# IOB-CACHE

## User Guide, 0.1 , Build f9d8890


May 1, 2022

  

# Contents

**1 Introduction** **5**

**2 Symbol** **5**

**3 Features** **5**

**4 Benefits** **5**

**5 Deliverables** **6**

**6 Block Diagram and Description** **6**

**7 Interface Signals** **7**

**8 Instantiation and External Circuitry** **7**

**9 Simulation** **9**

**10 Synthesis** **10**

10.1 Synthesis Macros . . . . . 10

10.2 Synthesis Parameters . . . . . 10

10.3 Synthesis Script and Timing Constraints . . . . . 10

**11 Macros and Configuration Parameters** **12**

**12 Registers** **12**

# List of Tables

1 General Interface Signals . . . . . 7

2 IObundle Master Interface Signals . . . . . 7

3 Control-Status Interface Signals . . . . . 7

4 IObundle Slave Interface Signals . . . . . 7

 

5 Synthesis Macros. . . . . 10

6 Synthesis 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 an open source pipelined-memory cache. It is a performance-wise and highly configurable IP core. The cache core is isolated from the processor and memory interfaces in order to make it easy to adopt new processors or memory controllers while keeping the core functionality intact. It implements a simple front-end native interface. It also implements an AXI4 interface with configurable data width which allows maximum use of the available memory bandwidth. The IObundle CACHE can be implemented as a Direct-Mapped cache or K-Way Set-Associative cache. It supports both fixed write-through not-allocate policy and write-back policy.

# 2 Symbol

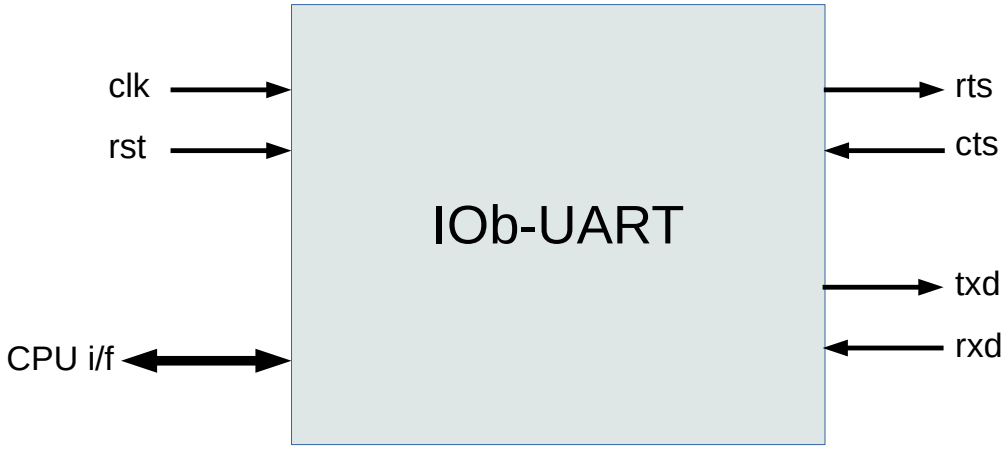


Figure 1: IP Core Symbol.

# 3 Features

- AXI4 interface with configurable data width
- Simple front-end native interface
- Direct-Mapped or K-Way Set-Associative
- Fixed write-through not-allocate policy
- Write-back policy
- Pipelined-memory (1 request/clock-cycle)

# 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

Figure 2 presents a high-level block diagram of the core, followed by a brief description of each block.

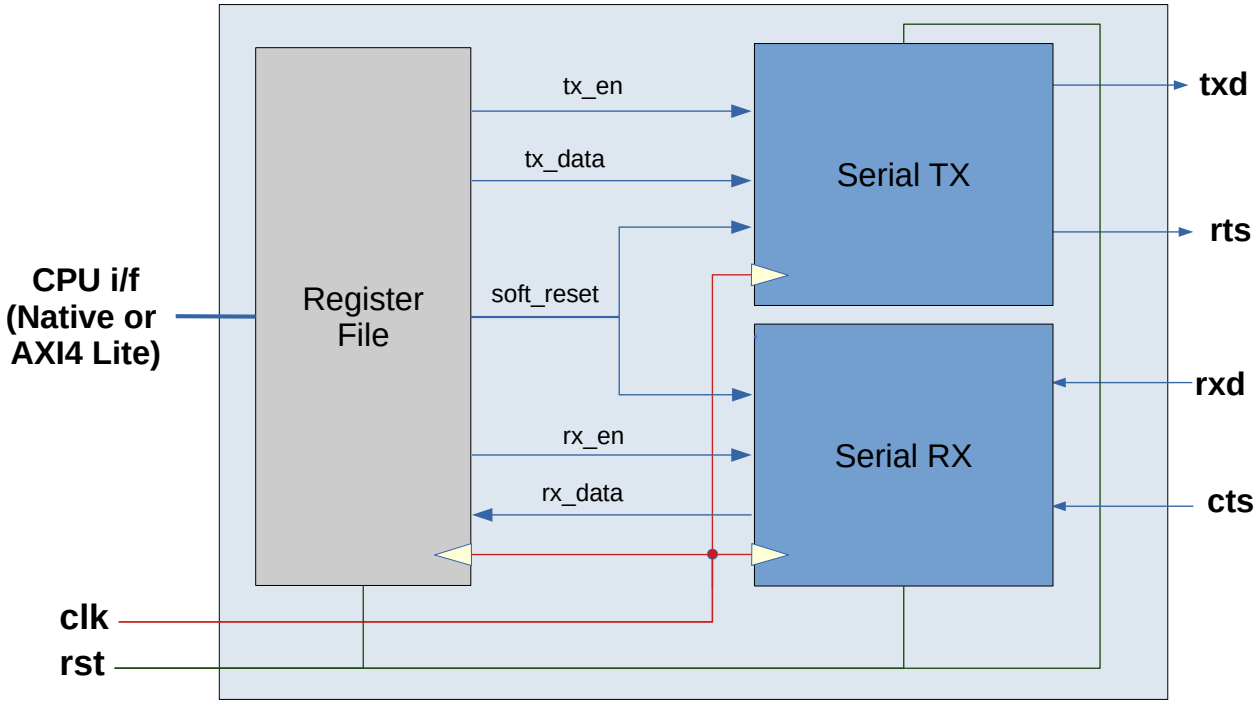


Figure 2: High-Level Block Diagram.

**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 applicable, the required external blocks. A Verilog instantiation template is provided for convenience.

bla bla bla...

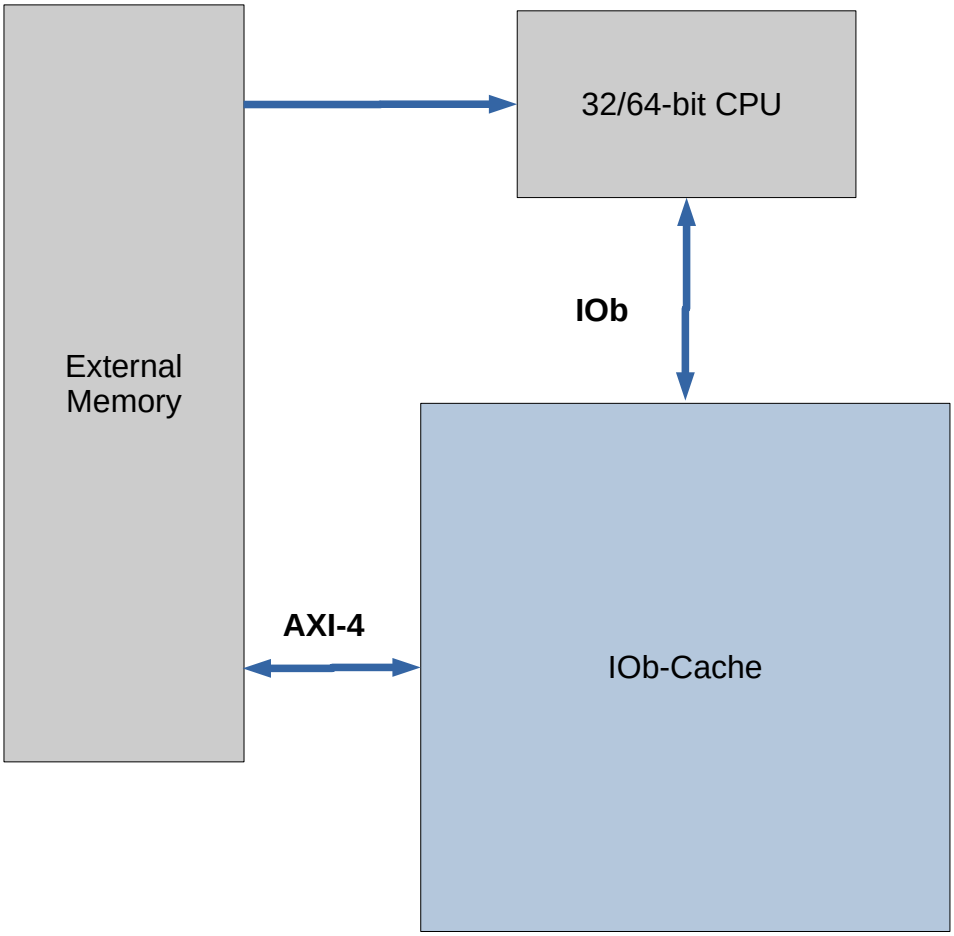


Figure 3: Core Instance and Required Surrounding Blocks

# 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 organized in a modular fashion, with each test described in a separate file. The test suite consists of all the test case files to make adding, modifying, or removing tests easy.

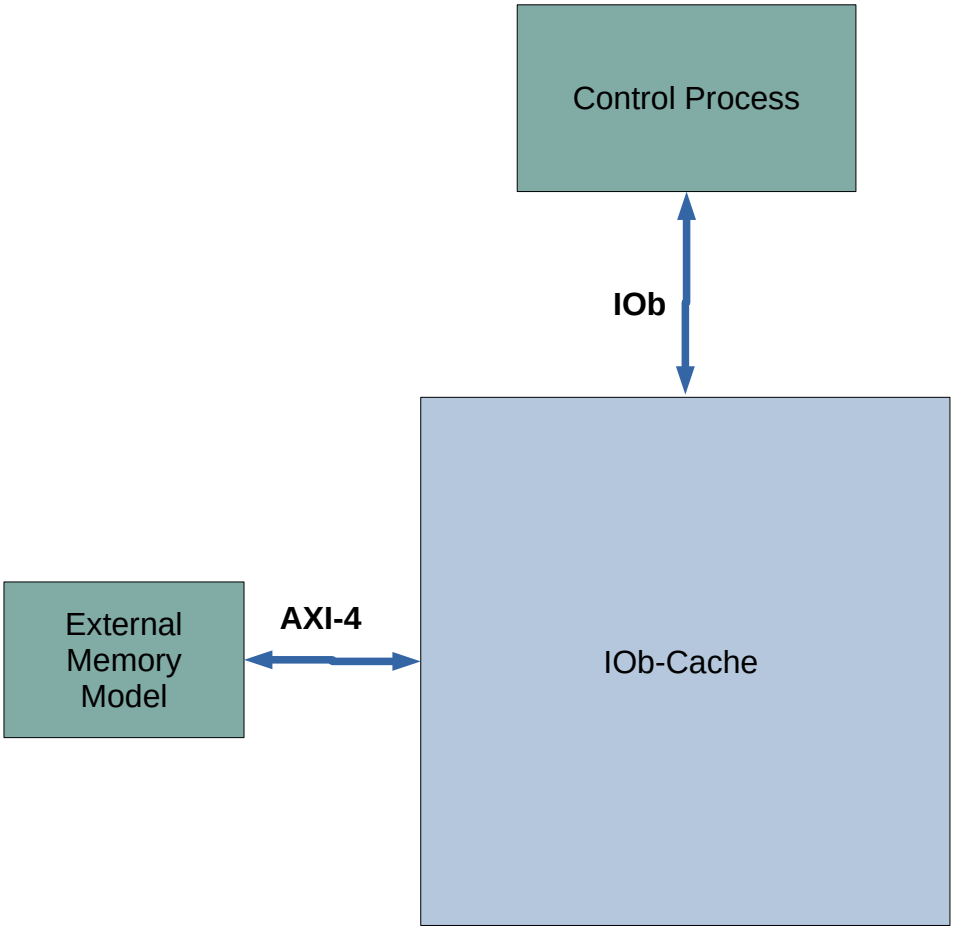


Figure 4: Testbench Block Diagram

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

The synthesis macros apply to all instances of the core, and are listed in Table 5.

| Parameter | Min | Typ | Max | Description |
|---|---|---|---|---|
| WRITE_POL | ? | WRITE_THROUGH | ? | write policy: write-through (0), write-back (1) |

Table 5: Synthesis Macros.

## 10.2 Synthesis Parameters

The generic synthesis parameters of the core are presented in Table 6. Generic parameters can vary from instance to instance.

## 10.3 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.

| Parameter | Min | Typ | Max | 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 |
| 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 - |

# 11 Macros and Configuration Parameters

# 12 Registers

  