IOB-CACHE

User Guide, 0.1 , Build 30074a8

April 12, 2022

Contents

1 Introduction **5**

2 Symbol **5**

3 Features **5**

4 Benefits **5**

5 Deliverables **5**

6 Block Diagram and Description **6**

7 Interface Signals **6**

8 Instantiation and External Circuitry **7**

9 Simulation **9**

10 Synthesis **10**

10.1 Synthesis Macros and Parameters 10

10.2 Synthesis Script and Timing Constraints 10

11 Implementation Results **10**

12 Configuration Parameters **10**

13 Registers **12**

List of Tables

1 General Interface Signals 6

2 IObundle Master Interface Signals 7

3 Control-Status Interface Signals 7

4 IObundle Slave Interface Signals 7

5 FPGA results for Kintex Ultrascale (left) and Cyclone V GT (right). 10

6 Configurable Parameters 11

List of Figures

1 IP core symbol. 5

2 High-level block diagram. 6

3 Core instance and required surrounding blocks 8

4 Testbench block diagram 9

1 Introduction

The IObundle CACHE is ....

2 Symbol

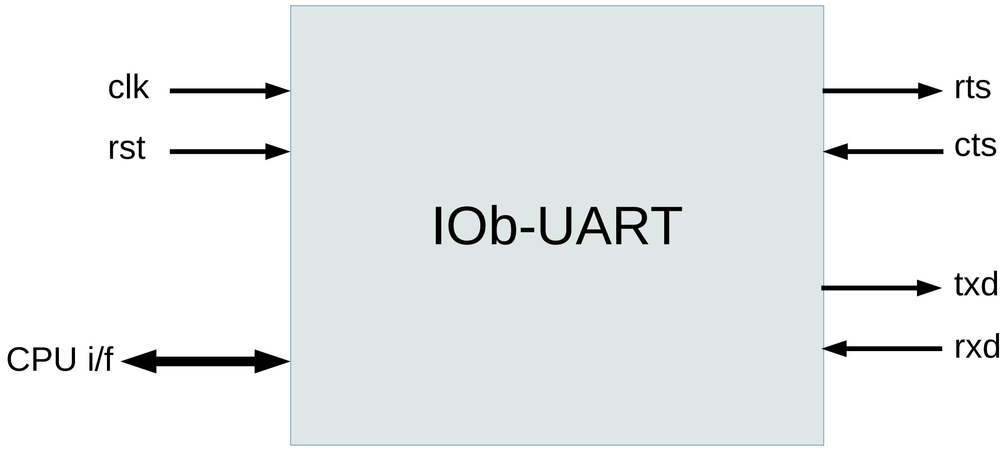

Figure 1: IP core symbol.

3 Features

- feature1
- feature2

4 Benefits

- Compact and easy to integrate hardware and software implementation
- Can fit many instances in low cost FPGAs and ASICs
- Low power consumption

5 Deliverables

- ASIC or FPGA synthesized netlist or Verilog source code, and respective synthesis and implementation scripts
- ASIC or FPGA verification environment by simulation and emulation
- Bare-metal software driver and example user software
- User documentation for easy system integration
- Example integration in IOb-SoC (optional)

6 Block Diagram and Description

A high-level block diagram of the core is presented in Figure 2, followed by a brief description of each of the blocks.

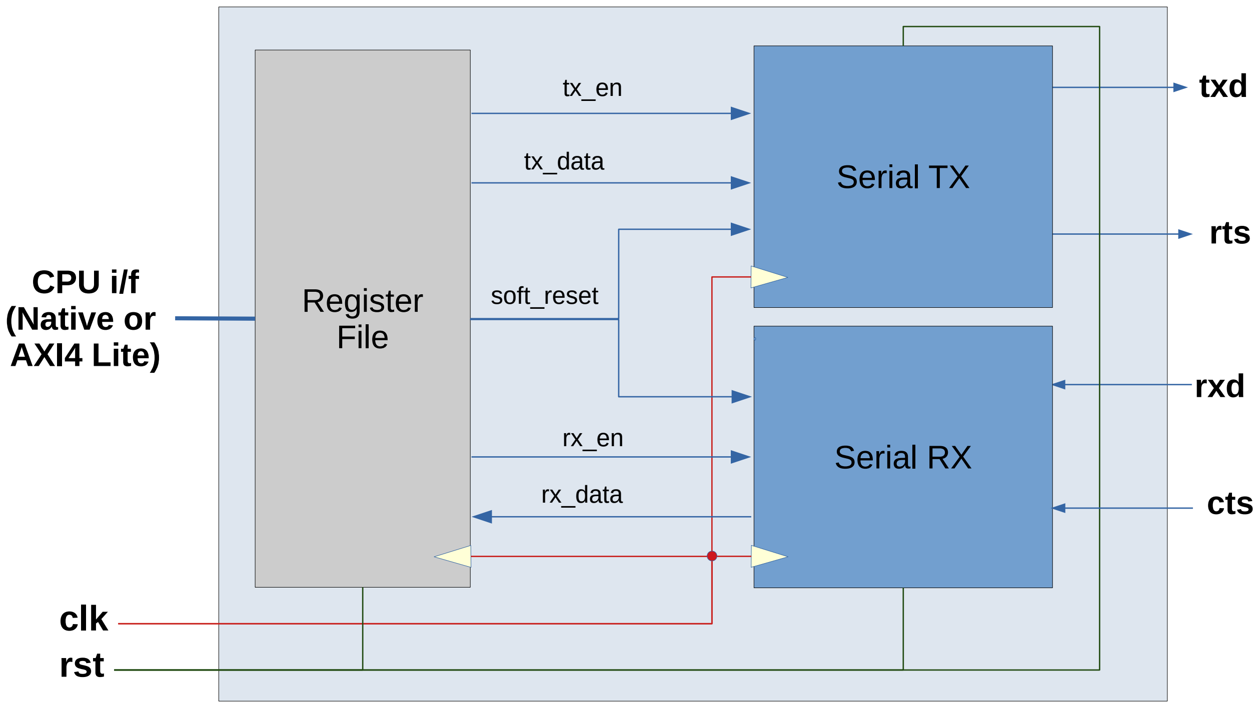

Figure 2: High-level block diagram.

FRONT-END Front-end block.

CACHE MEMORY Cache memory block.

BACK-END Back-end block.

CACHE CONTROL Cache control block.

7 Interface Signals

Name	Direction	Width	Description
clk	INPUT	1	System clock input
reset	INPUT	1	System reset, asynchronous and active high

Table 1: General Interface Signals

Name	Direction	Width	Description
valid	INPUT	1	Native CPU interface valid signal
addr	INPUT	CTRL_CACHE + FE_ADDR_W - FE_BYTE_W	Native CPU interface address signal
addr	INPUT	CTRL_CACHE + FE_ADDR_W	Native CPU interface address signal
wdata	INPUT	FE_DATA_W	Native CPU interface data write signal
wstrb	INPUT	FE_NBYTES	Native CPU interface write strobe signal
rdata	OUTPUT	FE_DATA_W	Native CPU interface read data signal
ready	OUTPUT	1	Native CPU interface ready signal

Table 2: IObundle Master Interface Signals

Name	Direction	Width	Description
force_inv_in	INPUT	1	force 1'b0 if unused
force_inv_out	OUTPUT	1	cache invalidate signal
wtb_empty_in	INPUT	1	force 1'b1 if unused
wtb_empty_out	OUTPUT	1	write-through buffer empty signal

Table 3: Control-Status Interface Signals

Name	Direction	Width	Description
mem_valid	OUTPUT	1	Native CPU interface valid signal
mem_addr	OUTPUT	BE_ADDR_W	Native CPU interface address signal
mem_wdata	OUTPUT	BE_DATA_W	Native CPU interface data write signal
mem_wstrb	OUTPUT	BE_NBYTES	Native CPU interface write strobe signal
mem_rdata	INPUT	BE_DATA_W	Native CPU interface read data signal
mem_ready	INPUT	1	Native CPU interface ready signal

Table 4: IObundle Slave Interface Signals

8 Instantiation and External Circuitry

Figure 4 illustrates how to instantiate the IP core and, if apllicable, the required external blocks. A Verilog file describing this setup is provided.

bla bla bla...

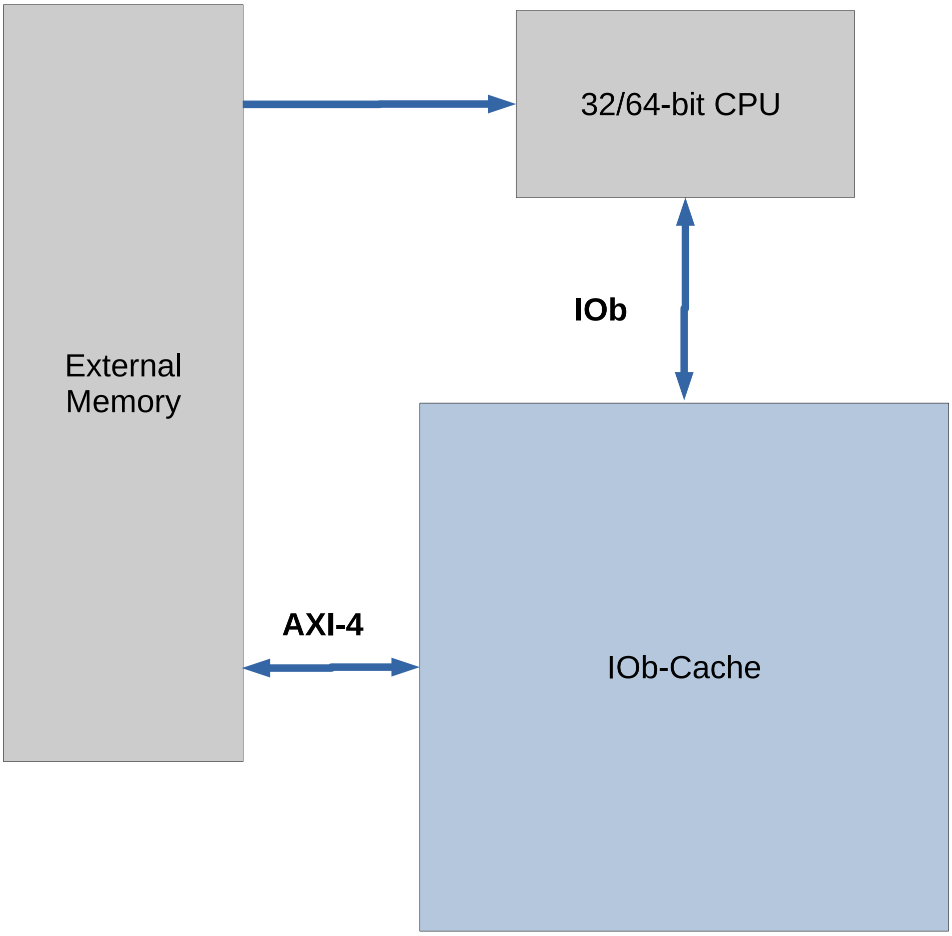

Figure 3: Core instance and required surrounding blocks

9 Simulation

The provided testbench uses the core instance described in Section 8. A high level block diagram of the testbench is shown in Figure 4. The testbench is organised in modular fashion with each test described in a separate file. The test suite consists of all the test case files, so that it becomes easy to add, modify or remove tests.

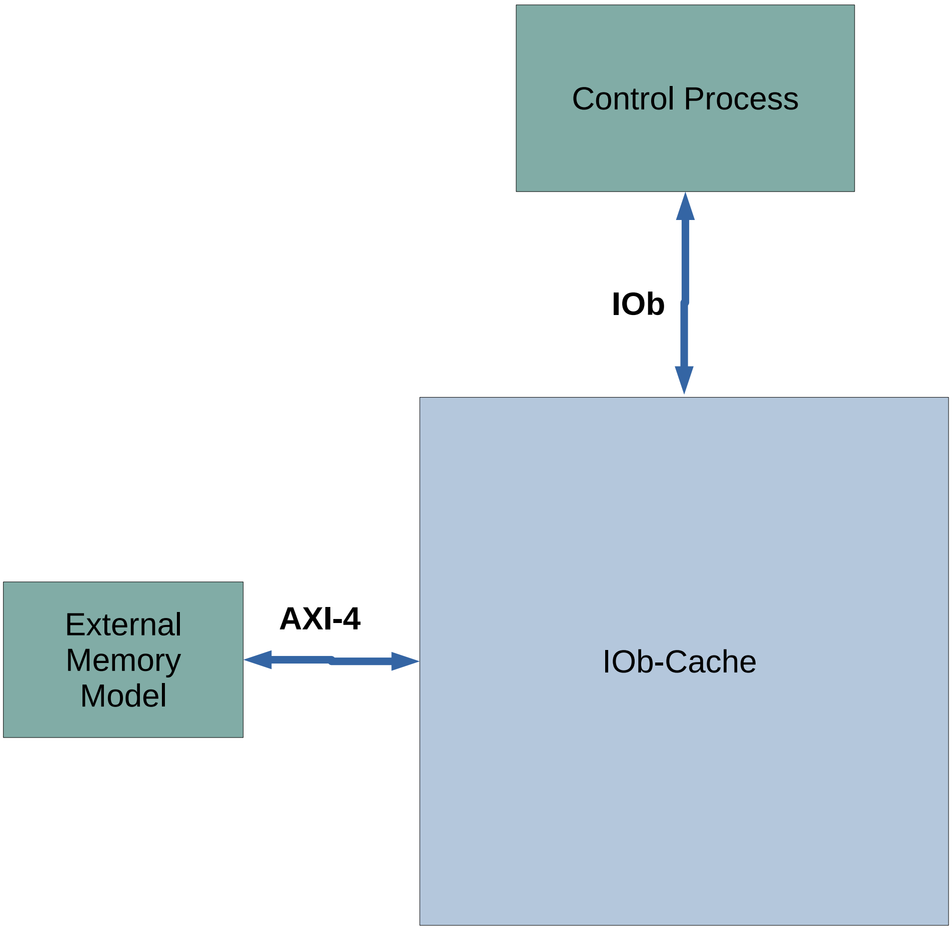

Figure 4: Testbench block diagram

In this preliminary version, simulation is not yet fully functional. The provided testbench merely allows compilation for simulation, and drives the clock and reset signals. Behavioural memory models to allow pre-synthesis simulation are already included. In the case of ROMs, their programming data is also included in the form of .hex files.

10 Synthesis

10.1 Synthesis Macros and Parameters

The synthesis macros and parameters of the core are presented in this section. For each macro or parameter, the minimum, default and maximum values are provided. The minimum and maximum values may be marked with

Not Applicable(NA): the macro or parameter value cannot be set by the user; its value is set automatically by the system.

Not Specified (NS): the minimum or maximum value for the macro or parameter has not been determined; if needed, values other than the default should be determined and tested by the user.

10.2 Synthesis Script and Timing Constraints

A simple `.tcl` script is provided for the Cadence Genus synthesis tool. The script reads the technology files, compiles and elaborates the design, and proceeds to synthesise it. The timing constraints are contained within the constraints file provided, or provided in a separate file.

After synthesis, reports on silicon area usage, power consumption, and timing closure are generated. A post-synthesis Verilog file is created, to be used in post-synthesis simulation.

In this preliminary version, synthesis of the IP core without the memories is functional. The memories are for now treated as black boxes.

It is important not to include the memory models provided in the simulation directory in synthesis, unless they are to be synthesised as logic. In a next version, the memories will be generated for the target technology and included.

11 Implementation Results

The following are FPGA implementation results for two FPGA families. The following are FPGA implementation results for two FPGA families.

Resource	Used
LUTs	2168
Registers	1227
DSPs	0
BRAM	0

Resource	Used
ALM	826
FF	610
DSP	0
BRAM blocks	68
BRAM bits	72,768

Table 5: FPGA results for Kintex Ultrascale (left) and Cyclone V GT (right).

12 Configuration Parameters

Name	Value	Description
FE_ADDR_W	32	Address width - width of the Master's entire access address (including the LSBs that are discarded, but discarding the Controller's)
FE_DATA_W	32	Data width - word size used for the cache
N_WAYS	2	Number of Cache Ways (Needs to be Potency of 2: 1, 2, 4, 8, ..)
LINE_OFF_W	7	Line-Offset Width - 2**NLINE_W total cache lines
WORD_OFF_W	3	Word-Offset Width - 2**OFFSET_W total FE_DATA_W words per line - WARNING about LINE2MEM_W (can cause word_counter [-1:0]
WTBUF_DEPTH_W	5	Depth Width of Write-Through Buffer
REP_POLICY	PLRU_mru	LRU - Least Recently Used; PLRU_mru (1) - mru-based pseudoLRU; PLRU_tree (3) - tree-based pseudoLRU
NWAY_W	clog2(N_WAYS)	Cache Ways Width
FE_NBYTES	FE_DATA_W/8	Number of Bytes per Word
FE_BYTE_W	clog2(FE_NBYTES)	Byte Offset
BE_ADDR_W	FE_ADDR_W	Address width of the higher hierarchy memory
BE_DATA_W	FE_DATA_W	Data width of the memory
BE_NBYTES	BE_DATA_W/8	Number of bytes
BE_BYTE_W	clog2(BE_NBYTES)	Offset of Number of Bytes
LINE2MEM_W	WORD_OFF_W-clog2(BE_DATA_W/FE_DATA_W)	Logarithm Ratio between the size of the cache-line and the BE's data width
WRITE_POL	WRITE_THROUGH	write policy: write-through (0), write-back (1)
CTRL_CACHE	0	Adds a Controller to the cache, to use functions sent by the master or count the hits and misses
CTRL_CNT	0	Counters for Cache Hits and Misses - Disabling this and previous, the Controller only store the buffer states and allows cache invalidation

Table 6: Configurable Parameters

13 Registers

