Summary - Zero Padding

This worker will be deprecated in OpenCPI 2.0. Use the Zero Pad component for new designs.

Name	zero_padding
Worker Type	Application
Version	v1.3
Release Date	February 2018
Component Library	ocpi.assets.util_comps
Workers	zero_padding.rcc zero_padding.hdl
Tested Platforms	xsim, isim, modelsim, alst4, ml605, ZedBoard(PL), Matchstiq-Z1(PL), linux-x13_3-arm, c7-x86_64

Functionality

The Zero Padding component functions to expand input bits into signed Qm.n output samples within the range of ±1.0, while inserting a variable number of zeros between output samples.

Output data widths of 8/16/32/64 are supported resulting in (respectively) Q0.7, Q0.15, Q0.31, and Q0.63 output formats.

Worker Implementation Details

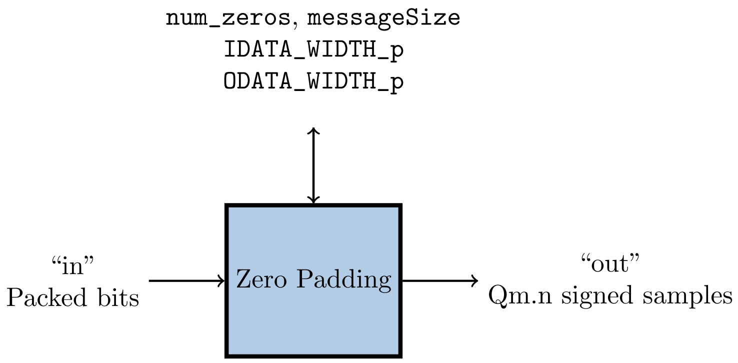
The Zero Padding component couples the underlying data, or protocol, with the size of the output data plane in order to fully load a Qm.n sample within the output bus width. In order to be maximally flexible, the component does not define input/output protocols explicitly. Since the input is simply bits, the input protocol is irrelevant and defined by the component feeding the Zero Padding, such as the File Reader. The input/output data widths are defined at build time, which in turn define the respective input/output sample sizes.

Theory

The Qm.n format defines the range to be -2^m to $2^m - 2^{-n}$, with a resolution of 2^{-n}. For the Zero Padding component, m is equal to zero, while n is defined at compile-time and is equal to the size of the output data width minus one. For example, an output data width of 16 results in Q0.15 format, where numbers are in the range of -1 to $+0.999969482421875$ (almost $+1$) with a bit resolution of 0.000030517578125.

Block Diagrams

Top level



State Machine

Two finite-state machines (FSMs) are implemented by this worker. One FSM implements worker functionality while the other supports Zero-Length Messages.

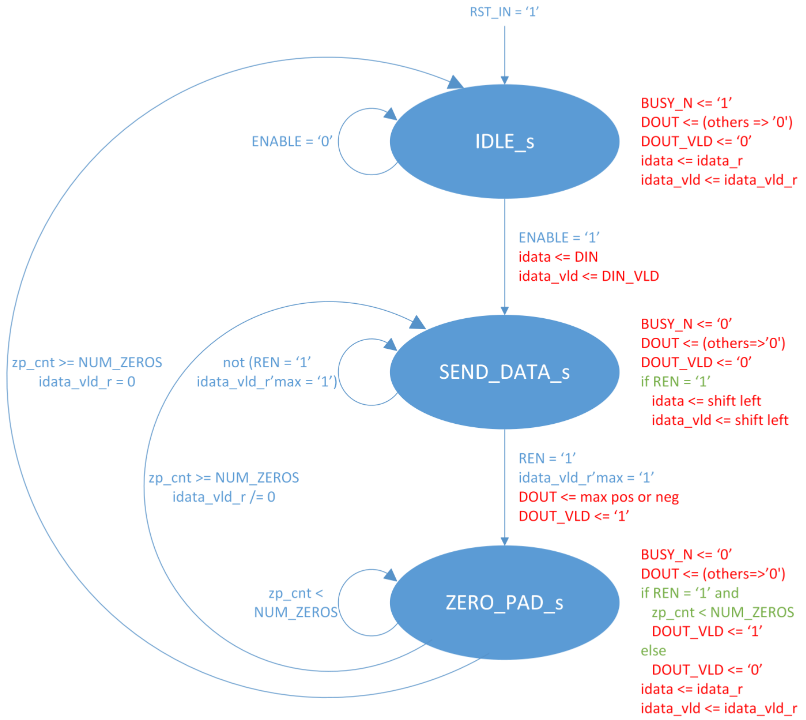


Figure 1: Zero Padding FSM

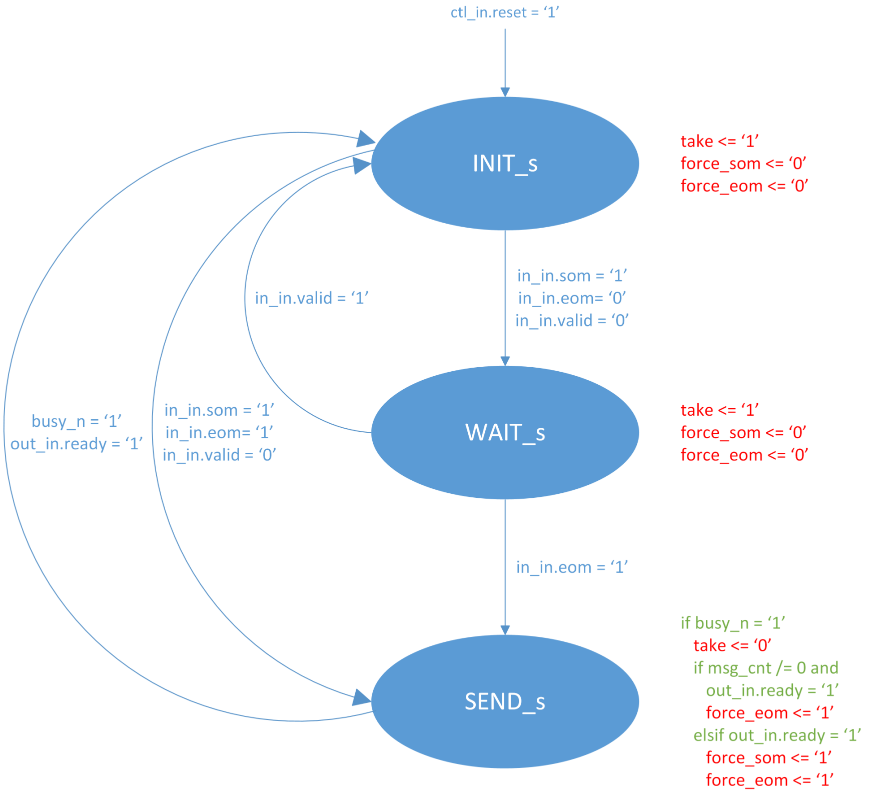


Figure 2: Zero-Length Message FSM

Source Dependencies

zero_padding.rcc

- ocpiassets/components/util_comps/zero_padding.rcc/zero_padding.cc

zero_padding.hdl

- ocpiassets/components/util_comps/zero_padding.hdl/zero_padding.vhd
- ocpiassets/hdl/primitives/util_prims/util_prims_pkg.vhd
  ocpiassets/hdl/primitives/util_prims/zp/src/zero_padding_gen.vhd

Component Spec Properties

Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Usage
IDATA_WIDTH_p	ulong	-	-	Readable, Parameter	8/16/32/64	32	Input port data width
ODATA_WIDTH_p	ulong	-	-	Readable, Parameter	8/16/32/64	32	Output port data width
num_zeros	ushort	-	-	Readable, Writable	Standard	-	number of zeros to be inserted between output samples
messageSize	ushort	-	-	Readable, Writable	Standard	8192	number of bytes in output message

Worker Properties

zero_padding.hdl

Type	Name	Type	SequenceLength	ArrayDimensions	Accessibility	Valid Range	Default	Usage
Property	MAX_NUM_ZEROS_p	ulong	-	-	Readable, Parameter	0-255	255	Maximum number of zeros

Component Ports

Name	Producer	Protocol	Optional	Advanced	Usage
in	False	-	False	-	Packed bits
out	True	-	False	-	Qm.n signed samples representing 1.0

Worker Interfaces

zero_padding.hdl

Type	Name	DataWidth	Advanced	Usage
StreamInterface	in	IDATA_WIDTH_p	-	Size defined by IDATA_WIDTH_p
StreamInterface	out	ODATA_WIDTH_p	-	Sample size defined by ODATA_WIDTH_p

Control Timing and Signals

zero_padding.hdl

This worker implementation uses the clock from the Control Plane and standard Control Plane signals.

Performance and Resource Utilization

zero_padding.rcc

Table entries are a result of compiling the worker with the following parameter/property set:

- `IDATA_WIDTH_p`=16
- `ODATA_WIDTH_p`=16
- `num_zeros`=1

Processor Type	Processor Frequency	Run Function Time
linux-c6-x86_64 Intel(R) Xeon(R) CPU E5-1607	3.00 GHz	∼ 5 ms
linux-c7-x86_64 Intel(R) Core(TM) i7-3630QM	2.40 GHz	∼ 5 ms
linux-x13_3-arm ARMv7 Processor rev 0 (v7l)	666 MHz	∼ 21 ms

zero_padding.hdl

Worker Build Configuration "0":
Table entries are a result of building the worker with the following parameter sets:

- `IDATA_WIDTH_p=8`
- `ocpi_debug=false`
- `ODATA_WIDTH_p=8`
- `MAX_NUM_ZEROS_p=255`
- `ocpi_endian=little`

Table 1: Worker Build Configuration "0"

OpenCPI Target	Tool	Version	Device	Registers	LUTs	Fmax (MHz)	Memory/Special Function
stratix4	Quartus	15.1.0	EP4SGX230KF40C2	187	238	N/A	N/A
virtex6	ISE	14.7	6vlx240tff1156-1	181	254	282.685	N/A
zynq	Vivado	2017.1	xc7z020clg484-1	186	177	201.369	N/A
zynq_ise	ISE	14.7	7z020clg484-1	181	254	306.560	N/A

Test and Verification

Both input and output data widths of 8/16/32/64 are supported and fully tested on both RCC and HDL worker implementations. The sixteen cross products of these input/output data width combinations are built for both RCC and HDL workers. These input/output combinations are each tested with `num_zeros` equal to 0, 1, 128, and 255 resulting in 64 test cases for both RCC and HDL workers.

Input data is generated by a python script with an input parameter that defines the number of 32-bit words to produce. The input file consists of a repeating pattern of 0x0123456789ABCDEF. The number of 32-bit words for each test case is 2048, which results in 1024 64-bit samples, 2048 32-bit samples, 4096 16-bit samples, or 8192 8-bit samples. Thus for each test case the 64-bit test pattern is repeated 1024 times to produce a file of 65,536 bits (or 8192 bytes).

The Zero Padding component inputs each bit and expands the bit into Qm.n format within the range of ±1.0, where $m = 0$, and n is defined by the width of the output data bus. Then `num_zeros` zeros are inserted between each output sample of width `ODATA_WIDTH_p`.

For verification, the output file is first checked that the data is not all zero, and is then checked for the expected length. Once these quick checks are made the output data is compared against expected results sample-by-sample without use of any gold files.