OpenCPI
CIC Decimator (TimeStamped) Component Data Sheet

OpenCPI Release: v2.4.7

Summary - CIC Decimator (TimeStamped)

Name	cic_dec_ts
Worker Type	Application
OpenCPI Release	v2.4.7
Last Update	4/2019
Component Library	ocpi.assets_ts.components
Workers	cic_dec_ts.hdl
Tested Platforms	alst4, isim, Matchstiq-Z1(PL), ml605, modelsim, xsim, ZedBoard(PL)

Functionality

The CIC decimator has `N` cascaded integrator stages with an input data rate of f_s, followed by a rate change by a factor `R`, followed by `N` cascaded comb stages with an output data rate of $\frac{f_s}{R}$. The differential delay, `M`, affects the slope of the transition region. Figure 1 diagrams the decimating CIC filter.

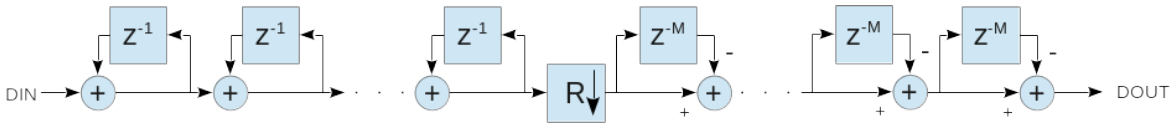

Figure 1: Cascaded Integration Comb Decimation filter Block Diagram

Worker Implementation Details

cic_dec_ts.hdl

Number of Stages

The generic `N` sets the number of integrators and comb stages in the filter. Increasing the number of stages increases the attenuation in the sidelobes and decreases the 3dB bandwidth of the passband. The recommended range for this parameter is 3 to 6. Consult the reference material for an in depth discussion of the frequency response of the filter as a function of the generics in this module.

Bit Growth

For this design, the output data width for the comb stages is configurable via `ACC_WIDTH`. To adjust for bit growth in the data path and to ensure no quantization error at the output, this equation should be used to determine the value of `ACC_WIDTH`.

$$ACC\_WIDTH = N * CEIL(log_2(R * M)) + DIN\_WIDTH \quad (1)$$

Theory

A CIC filter is comprised of `N` integrator sections cascaded together with `N` comb sections. Combining the transfer functions for all sections results in the system response function seen in Equation 2. Note that the filter has zeros at integer multiples of $\frac{f_s}{RM}$ Hz.

$$H(z) = \Big[H_{int}(z)\Big]^N \Big[H_{comb}(z)\Big]^N = \Big[\frac{1}{(1-z^{-1})^N}\Big]\Big[(1-z^{-RM})^N\Big] = \frac{(1-z^{-RM})^N}{(1-z^{-1})^N} \quad (2)$$

Block Diagrams

Top level

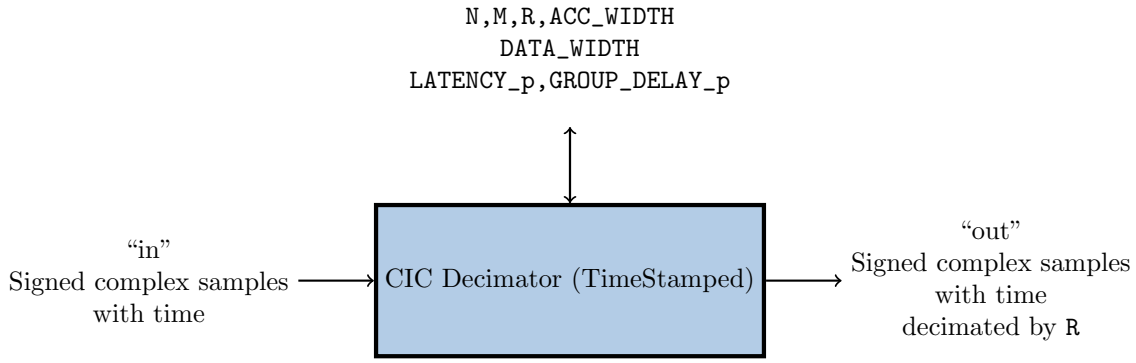

Figure 2: Top Level Block Diagram

Source Dependencies

cic_dec_ts.hdl

- assets_ts/components/cic_dec_ts.hdl/cic_dec_ts.vhd
- assets_ts/components/cic_dec_ts.hdl/cic_dec_gen2.vhd

Component Spec Properties

Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Usage
N	UChar	-	-	Parameter	-	3	Number of Stages
M	UChar	-	-	Parameter	-	1	Differential Delay
R	UShort	-	-	Parameter	-	4	Decimation Factor
ACC_WIDTH	UChar	-	-	Parameter	-	-	Accumulation Width *(2)
DATA_WIDTH	UChar	-	-	Readable	-	16	Data width

Worker Properties

cic_dec_ts.hdl

Type	Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Usage
Property	LATENCY_p	Ushort	-	-	Parameter	-	1	Number of clock cycles between a valid input and a valid output
Property	GROUP_DELAY_p	Ushort	-	-	Parameter	-	(N+1)*R	Number of valid inputs before a valid output is given

Component Ports

Name	Producer	Protocol	Optional	Advanced	Usage
in	False	Complex_Short_With_Metadata	False	-	Complex signed samples with time
out	True	Complex_Short_With_Metadata	False	-	Complex signed samples with time

Worker Interfaces

cic_dec_ts.hdl

Type	Name	DataWidth	Advanced	Usage
StreamInterface	in	32		Complex signed samples with time
StreamInterface	out	32	insertEOM=true, workerEOF='true'	Complex signed samples with time

Control Timing and Signals

The CIC Dec worker uses the clock from the Control Plane and standard Control Plane signals.

Worker Configuration Parameters

cic_dec_ts.hdl

Performance and Resource Utilization

cic_dec_ts.hdl

Test and Verification

Two test cases are implemented to validate the CIC Decimator component:

1. Impulse response - Input file consists of max Q0.15 values for I and Q followed by zeros
2. Step response - Input file consists of max Q0.15 values

For both test cases, verification of the output file is performed using the input file and a python model of the CIC Decimator. The python model can be found in the *opencpi.dsp_utils* module included with the framework. This unit test also implements *view*, which calls a script for plotting the input and output data files.

In both of the test cases, the samples data is interleaved with all of the operations of the Complex_Short_With_Metadata protocol in the following sequence:

1. Interval
2. Time
3. Samples
4. Time
5. Samples
6. Flush
7. Sync

References

(1) Ronald E. Crochiere and Lawrence R. Rabiner. Multirate Digital Signal Processing. Prentice-Hall Signal Processing Series. Prentice Hall, Englewood Cli_s, 1983.
(2) Eugene B. Hogenauer, An Economical Class of Digital Filters for Decimation and Interpolation, IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-29, No. 2, April 1981.
(3) Matthew P. Donadio, CIC Filter Introduction, http://home.mit.bme.hu/ kollar/papers/cic.pdf