Digital Processing with Focus onto Neutron Detection

SNRI-V INFN, Padova

Gabriele Pasquali
(pasquali@fi.infn.it)

Università di Firenze
and INFN-Sezione di Firenze

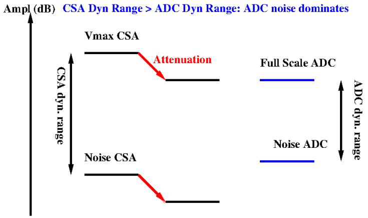
25-26 October 2016

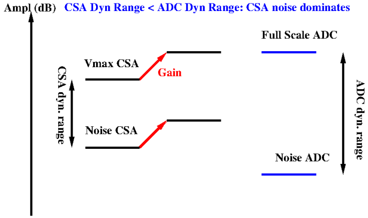
Second Lesson (2016-10-26)

Timing as a study case

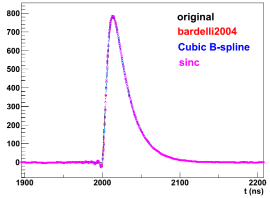
Remind from first lesson

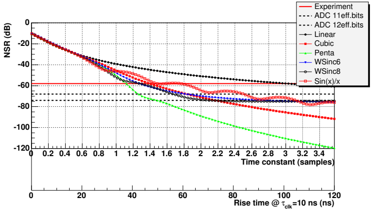
- ADCs can add noise to your signal ($\propto 2^{N-\mathrm{ENOB}}$)





- Signal reconstruction can add artifacts and "noise" for fast transients ($\leq \mathrm{Kernel\,Lenght} \times T_s$)





Timing with LED

- Timing: extracting a "time mark" from a signal, e.g. with a leading edge discriminator (LED);

Timing with LED

- Timing: extracting a “time mark” from a signal, e.g. with a leading edge discriminator (LED);
- LED: device emitting a logic “true” signal when input voltage crosses a fixed threshold (e.g. oscilloscope trigger)

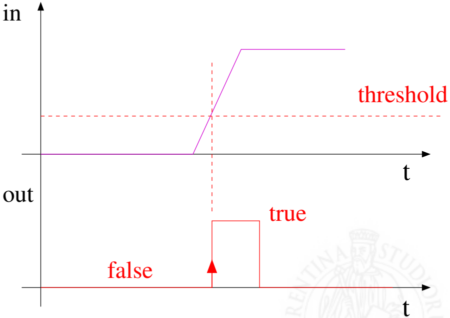


Figure 4: Leading edge discriminator

LED and *amplitude walk*

In a LED, threshold crossing depends on amplitude for a fixed risetime. Reason: threshold is fixed.

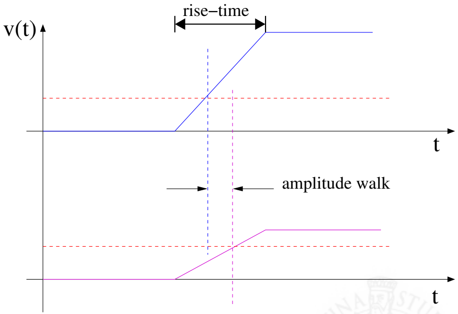


Figure 5: Amplitude walk of a LED.

Constant Fraction Discrimination

- a Constant Fraction Discriminator acts as if its threshold could move dynamically: threshold is a fixed fraction *f* of full amplitude;

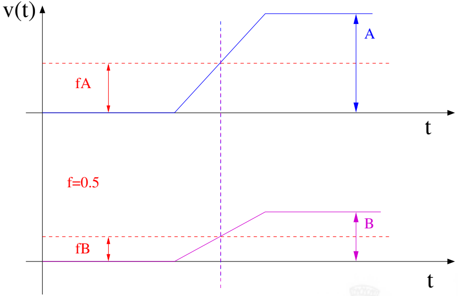


Figure 6: Constant Fraction Discriminator principle.

Constant Fraction Discrimination

- a Constant Fraction Discriminator acts as if its threshold could move dynamically: threshold is a fixed fraction f of full amplitude;

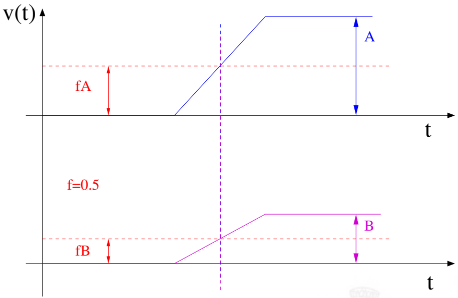
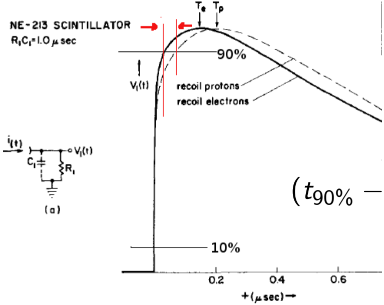


Figure 6: Constant Fraction Discriminator principle.

- amplitude walk reduced (eliminated exactly for a linear rising edge)

CFD and PSD

CFD useful also in Pulse Shape Discrimination: NE-213 anode current signal integrated on RC parallel $\Longrightarrow$ the slower component of a proton signal (i.e. neutron detected) is associated to a longer risetime with respect to electron signal (i.e. gamma detected)

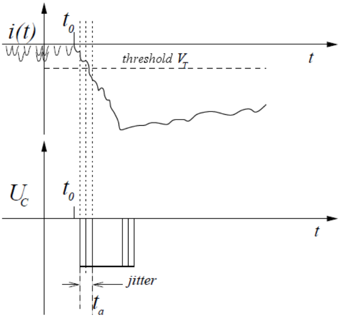


$$(t_{90\%} - t_{10\%})_{\mathrm{p}} > (t_{90\%} - t_{10\%})_{\mathrm{e}}$$

Figure 7: PSD from risetime (adapted from [Roush1964]).

Timing and noise: jitter

noise fluctuations affect signal $\Longrightarrow$ time mark fluctuates around average



$\Longrightarrow$ *jitter*: statistical time-mark fluctuations

Jitter: a simple model

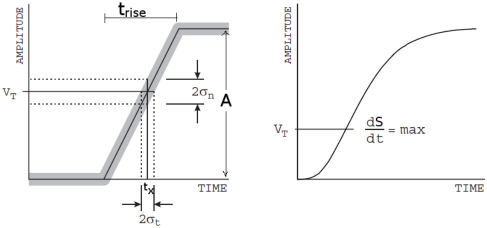


Figure 8: Noise and jitter (adapted from [Spieler2005])

- σ_n std. dev. **amplitude** fluctuations → "noise band" $2\sigma_n$ wide

Jitter: a simple model

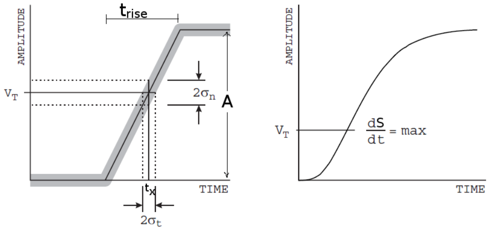


Figure 8: Noise and jitter (adapted from [Spieler2005])

- σ_n std. dev. **amplitude** fluctuations → "noise band" $2\sigma_n$ wide
- project σ_n on time axis: $\sigma_t = \dfrac{\sigma_n}{[|dS/dt|_{t_x}]}$ where $S(t_x) = V_T$

Jitter: a simple model

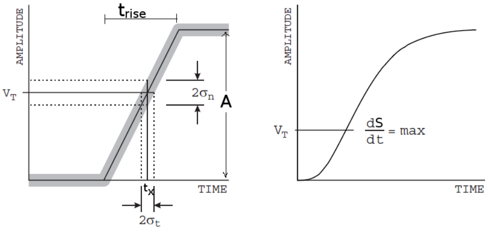


Figure 8: Noise and jitter (adapted from [Spieler2005])

- σ_n std. dev. **amplitude** fluctuations → "noise band" $2\sigma_n$ wide
- project σ_n on time axis: $\sigma_t = \dfrac{\sigma_n}{[|dS/dt|_{t_x}]}$ where $S(t_x) = V_T$
- we put threshold where $|dS/dt|$ is max $\implies \sigma_t$ minimum

Jitter: a simple model

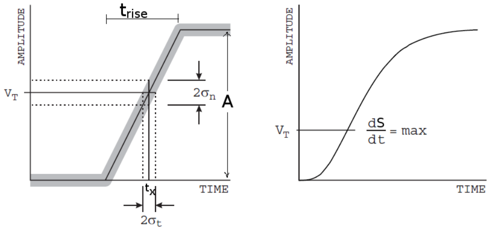


Figure 8: Noise and jitter (adapted from [Spieler2005])

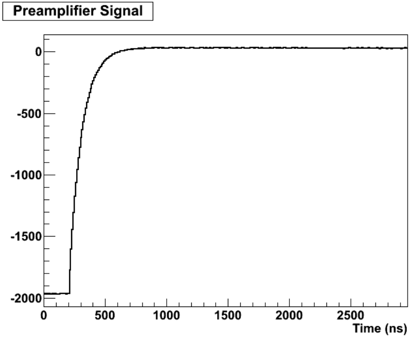
- σ_n std. dev. **amplitude** fluctuations $\rightarrow$ "noise band" $2\sigma_n$ wide
- project σ_n on time axis: $\sigma_t = \dfrac{\sigma_n}{[|dS/dt|_{t_x}]}$ where $S(t_x) = V_T$
- we put threshold where $|dS/dt|$ is max $\Longrightarrow \sigma_t$ minimum
- linear signal front: $\left|\dfrac{dS}{dt}\right| = \dfrac{A}{t_{rise}} \Longrightarrow$

$$\sigma_t = \frac{\sigma_n t_{rise}}{A} \propto \frac{t_{rise}}{\mathrm{SNR}} \tag{3}$$

A digital-CFD (dCFD)

CFD procedure for a "tail" signal (e.g. from charge preamp):

1) apply pole-zero cancellation + integration
to get rid of tail



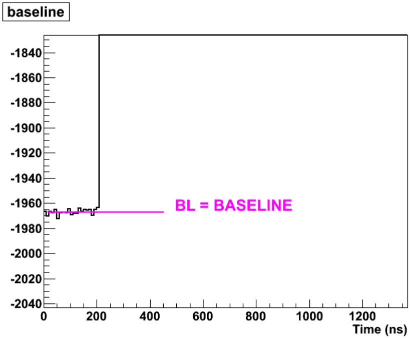
Please note:

1. the time axis unit is ns;
2. original (not interpolated) signal has $T_s = 10\,\mathrm{ns}$.

A digital-CFD (dCFD)

CFD procedure for a "tail" signal (e.g. from charge preamp):

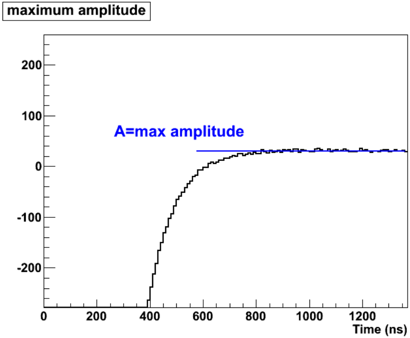
2) calculate the baseline BL (e.g. averaging flat part: also consider noise autocorrelation, e.g. when calculating rise-time)



A digital-CFD (dCFD)

CFD procedure for a “tail” signal (e.g. from charge preamp):

3) calculate max amplitude A (samples average or amplitude of unit gain shaper); step amplitude $= A - BL$

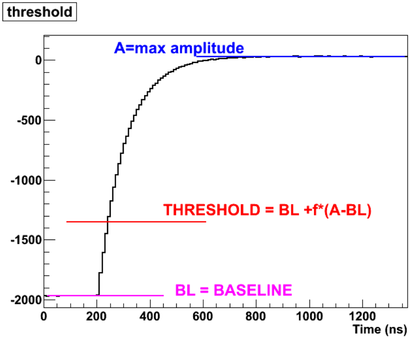


A digital-CFD (dCFD)

CFD procedure for a "tail" signal (e.g. from charge preamp):

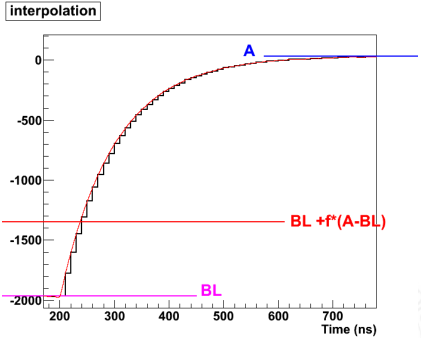
4) calculate dynamic threshold as

$T = BL + f\,(A - BL)$



A digital-CFD (dCFD)

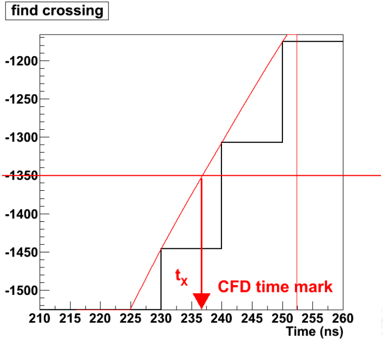
CFD procedure for a “tail” signal (e.g. from charge preamp):
5) apply interpolation (whole signal shown...
in real-life region around threshold is enough)



A digital-CFD (dCFD)

CFD procedure for a "tail" signal (e.g. from charge preamp):

6) time mark = intersection interpolation-threshold
(find it iteratively in complex cases)



Intersection time t_X is in units of T_s (fraction of the sampling period). Time in seconds from first sample $= t_X \cdot T_s$. If $x[n]$ last sample before t_X then $0 < t_x - n < 1$ (in this example, $n = 23$ $t_X \sim 23.68$).

t-measurement: Sampling ADC vs Analog

Effects affecting resolution of digital timing:

- the sampling ADC adds noise to that already present in our system $\Longrightarrow$ this will tend to increase our jitter

t-measurement: Sampling ADC vs Analog

Effects affecting resolution of digital timing:

- the sampling ADC adds noise to that already present in our system $\Longrightarrow$ this will tend to increase our jitter
- digitizing systems usually employ some kind of low-pass filter (antialias filter) before the ADC $\Longrightarrow$ also rise-time will be affected (i.e. slowed down) $\Longrightarrow$ jitter fluctuations increase

t-measurement: Sampling ADC vs Analog

Effects affecting resolution of digital timing:

- the sampling ADC adds noise to that already present in our system $\Longrightarrow$ this will tend to increase our jitter
- digitizing systems usually employ some kind of low-pass filter (antialias filter) before the ADC $\Longrightarrow$ also rise-time will be affected (i.e. slowed down) $\Longrightarrow$ jitter fluctuations increase
- on the other hand, low pass antialias filter will attenuate high frequency noise $\Longrightarrow$ jitter reduction

t-measurement: Sampling ADC vs Analog

Effects affecting resolution of digital timing:

- the sampling ADC adds noise to that already present in our system $\Longrightarrow$ this will tend to increase our jitter
- digitizing systems usually employ some kind of low-pass filter (antialias filter) before the ADC $\Longrightarrow$ also rise-time will be affected (i.e. slowed down) $\Longrightarrow$ jitter fluctuations increase
- on the other hand, low pass antialias filter will attenuate high frequency noise $\Longrightarrow$ jitter reduction
- detector signals have wide frequency bandwidth (*wideband* signals) $\Longrightarrow$ signal reconstruction from samples affected by interpolation errors $\Longrightarrow$ timing affected by interpolation "noise" (an effect not present in analog chains)

Jitter in a dCFD

- we get jitter as in analog LED or CFD [Bardelli2004];

Jitter in a dCFD

- we get jitter as in analog LED or CFD [Bardelli2004];
- assume signal perfectly reconstructed (e.g. original signal linear around threshold $\Longrightarrow$ linear interpolation perfect!), then

$$\begin{aligned} \sigma_t &\leq \frac{\sigma_{e+q}}{\left|\frac{\mathrm{d}S}{\mathrm{d}t}\right|_{t_x}} \\ \sigma_{e+q}^2 &= \sigma_e^2 + \frac{1}{12 \cdot 4^{\mathrm{ENOB}}} \end{aligned} \tag{4}$$

Jitter in a dCFD

- we get jitter as in analog LED or CFD [Bardelli2004];
- assume signal perfectly reconstructed (e.g. original signal linear around threshold $\Longrightarrow$ linear interpolation perfect!), then

$$\begin{aligned} \sigma_t &\leq \frac{\sigma_{e+q}}{\left|\frac{\mathrm{d}S}{\mathrm{d}t}\right|_{t_x}} \\ \sigma_{e+q}^2 &= \sigma_e^2 + \frac{1}{12 \cdot 4^{\mathrm{ENOB}}} \end{aligned} \tag{4}$$

- we are using units $R = 1$ (R: full range of the ADC)

Jitter in a dCFD

- we get jitter as in analog LED or CFD [Bardelli2004];
- assume signal perfectly reconstructed (e.g. original signal linear around threshold $\Longrightarrow$ linear interpolation perfect!), then

$$\begin{aligned} \sigma_t &\leq \frac{\sigma_{e+q}}{\left|\frac{\mathrm{d}S}{\mathrm{d}t}\right|_{t_x}} \\ \sigma_{e+q}^2 &= \sigma_e^2 + \frac{1}{12 \cdot 4^{\mathrm{ENOB}}} \end{aligned} \tag{4}$$

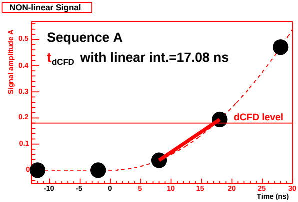
- we are using units $R = 1$ (R: full range of the ADC)
- NB: analog CFD similar formula except: equal sign, no ADC noise, a factor $\sqrt{1+f^2}$

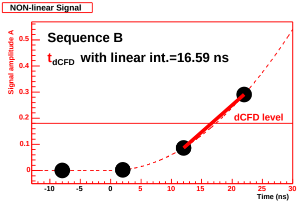
dCFD simulation: asynchronous sampling

- asynchronous sampling + interpolation $\Longrightarrow$ time mark fluctuation!

dCFD simulation: asynchronous sampling

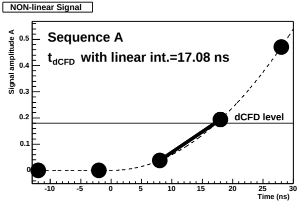
- asynchronous sampling + interpolation $\Longrightarrow$ time mark fluctuation!
- effect of interpolation in a simple case: linear interpolation

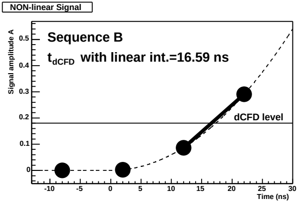




dCFD simulation: asynchronous sampling

- asynchronous sampling + interpolation $\Longrightarrow$ time mark fluctuation!
- effect of interpolation in a simple case: linear interpolation

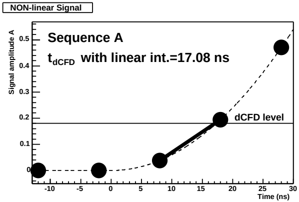


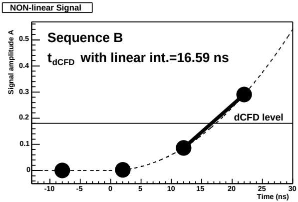


- non linear front $\Longrightarrow$ reconstruction not perfect

dCFD simulation: asynchronous sampling

- asynchronous sampling + interpolation $\Longrightarrow$ time mark fluctuation!
- effect of interpolation in a simple case: linear interpolation

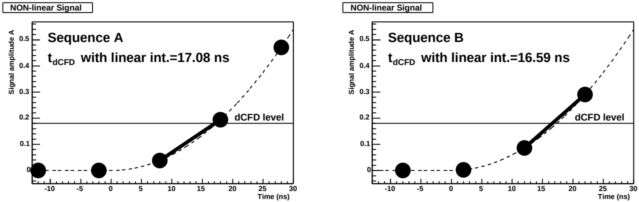




- non linear front $\Longrightarrow$ reconstruction not perfect
- for a fixed signal shape, t_x depends on **where** samples are taken

dCFD simulation: asynchronous sampling

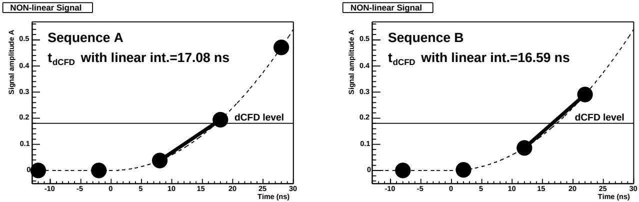
- asynchronous sampling + interpolation $\Longrightarrow$ time mark fluctuation!
- effect of interpolation in a simple case: linear interpolation



- non linear front $\Longrightarrow$ reconstruction not perfect
- for a fixed signal shape, t_x depends on **where** samples are taken
- i.e. on phase of sampling clock w/ respect to signal front

dCFD simulation: asynchronous sampling

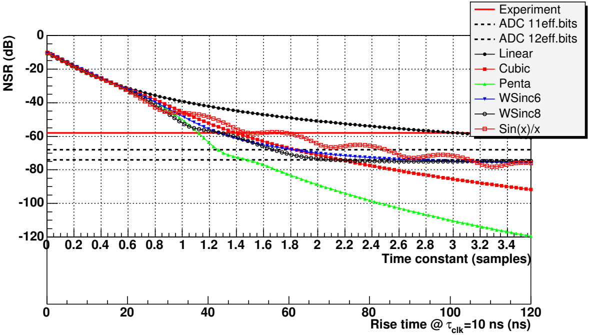
- asynchronous sampling + interpolation $\Longrightarrow$ time mark fluctuation!
- effect of interpolation in a simple case: linear interpolation



- non linear front $\Longrightarrow$ reconstruction not perfect
- for a fixed signal shape, t_x depends on **where** samples are taken
- i.e. on phase of sampling clock w/ respect to signal front
- will happen anyway w/ other kernels (not BW limited signal)

Questions about interpolation "noise"

- effect of interpolation different for linear and cubic;
- we know there are many kernels available...
- which kernel is the "best" one?
- for a given T_s what is the minimum risetime safe from interpolation noise?
- from the previous lesson:



dCFD simulation: basic principle

- simulate signals having different risetimes (jitter is expected to increase with risetime);

dCFD simulation: basic principle

- simulate signals having different risetimes (jitter is expected to increase with risetime);
- signal are sampled asynchronously with respect to the signal itself (i.e. for each event the sampling comb is translated rigidly, keeping the T_s separation between samples);

dCFD simulation: basic principle

- simulate signals having different risetimes (jitter is expected to increase with risetime);
- signal are sampled asynchronously with respect to the signal itself (i.e. for each event the sampling comb is translated rigidly, keeping the T_s separation between samples);
- sampling comb shift extracted from uniform distribution in $(-T_s/2,\ T_s/2)$;

dCFD simulation: basic principle

- simulate signals having different risetimes (jitter is expected to increase with risetime);
- signal are sampled asynchronously with respect to the signal itself (i.e. for each event the sampling comb is translated rigidly, keeping the T_s separation between samples);
- sampling comb shift extracted from uniform distribution in $(-T_s/2,\ T_s/2)$;
- same procedure employed for simulation of interpolation noise;

dCFD simulation: basic principle

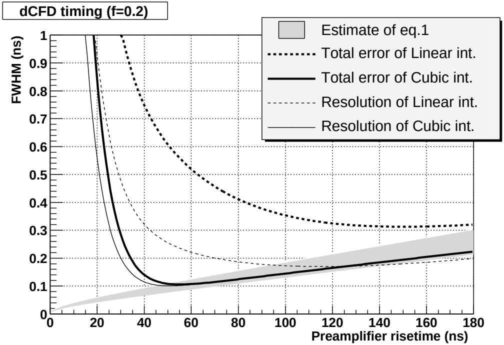
- simulate signals having different risetimes (jitter is expected to increase with risetime);
- signal are sampled asynchronously with respect to the signal itself (i.e. for each event the sampling comb is translated rigidly, keeping the T_s separation between samples);
- sampling comb shift extracted from uniform distribution in $(-T_s/2,\ T_s/2)$;
- same procedure employed for simulation of interpolation noise;
- random noise added to each signal (noise standard deviation constant for all signals);

dCFD simulation: basic principle

- simulate signals having different risetimes (jitter is expected to increase with risetime);
- signal are sampled asynchronously with respect to the signal itself (i.e. for each event the sampling comb is translated rigidly, keeping the T_s separation between samples);
- sampling comb shift extracted from uniform distribution in $(-T_s/2,\ T_s/2)$;
- same procedure employed for simulation of interpolation noise;
- random noise added to each signal (noise standard deviation constant for all signals);
- noise variance and spectrum depends on two contributions: the simulated front-end electronics bandwidth (σ_e in eq. (4)) and the simulated ADC noise, derived from ENOB ($\sigma_q = \frac{1}{\sqrt{12} \cdot 2^{\mathrm{ENOB}}}$ in eq. (4)).

dCFD simulation: 12 bit, 10.8 ENOB, 100 MHz ADC

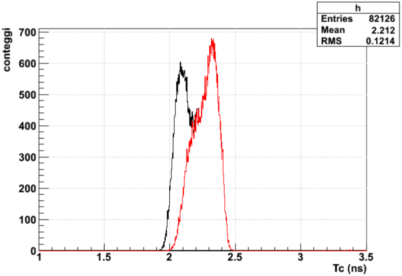
- FWHM of t_x spectrum vs signal risetime [Bardelli2004]:



- cubic interpolation much better than linear: min{FWHM}=100 ps!
- $t = 0$ known $\Longrightarrow$ fluctuations due to t_x determination only
- $t_{rise} > 60\,\text{ns} \Longrightarrow \text{FWHM} \propto t_r$ (SNR constant!) (cfr. eq. (3));
- FWHM increases rapidly as risetime decreases under 60 ns.

Interpolation artifacts: double coincidence peak

- When interpolation dominates resolution strange artifacts appear;
- Example: experimental data (time coincidence between two Si detectors exposed to diffused UV pulsed laser) [Pastore2013]:



- rise-time less than $4T_s$; cubic interp. (4 consecutive samples)
- coincidence peak not gaussian; left peak: signals for which first (out of 4) interpolation node (sample) lies on baseline; right peak: signals for which first node already above baseline.

Questions about ADC's

- fast signals (characteristic times $\leq 3 \div 4\ T_s$): interpolation affects FWHM;
- the faster the ADC the better? buy the ADCs with highest F_s?
- remember ENOB? lower ENOB $\Longrightarrow$ more time jitter;
- in real ADCS, high ENOB and high F_s are conflicting requirements.

Timing measurement: simulation of different ADC's

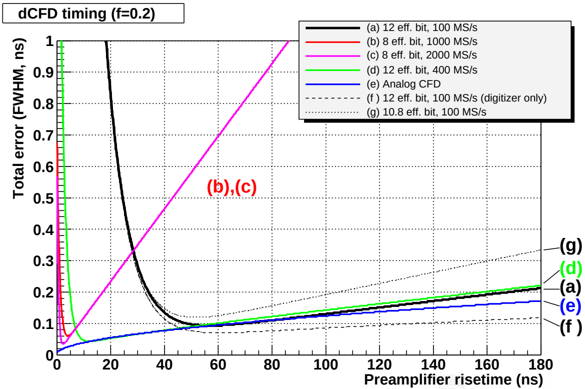


Figure 9: Time resolution (FWHM) for different ENOB/F_s combinations vs charge preamp risetime [Bardelli2004]. Cubic interpolation used.

high sampling rate and high ENOB: conflicting requirements

Timing measurement: simulation of different ADC's

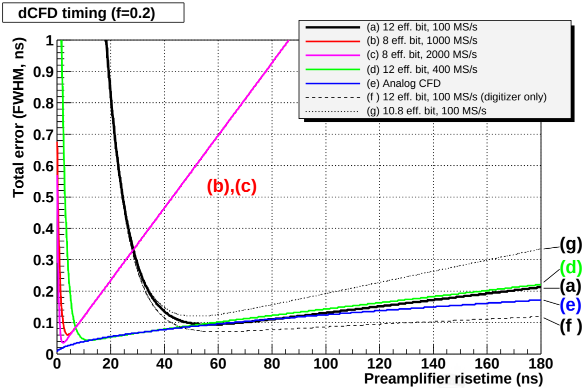


Figure 9: Time resolution (FWHM) for different ENOB/F_s combinations vs charge preamp risetime [Bardelli2004]. Cubic interpolation used.

we use the analog CFD curve (curve e), in blue) as reference

Timing measurement: simulation of different ADC's

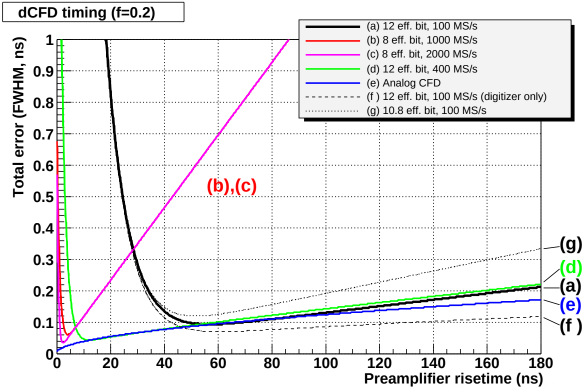


Figure 9: Time resolution (FWHM) for different ENOB/F_s combinations vs charge preamp risetime [Bardelli2004]. Cubic interpolation used.

risetime >60ns: ENOB= 12 (a, d, f) ≈ analog CFD at 400 **and** 100 MS/s

Timing measurement: simulation of different ADC's

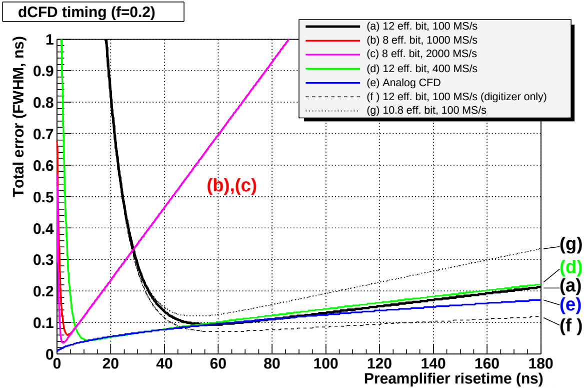


Figure 9: Time resolution (FWHM) for different ENOB/F_s combinations vs charge preamp risetime [Bardelli2004]. Cubic interpolation used.

ENOB$=8$ at 1 GS/s too noisy! far from analog (except for rise-time $\sim 2 \div 3$ ns); worse than 12 ENOB at 100 MS/s for rise-time > 30 ns.

Timing measurement: simulation of different ADC's

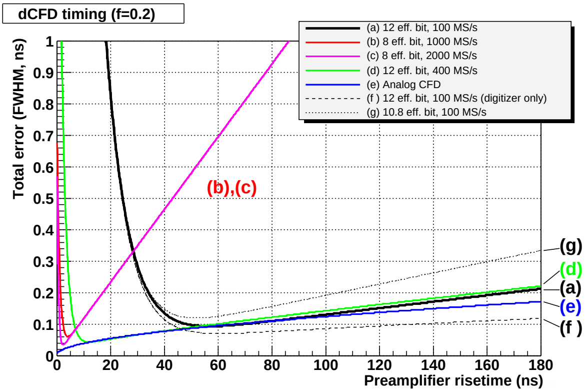


Figure 9: Time resolution (FWHM) for different ENOB/F_s combinations vs charge preamp risetime [Bardelli2004]. Cubic interpolation used.

risetime $\approx 60\,\mathrm{ns}$: even ENOB=10.8 100 MS/s comes close to analog ENOB= 12, $T_s = 10\,\mathrm{ns} \Longrightarrow \min\{\mathrm{FWHM}\} = 100 \div 200\,\mathrm{ps}!$

Timing measurement: simulation of different ADC's

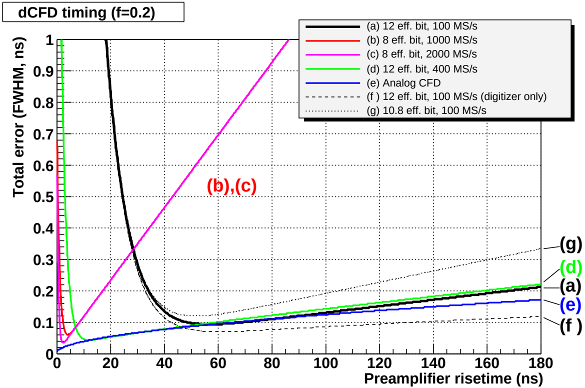


Figure 9: Time resolution (FWHM) for different ENOB/F_s combinations vs charge preamp risetime [Bardelli2004]. Cubic interpolation used.

risetime <60 ns: at 100 MS/s interpolation dominates! $\Longrightarrow$ $F_s = 100\,\mathrm{MS/s}$ not enough

Timing measurement: simulation of different ADC's

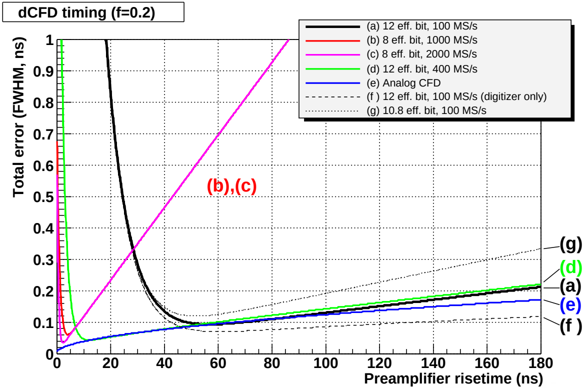
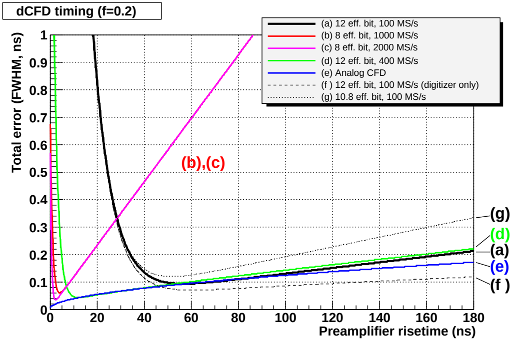


Figure 9: Time resolution (FWHM) for different ENOB/F_s combinations vs charge preamp risetime [Bardelli2004]. Cubic interpolation used.

N.B. (ENOB$=$ 12 F_s $=400\,\mathrm{MS/s}$) better than (ENOB$=$ 8 F_s $= 1 \div 2\,\mathrm{GHz}$) down to risetime$=$ 7 ns!

Final message



enough samples on front (about 4÷5)

$\Longrightarrow$ better high ENOB than high F_s

Time resolution and PSD in Si detectors

- FAZIA (Four π A Z Identification Array) collaboration;
- charge (Z) id of nuclei stopped in 300 μm thick Si;

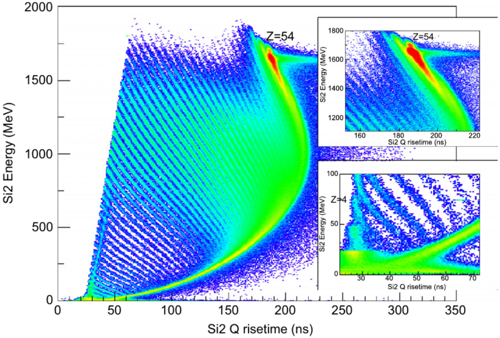
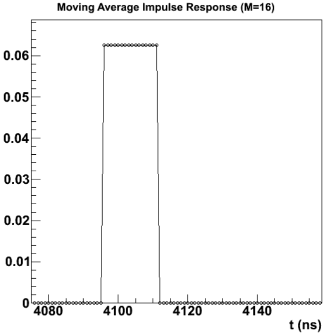


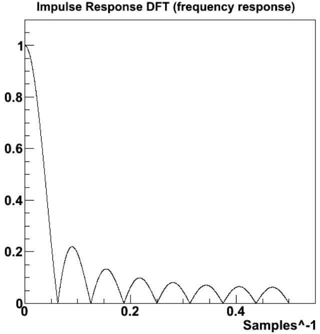
Figure 10: "Si-Energy vs Charge rise-time" (from [Carboni2012]).

- elements from Z=2 to Z=54 are resolved;
- risetimes from 20 to 220 ns $\Longrightarrow$ Z id possible thanks to $\approx$100 ps resolution. (ADC is 14 bit, 100 MS/s, digitizer ENOB=11.2);

Moving average, a simple Low Pass filter

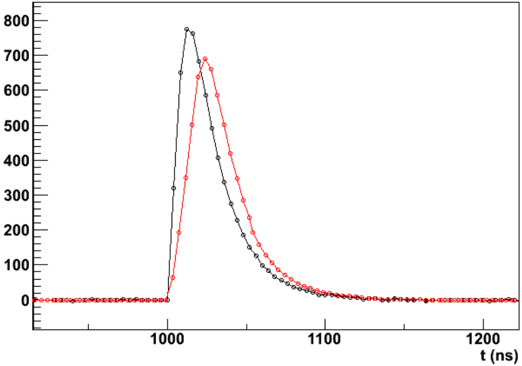
Causal mov. average of M samples from $x[n-M+1]$ to $x[n]$
Convolution: $y[n] = \frac{1}{M}\sum_{i=0}^{M-1} x[n-i]$
Also recursion works: $y[n] = y[n-1] + \frac{1}{M}(x[n] - x[n-M])$
Frequency response: Low Pass Filter





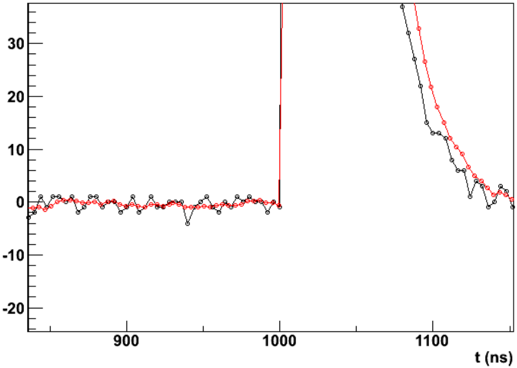
Moving average, a simple Low Pass filter

Effect of moving average on a detector pulse. The processed signal is in red. Transients are slowed down (low-pass!).



Moving average, a simple Low Pass filter

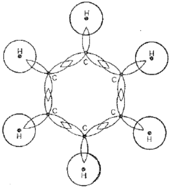
The same picture expanded to show how the noise on the baseline is reduced by the moving average.



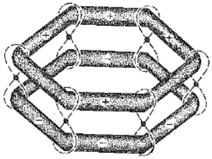
Application to n/γ PSD

n/γ PSD: introduction

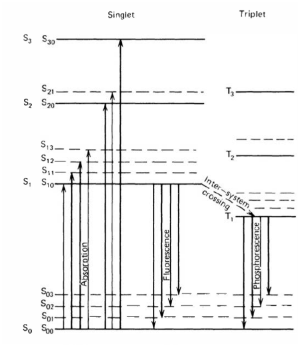
- liquid organic scintill. (e.g. BC501), cyclic aromatic compounds



The σ-hybrid orbitals of the carbon atoms of benzene (Coulson, 1952).



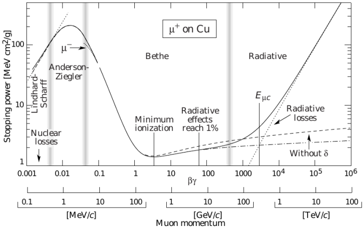
The π-molecular orbitals in benzene (Coulson, 1952).



- scintillation emitted by excited molecules featuring π level structure
- emission involving only singlet states $\Longrightarrow$ shorter emission time
- emission through triplet states $\Longrightarrow$ longer emission time

n/γ PSD: introduction

- density of triplet states along particle track affects overall emission time
- remind: γ must transfer energy to an electron, neutron to a proton
- density of triplet states greater where greater specific energy loss:



- take 1 MeV kinetic energy: then $(\beta\gamma)_{electron} = 2.8$ and $(\beta\gamma)_{proton} = 4.5\ 10^{-2}$
- much higher density for p $\Longrightarrow$ longer emission time ("tail" in signal).

PSD with charge comparison 1

- two integrations, usually *slow* (a.k.a. *tail*) and *total*

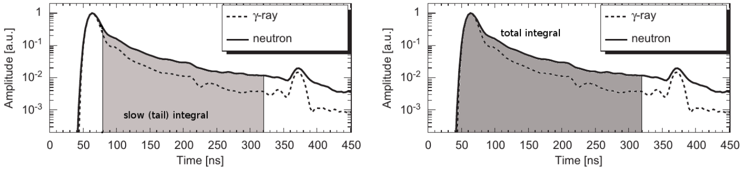


Figure 11: Slow and total integral (adapted from [Söderstrom2008]).

PSD with charge comparison 1

- two integrations, usually *slow* (a.k.a. *tail*) and *total*

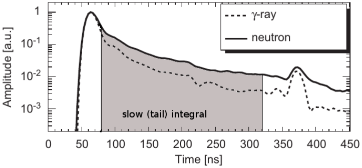
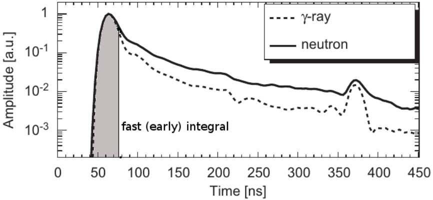


Figure 11: Slow and total integral (adapted from [Söderstrom2008]).

- sometimes *fast* (a.k.a. *early*) and *total*



PSD with charge comparison 2

- most used PSD method (see ref. table at the end);

PSD with charge comparison 2

- most used PSD method (see ref. table at the end);
- baseline statistical uncertainty: keep it below other causes of uncertainty (use enough samples for average), see [Bardelli2006].

PSD with charge comparison 2

- most used PSD method (see ref. table at the end);
- baseline statistical uncertainty: keep it below other causes of uncertainty (use enough samples for average), see [Bardelli2006].
- interpolation: from what we have learnt, we can exploit it:

PSD with charge comparison 2

- most used PSD method (see ref. table at the end);
- baseline statistical uncertainty: keep it below other causes of uncertainty (use enough samples for average), see [Bardelli2006].
- interpolation: from what we have learnt, we can exploit it:
  1. to determine the time mark reference for integral start (either with a LED or CFD algorithm or using interpolation to find “real” maximum);

PSD with charge comparison 2

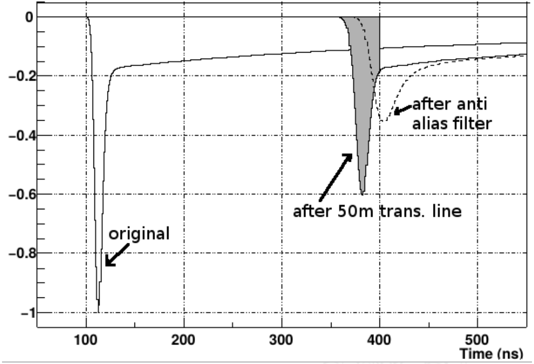
- most used PSD method (see ref. table at the end);
- baseline statistical uncertainty: keep it below other causes of uncertainty (use enough samples for average), see [Bardelli2006].
- interpolation: from what we have learnt, we can exploit it:
  1. to determine the time mark reference for integral start (either with a LED or CFD algorithm or using interpolation to find “real” maximum);
  2. to evaluate integrals starting/ending “in between samples” (most often previous point will take you in between);

PSD with charge comparison 2

- most used PSD method (see ref. table at the end);
- baseline statistical uncertainty: keep it below other causes of uncertainty (use enough samples for average), see [Bardelli2006].
- interpolation: from what we have learnt, we can exploit it:
  1. to determine the time mark reference for integral start (either with a LED or CFD algorithm or using interpolation to find "real" maximum);
  2. to evaluate integrals starting/ending "in between samples" (most often previous point will take you in between);
- really consider 2) if $\Delta t \approx T_s$ ($\Delta t \gg T_s$: it is OK to just sum samples);

PSD with charge comparison 3

- antialiasing filter could slow down first part $\Longrightarrow$ increase Δt of *fast* with respect to analog FEE (part of fig.1 in [Bardelli2002]);



PSD with charge comparison 4

- to minimize ADC noise fluctuations, *fast* (shorter) could be better than *slow* (longer). If $s[i]$ is signal and $n[i]$ is noise

$$\mathrm{V}\left\{\sum_{i=1}^{M}(s[i] + n[i])\right\} = \mathrm{V}\left\{\sum_{i=1}^{M} s[i]\right\} + \mathrm{V}\left\{\sum_{i=1}^{M} n[i]\right\} =$$

$$= \mathrm{V}\left\{\sum_{i=1}^{M} s[i]\right\} + M\,\mathrm{V}\left\{n[i]\right\}$$

where $\mathrm{V}\left\{\cdot\right\}$ is variance operator and we assume same noise variance on all samples $\Longrightarrow$ noise contribution $\propto M$;

PSD with charge comparison 4

- more complex weigthing function $w(t)$ than "rectangular gated" integral can be used [Gatti1962, Söderstrom2008]
- the optimal is very close to rectangular anyway:

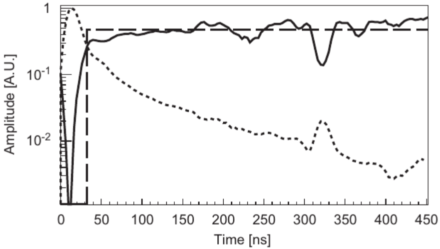


Figure 12: Optimal weigthing function (solid) and rectangular slow integral (dashed). An average neutron signal shape is also shown, from [Söderstrom2008].

PSD: zero crossing and risetime

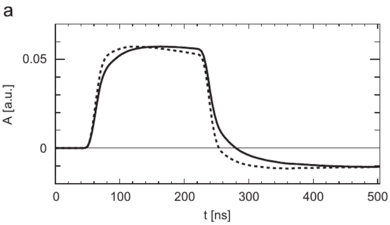


Figure 13: Zero crossing and risetime methods, from [Södestrom2008].

- zero-crossing signal obtained differentiating charge signal (e.g. bipolar DL shaping, usually need also low-pass: mov. average);

PSD: zero crossing and risetime

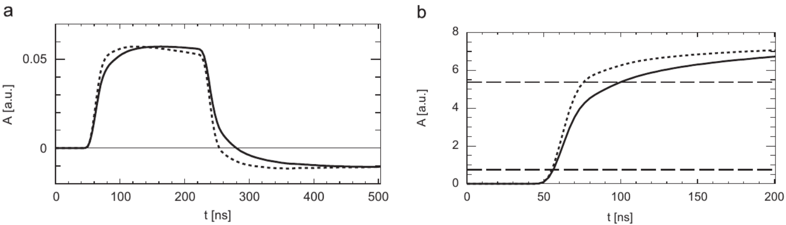


Figure 13: Zero crossing and risetime methods, from [Södestrom2008].

- zero-crossing signal obtained differentiating charge signal (e.g. bipolar DL shaping, usually need also low-pass: mov. average);
- zc-time: time from signal start to zero crossing of bipolar;

PSD: zero crossing and risetime

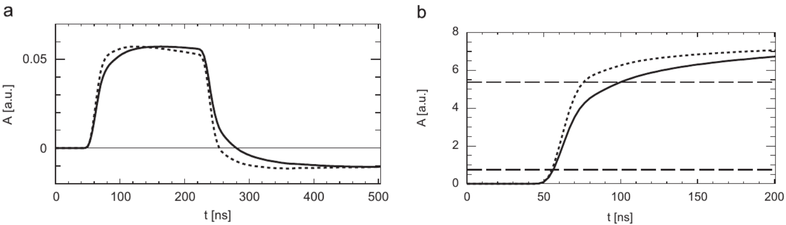


Figure 13: Zero crossing and risetime methods, from [Södestrom2008].

- zero-crossing signal obtained differentiating charge signal (e.g. bipolar DL shaping, usually need also low-pass: mov. average);
- zc-time: time from signal start to zero crossing of bipolar;
- risetime: time for amplitude to go from, e.g., 10% to 90% of maximum;

PSD: zero crossing and risetime

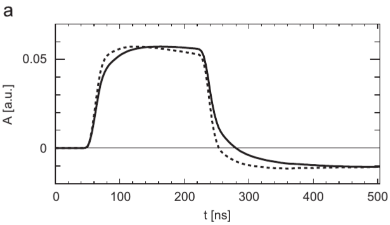


Figure 13: Zero crossing and risetime methods, from [Södestrom2008].

- zero-crossing signal obtained differentiating charge signal (e.g. bipolar DL shaping, usually need also low-pass: mov. average);
- zc-time: time from signal start to zero crossing of bipolar;
- risetime: time for amplitude to go from, e.g., 10% to 90% of maximum;
- strictly related: zc-time $\leftrightarrow$ time of zero derivative $\leftrightarrow$ time of max;

PSD: zero crossing and risetime

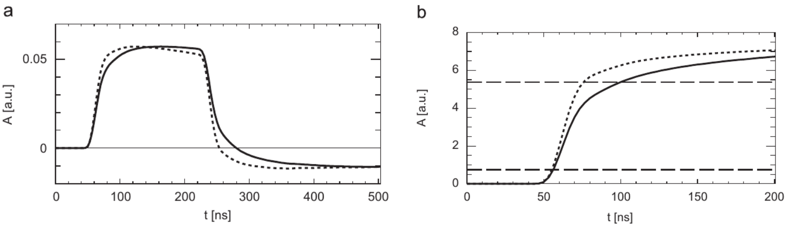


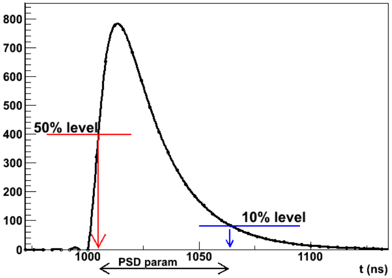
Figure 13: Zero crossing and risetime methods, from [Södestrom2008].

- zero-crossing signal obtained differentiating charge signal (e.g. bipolar DL shaping, usually need also low-pass: mov. average);
- zc-time: time from signal start to zero crossing of bipolar;
- risetime: time for amplitude to go from, e.g., 10% to 90% of maximum;
- strictly related: zc-time $\leftrightarrow$ time of zero derivative $\leftrightarrow$ time of max;
- most used, together with charge comparison;

PSD: Time over Threshold and Q-Risetime

- relevance of interpolation for precise time mark evaluation (both $t = 0$ mark and zero crossing);
- risetime: digital integration+interpolation based dCFD algorithm;
- risetime equivalent: “time over threshold”;

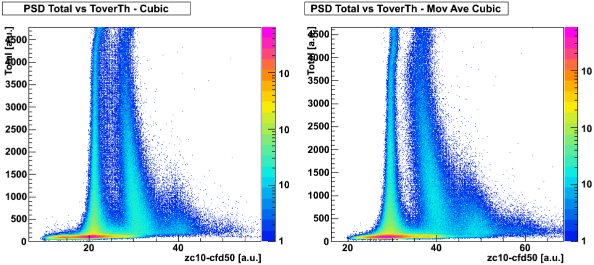
Basic principle of Time over Threshold



PSD: Time over Threshold and Q-Risetime

- relevance of interpolation for precise time mark evaluation (both $t = 0$ mark and zero crossing);
- risetime: digital integration+interpolation based dCFD algorithm;
- risetime equivalent: "time over threshold";

BC501 sampled with 12 bit, 250 MSPS, 10.5 ENOB, Am-Be source, Time Over Threshold

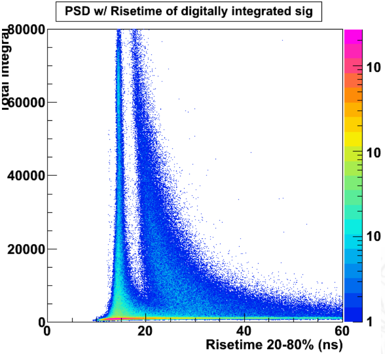


Cubic interpolation: moving average helps getting better separation.

PSD: Time over Threshold and Q-Risetime

- relevance of interpolation for precise time mark evaluation (both $t = 0$ mark and zero crossing);
- risetime: digital integration+interpolation based dCFD algorithm;
- risetime equivalent: "time over threshold";

BC501 12 bit etc., Am-Be source, Charge Risetime



Cubic dCFD at 20 and 80% to get t_{rise} of integrated PMT signal.

PSD with reference shapes (a.k.a. NGMA)

- exploits “reference” shapes;

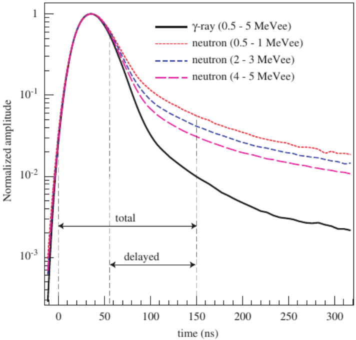


Figure 14: Reference signal shapes, from [Guerrero2008]. Note the energy dependence.

PSD with reference shapes (a.k.a. NGMA)

- exploits "reference" shapes;
- compares digitized signal to reference, a "similarity" parameter is extracted (e.g. $\sum_i (S[i] - \mathcal{S}_{ref}[i])^2$, etc.);

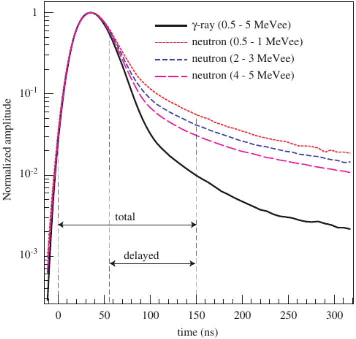


Figure 14: Reference signal shapes, from [Guerrero2008]. Note the energy dependence.

PSD with reference shapes (a.k.a. NGMA)

- exploits "reference" shapes;
- compares digitized signal to reference, a "similarity" parameter is extracted (e.g. $\sum_i (S[i] - \mathcal{S}_{ref}[i])^2$, etc.);
- "most similar" type is assigned;

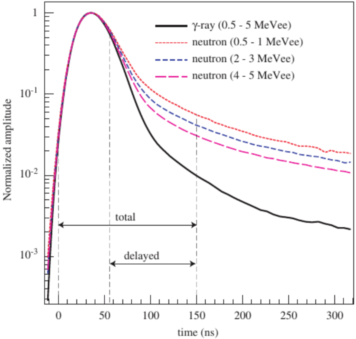


Figure 14: Reference signal shapes, from [Guerrero2008]. Note the energy dependence.

PSD with reference shapes (a.k.a. NGMA)

- exploits "reference" shapes;
- compares digitized signal to reference, a "similarity" parameter is extracted (e.g. $\sum_i (S[i] - \mathcal{S}_{ref}[i])^2$, etc.);
- "most similar" type is assigned;
- reference shapes: averages over thousands of digitized signals

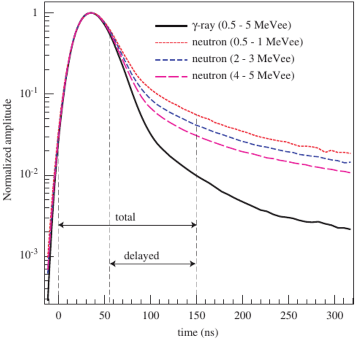


Figure 14: Reference signal shapes, from [Guerrero2008]. Note the energy dependence.

PSD with reference shapes (a.k.a. NGMA)

- exploits "reference" shapes;
- compares digitized signal to reference, a "similarity" parameter is extracted (e.g. $\sum_i (S[i] - \mathcal{S}_{ref}[i])^2$, etc.);
- "most similar" type is assigned;
- reference shapes: averages over thousands of digitized signals
- asynchronous sampling clock $\Longrightarrow$ carefully align shapes before averaging

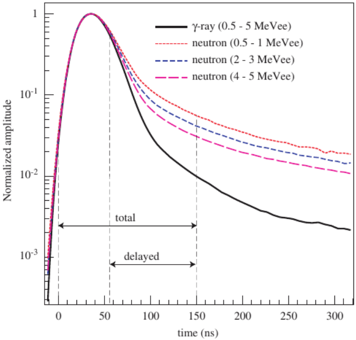


Figure 14: Reference signal shapes, from [Guerrero2008]. Note the energy dependence.

PSD with reference shapes (a.k.a. NGMA)

- exploits "reference" shapes;
- compares digitized signal to reference, a "similarity" parameter is extracted (e.g. $\sum_i (S[i] - \mathcal{S}_{ref}[i])^2$, etc.);
- "most similar" type is assigned;
- reference shapes: averages over thousands of digitized signals
- asynchronous sampling clock $\Longrightarrow$ carefully align shapes before averaging
- interpolation can help:

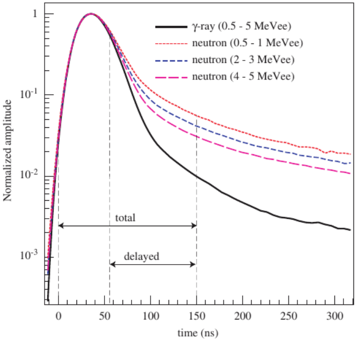


Figure 14: Reference signal shapes, from [Guerrero2008]. Note the energy dependence.

PSD with reference shapes (a.k.a. NGMA)

- exploits "reference" shapes;
- compares digitized signal to reference, a "similarity" parameter is extracted (e.g. $\sum_i (S[i] - \mathcal{S}_{ref}[i])^2$, etc.);
- "most similar" type is assigned;
- reference shapes: averages over thousands of digitized signals
- asynchronous sampling clock $\Longrightarrow$ carefully align shapes before averaging
- interpolation can help:
  1. evaluating real start of the signal (dCFD);

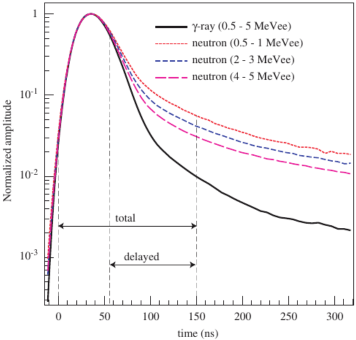


Figure 14: Reference signal shapes, from [Guerrero2008]. Note the energy dependence.

PSD with reference shapes (a.k.a. NGMA)

- exploits "reference" shapes;
- compares digitized signal to reference, a "similarity" parameter is extracted (e.g. $\sum_i (S[i] - \mathcal{S}_{ref}[i])^2$, etc.);
- "most similar" type is assigned;
- reference shapes: averages over thousands of digitized signals
- asynchronous sampling clock $\Longrightarrow$ carefully align shapes before averaging
- interpolation can help:
  1. evaluating real start of the signal (dCFD);
  2. calculating samples "in between" $\Longrightarrow$ "oversampled" shapes can be aligned with better precision;

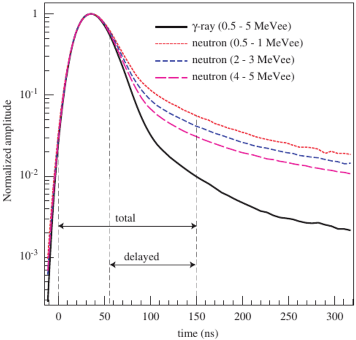
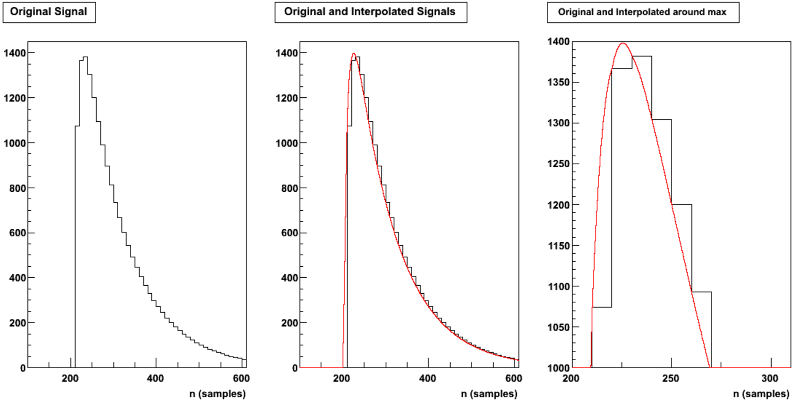


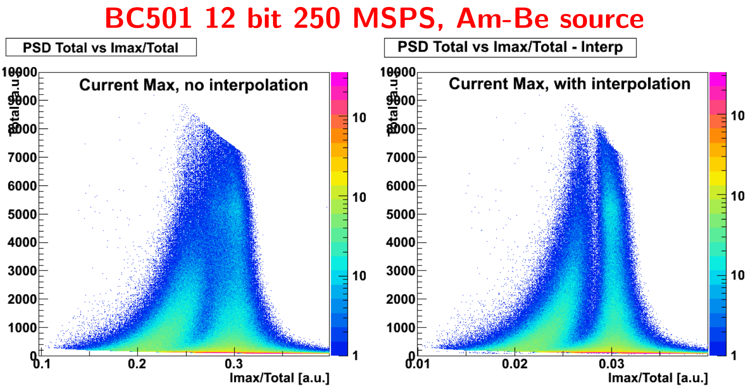
Figure 14: Reference signal shapes, from [Guerrero2008]. Note the energy dependence.

PSD: current maximum

Maximum of current signal (at a given energy) depends on signal duration. In [Cavallaro2013] it is implemented with analog electronics. Digital signals: interpolation critical to get real maximum!



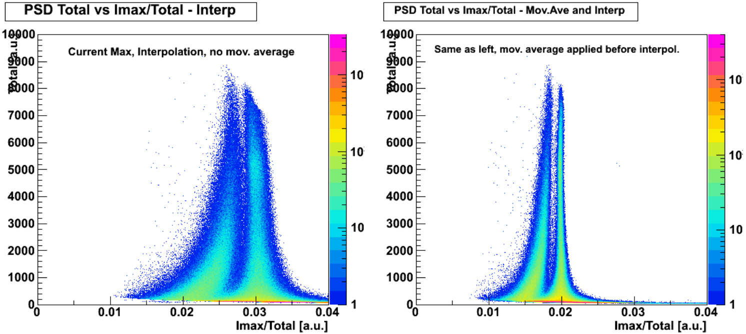
PSD w/ BC501: Current Maximum



Comparing left to right: beneficial effect of interpolation (Imax).

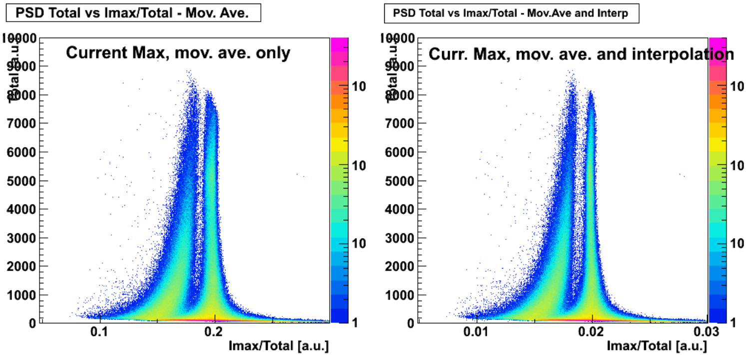
PSD w/ BC501: Current Maximum

BC501 12 bit 250 MSPS, Am-Be source



Comparing left to right: beneficial effect of moving average w/ interp.

PSD w/ BC501: Current Maximum



Comparing left to right: with mov. ave. you get "almost" there...

PSD: Pulse Gradient Analysis [D'Mellow2007]

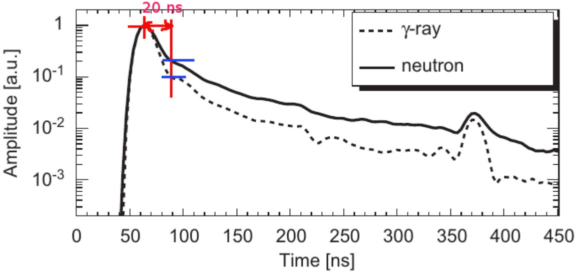


Figure 15: Principle of PGA according to [D'Mellow2007] (picture taken from [Söderstrom2008]).

- normalized shape; PSD param.=amplitude at Δt after max;
- interpolation: both peak determination and amplitude after Δt;
- "smoothing" needed to reduce noise/fluctuations (method relies on a single amplitude, there is no intrinsic averaging).

Selected n-γ PSD literature (1)

	Scint.	Analog	Digital	ADC	Ref.
Adams1978	NE213	CC			NIM 156(1978)459
Alexander1961	NE213, UGLLT	ZC			NIM 13(1961)244
Ambers2011	EJ-309		CC+NGMA	12bit/250MHz	NIM A638(2011)116
Barnabà1998	BC501A	ZC			NIM A410(1998)220
Bell1981	NE213	CC			NIM 188(1981)105
Cao1988	NE213	ZC			NIM A416(1988)32
Cavallaro2013	NE213	IMAX			NIM A700(2013)65
Ĉerny2004	BC501	CC			NIM A527(2004)512
Cester2013	EJ-309		CC	10bit/1GHz	NIM A719(2013)81
Cester2014	EJ-299-33		CC	12bit/250MHz	NIM A735(2014)202
D'Mellow2007	EJ301		CC, PGA	10bit/250MHz	NIM A578(2007)191
Esposito2004	stil, NE213		CC	12bit/200MHz	NIM A518(2004)626
Flaska2007	BC-501A		CC	8bit/5GHz	NIM A577(2007)654
Flaska2009	BC-253A		CC	12bit/250MHz	NIM A599(2009)221
Flaska2013	EJ-309		CC	10÷14bit/0.25÷2GHz	NIM A729(2013)456
Gamage2011	BC501A		PGA,CC,NGMA,SD	12bit/500MHz	NIM A642(2011)78
Guerrero2008	BC501A		NGMA	8bit/1GHz	NIM A597(2008)212
Hawkes2013	cust. plast.		shape study	8bit/2.5GHz	NIM A729(2013)522
Hellesen2013	BC400, NE213		CC	12bit/2GHz	NIM A720(2013)135
Heltsley1988	NE213	CC			NIM A263(1988)441
Kaplan2013	EJ309		CC	12bit/250MHz	NIM A729(2013)463
Kaschuck2005	ant,stil,NE213		CC	12bit/200MHz	NIM A551(2005)420

Selected n-γ PSD literature (2)

	Scint.	Analog	Digital	ADC	Ref.
Kalyna1970	NE213	ZC			NIM 88(1970)277
Jastaniah2002	BC523A		RT,ToT	8bit/500MHz	NIM A517(2004)202
Jhingan2008	BC501	CC			NIM A585(2008)165
Pai1989	NE213	ZC			NIM A278(1989)749
Pawelzak2013	EJ309		CC	12bit/200MHz	NIM A711(2013)21
Savran2010	BC501A		CC, NGMA	12bit/500MHz	NIM A624(2010)675
Söderstrom2008	BC501	ZC	WCC, ZC, CC	14bit/100MHz	NIM A594(2008)79
Wolski1995	BC501A	ZC, CC			NIM A360(1995)584
Nakhostin2010	NE213		ZC	8bit/1GHz	NIM A621(2010)498
Roush1964	NE213	ZC			NIM 31(1964)112
Söderstrom2008	BC501	ZC	ZC, CC	14bit/100MHz	NIM A594(2008)79
Stevanato2012	LaBr(Ce)	CC	CC	12bit/250MHz	NIM A678(2012)83
Yousefi2009	phoswich for β/γ disc.		wavelets	12bit/100MHz	NIM A599(2009)66
Zaitseva2012	cust. plast.		CC	14bit/200MHz	NIM A668(2012)88

CC=charge comparison
WCC=weigthed charge comparison (see Gatti1962)
ZC=zero crossing
ToT=Time over threshold
NGMA=neutron gamma model analysis (a.k.a true shape)
PGA=pulse gradient analysis
SD=simplified digital charge collection
RT=rise time
IMAX=maximum of current (anode) signal

Summary

- Many advantages: digitizers are going to stay with us;

Summary

- Many advantages: digitizers are going to stay with us;
- Already used on large scale (e.g. γ's: AGATA, GRETINA; HCP: FAZIA, GARFIELD...many more planned/developed)

Summary

- Many advantages: digitizers are going to stay with us;
- Already used on large scale (e.g. γ's: AGATA, GRETINA; HCP: FAZIA, GARFIELD...many more planned/developed)
- Two important things to keep in mind:

Summary

- Many advantages: digitizers are going to stay with us;
- Already used on large scale (e.g. γ's: AGATA, GRETINA; HCP: FAZIA, GARFIELD...many more planned/developed)
- Two important things to keep in mind:
  - Sampling ADCs can add noise to your signal (ENOB)

Summary

- Many advantages: digitizers are going to stay with us;
- Already used on large scale (e.g. γ's: AGATA, GRETINA; HCP: FAZIA, GARFIELD...many more planned/developed)
- Two important things to keep in mind:
  - Sampling ADCs can add noise to your signal (ENOB)
  - Issues related to signal reconstruction (artifacts, interpolation noise, etc.) (F_s)

Summary

- Many advantages: digitizers are going to stay with us;
- Already used on large scale (e.g. γ's: AGATA, GRETINA; HCP: FAZIA, GARFIELD...many more planned/developed)
- Two important things to keep in mind:
  - Sampling ADCs can add noise to your signal (ENOB)
  - Issues related to signal reconstruction (artifacts, interpolation noise, etc.) (F_s)
- Be aware when you are time averaging (e.g. energy estimation) and when instead your info is localized in time and more prone to noise (e.g. timing, some PSD algorithms).

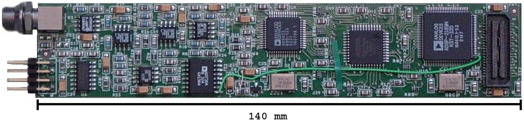
Summary

- Many advantages: digitizers are going to stay with us;
- Already used on large scale (e.g. γ's: AGATA, GRETINA; HCP: FAZIA, GARFIELD...many more planned/developed)
- Two important things to keep in mind:
  - Sampling ADCs can add noise to your signal (ENOB)
  - Issues related to signal reconstruction (artifacts, interpolation noise, etc.) (F_s)
- Be aware when you are time averaging (e.g. energy estimation) and when instead your info is localized in time and more prone to noise (e.g. timing, some PSD algorithms).
- we can learn from $>$ years experience in imaging and telecommunication (cubic convolution, splines, smoothing splines, wavelets,...)... literature is rich on this topic (some papers in the references of these lessons).

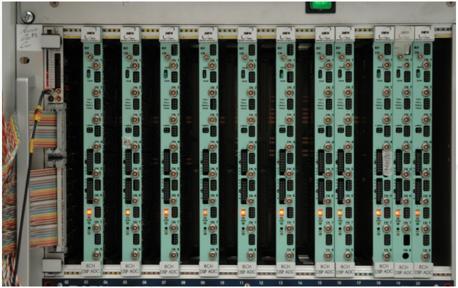
Summary

- Many advantages: digitizers are going to stay with us;
- Already used on large scale (e.g. γ's: AGATA, GRETINA; HCP: FAZIA, GARFIELD...many more planned/developed)
- Two important things to keep in mind:
  - Sampling ADCs can add noise to your signal (ENOB)
  - Issues related to signal reconstruction (artifacts, interpolation noise, etc.) (F_s)
- Be aware when you are time averaging (e.g. energy estimation) and when instead your info is localized in time and more prone to noise (e.g. timing, some PSD algorithms).
- we can learn from > years experience in imaging and telecommunication (cubic convolution, splines, smoothing splines, wavelets,...)... literature is rich on this topic (some papers in the references of these lessons).
- Sometimes better use your human/technical resources (if available!) to design your own digitizer

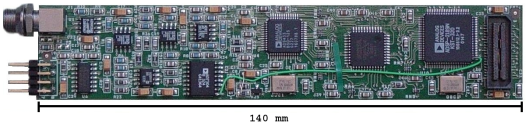
GARFIELD+RCo at LNL: digitizers [Pasquali2007]



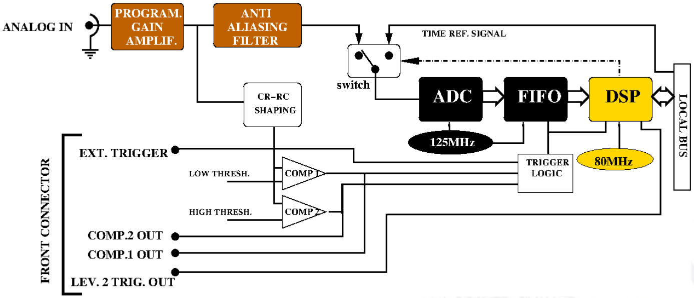
- 1 channel/board
- 12 bit; 125 MSPS
- 9.5 ENOB
- sel. polarity



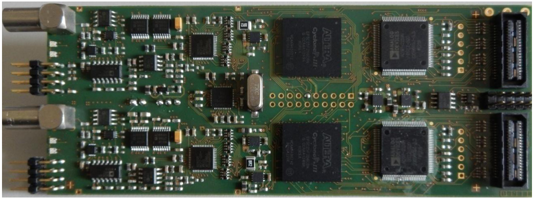
GARFIELD+RCo at LNL: digitizers [Pasquali2007]



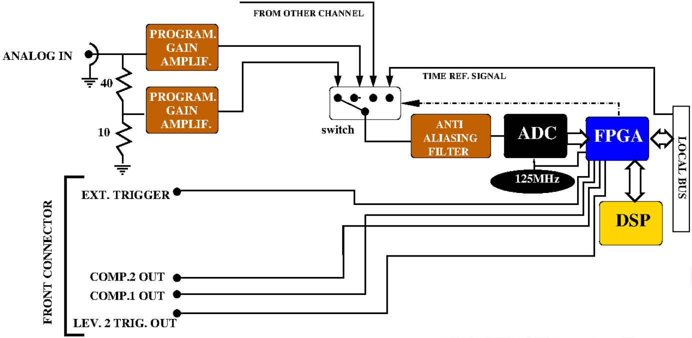
- 1 channel/board
- 12 bit; 125 MSPS
- 9.5 ENOB
- sel. polarity



GARFIELD+RCo at LNL: new digitizers (start 2011)



- 2 channel/board
- 14 bit; 125 MSPS
- 11.5 ENOB
- adj. DC offset



New digitizer

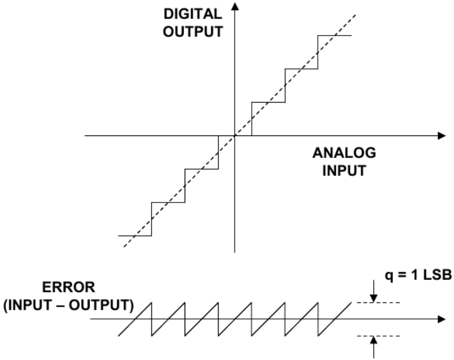
- design: Stefano Meneghini (INFN-Bo), Luigi Bardelli, Maurizio Bini, G.P.
- 14 bit; 125 MSPS;
- two coarse dynamic ranges (better SNR)+ fine gain (12 bit DAC); adjustable range from 100 mV to 10 V
- DC coupled
- adjustable DC offset (polarity selection)
- two channels per board (sampling clocks have opposite phase)
- FPGA centric
- cost: about 300 euros/channel
- DSP: ADSP2189N; FPGA: Altera Cyclone III; Clock gen: AD9572
- VCA: AD8337; ADC: AD9255

Thank you!

Backup slides

Quantization noise: a picture

- comes from <u>second</u> step of A/D conversion (quantization)
- subtract quantized and not-yet-quantized signals:



- the difference is usually correlated to the input for simple signals, e.g. sine (cfr. exercise with pClasses test_quant_noise() in test.C)

Quantization noise: some math

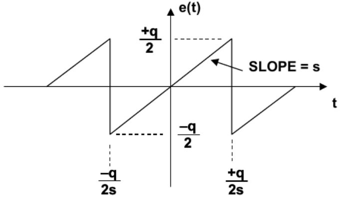
- N-bit ADC $\Longrightarrow 2^{\mathrm{N}}$ possible values $(0 \div 2^{\mathrm{N}} - 1)$;

Quantization noise: some math

- N-bit ADC $\Longrightarrow 2^{\mathrm{N}}$ possible values $(0 \div 2^{\mathrm{N}} - 1)$;
- quantized values $\neq$ "exact values": $e(t) = x_c(t) - \mathcal{Q}\left\{x_c(t)\right\} \neq 0$

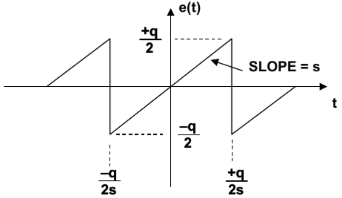
Quantization noise: some math

- N-bit ADC $\Longrightarrow 2^{\mathrm{N}}$ possible values ($0 \div 2^{\mathrm{N}} - 1$);
- quantized values $\neq$ "exact values": $e(t) = x_c(t) - \mathcal{Q}\{x_c(t)\} \neq 0$
- $e(t)$ (quantization error, neglecting sampling) varies with time;



Quantization noise: some math

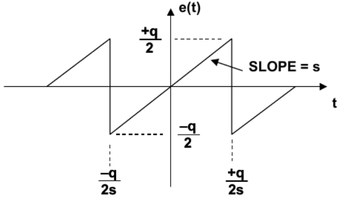
- N-bit ADC $\Longrightarrow 2^{\mathrm{N}}$ possible values ($0 \div 2^{\mathrm{N}} - 1$);
- quantized values $\neq$ "exact values": $e(t) = x_c(t) - \mathcal{Q}\{x_c(t)\} \neq 0$
- $e(t)$ (quantization error, neglecting sampling) varies with time;



- mean square value of e: $\overline{e^2(t)} = \frac{s}{q} \int_{-q/2s}^{+q/2s} (st)^2 \, \mathrm{d}t = \frac{q^2}{12}$

Quantization noise: some math

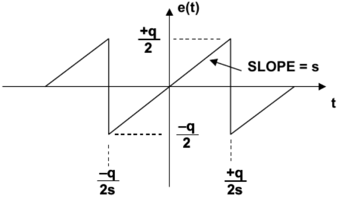
- N-bit ADC $\Longrightarrow 2^{\mathrm{N}}$ possible values $(0 \div 2^{\mathrm{N}} - 1)$;
- quantized values $\neq$ "exact values": $e(t) = x_c(t) - \mathcal{Q}\{x_c(t)\} \neq 0$
- $e(t)$ (quantization error, neglecting sampling) varies with time;



- mean square value of e: $\overline{e^2(t)} = \dfrac{s}{q} \displaystyle\int_{-q/2s}^{+q/2s} (st)^2 \,\mathrm{d}t = \dfrac{q^2}{12}$
- $q = R/2^{\mathrm{N}}$; R=range in Volt (take $R = 2^{\mathrm{N}}$ to get the equivalent in bits);

Quantization noise: some math

- N-bit ADC $\Longrightarrow 2^{\mathrm{N}}$ possible values $(0 \div 2^{\mathrm{N}} - 1)$;
- quantized values $\neq$ "exact values": $e(t) = x_c(t) - \mathcal{Q}\{x_c(t)\} \neq 0$
- $e(t)$ (quantization error, neglecting sampling) varies with time;



- mean square value of e: $\overline{e^2(t)} = \dfrac{s}{q}\int_{-q/2s}^{+q/2s} (st)^2 \,\mathrm{d}t = \dfrac{q^2}{12}$
- $q = R/2^{\mathrm{N}}$; R=range in Volt (take $R = 2^{\mathrm{N}}$ to get the equivalent in bits);
- rms value $\dfrac{q}{\sqrt{12}}$ same as uniform distribution in $(-q/2, q/2)$

Quantization noise: frequency spectrum

- quantized levels "close enough" + complex signal (e.g. speech) $\Longrightarrow$ difference fluctuates randomly from sample to sample [Oppenheim10];

Quantization noise: frequency spectrum

- quantized levels "close enough" + complex signal (e.g. speech) $\Longrightarrow$ difference fluctuates randomly from sample to sample [Oppenheim10];
- also true for simple signals + wide BW noise (detector pulse!);

Quantization noise: frequency spectrum

- quantized levels "close enough" + complex signal (e.g. speech) $\Longrightarrow$ difference fluctuates randomly from sample to sample [Oppenheim10];
- also true for simple signals + wide BW noise (detector pulse!);
- a (almost always) good approximation: constant frequency spectrum (white spectral density) in $(0, \frac{F_s}{2})$

Quantization noise: frequency spectrum

- quantized levels “close enough” + complex signal (e.g. speech) $\Longrightarrow$ difference fluctuates randomly from sample to sample [Oppenheim10];
- also true for simple signals + wide BW noise (detector pulse!);
- a (almost always) good approximation: constant frequency spectrum (white spectral density) in $(0, \frac{F_s}{2})$
- our quant. noise model: “white” noise of variance $\sigma_Q^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$

Quantization noise: frequency spectrum

- quantized levels "close enough" + complex signal (e.g. speech) $\Longrightarrow$ difference fluctuates randomly from sample to sample [Oppenheim10];
- also true for simple signals + wide BW noise (detector pulse!);
- a (almost always) good approximation: constant frequency spectrum (white spectral density) in $(0, \frac{F_s}{2})$
- our quant. noise model: "white" noise of variance
  $$\sigma_Q^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$$
- $\Longrightarrow$ "white" noise of spectral density $w = 2\frac{\sigma_Q^2}{F_s} = \frac{1}{6\,F_s}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$ in $(0, \frac{F_s}{2})$

Ideal ADC: Signal-to-Noise Ratio (SNR)

- SNR: ratio of rms signal amplitude to rms noise amplitude

Ideal ADC: Signal-to-Noise Ratio (SNR)

- SNR: ratio of rms signal amplitude to rms noise amplitude
- Consider a sine wave with $V_{pp} = FS$:

$$SNR = \frac{\overline{v_{sine}^2}}{\overline{v_q^2}} = \frac{FS}{2\sqrt{2}} \frac{\sqrt{12}}{q} = \sqrt{\frac{3}{2}} \frac{FS}{q}$$

Ideal ADC: Signal-to-Noise Ratio (SNR)

- SNR: ratio of rms signal amplitude to rms noise amplitude
- Consider a sine wave with $V_{pp} = FS$:

$$SNR = \frac{\sqrt{\overline{v_{sine}^2}}}{\sqrt{\overline{v_q^2}}} = \frac{FS}{2\sqrt{2}} \frac{\sqrt{12}}{q} = \sqrt{\frac{3}{2}} \frac{FS}{q}$$

- useful to calculate SNR in dB:

$$SNR(dB) = 20\, log_{10}\left(\frac{FS}{q}\right) + 20\, log_{10}\left(\sqrt{\frac{3}{2}}\right) =$$

$$= 20\, log_{10}(2^N) + 1.76 = 6.02N + 1.76$$

Ideal ADC: Signal-to-Noise Ratio (SNR)

- SNR: ratio of rms signal amplitude to rms noise amplitude
- Consider a sine wave with $V_{pp} = FS$:

$$SNR = \frac{\overline{v_{sine}^2}}{\overline{v_q^2}} = \frac{FS}{2\sqrt{2}} \frac{\sqrt{12}}{q} = \sqrt{\frac{3}{2}} \frac{FS}{q}$$

- useful to calculate SNR in dB:

$$SNR(dB) = 20\,log_{10}\left(\frac{FS}{q}\right) + 20\,log_{10}\left(\sqrt{\frac{3}{2}}\right) =$$

$$= 20\,log_{10}(2^N) + 1.76 = 6.02N + 1.76$$

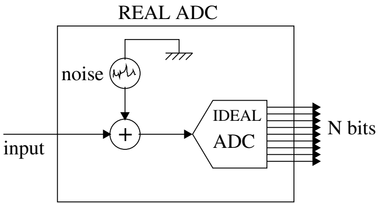
- Some values:

N (bits)	SNR (dB)
10	61.96
12	74.00
14	86.04

as a rule of thumb: 6 dB per bit!

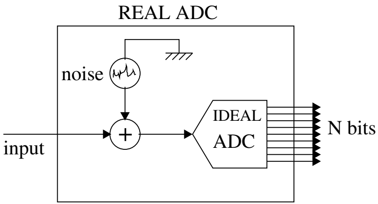
Real ADC and noise...

- we know that a real ADC can be modelled as an "ideal" ADC plus a noise generator adding noise to the input (see figure);



Real ADC and noise...

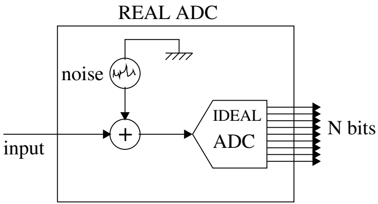
- we know that a real ADC can be modelled as an "ideal" ADC plus a noise generator adding noise to the input (see figure);



- now we can include quantization noise into the generator and assume no need for quantization in the "ideal" ADC;

Real ADC and noise...

- we know that a real ADC can be modelled as an “ideal” ADC plus a noise generator adding noise to the input (see figure);



- now we can include quantization noise into the generator and assume no need for quantization in the “ideal” ADC;
- real ADC noise has variance $\sigma^2_{eff} > \sigma^2_Q$

Actual Noise: SINAD

- to express the actual amount of added noise manufacturers quote SINAD (signal-to-noise-and-distortion) or ENOB (effective-number-of-bits)

Actual Noise: SINAD

- to express the actual amount of added noise manufacturers quote SINAD (signal-to-noise-and-distortion) or ENOB (effective-number-of-bits)
- SINAD: take Fourier Transform of sampled sine wave ($V_{pp} \approx$FS).

Actual Noise: SINAD

- to express the actual amount of added noise manufacturers quote SINAD (signal-to-noise-and-distortion) or ENOB (effective-number-of-bits)
- SINAD: take Fourier Transform of sampled sine wave ($V_{pp} \approx$FS).
- signal power: from spectrum peak at signal frequency.

Actual Noise: SINAD

- to express the actual amount of added noise manufacturers quote SINAD (signal-to-noise-and-distortion) or ENOB (effective-number-of-bits)
- SINAD: take Fourier Transform of sampled sine wave ($V_{pp} \approx$FS).
- signal power: from spectrum peak at signal frequency.
- noise-and-distortion: integral of all other components (harmonics, broadband noise)

Actual Noise: SINAD

- to express the actual amount of added noise manufacturers quote SINAD (signal-to-noise-and-distortion) or ENOB (effective-number-of-bits)
- SINAD: take Fourier Transform of sampled sine wave (V_{pp} $\approx$FS).
- signal power: from spectrum peak at signal frequency.
- noise-and-distortion: integral of all other components (harmonics, broadband noise)
- SINAD takes into account the dynamic (AC) performance

Actual Noise: ENOB

- textbook definition of ENOB: start from ideal SNR

$$SNR(dB) = 20\,log_{10}(2^N) + 1.76 = 6.02N + 1.76$$

Actual Noise: ENOB

- textbook definition of ENOB: start from ideal SNR

$$SNR(dB) = 20\,log_{10}(2^N) + 1.76 = 6.02N + 1.76$$

it's useful to invert it: we get a definition of N

$$N = \frac{SNR - 1.76}{6.02}$$

Actual Noise: ENOB

- textbook definition of ENOB: start from ideal SNR

$$SNR(dB) = 20\,log_{10}(2^N) + 1.76 = 6.02N + 1.76$$

it's useful to invert it: we get a definition of N

$$N = \frac{SNR - 1.76}{6.02}$$

substituting the actual SNR (SINAD) to ideal we obtain "effective" number of bits (ENOB)

$$ENOB \equiv \frac{SINAD - 1.76}{6.02}$$

Actual Noise: ENOB

- textbook definition of ENOB: start from ideal SNR

$$SNR(dB) = 20\, log_{10}(2^N) + 1.76 = 6.02N + 1.76$$

it's useful to invert it: we get a definition of N

$$N = \frac{SNR - 1.76}{6.02}$$

substituting the actual SNR (SINAD) to ideal we obtain "effective" number of bits (ENOB)

$$ENOB \equiv \frac{SINAD - 1.76}{6.02}$$

- $ENOB$: realistic estimate of ADC resolution, $ENOB < N$

Actual Noise: ENOB

- textbook definition of ENOB: start from ideal SNR

$$SNR(dB) = 20\,log_{10}(2^N) + 1.76 = 6.02N + 1.76$$

it's useful to invert it: we get a definition of N

$$N = \frac{SNR - 1.76}{6.02}$$

substituting the actual SNR (SINAD) to ideal we obtain "effective" number of bits (ENOB)

$$ENOB \equiv \frac{SINAD - 1.76}{6.02}$$

- $ENOB$: realistic estimate of ADC resolution, $ENOB < N$
- two ADC's with same $ENOB$ and different N give similar performances

Actual noise: ENOB in nuclear physics

- my definition of ENOB:

Actual noise: ENOB in nuclear physics

- my definition of ENOB:

ENOB is the number you need instead of N in $\sigma_Q^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$

to get σ_{eff}^2, i.e. $\sigma_{eff}^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{ENOB}}}\right)^2$

$$\mathrm{ENOB} = log_2\left(\frac{R}{\sqrt{12}\,\sigma_{eff}}\right)$$

Actual noise: ENOB in nuclear physics

- my definition of ENOB:

ENOB is the number you need instead of N in $\sigma_Q^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$

to get σ_{eff}^2, i.e. $\sigma_{eff}^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{ENOB}}}\right)^2$

$$\mathrm{ENOB} = log_2\left(\frac{R}{\sqrt{12}\,\sigma_{eff}}\right)$$

- the two defs are equivalent if σ_{eff} is dominant contribution to *SINAD* (usually the case in nuclear physics).

Actual noise: ENOB in nuclear physics

- my definition of ENOB:

ENOB is the number you need instead of N in $\sigma_Q^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$

to get σ_{eff}^2, i.e. $\sigma_{eff}^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{ENOB}}}\right)^2$

$$\mathrm{ENOB} = log_2\left(\frac{R}{\sqrt{12}\,\sigma_{eff}}\right)$$

- the two defs are equivalent if σ_{eff} is dominant contribution to *SINAD* (usually the case in nuclear physics).
- properties of ENOB:

Actual noise: ENOB in nuclear physics

- my definition of ENOB:

ENOB is the number you need instead of N in $\sigma_Q^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$

to get σ_{eff}^2, i.e. $\sigma_{eff}^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{ENOB}}}\right)^2$

$$\mathrm{ENOB} = log_2\left(\frac{R}{\sqrt{12}\,\sigma_{eff}}\right)$$

- the two defs are equivalent if σ_{eff} is dominant contribution to *SINAD* (usually the case in nuclear physics).
- properties of ENOB:
  1. doubling σ_{eff} we loose 1 bit (1 unit in ENOB);

Actual noise: ENOB in nuclear physics

- my definition of ENOB:

  ENOB is the number you need instead of N in $\sigma_Q^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$

  to get σ_{eff}^2, i.e. $\sigma_{eff}^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{ENOB}}}\right)^2$

$$\mathrm{ENOB} = log_2\left(\frac{R}{\sqrt{12}\,\sigma_{eff}}\right)$$

- the two defs are equivalent if σ_{eff} is dominant contribution to *SINAD* (usually the case in nuclear physics).
- properties of ENOB:
  1. doubling σ_{eff} we loose 1 bit (1 unit in ENOB);
  2. $\sigma_{eff} \propto \frac{R}{2^{\mathrm{ENOB}}}$

Actual noise: ENOB in nuclear physics

- my definition of ENOB:

ENOB is the number you need instead of N in $\sigma_Q^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{N}}}\right)^2$

to get σ_{eff}^2, i.e. $\sigma_{eff}^2 = \frac{1}{12}\left(\frac{R}{2^{\mathrm{ENOB}}}\right)^2$

$$\mathrm{ENOB} = log_2\left(\frac{R}{\sqrt{12}\,\sigma_{eff}}\right)$$

- the two defs are equivalent if σ_{eff} is dominant contribution to *SINAD* (usually the case in nuclear physics).
- properties of ENOB:
  1. doubling σ_{eff} we loose 1 bit (1 unit in ENOB);
  2. $\sigma_{eff} \propto \frac{R}{2^{\mathrm{ENOB}}}$
  3. in bits ($R = 2^{\mathrm{N}}$) we get $\sigma_{eff} \propto 2^{\mathrm{N-ENOB}} \Longrightarrow \mathrm{N} - \mathrm{ENOB}$ controls how much noise we get;

ENOB: equivalence of the two definitions

$$\sigma^2_{eff} = \frac{1}{12}\left(\frac{R}{2^{\mathrm{ENOB}}}\right)^2 \Longrightarrow \mathrm{ENOB} = \tfrac{1}{2}\, log_2\left(\tfrac{R^2}{12\,\sigma^2_{eff}}\right) \qquad (*)$$

Sine wave, amplitude $R \Longrightarrow V_{rms} = \dfrac{R}{2\sqrt{2}}$

If σ_{eff} only contribution to SNR: $SNR = \dfrac{V_{rms}}{\sigma_{eff}} = \dfrac{R}{2\sqrt{2}\sigma_{eff}}$.

Invert and obtain: $\dfrac{R^2}{\sigma^2_{eff}} = (2\sqrt{2}\,SNR)^2$ and substitute in (*) to find

$$\mathrm{ENOB} = \tfrac{1}{2}\, log_2 \tfrac{(2\sqrt{2}\,SNR)^2}{12} = log_2\, SNR - log_2 \tfrac{\sqrt{12}}{2\sqrt{2}} = log_2\, SNR - log_2 \sqrt{\tfrac{3}{2}}$$

1. multiply and divide by $log_{10}\, 2 = 0.301$, then use log rules to change log base to 10;
2. multiply and divide by 20, so that $20\, log_{10} SNR = SNR(db)$.

$$\mathrm{ENOB} = \tfrac{SNR(dB) - 20 log_{10}\sqrt{1.5}}{20\, log_{10} 2} = \tfrac{SNR(dB) - 1.76}{6.02}$$ Q.E.D.

Shannon: Ideal continuous-discrete time converter

- Ideal C/D converter: $x_c(t) \Longrightarrow x[n] = x_c(n\,T_s)$ (no quantization)

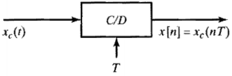


Figure 4.1 Block diagram representation of an ideal continuous-to-discrete-time (C/D) converter.

Shannon: Ideal continuous-discrete time converter

- Ideal C/D converter: $x_c(t) \Longrightarrow x[n] = x_c(n\,T_s)$ (no quantization)

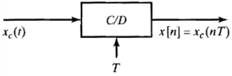


Figure 4.1 Block diagram representation of an ideal continuous-to-discrete-time (C/D) converter.

- let's divide C/D conversion into two steps [Oppenheim10]:

Shannon: Ideal continuous-discrete time converter

- Ideal C/D converter: $x_c(t) \Longrightarrow x[n] = x_c(n\,T_s)$ (no quantization)

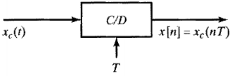


Figure 4.1 Block diagram representation of an ideal continuous-to-discrete-time (C/D) converter.

- let's divide C/D conversion into two steps [Oppenheim10]:
  1. modulation by an impulse train $s(t) = \sum_{n=-\infty}^{+\infty} \delta(t - nT_s) \Longrightarrow$ $x_s(t) = x_c(t)\,s(t) = \sum_{n=-\infty}^{+\infty} x_c(nT_s)\delta(t - nT_s)$

Shannon: Ideal continuous-discrete time converter

- Ideal C/D converter: $x_c(t) \Longrightarrow x[n] = x_c(n\,T_s)$ (no quantization)

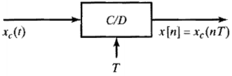


Figure 4.1 Block diagram representation of an ideal continuous-to-discrete-time (C/D) converter.

- let's divide C/D conversion into two steps [Oppenheim10]:
  1. modulation by an impulse train $s(t) = \sum_{n=-\infty}^{+\infty} \delta(t - nT_s) \Longrightarrow$ $x_s(t) = x_c(t)\,s(t) = \sum_{n=-\infty}^{+\infty} x_c(nT_s)\delta(t - nT_s)$
  2. conversion of $x_s(t)$ into $x[n]$ ($x[n]$=area of n-th pulse).

Shannon: Ideal continuous-discrete time converter

- Ideal C/D converter: $x_c(t) \Longrightarrow x[n] = x_c(n\,T_s)$ (no quantization)

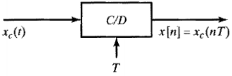
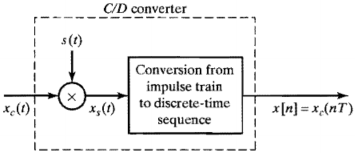


Figure 4.1 Block diagram representation of an ideal continuous-to-discrete-time (C/D) converter.

- let's divide C/D conversion into two steps [Oppenheim10]:
  1. modulation by an impulse train $s(t) = \sum_{n=-\infty}^{+\infty} \delta(t - nT_s) \Longrightarrow$ $x_s(t) = x_c(t)\,s(t) = \sum_{n=-\infty}^{+\infty} x_c(nT_s)\delta(t - nT_s)$
  2. conversion of $x_s(t)$ into $x[n]$ ($x[n]$=area of n-th pulse).



Shannon: signal reconstruction

- $x_s(t)$ is defined also for $t \neq nT_s$ (though it is $= 0$).

Shannon: signal reconstruction

- $x_s(t)$ is defined also for $t \neq nT_s$ (though it is $= 0$).
- Trick: move to frequency domain ($\Omega_s = 2\pi/T_s$) where we calculate the Fourier Transform ($\mathcal{FT}$) of our comb $s(t)$:

$$S(j\Omega) = \mathcal{FT}\{s(t)\} = \frac{2\pi}{T_s} \sum_{k=-\infty}^{+\infty} \delta(\Omega - k\Omega_s)$$

Shannon: signal reconstruction

- $x_s(t)$ is defined also for $t \neq nT_s$ (though it is $= 0$).
- Trick: move to frequency domain ($\Omega_s = 2\pi/T_s$) where we calculate the Fourier Transform ($\mathcal{FT}$) of our comb $s(t)$:

$$S(j\Omega) = \mathcal{FT}\{s(t)\} = \frac{2\pi}{T_s} \sum_{k=-\infty}^{+\infty} \delta(\Omega - k\Omega_s)$$

- well known property of $\mathcal{FT}$: product in t-domain (f-domain) it's equivalent to convolution in f-domain (t-domain): $X_s(j\Omega) = \mathcal{FT}\{x_s(t)\} = \frac{1}{2\pi} X_c(j\Omega) * S(j\Omega) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X_c(\Omega - k\Omega_s)$

Shannon: signal reconstruction

- $x_s(t)$ is defined also for $t \neq nT_s$ (though it is $= 0$).
- Trick: move to frequency domain ($\Omega_s = 2\pi/T_s$) where we calculate the Fourier Transform ($\mathcal{FT}$) of our comb $s(t)$:

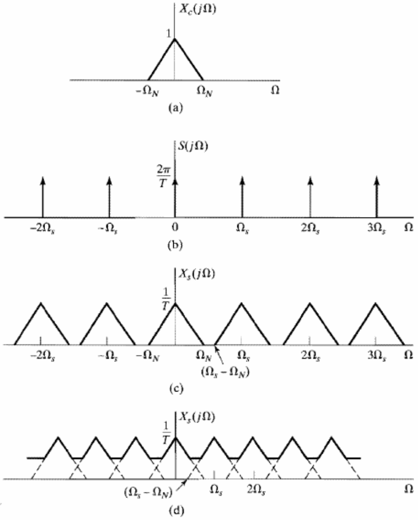
$$S(j\Omega) = \mathcal{FT}\{s(t)\} = \frac{2\pi}{T_s} \sum_{k=-\infty}^{+\infty} \delta(\Omega - k\Omega_s)$$

- well known property of $\mathcal{FT}$: product in t-domain (f-domain) it's equivalent to convolution in f-domain (t-domain):
$X_s(j\Omega) = \mathcal{FT}\{x_s(t)\} = \frac{1}{2\pi} X_c(j\Omega) * S(j\Omega) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X_c(\Omega - k\Omega_s)$
- the $\mathcal{FT}\{x_s(t)\}$ is made of f-shifted images of $\mathcal{FT}\{x_c(t)\}$ (exploiting linearity of convolution and exploiting the result $X_c(j\Omega) * \delta(\Omega - k\Omega_s) = X_c(\Omega - k\Omega_s)$)

Shannon: periodic frequency spectrum

$$X_s(j\Omega) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X_c(\Omega - k\Omega_s)$$

original $X_c(j\Omega)$ plus ∞ copies shifted by $k\Omega_s$

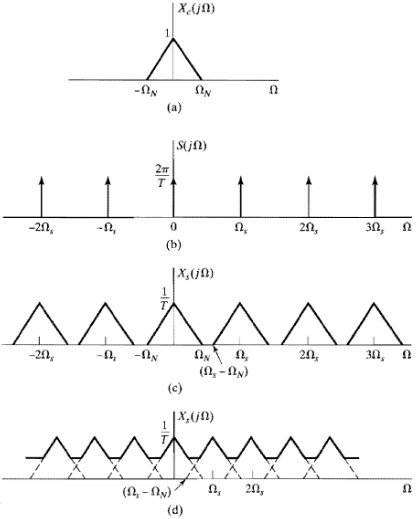


Shannon: periodic frequency spectrum

$$X_s(j\Omega) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X_c(\Omega - k\Omega_s)$$

original $X_c(j\Omega)$ plus ∞ copies shifted by $k\Omega_s$

- To re-construct the original $\mathcal{FT}$: use frequency-selective filter keeping the original and discarding the copies

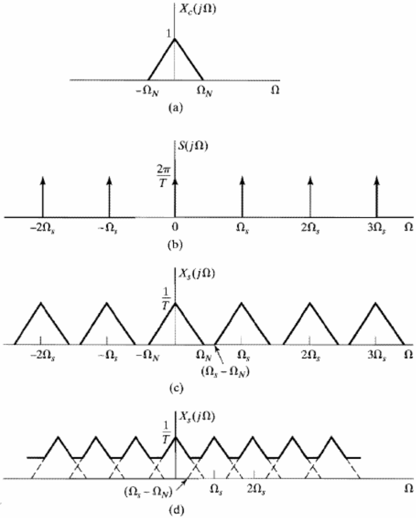


Shannon: periodic frequency spectrum

$$X_s(j\Omega) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X_c(\Omega - k\Omega_s)$$

original $X_c(j\Omega)$ plus ∞ copies shifted by $k\Omega_s$

- To re-construct the original $\mathcal{FT}$: use frequency-selective filter keeping the original and discarding the copies
- use inverse $\mathcal{FT}$ to obtain $x_c(t)$



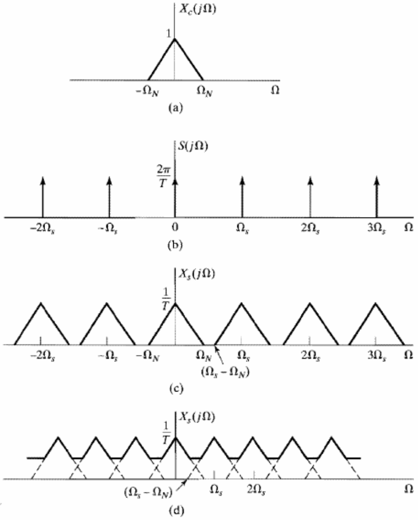
Shannon: periodic frequency spectrum

$$X_s(j\Omega) = \frac{1}{T_s} \sum_{k=-\infty}^{+\infty} X_c(\Omega - k\Omega_s)$$

original $X_c(j\Omega)$ plus ∞ copies shifted by $k\Omega_s$

- To re-construct the original $\mathcal{FT}$: use frequency-selective filter keeping the original and discarding the copies
- use inverse $\mathcal{FT}$ to obtain $x_c(t)$
- copies must NOT overlap $\Longrightarrow$ if Ω_N is maximum frequency in $x_c(t)$ then we want

$$\Omega_s - \Omega_N \geq \Omega_N \implies \Omega_s \geq 2\Omega_N$$



Shannon: filtering out images in f-domain

- t-domain **reconstruction** of limited bandwidth (BW$< F_N$) signal $x(t)$ – sampled at $F_s > 2F_N$ – from samples $x[n] = x(nT_s)$:

Shannon: filtering out images in f-domain

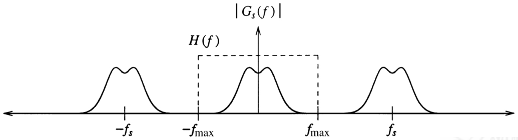
- t-domain **reconstruction** of limited bandwidth (BW$< F_N$) signal $x(t)$ – sampled at $F_s > 2F_N$ – from samples $x[n] = x(nT_s)$:
- First: construct a pseudo-continuous function $x_s(t) = \sum_n x[n]\delta(t - nT_s)$ (pulse train)

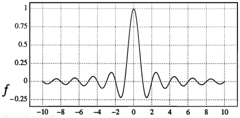
Shannon: filtering out images in f-domain

- t-domain **reconstruction** of limited bandwidth (BW$< F_N$) signal $x(t)$ – sampled at $F_s > 2F_N$ – from samples $x[n] = x(nT_s)$:
- First: construct a pseudo-continuous function $x_s(t) = \sum_n x[n]\delta(t - nT_s)$ (pulse train)
- we know $\mathcal{FT}$ of $x_s(t)$ is made of shifted copies of some $X_r(j\Omega)$, centered at $n\,F_s$ with $F_s = 1/T_s$

Shannon: filtering out images in f-domain

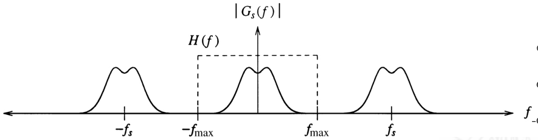
- t-domain **reconstruction** of limited bandwidth (BW$< F_N$) signal $x(t)$ – sampled at $F_s > 2F_N$ – from samples $x[n] = x(nT_s)$:
- First: construct a pseudo-continuous function $x_s(t) = \sum_n x[n]\delta(t - nT_s)$ (pulse train)
- we know $\mathcal{FT}$ of $x_s(t)$ is made of shifted copies of some $X_r(j\Omega)$, centered at $n\,F_s$ with $F_s = 1/T_s$
- first, filter out the extra images in f-domain (those with $n \neq 0$) multiplying $\times$ brick-wall filter response (cut at f_{max})

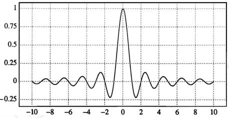




Shannon: filtering out images in f-domain

- t-domain **reconstruction** of limited bandwidth (BW$< F_N$) signal $x(t)$ – sampled at $F_s > 2F_N$ – from samples $x[n] = x(nT_s)$:
- First: construct a pseudo-continuous function $x_s(t) = \sum_n x[n]\delta(t - nT_s)$ (pulse train)
- we know $\mathcal{FT}$ of $x_s(t)$ is made of shifted copies of some $X_r(j\Omega)$, centered at $n F_s$ with $F_s = 1/T_s$
- first, filter out the extra images in f-domain (those with $n \neq 0$) multiplying $\times$ brick-wall filter response (cut at f_{max})





- multiplication in f-domain $\Longrightarrow$ convolution with filter's impulse response (right picture) in t-domain (N.B: $T_s = 1$ in right panel)

Sinc interpolation

- in t-domain, convolution of $x_s(t)$ with $\mathcal{FT}^{-1}$ of brick-wall f-response: $sinc(t) = \frac{sin(\pi t/T_s)}{\pi t/T_s}$ (*normalized sinc*). We assumed a cut-off at $f_{max} = \frac{1}{2}\frac{1}{T_s}$

$$x_r(t) = x_s(t) * sinc(t) = \sum_n x[n] \int_{-\infty}^{+\infty} sinc(x)\delta(t - nT_s - x)dx =$$

$$= \sum_n x[n] sinc\,(t \; - \; nT_s)$$

Sinc interpolation

- in t-domain, convolution of $x_s(t)$ with $\mathcal{FT}^{-1}$ of brick-wall f-response: $sinc(t) = \frac{sin(\pi t/T_s)}{\pi t/T_s}$ (*normalized sinc*). We assumed a cut-off at $f_{max} = \frac{1}{2}\frac{1}{T_s}$

$$x_r(t) = x_s(t) * sinc(t) = \sum_n x[n] \int_{-\infty}^{+\infty} sinc(x)\delta(t - nT_s - x)dx =$$

$$= \sum_n x[n] sinc\left(t - nT_s\right)$$

- interpolation: for $t = mT_s$, $sinc\left(mT_s - nT_s\right) = 0\ \forall m \in \mathbb{Z}$ except $m = n$ where $sinc(0) = 1 \Longrightarrow x(nT_s) = x[n] \Longrightarrow x_r(t)$ goes through known samples; in between we get "interpolated" values

Anti-aliasing stage: general remarks

- antialiasing filter: crucial pre-ADC element! Usually Low-Pass filter (Shannon Theorem!)

Anti-aliasing stage: general remarks

- antialiasing filter: crucial pre-ADC element! Usually Low-Pass filter (Shannon Theorem!)
- REMEMBER: ADC receives the output of the antialias, NOT the input of the digitizer (i.e. the *original* signal)!

Anti-aliasing stage: general remarks

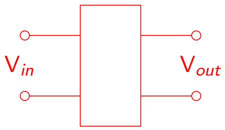
- antialiasing filter: crucial pre-ADC element! Usually Low-Pass filter (Shannon Theorem!)
- REMEMBER: ADC receives the output of the antialias, NOT the input of the digitizer (i.e. the *original* signal)!
- role: to attenuate frequencies beyond $F_s/2$ which would alias into $(0, F_s/2)$

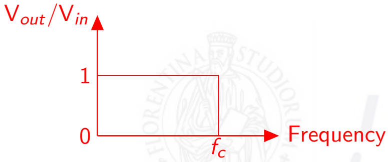
Anti-aliasing stage: general remarks

- antialiasing filter: crucial pre-ADC element! Usually Low-Pass filter (Shannon Theorem!)
- REMEMBER: ADC receives the output of the antialias, NOT the input of the digitizer (i.e. the *original* signal)!
- role: to attenuate frequencies beyond $F_s/2$ which would alias into $(0,\ F_s/2)$
- $\Longrightarrow$ changes signal shape in t-domain (and of course its frequency content)

Anti-aliasing stage: general remarks

- antialiasing filter: crucial pre-ADC element! Usually Low-Pass filter (Shannon Theorem!)
- REMEMBER: ADC receives the output of the antialias, NOT the input of the digitizer (i.e. the *original* signal)!
- role: to attenuate frequencies beyond $F_s/2$ which would alias into $(0, F_s/2)$
- $\Longrightarrow$ changes signal shape in t-domain (and of course its frequency content)
- the ideal antialias has a "brick wall" response cutting at $f_c = F_s/2$





Anti-aliasing stage: general remarks

- a perfect brick wall response not possible in analog circuit...

Anti-aliasing stage: general remarks

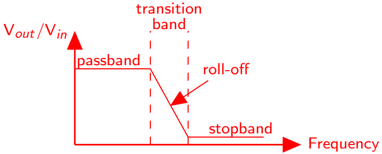
- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...

Anti-aliasing stage: general remarks

- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...

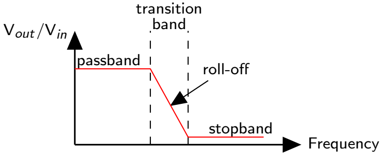
Anti-aliasing stage: general remarks

- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...



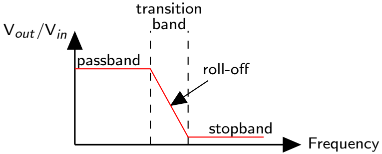
Anti-aliasing stage: general remarks

- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...



Anti-aliasing stage: general remarks

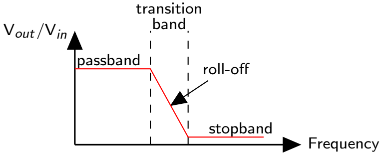
- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...



- passband: f-interval where frequencies are unaltered

Anti-aliasing stage: general remarks

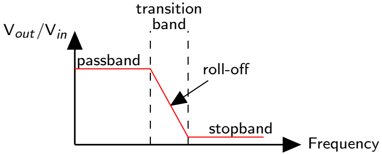
- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...



- passband: f-interval where frequencies are unaltered
- stopband: f-interval where frequencies are blocked

Anti-aliasing stage: general remarks

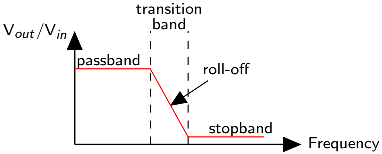
- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...



- passband: f-interval where frequencies are unaltered
- stopband: f-interval where frequencies are blocked
- transition band: between pass and stop bands (starts at the *cutoff frequency*)

Anti-aliasing stage: general remarks

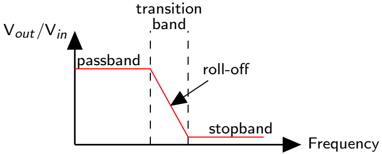
- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...



- passband: f-interval where frequencies are unaltered
- stopband: f-interval where frequencies are blocked
- transition band: between pass and stop bands (starts at the *cutoff frequency*)
- a fast roll-off is desired to separate frequencies (this usually completely spoils t-domain response! you can't have it all)

Anti-aliasing stage: general remarks

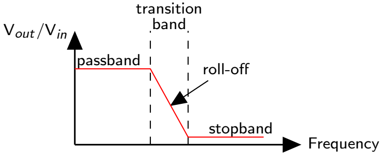
- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...



- passband: f-interval where frequencies are unaltered
- stopband: f-interval where frequencies are blocked
- transition band: between pass and stop bands (starts at the *cutoff frequency*)
- a fast roll-off is desired to separate frequencies (this usually completely spoils t-domain response! you can't have it all)
- constant passband gain desired (no passband *ripple*)

Anti-aliasing stage: general remarks

- a perfect brick wall response not possible in analog circuit...
- ...actual filters have pass-band ripples, transition band not infinitely narrow (roll-off slope is finite), finite attenuation in stop-band...



- passband: f-interval where frequencies are unaltered
- stopband: f-interval where frequencies are blocked
- transition band: between pass and stop bands (starts at the *cutoff frequency*)
- a fast roll-off is desired to separate frequencies (this usually completely spoils t-domain response! you can't have it all)
- constant passband gain desired (no passband *ripple*)

- N.B. attenuation usually not constant in stopband: stopband begins when a certain minimum attenuation is reached

Filter impulse, step and frequency response

- filter has: **impulse**, **step** and **frequency response**

Filter impulse, step and frequency response

- filter has: **impulse**, **step** and **frequency response**
- impulse response: filter output when input is a pulse (Dirac's δ)

Filter impulse, step and frequency response

- filter has: **impulse**, **step** and **frequency response**
- impulse response: filter output when input is a pulse (Dirac's δ)
- step response: filter output when input is a perfect step

Filter impulse, step and frequency response

- filter has: **impulse**, **step** and **frequency response**
- impulse response: filter output when input is a pulse (Dirac's δ)
- step response: filter output when input is a perfect step
- each one contains complete info about the filter

Filter impulse, step and frequency response

- filter has: **impulse**, **step** and **frequency response**
- impulse response: filter output when input is a pulse (Dirac's δ)
- step response: filter output when input is a perfect step
- each one contains complete info about the filter
- frequency response $\Longleftrightarrow$ filter action on f-domain info

Filter impulse, step and frequency response

- filter has: **impulse**, **step** and **frequency response**
- impulse response: filter output when input is a pulse (Dirac's δ)
- step response: filter output when input is a perfect step
- each one contains complete info about the filter
- frequency response $\Longleftrightarrow$ filter action on f-domain info
- step response $\Longleftrightarrow$ filter action on t-domain info

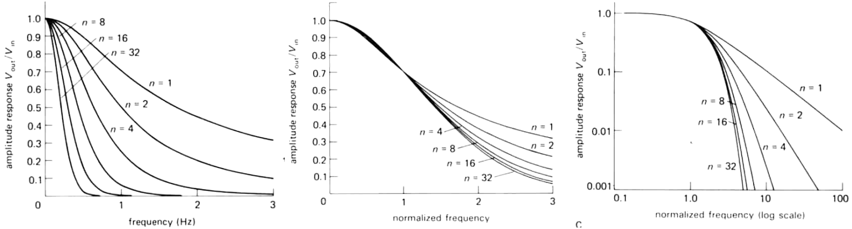
Filter impulse, step and frequency response

- filter has: **impulse**, **step** and **frequency response**
- impulse response: filter output when input is a pulse (Dirac's δ)
- step response: filter output when input is a perfect step
- each one contains complete info about the filter
- frequency response $\Longleftrightarrow$ filter action on f-domain info
- step response $\Longleftrightarrow$ filter action on t-domain info
- time domain coded info: any sample contains some info

Filter impulse, step and frequency response

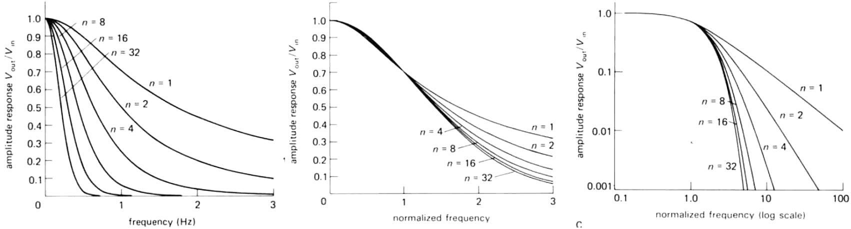
- filter has: **impulse**, **step** and **frequency response**
- impulse response: filter output when input is a pulse (Dirac's δ)
- step response: filter output when input is a perfect step
- each one contains complete info about the filter
- frequency response $\Longleftrightarrow$ filter action on f-domain info
- step response $\Longleftrightarrow$ filter action on t-domain info
- time domain coded info: any sample contains some info
- frequency domain coded info: relationship between many samples (no info in single sample)

Anti-aliasing stage: RC low pass stage [Horowitz1989]



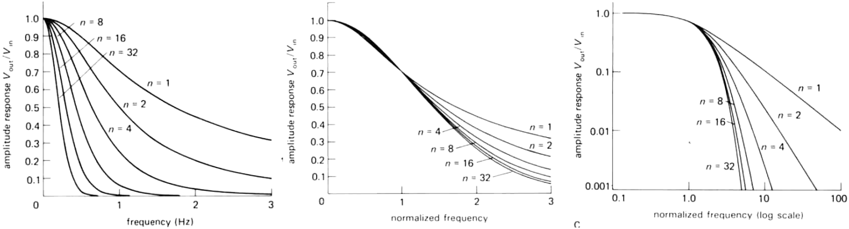
- simple RC low pass $\Longrightarrow$ -3 dB cutoff at $\omega_c = 1/RC$; roll-off = 20 dB/decade (6 db/octave);

Anti-aliasing stage: RC low pass stage [Horowitz1989]



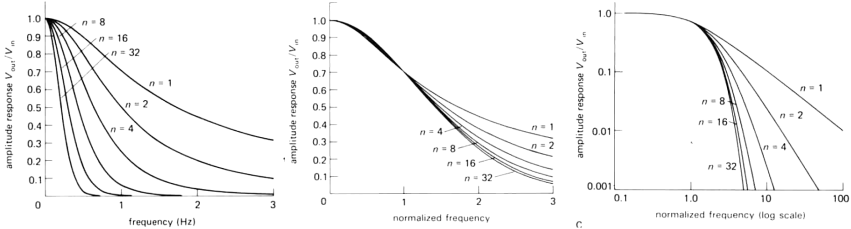
- simple RC low pass $\Longrightarrow$ -3 dB cutoff at $\omega_c = 1/RC$; roll-off = 20 dB/decade (6 db/octave);
- 20 dB $\equiv \times 10 \Longrightarrow$ after two decades gain is 1 % of DC value

Anti-aliasing stage: RC low pass stage [Horowitz1989]



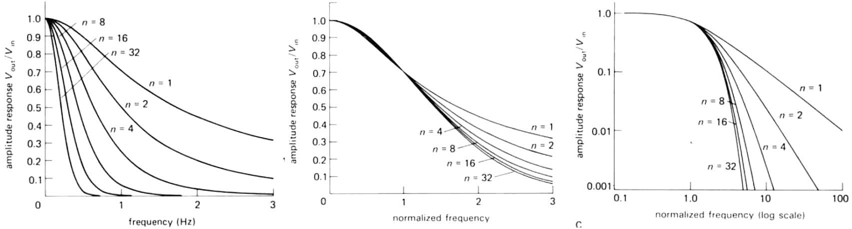
- simple RC low pass $\Longrightarrow$ -3 dB cutoff at $\omega_c = 1/RC$; roll-off $=$ 20 dB/decade (6 db/octave);
- 20 dB $\equiv \times 10 \Longrightarrow$ after two decades gain is 1 % of DC value
- cascading n$\times$RC $\Longrightarrow$ increases slope (n $\times$ 20 dB/decade)

Anti-aliasing stage: RC low pass stage [Horowitz1989]



- simple RC low pass $\Longrightarrow$ -3 dB cutoff at $\omega_c = 1/RC$; roll-off $=$ 20 dB/decade (6 db/octave);
- 20 dB $\equiv \times 10 \Longrightarrow$ after two decades gain is 1 % of DC value
- cascading n$\times$RC $\Longrightarrow$ increases slope (n $\times$ 20 dB/decade)
- n is a.k.a. the number of "poles" (zeros at denominator in the transfer function, cfr. Laplace or Fourier transform)

Anti-aliasing stage: RC low pass stage [Horowitz1989]



- simple RC low pass $\Longrightarrow$ -3 dB cutoff at $\omega_c = 1/RC$; roll-off = 20 dB/decade (6 db/octave);
- 20 dB $\equiv \times 10 \Longrightarrow$ after two decades gain is 1 % of DC value
- cascading n$\times$RC $\Longrightarrow$ increases slope (n $\times$ 20 dB/decade)
- n is a.k.a. the number of "poles" (zeros at denominator in the transfer function, cfr. Laplace or Fourier transform)
- however we don't get a sharper knee at -3dB cutoff: "many soft knees do not a hard knee make" (cit. Horowitz-Hill); this clearly appears when plotting response vs f/f_c (normalized frequency)

Anti-aliasing stage: Butterworth, Chebyshev, Bessel

- solution: active filters using amplifiers and feedback

Anti-aliasing stage: Butterworth, Chebyshev, Bessel

- solution: active filters using amplifiers and feedback
- one discovers that a flat passband response and a fast roll-off are in competion, we must trade in one for the other

Anti-aliasing stage: Butterworth, Chebyshev, Bessel

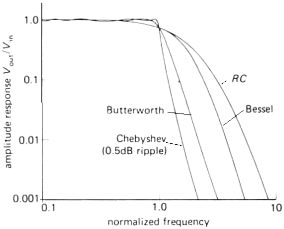
- solution: active filters using amplifiers and feedback
- one discovers that a flat passband response and a fast roll-off are in competion, we must trade in one for the other
- in filter theory multipole filters are classified, according to the compromises they make, as: Chebyshev, Butterworth and Bessel

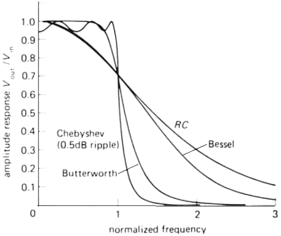
Anti-aliasing stage: Butterworth, Chebyshev, Bessel

- solution: active filters using amplifiers and feedback
- one discovers that a flat passband response and a fast roll-off are in competion, we must trade in one for the other
- in filter theory multipole filters are classified, according to the compromises they make, as: Chebyshev, Butterworth and Bessel
- it doesn't matter the particular circuit used to obtain the response: the name is associated to the response.

Anti-aliasing stage: Butterworth, Chebyshev, Bessel

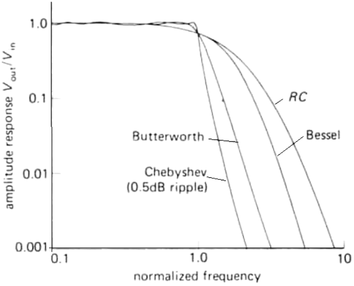
- solution: active filters using amplifiers and feedback
- one discovers that a flat passband response and a fast roll-off are in competion, we must trade in one for the other
- in filter theory multipole filters are classified, according to the compromises they make, as: Chebyshev, Butterworth and Bessel
- it doesn't matter the particular <u>circuit</u> used to obtain the response: the name is associated to the <u>response</u>.
- frequency response for 6-poles active filters [Horowitz1989]

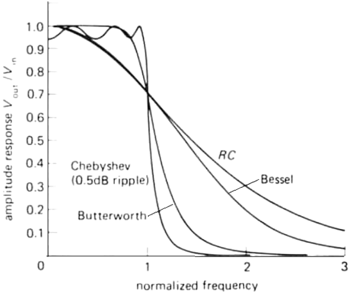




Anti-aliasing stage: f-response

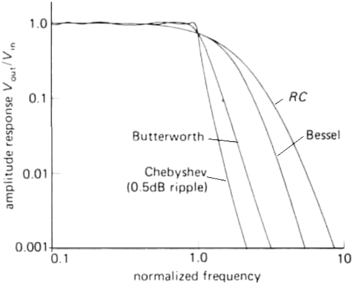
- frequency response for 6-poles active filters

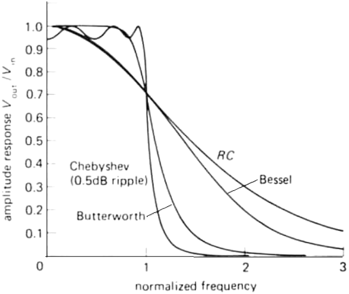




Anti-aliasing stage: f-response

- frequency response for 6-poles active filters

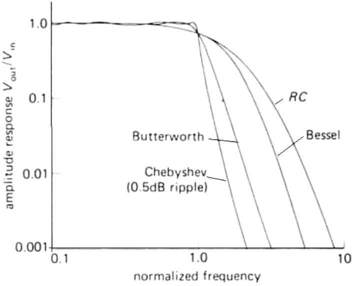


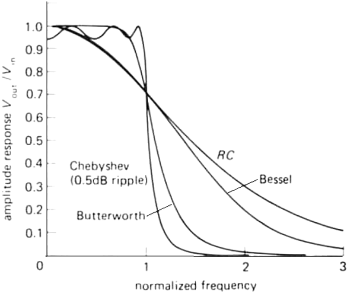


- Butterworth: maximally flat passband response

Anti-aliasing stage: f-response

- frequency response for 6-poles active filters





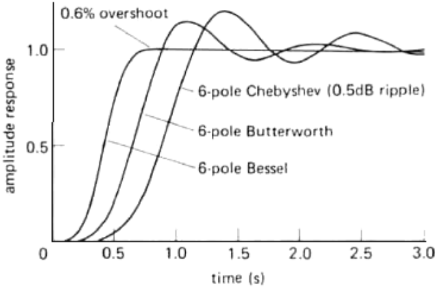
- Butterworth: maximally flat passband response
- Chebyshev: accept some passband ripple to get steeper roll-off

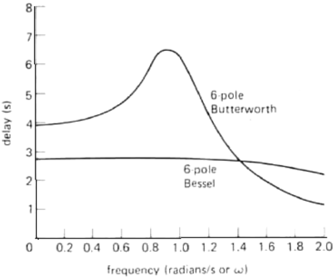
Anti-aliasing stage: t-response

- step response for 6-poles active filters

Anti-aliasing stage: t-response

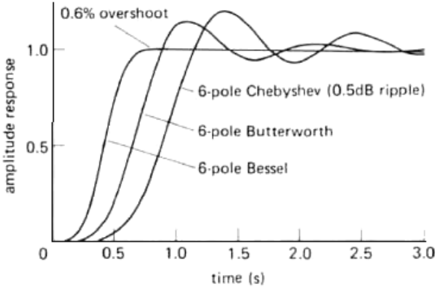
- step response for 6-poles active filters
- Butterworth and Chebyshev: bad step-response (left) due to not constant delay ($\equiv$ non linear phase resp.) (right)

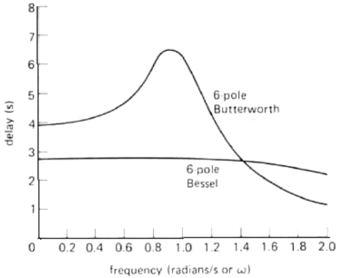




Anti-aliasing stage: t-response

- step response for 6-poles active filters
- Butterworth and Chebyshev: bad step-response (left) due to not constant delay ($\equiv$ non linear phase resp.) (right)



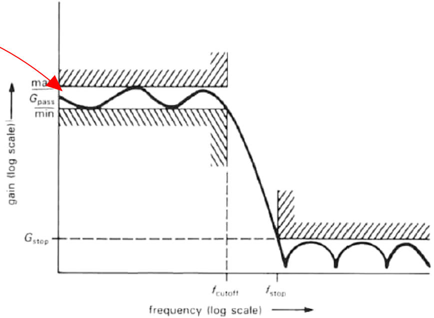


- Bessel: trades roll-off slope for step-response

Designing an anti-aliasing filter

To design a LPF:

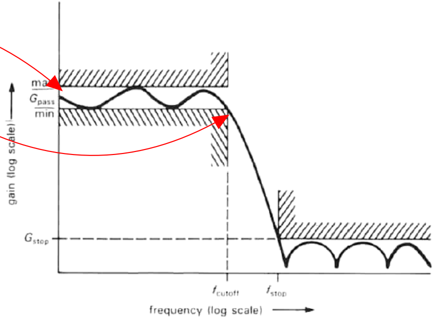
- choose allowed range of gain in passband (ripple)



Designing an anti-aliasing filter

To design a LPF:

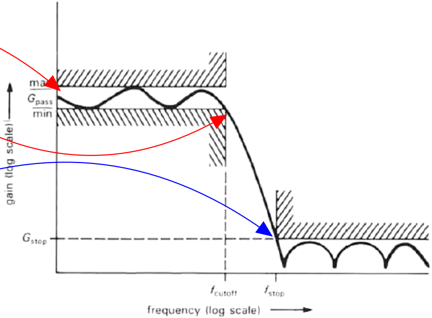
- choose allowed range of gain in passband (ripple)
- choose minimum frequency for which response leaves passband



Designing an anti-aliasing filter

To design a LPF:

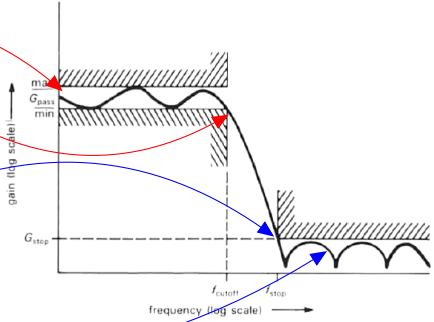
- choose allowed range of gain in passband (ripple)
- choose minimum frequency for which response leaves passband
- choose maximum frequency for which it enter the stopband



Designing an anti-aliasing filter

To design a LPF:

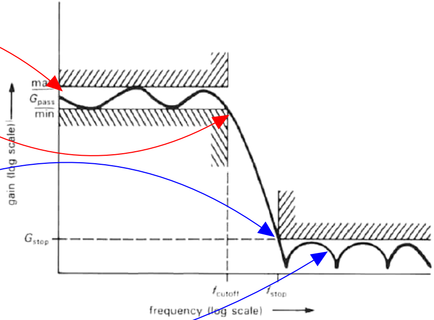
- choose allowed range of gain in passband (ripple)
- choose minimum frequency for which response leaves passband
- choose maximum frequency for which it enter the stopband
- choose minimum attenuation in stopband



Designing an anti-aliasing filter

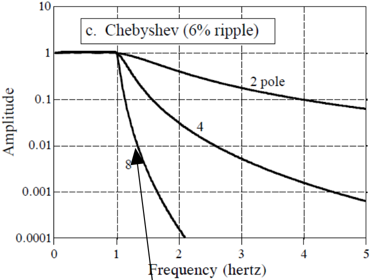
To design a LPF:

- choose allowed range of gain in passband (ripple)
- choose minimum frequency for which response leaves passband
- choose maximum frequency for which it enter the stopband
- choose minimum attenuation in stopband
- not necessarily in this order...



Exercise: design an anti-aliasing filter!

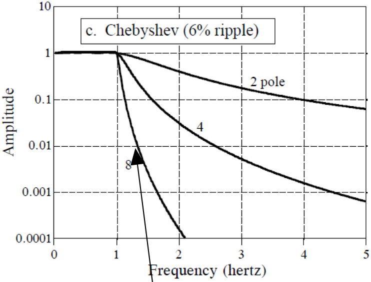
DATA: ADC has $F_s = 100$ MHz, 12 bit; allow for 6 % ripple in p-b and require at least 10^{-2} attenuation (-40 dB) at Nyquist frequency ($F_s/2 = 50$ MHz)



- 8-pole Cheb (6 % ripple): -40 dB at $1.35 \times f_c \Longrightarrow 1.35 \times f_c = 50$ MHz $\Longrightarrow f_c = 37$ MHz $\Longrightarrow$ choose 8-pole filter

Exercise: design an anti-aliasing filter!

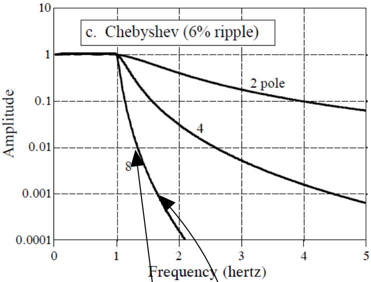
DATA: ADC has $F_s = 100$ MHz, 12 bit; allow for 6 % ripple in p-b and require at least 10^{-2} attenuation (-40 dB) at Nyquist frequency ($F_s/2 = 50$ MHz)



- 8-pole Cheb (6 % ripple): -40 dB at $1.35 \times f_c \implies 1.35 \times f_c = 50$ MHz $\implies f_c = 37$ MHz $\implies$ choose 8-pole filter
- 37 to 50 MHz = wasted land. Question: a real 12 bit ADC has $\approx$ 60 dB dynamic range... is 40 dB at $F_s/2$ enough atten.?

Exercise: design an anti-aliasing filter!

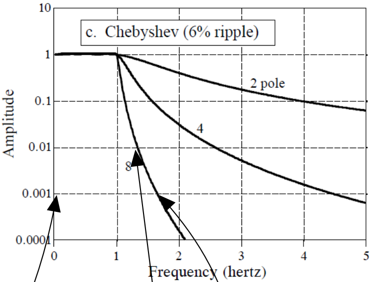
DATA: ADC has $F_s = 100$ MHz, 12 bit; allow for 6 % ripple in p-b and require at least 10^{-2} attenuation (-40 dB) at Nyquist frequency ($F_s/2 = 50$ MHz)



- 8-pole Cheb (6 % ripple): -40 dB at $1.35 \times f_c \implies 1.35 \times f_c = 50$ MHz $\implies f_c = 37$ MHz $\implies$ choose 8-pole filter
- 37 to 50 MHz = wasted land. Question: a real 12 bit ADC has $\approx$ 60 dB dynamic range... is 40 dB at $F_s/2$ enough atten.?
- passband stops at 37 MHz $\implies$ alias in passband for $f > 50 + (50 - 37) = 100 - 37 = 63$ MHz $(= 1.7 f_c)$

Exercise: design an anti-aliasing filter!

DATA: ADC has $F_s = 100$ MHz, 12 bit; allow for 6 % ripple in p-b and require at least 10^{-2} attenuation (-40 dB) at Nyquist frequency ($F_s/2 = 50$ MHz)

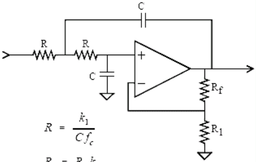


- 8-pole Cheb (6 % ripple): -40 dB at $1.35 \times f_c \implies 1.35 \times f_c = 50$ MHz $\implies f_c = 37$ MHz $\implies$ choose 8-pole filter
- 37 to 50 MHz = wasted land. Question: a real 12 bit ADC has $\approx$ 60 dB dynamic range... is 40 dB at $F_s/2$ enough atten.?
- passband stops at 37 MHz $\implies$ alias in passband for $f > 50 + (50 - 37) = 100 - 37 = 63$ MHz $(= 1.7 f_c)$
- at $1.7 f_c$ attenuation is ≈ 0.001 (60 dB), compatible with effective dynamic range

Sallen-Key circuit

How is the antialias implemented in electronics? Most used electronic scheme to get Bessel/Chebyshev/Butterworth response: Sallen-Key architecture. Same circuit gives all responses by suitable choice of ratios k_1 and k_2

FIGURE 3-8
The modified Sallen-Key circuit, a building block for active filter design. The circuit shown implements a 2 pole low-pass filter. Higher order filters (more poles) can be formed by cascading stages. Find k_1 and k_2 from Table 3-1, arbitrarily select R_1 and C (try 10K and 0.01µF), and then calculate R and R_f from the equations in the figure. The parameter, f_c, is the cutoff frequency of the filter, in hertz.

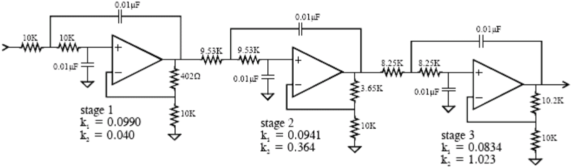


$$R = \frac{k_1}{C f_c}$$

$$R_f = R_1 k_2$$

TABLE 3-1
Parameters for designing Bessel, Butterworth, and Chebyshev (6% ripple) filters.

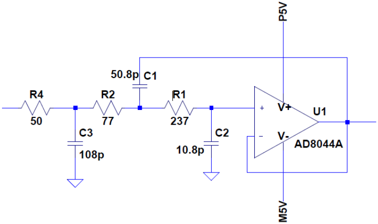
# poles		Bessel k_1	Bessel k_2	Butterworth k_1	Butterworth k_2	Chebyshev k_1	Chebyshev k_2
2	stage 1	0.1251	0.268	0.1592	0.586	0.1293	0.842
4	stage 1	0.1111	0.084	0.1592	0.152	0.2666	0.582
	stage 2	0.0991	0.759	0.1592	1.235	0.1544	1.660
6	stage 1	0.0990	0.040	0.1592	0.068	0.4019	0.537
	stage 2	0.0941	0.364	0.1592	0.586	0.2072	1.448
	stage 3	0.0834	1.023	0.1592	1.483	0.1574	1.846
8	stage 1	0.0894	0.024	0.1592	0.038	0.5359	0.522
	stage 2	0.0867	0.213	0.1592	0.337	0.2657	1.379
	stage 3	0.0814	0.593	0.1592	0.889	0.1848	1.711
	stage 4	0.0726	1.184	0.1592	1.610	0.1582	1.913



Sallen-Key circuit

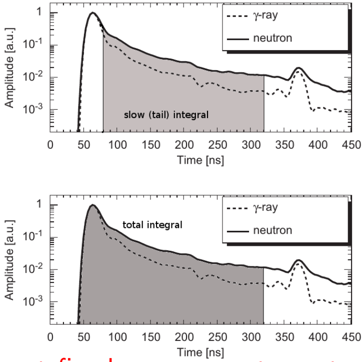
Many stages: increase complexity, noise, power dissipation...can we use just one?

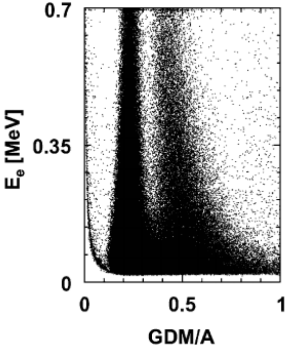
Example of 1-stage 3-pole Bessel Sallen-Key as implemented in Luigi Bardelli's “year 2000” board



One usually studies actual response using circuit simulators (spice, pspice, ltspice...). In this case, we get ≈ 20 dB attenuation of aliased frequencies in passband...is it acceptable?

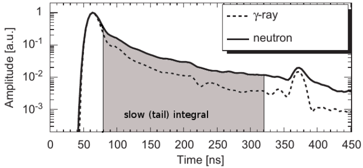
PSD: n/γ discrim. in liquid organic scintillators

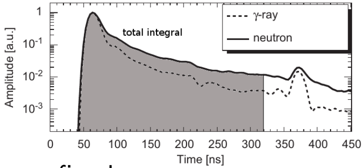


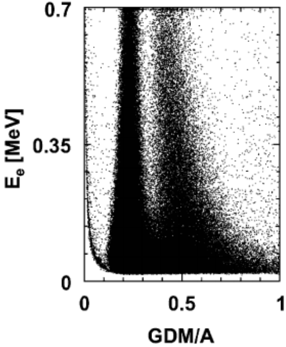


- at fixed energy, protons stopping power $\gg$ than electrons

PSD: n/γ discrim. in liquid organic scintillators

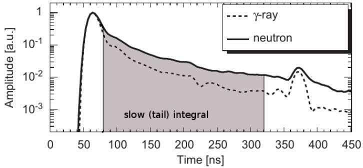


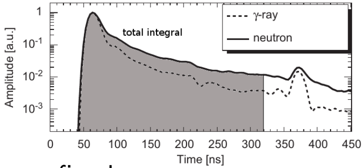


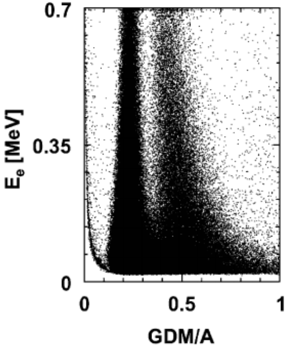


- at fixed energy, protons stopping power $\gg$ than electrons
- higher density of triplet states along track $\Longrightarrow$ signal has longer tail

PSD: n/γ discrim. in liquid organic scintillators

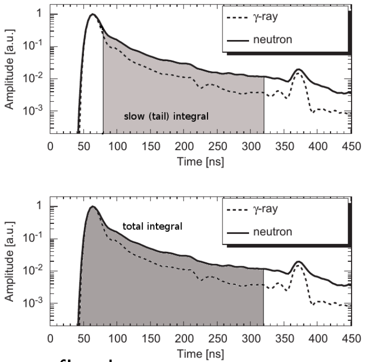


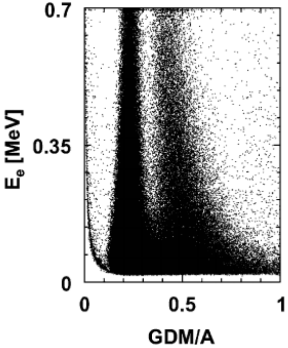




- at fixed energy, protons stopping power $\gg$ than electrons
- higher density of triplet states along track $\Longrightarrow$ signal has longer tail
- two integrations, usually *slow* (top left) and *total* (bottom left)

PSD: n/γ discrim. in liquid organic scintillators





- at fixed energy, protons stopping power $\gg$ than electrons
- higher density of triplet states along track $\Longrightarrow$ signal has longer tail
- two integrations, usually *slow* (top left) and *total* (bottom left)
- Right picture: "total (E) vs slow (GDM)". GDM normalized to pulse amplitude. Gammas (i.e. electrons) on the left, neutrons (i.e. protons) on the right.

Bibliography

[Akkoyun12] Nucl. Instr. and Meth. in Phys. Res. A **668** (2012) 26–58

[Bardelli2002] L.Bardelli, *et al.*, Nucl. Instr. and Meth. in Phys. Res. A 491 (2002) 244.

[Bardelli2004] L.Bardelli, *et al.*, Nucl. Instr. and Meth. in Phys. Res. A 521 (2004) 480–492.

[Bardelli2005] L.Bardelli, "Development of sampling and digital signal processing techniques with applications to Nuclear Physics detectors", Ph.D. Thesis, University of Florence, 2005 (downloadable from http://www.infn.it/thesis/index.php).

[Bardelli2006] L.Bardelli and G.Poggi, Nucl. Instr. and Meth. in Phys. Res. A 560 (2006) 517–523; L.Bardelli and G.Poggi, Nucl. Instr. and Meth. in Phys. Res. A 560 (2006) 524–538

[Bardelli2007] L.Bardelli et al., Nucl. Instr. and Meth. A 572 (2007) 882

[Carboni2012] S. Carboni, *et al.*, et al. Nucl. Instr. and Meth. in Phys. Res. A 664 (2012) 251–263

[Goulding1972] F.S. Goulding, "Pulse-shaping in low-noise nuclear amplifiers: a physical approach to noise analysis", Nucl. Instr. Meth. 100 (1972) 493

[Hellesen2013] Hellesen et al, Nucl. Instr. and Meth. in Phys. Res. A 720(2013) 135

[Horowitz1989] Horowitz and Hill, "The Art of Electronics", Cambridge Univ. Press, 1989

[Hou1978] Hou and Andrews, "Cubic Splines for image Interpolation and Filtering" IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, VOL. ASSP-26, N 0 . 6 , DECEMBER 1978

Bibliography (cont.ed)

[Kester04] Analog Devices Data Conversion Handbook, W. Kester ed., 2004

[Keys1981] R.G.Keys, Cubic Convolution Interpolation for Digital Image Processing,
IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-29, Dec. 1981

[Knoll00] G.F.Knoll, Radiation Detection and Measurement 3rd ed., Wiley, 2000

[Oppenheim10] A.V.Oppenheim, R.W.Schafer, Discrete-Time Signal Processing, Pearson, 2010

[Ottanelli2016] P.Ottanelli, Master Thesis, University of Florence, 2016

[Pastore2013] G.Pastore, Master Thesis, University of Florence, 2013

[Smith97] Steven W. Smith, "The Scientist and Engineer's Guide to Digital Signal Processing", copyright ©1997-1998 - (book's website: www.DSPguide.com)

[Spieler05] H.Spieler, Semiconductor Detector Systems, Oxford University Press, 2005

[Unser1993] M.Unser, "B-Spline Signal Processing: Part I-Theory",
IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 41, NO. 2, FEBRUARY 1993
M. Unser, "B-Spline Signal Processing: Part II-Efficient Design and Applications",
IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL. 41, NO. 2, FEBRUARY 1993

[Unser2000] M.Unser, Sampling—50 Years After Shannon, PROC. OF THE IEEE, VOL. 88, NO. 4, APRIL 2000