17. Juli 2001

GSI

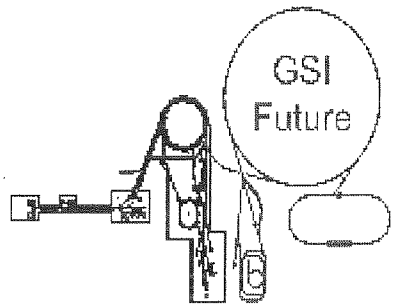
Report 2001 - 03
Juli

Study and realization of a digital *Notch* filter for the TFS in the SIS

A. D' Elia

Gesellschaft für Schwerionenforschung mbH
Planckstraße 1 • D-64291 Darmstadt • Germany
Postfach 11 05 52 • D-64220 Darmstadt • Germany

Gesellschaft für Schwerionenforschung (GSI)



Study and realization of a digital *Notch* filter for the TFS in the SIS.

By

Alessandro D' Elia[1].

Darmstadt 2001

[1] Visitor from Physic Department of " Federico II " University, Naples, Italy. E-mail address: delia@na.infn.it.

Index.

Introduction..... 1

1. The main board..... 2
2. Spectral analysis of the beam..... 3
3. Treatment of the signal..... 6
3.1. *BOSS*..... 7
3.2. The *notch* filter..... 7
4. Theoretical efficiency calculation for the *notch*..... 11
5. *BOSS* theoretical measurements..... 13
6. The real shape of the *notch*: comparison with the *BOSS*, why do we prefer the *notch*?..... 15
7. *NOTCH* filter practical realization..... 16
7.1. The PHILIPS..... 16
7.1.1. General overview..... 16
7.1.2. DEMUX_MEM..... 16
7.1.3. DEMUX_ADC..... 17
7.1.4. CLOCK_DIV..... 17
7.1.5. ADD_SUB1..... 17
7.1.6. ADD_SUB2..... 18
7.1.7. MUX..... 18
7.1.8. What's wrong?..... 18
7.2. The ALTERA..... 20
7.2.1.A general view of the project..... 20
7.2.2. ADC_MANAGER..... 21
7.2.3. MEM_MANAGER..... 21
7.2.4. A_ALU..... 21
7.2.5. DAC_MANAGER..... 23
7.2.6. BIG_MEMORY..... 24
7.2.7. COUNT..... 24
8. Conclusions and outlooks..... 25

Appendix 1: Mathcad calculations.
Appendix 2: Philips schematics and phdl program.
Appendix 3: Altera schematics.

Bibliography.

Introduction.

The German Research Facility for Heavy Ion Research (GSI) intends to increase the beam current in the synchrotron (SIS) by a factor of 100 in the medium and high ion mass range, at the same time this improvement can lead to transversal or longitudinal instabilities.

In order to damp the coherent transverse beam oscillations a Transversal Feedback System (TFS) has been designed. In particular a transversal feedback system exists for damping collective betatron oscillations at frequencies much lower than those of transverse stochastic cooling systems; it corrects the coherent motion of the particles (macroscopic) whereas the stochastic cooling systems correct the incoherent particle motion (microscopic) [1].

The TFS consists of two *pick-ups*, a *variable delay*, a *power amplifier*, a *kicker*, a *Beam Offset Orbit Suppressor (BOSS)* or a *notch filter.* For operating, the system needs three data ramps: the *revolution time, A* and *B vectors*. These column vectors come from the Twiss matrix elements [2] and they link position and momentum of the beam in the kicker to those relative to *pick-ups* 1 and 2 [3]. They are used also for connecting the two *pick-up* relative positions. During the acceleration, the beam parameters change and Twiss matrix too respect to the initial conditions. By selecting two position monitors, betatron phase relation between the *pick-ups* and the feedback *kicker* can be controlled by adjusting *A* and *B vectors*. They are precalculated, stored in tables and used for setting up some attenuators when the feedback system is running. The system will provide 16 different types of cycles and for each cycle several coefficients will be calculated.

The TFS layout consists of two boards: *control* and *main.* The *control board* has the task to process the data from control room (*revolution time, A* and *B vectors*) and for this purpose a fixed point DSP TMS320C6201 clocked at 200 MHz is used [4]. The task of the *main board* is to provide the variable delay required to adjust in time the relative position of *pick-up* 1 and 2 and between them and *kicker.*

This report is centered on the *main board* and on the *notch filter* there implemented.

In the table 1 below some important parameters of the SIS are shown.

Table1: SIS parameters.

Circumference of the SIS	216.7 m
Number of the bunches	1-4 (optional 6)
Bunching factor	0.3
Bunch length (FWHM)	30 ns- 420 ns
Revolution time	765 ns- 4.6 μs
Q-horizontal	Q_h =4.1- 4.4
Q-vertical	Q_v =3.1- 3.4
β= v/c	0.156- 0.95
Injection energy	11.5 Mev/u
Acceleration time	100 ms– 400 ms

1.1. The main board.

The *main board* task is to provide the variable digital delay. The digital system will be initialized only once from the *control board* by an address word that represents the delay at the injection, then the new delay value during acceleration process will be settled up by means only of one digital signal. In the table 2 below some parameters of the TFS are shown [3].

Table2:TFS parameters.

Min. delay (β=1)	630 ns
Max. delay (β=0.16)	4800 ns
Min. step	$\approx$ 10 ns
Frequency range of the analog signal	20 kHz – 40 MHz
Impedance	50 Ω
Dynamic range	> 60 dB

At the beginning there were two versions of this board: one for the Philips chip and another for the Altera chip. But later on only the Altera one was implemented; in the following the reasons of this choice will be explained. The board is composed by two ADC at 100 MSPS with 12-bit resolution, a DAC at 100 MSPS with 14-bit resolution, a *small memory*, a chip Altera or Philips and a *big memory*. The ADCs digitalize the input signals from the *pick-ups*, while the DAC gives the analogical output for the *kicker*. After the digitalization from the AIN1 (see Fig.1) the signal is stored in the *small memory* that contains the small delay between both *pick-ups* (50 ns – 1 μs) read out from the *control board*. Then the signals which come from that memory and from AIN2 are processed in the chip and brought to the *big memory*. This one represents the big delay between *pick-ups* and *kicker* read from the *control board* as for the previous memory. In the Altera version the *big memory* is implemented on the chip.

In order to give a correct kick to the beam it is important not only to realize a correct timing between *kicker* and *pick-ups* and among *pick-ups*, but it is also important to manipulate properly the signal from the monitors: the *closed orbit offset* part has to be eliminated. In fact the exciter efficiency in damping the instabilities decreases drastically when an unwanted signal, like the one coming from the *closed orbit offset*, is present in the beam. For that a *BOSS* or a *notch filter* has to be used in order to clean the signal before its processing and final correction. Our *notch filter* is implemented in the Altera or the Philips chip.

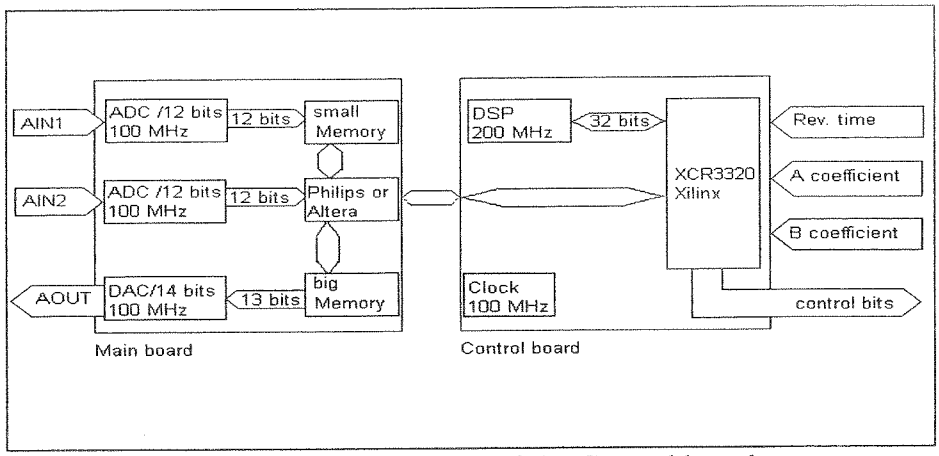


Fig.1: Scheme of the Main board and of the Control board.

2. Spectral analysis of the beam.

For a better understanding of the *notch filter* function it is important to discuss briefly about the nature of the signal coming from the *pick-ups*.
In circular machines like a synchrotron or an accumulator the particles obey to Hill's equation in the transverse plane (z stands for x or y transverse coordinates and s is the longitudinal one):

$$\frac{d^2 z(s)}{ds^2} + K_z(s) z(s) = 0 \qquad (1)$$

where the function K is assumed to be periodic over a period L_z:

$$K_z(s) = K_z(s + L_z) \qquad (2).$$

The solution of this equation gives the *betatron oscillation:*

$$z(s) = \sqrt{\varepsilon_z \beta_z(s)}\, cos[\, \mu_z(s) - \Phi_{0z}\,] \qquad (3);$$

where ε_z is a costant of the motion called *emittance* and Φ_{0z} depends on the initial condition; $\mu(s)$ is the phase advance of the betatron oscillation. The $\beta_z(s)$ function is the amplitude of the oscillation and it is a periodic solution of the equation:

$$\frac{\beta_z(s)\beta_z'(s)}{2} - \frac{(\beta'_z(s))^2}{4} + K_z(s)\beta_z^2(s) = 1 \qquad (4).$$

The betatron phase advance from the origin is given by:

$$\mu_z(s) = \int_0^s \frac{d\sigma}{\beta_z(s)} \qquad (5).$$

The betatron *tune* Q is defined as the number of betatron oscillations done by the beam in a turn of the ring. Then if Q_z indicates Q_x or Q_y:

$$Q_z = \frac{\mu_z(C_0)}{2\pi} = \frac{1}{2\pi}\int_s^{s+C_0} \frac{d\sigma}{\beta_z(\sigma)} \qquad (6),$$

where C_0 is the total length of the machine.
To avoid the excitation of the resonances due to unavoidable imperfections of the machine, Q has to be neither entire nor rational [2]. Typically it has to be avoided:

$$mQ_x + nQ_y = \text{integer} \qquad (7).$$

The equation (3) can be written now as:

$$z(s) = \sqrt{\varepsilon_z \beta_{n_z}}\, cos(\frac{Q_z s}{R} - \Phi_{0z}) \qquad (8),$$

where the longitudinal coordinate is expressed by $s = v\,t$ and the revolution pulsation by $\omega_{rev} = v/R$, it is possible to write:

$$z(t) = \sqrt{\varepsilon_z \beta_{n_z}}\, cos[\,Q_z \omega_{rev} t - \Phi_{0z}\,] \qquad (9).$$

By putting $A = \sqrt{\varepsilon_z \beta_z}$, and writing the betatron function in an exponential form where the particle motion corresponds to the real part of the function, we have:

$$z(t) = Ae^{j(Q_z \omega_{rev} t - \Phi_{0z})} \qquad (10).$$

In order to measure the transverse displacement of the beam we can use a *transverse pick-up*, which detects the beam dipole moment [5].

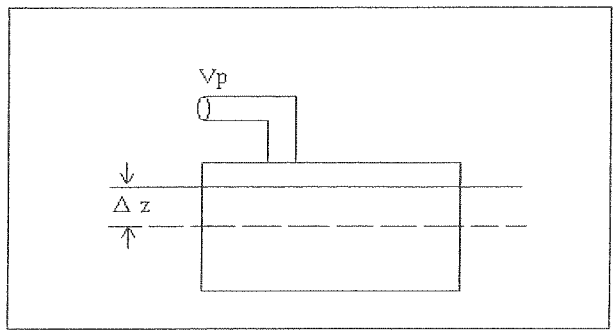


Fig.2: Schematic model of transversal *pick-up*: Δz is the displacement and *Vp* is the *pick -up* voltage.

The beam dipole moment will be:

$d(t) = I(t) \cdot \Delta z(t)$ (11), where $\Delta z(t) = z(t) - z_0$, and z_0 is the beam displacement.

Assuming *Ne* charged particles, with revolution frequency f_{rev}, and considering $\Phi_{0z} = 0$ we can expand the particle dipole as Fourier series (*Schottky* spectrum):

$$d(t) = ANef_{rev} \sum_{n'=-\infty}^{\infty} e^{jQ\omega_{rev}t} e^{jn'\omega_{rev}t} + z_0 Nef_{rev} \sum_{n'=-\infty}^{\infty} e^{jn'\omega_{rev}t} \qquad (12)$$

Where we dropped the pedix z from Q_z in (10).
In a more useful form:

$$d(t) = z_0 Nef_{rev} \sum_{n'=-\infty}^{\infty} e^{jn'\omega_{rev}t} + ANef_{rev} \sum_{n'=-\infty}^{\infty} e^{j(n'+Q)\omega_{rev}t} \qquad (13)$$

The betatron motion produces two lines for each frequency f_{rev} in the transverse spectrum plus additional lines of amplitude $2z_0 Nef_{rev}$ at each integer multiple of f_{rev}. Below in Fig.3 it is possible to see the betatron motion and his spectrum sampled by a *pick-up*; $x_{pu}(t)$ is our $d(t)$ and $x_{pu}(f)$ is our $d(t)$ in the frequency range.

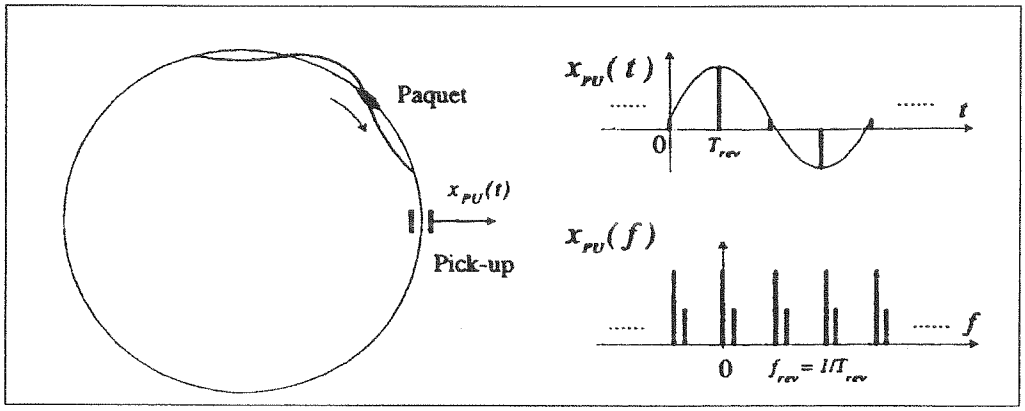


Fig.3: The betatron motion sampled by *pick-up* becomes a train of the pulses.

By considering $Q=Q^*+q$, where the Q^* is the integer part and q its fractional part:

$$d(t) = ANef_{rev} \sum_{n'=-\infty}^{\infty} e^{j(n'+Q^*+q)\omega_{rev}t} + z_0 Nef_{rev} \sum_{n'=-\infty}^{\infty} e^{jn'\omega_{rev}t} \qquad (14).$$

Hence, if we pose $n'=n'+Q^*$ because they are both integer:

$$d(t) = ANef_{rev} \sum_{n'=-\infty}^{\infty} e^{j(n'+q)\omega_{rev}t} + z_0 Nef_{rev} \sum_{n'=-\infty}^{\infty} e^{jn'\omega_{rev}t} \qquad (15).$$

And in a most understandable form:

$$d(t) = Nef_{rev}\left[2\sum_{n'=0}^{\infty} z_0 e^{jn'\omega_{rev}t} + A\left(\sum_{n'=1}^{\infty} e^{j(q\pm n')\omega_{rev}t} + e^{jq\omega_{rev}t}\right)\right] \qquad (16).$$

By fixing $A=1$ and for a particular n mode:

$$d_n(t) = 2Nef_{rev} z_0 e^{jn'\omega_{rev}t} + Nef_{rev} e^{j(n'+q)\omega_{rev}t} + Nef_{rev} e^{j(q-n')\omega_{rev}t} \qquad (17).$$

In Fig. 4 the frequency spectrum corresponding to (17) is shown [6].

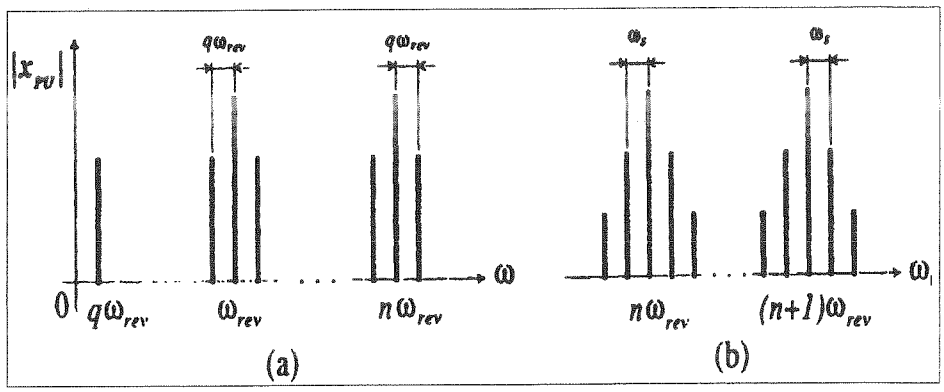


Fig.4: frequency spectrum of the betatron oscillation given by Fourier analysis of the *pick-up* signal: (a) only betatron lines; (b) synchrotron satellite lines in the spectrum in a bunched beam.

In the previous spectrum some other lines appear, called *synchrotron satellites*, due to the energy spread of the particles around the synchronous particle in a bunched beam.

3. Treatment of the signal.

As previously explained, the beam position signal coming from the *pick-ups* contains also the harmonics of the revolution frequency due to the *closed orbit offset*. This *offset* is in a first approximation constant during the machine turns. These harmonics are in general more than one thousand bigger than the others, and they do not give any useful information about the transversal motion. Because of the big difference in the levels of the two kinds of harmonics, very powerful amplifiers are required to read out the "smaller" amplitudes, leading to bad efficiency of the system. One way to resolve the problem is to eliminate the central lines at ω_{rev} and at $n\omega_{rev}$, leaving the others unchanged. This can be done by using a *BOSS* or a *notch filter*.

3.1. BOSS.

One solution for the suppression of *closed orbit offset* components is shown below in Fig.5, the *BOSS* principle scheme is presented [7].

The signals *e1, e2* coming from the *pick-ups* can be written as:

$$e1(t)=[1+h(t)+g(t)]f(t); \qquad e2(t)=[1-h(t)-g(t)]f(t) \qquad (18).$$

The function $h(t)$ represents our *closed orbit offset* signal as a periodic function not overlapping with $g(t)$ which represents our sidebands in the frequency spectrum. Both are modulated by the same amplitude function $f(t)$ with opposite phases. We suppose that the three signals are independent from each other. The analogical multiplier does the product between $(e1+e2)$ and $(e1-e2)$; one of the modulating envelopes is selected by the *low pass* and fed the *fets* for having a new balance condition of the bridge. Finally the new output signal, e_s, will be:

$$e_s(t)=e1(t)[1-h(t)]-e2(t)[1+h(t)]=2g(t)f(t) \qquad (19).$$

In the formula we can see the rejection of the *closed orbit* signal, while the sidebands remain untouched. This suppression mechanism takes more then 100 ms (many machine turns) to be effective.

The output signal can be then used for the TFS system.

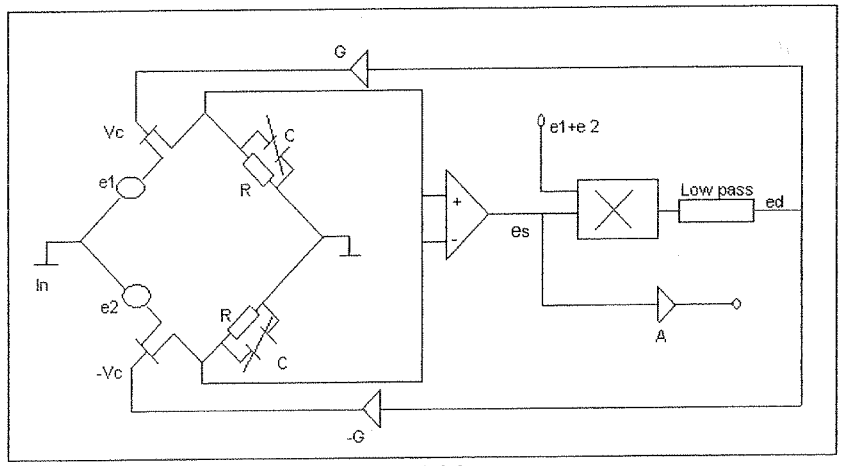


Fig.5: principle scheme of the *BOSS*.

3.2. The notch filter.

The *notch* principle is the following: the signal from the first *pick-up* is delayed for a time τ_d and, bypassing the electronic delay, it is added to the direct one from the second *pick-up*. The delayed signal combines with the direct one at the frequency equal to $1/\tau_d$ cutting this frequency at all harmonics. If $1/\tau_d$ is a multiple of the revolution frequency, the filter will cut

the longitudinal signal at all harmonics of this frequency. It is obvious that we need to know as correctly as possible the revolution frequency [8].

It is possible to describe the *notch* as a recursive filter (IIR) [9].

A continuos signal $x(t)$ can be sampled taking in to account the *Shannon theorem*: " *A signal which is band limited up to a maximum frequency w is entirely determined by its values at regular intervals spaced by* $T_S = 1/f_S = 1/(2w)$ ". Hence we can substitute the continuos signal $x(t)$ by a series of values $x(n)$ taken every T_S interval. It is possible to write $x(n)$ by a linear combination of $\delta(n)$ pulses:

$$x(n) = \sum_{k=-\infty}^{\infty} x(k)\delta(n-k) \qquad (20),$$

where $x(k)$ are coefficients and the function $\delta(n-k)$ is defined by:

$$\delta(n-k) = 1 \quad \text{for} \quad n-k = 0,$$
$$\delta(n-k) = 0 \quad \text{for} \quad n-k \neq 1.$$

For a *linear time invariant system* the output sequence $y[n]$ will be:

$$y[n] = \sum_{k=-\infty}^{\infty} x(k)h(n-k) \qquad (21),$$

where $h(n)$ is the response to unit sample sequence.

For a large class of *linear time invariant systems*, we can write a linear constant coefficient difference equation:

$$y[n] = \sum_{i=0}^{N} b_i x[n-i] + \sum_{i=1}^{M} a_i y[n-i] \qquad (22).$$

This means that the output at the sample n is determined by the $(n-i)$ inputs sampled for $i=0..N$ and by the $(n-i)$ outputs sampled for $i=1..M$. More clearly, we could say that the output at " time " n depends on the inputs and outputs at " time " $(n-i)$.

By using the z transform it can be written as:

$$y[n] = x[n]\sum_{i=0}^{N} b_i z^{-i} + y[n]\sum_{i=1}^{M} a_i z^{-i} \qquad (23).$$

And the *transfer function* will be:

$$H(z)=\frac{Y(z)}{X(z)}=\frac{\sum_{i=0}^{N} b_i z^{-i}}{1-\sum_{i=1}^{M} a_i z^{-i}} \qquad (24).$$

If M=0 we have the *finite impulse response* system (FIR), in the other case we have a *infinite impulse response* system (IIR); this last one is our case [10].

In general, sampled filters, which operate on the values taken at discrete time intervals, are not necessarily digital; *x(n)* could be for instance the charges stored in *CCD (charged coupled device)* processors. In our case we speak about a digital filter.

In Fig.1 the *main* and *control board* are shown. The *notch filter* is fitted in the *main board* in the Altera chip. Two 12 bits ADCs provide the digitalization of the signal from *pick-ups*. The filter has two inputs: one directly from an ADC and a second one from a *small memory,* which comes from another ADC and which gives the beam delay between both *pick-ups*. The output goes to the *big memory* which makes the delay between *pick-ups* to *kicker*; after this memory there is a 14 bits DAC for creating the analog signal (for more technical information see the paragraph " **7. *Notch* filter practical realization** ").

The *notch filter* can be seen as a *comb* filter combined with a subtractor. A *comb* filter connects the *N path technique* with a *first order low pass* IIR filter. In the z transform, z^{-1} can be read as the unit delay; for a *first order low pass* IIR filter we can write the *transfer function*:

$$H(z)=\frac{1}{1-az^{-1}} \qquad (25).$$

The delay is also connected to the number of bunches which run in the ring. If the number of bunches is more than one we can substitute to z^{-1}, z^{-h} where h is the number of the bunches. When we substitute the unit delay z^{-1}, with h delays, z^{-h}, we built a *comb* filter.

The frequency response can be known by substituting to z, $e^{j\Theta}$,that means:

$$H(e^{j\Theta})=\frac{1}{1-ae^{j\Theta}} \qquad (26).$$

And the module and argument of the function will be:

$$\left|H(e^{j\Theta})\right|=\frac{1}{\sqrt{1+a^2-2a\cos\Theta}} \qquad (27); \qquad arg\left[H(e^{j\Theta})\right]=arctg\left[-\frac{a\sin\Theta}{1-a\cos\Theta}\right] \qquad (28).$$

In Fig. 6 it is possible to see for h=20 the shape of the function module and also of the argument, moreover it is possible to see the representation of this function in the unit circle.

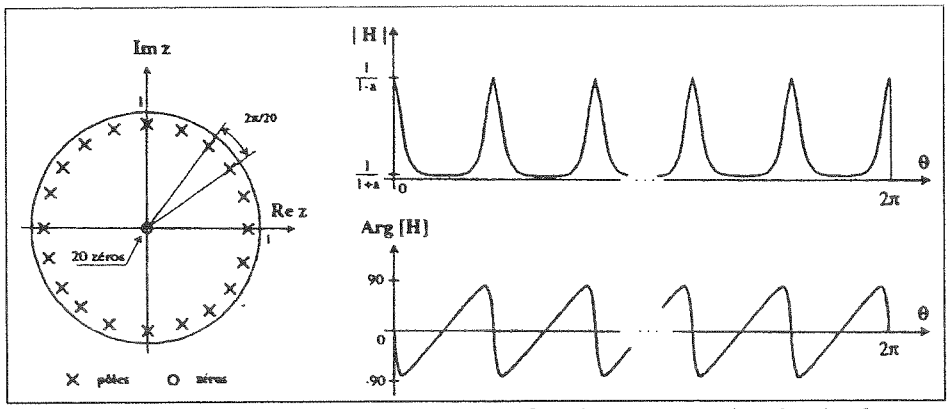


Fig.6: In the complex plane the position for the zeros and poles is shown in the case a=0.9. The graphics show the module and argument of the function with $\theta = \omega t$.

The *transfer function* of our *notch* filter will be:

$$G(z) = 1 - K_H H(z) \qquad (29).$$

The coefficient $K_H = 1/c$ is a normalization coefficient, where c is expressed as a power of 2. The condition $|K_H H(z)| = 1$ gives the relation between a and c coefficients:

$$\frac{1}{c} = 1 - a \qquad (30).$$

So it is possible to write:

$$G(z) = 1 - \frac{1}{c - bz^{-1}} \qquad (31) \quad \text{with } b = c - 1 \qquad (32).$$

The algorithm to logically implement the filter is:

$$y^* = \frac{1}{c}(x + by^*_{-1}) \qquad (33).$$

Anyway the $G(z)$ becomes:

$$G(z) = a\frac{1 - z^{-1}}{1 - az^{-1}} \qquad (34).$$

And in the frequency domain it is:

$$G(e^{-j\Theta}) = a\frac{1 - e^{-j\Theta}}{1 - ae^{-j\Theta}} \qquad (35).$$

Below the response of a twenty bunch signal is shown (Fig.7, 8) [6].

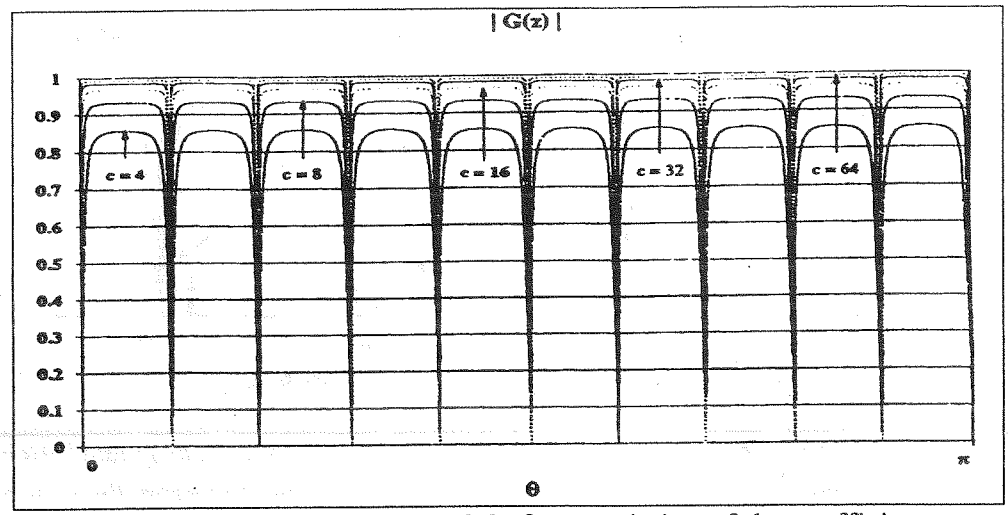


Fig.7: variation of the function module for a variation of the coefficient *c*.

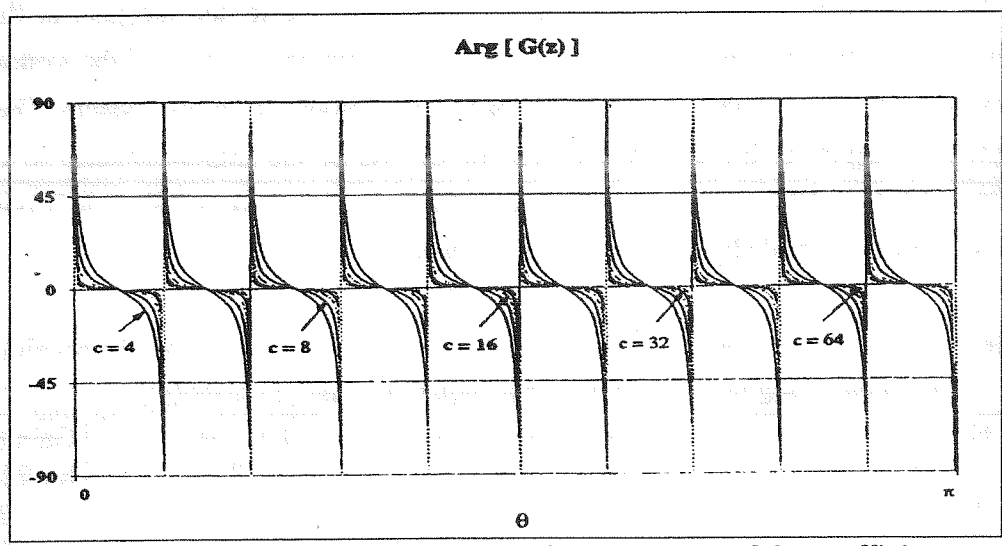


Fig.8: variation of the function argument for a variation of the coefficient *c*.

4. Theoretical efficiency calculation for the *notch*.

The efficiency is defined as the ratio between damping of the longitudinal signal compared to the damping of the sidebands in the spectrum. In Fig.3 the behavior of the spectrum from the *pick-ups* has been shown, when a *notch* is applied, there will be a shape like the following (Fig.9):

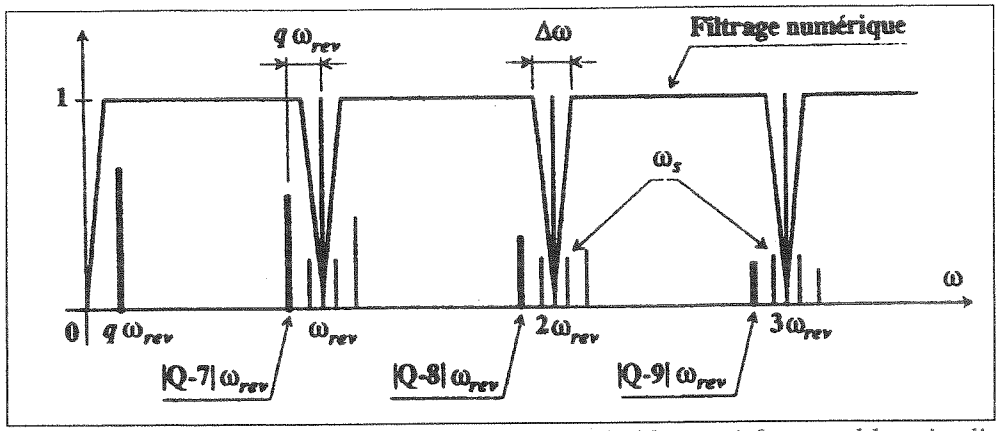


Fig.9: Spectrum of a not centered signal sampled by a *pick-up* and longitudinal signal rejection at different harmonics; ω_s represent the synchrotron satellite lines.

For the theoretical *notch filter* in the table 3 it is possible to see the gain at the first three harmonics of the revolution frequency for different numbers of bits of the system. The gain for the sidebands is 0, that means the sidebands are untouched by the system. The gain of the longitudinal signal is limited by the number of bits of the digital system: in our case the number of bits is 12 and we have a gain equal to -78 dB. In **Appendix 1** it is possible to see these calculations made by using the code *mathcad*.

Table3: Gain of a *notch* at injection level for the first two harmonics of the revolution frequency and for 4,12,24-bits. The sidebands are untouched.

Harmonic number	Frequency (kHz)	Gain (dB) For 4-bits	Gain (dB) for 12-bits	Gain (dB) for 24-bits
Rev. Freq.	$\nu_{rev} = 218.6$	-29.8	-78.3	-150.5
h=1	$\nu_{rev1} = 437.2$	-29.8	-78.3	-150.5
h=2	$\nu_{rev2} = 655.8$	-29.8	-78.3	-150.5

In a first approximation it is possible to write the formula:

$$\eta_v = \frac{2u_1^2}{2u_1^2 + u_2^2} \qquad (36)$$

where u_1 is the amplitude of the sideband and u_2 is the amplitude of longitudinal signal in the spectrum.
Putting $v = u_2 / u_1$, it is possible to write η_v in this form:

$$\eta_v = \frac{2}{2 + v^2} \qquad (37).$$

If the amplitude of the longitudinal signal and of the relative sidebands is known, one can estimate the efficiency of the system. The amplitude of this signal depends on the charge state

and on the number of ions in the accelerated bunches. If those parameters are fixed the amplitude is constant. In Fig.10 there is an example of this kind of spectrum, which is obtained by sending in a spectrum analyzer a Δ signal like the signal from the *pick-ups*. In Fig.11 the graphic representation of the formula (38) is shown with v expressed in decibel. In practice the damping of those signals is important in order to have a smaller amplification, that means less noise problems and a better control of the system. For instance, from the spectrum below we can read a gain for the longitudinal line G=5 dBm, and G=-45 dBm for the sidebands of the first signals, this means u_1=0.040 V and u_2=12.6 V. The efficiency will be η_v= 0.0002. If the *notch* damping equal to –78 dB is applied the new value of the efficiency will be u_2= 0.0016 V, u_1= 0.040 V and η_v=0.9992.

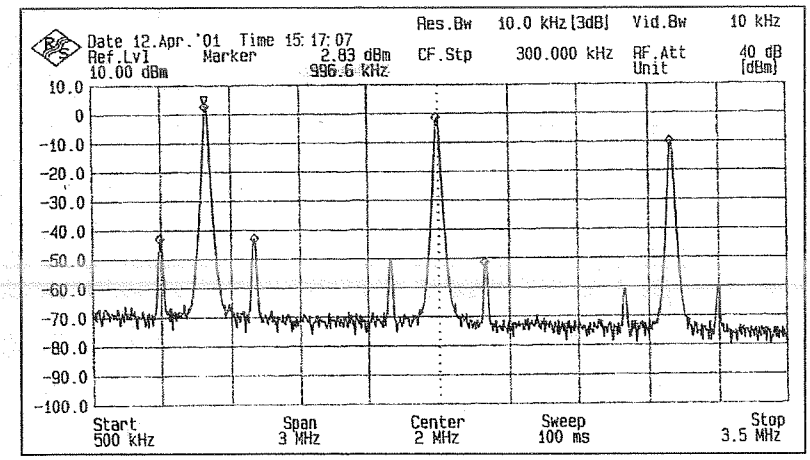


Fig. 10: Schottky spectrum obtained by putting a Δ signal in a spectrum analyzer.

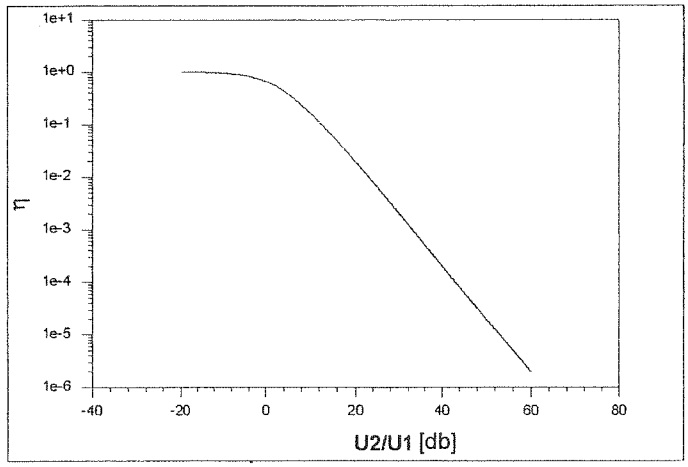


Fig.11: efficiency η_v depending on the fraction u_2/u_1 in dB.

5. *BOSS* theoretical measurements.

A brief description of the *BOSS* and a principle scheme have been shown above. The following measurements are obtained by putting at the input a Δ signal in order to simulate

those coming from the *pick-up*; the longitudinal signal and its sidebands have been measured with the *BOSS* feedback switched off and on. In the table 4 are shown the results of the measurements, the sideband values are not shown in the table because, as for the *notch*, there is no appreciable variation of the signal level.

Table 4: *BOSS* measurements: u is the revolution frequency; h_1, h_2, h_3 are the harmonics 1th, 2th and 3th of the revolution frequency; Δu is the difference between the two signals coming from the *pick- up's* plates; G is the gain of the system.

Δ signal (V)	Without feedback (dB)				Suppression (dB)	With feedback (dB)			
Δu	u	h_1	h_2	h_3	G	U	h_1	h_2	h_3
0.02	4.50	-1.53	-18.80	-15.30	4.50	0.00	-5.90	-23.10	-19.70
0.02	-0.41	-6.28	-22.90	-20.10	15.79	-16.20	-21.90	-38.50	-36.40
0.01	-4.80	-10.60	-26.80	-24.90	43.70	-48.50	-50.00	-60.00	-57.00
0.01	-9.72	-15.90	-31.90	-29.90	39.18	-48.90	-48.50	-60.50	-56.00
3.824×10^{-3}	-15.20	-21.60	-37.20	-35.80	36.00	-51.20	-49.90	-62.00	-57.30
2.104×10^{-3}	-20.30	-25.90	-42.00	-40.70	31.70	-52.00	-50.60	-62.70	-57.30
1.204×10^{-3}	-25.00	-30.90	-47.00	-46.50	27.50	-52.50	-51.00	-63.00	-58.00
647.6×10^{-6}	-30.00	-36.00	-51.80	-52.20	21.50	-51.50	-50.10	-62.00	-57.20
360.2×10^{-6}	-34.60	-40.00	-55.80	-56.50	17.40	-52.00	-50.60	-62.50	-57.70
187.7×10^{-6}	-39.20	-44.00	-58.80	-58.90	12.60	-51.80	-50.40	-62.30	-57.50
98.50×10^{-6}	-45.00	-49.00	-58.60	-64.40	13.30	-58.30	-56.30	-63.00	-68.50

There is an appreciable variation of the longitudinal signal when the feedback system is on. In Fig. 12 it is shown a graphic for the evaluation of the best suppression in the system depending on the value of the Δ signal for the revolution frequency and 1th, 2th and 3th harmonics.

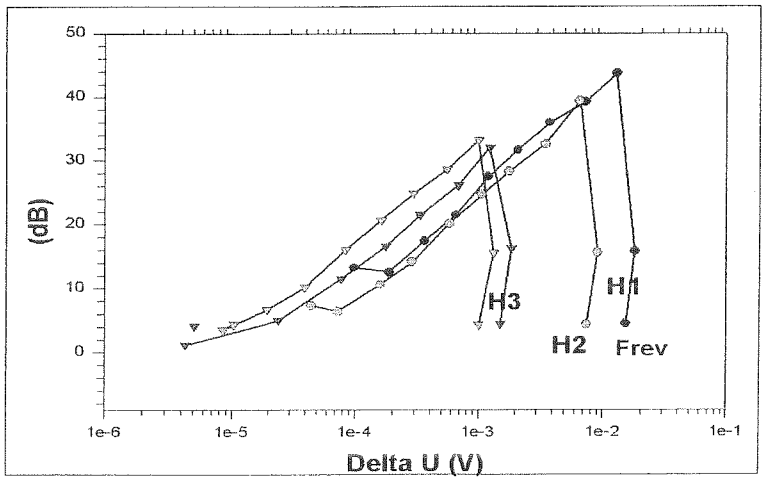


Fig.12: the behavior of the damping in dB for the revolution frequency and for the first three harmonics (H1, H2, H3) as function of the Δ signal in Volt.

6. The real shape of the *notch*: comparison with the *BOSS*, why do we prefer the *notch*?

The calculations made for our *notch* filter have shown a constant damping value very high (78 dB for our 12 bits!) for every frequency. In reality the *notch* shape is different from the one used for the calculation. In the Fig.13 it is possible to see some measurements taken by a analog *notch* filter with a cut frequency at 500 KHz.

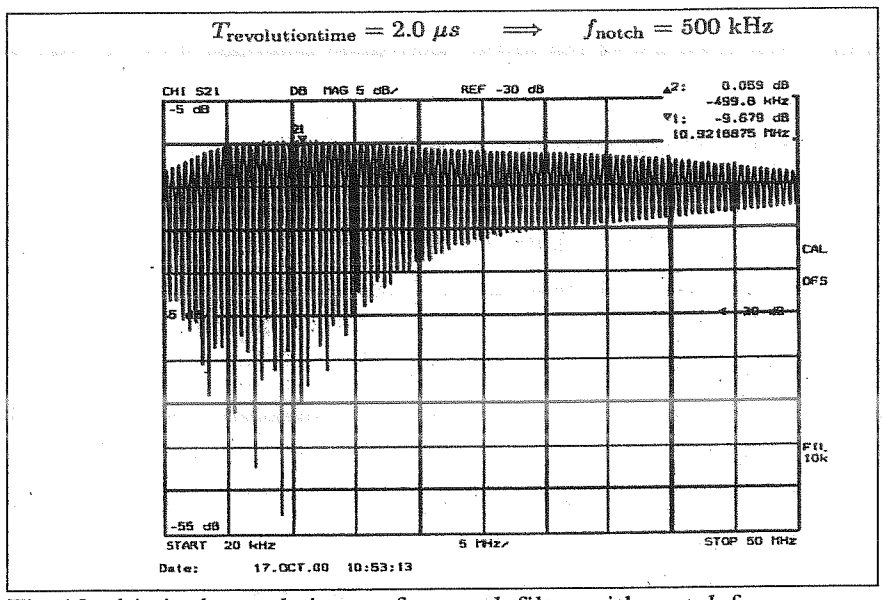


Fig.13: this is the real shape of a *notch* filter with *notch* frequency equal to 500 KHz (2.0 μs revolution frequency) measured by a spectrum analyzer. Note the damping for the high frequency.

The behaviour of this filter is to have less damping value for the high frequency. On one hand, this is a normal consequence of the parasitic impedance which dominates when increasing the frequency; on the other hand, the output signal is the convolution of the *notch* and input signal functions, which, in this measurement, is simulated and it is a pulse train. Anyway the gain of the system is estimated to be up to 40 dB for a long range of the values, and, however, one must not forget that this filter is a first version; we can use more bits and the theoretical calculation (table 3) shows a big gain increase for a growing number of the bits. The *BOSS* seems to have the same behavior in efficiency, but the system is analog and slower than the *notch*. An analogical system has much less possibility of control, while a digital system is always much more flexible and introduces less noise. The fundamental reason to prefer the *notch* to the *BOSS* is the possibility to have a correction in each turn of the beam. The *BOSS* can correct the system after at least 100 ms because it has to drive the amplifiers for giving the new voltage values to fets. The *notch* needs only to operate the calculation for the correction; its work frequency is for instance for the ALTERA of 130 MHz and it works in real time. Really the first good value for the correction comes out from the circuit after roughly 150 ns after the initialization; in any case the first correction is required not before

765 ns (smaller *revolution time*) in the worst case. This work has been written before implementing the board with the ALTERA component installed, and there are no measurements for our system to be compared with the *BOSS* ones, but the behaviour of the *notch* is very interesting and in the TFS project it is very important to correct as fast as possible the beam during the acceleration process.

7. *NOTCH* FILTER PRACTICAL REALIZATION.

In the beginning two versions of the *notch* filter were supposed to exist: one for the PHILIPS and another one for the ALTERA. But for the reasons explained below it has been impossible to fit the digital *notch* in the PHILIPS chip. The filter version that has been realized is a simple version. In fact the algorithm implemented is not the same described above, equation (25), but the *notch* makes just the sum of the signal from the first *pick-up* delayed of a time τ_d plus the direct one from the second *pick-up* (see paragraph **3.2**).

7.1. THE PHILIPS.

7.1.1. General overview.

The Philips has been designed by means of the CAD XPLA331. It is possible to do the project by using a text editor in PHDL language or a graphic interface. In this work it has been used the text editor in PHDL. One can see in the **Appendix 2** the program in the PHDL language and for a better understanding some graphic primitives of the circuit are also included. It is composed by five parties:

- DEMUX_MEM: a real time demultiplexer from the SMALL_MEMORY;
- DEMUX_ADC: a real time demultiplexer from the ADC;
- CLOCK_DIV: a clock divider x 2;
- ADD_SUB1: it is an adder-sub tractor device;
- ADD_SUB2: it is an other adder-sub tractor device;
- MUX: it is a multiplexer for the output data.

7.1.2. DEMUX_MEM.

This device provides the splitting of the data from the SMALL_MEMORY in two ways. The outputs go respectively in ADD_SUB1 or ADD_SUB2. This solution was necessary because the problem of the PHILIPS chip was the difference between the internal clock of the chip and the frequency range of the ADC data. Indeed whereas the internal clock was at 80 MHz, the data from ADC arrived every 10 ns that means 100 MHz frequency. By means of this solution we could split the summation-subtraction in two and operate at a frequency lower

than the data from the external device. The SELECT pin chooses the way and it comes from a clock divider for working in real time.

7.1.3. DEMUX_ADC.

It is the same function of the DEMUX_MEM, but for the data from the ADC which arrives every 10 ns. It is driven by the SELECT pin and it works in real time.

7.1.4. CLOCK_DIV.

It is a simple clock divider for generating the select signal for driving both the demultiplexers. There is a block which provides to give a costant value equal 1. By this divider it is possible to drive the data in real time: when the circuit starts every clock time a data will be driven or in ADD_SUB1 or in ADD_SUB2.

7.1.5. ADD_SUB1.

This is an adder-subtractor. Since there are these time constrains, a very fast adder was required. The choice has been to implement a CARRY-LOOK-AHEAD adder [11], [12]. The equations for this kind of the device are the followings:

$$C_1 = G_1 + P_1 C_{in}$$

$$C_2 = G_2 + P_2 C_1 = G_2 + P_2(G_1 + P_1 C_{in}) = G_2 + P_2 G_1 + P_2 P_1 C_{in}$$

$$C_3 = G_3 + P_3 C_2 = \ldots = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 C_{in}$$

.
.
.

$$C_n = G_n + P_n C_{n-1} = \ldots = G_n + P_n G_{n-1} + P_n P_{n-1} G_{n-2} + \ldots + P_n P_{n-1} \ldots P_2 G_1 + P_n P_{n-1} \ldots P_2 P_1 C_{in}$$

Where the terms $G_n = A_n B_n$ are called " *carry generate* ", while the terms kind of $P_n = A_n + B_n$ are called " *carry propagate* ". The Fig. 14 and 15 show graphically how you can implement this device on the chip, and the circuit for " *carry generate* ", and " *carry propagate* ".

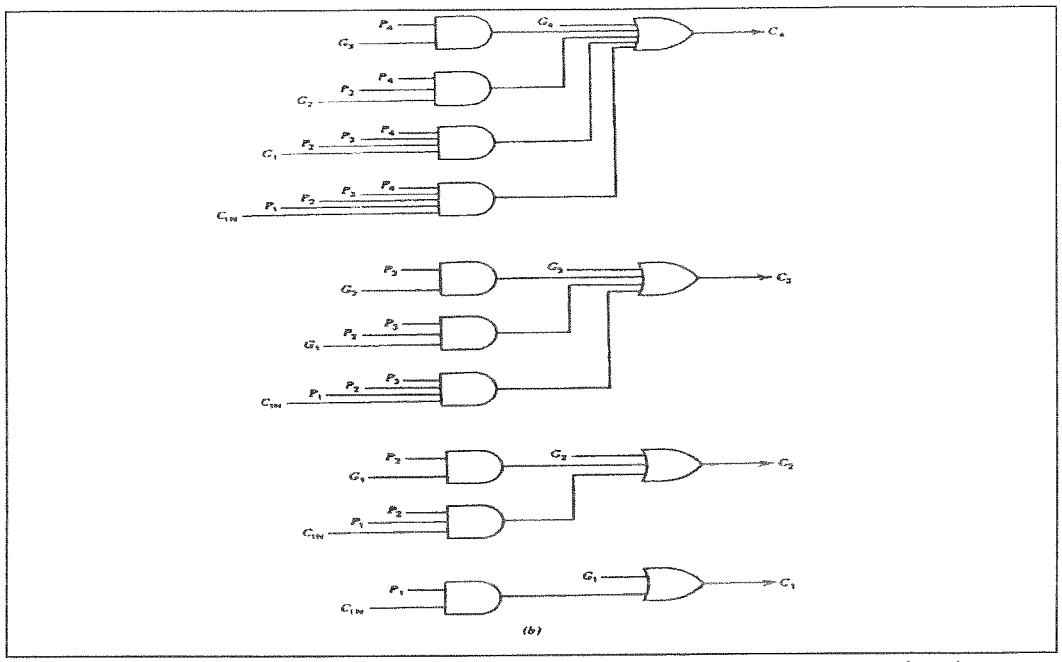


Fig.14: gates for connecting the *carry propagate* and *carry generate* circuits.

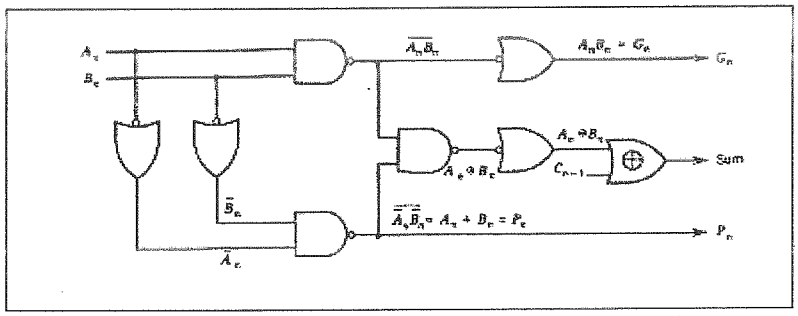


Fig.15: *carry propagate* and *carry generate* example.

The subtraction is made in the same way of the ALTERA chip but the used adder is that one described above, for more information about this operation you can see in the ALTERA chapter paragraph A_ALU.

7.1.6. ADD_SUB2.

This is the same device of ADD_SUB1.

7.1.7. MUX.

This device provides to give the data to BIG_MEMORY every 20 ns. Note the SELECT pin is the same but inverted from the CLOCK_DIV, this means that every 1 clock cycle the data are sent to the memory, but the operations can be made in 2 clock cycles.

7.1.8. What's wrong?

Between the PHILIPS device the component PZ3320 has been used. This choice was made because the other devices are slower even if they have a larger scale of the integration. One needs to satisfy two requests in this chip: a first one regarding the signal processing speed,

which comes every 10 ns from the ADC, and a second relative to the fitting of this device on the chip. For the speed it was implemented an algorithm like the upper one, but a CARRY-LOOK-AHAED adder requires more logic, and then it was impossible to fit it on the chip. We tried to make a 3-level CARRY-LOOK-AHAED, that means 3 different levels for having a good speed and optimizing the resources, but it was not enough for fitting. When we tried to use another device, for instance PZ5032, the circuit was fitted but the speed was twice slower than required time constraints.

However this was not a problem; indeed this device was thought for making a first study about the NOTCH response and it has been useful to take some confidence about the project and to develop some ideas for the ALTERA chip. In Fig. 16 below you can see an example for a 4-level CARRY-LOOK-AHAED adder and in Fig. 17 the board which had to support the PZ3320.

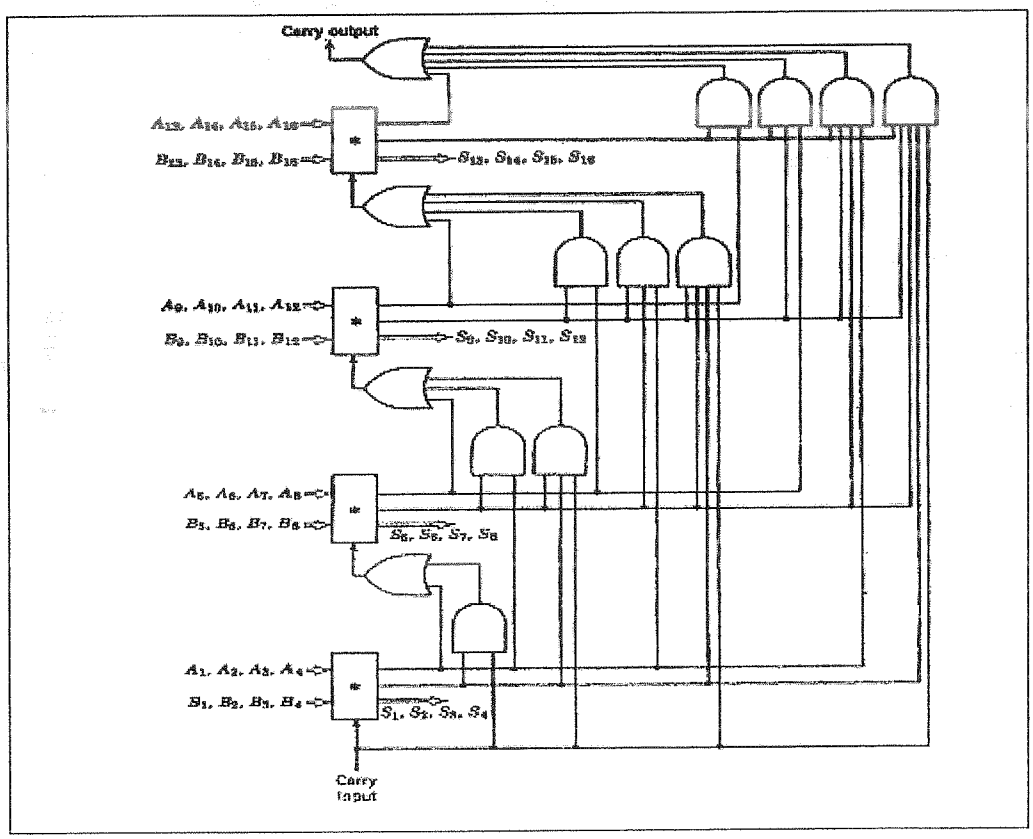


Fig.16: 4-level CARRY-LOOK-AHAED.

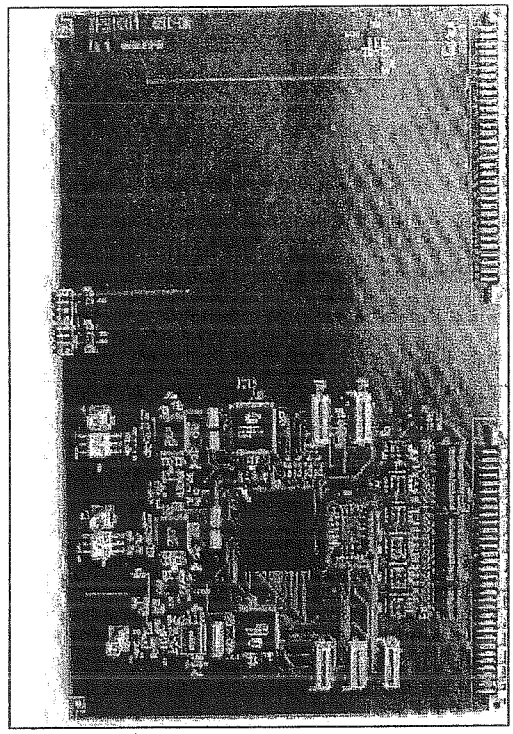


Fig.17: PHILIPS main board.

7.2. THE ALTERA.

7.2.1. A general view of the project.

The ALTERA chip has been designed by the "MAX + PLUS II" CAD. This one can be programmed in two different ways: it is possible to create a text file written in AHDL language, or to create a graphic design file. In this work a graphic interface of the CAD was used for drawing the project. In **Appendix 3** you can see in the NOTCH the top level of the hierarchy of the project. We have seven symbols:

- ADC_MANAGER: it is the circuit for managing the data from the ADC;
- MEM_MANAGER: it is the circuit for managing the data from the memory for small delay;
- A_ALU: it is an ALU (Arithmetic Logic Unit);
- BIG_MEM: it is a dual port memory;
- COUNT_A: it is a counter;
- COUNT_B: it is a counter;
- DAC_MANAGER: it is the circuit for managing the data to the DAC.

7.2.2. ADC_MANAGER.

The data from the ADC are at 12 bits. Since a sign summation must be done, the data from the ADC has to be changed. The first bit (the sign bit) 1 was chosen for the positive, 0 for the negative: on the contrary of the ADC data [13]. Modification of the negative data is also important: we want, for example, the "– 1" which correspond to 0000 0000 0001, and so on. In the table 5 it is possible to see the codes and the digital outputs:

Table 5: ADC output code.

CODE	DIGITAL OUTPUT
+ 2047	0111 1111 1111
•	•
•	•
0	0000 0000 0000
- 1	1111 1111 1111
•	•
•	•
- 2048	1000 0000 0000

The algorithm used for this realization has been the following: if the sign bit from the ADC is 1 (that means negative number) the data are added to 2047 (in decimal) and the sign bit is inverted; if the same sign bit is 0, this one is always inverted, but the data exit without changing. In the ADC_MANAGER design is shown the circuit for that. The FF_2, FF_DATA, FF blocks are *flip-flops* for synchronizing the data.

7.2.3. MEM_MANAGER.

The MEM_MANAGER is the same circuit of ADC_MANAGER. Indeed the data, stored in this memory, come from another ADC, that means we have to manage it in the same way of the previous data.

7.2.4. A_ALU.

This provides to do the summation between the 12 bits from the ADC and 12 bits from the SMALL MEMORY after the management by ADC_MANAGER and MEM_MANAGER. This one makes a sign summation. It is possible to see the circuit in the **Appendix 3**. There is a first block for managing the possibility to do the summation or subtraction by one *comparator*, two *multiplexer* and one *xor* gate. The sign bits enter in the A_ALU and they are driven in the *xor*; this one drives the ALDO input for selecting between summation or subtraction. Below the *xor* truth table is shown in the table 6.

Table 6: XOR for selecting the summation or subtraction.

In_mem 11	In_adc 11	Xor	Operation
1	1	0	Summation
0	0	0	
0	1	1	Subtraction
1	0	1	

For doing a summation, the following algorithm has to be implemented:

$$Res[11..0] = In_adc[10..0] + In_mem[10..0]$$

For a subtraction, the implemented algorithm is:

$$Res[11..0] = \overline{In_adc[10..0] - In_mem[10..0]}$$

if

$$In_adc \geq In_mem\ ;$$

or

$$Res[11..0] = \overline{In_mem[10..0] - In_adc[10..0]}$$

if

$$In_mem \geq In_adc$$

The AGB signal from the *comparator* is used for selecting the data from the *multiplexer*, driven to ALDO. A second block is characterized by the logic for putting out the *carry* (RES11) of the summation and the sign of the operation (RES12). Below (tables 7,8) there is the truth table and relative equations.

Table 8: logic for the carry Res11.

In_adc11	In_mem11	cout	Res11
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Table 7: logic for the sign of the operation Res12.

AGB	EAB	In_adc11	In_mem11	Res12
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

$$Res12 = \overline{AGB} \cdot \overline{AEB}(\overline{In_adc11} \cdot In_mem11 + In_adc11 \cdot In_mem11) +$$
$$+ \overline{AEB} \cdot In_adc11 \cdot AGB + AEB \cdot In_adc11 \cdot In_mem11$$

$$Res11 = Cout \cdot (\overline{In_adc11} \cdot \overline{In_mem11} + In_adc11 \cdot In_mem11).$$

7.2.5. DAC_MANAGER.

The DAC is a 14 bits device, with *binary code offset* [14]. In the table 9 it is possible to see the output behaviors:

Table 9: DAC output code.

CODE	OUTPUT
11 1111 1111 1111	16383
•	•
•	•
10 0000 0000 0000	0
•	•
•	•
00 0000 0000 0000	-16383

The data arrive to DAC not in this code, therefore a change has to be done. In **Appendix 3** you can see the circuit for making this one: there is a simple *multiplexer* for giving out the data from the BIG_MEMORY or the complement. For choosing the data or complement which have to pass, the pin SELECT is managed by the sign RES12. Obviously only 13 bits goes to the ouput from the BIG_MEMORY, this means that only 13 bits of the DAC have to be used.

7.2.6. BIG_MEMORY.

It is a " *dual-port* " memory used for generating the " *BIG_DELAY* " between the pickup electrode and the feedback kick. Since this delay varies between 10 ns ÷ 1 μs, roughly 1000 cells are required for simulating this delay. Indeed the clock of the board is 100 MHz (=10 ns), that means 10^3 ns= 1 μs. For this reason the memory is provided of 10 " write " and it reads addresses managed by the counters. There are also two different clocks, one for reading and the other for writing. The write and read operations of the BIG_MEMORY are driven by the DSP from CONTROL BOARD.

7.2.7. COUNT.

The counters manage the BIG_MEMORY and they are driven by DSP. They provide the 10 bits for giving the correct position to the data from A_ALU in the BIG_MEMORY. In this way one can simulate the *BIG_DELAY* between pickup electrode and feedback kick. A SYNC_LOAD input is used for charging the data in the counter by DATA_A and DATA_B input, so the difference between those data gives the *BIG_DELAY* value. Indeed one counter is used for selecting the write address (WRADDRESS) in the BIG_MEMORY, the other is used for reading address (RDADDRESS), DATA_A for writing, DATA_B for reading.

8. Conclusions and outlooks.

The *notch* filter can damp the longitudinal signal of the *Schottky* spectrum due to the *closed orbit offset* and coming from a bad centering of the beam detected by *pick-up* plates. This operation is fundamental in the TFS system for having less power requirements for the amplifiers. In fact the longitudinal spectrum lines are roughly one thousand times bigger than the sideband lines. These sideband lines have to be processed by the system in order to have correct information about the transversal motion of the beam and consequently, for giving the correct kick to the beam by the *kicker*. To reduce the power requirements we need to have a more clearer and then much more exact signal to kick the beam.

The same purpose can be served by a *BOSS*, which drives the amplifiers to change the fet voltage in a bridge; the equilibrium condition of this bridge changes and under some condition for the signal functions, we obtain a deletion of the longitudinal components in the spectrum. The *BOSS* can correct the signal only after 100 ms (many machine turns). Instead of the *notch* that can operate every machine turn.

This one is the first reason to prefer the *notch* to the *BOSS*: we can control and correct the beam every turn.

Another reason is that the *BOSS* is an analogical system, whereas the *notch* is digital: a digital system is always much more flexible than an analogical one.

This work was made before the implementation of the ALTERA Main Board and so for it we can not speak about real measurements to be compared with the *BOSS*. In any case the theoretical behaviours of our filter are very good and in many case they are much better than the *BOSS*. In this work we have not spoken about the phase error in the system. It is very important to know the effects in the TFS system driven by possible phase errors in our filter or equivalently in the *BOSS*. In fact the presence of a phase errors has a straight forward consequence in the TFS: we can see a mismatch between the calculated positions *pick-up 1-pick-up 2, pick-ups-kicker* and their "real" position. This means that we do not kick the beam exactly at point where "we should do it". Really the problem is bigger, because a mismatch, between the "real" and "calculated" positions, could cause an overlapping of the signals in the cable and so totally changing the shape of the correction signal.

Anyway these phase problems are subject to further investigation.

APPENDIX 1.

Mathcad calculations.

NOTCH EFFICIENCY

Calculus made at injection level:

$E_{kin} := 11.5 \quad MeV \qquad \gamma := 1.013 \qquad \beta := 0.158 \qquad Q_x := 4.290 \qquad Q_y := 3.290 \qquad q_{xy} := 0.290$

$\nu_{rev} := 218.6 \cdot KHz$

Bits number $n := 4$

Revolution frequency

$c := 2^n \qquad a := \frac{(c-1)}{c} \qquad \nu_1 := \nu_{rev} \rightarrow 218.6 \cdot KHz$

sidebands calculation

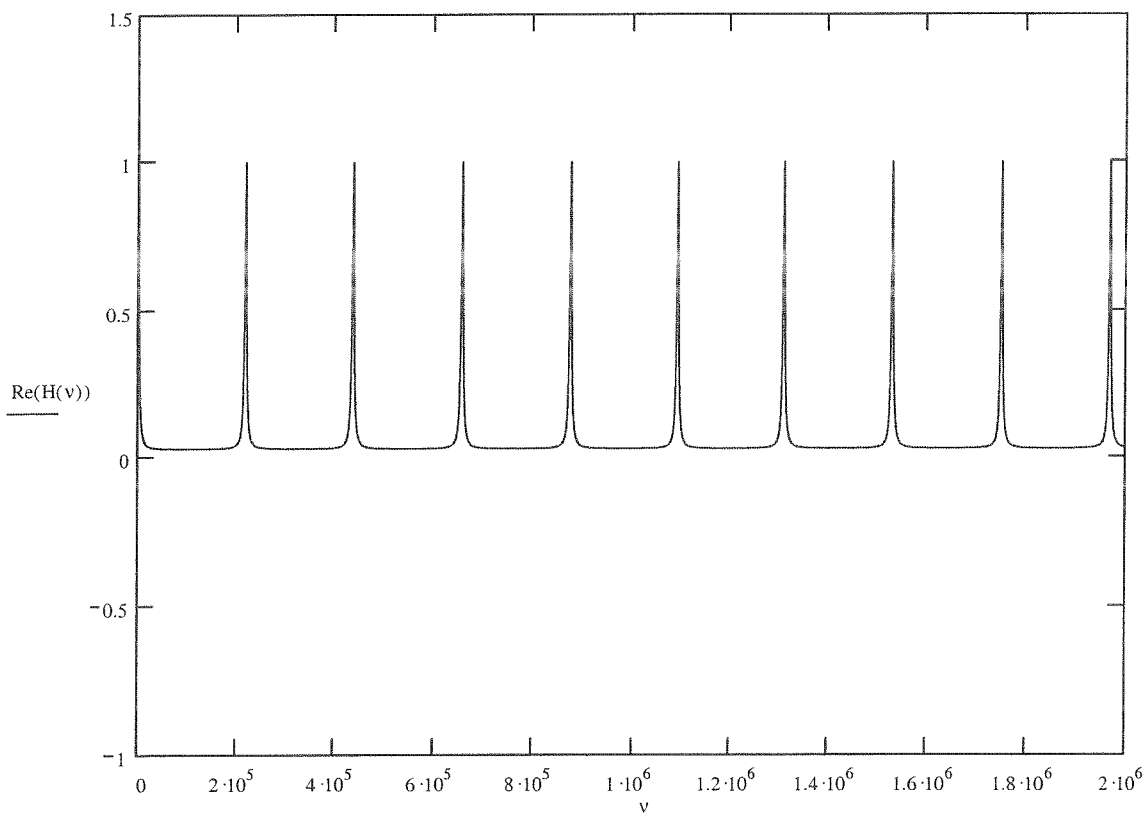
$\Delta := q_{xy} \cdot \nu_1 \rightarrow 63.3940 \cdot KHz$

$\nu_{s1} := \nu_1 - \Delta \rightarrow 155.2060 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 281.9940 \cdot KHz$

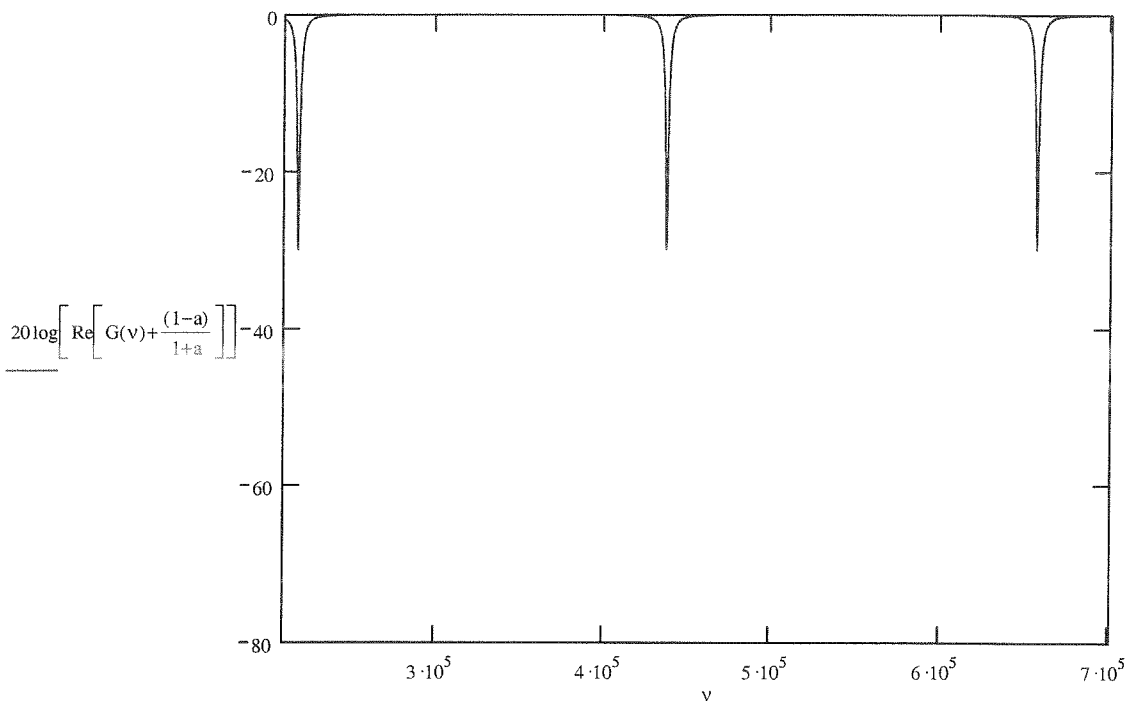
notch function

$$H(\nu) := \frac{1 - a}{1 - a \cdot e^{-i \cdot \frac{2 \cdot \pi \cdot \nu}{\nu_1}}} \qquad \nu := 0, 10^2 .. 10^7$$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20 \log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1-a)}{1+a} \right] \right] = -29.827 \quad \text{db}$$

-lower sidebands

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1-a)}{1+a} \right] \right] = -5.261 \times 10^{-3} \quad \text{db}$$

-upper sideband

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1-a)}{1+a} \right] \right] = -5.261 \times 10^{-3} \quad \text{db}$$

First harmonic $h := 2$

$c := 2^n$ $\quad a := \frac{(c-1)}{c}$ $\quad \nu_1 := h \cdot \nu_{rev} \rightarrow 437.2 \cdot KHz$

sidebands calculation

$\Delta := q_{xy} \cdot \nu_1 \rightarrow 126.7880 \cdot KHz$

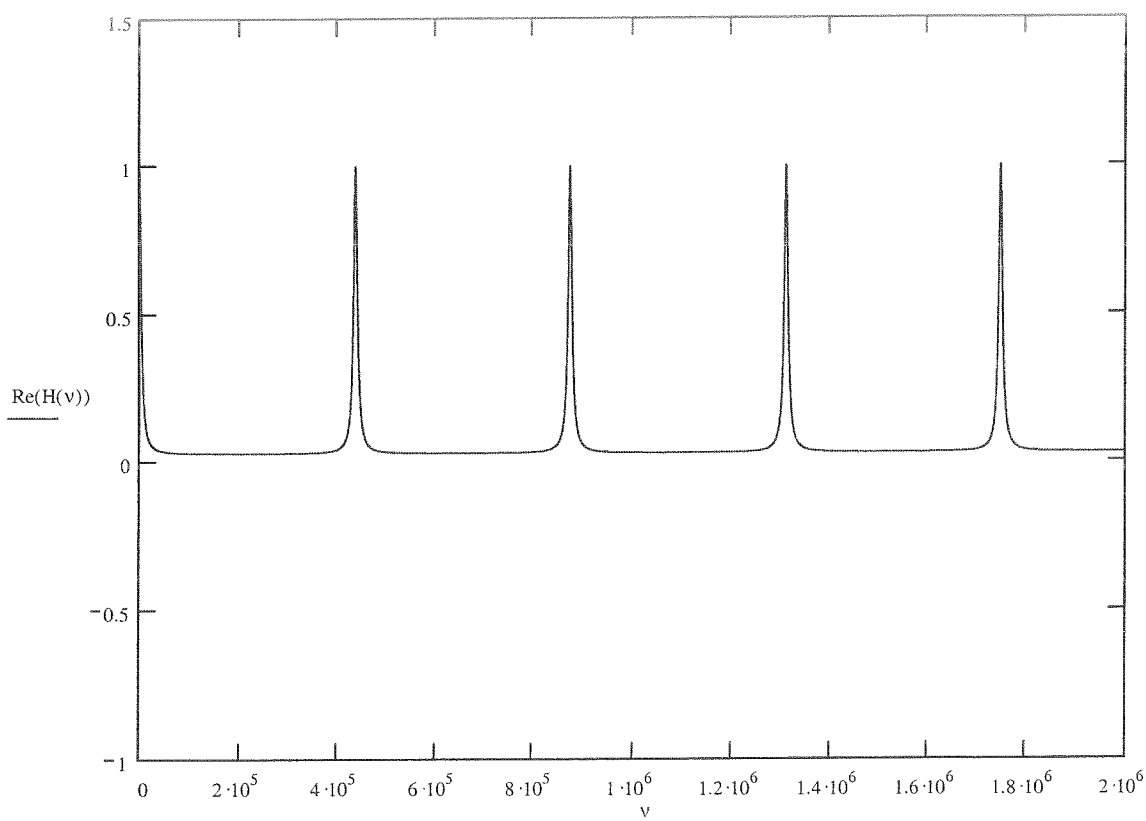
$\nu_{s1} := \nu_1 - \Delta \rightarrow 310.4120 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 563.9880 \cdot KHz$

notch function

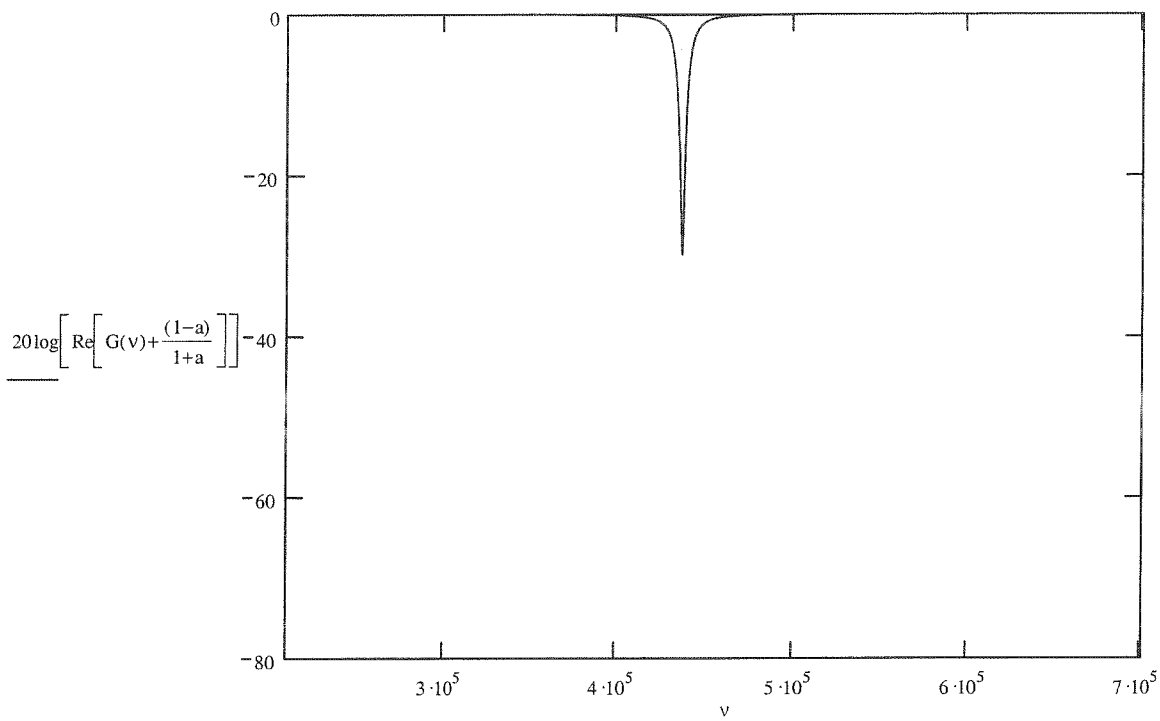
$$H(\nu) := \frac{1-a}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$

$\nu := 0, 10^2 .. 10^7$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20\log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1-a)}{1+a}\right]\right] = -29.827 \text{ db}$$

-lower sidebands

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1-a)}{1+a}\right]\right] = -5.261 \times 10^{-3} \text{ db}$$

-upper sideband

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1-a)}{1+a}\right]\right] = -5.261 \times 10^{-3} \text{ db}$$

Third harmonic $h := 3$

$c := 2^n$ $\quad a := \frac{(c-1)}{c}$ $\quad \nu_1 := h \cdot \nu_{rev} \rightarrow 655.8 \cdot KHz$

sidebands calculation

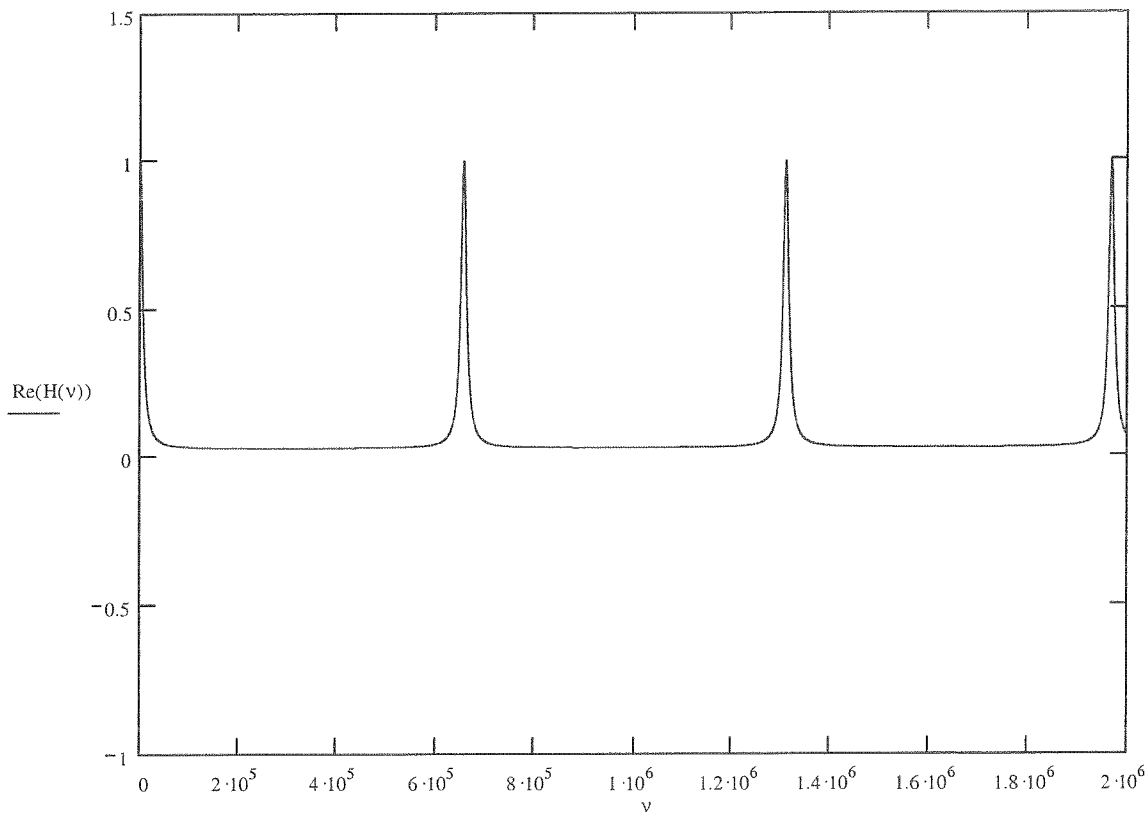
$\Delta := q_{xy} \cdot \nu_1 \rightarrow 190.1820 \cdot KHz$

$\nu_{s1} := \nu_1 - \Delta \rightarrow 465.6180 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 845.9820 \cdot KHz$

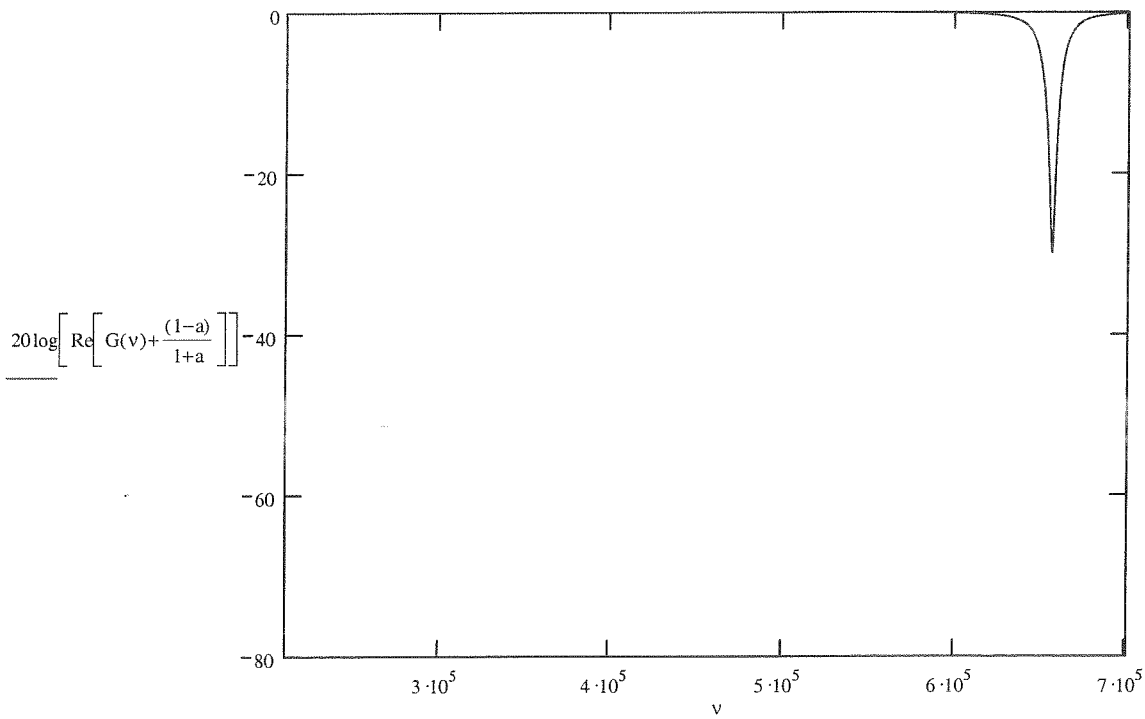
notch function

$$H(\nu) := \frac{1-a}{1-a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}} \qquad \nu := 0, 10^2 .. 10^7$$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20 \log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1 - a)}{1 + a} \right] \right] = -29.827 \quad \text{db}$$

-lower sidebands

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1 - a)}{1 + a} \right] \right] = -5.261 \times 10^{-3} \quad \text{db}$$

-upper sideband

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1 - a)}{1 + a} \right] \right] = -5.261 \times 10^{-3} \quad \text{db}$$

Bits number $n := 12$

Revolution frequency

$c := 2^n$ $\quad a := \frac{(c-1)}{c}$ $\quad \nu_1 := \nu_{rev} \rightarrow 218.6 \cdot KHz$

sidebands calculation

$\Delta := q_{xy} \cdot \nu_1 \rightarrow 63.3940 \cdot KHz$

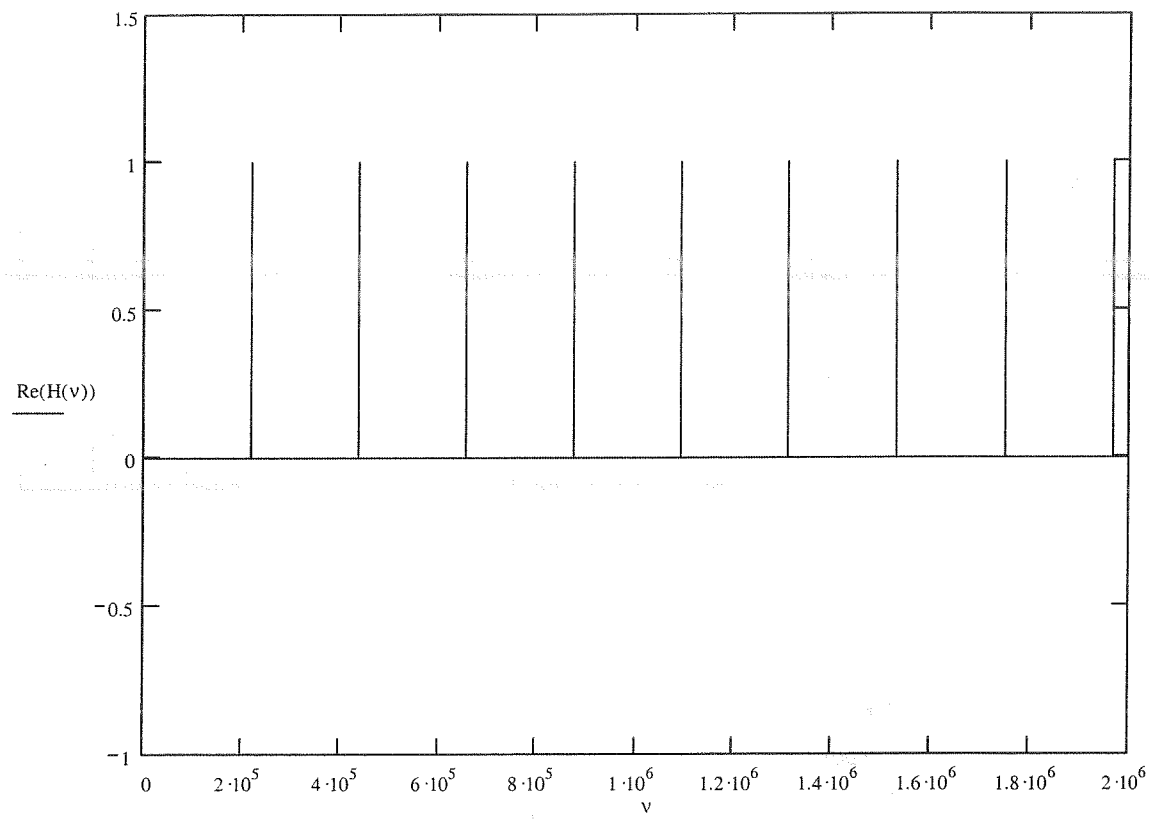
$\nu_{s1} := \nu_1 - \Delta \rightarrow 155.2060 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 281.9940 \cdot KHz$

notch function

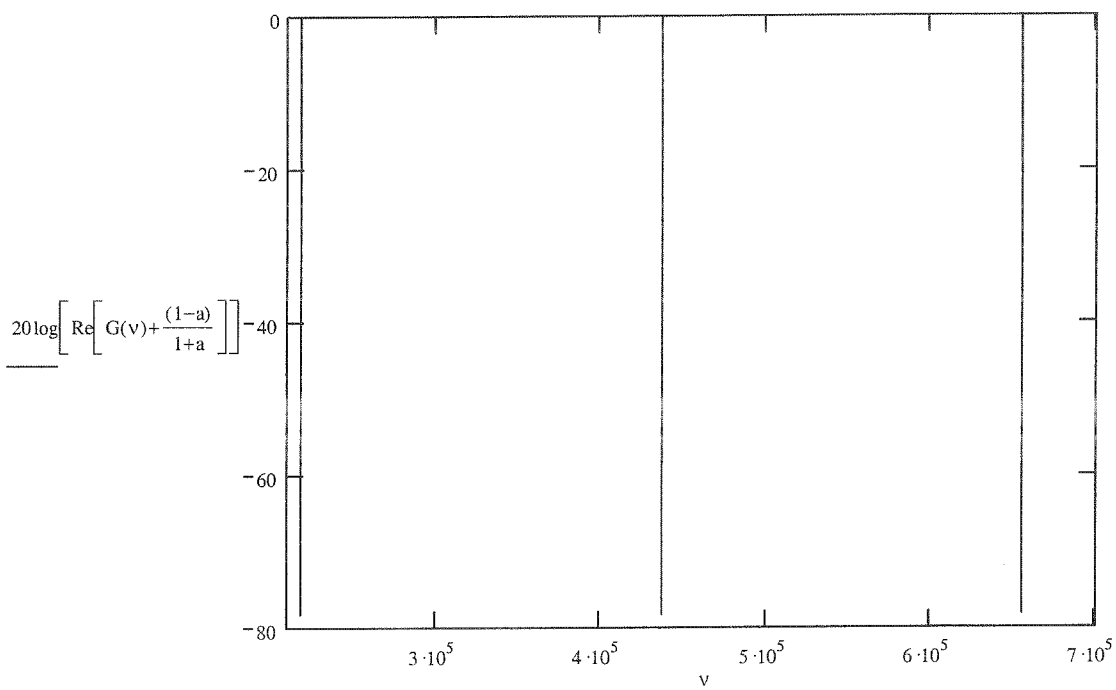
$$H(\nu) := \frac{1 - a}{1 - a \cdot e^{-i \cdot \frac{2 \cdot \pi \cdot \nu}{\nu_1}}}$$

$$\nu := 0, 10^2 .. 10^7$$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20\log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1-a)}{1+a}\right]\right] = -78.267 \text{ db}$$

-lower sidebands

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1-a)}{1+a}\right]\right] = -7.788 \times 10^{-8} \text{ db}$$

-upper sideband

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1-a)}{1+a}\right]\right] = -7.788 \times 10^{-8} \text{ db}$$

First harmonic $h := 2$

$c := 2^n$ $\quad a := \frac{(c-1)}{c}$ $\quad \nu_1 := h \cdot \nu_{rev} \rightarrow 437.2 \cdot KHz$

sidebands calculation

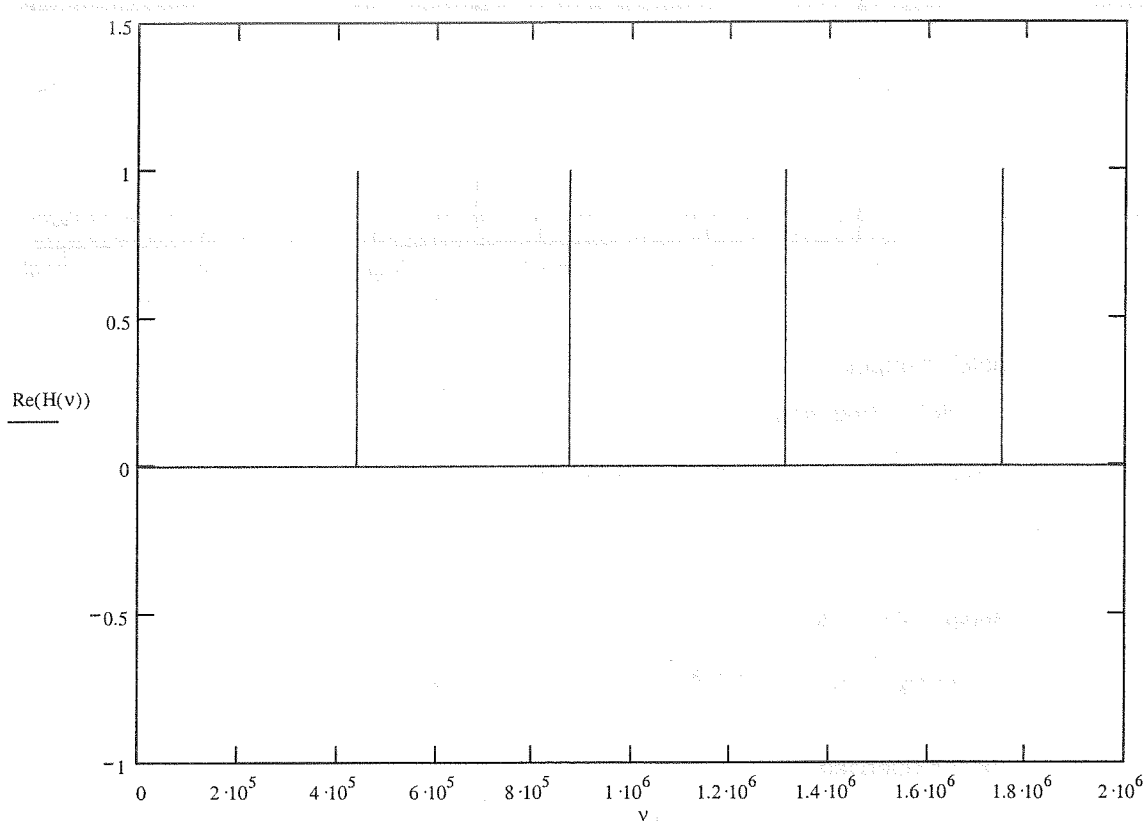
$\Delta := q_{xy} \cdot \nu_1 \rightarrow 126.7880 \cdot KHz$

$\nu_{s1} := \nu_1 - \Delta \rightarrow 310.4120 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 563.9880 \cdot KHz$

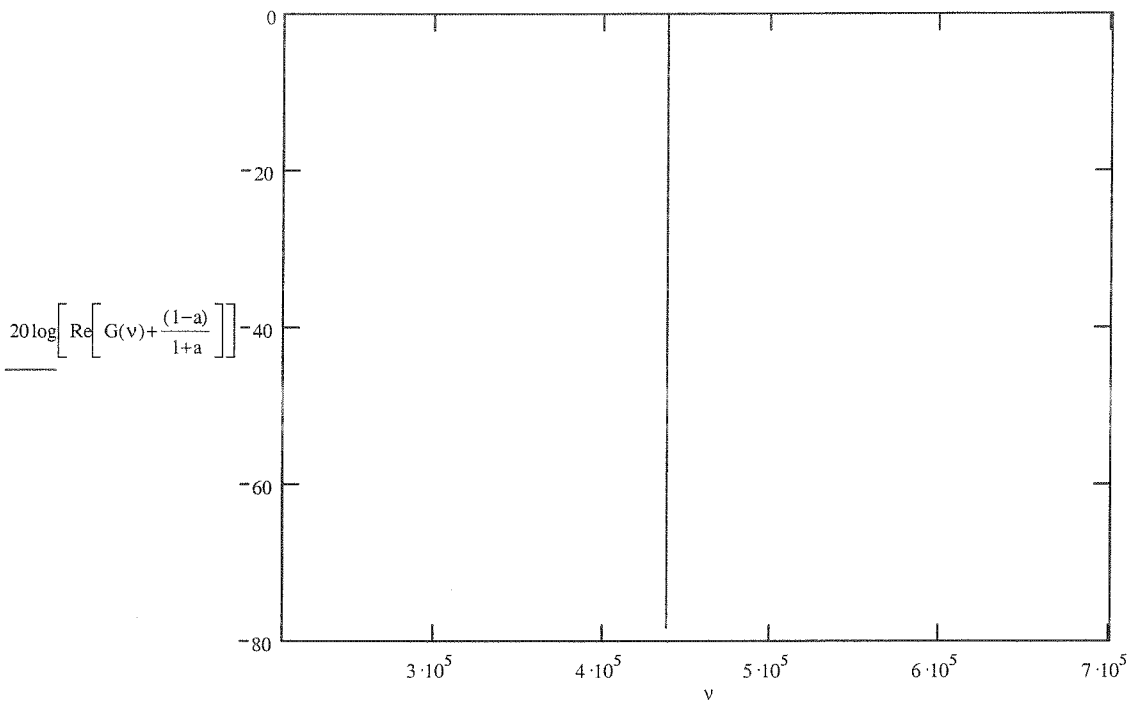
notch function

$$H(\nu) := \frac{1-a}{1-a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}} \qquad \nu := 0, 10^2 .. 10^7$$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20\log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1-a)}{1+a}\right]\right] = -78.267 \text{ db}$$

-lower sidebands

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1-a)}{1+a}\right]\right] = -7.788 \times 10^{-8} \text{ db}$$

-upper sideband

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1-a)}{1+a}\right]\right] = -7.788 \times 10^{-8} \text{ db}$$

Third harmonic $h := 3$

$c := 2^n$ $\quad a := \frac{(c-1)}{c}$ $\quad \nu_1 := h \cdot \nu_{rev} \rightarrow 655.8 \cdot KHz$

sidebands calculation

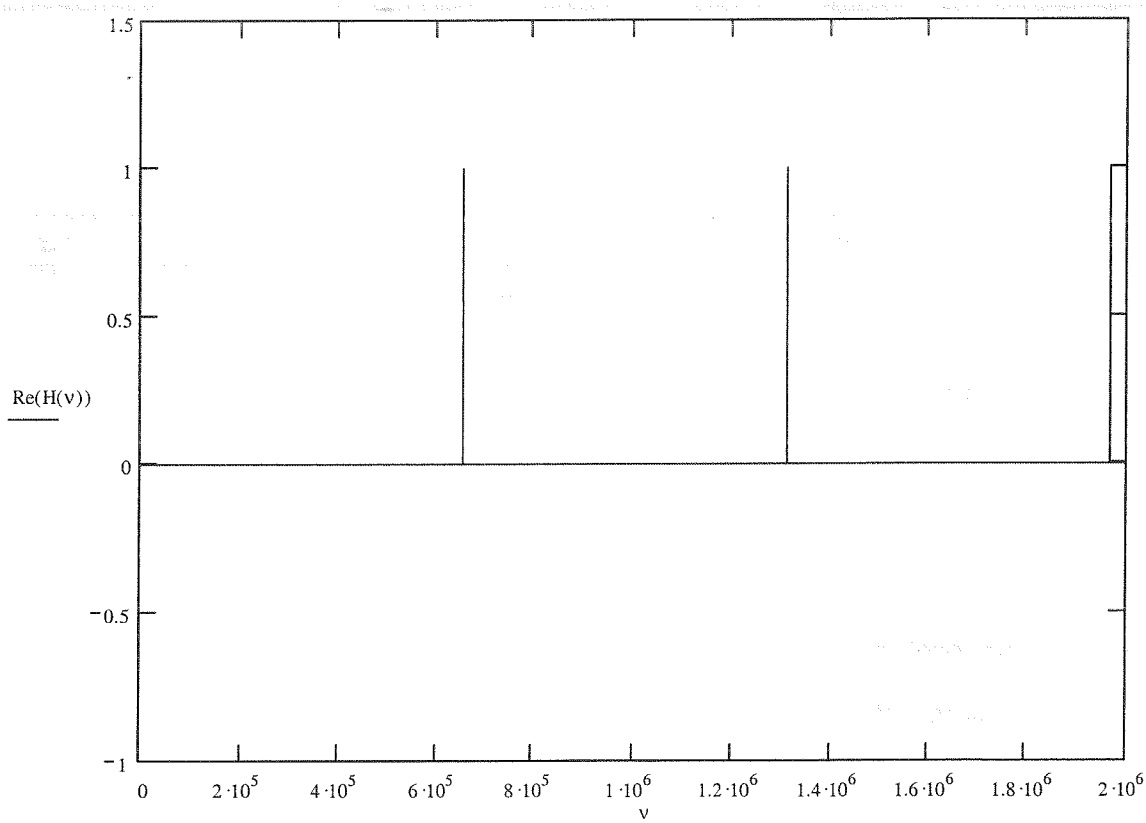
$\Delta := q_{xy} \cdot \nu_1 \rightarrow 190.1820 \cdot KHz$

$\nu_{s1} := \nu_1 - \Delta \rightarrow 465.6180 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 845.9820 \cdot KHz$

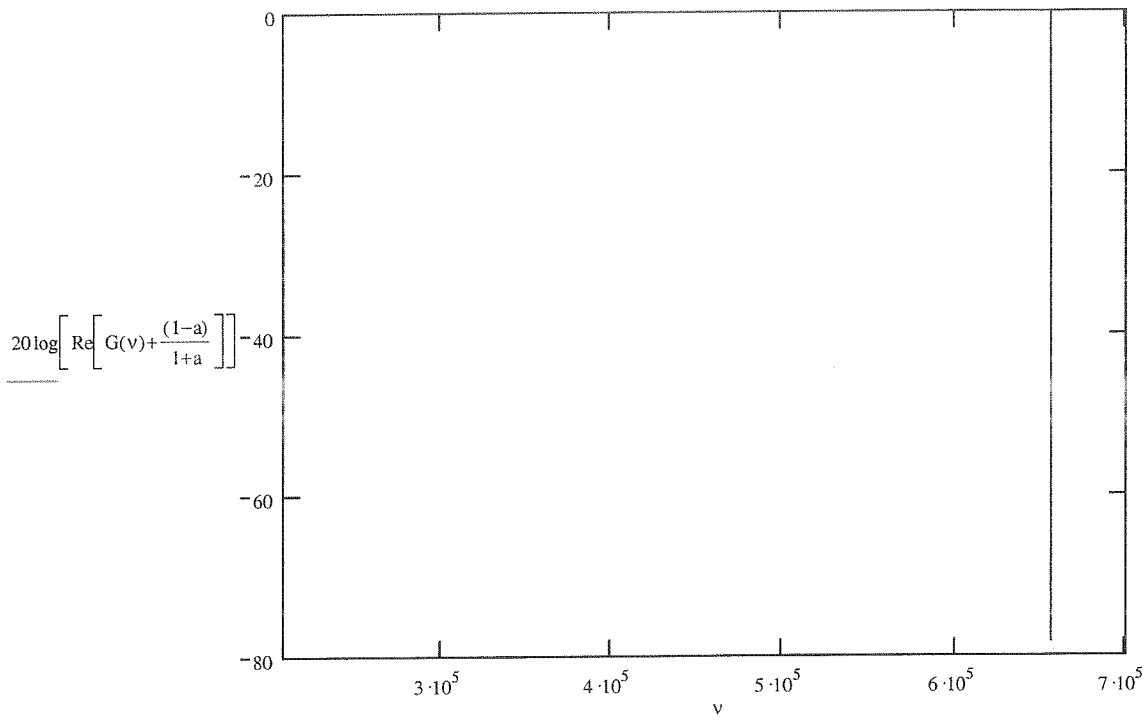
notch function

$$H(\nu) := \frac{1-a}{1-a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}} \qquad \nu := 0, 10^2 .. 10^7$$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20 \log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1-a)}{1+a} \right] \right] = -78.267 \text{ db}$$

-lower sidebands

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1-a)}{1+a} \right] \right] = -7.788 \times 10^{-8} \text{ db}$$

-upper sideband

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1-a)}{1+a} \right] \right] = -7.788 \times 10^{-8} \text{ db}$$

Bits number $n := 24$

Revolution frequency

$c := 2^n$ $\quad a := \frac{(c-1)}{c}$ $\quad \nu_1 := \nu_{rev} \rightarrow 218.6 \cdot KHz$

sidebands calculation

$\Delta := q_{xy} \cdot \nu_1 \rightarrow 63.3940 \cdot KHz$

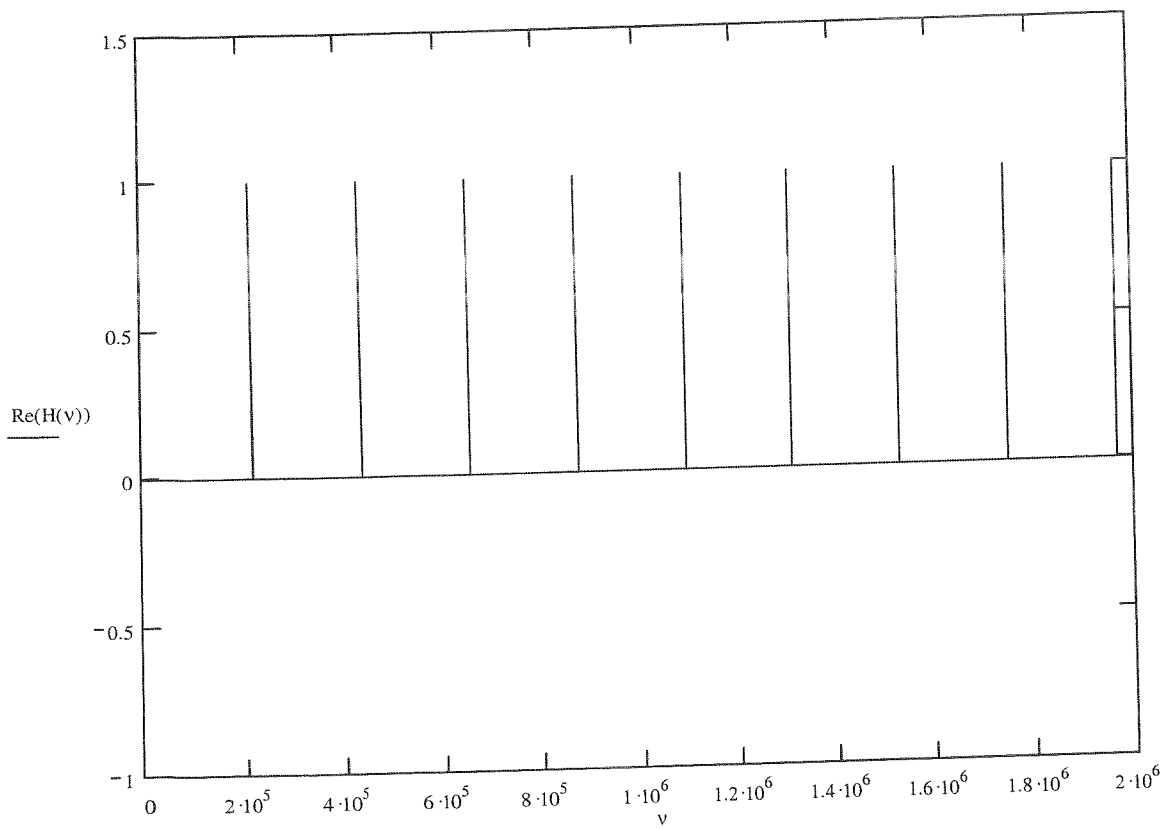
$\nu_{s1} := \nu_1 - \Delta \rightarrow 155.2060 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 281.9940 \cdot KHz$

notch function

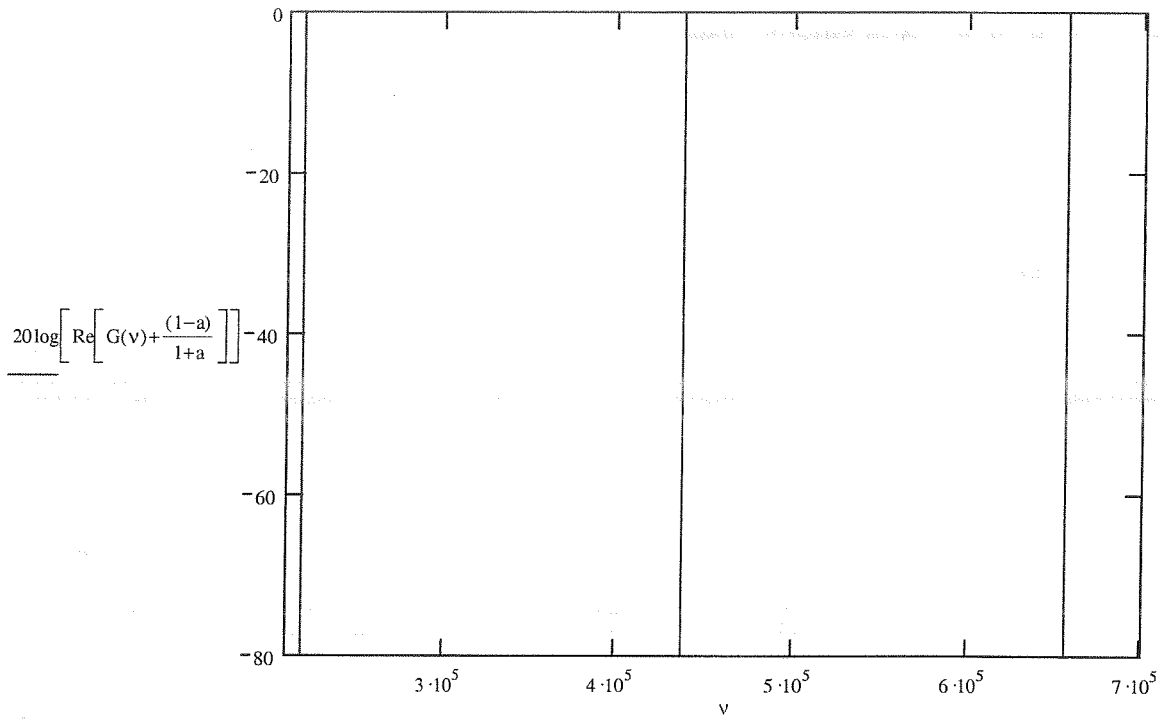
$$H(\nu) := \frac{1 - a}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$

$$\nu := 0, 10^2 .. 10^7$$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20\log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1-a)}{1+a}\right]\right] = -150.515\ \mathrm{db}$$

-lower sidebands

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1-a)}{1+a}\right]\right] = -4.822 \times 10^{-15}\ \mathrm{db}$$

-upper sideband

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1-a)}{1+a}\right]\right] = -4.822 \times 10^{-15}\ \mathrm{db}$$

First harmonic $h := 2$

$c := 2^n$ $\quad a := \frac{(c-1)}{c}$ $\quad \nu_1 := h \cdot \nu_{rev} \rightarrow 437.2 \cdot KHz$

sidebands calculation

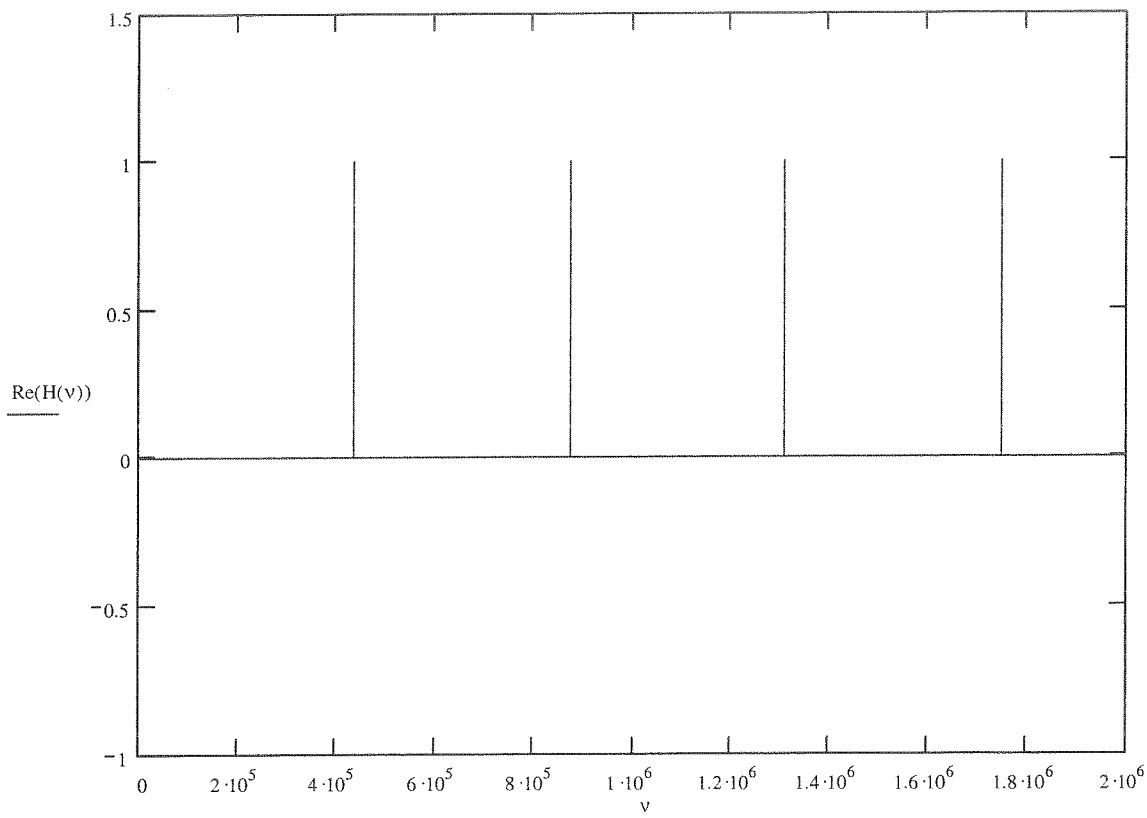
$\Delta := q_{xy} \cdot \nu_1 \rightarrow 126.7880 \cdot KHz$

$\nu_{s1} := \nu_1 - \Delta \rightarrow 310.4120 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 563.9880 \cdot KHz$

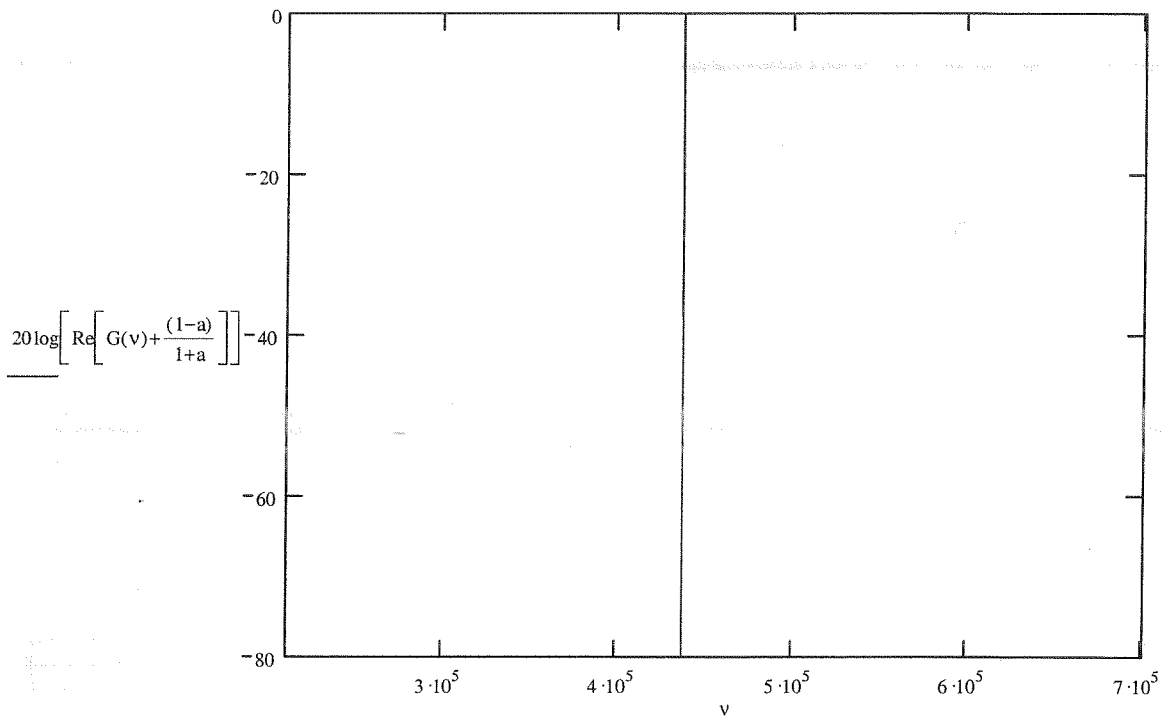
notch function

$$H(\nu) := \frac{1-a}{1-a \cdot e^{-i \cdot \frac{2 \cdot \pi \cdot \nu}{\nu_1}}} \qquad \nu := 0, 10^2 .. 10^7$$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20 \log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1-a)}{1+a}\right]\right] = -150.515 \text{ db}$$

-lower sidebands

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1-a)}{1+a}\right]\right] = -4.822 \times 10^{-15} \quad \text{db}$$

-upper sideband

$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1-a)}{1+a}\right]\right] = -4.822 \times 10^{-15} \quad \text{db}$$

Third harmonic $h := 3$

$c := 2^n$ $\quad a := \frac{(c-1)}{c}$ $\quad \nu_1 := h \cdot \nu_{rev} \rightarrow 655.8 \cdot KHz$

sidebands calculation

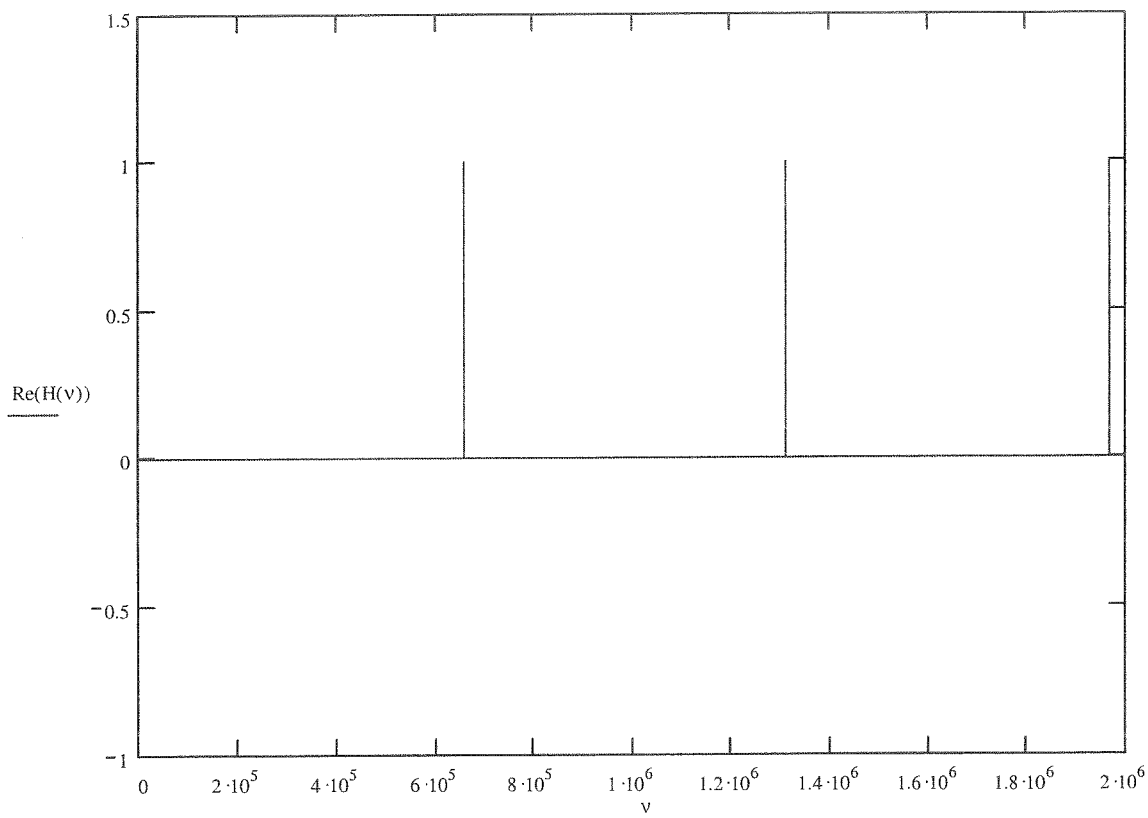
$\Delta := q_{xy} \cdot \nu_1 \rightarrow 190.1820 \cdot KHz$

$\nu_{s1} := \nu_1 - \Delta \rightarrow 465.6180 \cdot KHz$

$\nu_{s2} := \nu_1 + \Delta \rightarrow 845.9820 \cdot KHz$

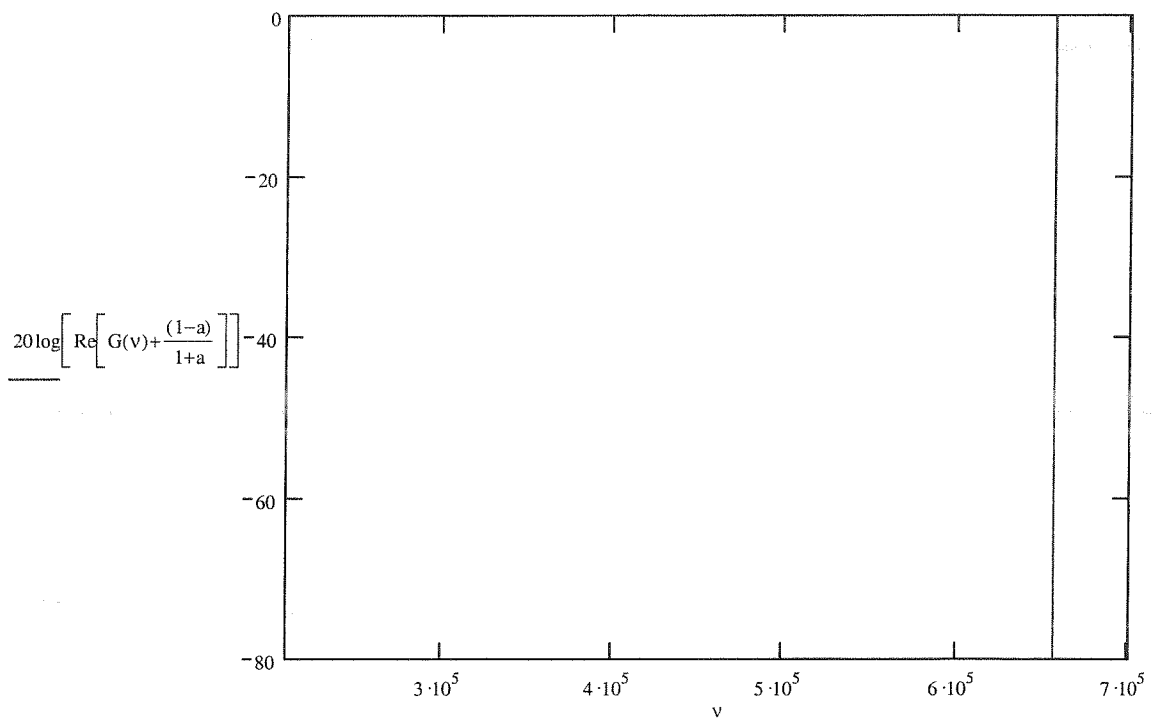
notch function

$$H(\nu) := \frac{1-a}{1-a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}} \qquad \nu := 0, 10^2 .. 10^7$$



transfer function

$$G(\nu) := \frac{a \cdot \left(1 - e^{-i \cdot \frac{2\pi\nu}{\nu_1}}\right)}{1 - a \cdot e^{-i \cdot \frac{2\pi\nu}{\nu_1}}}$$



notch damping

-revolution frequency

$$20 \log\left[\mathrm{Re}\left[G(\nu_1) + \frac{(1-a)}{1+a}\right]\right] = -150.515 \text{ db}$$

-lower sidebands

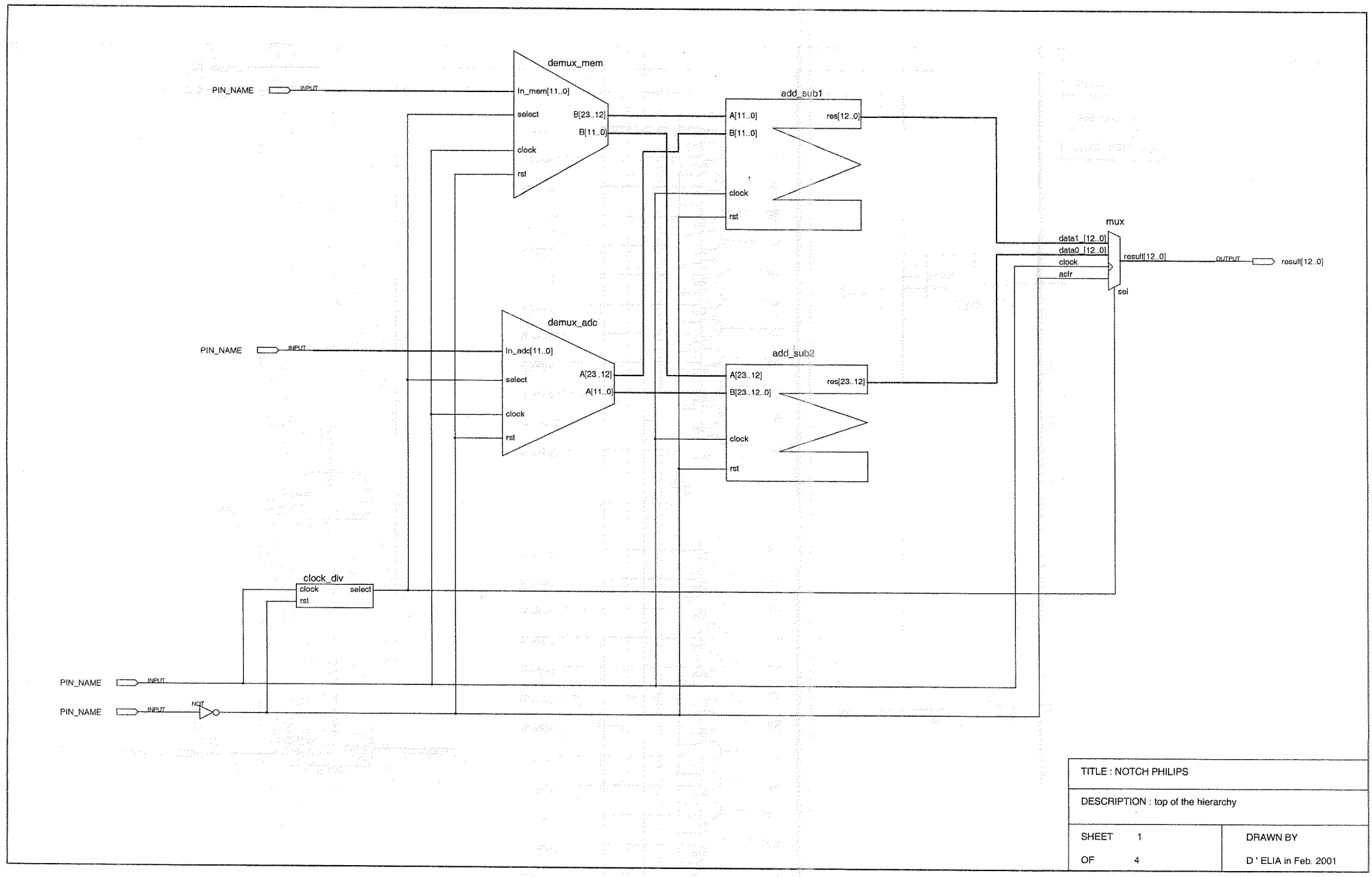
$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s1}) + \frac{(1-a)}{1+a}\right]\right] = -4.822 \times 10^{-15} \text{ db}$$

-upper sideband

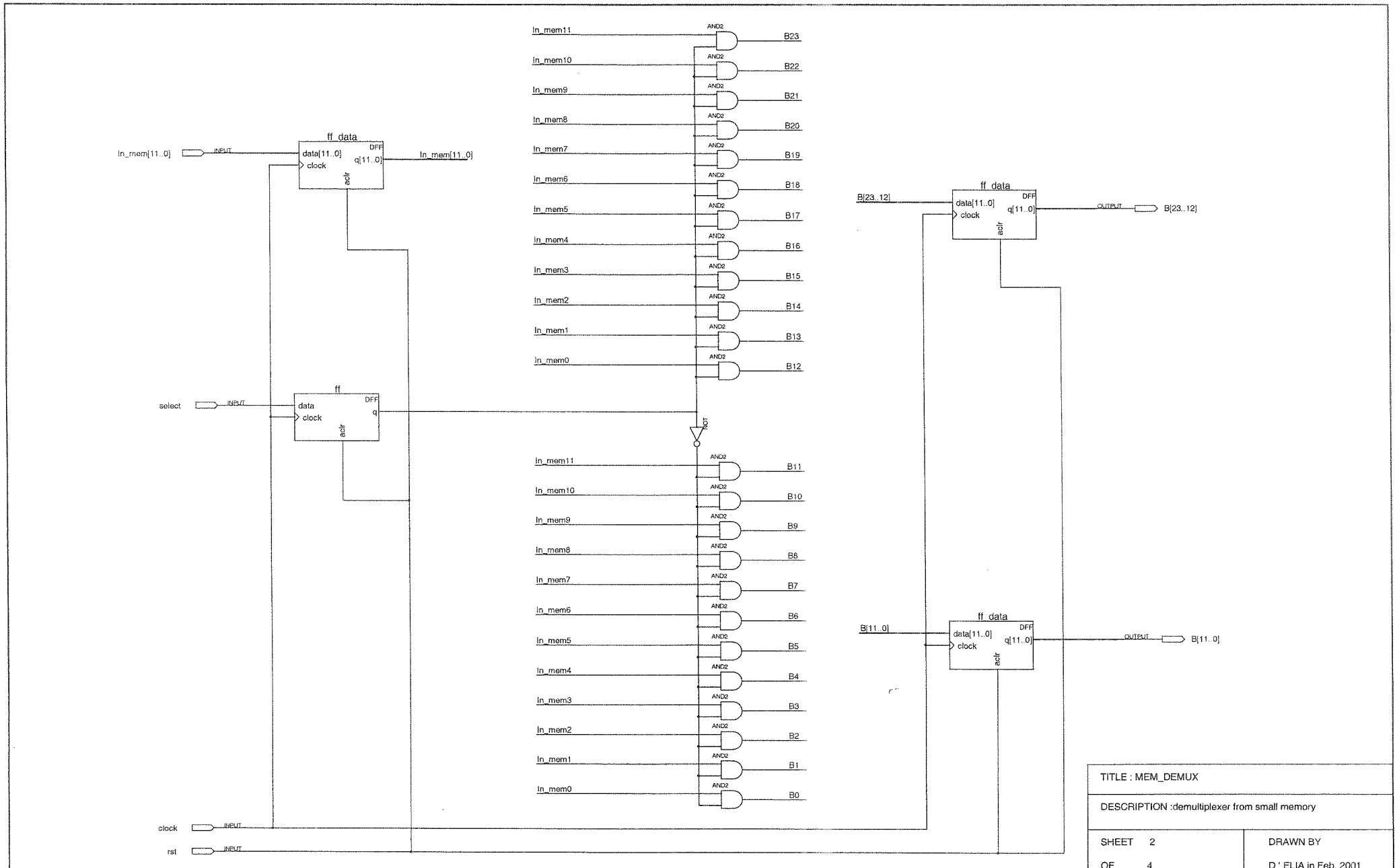
$$20 \cdot \log\left[\mathrm{Re}\left[G(\nu_{s2}) + \frac{(1-a)}{1+a}\right]\right] = -4.822 \times 10^{-15} \text{ db}$$

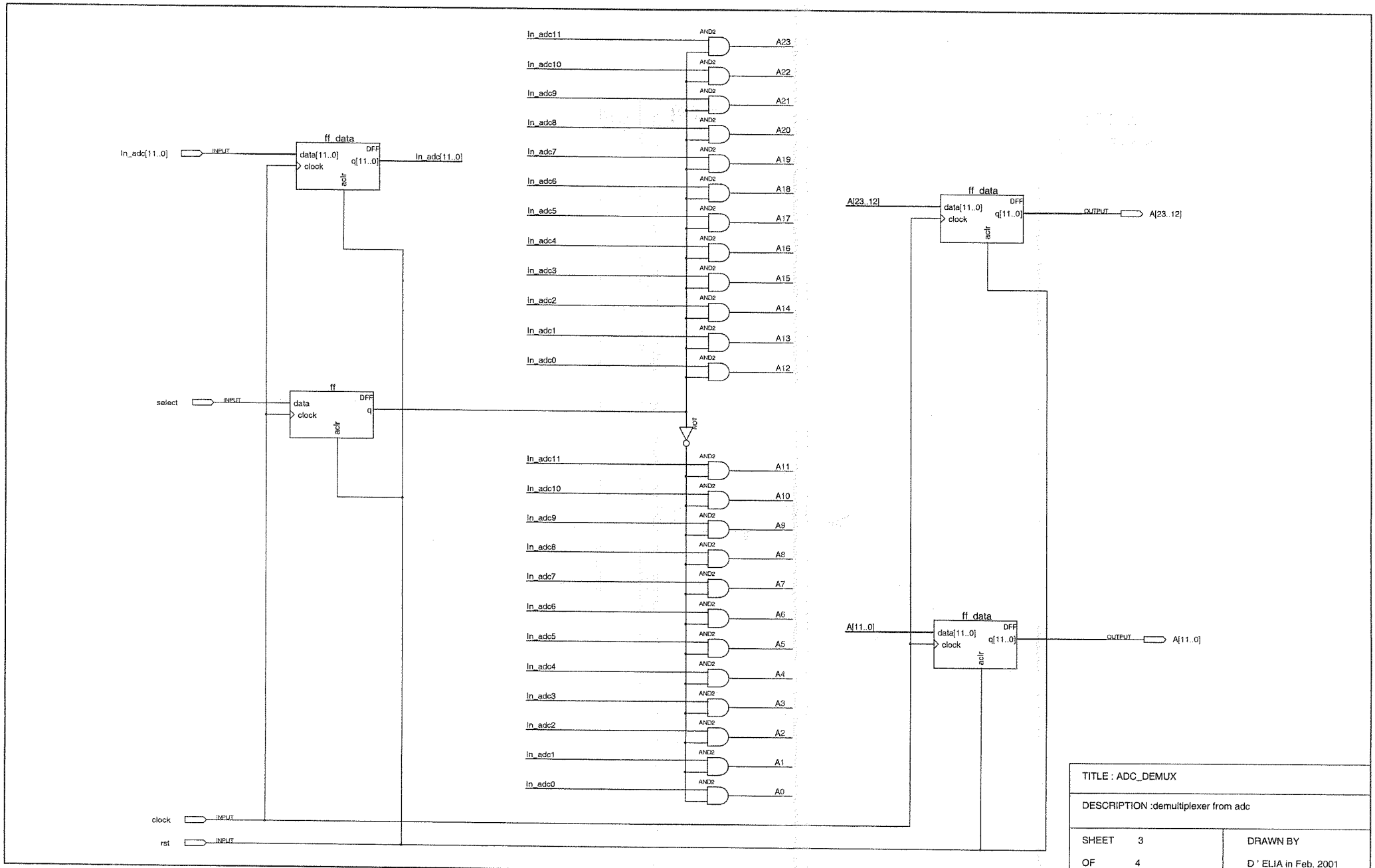
APPENDIX 2.

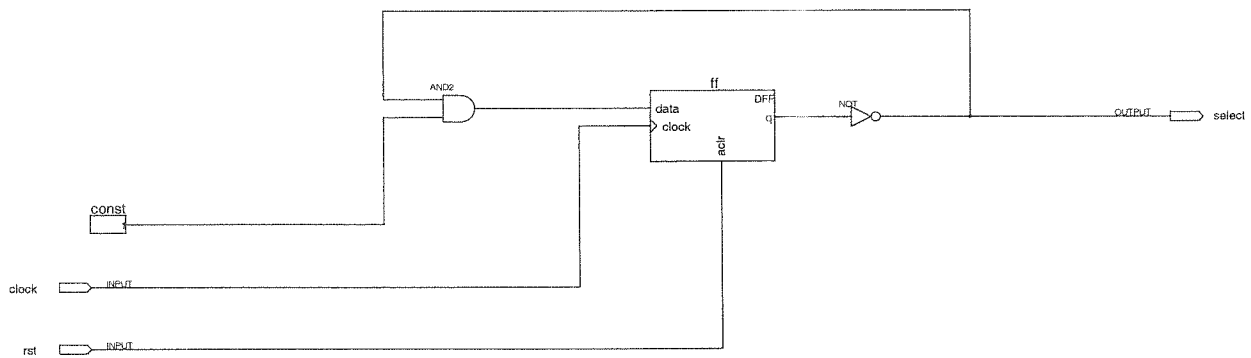
Philips schematics and phdl program.



TITLE : NOTCH PHILIPS	
DESCRIPTION : top of the hierarchy	
SHEET 1	DRAWN BY
OF 4	D ' ELIA in Feb. 2001


ff_data
In_mem[11..0]
INPUT
data[11..0]
clock
aclr
DFF
q[11..0]
In_mem[11..0]
ff
select
data
clock
aclr
DFF
q
NOT
In_mem11
In_mem10
In_mem9
In_mem8
In_mem7
In_mem6
In_mem5
In_mem4
In_mem3
In_mem2
In_mem1
In_mem0
AND2
B23
B22
B21
B20
B19
B18
B17
B16
B15
B14
B13
B12
B11
B10
B9
B8
B7
B6
B5
B4
B3
B2
B1
B0
B[23..12]
OUTPUT
B[11..0]
clock
rst
TITLE : MEM_DEMUX
DESCRIPTION :demultiplexer from small memory
SHEET 2
OF 4
DRAWN BY
D ' ELIA in Feb. 2001


In_adc[11..0]
INPUT
ff_data
data[11..0]
clock
aclr
DFF
q[11..0]
In_adc[11..0]
select
ff
data
clock
aclr
DFF
q
INPUT
clock
INPUT
rst
INPUT
In_adc11
In_adc10
In_adc9
In_adc8
In_adc7
In_adc6
In_adc5
In_adc4
In_adc3
In_adc2
In_adc1
In_adc0
AND2
A23
A22
A21
A20
A19
A18
A17
A16
A15
A14
A13
A12
NOT
A11
A10
A9
A8
A7
A6
A5
A4
A3
A2
A1
A0
A[23..12]
ff_data
data[11..0]
clock
aclr
DFF
q[11..0]
OUTPUT
A[23..12]
A[11..0]
ff_data
data[11..0]
clock
aclr
DFF
q[11..0]
OUTPUT
A[11..0]
TITLE : ADC_DEMUX
DESCRIPTION :demultiplexer from adc
SHEET 3
OF 4
DRAWN BY
D ' ELIA in Feb. 2001



TITLE : CLOCK_DIV	
DESCRIPTION : clock divisor x 2	
SHEET 4	DRAWN BY
OF 4	D ' ELIA in Feb. 2001

```
MODULE notch_sim
TITLE   'notch filter simulator'

DECLARATIONS

"inputs

in                  pin;
In_adc11..In_adc0       pin;
In_mem11..In_mem0       pin;
clk                 pin;
rst                 pin;

"nodes

A23..A0             node istype 'reg';
B23..B0             node istype 'reg';
select              node istype 'reg';
G10..G0             node istype 'com';
G22..G12            node istype 'com';
C11..C1             node istype 'com';
C23..C13            node istype 'com';
P10..P1             node istype 'com';
P22..P13            node istype 'com';
S10..S0             node istype 'reg';
S22..S12            node istype 'reg';
O12,O24             node istype 'reg';
Sig1,Sig2           node istype 'reg';

"output

out12..out0         pin istype 'reg';


EQUATIONS

"clock divisor x 2

select.ar=!rst;
select.clk=clk;
in=1;
select=in&!select;

"demultiplexer from adc

(A23..A0).ar=!rst;
(A23..A12).clk=clk;
(A11..A0).clk=clk;

(A23..A12)=(In_adc11..In_adc0)&select;
(A11..A0)=(In_adc11..In_adc0)&!select;

"demultiplexer from memory small

(B23..B0).ar=!rst;
(B23..B12).clk=clk;
(B11..B0).clk=clk;

(B23..B12)=(In_mem11..In_mem0)&select;
(B11..B0)=(In_mem11..In_mem0)&!select;

"adder_subtractor 1

(Sig1,O12,S10..S0).ar=!rst;
(Sig1,O12,S10..S0).clk=clk;
```

```
When (A11$B11==1) then
{
"subtractor

G0=A0&!B0;
G1=A1&!B1;
G2=A2&!B2;
G3=A3&!B3;


P1=A1#!B1;
P2=A2#!B2;
P3=A3#!B3;

C1=G0;
C2=G1#P1&G0;
C3=G2#P2&G1#P2&P1&G0;
C4=G3#P3&G2#P3&P2&G1#P3&P2&P1&G0;

G4=A4&!B4;
G5=A5&!B5;
G6=A6&!B6;
G7=A7&!B7;

P4=A4#!B4;
P5=A5#!B5;
P6=A6#!B6;
P7=A7#!B7;

C5=G4#P4&C4;
C6=G5#P5&G4#P5&P4&C4;
C7=G6#P6&G5#P6&P5&G4#P6&P5&P4&C4;
C8=G7#P7&G6#P7&P6&G5#P7&P6&P5&G4#P7&P6&P5&P4&C4;

G8=A8&!B8;
G9=A9&!B9;
G10=A10&!B10;

P8=A8#!B8;
P9=A9#!B9;
P10=A10#!B10;

C9=G8#P8&C8;
C10=G9#P9&G8#P9&P8&C8;
C11=G10#P10&C10;
{
  when (C11==0) then
   {
    S0:=!(A0$!B0);
    S1:=!(G1$P1$C1);
    S2:=!(G2$P2$C2);
    S3:=!(G3$P3$C3);
    S4:=!(G4$P4$C4);
    S5:=!(G5$P5$C5);
    S6:=!(G6$P6$C6);
    S7:=!(G7$P7$C7);
    S8:=!(G8$P8$C8);
    S9:=!(G9$P9$C9);
    S10:=!(G10$P10$C10);
   }
  else
   {
    G0=A0&!B0;
    G1=A1&!B1;
```

```
    G2=A2&!B2;
    G3=A3&!B3;


    P1=A1#!B1;
    P2=A2#!B2;
    P3=A3#!B3;

    C1=G0#C11;
    C2=G1#P1&C1;
    C3=G2#P2&G1#P2&P1&C1;
    C4=G3#P3&G2#P3&P2&G1#P3&P2&P1&C1;

    G4=A4&!B4;
    G5=A5&!B5;
    G6=A6&!B6;
    G7=A7&!B7;

    P4=A4#!B4;
    P5=A5#!B5;
    P6=A6#!B6;
    P7=A7#!B7;

    C5=G4#P4&C4;
    C6=G5#P5&G4#P5&P4&C4;
    C7=G6#P6&G5#P6&P5&G4#P6&P5&P4&C4;
    C8=G7#P7&G6#P7&P6&G5#P7&P6&P5&G4#P7&P6&P5&P4&C4;

    G8=A8&!B8;
    G9=A9&!B9;
    G10=A10&!B10;

    P8=A8#!B8;
    P9=A9#!B9;
    P10=A10#!B10;

    C9=G8#P8&C8;
    C10=G9#P9&G8#P9&P8&C8;
    C11=G10#P10&C10;

     S0:=A0$!B0$C11;
     S1:=G1$P1$C1;
     S2:=G2$P2$C2;
     S3:=G3$P3$C3;
     S4:=G4$P4$C4;
     S5:=G5$P5$C5;
     S6:=G6$P6$C6;
     S7:=G7$P7$C7;
     S8:=G8$P8$C8;
     S9:=G9$P9$C9;
     S10:=G10$P10$C10;
    }
   }
}
else
{
"adder
 G0=A0&B0;
 G1=A1&B1;
 G2=A2&B2;
 G3=A3&B3;


 P1=A1#B1;
 P2=A2#B2;
```

```
  P3=A3#B3;

  C1=G0;
  C2=G1#P1&G0;
  C3=G2#P2&G1#P2&P1&G0;
  C4=G3#P3&G2#P3&P2&G1#P3&P2&P1&G0;

  G4=A4&B4;
  G5=A5&B5;
  G6=A6&B6;
  G7=A7&B7;

  P4=A4#B4;
  P5=A5#B5;
  P6=A6#B6;
  P7=A7#B7;

  C5=G4#P4&C4;
  C6=G5#P5&G4#P5&P4&C4;
  C7=G6#P6&G5#P6&P5&G4#P6&P5&P4&C4;
  C8=G7#P7&G6#P7&P6&G5#P7&P6&P5&G4#P7&P6&P5&P4&C4;

  G8=A8&B8;
  G9=A9&B9;
  G10=A10&B10;

  P8=A8#B8;
  P9=A9#B9;
  P10=A10#B10;

  C9=G8#P8&C8;
  C10=G9#P9&G8#P9&P8&C8;
  C11=G10#P10&C10;

  S0:=A0$B0;
  S1:=G1$P1$C1;
  S2:=G2$P2$C2;
  S3:=G3$P3$C3;
  S4:=G4$P4$C4;
  S5:=G5$P5$C5;
  S6:=G6$P6$C6;
  S7:=G7$P7$C7;
  S8:=G8$P8$C8;
  S9:=G9$P9$C9;
  S10:=G10$P10$C10;
}
Sig1:=(A11&C11)#(B11&!C11);
O12:=C11&(!B11&!A11#B11&A11);

"adder_subtractor 2

(Sig2,O24,S22..S12).ar=!rst;
(Sig2,O24,S22..S12).clk=clk;

when (A23$B23==1) then
{
 "subtractor

G12=A12&!B12;
G13=A13&!B13;
G14=A14&!B14;
G15=A15&!B15;


P13=A13#!B13;
```

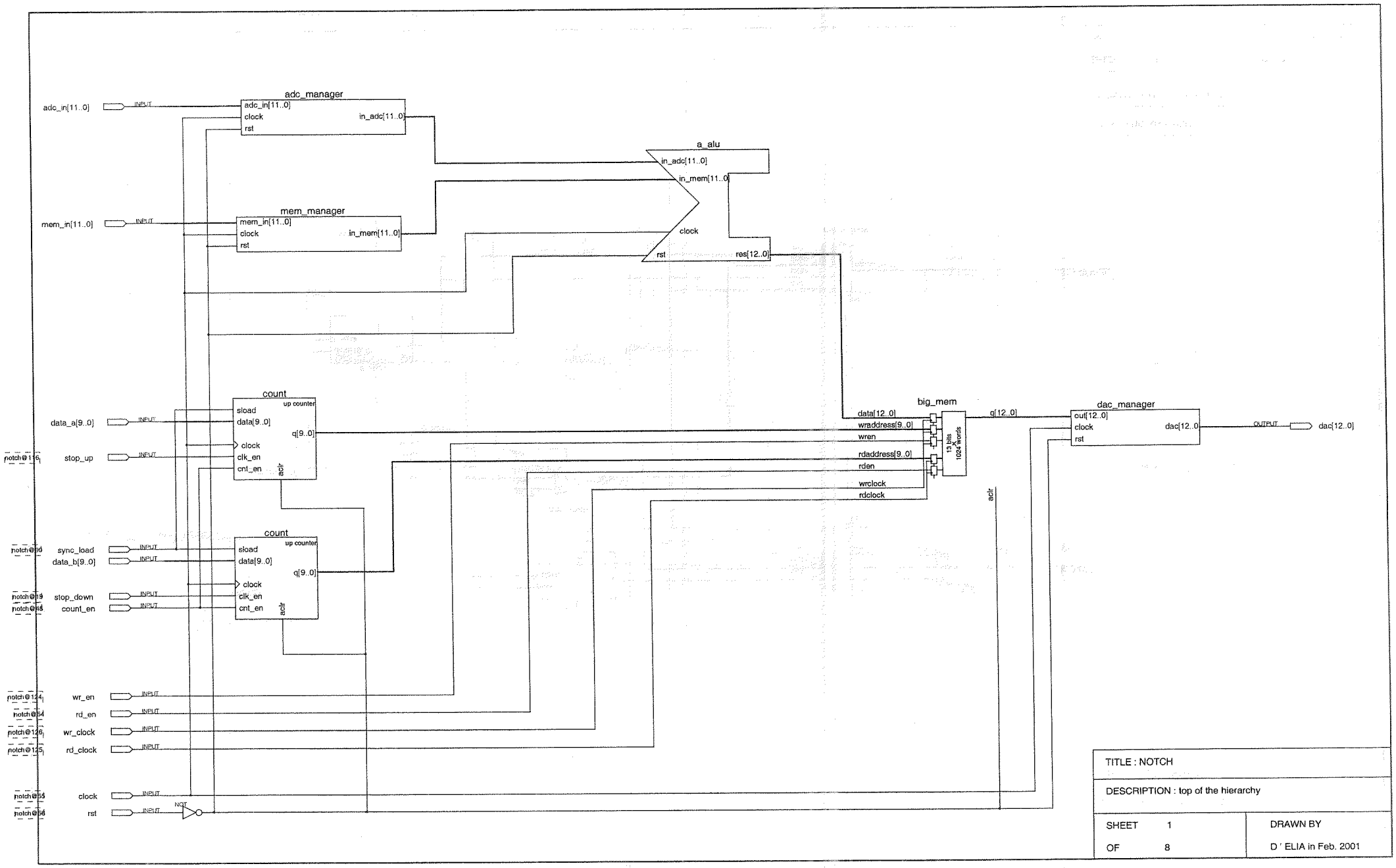
```
P14=A14#!B14;
P15=A15#!B15;

C13=G12;
C14=G13#P13&G12;
C15=G14#P14&G13#P14&P13&G12;
C16=G15#P15&G14#P15&P14&G13#P15&P14&P13&G12;

G16=A16&!B16;
G17=A17&!B17;
G18=A18&!B18;
G19=A19&!B19;

P16=A16#!B16;
P17=A17#!B17;
P18=A18#!B18;
P19=A19#!B19;

C17=G16#P16&C16;
C18=G17#P17&G16#P17&P16&C16;
C19=G18#P18&G17#P18&P17&G16#P18&P17&P16&C16;
C20=G19#P19&G18#P19&P18&G17#P19&P18&P17&G16#P19&P18&P17&P16&C16;

G20=A20&!B20;
G21=A21&!B21;
G22=A22&!B22;

P20=A20#!B20;
P21=A21#!B21;
P22=A22#!B22;

C21=G20#P20&C20;
C22=G21#P21&G20#P21&P20&C20;
C23=G22#P22&C22;
 {
  when (C23==0) then
  {
   S12:=!(A12$!B12);
   S13:=!(G13$P13$C13);
   S14:=!(G14$P14$C14);
   S15:=!(G15$P15$C15);
   S16:=!(G16$P16$C16);
   S17:=!(G17$P17$C17);
   S18:=!(G18$P18$C18);
   S19:=!(G19$P19$C19);
   S20:=!(G20$P20$C20);
   S21:=!(G21$P21$C21);
   S22:=!(G22$P22$C22);
  }
  else
  {
   G12=A12&!B12;
   G13=A13&!B13;
   G14=A14&!B14;
   G15=A15&!B15;


   P13=A13#!B13;
   P14=A14#!B14;
   P15=A15#!B15;

   C13=G12#C23;
   C14=G13#P13&C23;
   C15=G14#P14&G13#P14&P13&C23;
   C16=G15#P15&G14#P15&P14&G13#P15&P14&P13&C23;
```

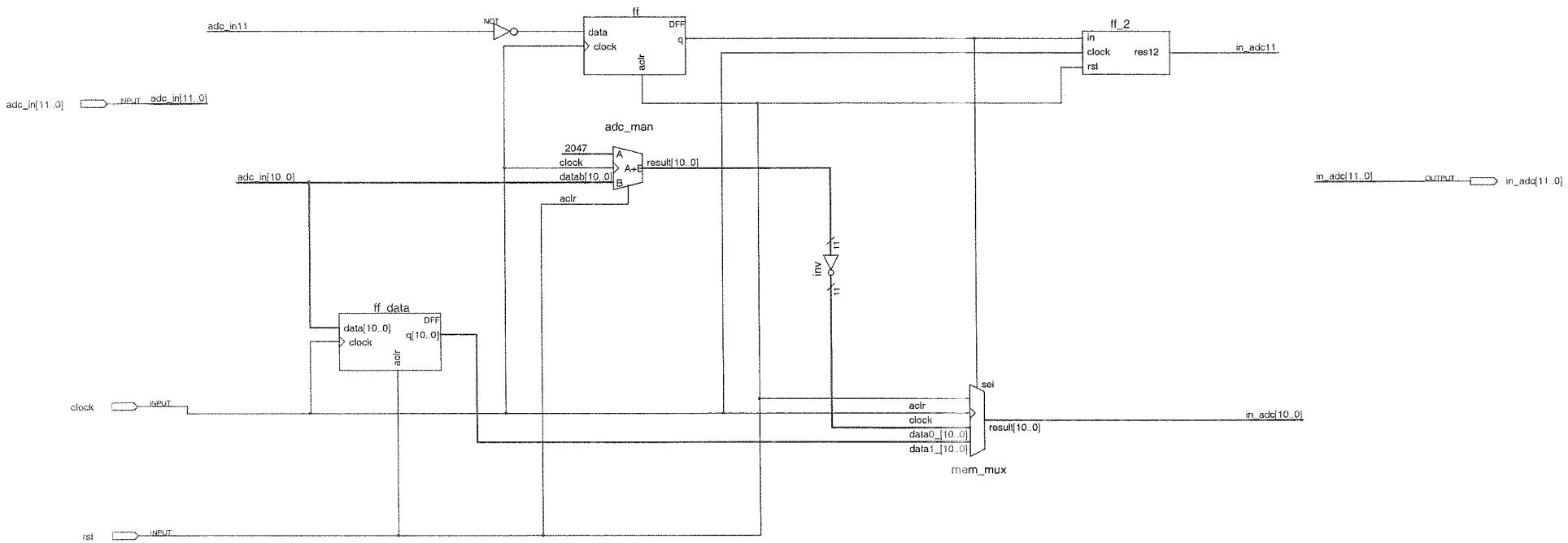
```
  G16=A16&!B16;
  G17=A17&!B17;
  G18=A18&!B18;
  G19=A19&!B19;

  P16=A16#!B16;
  P17=A17#!B17;
  P18=A18#!B18;
  P19=A19#!B19;

  C17=G16#P16&C16;
  C18=G17#P17&G16#P17&P16&C16;
  C19=G18#P18&G17#P18&P17&G16#P18&P17&P16&C16;
  C20=G19#P19&G18#P19&P18&G17#P19&P18&P17&G16#P19&P18&P17&P16&C16;

  G20=A20&!B20;
  G21=A21&!B21;
  G22=A22&!B22;

  P20=A20#!B20;
  P21=A21#!B21;
  P22=A22#!B22;

  C21=G20#P20&C20;
  C22=G21#P21&G20#P21&P20&C20;
  C23=G22#P22&C22;

  S12:=A12$!B12$C23;
  S13:=G13$P13$C13;
  S14:=G14$P14$C14;
  S15:=G15$P15$C15;
  S16:=G16$P16$C16;
  S17:=G17$P17$C17;
  S18:=G18$P18$C18;
  S19:=G19$P19$C19;
  S20:=G20$P20$C20;
  S21:=G21$P21$C21;
  S22:=G22$P22$C22;
  }
 }
}
else
 {
  G12=A12&B12;
  G13=A13&B13;
  G14=A14&B14;
  G15=A15&B15;


  P13=A13#B13;
  P14=A14#B14;
  P15=A15#B15;

  C13=G12;
  C14=G13#P13&G12;
  C15=G14#P14&G13#P14&P13&G12;
  C16=G15#P15&G14#P15&P14&G13#P15&P14&P13&G12;

  G16=A16&B16;
  G17=A17&B17;
  G18=A18&B18;
  G19=A19&B19;

  P16=A16#B16;
```

```
  P17=A17#B17;
  P18=A18#B18;
  P19=A19#B19;

  C17=G16#P16&C16;
  C18=G17#P17&G16#P17&P16&C16;
  C19=G18#P18&G17#P18&P17&G16#P18&P17&P16&C16;
  C20=G19#P19&G18#P19&P18&G17#P19&P18&P17&G16#P19&P18&P17&P16&C16;

  G20=A20&B20;
  G21=A21&B21;
  G22=A22&B22;

  P20=A20#B20;
  P21=A21#B21;
  P22=A22#B22;


  C21=G20#P20&C20;
  C22=G21#P21&G20#P21&P20&C20;
  C23=G22#P22&C22;

   S12:=A12$B12;
   S13:=G13$P13$C13;
   S14:=G14$P14$C14;
   S15:=G15$P15$C15;
   S16:=G16$P16$C16;
   S17:=G17$P17$C17;
   S18:=G18$P18$C18;
   S19:=G19$P19$C19;
   S20:=G20$P20$C20;
   S21:=G21$P21$C21;
   S22:=G22$P22$C22;
  }
Sig2:=(A23&C23)#(B23&!C23);
O24:=C23&(!B23&!A23#B23&A23);

"multiplexer to big memory

(out12..out0).ar=!rst;
(out12..out0).clk=clk;
(out12..out0):=((Sig2,O24,S22..S12)&select)#((Sig1,O12,S10..S0)&!select);


END
```

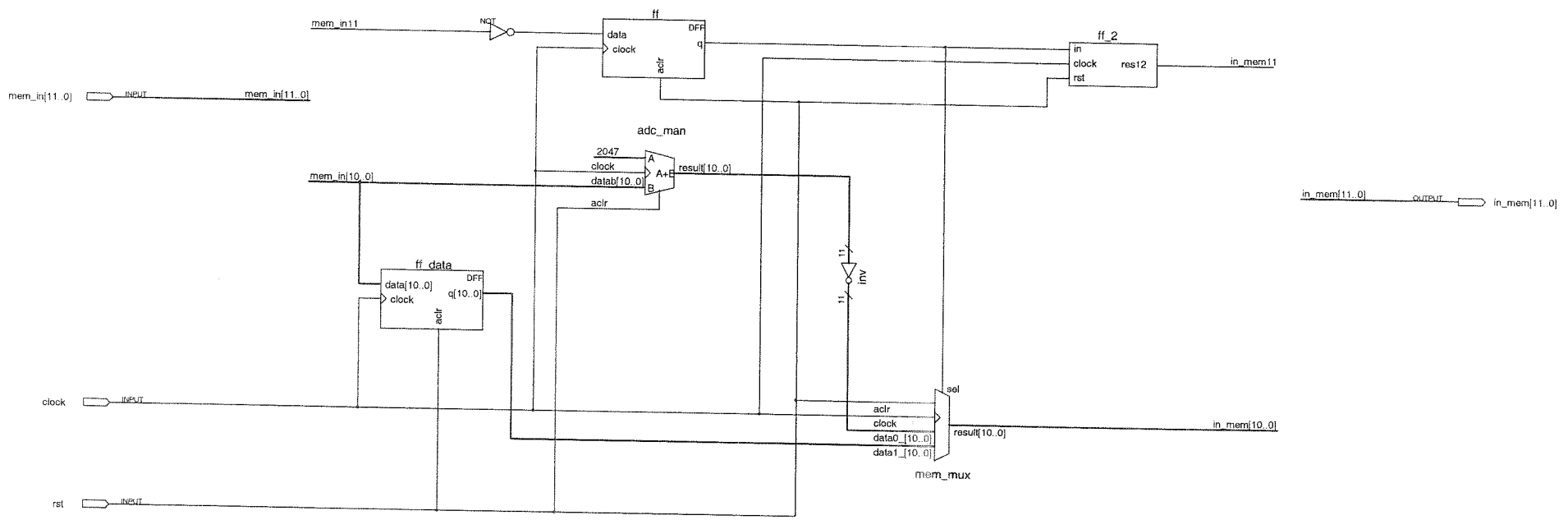
APPENDIX 3.

Altera schematics.

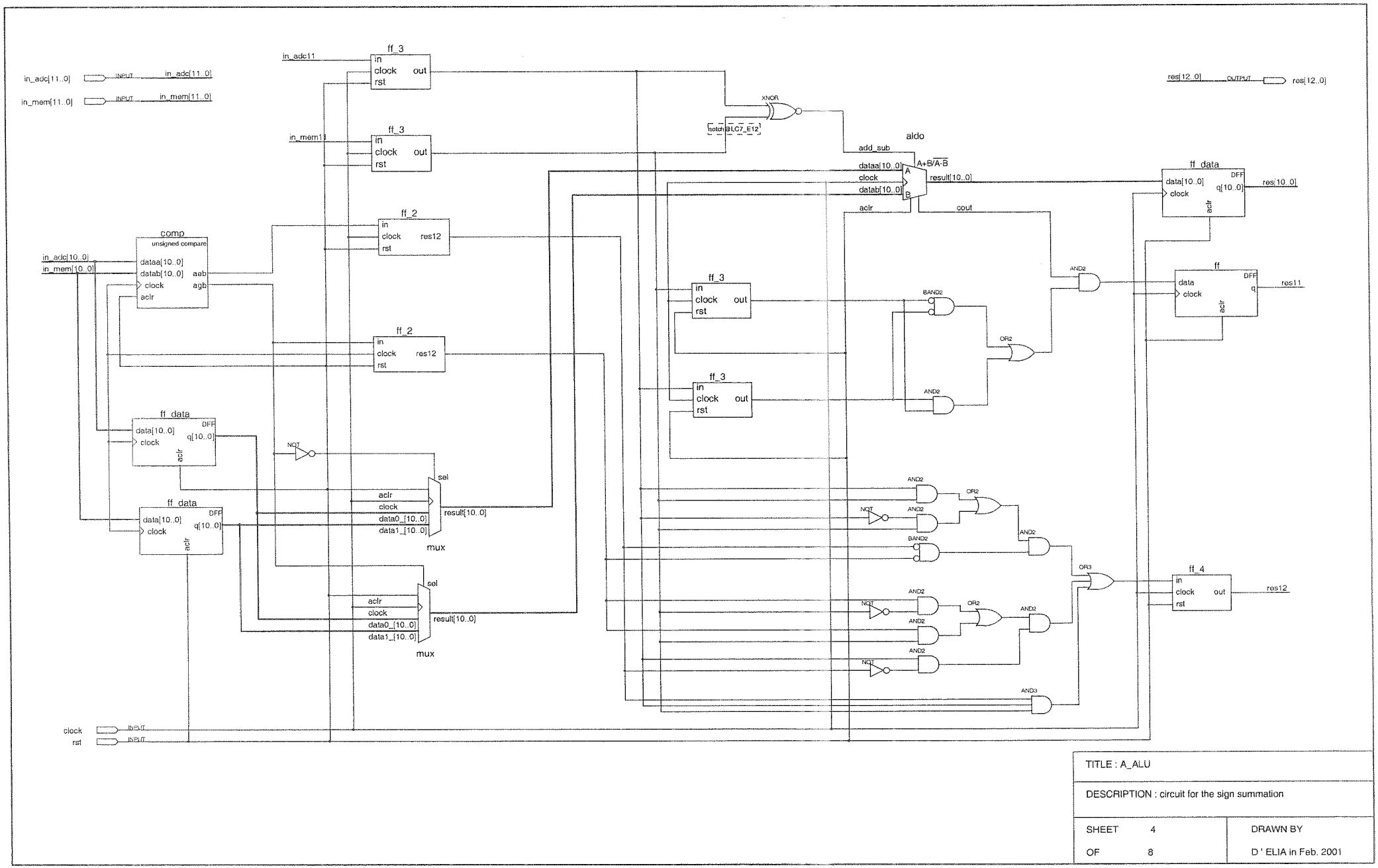

adc_manager
adc_in[11..0]
clock
rst
in_adc[11..0]
adc_in[11..0]
INPUT
a_alu
in_adc[11..0]
in_mem[11..0]
clock
rst
res[12..0]
mem_manager
mem_in[11..0]
clock
rst
in_mem[11..0]
mem_in[11..0]
INPUT
count
up counter
sload
data[9..0]
q[9..0]
clock
clk_en
cnt_en
aclr
data_a[9..0]
stop_up
notch@116
count
up counter
sload
data[9..0]
q[9..0]
clock
clk_en
cnt_en
aclr
sync_load
data_b[9..0]
stop_down
count_en
notch@90
notch@119
notch@118
big_mem
data[12..0]
wraddress[9..0]
wren
rdaddress[9..0]
rden
wrclock
rdclock
13 bits
1024 words
q[12..0]
dac_manager
out[12..0]
clock
rst
dac[12..0]
OUTPUT
dac[12..0]
aclr
wr_en
rd_en
wr_clock
rd_clock
notch@124
notch@54
notch@126
notch@125
clock
rst
notch@55
notch@56
NOT
TITLE : NOTCH
DESCRIPTION : top of the hierarchy
SHEET 1
OF 8
DRAWN BY
D ' ELIA in Feb. 2001

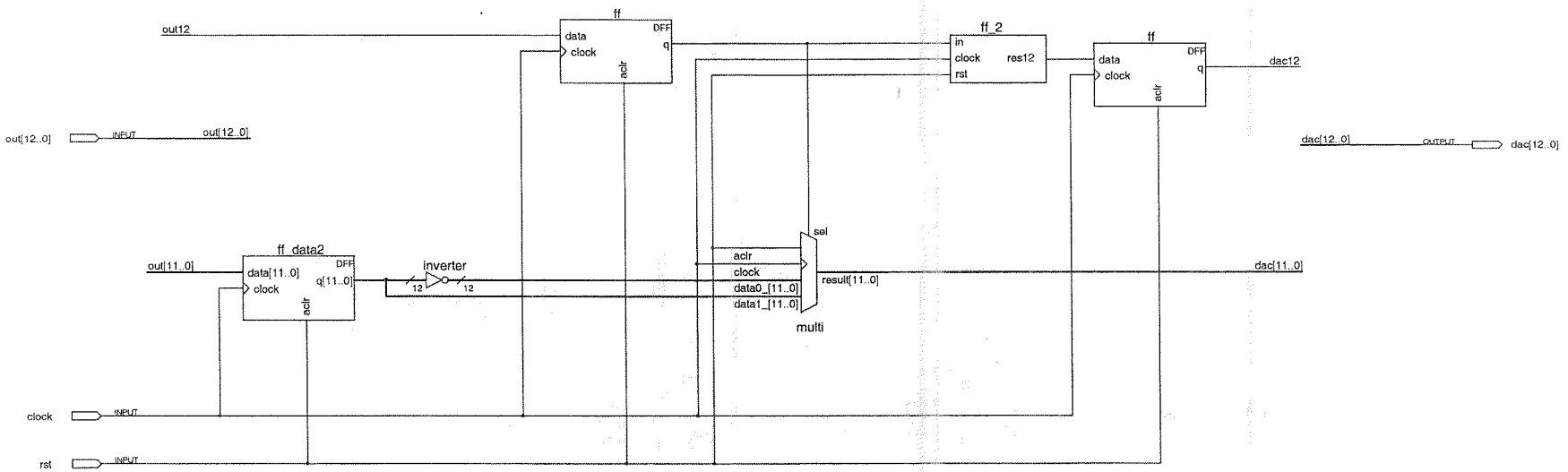


TITLE : ADC_MANAGER		
DESCRIPTION : circuit for ADC data managing		
SHEET 2 OF 8		DRAWN BY D ' ELIA in Feb. 2001

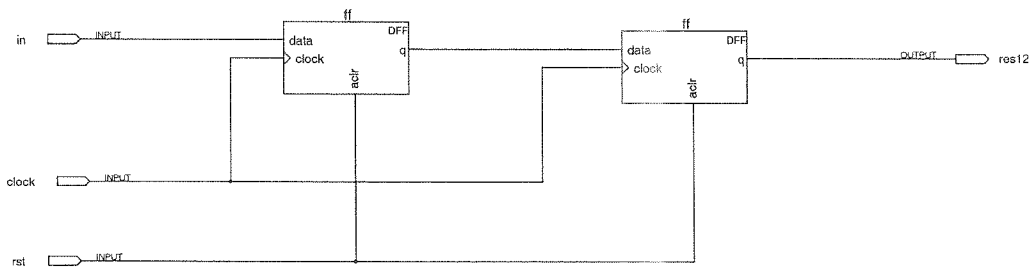


TITLE : MEM_MANAGER	
DESCRIPTION : circuit for MEMORY data managing	
SHEET 3 OF 8	DRAWN BY D ' ELIA in Feb. 2001

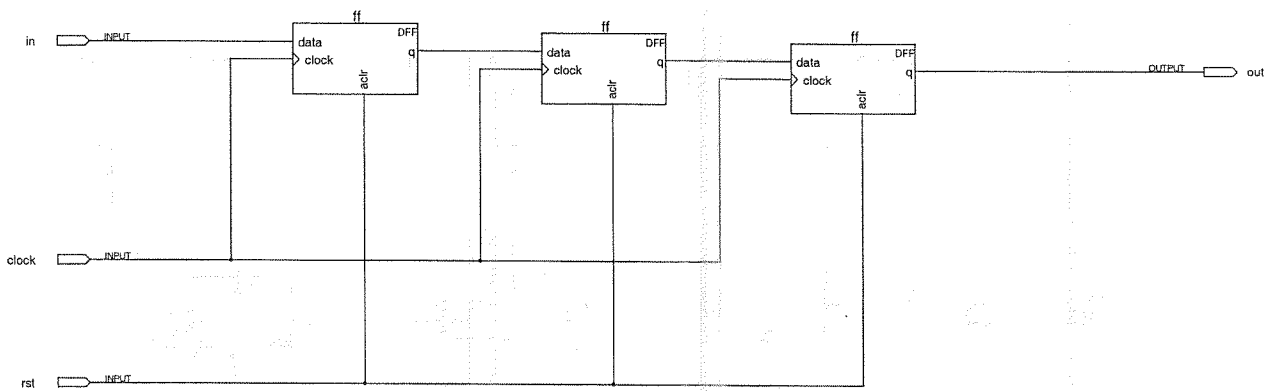

in_adc[11..0]
INPUT
in_adc[11..0]
in_mem[11..0]
INPUT
in_mem[11..0]
res[12..0]
OUTPUT
res[12..0]
ff_3
in
clock
out
rst
in_adc11
in_mem11
XNOR
notch@1C7_E12
aldo
add_sub
dataa[10..0]
clock
datab[10..0]
A+B/A-B
result[10..0]
aclr
cout
A
B
ff_data
data[10..0]
clock
aclr
DFF
q[10..0]
res[10..0]
ff_2
res12
comp
unsigned compare
in_adc[10..0]
in_mem[10..0]
dataa[10..0]
datab[10..0]
clock
aclr
aeb
agb
ff
data
q
res11
BAND2
OR2
AND2
NOT
sel
data0_[10..0]
data1_[10..0]
mux
OR3
ff_4
res12
AND3
clock
rst
INPUT
TITLE : A_ALU
DESCRIPTION : circuit for the sign summation
SHEET 4
OF 8
DRAWN BY
D ' ELIA in Feb. 2001



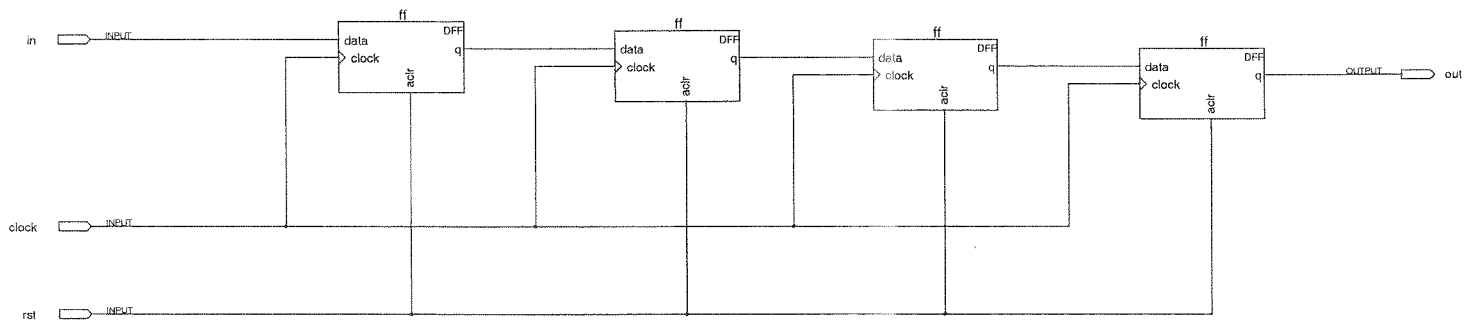
TITLE : DAC_MANAGER		
DESCRIPTION : circuit for data managing to DAC		
SHEET 5		DRAWN BY
OF 8		D ' ELIA in Feb. 2001



TITLE : FF_2	
DESCRIPTION : circuit for providing a delay of 2 clock times	
SHEET 6 OF 8	DRAWN BY D ' ELIA in Feb. 2001



TITLE : FF_3	
DESCRIPTION : delay block	
SHEET 7 OF 8	DRAWN BY D ' ELIA in Feb. 2001



TITLE : FF_4	
DESCRIPTION : delay block	
SHEET 8 OF 8	DRAWN BY D ' ELIA in Feb. 2001

Acknowledgements.

I want to thank Prof. Vittorio Vaccaro, my boss at University " Federico II " in Naples for the availability and for having given to me the possibility of going to GSI for doing this work.
Also I want to thank Dr. Maria Rosaria Masullo for her help in writing this report and for her support during my stay in Germany.
Many thanks to Prof. Sergio Cavalieri for his precious help in matter of digital filters and signal analysis.
I have to thank Prof. Ingo Hofmann, my boss at GSI in Darmstadt for his kindness and for having introduced me in the GSI environment.
Many thanks to Dr. Udo Blell, my Supervisor at GSI, for the patience and time spent for me, and for his support during my work.
Also I have to thank very much the future Dr. Roman Rojko who gave me many advises and ideas for my work, and for having subtracted time to his PhD thesis for me.
Also I want to thank affectionately Dr. Giovanni Rumolo for his competence in giving me advises for my theoretical doubt in my work, and, not the last, for his sincere friendship.
Many thanks to Dr. Ralph Baer for the friendly comprehension and for his support during all the time in GSI and in Germany. With him I want to thanks my office colleagues Dipl.-Phys. Yong Liu, and Dr. Bjoern Eberlein.
I have to thank very much all the GSI people, so many than I can not mention all of the them in the acknowledgements, for the kindness and friendship; if I could work so well there it was for them.
Last but not least, I thank my friends in Naples for their moral support during all my staying in Germany.

Bibliography.

[1] R. Rojko, " New concepts for Transverse Beam Stability in High Current Heavy Ion Synchrotron ", Fachbereich für Elektrotechnik und Informationstechnik der Technische Universität Darmstadt, Darmstadt 2000.

[2] E. Wilson, " Non linearities and resonances ", CERN accelerator school (CAS), Gif-sur-Yvette, Paris, France, 3-14 September 1984.

[3] J. Rossbach, P. Schmüsser, " Basic course on accelerators optics ", CERN accelerator school (CAS), fifth general accelerator physics course, 7-18 September 1992, university of Jyväskylä, Finland.

[4] R. Rojko, U. Blell, " A transverse feedback system (TFS) for the Heavy Ion Synchrotron (SIS) at the GSI " .

[5] D.A. Goldberg, G.R. Lambertson, " Dynamic Devices: A Primer On Pickups and Kickers ", LBL-31664, ESG-160, Lawrence Berkely Laboratory, University of California, Berkeley, CA 94720.

[6] J.L. Gonzalez, " Correction des instabilités transversales des faisceaux de protons de haute densité, dans l' accelerateurs PS ", CERN/PS 93-17 (BD).

[7] C. F. Christiansen, G. Gelato, " A Frequency-Selective Self-Balancing Bridge – An application to closed-orbit signal suppression ", CERN/PS/BR 80-6, 25 March 1980, Geneve, Switzerland.

[8] M. Stampfer, " Development of an active feedback system to damp transverse coherent instabilities in the Heidelberg Test Storage Ring ", diplomarbeit Max-Planck-Institüt für kernphysic.

[9] D. Boussard, " Signal analysis ", Springer-Verlag " Frontiers of particle beams; Observation, Diagnosis and correction ", Proceedings, Anacapri, Isola di Capri, Italy 1988.

[10] A.V. Oppenheim, R. W. Schaffer, " Elaborazione numerica dei segnali ", Franco Angeli s.r.l., Milano, Italia.

[11] F. Cevenini, appunti del corso di " Laboratorio di Elettronica ", A.A. 1999-2000, corso di laurea in fisica, università " Federico II ", Napoli, Italia.

[12] G. Biondo, E. Sacchi, " Manuale di elettronica e telecomunicazioni ", quarta edizione, Hoepli editore, Milano, Italia.

[13] Datasheets, " 12-bit, 80 MSPS/105 MSPS A/D Converter ", Analog Devices Inc. 2000, One Technology Way, Norwood, Ma, U.S.A.

[14] Datasheets, " 14-bit, 125 MSPS High Performance TxDac D/A Converter ", Analog Devices Inc. 2000, One Technology Way, Norwood, Ma, U.S.A.