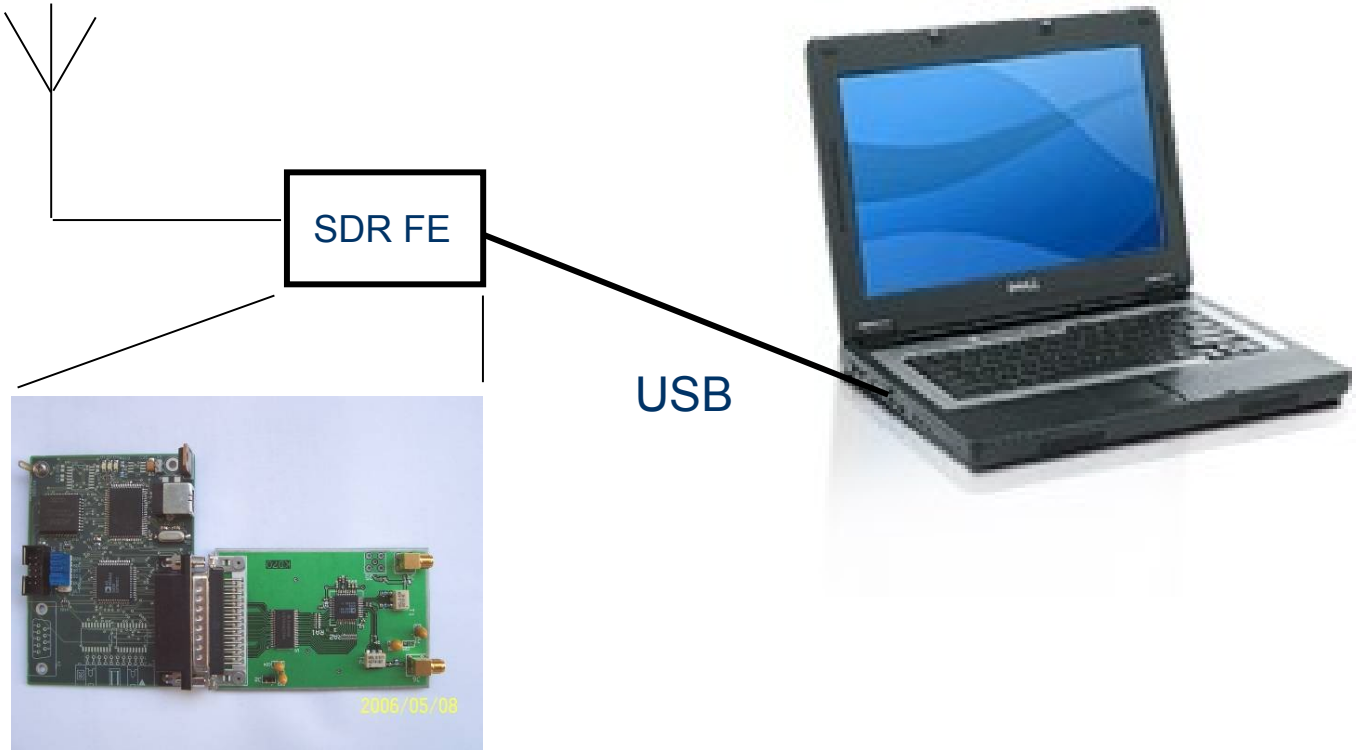


SDR*ham*, A Software Defined Radio Project
Nordic VHF/UHF/SHF Meeting 2007

- Project started in 2004
- Project goals
  - Develop an SDR FE (Front End) hardware
  - Develop SDR PC software
  - Learn SDR technology
  - Learn Digital Signal Processing
  - Use SDR in amateur radio applications where high performance is needed
    - Weak signal reception in crowded HF bands
    - EME (Earth-Moon-Earth) communication in the VHF band
  - To have a plattform, SDR FE and PC, for development of analog and digital modes

Status May 2007

- First SDR FE running
- A Windows/PC test application with SSB/CW demodulator is developed
- Some initial performance testing is done

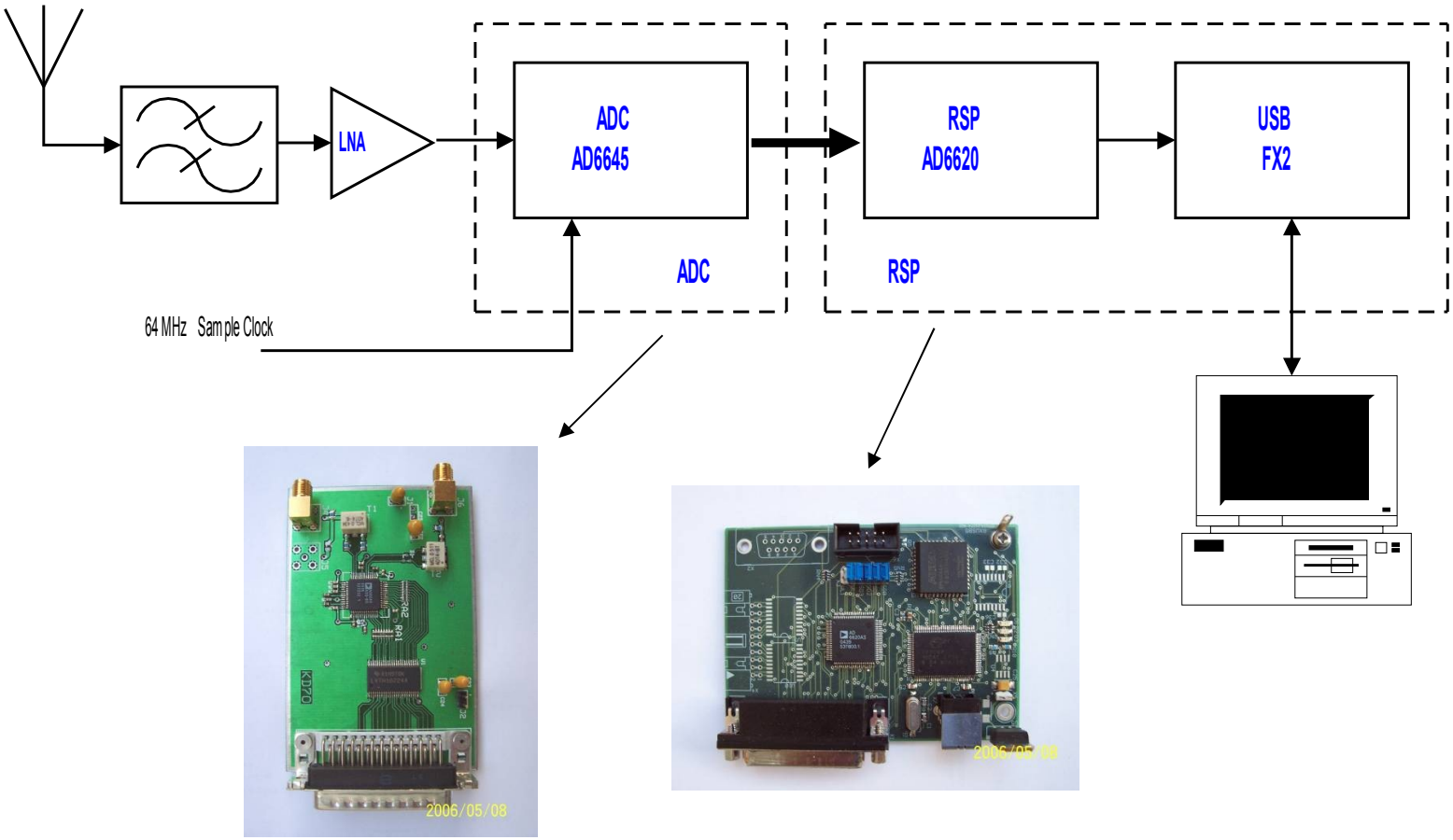
Participants

- LA9CY, Kjell Syverud
- LA7OU, Stein Erik Ellevseth
- LA8TO, Odd Arild Olsen
- LB8X, Tom Twist
- LA7BO, Halvor Liland

Agenda

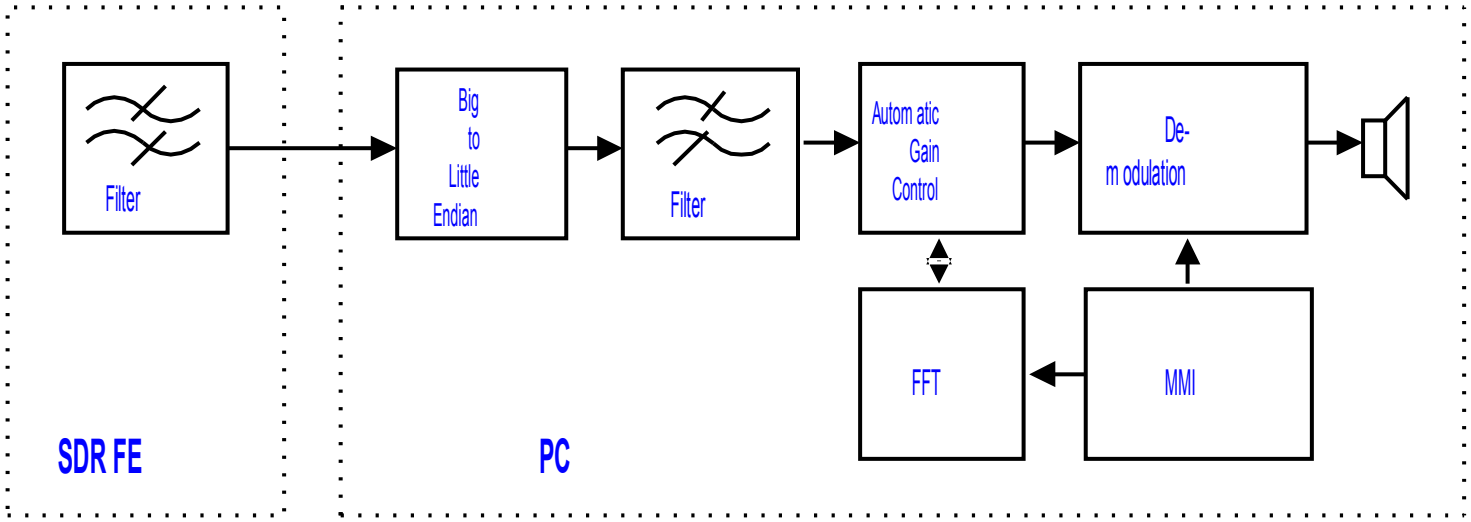
- An introduction to SDR*ham*
- Applied technology and theory
  - Sampling
  - Filter design, time and frequency domain
  - USB, sample streaming
  - Signal Processing on PC
    - MMX and SSE instruction sets in Pentium CPU
  - Demodulation techniques
- Tools
- Receiver design

The SDR Architecture, HW



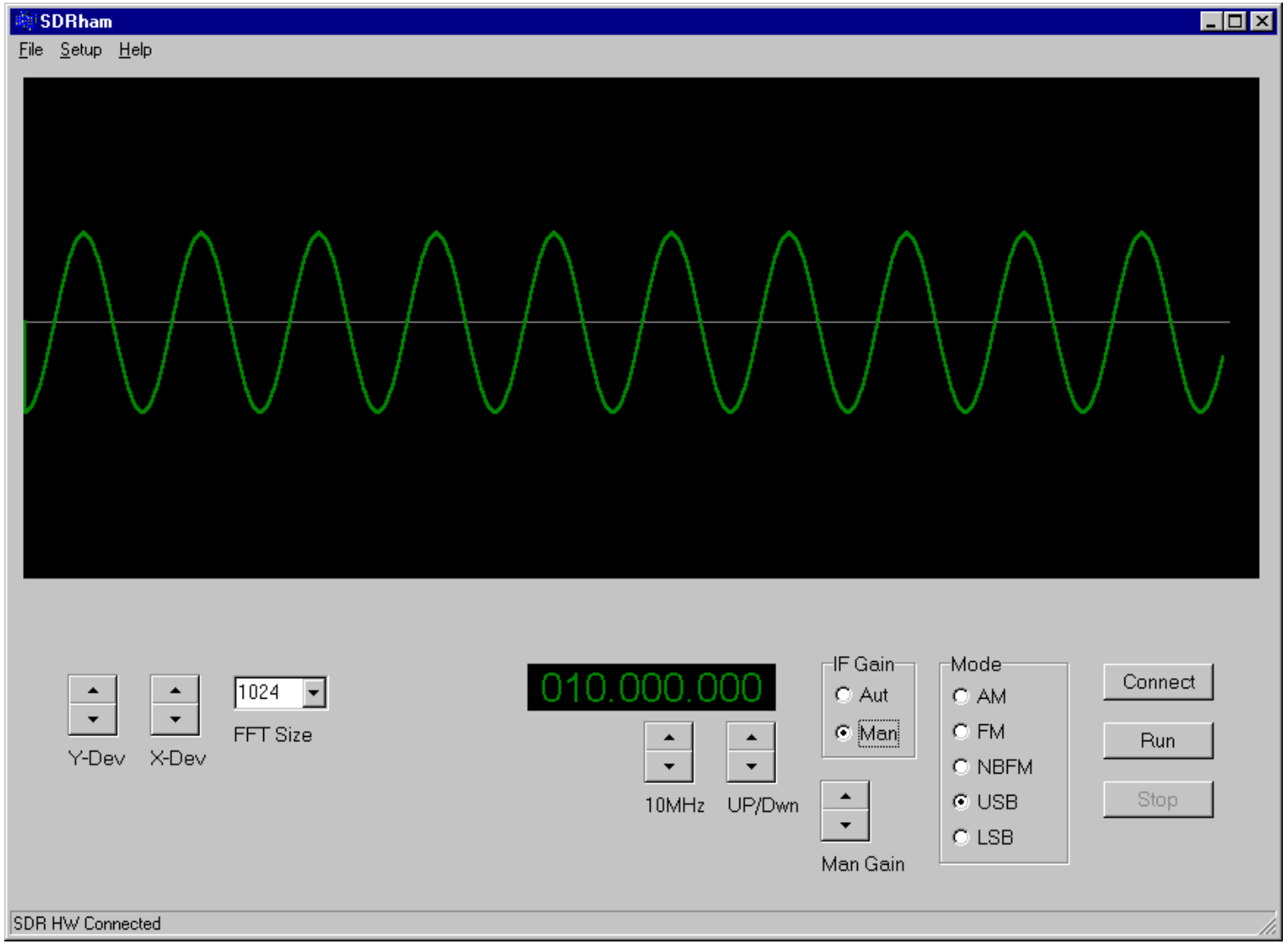
SDR*ham*

The SDR Architecture, Test SW



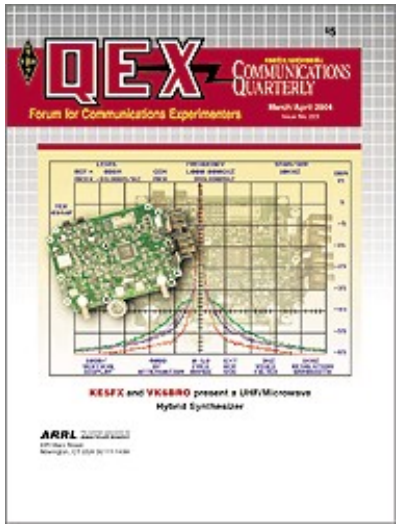
SDR*ham*

SDR Test Application User Interface



SDR*ham*

Ideas of Project



A High-Performance Digital-Transceiver Design, Part 1

Data-converter technology has made tremendous strides in the past several years. Let's take a look at how we can achieve high performance in an almost-all-digital radio design.

By James Scarlett, KD7O

Linrad: *New Possibilities for the Communications Experimenter, Part 1*

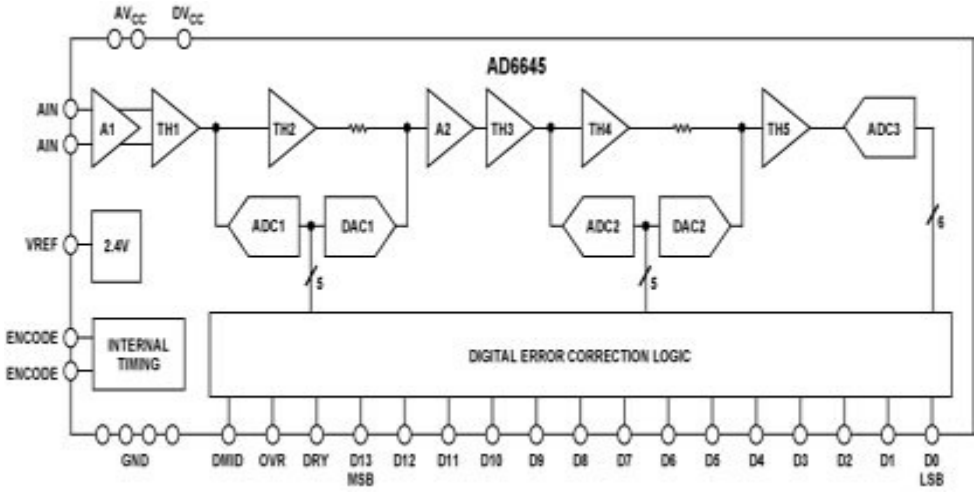
Discussion opens with analog versus digital RF-input techniques and attendant performance considerations.

By Leif Åsbrink, SM5BSZ

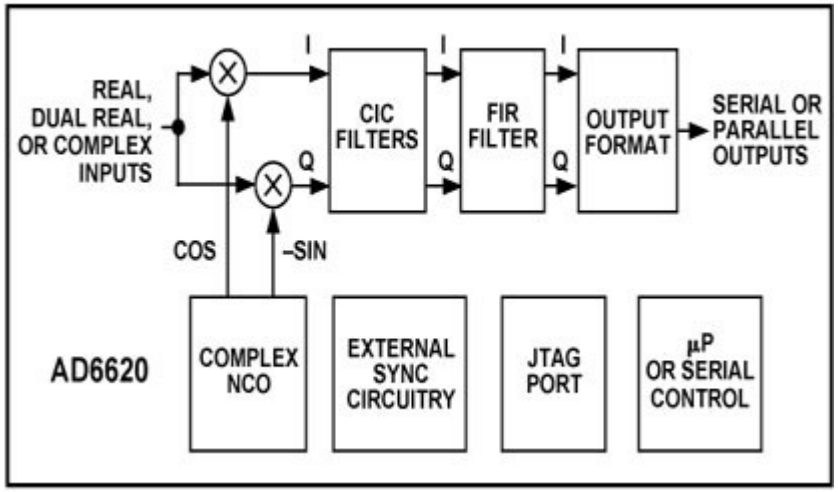
SDR*ham*

The Analog Devices Integrated Circuits

AD6645
14-Bit, 80 MSPS
AD Converter



AD6620
Digital Receive Signal
Processor



SDR*ham*

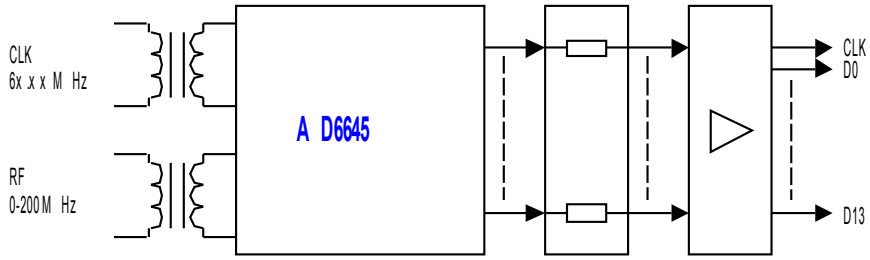
Technology

- Direct RF sampling
  - to 200 MHz
- USB
- Digital Signal Processing on PC
  - MMX and SSE instruction set

Tools

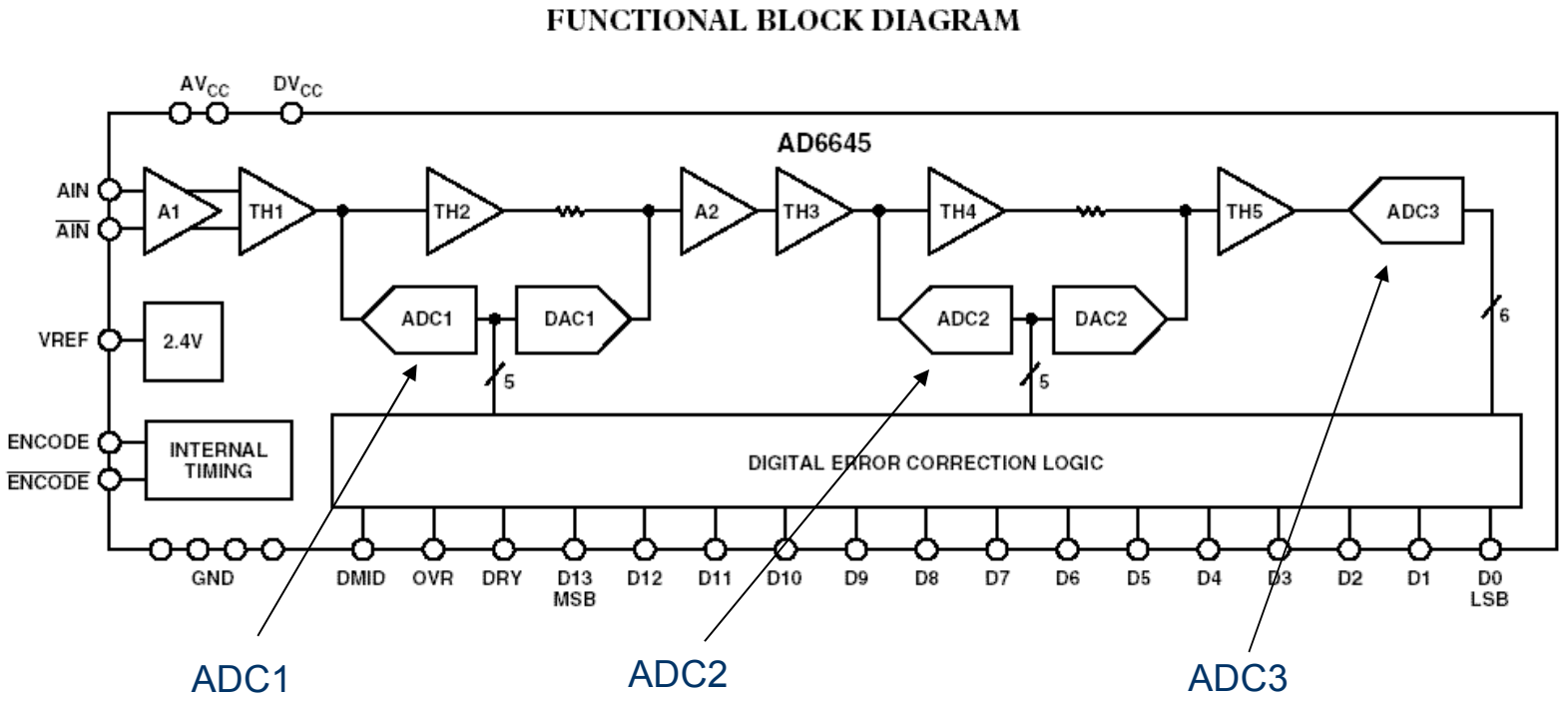
- Borland C++ Builder6
  - PC SDR SW
- Keil µVision2
  - EZ USB FX2 8051 code
- Cypress EZ-USB Control Panel
  - USB communication
- Altera Quartus II
  - EPLD
- MathCAD
  - Filter design, analyses and simulation
- Ansoft Harmonica
  - Analog Simulation

Design of KD7O



SDR*ham*

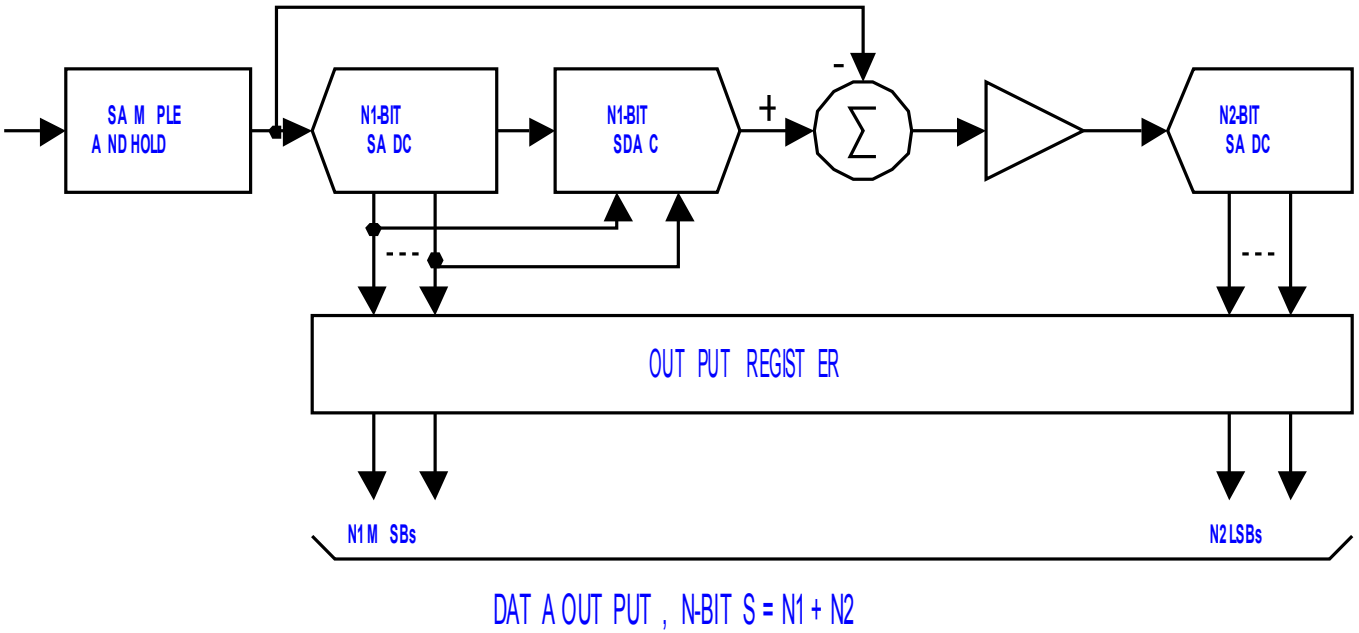
Analog to Digital Converter, AD6645 14-bit 80 MSPS



The AD6645 employs a three stage subrange architecture

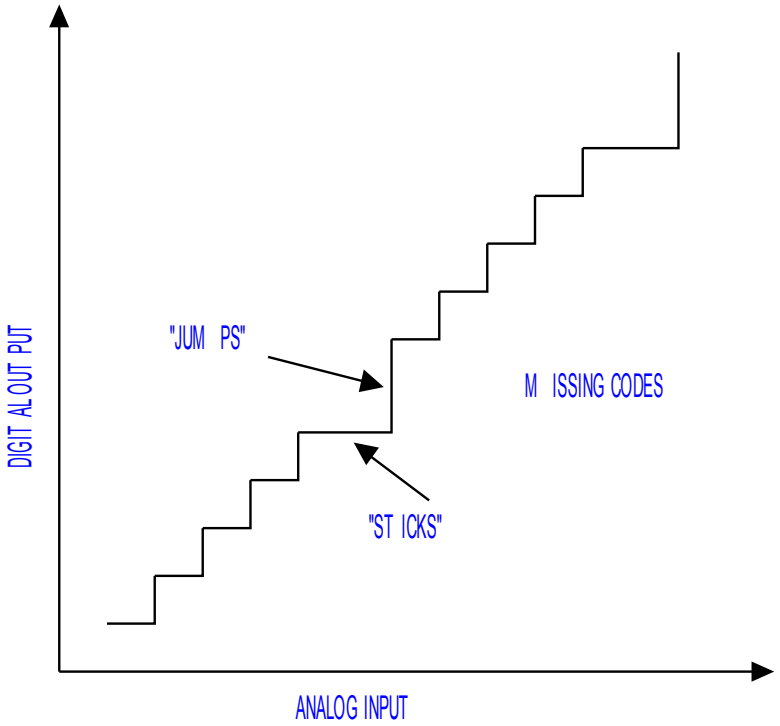
80 MSPS Guaranteed Sample Rate
SNR = 75 dB, f_{IN} 15 MHz @ 80 MSPS
SNR = 72 dB, f_{IN} 200 MHz @ 80 MSPS
SFDR = 89 dBc, f_{IN} 70 MHz @ 80 MSPS

SDR*ham*



N-bit Two-Stage Subranging ADC

Missing Codes



Missing Codes Due to
MSB SADC Nonlinearity
or Interstage Misalignment

SDR*ham*

SNR, SFDR and Process Gain

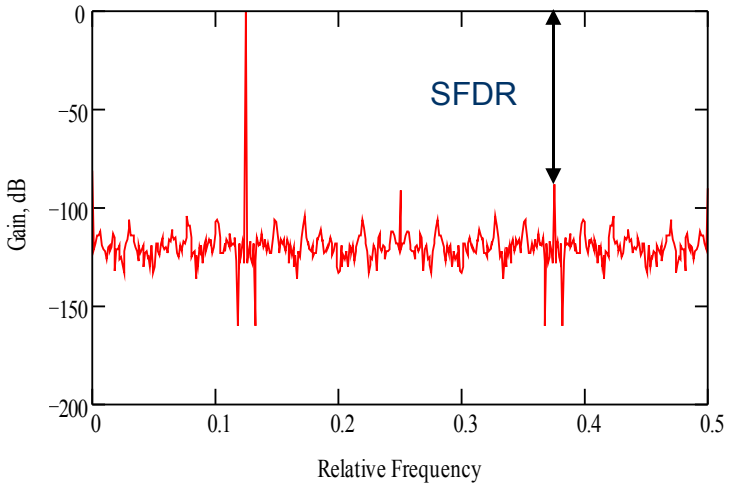
$$SNR = ٦.٠٢N + ١.٧٦$$

(Measured over the Nyquist Bandwidth: DC to $f_S/2$)

Calculated SNR for an 14-bit ADC

$$SNR = 6.02 \cdot 14 + 1.76 = 86 \text{ dB}$$

$$SNR = 6.02N + 1.76 + \underbrace{10 \log \frac{f_S}{2 \cdot BW}}_{\text{Process Gain}}$$



f_s = 64 MHz, BW = 3.2 kHz

$$SNR = 6.02 \cdot 14 + 1.76 + 10 \log \frac{64 \cdot 10^6}{2 \cdot 3.2 \cdot 10^3} = 126 \text{ dB}$$

SDR*ham*

Some “Magic” of Process Gain

The AD6645 Input Voltage Range is 2.2 $V_{p\text{-}p}$.The voltage for LSB (Least Significant Bit) is then

$$V_{LSB} = \frac{2.2\text{V}}{2^{14}} = 134 \cdot 10^{-6} \quad (134\ \text{uV})$$

The power P in a signal of -110 dBm is found from

$$-110 = 10\log\left(\frac{P}{10^{-3}}\right) \Rightarrow$$

$$P = 10^{-3} \cdot 10^{\frac{-110}{10}} = 1 \cdot 10^{-14}$$

The voltage is then

$$P = \frac{U^2}{R} \Rightarrow U = \sqrt{P \cdot R} = \sqrt{1 \cdot 10^{-14} \cdot 50} = 0.7 \cdot 10^{-6}$$

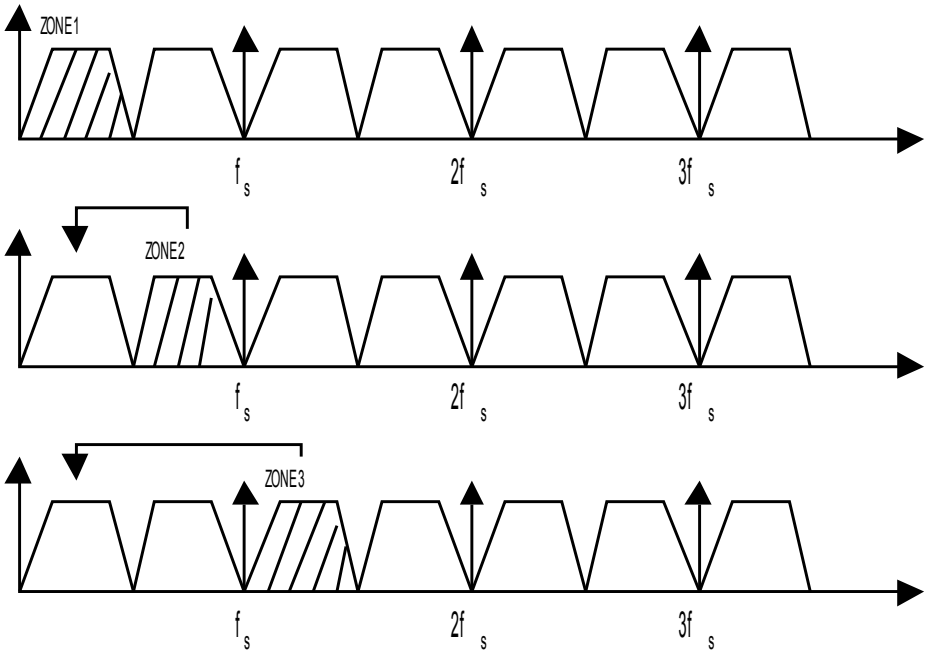
= 1 uV_{PEAK}

How can the ADC detect voltage levels of one hundred (-40 dB) of its LSB?

Answer: The signal voltage is added to the ADC internal noise voltage, which is more than many LSBs. With a process gain of 40 dB the the signal will appear again.

SDR*ham*

Sampling, Nyquist Zones



Undersampling and Frequency Translation between Nyquist Zones

The process of sampling a signal outside the first Nyquist zone is often referred to as *undersampling* or *harmonic sampling*

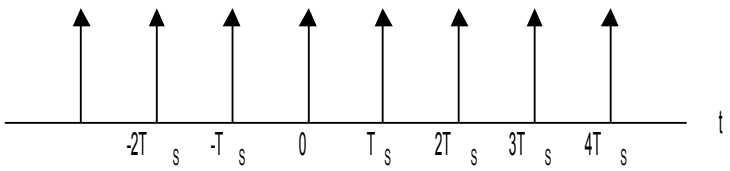
A restate of the Nyquist criteria:

A signal must be sampled at a rate equal to or greater than twice its bandwidth in order to preserve all the signal information

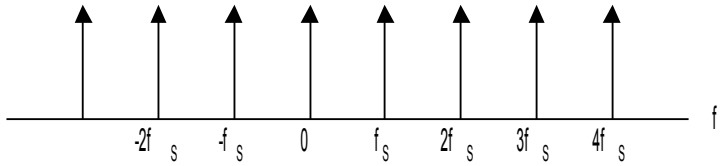
SDR*ham*

Mathematics of Undersampling

$$x(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT_S)$$



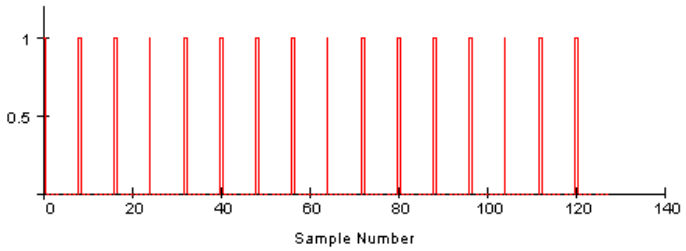
$$X(f) = \frac{1}{T_S} \sum_{n=-\infty}^{\infty} \delta(f - nf_S)$$



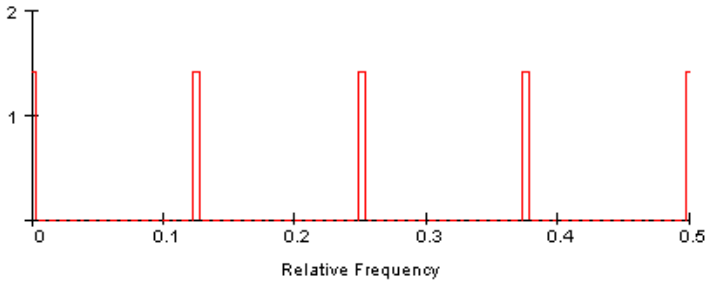
A MathCad simulation proves theory

$N := 128$ $\quad n := 0..N-1$ $\quad i := 0,8..N-1$

$s_n := 0$ $\quad s_i := 1$



$S := \mathrm{fft}(s)$ $\quad k := 0..\frac{N}{2}$



SDR*ham*

Noise Factor (NF) of the AD6645

NF is an important parameter in receivers. To calculate a receiver NF one need to know the NF of each stage. NF is normally not specified by ADC manufacturers.

$$V_{FSRMS} := \frac{\frac{V_{PP}}{2}}{\sqrt{2}}$$

$$V_{NOISERMS} := V_{FSRMS} \cdot 10^{-\frac{SNR}{20}}$$

$$F := \frac{V_{NOISERMS}}{\sqrt{k \cdot T \cdot R \cdot B}}$$

$$NF := 20 \cdot \log(F) \qquad NF = 35$$

AD6645 calculated noise figure

$$F_{TRAFO} := 0$$

Noisfree transformer

$$G := 2$$

Voltage gain of the transformer

$$F_{WT} := F_{TRAFO} + \frac{F - 1}{G}$$

$$NF_{WT} := 20 \cdot \log\left(F_{WT}\right) \qquad NF_{WT} = 29$$

SDR*ham*

Maximum Input Signal to the AD6645

A 1:4 impedance transformer makes the 50 ohm antenna input to 200 ohm at the AD6645 input.

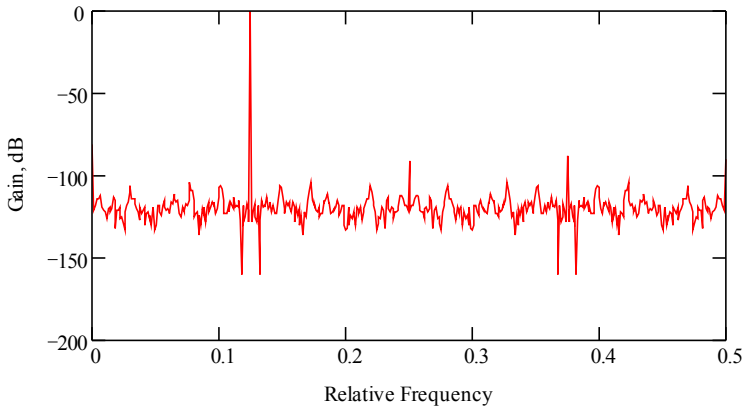
The maximum input voltage of the AD6645 is 2.2 V_{P-P}, that is 0.78 V_{RMS}. The power is then

$$P = 10\log\left(\frac{\frac{U^2}{R}}{10^{-3}}\right) = 10\log\left(\frac{\frac{0.78^2}{200}}{10^{-3}}\right) = 4.8 \text{ dBm}$$

SDR*ham*

Effect of Dither (MathCAD Simulation)

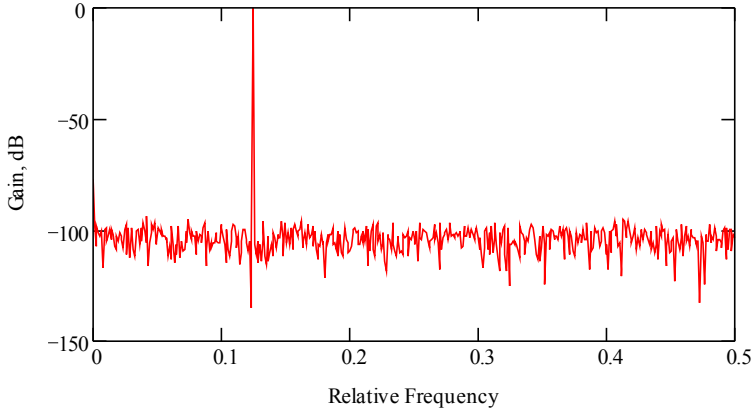
Without dither



$$SNR := 20 \cdot \log\left[\frac{|\max(S)|}{\sqrt{\sum_i \left(|S_i|\right)^2 - \left(|\max(S)|\right)^2}}\right]$$

$SNR = 79$

With dither (normal distributed noise)

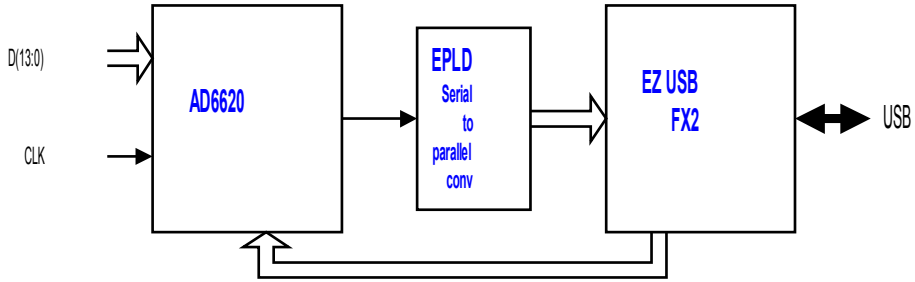


$$SNR := 20 \cdot \log\left[\frac{|\max(S)|}{\sqrt{\sum_i \left(|S_i|\right)^2 - \left(|\max(S)|\right)^2}}\right]$$

$SNR = 73$

SDR*ham*

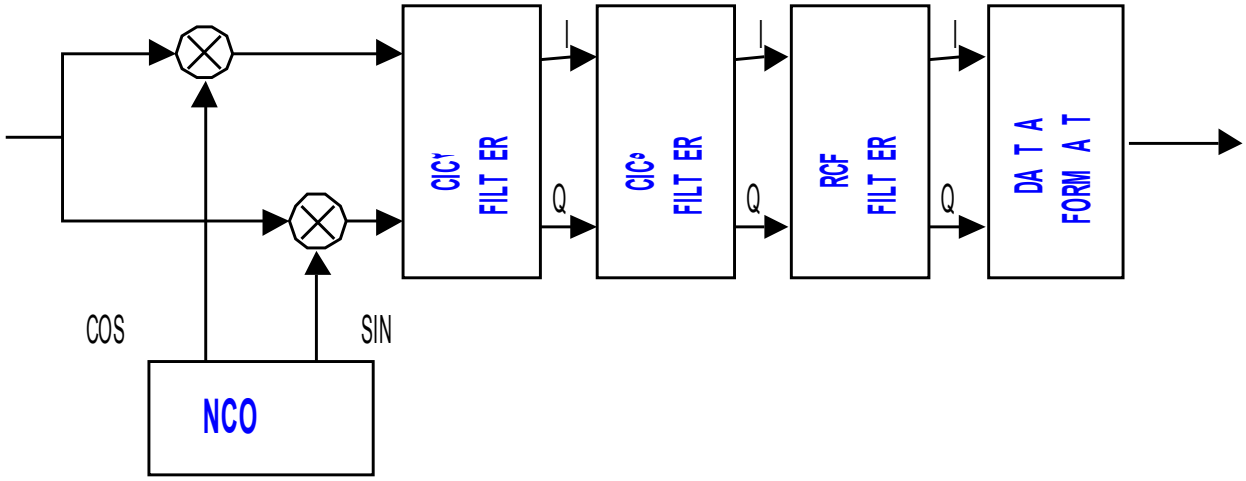
RSP (Receive Signal processor) Module



SDR*ham*

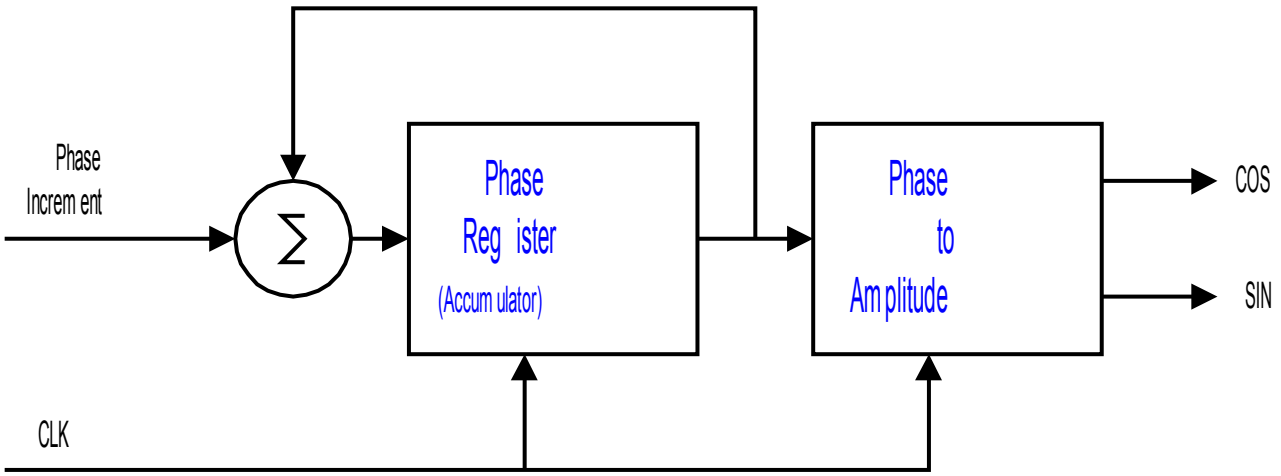
Digital Signal Receive Processor, AD6620

The AD6620 decimating receiver is designed to bridge the gap between high-speed ADCs and general purpose DSPs. It has four cascaded signal processing elements: a frequency translator, two fixed coefficient decimating filters and a programmable coefficient decimating filter.



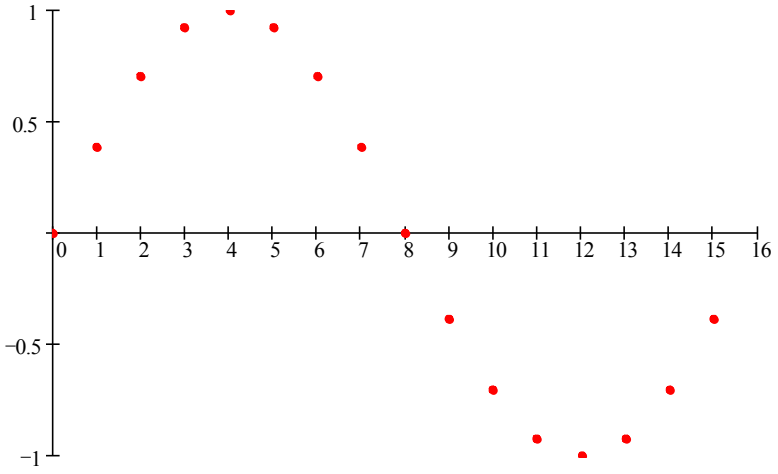
SDR*ham*

Numeric Controlled Oscillator, NCO - HW



SDR*ham*

Numeric Controlled Oscillator, NCO



Clk = 16 MHz
Phase increment = 1 -> f = 1 MHz
Phase increment = 2 -> f = 2 MHz

Frequency is phase change pr time interval. A phase change of 2л (360°) in 1 second is equal to a frequency of 1 Hz

$$f = \frac{d\varphi}{dt}$$

NCO_FREQ is interpreted as a 32-bit unsigned integer. NCO_FREQ is calculated by

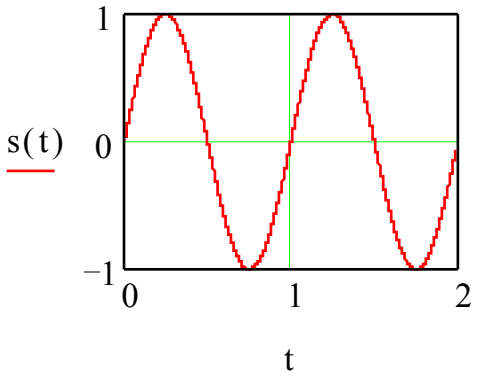
$$NCO_{FREQ} = 2^{32} \times \mathrm{mod}\left(\frac{f_{CH}}{f_{SAMP}}, 1\right)$$

$$f_{\min} = \frac{64 \cdot 10^6}{2^{32}} = 0.015$$

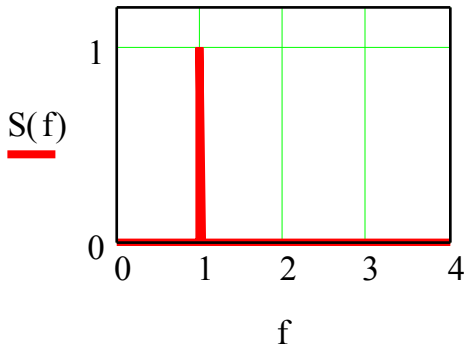
SDR*ham*

Time and Frequency Domain

Time Domain



Frequency Domain



Fourier transform og Fourier Series transform a signal between the time and frequency domain

$$s(t) = \int_{-\infty}^{\infty} S(f) \cdot e^{j2\pi ft} df$$

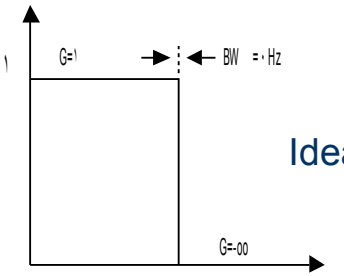
$$S(f) = \int_{-\infty}^{\infty} s(t) \cdot e^{-j2\pi ft} dt$$

$$s(t) = \sum_{n=-\infty}^{\infty} c_n e^{j2 \cdot \pi \cdot nft}$$

$$c_n = \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} s(t) \cdot e^{-j2 \cdot \pi \cdot nft} dt$$

SDR*ham*

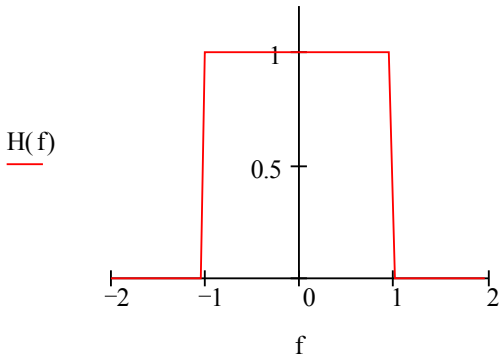
The ideal filter in the frequency and time domain

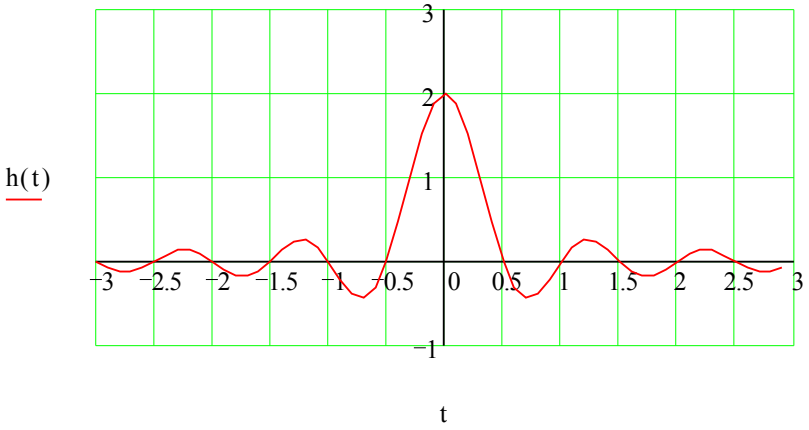


Ideal filter

$$h(t) := \int_{-2}^{2} H(f)\cdot e^{j\cdot 2\cdot \pi \cdot f\cdot t}\, df$$

$$H(f) := \Phi(f-(-1)) - \Phi(f-1)$$





Ideal Low Pass Filter
time domain response

$$y = \frac{\sin(x)}{x}$$

SDR*ham*

Filter, frequency domain vs time domain

Multiplication in the frequency domain is convolution in the time domain

$$y(t)=\int_{-\infty}^{\infty} h(\tau)s(t-\tau)d\tau$$ The Convolution Integral

$$Y(f)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h(\tau)s(t-\tau)d\tau\cdot e^{-j2\pi\cdot ft}dt$$

Changing variable with u = t - *t* and replace the integrals gives

$$Y(f)=\int_{-\infty}^{\infty} h(\tau)e^{-j2\pi\cdot f\tau}d\tau\cdot\int_{-\infty}^{\infty} s(u)e^{-j2\pi\cdot fu}du=H(f)S(f)$$

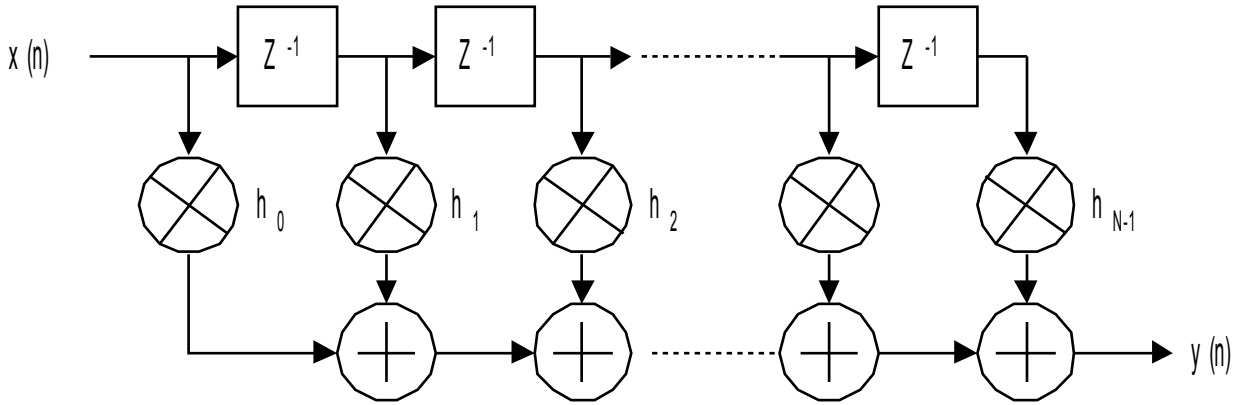
SDR*ham*

FIR (Finite Impulse Response) Filter

A finite impulse response (FIR) filter is a discrete linear time-invariant system whose output is based on the weighted summation of a finite number of past inputs.

In the discrete domain (a sampled digital data system) the convolution integral changes to a summation

$$y(n) = \sum_{k=0}^{N-1} h_k \cdot s(n-k)$$

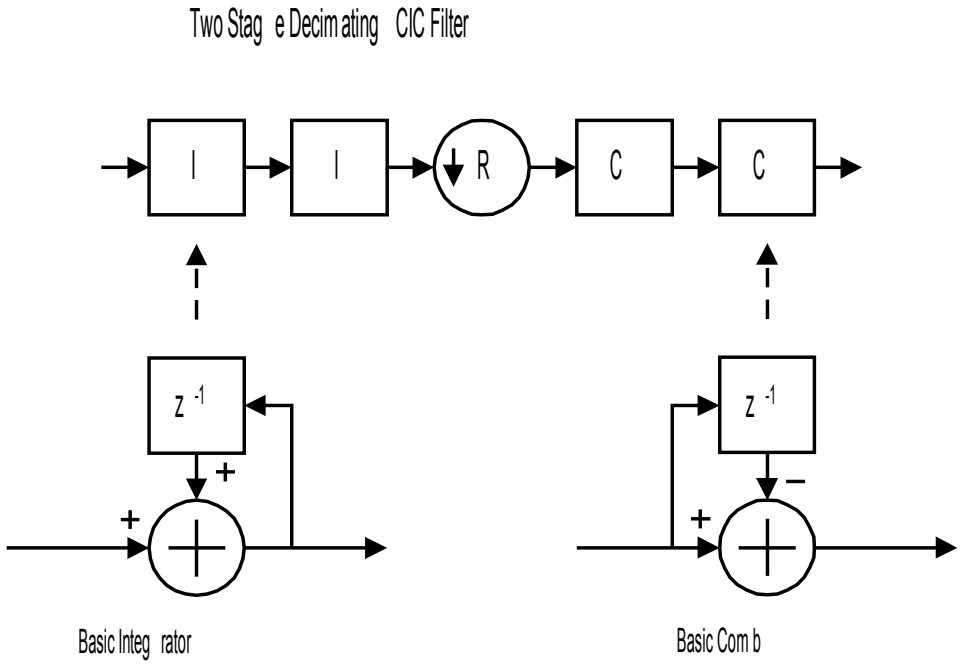


SDR*ham*

Cascaded Integrator Comb Filter, CIC

The CIC filter is a fixed-coefficient, multiplierless, decimating filter.
The characteristics
are defined by the decimation rate.

$$H(z)=\left(\frac{1-z^{-RM}}{1-z^{-1}}\right)^{N}$$



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

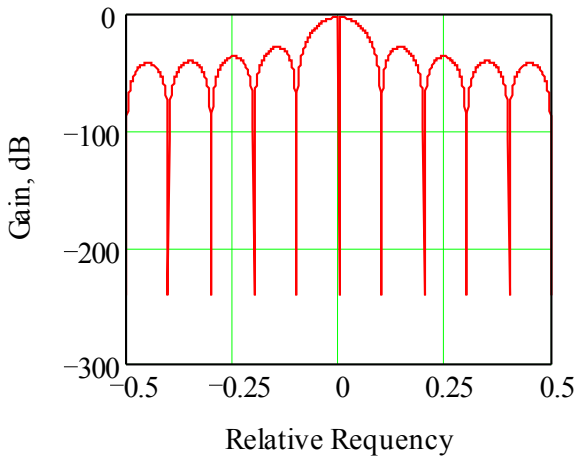
$$y[n]=x[n]-x[n-RM]$$

SDR*ham*

CIC2 Frequency Response

$$H(f) = \frac{1}{2^{S_{CIC2}+2}} \times \left(\frac{\sin(\pi \frac{M_{CIC2} ¿ f}{f_{SAMP}})}{\sin(\pi \frac{f}{f_{SAMP}})} \right)^2$$

$$DR(BW, M_{CIC2}) := 20 \cdot \log \left[\frac{\sum_{k=1}^{\text{floor}\left(\frac{M_{CIC2}}{2}\right)} \int_{-BW + \frac{k \cdot f_{SAMP}}{M_{CIC2}}}^{BW + \frac{k \cdot f_{SAMP}}{M_{CIC2}}} H(f, M_{CIC2})\, df}{\int_{0}^{BW} H(f, M_{CIC2})\, df} \right]$$

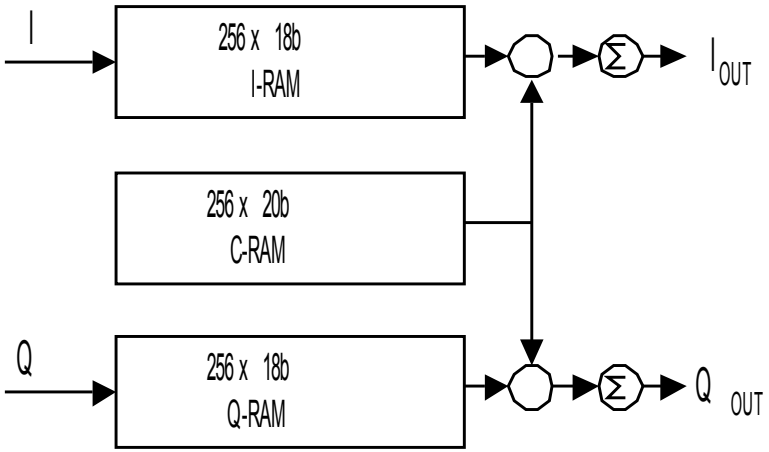


f_{SAMP} = 65 MHz, M_{CIC2} = 10

BW = 4 kHz -> DR = 127 dB
BW = 100 kHz -> DR = 71 dB

SDR*ham*

AD6620 RCF (RAM Coefficient Filter)



This filter is an FIR (Finite Impulse Response) filter

The length of the filter kernel limits the transition bandwidth

$$N \approx \frac{4}{BW}$$

Where BW is a fraction of sampling frequency

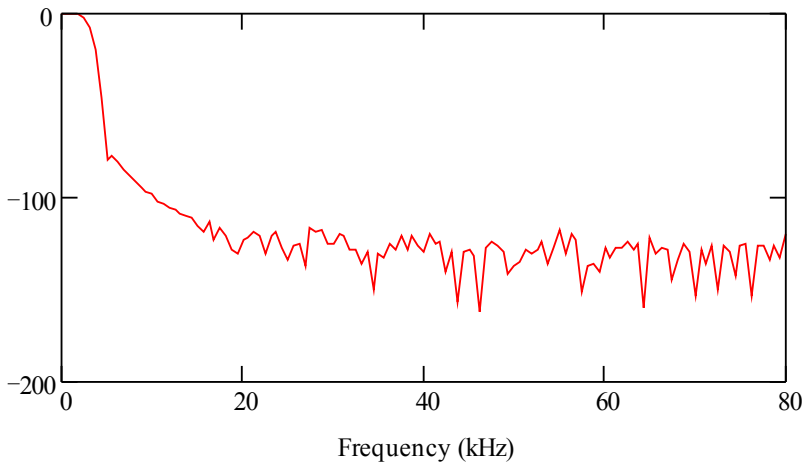
SDR*ham*

AD6620 Setup File, *ad6620.imp*

Low Pass Filter with 3 kHz cut off frequency

$$\text{Blackman}(i) := 0.42 - 0.5\cdot\cos\left(\frac{2\cdot\pi\cdot i}{M}\right) + 0.08\cdot\cos\left(\frac{4\cdot\pi\cdot i}{M}\right)$$

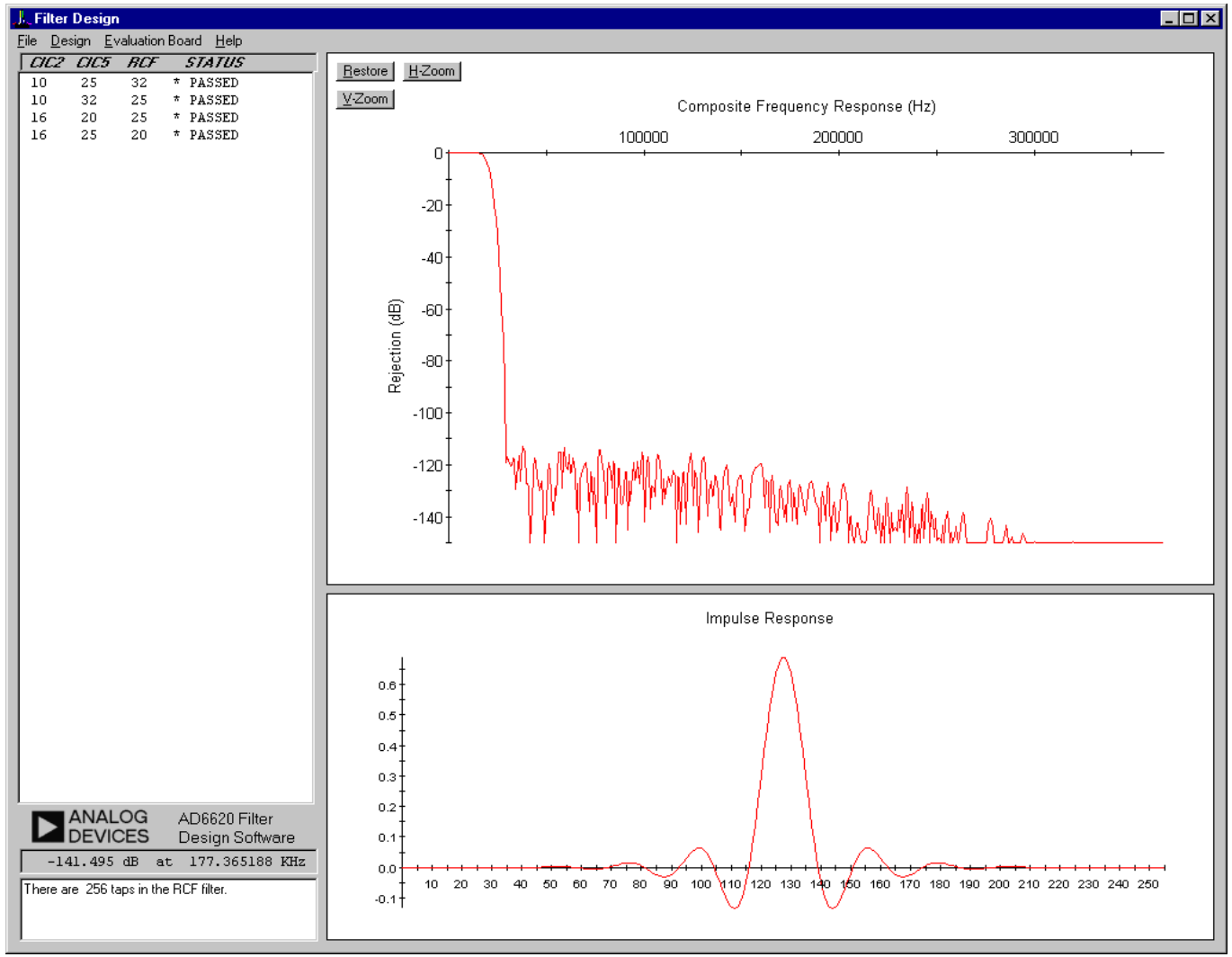
$$h_i := \text{if}\left[i = \frac{M}{2}, 2\cdot\pi\cdot f_c, \frac{\sin\left[2\cdot\pi\cdot f_c\cdot\left(i - \frac{M}{2}\right)\right]}{i - \frac{M}{2}}\cdot\text{Blackman}(i)\right]$$



16	M_{CIC2}
25	M_{CIC5}
20	M_{RCF}
64000000	f_{SAMP}
1	CHIP MODE
1	CLK MULT
0	
1	
4	
10	
19	FILTER
30	COEFF
44	
61	
80	
100	
121	
141	
158	
172	
181	
183	
175	

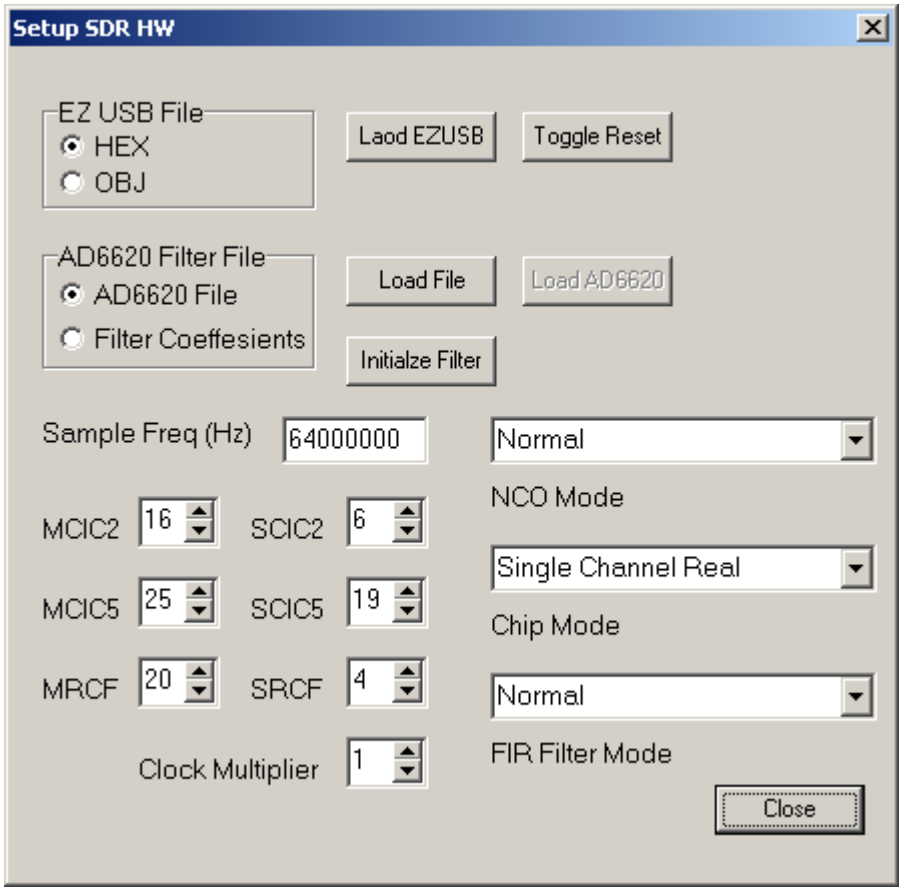
SDR*ham*

AD6620 Filter Design Software



SDR*ham*

AD6620 Setup Dialog Box

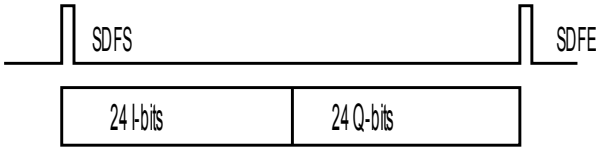


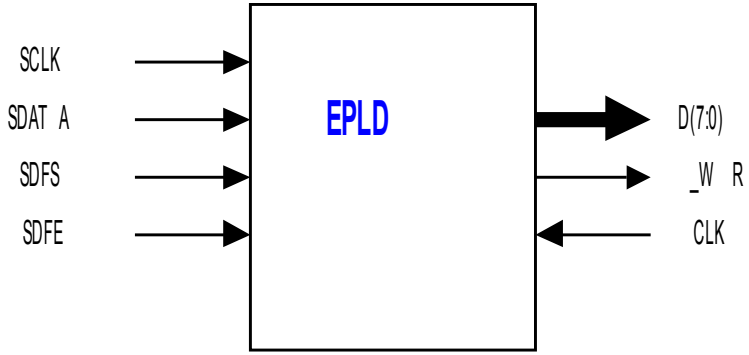
The dialog box reads parameters from a file or let the user change them

SDR*ham*

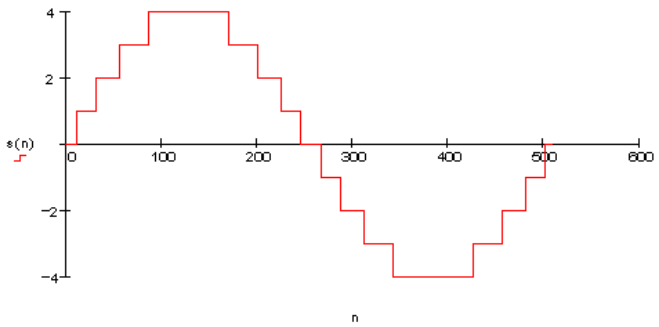
EPLD, Serial to Parallel Converter

In the SDR*ham* the serial output format of the AD6620 is used to get 24-bit resolution. An EPLD, Altera EPM3064, is used to convert the serial bit stream to the 8-bit parallel FIFO interface of the EZ USB FX2

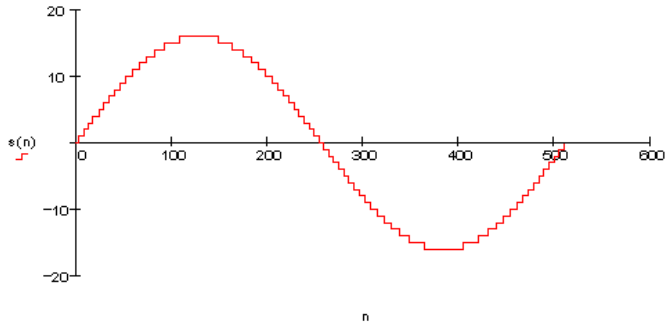




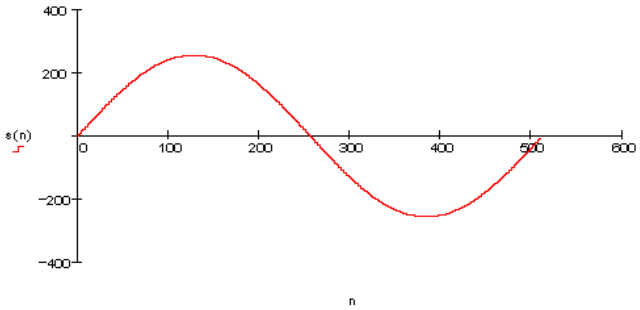
Why 24-bit when the Input is 14-bit ?



A sine wave sampled with 2-bit

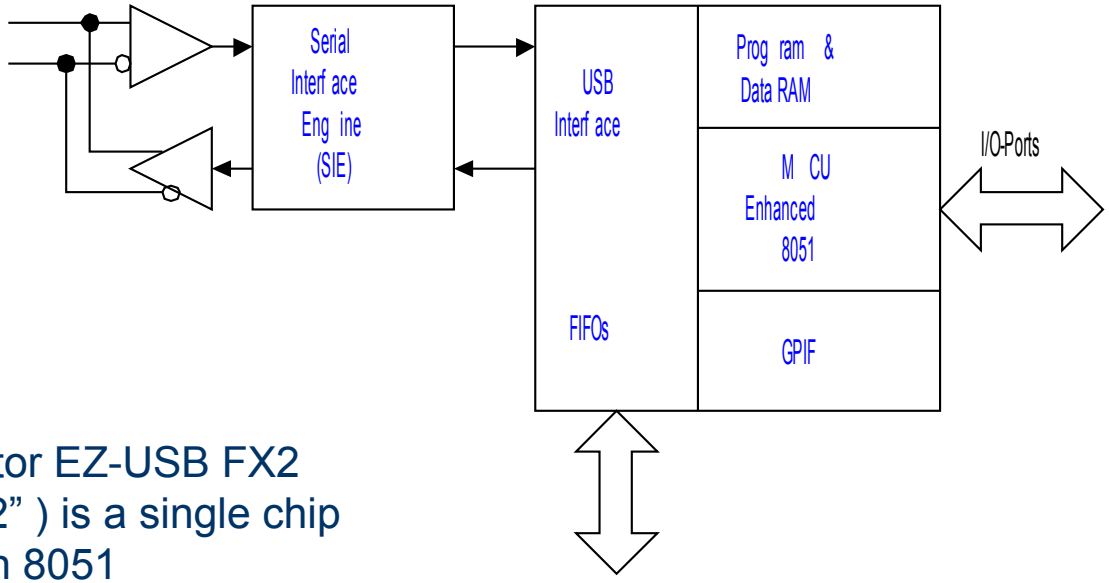


A sine wave sampled with 4-bit



A sine wave sampled with 8-bit

SDR*ham*



The Cypress Semiconductor EZ-USB FX2 (often abbreviated as “FX2”) is a single chip USB 2.0 peripheral with an 8051 microcontroller core

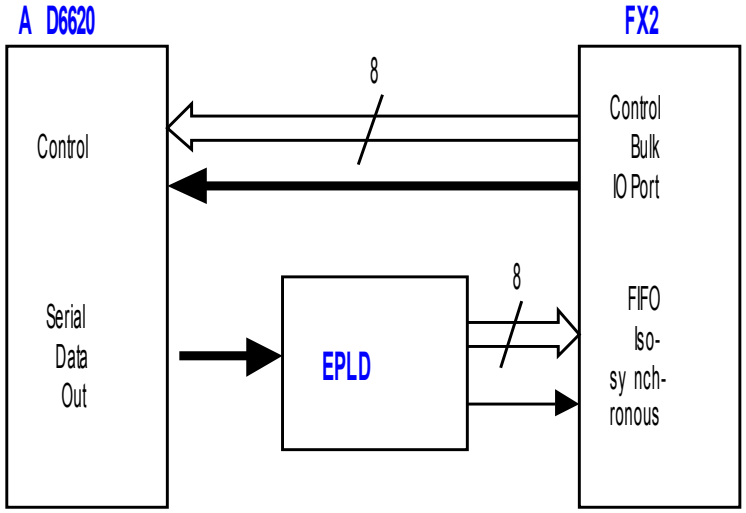
USB Transfer Types

- Bulk Transfers
  - Bulk data is *bursty*, traveling in packets of 8, 16, 32 or 64 bytes at full speed or 512 bytes at high speed. Bulk data has guaranteed accuracy, due to an automatic retry mechanism for erroneous data. The host schedules bulk packets when there is available bus time. Bulk transfers are typically used for printer, scanner, or modem data. Bulk data has built-in flow control provided by handshake packets.
- Interrupt Transfers
  - Interrupt data is like bulk data; it can have packet sizes of 1 through 64 bytes at full speed or up to 1024 bytes at high speed. Interrupt endpoints have an associated polling interval that ensures they will be polled (receive an IN token) by the host on a regular basis.
- Isochronous Transfers
  - Isochronous data is time-critical and used to *stream* data like audio and video. An isochronous packet may contain up to 1023 bytes at full speed, or up to 1024 bytes at high speed.

The FX2 Full-Speed Alternate Settings

Alternate Setting	0	1	2	3
ep0	64	64	64	64
ep1out	0	64 bulk	64 int	64 int
ep1in	0	64 bulk	64 int	64 int
ep2	0	64 bulk out (2x)	64 int out (2x)	64 iso out (2x)
ep4	0	64 bulk out (2x)	64 bulk out (2x)	64 bulk out (2x)
ep6	0	64 bulk in (2x)	64 int in (2x)	64 iso in (2x)
ep8	0	64 bulk in (2x)	64 bulk in (2x)	64 bulk in (2x)
Note: "0" means "not implemented", "2x" means double buffered.				

From the driver point of view, the endpoints are referenced as pipes



An OUT bulk endpoint is used for control (AD6620 setup)

An IN isosynchronous endpoint is used for sample streaming

FX2 ReNumeration

Once the Default USB Device enumerates, it downloads firmware and descriptor tables from the host into the FX2's on-chip RAM. The FX2 then begins executing the downloaded code, which electrically simulates a physical disconnect/connect from the USB and causes the FX2 to enumerate again as a second device, this time taking on the USB personality defined by the downloaded code and descriptors. This patented secondary enumeration process is called "ReNumeration™."

FX2 Access from Windows

The FX2 is accessed through Windows API functions and a Cypress supplied driver *ezusb.sys*

The following table shows the function prototype for DeviceIoControl(). The EZ-USB GPD IOCTL reference will use the same function argument names.

```
BOOL DeviceIoControl(
      HANDLE       hDevice,            // handle to device of interest
      DWORD        dwIoControlCode,    // control code of operation to perform
      LPVOID       lpInBuffer,         // pointer to buffer to supply input data
      DWORD        nInBufferSize,      // size of input buffer
      LPVOID       lpOutBuffer,        // pointer to buffer to receive output data
      DWORD        nOutBufferSize,     // size of output buffer
      LPDWORD      lpBytesReturned,    // pointer to variable to receive output
                                       // byte count
      LPOVERLAPPED lpOverlapped        // pointer to overlapped structure for
                                       // asynchronous operation
);
```

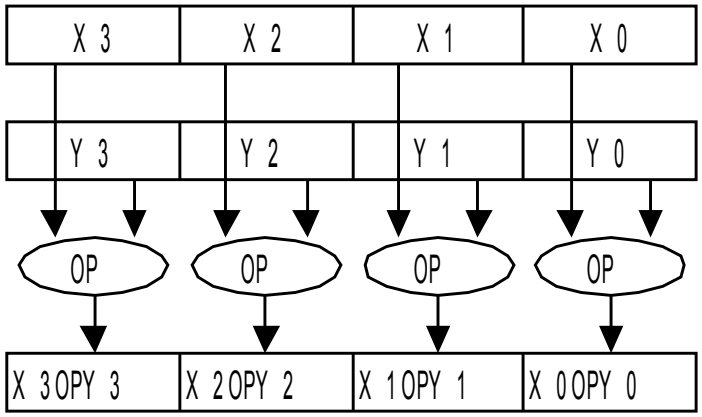
SDR*ham*

MMX Instruction Set

The MMX instructions defines a SIMD (Single Instruction Multiple Data) model to handle 64-bit packed integer data. MMX adds the following new features to the Intel IA-32 architecture:

- Eight new 64-bit MMX registers (aliased in th x87 register stack)
- Three new packed data types
  - 64-bit packed byte integers (signed and unsigned)
  - 64-bit packed word integers (signed and unsigned)
  - 64-bit packed double word integers (signed and unsigned)
- New instructions to support the new data types and to handle the MMX state management

MMX Execution Model and Instructions



MMX Instructions

- Data Transfer
- Arithmetic
- Comparison
- Unpacking
- Logical
- Shift
- Empty MMX State Instruction

SDR*ham*

MMX Convolution Routine

```
//---------------------------------------------------------------------------

void DSP001::MMXConvolute(int *src_ptr,int *dest_ptr,unsigned int size)
{
  int dummy[]={0,0};
  int *end_of_buffer=&psig_buf[(size_filter<<1)-2];
  int *pbuf_sig=psig_buf;
  int *pbuf_filt=pfilt_buf;
  int *pbuf_signal=psignal_buf;
  unsigned int filtersize=size_filter;

  asm
  {
      MOV     ECX,size
      TEST    ECX,-1
      JZ      no_convolute
                  //here to perform convolution
      MOV     EBX,pbuf_signal //EBX is pointer to data line
      MOV     ESI,src_ptr   //ESI is pointer to source data
      MOV     EDI,dest_ptr  //EDI is pointer to destination data
    next_conv:

filter_loop:
      MOVQ    MM0,QWORD PTR [EBX]
      PSRAD   MM0,8     //shift sourec to 16-bit
      PMULHW  MM0,QWORD PTR [EDI]
      PADDSW  MM1,MM0     //accumulated vale is in MM1
      CMP     EBX,end_of_buffer
      JGE     new_start_buffer
      ADD     EBX,8
      JMP     next_loop
    new_start_buffer:
      MOV     EBX,pbuf_sig
```

SDR*ham*

SSE Instruction Set

The SSE (Streaming SIMD Extensions) add the following features to the IA-32 architecture

- Eight 128-bit data registers, called XMM registers
- 128-bit packed single-precision floating point data type (four IEEE single-precision floating point values packed into a double quad word)
- Instructions that perform SIMD operations on single-precision floating point values and that extend the SIMD operations that can be performed on integers:
  - 128-bit packed and scalar single-precision floating point instructions that operate on data located in the XMM registers
  - 64-bit SIMD integer instructions that support additional operations on packled integer operands located in MMX registers

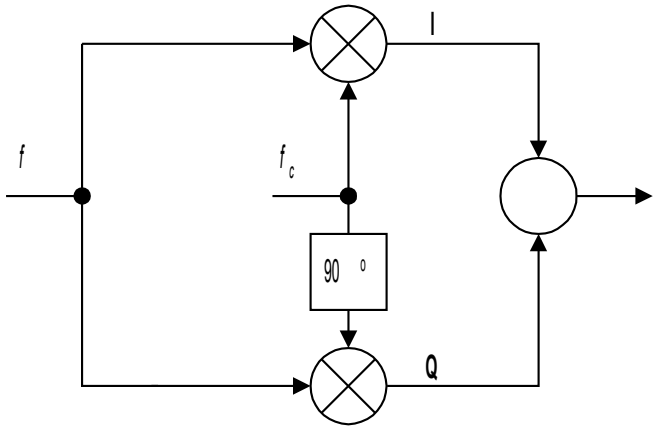
The SSE extensions were introduced in the Pentium III processor family

Give me In-Phase (I) and Quadrature (Q) signal components, and you can get everything. The I- and Q-components contain both amplitude and phase information of a received signal. Frequency is found from phase change per time interval.

$$AM = \sqrt{I^2 + Q^2}$$

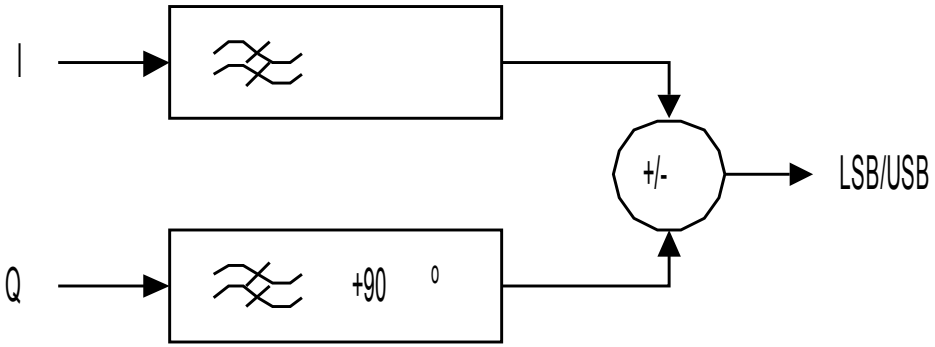
$$PM = \tan^{-1}\left(\frac{Q}{I}\right)$$

$$FM = \frac{d\left(\Theta(t)\right)}{dt}$$



$$SSB = \cos(\omega \cdot t)\cos(\omega_c \cdot t) \pm \cos(\omega \cdot t + 90^\circ)\cos(\omega_c \cdot t + 90^\circ)$$

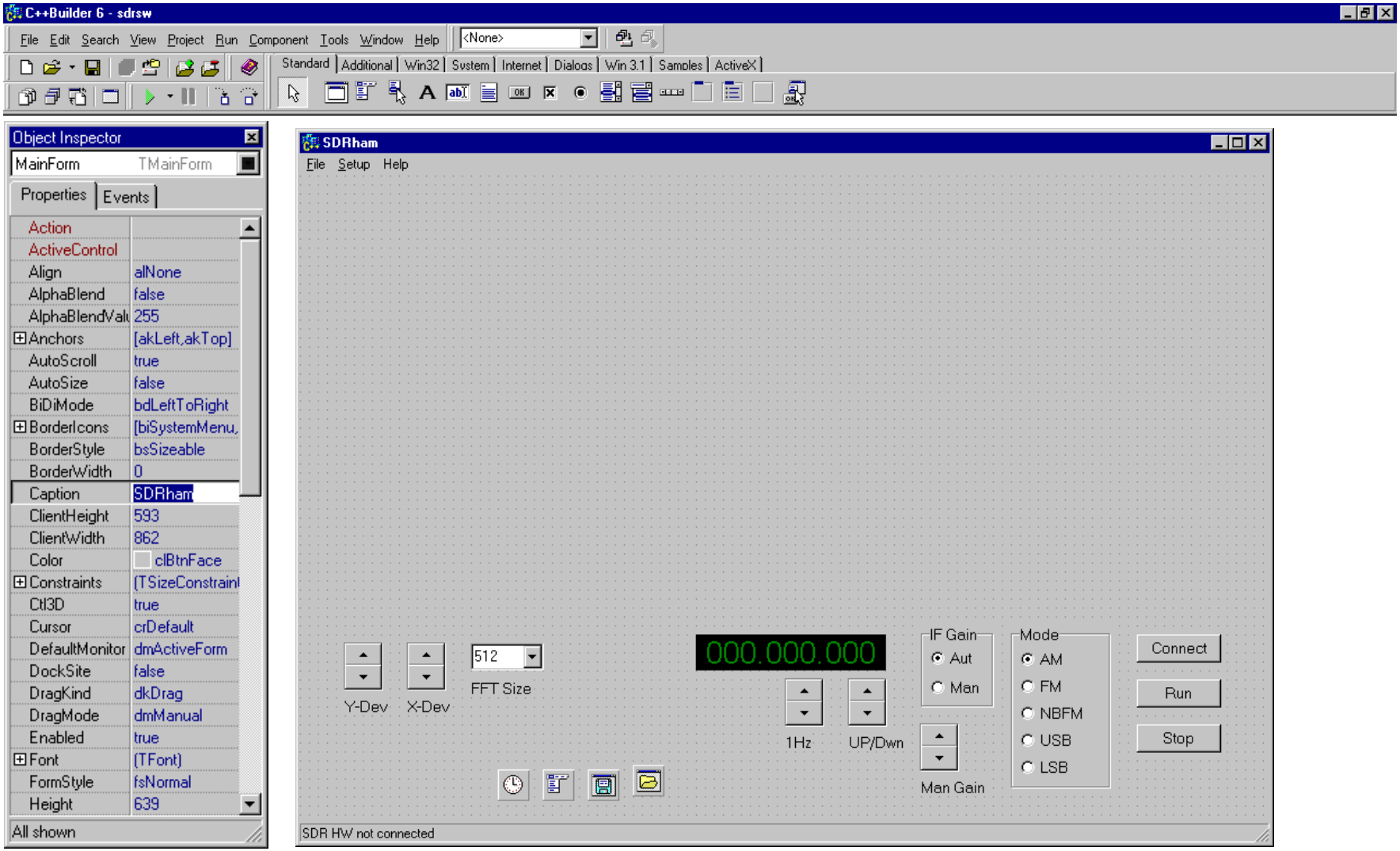
The phasing method



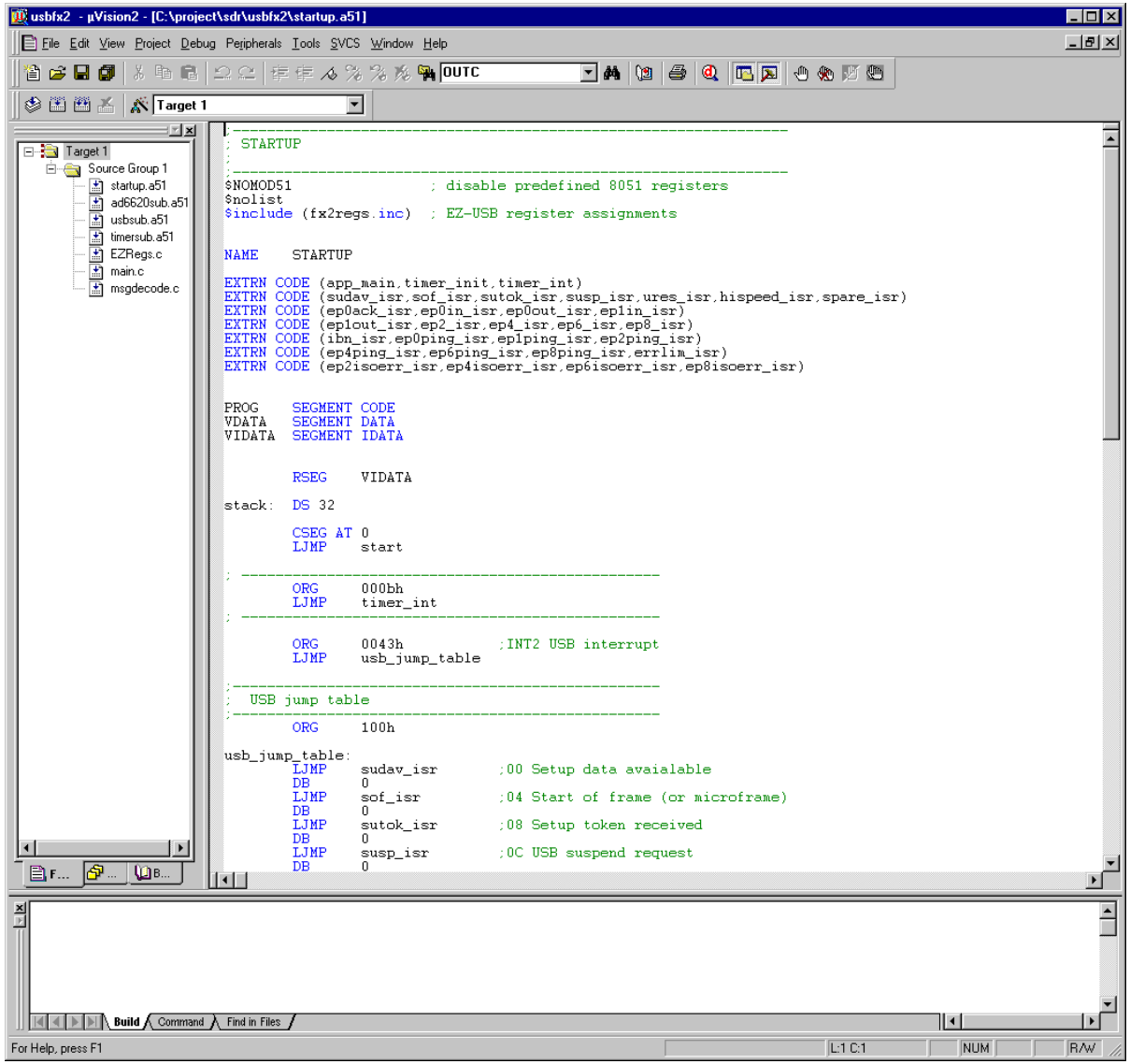
To obtain a filter with 90° phase shift over many octaves is not so easy in the analog world. This is not a problem in a digital world (FIR filter)

Tools

- Borland C++Builder6
  - Personal Edition, approx. EUR 100
- Keil uVision
  - Free evalution version, 4k byte code maximum
  - www.cypress.com or www.keil.com
- EZ USB Control Panel
  - Free, www.cypress.com
- Quartus II
  - Free WEB edition, www.altera.com
- MathCad, www.mathsoft.com
- Ansoft Harmonica
  - Free student version with limited number of nodes, www.ansoft.com

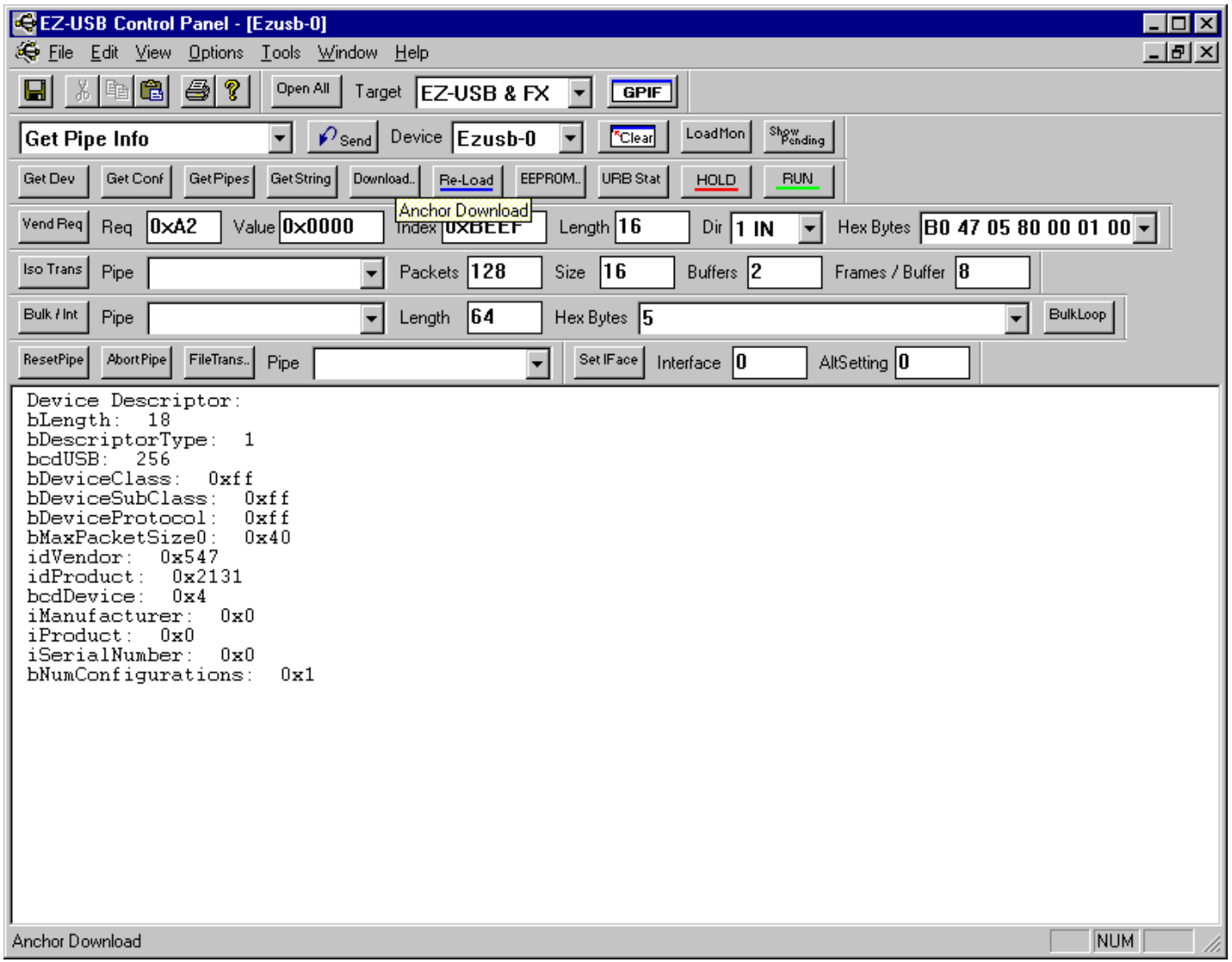

C++Builder 6 - sdrsw
File Edit Search View Project Run Component Tools Window Help
<None>
Standard Additional Win32 System Internet Dialogs Win 3.1 Samples ActiveX
Object Inspector
MainForm TMainForm
Properties Events
Action
ActiveControl
Align alNone
AlphaBlend false
AlphaBlendVal 255
Anchors [akLeft,akTop]
AutoScroll true
AutoSize false
BiDiMode bdLeftToRight
BorderIcons [biSystemMenu,
BorderStyle bsSizeable
BorderWidth 0
Caption SDRham
ClientHeight 593
ClientWidth 862
Color clBtnFace
Constraints (TSizeConstrain
Ctl3D true
Cursor crDefault
DefaultMonitor dmActiveForm
DockSite false
DragKind dkDrag
DragMode dmManual
Enabled true
Font (TFont)
FormStyle fsNormal
Height 639
All shown
SDRham
File Setup Help
Y-Dev
X-Dev
512
FFT Size
000.000.000
1Hz
UP/Dwn
IF Gain
Aut
Man
Man Gain
Mode
AM
FM
NBFM
USB
LSB
Connect
Run
Stop
SDR HW not connected

Keil uVision2

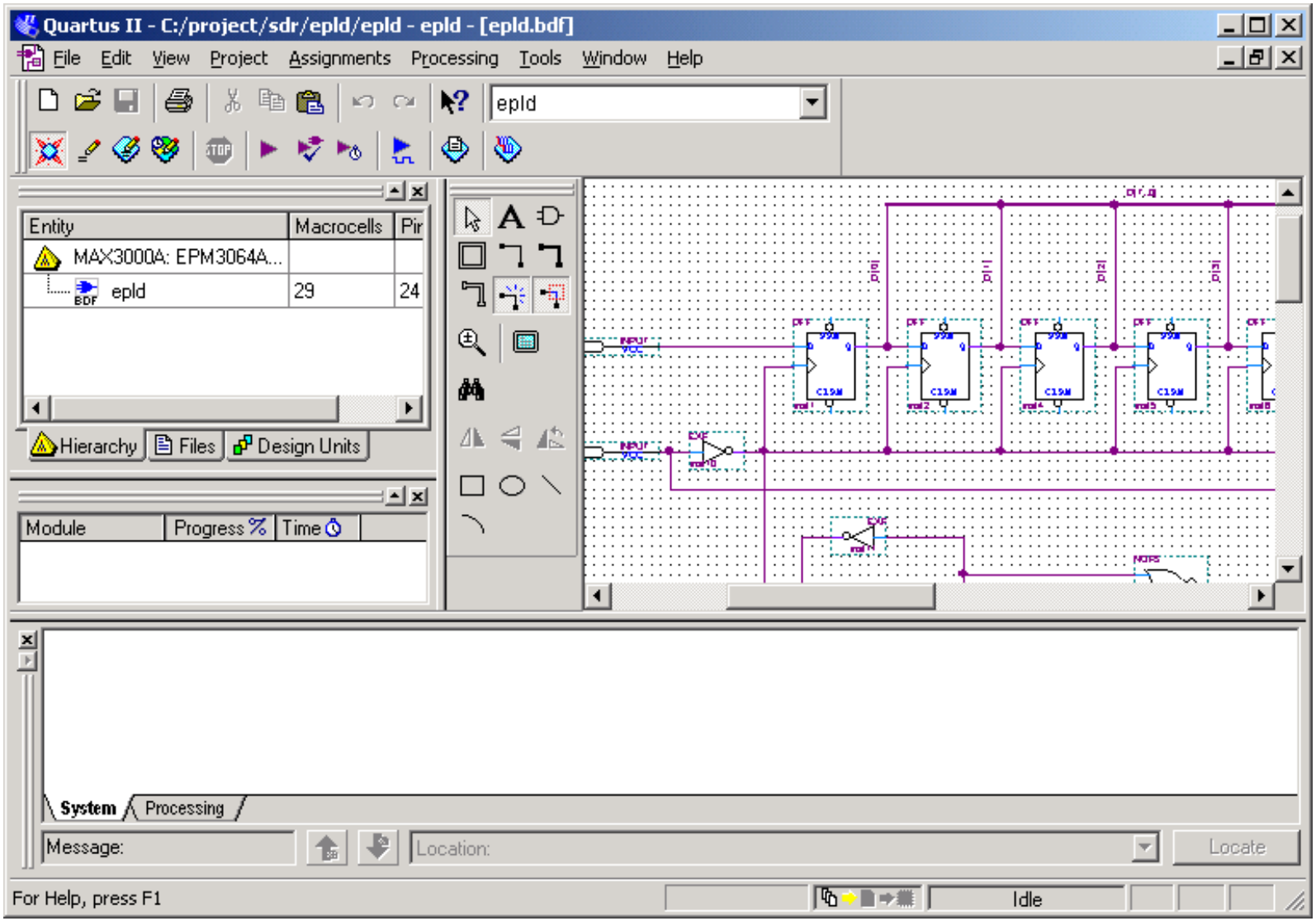


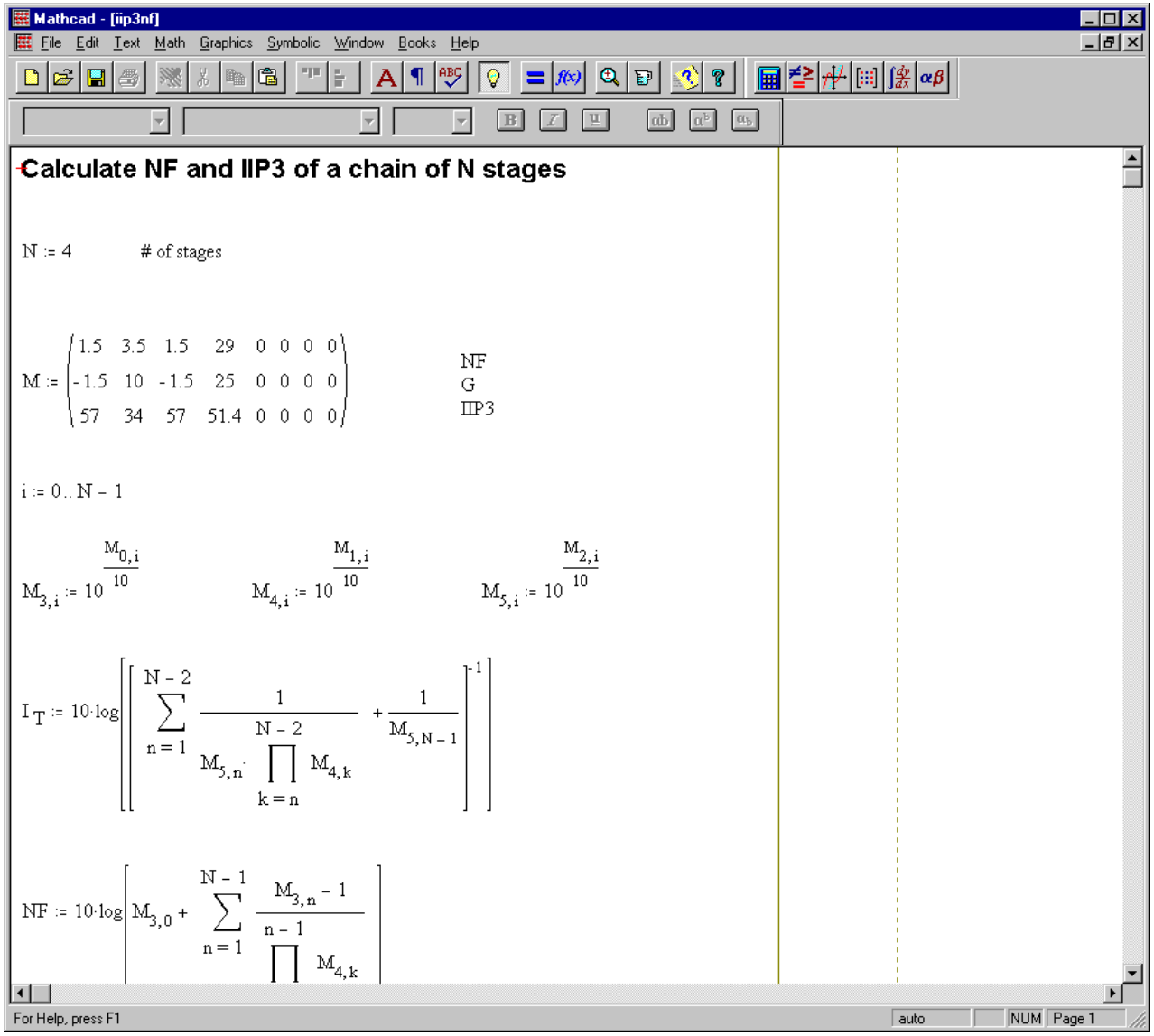
SDR*ham*

EZ-USB Control Panel

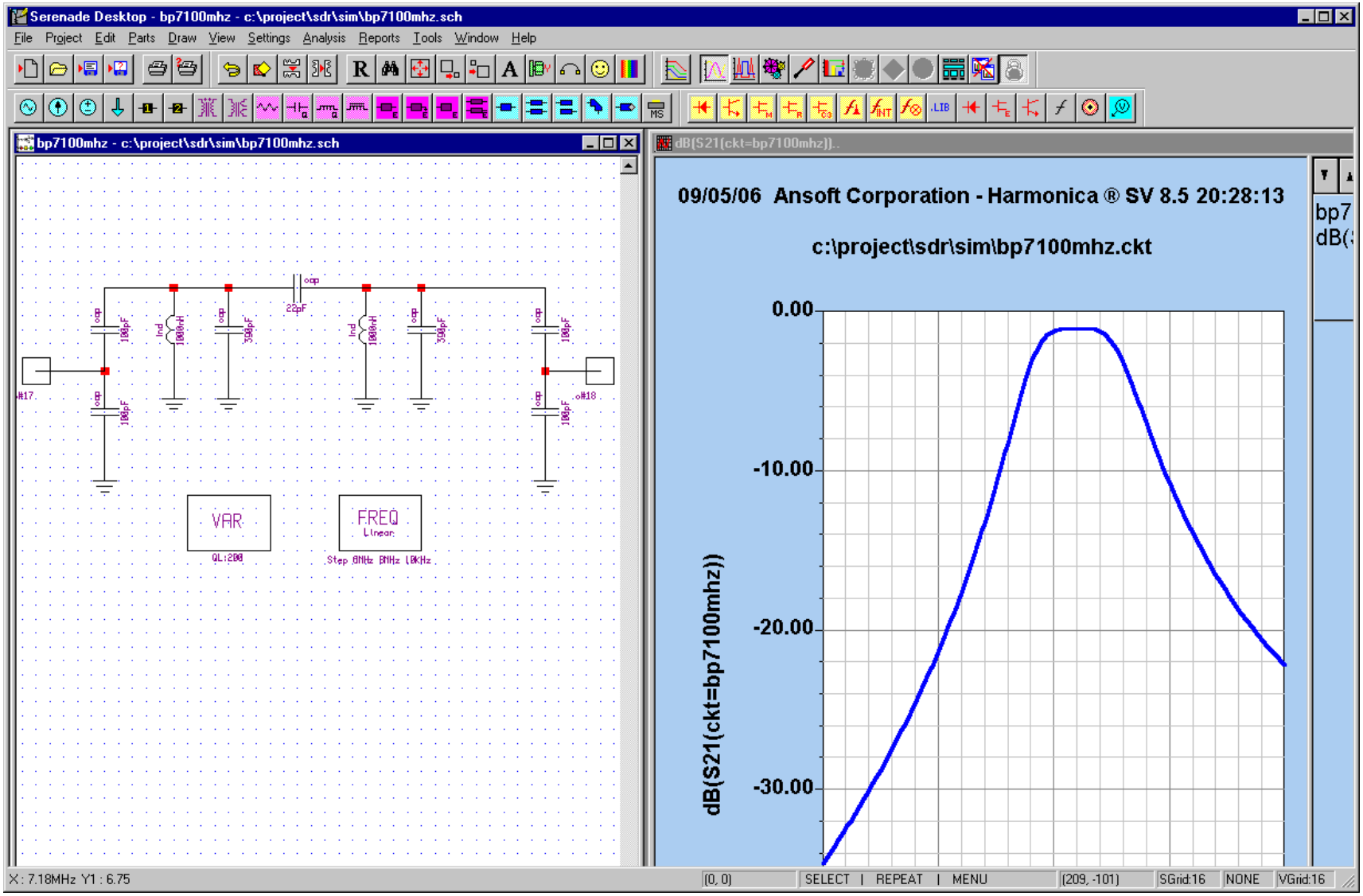


SDR*ham*


Quartus II - C:/project/sdr/epld/epld - epld - [epld.bdf]
File Edit View Project Assignments Processing Tools Window Help
epld
Entity
Macrocells
Pir
MAX3000A: EPM3064A...
epld
29
24
Hierarchy
Files
Design Units
Module
Progress
Time
System
Processing
Message:
Location:
Locate
For Help, press F1
Idle


Mathcad - [iip3nf]
File Edit Text Math Graphics Symbolic Window Books Help
Calculate NF and IIP3 of a chain of N stages
N := 4 # of stages
M := 1.5 3.5 1.5 29 0 0 0 0
-1.5 10 -1.5 25 0 0 0 0
57 34 57 51.4 0 0 0 0
NF
G
IIP3
i := 0 .. N - 1
M3,i := 10^(M0,i / 10)
M4,i := 10^(M1,i / 10)
M5,i := 10^(M2,i / 10)
IT := 10·log([Σ n=1..N-2 1 / (M5,n · Π k=n..N-2 M4,k) + 1 / M5,N-1]^-1)
NF := 10·log(M3,0 + Σ n=1..N-1 (M3,n - 1) / Π k=n-1..N-1 M4,k)
For Help, press F1
auto
NUM
Page 1

Ansoft Harmonica



SDR*ham*

Three important parameters

- MDS (Minimum Discernible Signal)
  - Capability to pick weak signals out of the noise
- DR (Dynamic Range)
  - Capability to read weak signals with other strong ones in the receiver front end pass band
- IIP3 (3rd order Input Intercept Point)
  - Gives information of the receiver two tone DR

The MDS is defined as the signal level at the receiver input that doubles the output signal, that is 3 dB output increase.

A calculated MDS for a ideal receiver (NF=0) with 3 kHz bandwidth

$$P_n = kTB = 1.38 \cdot 10^{-23} \cdot 290 \cdot 3000 = 1.2 \cdot 10^{-17} (\text{Watt})$$

$$P_n = 10 \cdot \log\left(\frac{p_n}{0.001}\right) = 10 \cdot \log\left(\frac{1.2 \cdot 10^{-17}}{1 \cdot 10^{-3}}\right) = -139\, dBm$$

P_n in 1Hz bandwidth is -174 dBm

$$MDS = -174\, dBm + 10 \cdot \log(B) + NF$$

MDS in the HF Bands

In the 14 MHz band (20 meters) the man made and atmospheric NF is estimated to 22 dB (KD7O). Then the receiver NF needs not to be much better. As a rule of thumb a receiver NF of 6 dB better than the background noise is enough.

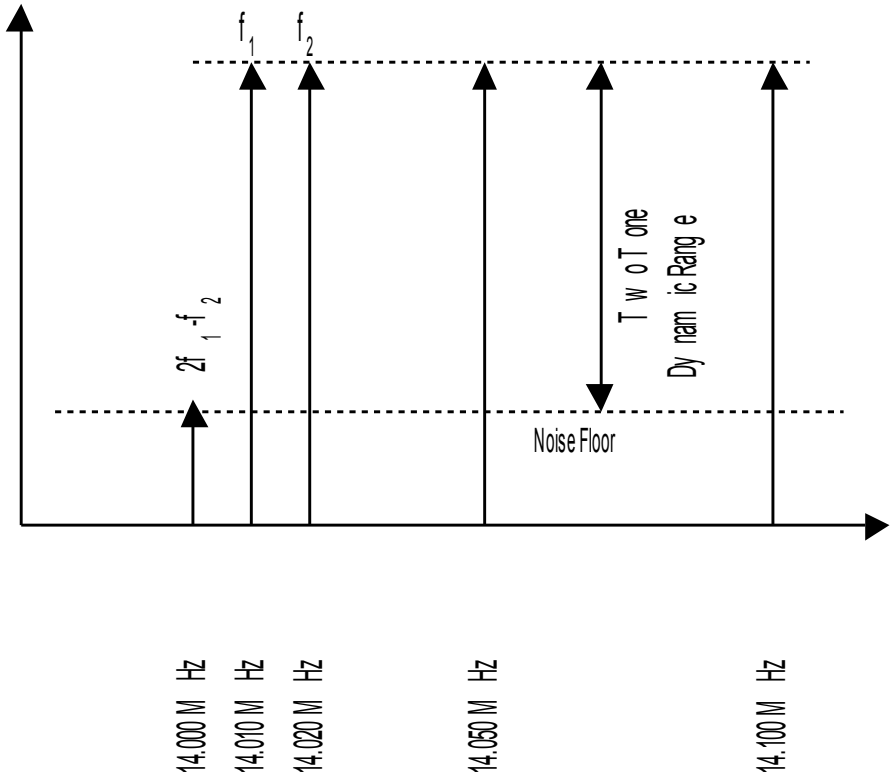
For the 14 MHz band an analog front end with some gain is needed. (The NF of the ADC is calculated to 29 dB). To maintain high DR this amplifier needs a high IIP3, which is a design challenge.

In the lower short wave band, 1.8 MHz (160 meter), 3.6 MHz (80 meter) and 7.0 MHz (40 meter) no gain is expected to be used in front of the ADC (to be confirmed by tests).

In the VHF band, where the background noise is low, an analog front end with a Low Noise Amplifier (LNA) is needed.

Two Tone Dynamic Range

Why is the two tone dynamic range so important?



Third order distortion products are impossible to remove by filters in the analog front end.

SDR*ham*

Mathematics of Third Order Distortion

An amplifier or ADC is not ideal. A transfer function is given by

$$V_{out} = K_0 + K_1 V_{in} + K_2 V_{in}^2 + K_3 V_{in}^3 + \ldots\ldots = \sum_0^\infty K_n V_{in}^n$$

$K_3 V_{IN}^3$ is 3rd order term and makes distortion products in the passband.
Expanding this term gives:

$$V_{out} = E_3 \left(E_1^3 \sin^3 \omega_1 t + E_2^3 \sin^3 \omega_2 t + 3 E_1^2 E_2 \sin^2 \omega_1 t \sin^2 \omega_2 t + 3 E_2^2 E_1 \sin^2 \omega_1 t \sin^2 \omega_2 t \right)$$

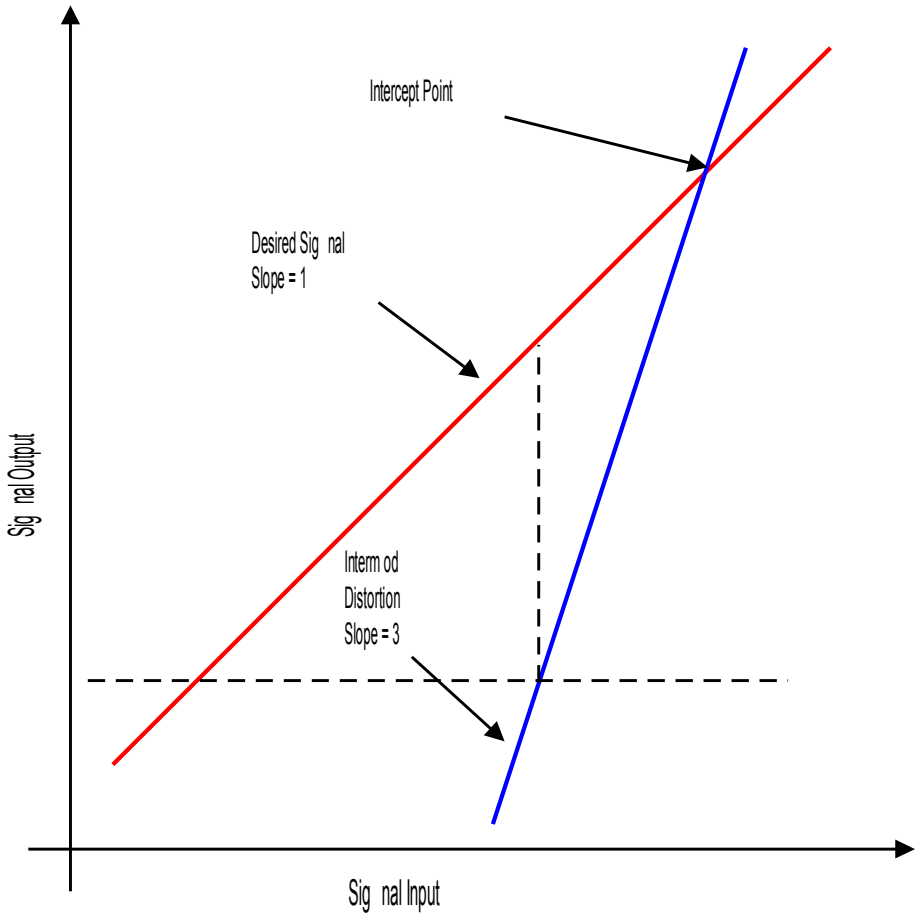
Expanding the last term gives:

$$\omega = 2 \cdot \pi \cdot f$$

$$V'_{out} = \frac{3 E_1^2 E_2 K_3}{2} \left\{ \sin \omega_2 t - \frac{1}{2} \left[\sin(2\omega_1 + \omega_2) t - \sin(2\omega_1 - \omega_2) t \right] \right\}$$

SDR*ham*

IMD Ratio and 3rd order Intercept Point

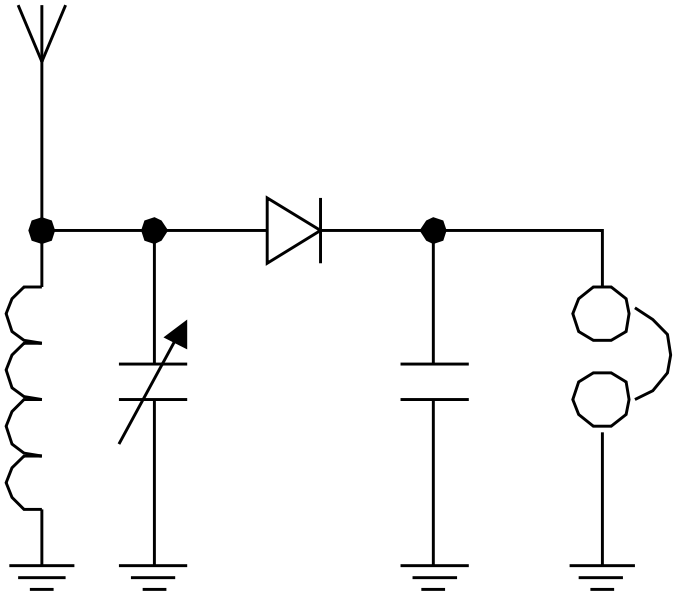


The intercept point is a figure of merit that is commonly used to describe the IMD (Inter Modulation Distortion) performance of an individual stage or a complete system

SDR*ham*

Signal Levels in MF and low HF Bands

In the MF and low HF bands extreme strong signals may appear. Remember from old days, the circuit below works.



A signal of S9 (defined to –73 dBm) +40 dB has a signal voltage given by

$$-33 = 10\log\left(\frac{P}{10^{-3}}\right)$$

$$P = 10^{-3}\,10^{\frac{-33}{10}} = 5\cdot 10^{-6}$$

$$U = \sqrt{PR} = \sqrt{5\cdot 10^{-6}\cdot 50} = 16\cdot 10^{-3}$$

SDR*ham*

A MathCad worksheet is developed to analyse receiver NF and IIP3

$$M := \begin{pmatrix} 1.5 & 3.5 & 1.5 & 29 & 0 & 0 & 0 & 0 \\ -1.5 & 10 & -1.5 & 25 & 0 & 0 & 0 & 0 \\ 57 & 34 & 57 & 51.4 & 0 & 0 & 0 & 0 \end{pmatrix} \quad \begin{matrix} NF \\ G \\ IIP3 \end{matrix}$$

$$IIP3 = \frac{1}{\frac{1}{IIP3_1} + \frac{G}{IIP3_2}}$$

$$I_T := 10 \cdot \log\left[\left[\sum_{n=1}^{N-2} \frac{1}{M_{5,n} \cdot \prod_{k=n}^{N-2} M_{4,k}} + \frac{1}{M_{5,N-1}}\right]^{-1}\right]$$

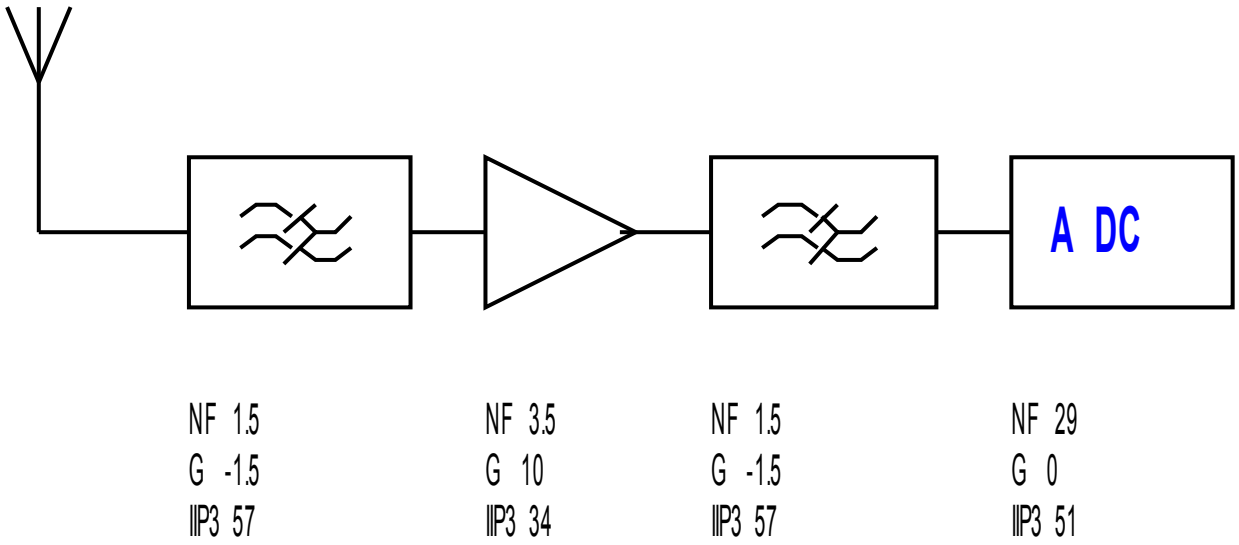
$$IIP_3 := I_T - \sum_{k=0}^{N-2} M_{1,k}$$

$$NF := 10 \cdot \log\left[M_{3,0} + \sum_{n=1}^{N-1} \frac{M_{3,n} - 1}{\prod_{k=0}^{n-1} M_{4,k}}\right]$$

$$F = F1 + \frac{F2 - 1}{G1}$$

SDR*ham*

A Receiver Example



NF = 22.1 dB IIP_3 = 34.8 dBm

SDR*ham*

Test Procedures Manual

by Michael Tracy, KC1SX, ARRL Test Engineer and
Mike Gruber, W1MG, ARRL EMC/RFI Engineer

Manufacturer's Specifications	Measured in ARRL Lab
Frequency coverage: Receive, 0.025-260 MHz.	Receive, as specified.[1]
Modes of operation: AM, FM, WFM, SSB, DSB, CW.[2]	As specified.
Power requirements: 0.2 A (max), 12 V dc.	0.39 A, tested at 12 V dc.
CW/SSB sensitivity: MDS –125 dBm at 500 Hz bandwidth, attenuator set to –10 dB and IF gain at +12 dB.	Noise floor (MDS), estimate: 14 MHz, 500 Hz bandwidth, –125 dBm.[3]
AM sensitivity: Not specified.	For 10 dB S+N/N: 14 MHz, 2.1 μV.
FM sensitivity: Not specified.	For 12 dB SINAD: 14 MHz, 0.9 μV.
Blocking dynamic range: Not specified.	14 MHz, ADC overload, 110 dB.[4]
Two-tone, third-order IMD dynamic range: Not specified.	20 kHz spacing: 14 MHz, 34 dB.
Third-order intercept: Not specified.	14 MHz, –15 dBm to +29 dBm.[5]
Second-order intercept: Not specified.	14 MHz, +62 dBm.
Size (height, width, depth): 1.5"×5.5"×6.5"; weight, 1.2 pounds.	

All dynamic range measurements were taken using the ARRL Lab standard spacing of 20 kHz. Third-order intercept points were determined using S5 reference.

SDR*ham*

$$MDS = -174\,dBm + 10 \cdot \log(B) + NF$$

Calculated MDS

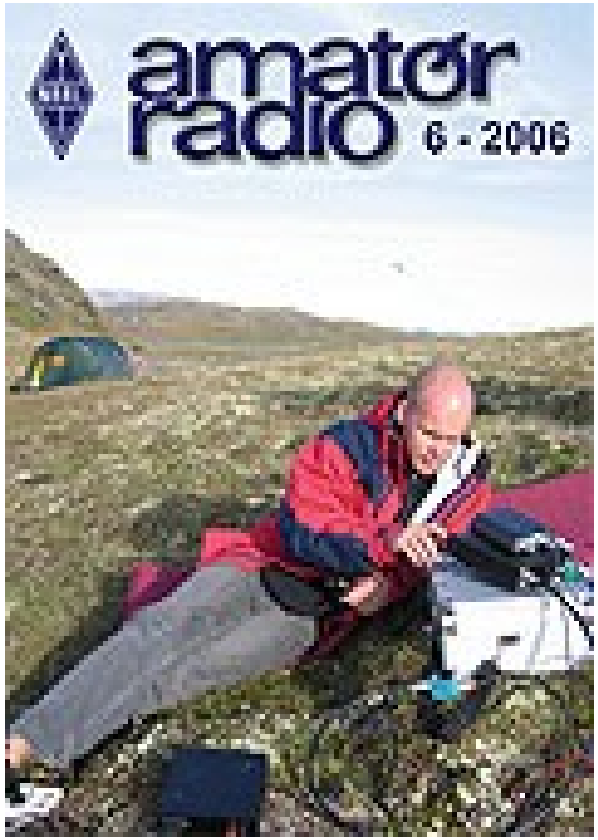
$$MDS = -174\,dBm + 10 \cdot \log(3000) + 29 = -110\,dBm$$

Measured MDS about as the calculated one

Some unexpected results seen in two tone dynamic range measurements.

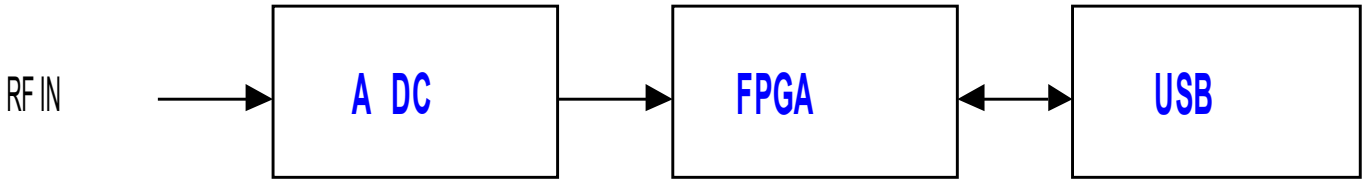
Tones at frequencies that say they are third order products, but their amplitudes do not follow the cube law.

Project Publication



A series of articles are and will be published in the NRRL (Norwegain Radio Relay League) bulletin AMATØR RADIO to promote SDR technology among Norwegian radio amateurs.

SDR*ham*



FE for receiver, spectrum analyser and oscilloscope applications

AD9254 - 14-Bit, 150 MSPS, 1.8 V ADC

- SNR = 71.7 dBc (72.7 dBFS) to 70 MHz input
- SFDR = 83 dBc to 70 MHz input
- Low power: 250 mW @ 150 MSPS
- Differential input with 650 MHz bandwidth