Debug / Width

Parameterization

Shepard Siegel

Atomic Rules LLC

© 2011 Atomic Rules LLC

# Overview

OpenCPI HDL components may be written to parameterize aspects of their implementation in a standardized fashion. This document describes how this has been accomplished for debug and width.

# Consistent Parameter Names

The use of consistent parameter names reduces the chance of error and allows for reuse and automation. By way of example, the module parameters for the SMAdapter.v code is shown and each parameter explained:

parameter integer HAS\_DEBUG\_LOGIC = 1

parameter integer WMI\_M0\_DATAPATH\_WIDTH = 32

parameter integer WSI\_S0\_DATAPATH\_WIDTH = 32

parameter integer WSI\_M0\_DATAPATH\_WIDTH = 32

parameter integer WORKER\_CTRL\_INIT = 1

The first entry, HAS\_DEBUG\_LOGIC, is an integer that used to indicate of the module does (1) or does not (0) have debug logic instantiated. This parameter would more correctly be a Boolean; however many Verilog simulators and synthesis tools do not support this SystemVerilog type. Instead, integer 1 indicates True; and 0 indicates False.

The next three entries are of the form XXX\_YY\_DATAPATH\_WIDTH and describe the width of this module’s data flow interfaces in bits. XXX calls out either WMI or WSI, one of the two OpenCPI WIP data flow interfaces. YY calls out if a particular interface is either a Master (M) or Slave (S) and its ordinal ID starting from 0. In this case all ordinal IDs are 0. If there was an additional WMI Master interface, for example, it would be WMI\_M1\_DATAPATH\_WIDTH.

Lastly, an ad hoc parameter chosen by the module author is shown. So long as it does not collide with the OpenCPI patterns previously described any string may be used the module author to parameterize their IP.

# Individual Module Results

Individual results are shown for debug/width sweep for the following modules BiasWorker, SMAdapter, DelayWorker, and OCDP.

## BiasWorker

Parameterized Width and Debug

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Width** | **Debug** | **FMax (MHz)** | **DFFs** | **6-LUTs** |
| 32 | 0 | 360 | 354 | 493 |
| 32 | 1 | 381 | 567 | 842 |
| 64 | 0 | 337 | 462 | 686 |
| 64 | 1 | 310 | 673 | 974 |
| 128 | 0 | 261 | 678 | 983 |
| 128 | 1 | 261 | 887 | 1300 |
| 256 | 0 | 166 | 1110 | 1683 |
| 256 | 1 | 166 | 1319 | 1952 |

Test Conditions:

ISE 13.3, XST defaults, Virtex6

## SMAdapter

Parameterized Width and Debug

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Width** | **Debug** | **FMax (MHz)** | **DFFs** | **6-LUTs** |
| 32 | 0 | 270 | 819 | 1638 |
| 32 | 1 | 277 | 1197 | 2075 |
| 64 | 0 | 278 | 1099 | 2101 |
| 64 | 1 | 277 | 1478 | 2280 |
| 128 | 0 | 259 | 1659 | 2365 |
| 128 | 1 | 258 | 2034 | 3080 |
| 256 | 0 | 222 | 2775 | 4035 |
| 256 | 1 | 222 | 3151 | 4365 |

Test Conditions:

ISE 13.3, XST defaults, Virtex6

## DelayWorker

Parameterized Width and Debug

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Width** | **Debug** | **FMax (MHz)** | **DFFs** | **6-LUTs** |
| 32 | 0 | 244 | 2132 | 3478 |
| 32 | 1 | 255 | 2531 | 4032 |
| 64 | 0 | 253 | 2248 | 3652 |
| 64 | 1 | 255 | 2649 | 4223 |
| 128 | 0 | 256 | 2463 | 4107 |
| 128 | 1 | 255 | 2864 | 4557 |
| 256 | 0 | 246 | 2894 | 4651 |
| 256 | 1 | 255 | 3295 | 5230 |

Test Conditions:

ISE 13.3, XST defaults, Virtex6

## OCDP

Parameterized Width and Debug

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Width** | **Push, Pull, Debug** | **FMax (MHz)** | **DFFs** | **6-LUTs** |
| 32 | 001 | 234 | 2681 | 4359 |
| 32 | 011 | 205 | 3209 | 5104 |
| 32 | 101 | 205 | 3227 | 5313 |
| 32 | 111 | 183 | 3360 | 5995 |
| 64 | 001 | 234 | 2718 | 4283 |
| 64 | 011 | 190 | 3252 | 5168 |
| 64 | 101 | 210 | 3315 | 5089 |
| 64 | 111 | 186 | 3399 | 5811 |
| 128 |  |  |  |  |
| 256 |  |  |  |  |

Test Conditions:

ISE 13.3, XST defaults, Virtex6

# Full Chip Build Results

It is possible to pass parameters such as debug and width to some or all modules in the design hierarchy. There are several ways this can be achieved; two of them are listed here:

1. The use of `define pre-processor.
2. The use of a static assignment to convey the parameter value.

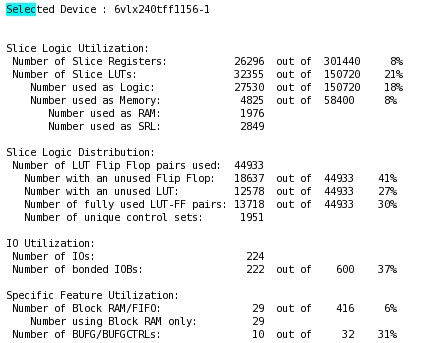
Other techniques are also permissible so long as they conform to the Verilog 1364-2001 specification.

Parameters represent constants; hence, it is illegal to modify their value at runtime. However, module parameters can be modified at compilation time to have values that are different from those specified in the declaration assignment. This allows customization of module instances. [1]

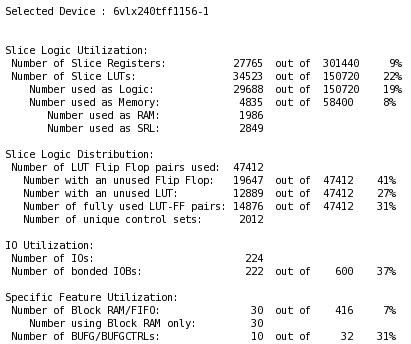
By globally switching the HAS\_DEBUG\_LOGIC parameters it is possible to easily produce a design that globally has (or doesn’t have) debug logic inserted. The next page shows the comparison for a ML605 build with a 32bit wide datapath. It was tested to be equally operational with all available regression software including both testRpl and NFT.

Unsurprisingly, the chip implementation without debug logic is smaller. In this case it used about 1 000 fewer DFFs and 6-LUTs. The synthesis report does not show what gain (or loss) this had with regard to timing closure as both design closed timing.

## With HAS\_DEBUG\_LOGIC = 0



## With HAS\_DEBUG\_LOGIC = 1



References

|  |  |
| --- | --- |
| **ID** | **Document Name** |
| 1 | IEEE Standard Verilog Hardware Description Language, March 2001 IEEE Std 1364-2001 |
| 2 | IEEE Standard VHDL Language Reference Manual, IEEE Std 1076-1993, 2002 |