CIC Filters

Dr Chris Dick
DSP Chief Scientist
Advanced Systems Technology Group
DSP Division

Cascaded Integrator-Comb (CIC) Filters

• CIC filters are efficient architectures for efficiently implementing large sample rate changes
  – interpolation
  – decimation
  – frequently employed in digital down converters

• Only addition and subtraction functional units are required
  – these are very efficiently implemented in FPGAs

• Hogenauer [1] was the first to publish the concept of multirate CIC filters

Boxcar Filter

\[ H(z) \]

\[ x(n) \rightarrow [\text{Boxcar Filter}] \rightarrow y(n) \]

NTAPS = 4

NTAPS = 5

NTAPS = 6

NTAPS = 7

NTAPS = 12

NTAPS = 14

NTAPS = 16
Cascade of Boxcar Filters

- For a single boxcar filter
  - Peak sidelobe level is ~13 dB independent of the number of terms (taps)
- The sidelobes may be reduced by repeated convolution
  - Cascade of Boxcar Filters
- Once the input signal bandwidth has been reduced the sample rate may be adjusted

\[ x(n) \rightarrow H(z) \rightarrow H(z) \rightarrow H(z) \rightarrow R^{-1} \rightarrow y_0(n) \]

\( N \) FIR filter segments

CIC Filters

- CIC filter is an efficient technique to implement a cascade of boxcar filters

\[
G(z) = [H(z)]^N \\
H(z) = \sum_{k=0}^{N-1} z^{-k} \\
G(z) = \left[ \sum_{k=0}^{N-1} z^{-k} \right]^N
\]

Now use the sum of a geometric series to obtain

\[
G(z) = \left[ \sum_{k=0}^{N-1} z^{-k} \right]^N = \left[ \frac{(1-z^{-1})^N}{1} \right] = H_i^N H_0^N
\]

where \( H_i \) is the transfer function of an integrator and \( H_0 \) is the transfer function of a differentiator
CIC Filters v2.1 © C. Dick

CIC Decimator

\[ x(n) \xrightarrow{H_l(z)} y_0(n) \]

\[ X(z) \xrightarrow{H(z)} Y(z) \]

CIC Interpolator

\[ G(z) = \left[ \sum_{k=0}^{N-1} z^{-k} \right]^N = \left[ \frac{(1-z^{-1})^N}{1-z^{-1}} \right] = \frac{1(z^N)}{(1-z^{-1})^N} = H_0^N H_1^N \]
CIC Filter - Frequency Response

• Frequency response of CIC filter is produced by evaluating

\[ H_{\text{CIC}}(z) = \left[ \sum_{k=0}^{RM-1} z^{-k} \right]^N \]

on the unit circle \( z = e^{j2\pi f/R} \) where \( f \) is the frequency wrt the low sampling rate \( fs/R \)

• The power response is

\[ P(f) = \left[ \frac{\sin \pi Mf}{\sin \frac{\pi f}{R}} \right]^{2N} \]

CIC Filter - Frequency Response

• For large \( R \) and using the small angle approximation, the power response is approximately

\[ \hat{P}(f) = \left[ \frac{\sin \pi Mf}{\pi Mf} \right]^{2N} \]
CIC Filters

- For CIC decimation filters the region around every Mth null is folded into the passband causing aliasing errors
- For interpolation filters imaging occurs in these same regions

example frequency response for $N = 4$, $M = 1$, $R = 7$, $f_c = 1/8$
Implementation Considerations

- While the implementation of a Hogenauer filter would appear to be straightforward
- Requires only
  - Addition
  - Subtraction
  - Delay
  - Resampling switch
- Register growth in the filter can be a challenge, impacting
  - Area
  - Max operating frequency

Register Growth

- Maximum register growth

\[ G = (RM)^N \]

- If input sample precision is \( B_{\text{in}} \) the most significant bit at the filter output is

\[ B_{\text{max}} = \lceil N \log_2(RM) + B_{\text{in}} \rceil \]
Register Growth

Datapath Pruning

- For a realistic set of CIC parameters the word growth will be significant
- Typically we will not support this register growth when the CIC output is transferred to the next stage in the system datapath
  - Noise is added as a result of rounding the high-precision CIC output to a lower precision value
- This provides an opportunity for bit-pruning the CIC internal datapath
  - Reduce FPGA area requirements
  - Shorten the add/sub carry chains
    - Resulting in high operating clock frequency