OpenCPI
CIC Decimator Component Data Sheet

OpenCPI Release: v2.4.7

Summary - CIC Decimator

Name	cic_dec
Worker Type	Application
OpenCPI Release	v2.4.7
Last Update	4/2019
Component Library	ocpi.assets.dsp_comps
Workers	cic_dec.hdl, cic_dec.rcc
Tested Platforms	alst4, CentOS7, 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.

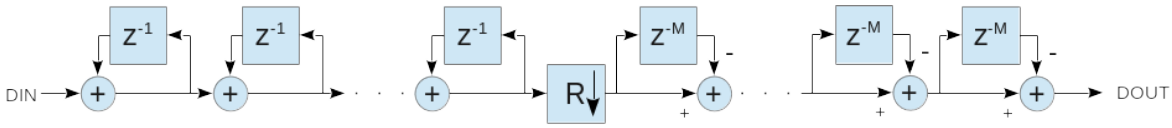

Figure 1: Cascaded Integration Comb Decimation filter Block Diagram

Worker Implementation Details

cic_dec.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`.

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

cic_dec.rcc

The RCC implementation for the `cic_dec` worker in C++ is to function as a work-alike of the `cic_dec.hdl` worker. This RCC worker is based on the python model used to verify the output of the HDL worker. The worker has three separate stages and is not complete until there is no data left to be output. The first stage of the worker reads in all the input data, buffer by buffer. The second stage the worker processes the data all, once of the data is read. The third stage outputs the data, buffer by buffer. Both the first and third stage run multiple times until there is no data left to be read in or out. The decimation is based on the total amount of data being read, so the worker has to figure out how large the incoming data set is.

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.

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

Block Diagrams

Top level

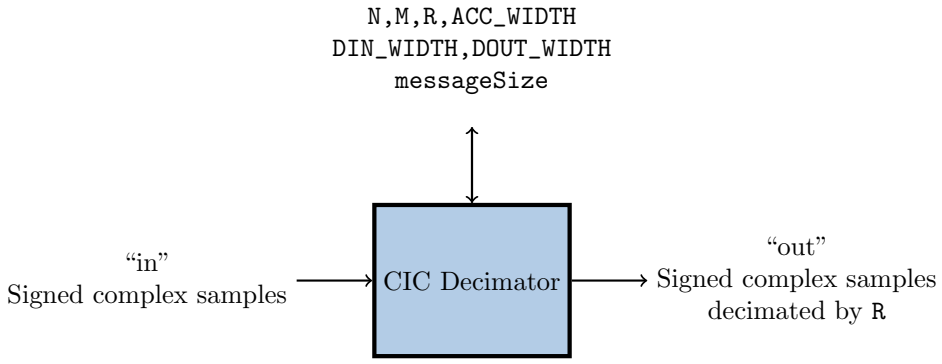

Figure 2: Top Level Block Diagram

Source Dependencies

cic_dec.hdl

- assets/components/dsp_comps/cic_dec.hdl/cic_dec.vhd
- assets/hdl/primitives/dsp_prims/dsp_prims_pkg.vhd
  assets/hdl/primitives/dsp_prims/cic/src/cic_dec_gen.vhd

Component Spec Properties

Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Usage
N	UChar	-	-	Readable	-	-	Number of Stages
M	UChar	-	-	Readable	-	-	Differential Delay
R	UShort	-	-	Readable	-	-	Decimation Factor
ACC_WIDTH	UChar	-	-	Readable	-	-	Accumulation Width *(2)
DIN_WIDTH	UChar	-	-	Readable	-	-	Input data width
DOUT_WIDTH	UChar	-	-	Readable	-	-	Output data width
messageSize	UShort	-	-	Readable, Writable	-	8192	Number of bytes in output message

Worker Properties

cic_dec.hdl

Type	Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Usage
SpecProperty	N	-	-	-	Parameter	3-6	3	Number of Stages
SpecProperty	M	-	-	-	Parameter	1-2	1	Differential Delay
SpecProperty	R	-	-	-	Parameter	4-8192	4	Decimation Factor
SpecProperty	DIN_WIDTH	-	-	-	Parameter	16	16	Input Data Width
SpecProperty	ACC_WIDTH	-	-	-	Parameter	*	22	Accumulation Width *(2)
SpecProperty	DOUT_WIDTH	-	-	-	Parameter	16	16	Output Data Width

cic_dec.rcc

Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Description
N	Uchar	-	-	Parameter	Standard	3	Number of Stages
M	Uchar	-	-	Parameter	Standard	1	Differential Delay
R	Ushort	-	-	Parameter	Standard	4	Decimation Factor. 1 output for every R inputs.
DIN_WIDTH	Uchar	-	-	Parameter	Standard	16	Width of I and Q as they appear on the input port
ACC_WIDTH	Uchar	-	-	Parameter	Standard	22	Accumulator width used inside CIC primitive
DOUT_WIDTH	Uchar	-	-	Parameter	Standard	16	Width of I and Q as they appear on the output port
messageSize	Ushort	-	-	Writable	Standard	8192	Unused property. Formerly used to set output message size

Component Ports

Name	Producer	Protocol	Optional	Advanced	Usage
in	False	iqstream_protocol	False	-	Complex signed samples (Q0.15 I, Q0.15 Q).
out	True	iqstream_protocol	False	-	Complex signed samples (Q0.15 I, Q0.15 Q).

Worker Interfaces

cic_dec.hdl

Type	Name	DataWidth	Advanced	Usage
StreamInterface	in	32	ZeroLengthMessages=true	Signed complex samples
StreamInterface	out	32	ZeroLengthMessages=true	Signed complex samples

cic_dec.rcc

Name	Producer	Protocol	Optional
in	False	iqstream_protocol.xml	False
out	True	iqstream_protocol.xml	False

Control Timing and Signals

The CIC Decimation filter HDL worker uses the clock from the Control Plane and standard Control Plane signals.

This worker has a latency of `N*2+1` valid input data clock cycles.

Latency
`N*2+1`

Worker Configuration Parameters

cic_dec.hdl and cic_dec.rcc

Performance and Resource Utilization

cic_dec.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.

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