Datasheet: Trencadís Register File

Version: v1.0.1

Last Updated: 2025-08-16

1. Overview

The `trencadis_register_file` is a flexible, synthesizable SystemVerilog module that implements a high-performance register file with one synchronous write port and a parameterizable number of asynchronous read ports. It is designed to be a core component in processor designs, particularly those requiring concurrent data access, such as in the execution stage of a pipelined CPU. Key features include a configurable register count and data width, and an optional mode to enforce that register 0 is hardwired to zero, making it directly compatible with the RISC-V integer instruction set architecture (ISA).

2. Features

- Single, synchronous write port.
- Parameterizable number of asynchronous read ports.
- Parameterizable register count (`REG_COUNT`) and data width (`DEPTH`).
- Optional RISC-V compatible zero-register functionality (register at address 0 is always zero).
- Fully synchronous design with a single clock domain for writes.
- Active-low asynchronous reset to initialize all registers to zero.
- Standard "read-before-write" behavior for simultaneous read/write to the same address.

3. Block Diagram

A conceptual block diagram is shown below. The number of read ports is determined by the `NUM_READ_PORTS` parameter.

Parametrable register file read ports

4. Parameters (Generics)

Parameter	Type	Default	Description
`NUM_READ_PORTS`	`integer`	`2`	Defines the number of concurrent read ports.
`REG_COUNT`	`integer`	`32`	Defines the total number of registers in the file.
`DEPTH`	`integer`	`32`	Defines the bit width of each individual register.
`ZERO_REG_IS_ZERO`	`bit`	`1`	If `1`, register at address `0` i s hardwired to zero. Writes to address `0` are ignored. If `0`, register `0` is a normal register.

5. Port Descriptions

Port Name	Direction	Width	Description
`clk_i`	`input`	`1`	System clock. All synchronous write

Port Name	Direction	Width	Description
			logic is clocked on the positive edge of this signal.
`rst_ni`	`input`	`1`	Active-low asynchronous system reset. When asserted (`0`), all registers are cleared to zero.
`waddr_i`	`input`	`$clog2(REG_COUNT)`	Write address. Selects the register to be written to.
`wdata_i`	`input`	`DEPTH`	Write data. The data to be written into the selected register.
`wen_i`	`input`	`1`	Write enable. A high level on this signal enables a write operation on the next positive clock edge.
`raddr_i`	`input`	`[NUM_READ_PORTS-1:0][$clog2(REG_COUNT)-1:0]`	Packed array of read addresses. `raddr_i[n]` is the address for the n-th read port.
`rdata_o`	`output`	`[NUM_READ_PORTS-1:0][DEPTH-1:0]`	Packed array of read data. `rdata_o[n]` is the data output from the n-th read port.

Clarity note on Packed Arrays: `raddr_i`: Packed array of read addresses. For the default `NUM_READ_PORTS=2`, this is a `logic [1:0][$clog2(REG_COUNT)-1:0]` signal. `rdata_` `o` : Packed array of read data ports. For the default `NUM_READ_PORTS=2`, this is a `logic [1:0][DEPTH-1:0]` signal.

6. Register Map*

* (This module is a simple logic core and does not contain a bus interface or register map.)

7. Functional Description

The `trencadis_register_file` module provides a simple and efficient memory structure commonly used in CPUs. Its operation is divided into three main functions: write, read, and reset.

Write Operation

A write operation is performed when the `wen` (write enable) signal is asserted high. On the next rising edge of `clk`, the data present on the `wdata` bus is written into the register selected by the `waddr` bus. The write is synchronous.

If the `ZERO_REG_IS_ZERO` parameter is set to `1`, any attempt to write to address `0` (`waddr == '0`) will be ignored, and the contents of register 0 will remain zero.

Read Operation

Read operations are asynchronous (combinatorial). The module supports `NUM_READ_PORTS` concurrent reads. For each read port `i`, the address on `raddr[i]` is used to select a register. The contents of that register are then immediately presented on the corresponding `rdata[i]` output bus.

If a read and a write occur to the same address in the same clock cycle, the read port will output the old data stored in the register before the write completes. This is standard "read-before-write" behavior.

If the `ZERO_REG_IS_ZERO` parameter is set to `1`, any read from address `0` (`raddr[i] == '0`) will result in `rdata[i]` being driven to all zeros, regardless of the physical value stored in register 0.

Reset

The module uses an active-low asynchronous reset (`rst_n`). When `rst_n` is pulled low, all physical registers within the file are immediately and asynchronously set to zero.

8. Timing Diagrams

This diagram shows a reset condition demonstrating that the output ports go to zero regardless of the clock

Asyncronous reset

This diagram shows a simultaneous read and write to the same register address. Note that `rdata` reflects the value of the register before the write operation completes on the next rising clock edge.

Read after write

9. Instatiation Template

Here is an example of how to instantiate the `trencadis_register_file` in SystemVerilog:

```
trencadis_register_file #(
    .NUM_READ_PORTS(2),
    .REG_COUNT(32),
    .DEPTH(32),
    .ZERO_REG_IS_ZERO(1),
) i_trencadis_register_file (
    // generic ports
    .clk_i(clk),
    .rst_ni(rst_n),
    // write ports
    .wen_i(write_enable),
    .waddr_i(write_address),
    .wdata_i(write_data),
    // read ports
    .raddr_i(read_address),    // Packed array of type logic [1:0][4:0]
    .rdata_o(read_data)        // packed array of type logic [1:0][31:0]
);
```

10. Bus Wrappers*

*(This module is a simple logic core and does not have any standard bus wrappers.)

11. Revision History

A log of changes to this document and the corresponding RTL module.

Version	Date	Author(s)	Changes
`v1.0.0`	2025-08-04	Adrià Babiano Novella	Initial release of the datasheet.
`v1.0.1`	2025-08-16	Adrià Babiano Novella	Update image path references and typos in document.