FIR filtering

FIR filtering theory is described in Lars presentation but other documents and presentation from Doris and Kazu are available at Indico search: FIR

Which FIR is applied with VeloTELL1CableFIRFilter

The filter is designed to be applied on pedestal and common mode subtracted non zero suppressed data.

It is applied in a way as close as possible to a TELL1 realistic scenario (thanks to Guido, which designed me this algorithm):
You want to apply a filter of order say 5. For every channel, you have to compute:

```
newADC[i] = g[-2] * ADC[i-2] + g[-1] * ADC[i-1] + g[0] * ADC[i] + g[1] * ADC[i+1] + g[2] * ADC[i+
this can be trivially written
newADC[i] = 1/128 * ( g[-2]*512 * (ADC[i-2]/4) + g[-1]*512 * (ADC[i-1]/4) + g[1]*512 * (ADC[i+1]/
g[0] is assumed to be one.
```

FIR coef inside the TELL1s must have a binary representation. The real values of the **g[k]** are of the order of the x-talk coefficients, that is O(0.05). Storing g[k]*512 on a 8 bit signed integer allows to explore the range +127/512 = 24% to -24% by step of 0.2%. Dividing the ADC value by 4 allows to drop the least significant bits which we do not want to enter the computation.

A VeloTELL1CableFIRFilter instance has a given N (x-talk order, defined by the amount of Kmi vector you give in options (see below)) and M (FIR order, given by the `M` option). Both have to be 3,5,7 or 9 (means you act on 1,2,3 or 4 chip channel neighbors).

The FIR filter acts only on a per-analog-cable basis, it cannot correct for a pulse that spreads over the boundary of two cables.

Algorithms sequence

VeloTELL1CableFIRFilter has to be inserted into the TELL1Emulator sequence. It requires the data to be pedestal subtracted or optionally also common mode subtracted. The reordering have to be done after the FIR. Actually, the FIR should be applied at the point it has been measured, that is, if the VeloXTalkComputer had an `InputDataLoc` taking CM data into account, you must apply the FIR after the CM. An minimal example is

```
TELL1Processing.Members += {
  "VeloTELL1EmulatorInit",
  "VeloTELL1PedestalSubtractor",
  "VeloTELL1LCMS",
  "VeloTELL1CableFIRFilter",
  "VeloTELL1Reordering",
  "VeloTELL1ClusterMaker"
};
VeloTELL1PedestalSubtractor.InputDataLoc  = "Raw/Velo/DecodedADC"; // to skip the VeloTELL1FIRFil
VeloTELL1PedestalSubtractor.OutputDataLoc = "Raw/Velo/SubtractedPedADCs";
VeloTELL1LCMS.InputDataLoc  = "Raw/Velo/SubtractedPedADCs";
VeloTELL1LCMS.OutputDataLocation = "Raw/Velo/ADCCMSuppressed";
VeloTELL1CableFIRFilter.InputDataLoc  = "Raw/Velo/ADCCMSuppressed";
VeloTELL1CableFIRFilter.OutputDataLocation = "Raw/Velo/somename";
VeloTELL1Reordering.InputDataLoc  = "Raw/Velo/somename";
```

In case there is a typo in this example, you can check `$VETRAROOT/options/TELL1Emulator.opts` to have a working example you can adapt.

The VeloTELL1CableFIRFilter options

- `InputDataLoc` and `OutputDataLoc` tells the algorithm where to read and write the data.
- `M` is the FIR order, the default value is 9, you acts on -4, +4 neighbors.
- `DebugCable = x;` prints for the given cable the ADC values before and after the FIR filter.

The `Tell1List` and coefficients options

- `Tell1List = { t1,t2,t3,...tn };` is the list of TELL1s to analyze the order matter. All the other TELL1s see their data duplicated to the `OutputDataLoc`.
- `Km4, Km3, Km2, Km1, Kp1, Kp2, Kp3, Kp4` are the **x-talk** (not FIR) real value coefficients per TELL1s and analog cable. For example Km2 defines all the h[-2] coefficients.

Each of the Kxy is a vector of double written with the following syntax:

```
Kxy = { t1_cable0_coef_xy, ..., t1_cable63_coef_xy, t2_cable0_coef_xy, ..., t2_cable63_coef_xy, .
```

the t1, t2, ..., tn being ordered like in the Tell1List vector. This is of course awful to write by yourself, but VeloXTalkComputer spits you a file with these lines out.

How to inverse the x-talk into FIR coefficients

The algebraic theory to inverse the x-talk values into FIR coeficients is summarized here (thanks Lars):

```
Basic equation to determine the transfer function of the
filter to cancel the x-talk :


d[n] = sum_-inf^+inf { g[k]*h[k-n] }

where
  d[n] is the kronecker delta
  g[n] is the transfer function of the FIR,
       i.e the coefficients for the filter.
  h[n] is the transfer function of the link,
       i.e the measured x-talk

if the g[n] is non-zero for only M terms, which means FIR
filter of order M, then the sum is reduced to:

d[n] = sum_(-M-1)/2}^{(M-1)/2 { g[k]*h[k-n] }

This can be considered as an infinite series of equations,
one for each value of n, that will give the relation between
g[n] and h[n]. These equations will have non-trivial
solution only for values of n where h[k-n] are not zero for
all terms in the sum. Assuming that the x-talk only extends
(N-1)/2 bins forwards and backwards, the requirements on n
for non trivial equations are:

-(M-1)/2 - n <=  (N-1)/2   (only first term in sum non-zero)
 (M-1)/2 - n >= -(N-1)/2   (only last term in sum non-zero)

This gives N+M-1 equations, -(M+N)/2+1 <= n <= (M+N)/2-1,
that defines the relation between g[n] and h[n].

Writing out the equations explicitly for M=3 and N=3:

                    k=-1          k=0          k=1

n=-2    d[-2] = g[-1]*h[ 1] +      0      +     0
n=-1    d[-1] = g[-1]*h[ 0] + g[ 0]*h[ 1] +     0
```

```
n= 0    d[ 0] = g[-1]*h[-1] + g[ 0]*h[ 0] + g[1]*h[ 1]
n= 1    d[ 1] =      0      + g[ 0]*h[-1] + g[1]*h[ 0]
n= 2    d[ 2] =      0      +      0      + g[1]*h[-1]

This system of equations can be put in matrix form

Y = (d[-2], d[-1], d[0], d[1], d[2])^T
  = (  0,     0,    1,    0,    0  )^T

G = (g[-1], g[0], g[1])^T


A = { (h[ 1],   0,     0  ),
      (h[ 0], h[ 1],   0  ),
      (h[-1], h[ 0], h[ 1]),
      (  0,   h[-1], h[ 0]),
      (  0,     0,   h[-1]) }

this give the equation AG = Y with M unknowns (the g[n]) and
N+M-1 equations that can be solved by the least square method.
```

-- Main.jborel - 30 Jun 2007

This topic: LHCb > VeloTELL1CableFIRFilter
Topic revision: r2 - 2007-08-21 - unknown