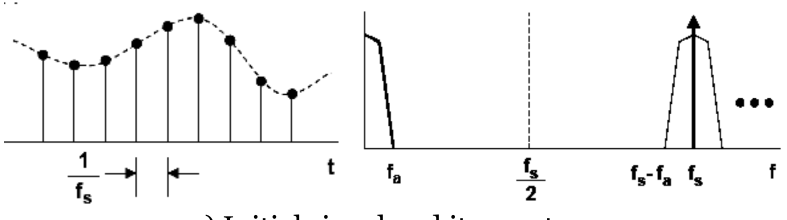
The goal is to achieve knowledge and practical experience in design of Decimator Filters for modern application specific computers, to get programming and debugging experience in VHDL language.

2 Theoretical information

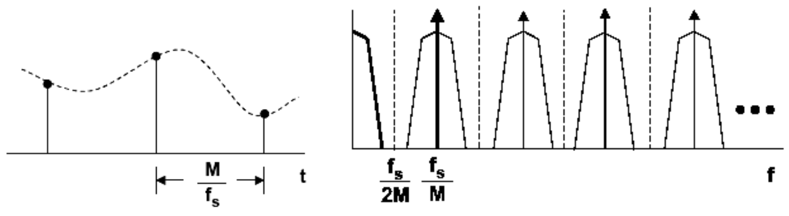
2.1. Decimator filters

There is a set of applications where the exchange of the quantization frequency of the digital signal is needed. The most common methods of such a transform are decimation (decrease of the quantization frequency in M times) and interpolation (quantization frequency increase in L times). Decimation and interpolation coefficients (M и L) are usually natural numbers. The decimation conception is illustrated by the Fig.1. The upper diagrams show that the quantization (sampling) frequency fs supercedes dramatically the frequency which is needed to represent the signal in the frequency range 0- fa, i.e. the signal in the band fa is redundant due to the sampling frequency. This is explained by the fact that the signal in the band (fa , fs-fa) does not contain any information.

Lower diagram (b) shows the same signal but with the sampling frequency which is lower in M times. Here the information is safe because the signal frequency bands before and after decimation are the same. But when we increase the coefficient M then in this example the information will be lost partically because the frequency aliasing effect will occur.



a) Initial signal and its spectrum



b) Signal after decimation in M times and its spectrum

Fig. 1. Signal decimation explanation

The process of the sampling frequency decrease is named as a downsampling. The example above shows that the downsampling must be preceded by the low pass filtering, which makes the signal spectrum like in the Fig. 1(a). Therefore, in the practical applications, downsampling has two stages: low pass filter (LPF) and decimation unit, shortly, decimator. Such an device is named as the decimation filter. Below we will consider some simple schemes of the decimation filters.

In QAM, to select the correct information the decimation filters are usually used at its I/Q outputs as well.

2.2. Decimator filter based on comb filter

A comb filter is an oddsymetric FIR filter described by the equation

$$y[n] = x[n] - x[n - RM]$$

Here M is a design parameter and is called the differential delay. M can be any positive integer, but it is usally limited to 1 or 2. The corresponding transfer function at fs is

$$H_C(z) = 1 - z^{-RM}$$

$$\begin{aligned} |H_C(e^{j\omega})|^2 &= 2(1-\cos RM\omega) \\ \mathrm{ARG}[H_C(e^{j\omega})] &= -\frac{RM\omega}{2} \end{aligned}$$

When R = 1 and M = 1, the power response is a high-pass function with 20 dB per decade (6 dB per octave) gain. When $RM \neq 1$, then the power response takes on the familiar raised cosine form with RM cycles from 0 to 2π. Then the power response function remains a comb by the form, which is the root of the filter name. The comb filter schema is shown in the Fig.2.

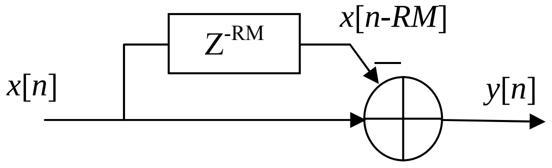


Fig.2.Comb filter schema

This filter response function has short transient band pass. But to transfer it to the LPF characteristic, RM-1 wavelets of it must be removed. This removing can be implemented by addition of an integrator.

An integrator is simply a single-pole IIR filter with a unity feedback coefficient:

$$y[n] = y[n-1] + x[n]$$

This system is also known as an accumulator. The transfer function for an integrator on the z-plane is

$$H_I(z) = \frac{1}{1-z^{-1}}$$

$$\begin{aligned} |H_I(e^{j\omega})|^2 &= \frac{1}{2(1-\cos\omega)} \\ \mathrm{ARG}[H_I(e^{j\omega})] &= -\tan^{-1}\left[\frac{\sin\omega}{1-\cos\omega}\right] \end{aligned}$$

The power response is basically a low-pass filter with a -20 dB per decade (-6 dB per octave) rolloff, but with infinite gain at DC. This is due to the single pole at z = 1; the output can grow without bound for a bounded input. In other words, a single integrator by itself is unstable. The integrator schema is shown in the Fig.3.

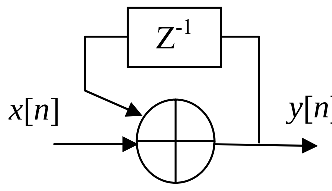


Fig.3. Integrator schema

When the integrator and the comb filter are connected sequentially, then the pole of the integrator is compensated by the zero of the comb filter. The resulting transfer function is

$$H(Z) = \frac{1-Z^{-RM}}{1-Z^{-1}}$$

In the Fig.4. the transfer function of such a filter for RM=8 is shown. Note, that the output signal magnitude is increased in RM times. Therefore, to prevent overflows and to get the proper output magnitude, the accumulator register bit width must be increased in]$\log_2(NM)$[times.

The suppression level of such a filter is rather small, it is equal to -12 db. It can be increased by connecting 2,3 or more such filter stages in a chain. The resulting two staged decimator filter is shown in the Fig.5. Here R with the down arrow does decimation in R times.

Due to the fact that this filter gives the same result that the running sum does, this filter has the synonym the running sum filter.

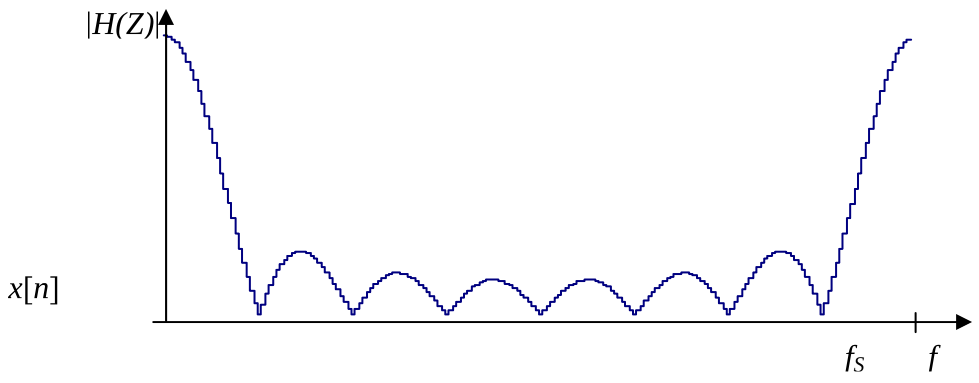


Fig.4.

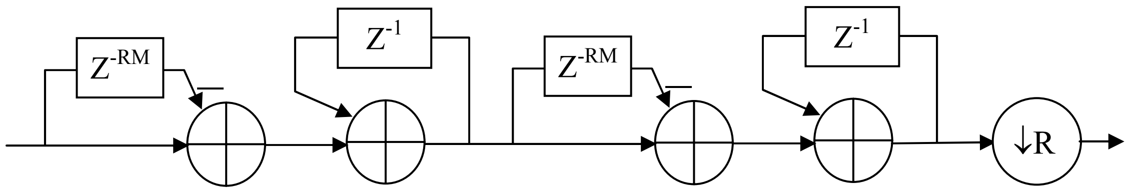


Fig.5

2.3. Cascaded integrator comb filter

Due to its name, the cascaded integrator comb filter, or CIC filters for short, are based on the integrators and comb filters introduced above. This kind of filters is not used separately from the decimation, because it was invented for this purpose. CIC decimator has N cascaded integrator stages clocked at *fs*, followed by a rate change by a factor R, followed by N cascaded comb stages running at *fs*/R . Fig.5. illustrates N = 3 Stage Decimating CIC Filter.

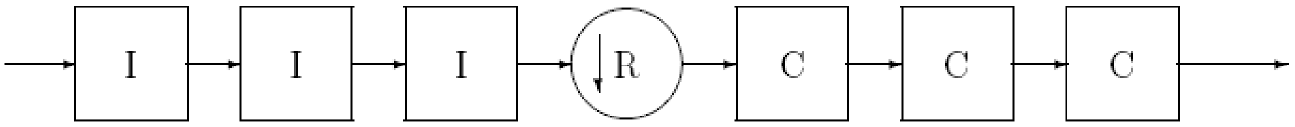


Fig.5.

Here the I unit means an integrator and C unit does a comb filter stage. The transfer function for a CIC _lter at fs is

$$H(z) = H_I^N(z) H_C^N(z) = \frac{(1 - z^{-RM})^N}{(1 - z^{-1})^N} = \left(\sum_{k=0}^{RM-1} z^{-k} \right)^N$$

This equation shows that even though a CIC has integrators in it, which by themselves have an infinite impulse response, a CIC filter is equivalent to N FIR filters, each having a rectangular impulse response (running sum filters). Since all of the coeffcients of these FIR filters are unity, and therefore symetric, a CIC filter also has a linear phase response and constant group delay. The magnitude response at the output of the filter can be shown to be

$$|H(f)| = \left| \frac{\sin \pi M f}{\sin \frac{\pi f}{R}} \right|^N$$

By using the relation sin$x \approx x$ for small x and some algebra, we can approximate this function for large R as

$$|H(f)| \approx \left| RM \frac{\sin \pi M f}{\pi M f} \right|^N \text{ for } 0 \leq f < \frac{1}{M}$$

Fig.6 illustrates this function for N=2, R=4, and M=2 with the logarithmic axis

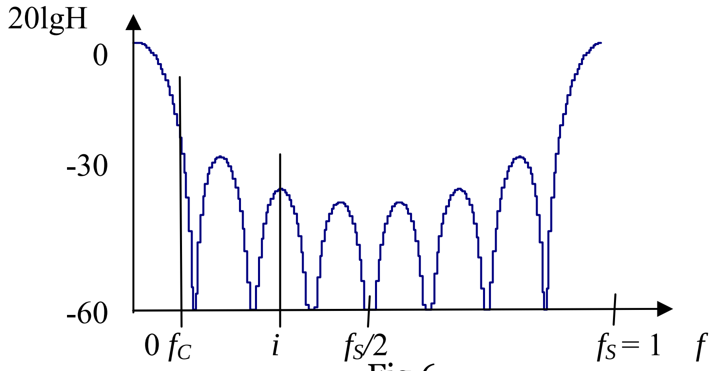


Fig.6.

We can notice a few things about the response. One is that the output spectrum has nulls at multiples of f = 1/M . In addition, the region around the null is where aliasing/imaging occurs. If we define fc to be the cutoff of the usable passband, then the aliasing/imaging regions are at

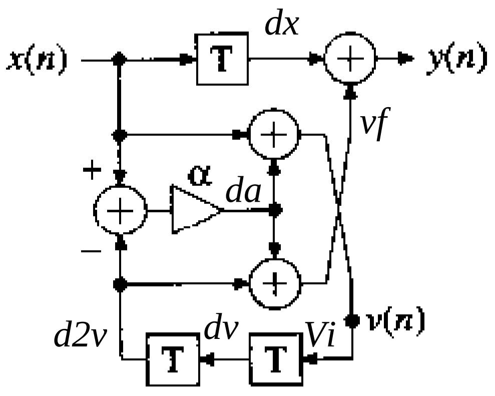
$$(i - fc) \leq f \leq (i + fc)$$

for $f \leq 1/2$ and i = 1; 2; ...]R/2[. If $fc \leq M/2$, then the maximum of these will occur at the lower edge of the first band, 1 – fc. The system designer must take this into consideration, and adjust R, M, and N as needed. Another thing is that the passband attenuation is a function of the number of stages. As a result, while increasing the number of stages improves the imaging/alias rejection, it also increases the passband “droop." We can also see that the DC gain of the filter is a function of the rate change.

Since their inception, CIC filters have become an important building block for DSP systems. They have found a particular niche in digital transmitters and receivers. They are currently used in highly integrated chips from Analog Devices, Infineon, Freescale as well as other manufacturers and custom designs.

2.4. Bireciprocal wave digital filter

The signal-flow graph of the bireciprocal wave digital filter is shown in Fig. 7, where the coefficient a is 0.375 = $(0.011)_2$. The filter is a bireciprocal (half-band) low pass filter, which also can be modified into a high pass filter by changing the last adder to a subtracter. It is also suitable for use in interpolation and decimation, where the filter can operate at the lower of the two sample rates. The attenuation of the filter is shown in Fig. 8.



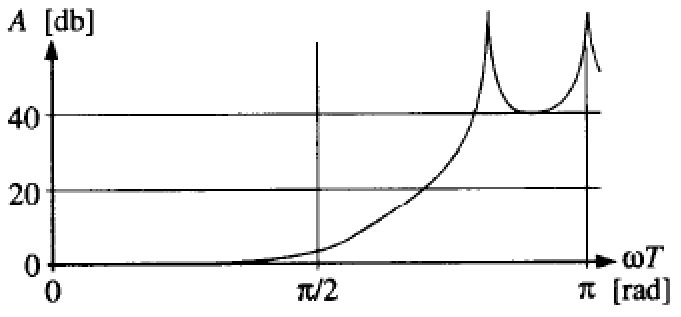


Fig.7

Fig.8

Due to the fact that this filter belongs to the set of wave digital filters, it is stable for any coefficients. Moreover, because of its simple and single coefficient its complexity is very small. Such an coefficient can be implemented using a single adder which adds the multiplicand shifted to 2 and 3 bits right.

A single filter stage can provide the downsampling to R=2. To achieve the downsampling to 4,8,16 etc. 2,3,4,... such downsampling stages are connected in a chain.

3. Decimator Filter design example

Consider the example of the filter design based on the bireciprocal wave digital filter. The downconversion factor is R=4. Input and output bit widths are Ni= No=10. The filter is based on the combinational multiplier.

Because a single stage makes decimation only in two times, we have to design 2 staged filter. Each next stage operates in two times slower than the previous does. Therefore the stage can be arranged as the entity, which has an additional enable input port. Such an entity is described as the following:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_SIGNED.all;
entity DECIM2 is port( CLK : in STD_LOGIC;
       RST : in STD_LOGIC;
       EI : in STD_LOGIC;                  --operation enable
       X : in STD_LOGIC_VECTOR(9 downto 0);
       EO : out STD_LOGIC;          -- enable signal for the next stage
       Y : out STD_LOGIC_VECTOR(9 downto 0) );
end DECIM2;
architecture BEH of DECIM2 is
 constant a:STD_LOGIC_VECTOR(3 downto 0):="0011";
 signal dx:STD_LOGIC_VECTOR(9 downto 0);
 signal v, dv,d2v,yf,yi:STD_LOGIC_VECTOR(11 downto 0);-- inner signals of
 --                                                    increased precision
 signal da: STD_LOGIC_VECTOR(15 downto 0);
 signal     EOi:std_logic;
begin
                             --arithmetical calculations
 da<=(x - d2v)*a;
 v<=x+da(14 downto 3);
 yf<=d2v + da(14 downto 3);
                             -- inner registers
 RR:    process(CLK,RST) begin  --
       if RST='1' then
             dx<=(others=>'0');
             dv<=(others=>'0');
             d2v<=(others=>'0');
             yi<=(others=>'0');
             EOi<='0';
       elsif CLK='1' and CLK'event then
             if EI='1' then    -- operation enable
                   dx<=x;
                   dv<=v;
                   d2v<=dv;
                   yi<=dx +yf +1;    --result with rounding
                   EOi<=not EOi;   -- enable output
             end if;
       end if;
 end process;
 Y<=yi(10 downto 1);
 EO<=EOi;
end BEH;
```

The behavioral description consists of two distinct parts. First of them describes the arithmetical calculations. Second of them describes the registers. All the signal dependences represent the data flow graph illustrated by the Fig.7. Here signals `dx,dv,d2v` represent the registered delays, `yi` represents the inner result. The resulting signal `y` is derived from `yi` by the right shift to a single bit to put it in the proper signal range.

The signal EO is equal to a 1 when the output signal Y is valid. It serves to strobe this signal in this signal destination unit.

Two stages of such a filter look as in the Fig.9:

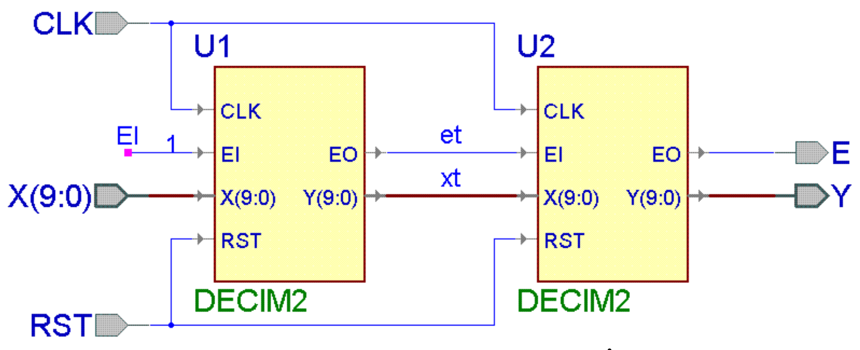


Fig.9.

This filter is described by the structural style as the following:

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
entity DECIM4 is port( CLK : in STD_LOGIC;
       RST : in STD_LOGIC;
       EI : in STD_LOGIC;                    --operation enable
       X : in STD_LOGIC_VECTOR(9 downto 0);
       EO : out STD_LOGIC;            -- enable signal for the next stage
       Y : out STD_LOGIC_VECTOR(9 downto 0) );
end DECIM4;
architecture BEH of DECIM4 is
 signal xt:STD_LOGIC_VECTOR(9 downto 0);
 signal et:std_logic;
 component  DECIM2 is port( CLK : in STD_LOGIC;
             RST : in STD_LOGIC;
             EI : in STD_LOGIC;                    --operation enable
             X : in STD_LOGIC_VECTOR(9 downto 0);
             EO : out STD_LOGIC;            -- enable signal for the next stage
             Y : out STD_LOGIC_VECTOR(9 downto 0) );
 end    component;
begin
 U1:DECIM2 port map(CLK, RST,  EI=>EI,
       X =>X,
       EO =>ET,
       Y =>xt);
 U2:DECIM2 port map(CLK, RST,  EI=>ET,
       X =>Xt,
       EO =>EO,
       Y =>Y);
end BEH;
```

4. Testbench and filter testing

To test the filter, some special signals are inputted in it and the filter reaction is investigated. The sine waves are usually selected as such input signals because they do not exchange their form but only magnitude do. This is explained by the fact that the sine wave is the eigenfunction for the linear filters. To say more precisely, the genuine eigenfunction is the analytical (complex) signal which is represented by the couple of cosine (inphase) and sine (quadrature) signals.

A set of reactions to the analytical signal with different frequencies is named as the frequency characteristic of the given filter. The magnitude-frequency characteristic and phase-frequency characteristic are usually distinguished for the filter. It is recommended to test the DSP filters by deriving these characteristics.

For this purposes the filter testbench component was designed and is proposed for the use in this laboratory exercise. The connection of this component to the tested filter entities is shown in the Fig.10.

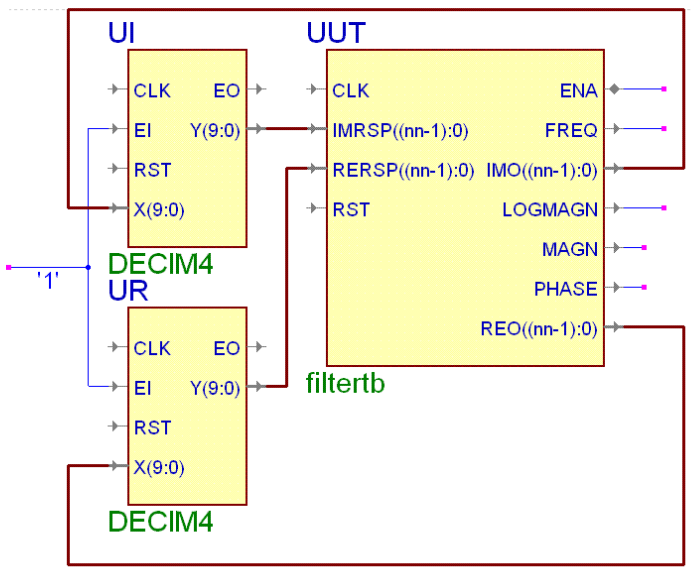


Fig.10

The component named FilterTB has the following entity declaration

```
entity FilterTB is
 generic(fsampl:integer := 1000;
      fstrt: integer:=0;
      deltaf:integer:=20;
      maxdelay:integer:=100;
      slowdown:integer:=3;
      magnitude:real:=1000.0;
      nn:natural:=16);  ---bit width

 port(CLK : in STD_LOGIC;
      RST : in STD_LOGIC;
      RERSP : in STD_LOGIC_VECTOR(nn-1 downto 0);
      IMRSP : in STD_LOGIC_VECTOR(nn-1 downto 0);
      REO : out STD_LOGIC_VECTOR(nn-1 downto 0);
      IMO : out STD_LOGIC_VECTOR(nn-1 downto 0);
      FREQ : out INTEGER;
      MAGN:out INTEGER;
      LOGMAGN:out REAL;
      PHASE: out REAL ;
      ENA: inout STD_LOGIC);
end FilterTB;
```

The generic and port meanings are shown in the following table

`fsampl`	Integer sampling frequency, for example, 1000 kHz
`fstrt`	Starting frequency fo, the first frequency which is analyzed
`deltaf`	Frequency increase d, so in k steps the generator will output the frequency fo + k*d
`maxdelay`	The delay in signal samples, after which the output signal parameters will be estimated. Usually it is slightly higher than the maximum (group) delay of the filter considered.
`slowdown`	Factor, in which the filter speed is slowed down. If the input samples enter each clock cycle then slowdown=1, if the samples go in odd clock cycles then slowdown=2, etc.
`nn`	Input and output data width
`magnitude`	Integer magnitude of the generated sine/cosine waves. For example, if nn=8 then magnitude is any positive number less than 127.
`REO,IMO`	Cosine/ sine waves, represented by the nn bit integers, outputted by the component
`RERSP,IMRSP`	Filter output signals , which are responses to the cosine/ sine waves, and which must be ported to the nn-bit width inputs
`FREQ`	Frequency code of the given sine/cosine waves in this but the previous frequency step, which is equal to fo + (k-1)*d
`MAGN`	Estimated magnitude of the signal `RERSP,IMRSP` at the frequency `FREQ`
`LOGMAGN`	Estimated magnitude of the signal `RERSP,IMRSP` at the frequency `FREQ` in the logarithmical scale, i.e. in decibels. Note, the signal with the given magnitude is 0 db
`PHASE`	Estimated phase of the signal `RERSP,IMRSP` at the frequency `FREQ` represented in the range $\pm\pi$
`ENA`	Enable signal, which strobes the filter inputs when `slowdown>1`

The magnitude-frequency characteristics which are measured by this testbench for the example written above are shown in the Fig. 11.

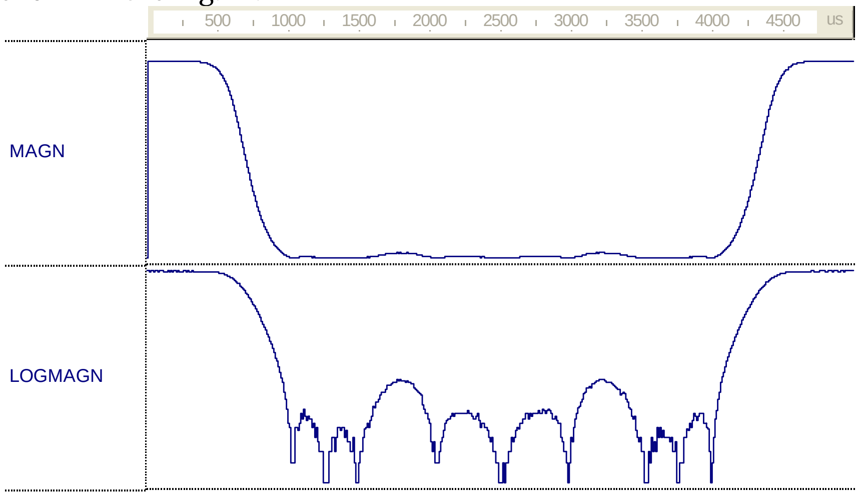


Fig.11

These characteristics show the ability of the designed filter to do properly its task. The measured suppression level is higher than 30 db, the low pass band width is equal to 0.1 f_s (at the level of -3 db), the intermediate band width is equal to 0.08 f_s . These diagrams illustrate the periodical nature of the frequency characteristics in DSP problems as well.

5. **Laboratory exercise implementation**

The decimator filter has to be built and tested as in the previous example.

Each exercise variant has a set of parameters, which are numbered by natural numbers. A set of them is derived from the record-book number of the student. Consider 3 last figures a_2,a_1,a_0, of the record-book number. Then the variant number is

$N = 100a_2 + 10a_1 + a_0 = 2^9b_9+2^8b_8+2^7b_7+2^6b_6+2^5b_5+2^4b_4+2^3b_3+2^2b_2+2^1b_9+b_0$,

where b_i are the bits of the number N in the binary representation.

The decimator parameters are input signal bit width *Ni*, output signal bit width *No*, scheme variant N_S, downsampling coefficient *R*, stage number *N*. They are selected from the Table 1 and Table 2.

Table 1

b_2,b_1, b_0,	*Ni*	*No*	N_S
000	12	10	0
001	13	10	2
010	14	12	1
011	15	12	3
100	14	14	0
101	15	14	2
110	15	16	1
111	16	16	3

Table 2

b_4,b_3, b_0,	*R*	*N*
000	4	2
001	2	1
010	8	3
011	4	2
100	10	2
101	8	3
110	12	3
111	16	4

Here $N_S = 0$ means decimator filter based on the comb filter, 1 – one based on the cascaded integrator comb filter, 2,3 – one based on the bireciprocal wave digital filter. Variants 2 and 3 are distinguished in that, that the first one is based on the multiplier to the coefficient, another one is based on the multiplication through the addition.

When the filter is designed the proper inner bit width is selected to prevent the overflows and decrease of the signal magnitude range.

6. Laboratory exercise report

The laboratory exercise report must contain:

- Goal of the work,
- Filter description,
- VHDL texts,
- Waveforms of testing,
- Conclusions.

Literature

1. Отнес Р., Эноксон Л. Прикладной анализ временных рядов. –М.:Мир. –1982. – 428 с.
2. Рабинер Л., Гоулд Б. Теория и применение цифровой обработки сигналов. –М.:Мир. –1978. – 848 с.