

OpenCPI

Pattern v2 Component Data Sheet

OpenCPI Release: v2.4.7

Revision History

Revision	Description of Change	Date
v1.4	Initial Release	10/2018
v1.5	• Fixed incorrect property accessibility "Writable" and changed it to "Initial" for the <code>messagesToSend</code> property. • Clarified that the opcode that is sent by the component is an 8 bit opcode.	4/2019
v1.6	Converted Worker to version 2	11/2019
v1.7	Table of Worker Configurations and Resource Utilization Table removed	5/2020
v2.2	Pattern v2 updated to use input port clock. RCC version of pattern_v2created. Functionality of messagesToSend slightly changed/improved.	7/2021

Summary - Pattern v2

Name	pattern_v2
Worker Type	
OpenCPI Release	v2.4.7
Last Update	05/2020
Component Library	ocpi.assets.util_comps
Workers	pattern_v2.hdl
Tested Platforms	isim, Matchstiq-Z1(PL), xsim, ZedBoard(PL), centos 7

Functionality

The `pattern_v2` component provides the ability to output a pattern of messages by allowing the user to create a record of messages each having a configurable number of bytes (bytes that are a multiple of 4) and associated 8 bit opcode. Through a set of properties, the component may send messages (data and opcode) up to the amount dictated by the build-time parameters.

The `messages` property defines the record of messages to send, as well as, defines the number of data bytes and an opcode for each message.

For example:

When `messages = {4, 255}`, one message will be sent having 4 bytes of data and an opcode of 255.

When `messages = {8, 251}, {12, 250}`, two messages will be sent, the first having 8 bytes of data and an opcode of 251, and the second message having 12 bytes of data and an opcode of 250.

Data to be sent with a message is defined by the `data` property and is referred to as the data buffer. The number of data words in the data buffer is the number of data bytes for the messages.

The component offers an additional feature when there are multiple messages via the `dataRepeat` property which indicates whether the a message starts at the beginning of the data buffer, or continues from its current index within the buffer.

For example:

Given `messages = {4, 251},{8, 252},{12, 253},{16, 254},{20, 255}`

If `dataRepeat = true`, then `numDataWords` is 5. To calculate the `numDataWords` when `dataRepeat` is true, divide the largest message size (in bytes) by 4. Dividing by four required because the data is output as a 4 byte data word. Since the largest message size in the given messages assignment is 20, $20/4 = 5$.

When `numDataWords = 5`, then a valid data assignment would be `data = {0, 1, 2, 3, 4}`, and the data within each message would look like: `msg1 = {0}`, `msg2 = {0, 1}`, `msg3 = {0, 1, 2}`, `msg4 = {0, 1, 2, 3}`, `msg5 = {0, 1, 2, 3, 4}`

If `dataRepeat = false`, then `numDataWords` is 15. To calculate the `numDataWords` when `dataRepeat` is false, divide the sum of all the message sizes (in bytes) by 4. Dividing by four is required because the data is output as a 4 byte data word. Since the sum of all message sizes in the given messages assignment is $(4+8+12+16+20)/4 = 15$.

When `numDataWords = 15`, then a valid data assignment would be `data = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14}`, and the data within each message would look like: `msg1 = {0}`, `msg2 = {1, 2}`, `msg3 = {3, 4, 5}`, `msg4 = {6, 7, 8, 9}`, `msg5 = {10, 11, 12, 13, 14}`

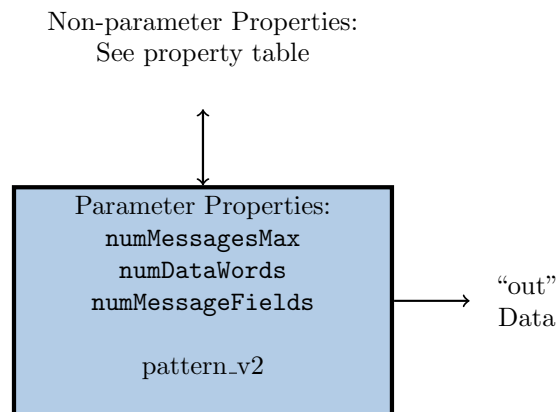
There is also a `messagesToSend` property that sets the number of messages to send and decrements as the messages are sent. When `dataRepeat` is true, `messagesToSend` is greater than `numMessagesMax`, and at the end of the messages buffer, the buffer wraps around and starts at the beginning of the messages buffer. When `dataRepeat` is

false, this value must be less than or equal to `numMessagesMax`. The HDL and RCC worker will check for this and report an error if `messagesToSend` is greater than `numMessagesMax`. The error checking for the HDL worker only happens in simulation.

When using `pattern_v2.hdl`, the `messagesToSend`, `messagesSent`, and `dataSent` properties should be checked at the end of an app run because they won't be stable until then. The worker doesn't use cdc crossing circuits for them because it takes advantage that they will have a stable value by the time the control plane reads those values at the end of an app run.

Block Diagrams

Top level



Source Dependencies

`pattern_v2.hdl`

- `assets/components/util_comps/pattern_v2.hdl/pattern_v2.vhd`
- `core/hdl/primitives/util/BRAM2.v`
- `core/hdl/primitives/util/util_pkg.vhd`

`pattern_v2.hdl`

- `assets/components/util_comps/pattern_v2.rcc/pattern_v2.cc`

Component Spec Properties

Name	Type	Default	SequenceLength	ArrayLength	ArrayDimensions	Parameter	Accessibility	Usage
dataRepeat	bool	false	-	-	-	false	Initial	True - Multiple messages sent from the beginning of the data buffer. False - Multiple messages sent from the current position of the data buffer.
numMessagesMax	uLong	5	-	-	-	true	-	Max number of messages to send.
messagesToSend	uLong	5	-	-	-	false	Volatile, Initial	Counter of messages to send and decrements as they are sent. When dataRepeat is true, messagesToSend greater than numMessagesMax, and at the end of the messages buffer, the buffer wraps around and starts at the beginning of the messages buffer. When dataRepeat is false, this value must be less than or equal to numMessagesMax.
messagesSent	uLong	-	-	-	-	false	Volatile	Messages sent counter. Initialized to 0.
dataSent	uLong	-	-	-	-	false	Volatile	Words sent counter. Initialized to 0.
numDataWords	uLong	15	-	-	-	true	-	Max number of four byte data for the data buffer. To calculate the numDataWords when dataRepeat is true, divide the largest message size (in bytes) by 4. To calculate the numDataWords when dataRepeat is false, divide the sum of all the message sizes (in bytes) by 4. Dividing by four required because the data is output as a 4 byte data word.
numMessageFields	uLong	2	-	-	-	true	-	Due to a limitation, cannot use constrained elements in unconstrained array declarations, so cannot directly set the second dimension for the messages property to 2. The numMessageFields property must always be 2 since there are 2 message fields; the number of data bytes (bytes that are a multiple of 4) and opcode. So the default value must not be changed.
messages	uLong	-	-	-	numMessagesMax, numMessageFields	false	Initial	Multidimensional array that defines the record of messages to send, as well as, defines the number of data bytes (bytes that are a multiple of 4) and an 8 bit opcode for each message.
data	uLong	-	-	numDataWords	-	false	Initial	Data buffer containing the data to be sent.

Component Ports

Name	Protocol	Producer	Optional	Usage
out	-	true	false	Data generated by the component

Worker Interfaces

pattern_v2.hdl

Type	Name	DataWidth (b)	Advanced	Usage
StreamInterface	out	32	DataValueWidth=8, NumberOfOpcodes='256', ZeroLengthMessages=true, clockDirection="in"	Data generated by the worker

Control Timing and Signals

The pattern_v2 worker uses the clock from the Data Plane and Control Plane. And it uses standard Control Plane signals.

Worker Configuration Parameters

pattern_v2.hdl

Performance and Resource Utilization

pattern_v2.hdl

Test and Verification

The pattern_v2 worker is tested by generating data for the **messages** and **data** properties and verifying that final value of the volatile properties and the output data are correct. Since the pattern_v2 worker's **messages** and **data** property array sizes depend on parameters, they have to be generated via scripts(**gen_messages.py** and **gen_data.py**)

Applications

For an example of the pattern_v2 component used in an application, please reference the tb_bias_v2 application located in **assets/applications/tb_bias_v2**.