Gustavo Cancelo
CEPA/ESE-CD
Nov. 17th 2004

The CIC filter

It is easy to understand the CIC filter in the frequency domain. As shown in Figure 1, the CIC filter is a cascade of digital integrators followed by a cascade of combs (digital differentiators) in equal number. Between the integrators and the combs there is a digital switch or decimator, used to lower the sampling frequency of the combs signal with respect to the sampling frequency of the integrators.

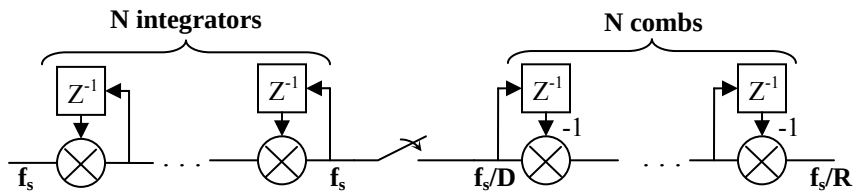


Figure 1

Each integrator contributes to the CIC transfer function with a pole. Each comb section contributes with a zero of order D, where D is the frequency decimation ratio. The CIC transfer function in the Z-plane becomes:

$$H(z) = H_I^N(z)\, H_C^N(z) = \frac{\left(1 - z^{-D}\right)^N}{\left(1 - z^{-1}\right)^N} = \left[\sum_{k=0}^{D-1} z^{-k}\right]^N$$

We must be careful here because we have two sampling frequencies in the system, related by D. If we evaluate the z-transference at the output sampling frequency $z=\exp(j2\pi f_s/D)$, the transference becomes

$$A(f) = \left[\frac{\sin(\pi f)}{\sin(\pi f / D)}\right]^N \approx \left[D.\frac{\sin(\pi f)}{\pi f}\right]^N \quad for \quad D >> 1$$

It is important to make few remarks:

- The filter gain is approximately D^N.
- The CIC transfer function has nulls at each multiple of the output sampling frequency $f=f_S/D$.
- There are only two control parameters the number of integrator/comb stages N and the decimation ratio D. In the Graychip N is fixed at 5.
- The CIC has a wide transition band. Strictly speaking, the passband is small (the amplitude "droops" quickly) and there is substantial aliasing specially around the first mirror image. This signify that the CIC filter must be accompanied of an anti-aliasing filter or be used on narrow-band spectrums.

For more detail on the frequency response and register length of the CIC please use Hogenauer paper [1].

Time domain analysis

Let u(k) be the input to the CIC. The integrator outputs will be called $x_1(k),\ldots,x_N(k)$. The decimator output will be called $x_D(k)$, and the comb outputs will be $y_1(k),\ldots,y_N(k)$

It is interesting to consider the cases of N=1 (one integrator/comb stage) and N=2 first. For N=1 the CIC is a simple integrator that can only remember the last D inputs.

$$x_1(n) = x_1(n-1) - u(n-1) = \sum_{k=0}^{n-1} u(k) \qquad (1)$$

The decimator output only picks one every D of $x_1(k)$ outputs

$$x_D(m) = x_1(Dm) = \sum_{k=0}^{Dm-1} u(k) \qquad (2)$$

In consequence, the output of the comb is

$$y_1(m) = x_D(m-1) - x_D(m-2) = \sum_{k=0}^{D(m-1)-1} u(k) - \sum_{k=0}^{D(m-2)-1} u(k)$$

$$y_1(m) = \sum_{k=D(m-2)}^{D(m-1)-1} u(k) \qquad (3)$$

If we ignore the filter's delay, each output is D times the average of the last D inputs to the filter.

For two stages the signal is integrated twice before decimating.

$$x_2(n) = x_2(n-1) - x_1(n-1) = \sum_{k=0}^{n-1} x_1(k)$$

$$x_2(n) = \sum_{k=0}^{n-1} x_1(k) = \sum_{k=0}^{n-1} \sum_{l=0}^{k} u(l) \qquad (4)$$

$x_2(k)$ is the running sum of $x_1(k)$ terms which grow in length as k increases.
$x_2(k) = u(0) + u(0)+u(1) + u(0)+u(1)+u(2) + \ldots + u(0)+u(1)+..+u(k)$

$$x_D(m) = x_1(Dm) = \sum_{k=0}^{Dm-1} \sum_{l=0}^{k} u(l) \qquad (5)$$

The first comb output is

$$y_1(m) = x_D(m-1) - x_D(m-2) = \sum_{k=0}^{D(m-1)-1} \sum_{l=0}^{k} u(l) - \sum_{k=0}^{D(m-2)-1} \sum_{l=0}^{k} u(l)$$

$$y_1(m) = \sum_{k=D(m-2)}^{D(m-1)-1} \sum_{l=0}^{k} u(l) \qquad (6)$$

The second comb output is

$$y_2(m) = y_1(m-1) - y_1(m-2) = \sum_{k=D(m-3)}^{D(m-2)-1} \sum_{l=0}^{k} u(l) - \sum_{k=D(m-4)}^{D(m-3)-1} \sum_{l=0}^{k} u(l)$$

$$y_2(m) = \sum_{k=D(m-3)}^{D(m-2)-1} \sum_{l=k-D+1}^{k} u(l) \qquad (7)$$

We observe that $y_2(m)$ is a double sum of input values u(k). The indices of each sum (***k*** and ***l***) have a range equal to D. In consequence each $y_2(m)$ output is the sum of D^2 samples. This process can be visualized in the graph below.

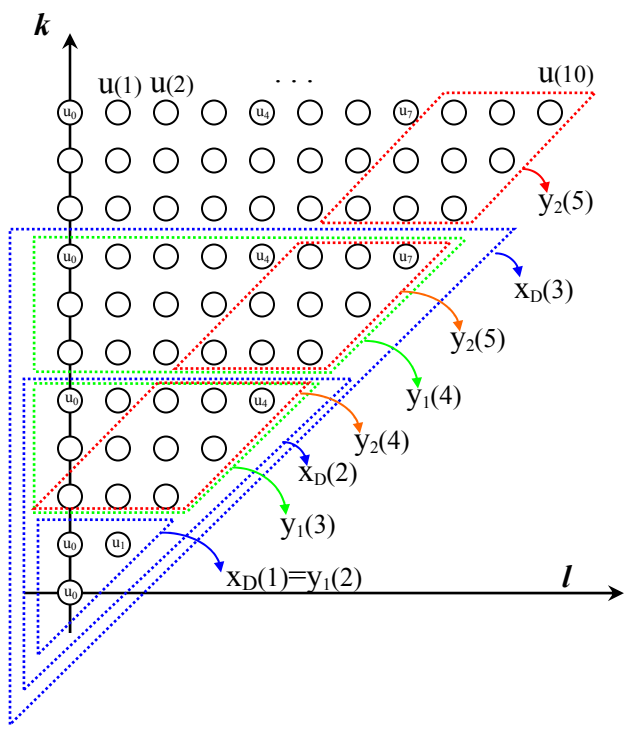


Figure 2

In Figure 2 the coordinate axis represent the indices ***k*** and ***l*** of the summations in equations (1) to (7). Each circle represents a u(.) input. Some u(.)'s have been added to the figure. They start at u(0) at the left most position in each row of the plot and increment by one as we move along the ***l*** axis. As shown in equation (4) the two-layer integrators create a pyramid or triangle like sum of u(k) inputs. (i.e. $x_2(k)$ = u(0) + u(0)+u(1) + u(0)+u(1)+u(2) + …+ u(0)+u(1)+..+u(k)). Integrator outputs $x_1(k)$ and $x_2(k)$ have been omitted from Figure 2 to avoid overloading the plot but can easily be spotted. Each row represent a $x_1(k)$ starting with $x_1(1)$= u(0) (first row). Note that the first row in the graph is $x_1(1)$ and not $x_1(0)$=0 due to the integrator delay. Consequently $x_2(k)$ starts at $x_2(2)$=u(0) because it has 2 unit delays. Each $x_2(k)$ is a small upside down triangle with its vertex in row 1, and its base in the row k-2 (i.e. -2 due to the delay). $x_2(k)$ adds all the u(.)'s included in the triangle.

The decimator output $x_D(.)$ copies one out of every D $x_2(.)$ samples. So each $x_D(.)$ also sums all the u(.)'s in a triangle, but the base of the triangle moves by D as $x_D(.)$ increments by 1. In Figure 2 D is assumed to be 3 and the dashed blue triangles represent $x_D(1)$, $x_D(2)$ and $x_D(3)$.
The combs subtract from the "current input" the "previous to the current input". In Figure 2, each output of the first comb subtracts the u(.)'s inside two consecutive blue triangles. Notice that the triangle corresponding to the "previous to the current input" is always inside the triangle corresponding to the "current input". In result, the output of the first comb takes the shape of the trapezoids drawn in dashed green. Figure 2 shows $y_1(3)$ and $y_1(4)$.

The second comb proceeds in the same way as the first comb. In this case the second comb subtracts the smaller trapezoid from the bigger trapezoid. It is interesting to note that in this case the polygons are not overlapping. However the u(.) terms contained in the small trapezoid are all included in the terms in the bigger trapezoid. As a result of the second comb operation we get the parallelograms shown in dashed red. We finally arrived to an interesting conclusion!

- As said, the output of the 2-stage CIC filter is the sum of D^2 number input samples (i.e. u(.)'s.).
- Each output is formed by D consecutive rows of Figure 2, having D consecutive u(.)'s in each row.
- Each row u(.)'s are displaced in time by 1 sample as we move in ascending k.
- The CIC output contains the "last" 2*D-1 input samples. The "last" means the last minus the CIC internal delay which is equal to the number of integrator/decimator stages.

The last bullet implies that as we increment the decimation factor D we are doing more averaging in the CIC filter. This will impact the signal-to-noise performance of the filter as it will be shown later. But first we need to generalize this result to a CIC filter with N integrator/comb stages.

The output of the Nth integrator is

$$x_N(n) = x_N(n-1) - x_{N-1}(n-1) = \sum_{k_1=0}^{n-1} x_{N-1}(k_1) = \sum_{k_1=0}^{n-1} \sum_{k_2=0}^{k_1} x_{N-1}(k_2) = \ldots$$

$$x_N(n) = \sum_{k_1=0}^{n-1} \sum_{k_2=0}^{k_1} \ldots \sum_{k_N=0}^{k_{N-1}} u(k_{N-1}) \qquad (8)$$

Equation (8) shows that the output of the Nth integrator contains N running sums. The indices of the sums, starting from the right most sum, go from 0 to the value of the index of the previous running sum.

$$x_D(m) = x_N(Dm) = \sum_{k_1=0}^{Dm-1} \sum_{k_2=0}^{k_1} \ldots \sum_{k_N=0}^{k_{N-1}} u(k_{N-1})$$

And the output of the last comb is

$$y_N(m) = \sum_{k_1=D(m-N)}^{D(m-N+1)-1} \sum_{k_2=k_1-D+1}^{k_1} \ldots \sum_{k_N=k_{N-1}-D+1}^{k_{N-1}} u(k_{N-1}) \qquad (9)$$

The output of a N-stage CIC is the sum of D^N of the "last" N(D-1) input samples. The delay in the output is N if D>N.

Signal to Noise

In the signal to noise analysis we assume that the input samples are affected by white Gaussian additive noise. That is the noise samples are independent and identically distributed like $n_i \sim N(0,\sigma^2)$. The autocorrelation function of the input noise is $R_{xx}(k)=\delta(0)$.

Parameter estimation from noisy signals is one of the main subjects in signal processing. As is the case in the BPM system, many signals become zero for a portion of the observation or measuring cycle. Without loss of generality let's assume that the signal has 2 values 0 and A. and is contaminated with white Gaussian noise.

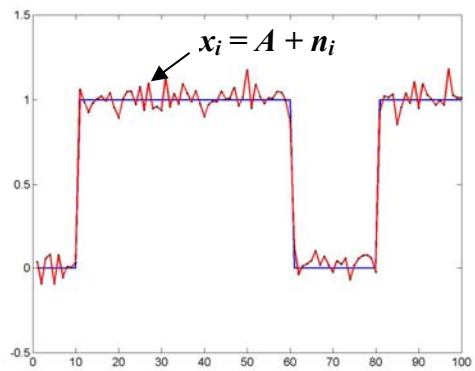


Figure 3

We are trying to estimate signal **A** from our measurements $\mathbf{x_i}$. For instance:

$x_1 = A + n_1$

And in general $x_k = A + n_k$

The signal to noise ratio of a single sample is

$$(S/N)_{1\ sample} = A^2/\sigma^2. \qquad (10)$$

It is well known that if we average two WGN samples the signal to noise ratio improves by a factor of 2 because the variance of the noise is smaller by that factor.

$x_3 =(x_1 + x_2)/2 = A + n_3$ where $n_3 = (n_1 + n_2)/2$ is distributed $n_3 \sim N(0,\sigma^2/2)$.

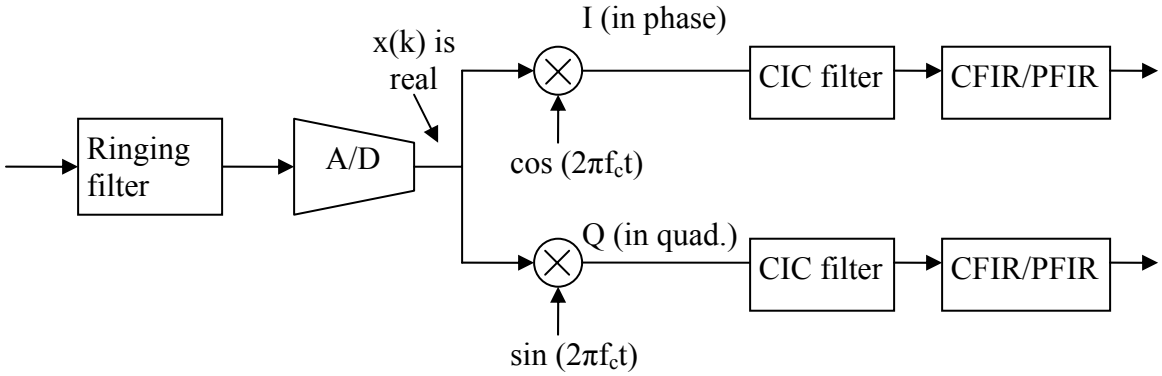
$$(S/N)_{2\ samples} = A^2/(\sigma^2/2) = 2*(S/N)_{1\ sample}$$

However, if the measurement of one of the samples is pure noise the signal to noise ratio is smaller by a factor of 2 even when the noise distribution has a lower sigma.

Let $x_1 = A + n_1$ and $x_2 = n_2$

$$(S/N)_{2\ samples} = (A/2)^2/(\sigma^2/2) = (S/N)_{1\ sample}/2 \qquad (11)$$

This analysis can be applied to the CIC filter. As we said the CIC's output is the sum of D^N of the "last" N(D-1) input samples. The CIC filter is used as a decimator to reduce the sampling frequency of the output. This is a common practice for narrow-band signals. In the BPM case the "information" (i.e. beam position) if narrow banded to few KHz, however the noise spectrum in the Echotek system before the CIC filter is several MHz wide. Figure 4 shows how the signal is down-converted before the CIC filter.



As shown in Figure 4 the input to the digitizer is a real signal. The digitized signal goes through the down-conversion where a complex envelope with "in-phase" and "quadrature" components is generated. The useful information is in the complex envelope signals I and Q that can be used to calculate amplitude and phase. The BPM system uses two channels similar to the one shown in Figure 4. The BPM pick-ups deliver a differential signal in channels A and B. Position is calculated as ***p=26mm*[|A| - |B|] /[|A| + |B|]***.

A and B signals are modulated by the ringing filter at ~53MHz. The A/D converter samples at high speed (~74MHz). After the down-conversion I and Q are clearly oversampled with respect to the bandwidth of interest for position calculation. To lower the sampling frequency we can use decimation in the CIC or FIR filters or otherwise use post processing in the Echotek FPGAs.

Let's assume that a fraction α.D of the last D inputs to the CIC filter are pure noise. For a single stage integrator/comb CIC, using equations (3) and (11), the signal to noise ratio becomes

$$(S/N)_{D\ samples} = (\alpha A)^2/(\sigma^2/D) = \alpha^2.D.(S/N)_{1\ sample} \qquad (12)$$

On one hand, the signal to noise ratio is improved by D because the noise becomes distributed a the sample mean of length D. On the other hand we have a counter effect due to the loss of signal energy. This effect is quadratic in α. Since α <1, it implies a reduction in the signal to noise ratio.

A CIC filter with more stages will have a similar effect but is more difficult to analyze because the decimation makes the CIC a non-stationary system. If we take a close look to equation (7)

$$y_2(m) = \sum_{k=D(m-3)}^{D(m-2)-1} \sum_{l=k-D+1}^{k} u(l)$$

we can develop this as:

$$y_2(m) = D.u(r) + (D-1).u(r-1) + (D-1).u(r+1) + \ldots + u(r+D/2) + u(r-D/2) \qquad (13)$$

where r is a variable related to m by the filters delay. (i.e. ***r=D*(m + [D/2] - 3)*** but this only complicates the equation above.

Equation (12) shows that there is a non uniform mix of input samples in the output of the CIC. Some samples appear as much as D times and others only one time. So the signal to noise ratio depends on what

sample is only noise. The worst case is given by the most popular sample. If that sample is pure noise α becomes $\alpha=(D^2-D)/D^2$. α drops significantly fast as more input samples are made pure noise.
Looking at equation (12) one can argue that if $\alpha\sim 1$ the CIC helped us gain a factor of D in signal to noise ratio. This is true; however, the same gain can be achieved by calculating the sample mean in an FPGA.