Laboratory exercise 4
Discriminator

1 Goal:

The goal is to achieve knowledge and practical experience in design of the discriminator for modern application specific modems, to get programming and debugging experience in VHDL language.

2 Theoretical information

The discriminator is the final part of digital data demodulators, industrial installation control loops, software radios, etc. Often it is named as a detector. Due to the input signal modulation principle, the amplitude (AM), phase(PM), amplitude-phase, or quadratur-amplitude (QAM), frequency (FM) discriminators are distinguished.

AM discriminator

AM discriminator is usually named as an AM demodulator or detector. An AM signal can be rectified without requiring a coherent demodulator. For example, the signal can be passed through an envelope detector. The output will follow the same curve as the input baseband signal.

When the input signal is an analytical one, i.e. $I + jQ$, then the AM discriminator output signal is the complex vector magnitude:

$$A = \sqrt{I^2 + Q^2}\ .$$

The square root function is rather complex to calculate. Therefore the direct calculation of the vector magnitude is implemented in discriminators rarely. But in many cases the output signal is enough to be calculated with the error ca 5%. The French mechanician Ponselet was the first who proposed the best linear approximation of this formula in 1828.

If $I > 0$, $Q > 0$, $I > Q$, then $A \approx 0.9605^*I + 0.3978^*Q$, and

if $I > 2Q$, then $A \approx 0.9859^*I + 0.2327^*Q$ and so on.

In the first situation the approximation error is less than 4%, and in the second one it is less than 1.4% .

The coefficients can be represented by the binary code as

$0.9605 \approx 0.11110110$ and $0.3978 \approx 0.0110110$,

and they are represented by the following sums of fractions

$0.11110110 = 1 - 1/16 + 1/64 + 1/128$ and $0.0110110 = 1/4 + 1/8 + 1/64 + 1/128$.

The multiplication to these coefficients can be substituted to the sum of the shifted operands $|I|$ and $|Q|$, where $|I|$ - is the absolute value of the two's complement number I. The resulting formula looks like the following:

$A \approx |I| - 1/16^*|I| + 1/64^*|I| + 1/128^*|I| + 1/4^*|Q| + 1/8^*|Q| + 1/64^*|Q| + 1/128^*|Q|$,

The function of the complex vector magnitude is described in VHDL as follows

```
function MAGN_APPROX(A,B: INTEGER) return INTEGER is
     variable Re,Im,Tmp,Magn: INTEGER;
     begin
     Re:=ABS(A);
     Im:=ABS(B);
     if Re<Im then
          Tmp:=Re; Re:=Im;Im:=Tmp;
     end if;
     Magn:=Re - Re/16+Re/64+Re/128+ Im/4 +Im/8+Im/64+Im/128;
     return Magn;
  end function;
```

Firstly the absolute values of the operands are calculated and then they are sorted to fall into the first octant of the Cartesian coordinate system i.e. to satisfy |A|>|B|. Then the Ponselet function is calculated.

FM Discriminator

There are several ways to demodulate an FM signal. The most common is to use a discriminator. This is composed of an electronic filter which decreases the amplitude of some frequencies relative to others, followed by an AM demodulator. If the filter response changes linearly with frequency, the final output will be proportional to the input frequency, as desired. Another is to feed the signal into a phase-locked loop and use the error signal as the demodulated signal.

Consider the signal modulated by the Binary Frequency Shift Keying (BFSK). Then the signal frequency is $f = f_0 - \Delta f/2$, when the symbol a 0 is received. and is $f = f_0 + \Delta f/2$, when the symbol a 1 is done. where $\Delta f T_C = 1$, T_C is the symbol period. Therefore, the discriminator must measure the difference $f - f_0$. Consider in the k-th clock cycle we have the sample vector of the signal $\boldsymbol{A}_{k-1} = I_{k-1} + jQ_{k-1} = |A|\exp(2\pi ft)$, and in the next cycle it is $\boldsymbol{A}_k = I_k + jQ_k = |A| \exp(2\pi ft+\phi)$. Then

$$\sin((2\pi ft)+\phi - 2\pi ft) = \sin(\phi) = (Q_k I_{k-1} - I_k Q_{k-1})/|A|,$$

Here $\phi = \Delta f T_S/2$ is proportional to $f - f_0$. where T_S is the sampling frequency. Then the searched frequency shift with the frequency scaling factor K can be found as

$$\Delta f = K\phi = K \arcsin(\sin(\phi)) \approx K \sin(\phi) = K(Q_k I_{k-1} - I_k Q_{k-1})/|A|.$$

When the signal magnitude is stable, for example, when it is regulated by the proper feedback, then the formula can be simplified:

$$\Delta f \approx K'(Q_k I_{k-1} - I_k Q_{k-1}).$$

The scaling factor K' can be taken into account when the decision is made, which of symbols is thransferred. Therefore, it represents some threshold value and it does not afford the multiplication.

PM Discriminator

The complex vector phase is usually calculated as $\theta = \varphi \pm \mathrm{arctg}(Q/I)$, where $\varphi = 0| \pm \pi/2$ depending on the angle quadrant. Both arctangent and division is rather complex operation to perform it in real time by hard hardware volume requirements.

Volder was the first who proposed to caclulate the phase using his algorithm named CORDIC in the 1950's. Moreover, it gives the vector magnitude in parallel with the angle. The CORDIC uses a sequence like successive approximation to reach its results. The key is it does this by adding/subtracting and shifting only.

Suppose we want to rotate a point(X,Y) by an angle(θ). The coordinates for the new point(Xnew, Ynew) are:

Xnew = X * cos(θ) - Y * sin(θ)
Ynew = Y * cos(θ) + X * sin(θ)

Or rewritten:

Xnew / cos(θ) = X - Y * tg(θ)
Ynew / cos(θ) = Y + X * tg(θ)

It is possible to break the angle into small pieces, such that the tangents of these pieces are always a power of 2. This results in the following equations:

$X_{i+1} = P_i * (X_i - Y_i 2^{-i})$
$Y_{i+1} = P_i * (Y_i + X_i 2^{-i})$
$\theta_i = \mathrm{arctg}(2^{-i})$

The arctg(2^{-i}) has to be pre-computed, because the algorithm uses it to approximate the angle. The P_i factor can be eliminated from the equations by pre-computing its final result. If we multiply all P_i 's together we get the aggregate constant. This is a constant which reaches 0.607...

Such an algorithm is fitted to calculate the arctg(Q/I) function. For this purpose the initial vector is given as $Y_0, X_0 = Q,I$. Then n iterations are performed:

sign ξ_i = sign Y_i;
$Y_{i+1} = Y_i - \xi_i X_i 2^{-i}$;
$X_{i+1} = X_i + \xi_i \xi Y_i 2^{-I}$;
$\theta_{i+1} = \theta_i + \xi_i \mathrm{arctg}(2^{-i})$.

Here sign ξ_i =1 when Y_i>0, and –1 when Y_i<0.

After n steps the results are θ_n = arctg(Q/I) and $X_n = 1/P*\sqrt{(I^2 + Q^2)}$. It is worth to be mentioned that for n steps n correct bits of the result are calculated. For this feature such an algorithm is often named as "digit by digit" algorithm.

Note that I is considered as $I>0$. When $I<0$ its absolute value must be calculated, and the resulting angle must be corrected as $\pm\pi - \theta_n$.

In discriminators it is worth to map the full angle $\pm\pi$ into the range ±1. Therefore, the arctangent constants are stored according to this fact. These coefficients for 16-bit data are represented in the following table

i	$\mathrm{arctg}(2^{-i})/\pi$	i	$\mathrm{arctg}(2^{-i})/\pi$
0	0,0100 0000 0000 0000 0	8	0,0000 0000 0101 0001 1
1	0,0010 0101 1100 1000 0	9	0,0000 0000 0010 1000 1
2	0,0001 0011 1111 0110 1	10	0,0000 0000 0001 0100 0
3	0,0000 1010 0010 0010 0	11	0,0000 0000 0000 1010 0
4	0,0000 0101 0001 0110 0	12	0,0000 0000 0000 0101 0
5	0,0000 0010 1000 1011 1	13	0,0000 0000 0000 0010 1
6	0,0000 0001 0100 0110 0	14	0,0000 0000 0000 0001 0
7	0,0000 0000 1010 0011 0	15	0,0000 0000 0000 0000 1

3. Discriminator design example

Consider the example of the discriminator design, which estimates the phase by the CORDIC algorithm. The initial data is of 10-bit width, the output phase has the bit width 6.

Due to the fact, that the most significant bit of the result means the coding of wether 1,4 quadrants, or 2,3 quadrants, we must to implement 6-1 = 5 algorithm iterations. For these iterations up to 5 additions of the angles are made. Therefore, the angle coefficients must be of $6+]\log_2 5[= 9$ bit width.

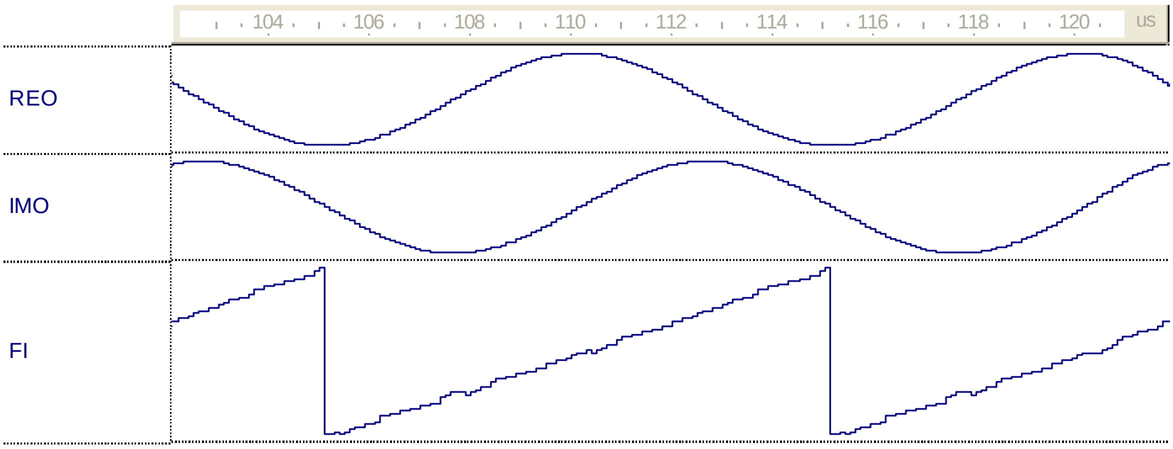
The discriminator is described by the following VHDL entity

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.STD_LOGIC_SIGNED.all;
entity CORDIC5 is port( CLK : in STD_LOGIC;
       RST : in STD_LOGIC;
       EI : in STD_LOGIC;                  --operation enable
       I : in STD_LOGIC_VECTOR(9 downto 0);
       Q : in STD_LOGIC_VECTOR(9 downto 0);
       FI : out STD_LOGIC_VECTOR(5 downto 0) );
end CORDIC5;
architecture BEH of  CORDIC5 is
 signal qi,qa,q1,q2,q3,q4,q5:STD_LOGIC_VECTOR(10 downto 0);
 signal ii,ia,i1,i2,i3,i4,i5:STD_LOGIC_VECTOR(10 downto 0);
 signal sq,si,s1,s2,s3,s4,s5:STD_LOGIC;-- sign of Q
 signal f1,f2,f3,f4,f5: STD_LOGIC_VECTOR(8 downto 0);
begin
 RGI:process(CLK,RST) begin    --   input data register
       if RST='1' then
             ii<=(others=>'0');
             qi<=(others=>'0');
       elsif CLK='1' and CLK'event then
             if EI='1' then
                   ii<=I(9)&I;
                   qi<=Q(9)&Q;
             end if;
       end if;
 end process;
 -- now we get the absolute value of I
 ia<=abs(ii);
 sq<=qi(9);     --data signs
 si<=ii(9);
 --the first iteration
 sq<=qi(9);
 q1<= qi - ia when sq='0' else
```

```
              qi + ia;
  i1<=  ia + qi when sq='0' else
              ia - qi;
  f1<= "001000000"when sq='0' else  --angle 45 grad.
        -"001000000";
  --the second iteration
  s1<=q1(9);
  q2<= q1 - SHR(i1,"001")  when s1='0' else
              q1 + SHR(i1,"001");
  i2<=i1 + SHR(q1,"001") when s1='0' else
              i1 -  SHR(q1,"001");
  f2<= f1 + "000100101" when s1='0' else    --angle 22,5 grad.
              f1 -   "000100101";
  --the 3-d iteration
  s2<=q2(9);
  q3<= q2 - SHR(i1,"010")  when s2='0' else
              q2 + SHR(i1,"010");
  i3<=i2 + SHR(q1,"010") when s2='0' else
              i2 -  SHR(q1,"010");
  f3<= f2 + "000010011" when s2='0' else      --angle 11 grad.
              f2  - "000010011";
  --the 4-th iteration
  s3<=q3(9);
  q4<= q3 - SHR(i1,"011")  when s3='0' else
              q3 + SHR(i1,"011");
  i4<=i3 + SHR(q1,"011") when s3='0' else
              i3 -  SHR(q1,"011");
  f4<= f3 + "000001010" when s3='0' else    --angle 5,5 grad.
              f3 - "000001010";
  --the 5-th iteration
  s4<=q4(9);
  q5<= q4 - SHR(i1,"100")  when s4='0' else
  q4 + SHR(i1,"100");
  i5<=i4 + SHR(q1,"100") when s4='0' else
              i4 -  SHR(q1,"100");
  f5<="010000000" when ia(10 downto 4)=0 and sq='0' else--angle 90 grad.
     "110000000" when ia(10 downto 4)=0 and sq='1' else--angle-90 grad.
     f4 + "000000100" when s4='0' else            --angle 2,7 grad.
     f4 - "000000100";

  process(CLK,RST) begin   --    --Result Register
        if RST='1' then
              FI<=(others=>'0');
        elsif CLK='1' and CLK'event then
              if si='0' then                    FI<=f5(8 downto 3);
              --   angles +-90 grad
              else
                    if sq='0' then
                         FI<="011111"-f5(8 downto 3)+1;-- angles+90+180 grad
                    else
                          FI<="100001"-f5(8 downto 3)-1; -- angle-90-180 grad
                    end if;
              end if;
        end if;
  end process;

end BEH;
```



4. **Laboratory exercise implementation**

The discriminator 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 discriminator parameters are input signal bit width *Ni* and output signal bit width *No* is selected from the Table 1.

Table 1

b_2, b_1, b_0	*Ni*	*No*
000	10	4
001	10	5
010	12	6
011	12	7
100	14	4
101	14	5
110	16	6
111	16	7

Discriminator type and testing type are given in the Table 2

Table 2

b_3, b_0	Discriminator type	Test signal generator
00	AM Ponselet	Generator from laboratory exercise 1 with the inputs clipped by a meander
01	FM	Testbench generator in laboratory exercise 3
10	PM CORDIC	Generator from laboratory exercise 1
11	AM CORDIC	Generator from laboratory exercise 1 with the inputs clipped by a meander

Testing

5. Laboratory exercise report

The laboratory exercise report must contain:

- Goal of the work,
- Discriminator 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.