YIFive SOC Specification

Version 0.1

**Author**

Dinesh Annayya

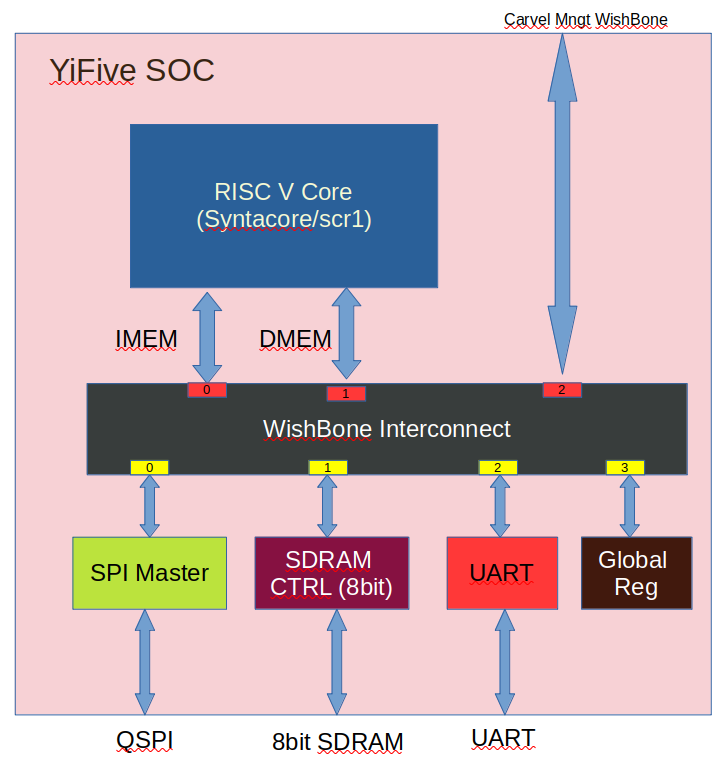
dinesha@opencores.org

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Owner | Comment |
| 0.0 | 10th June 2021 | Dinesh A | Initial Version |
| 0.1 | 14th June 2021 | Dinesh A | 1. Global Register Details 2. SPI Register Details 3. Memory Map Detail added |

# Overview

YiFive is a 32 bit RISC V based SOC design targeted for efabless Shuttle program. This project uses only open source tool set for simulation,synthesis and backend tools. The SOC flow follow the openlane methodology and SOC environment is compatible with efabless/carvel methodology.



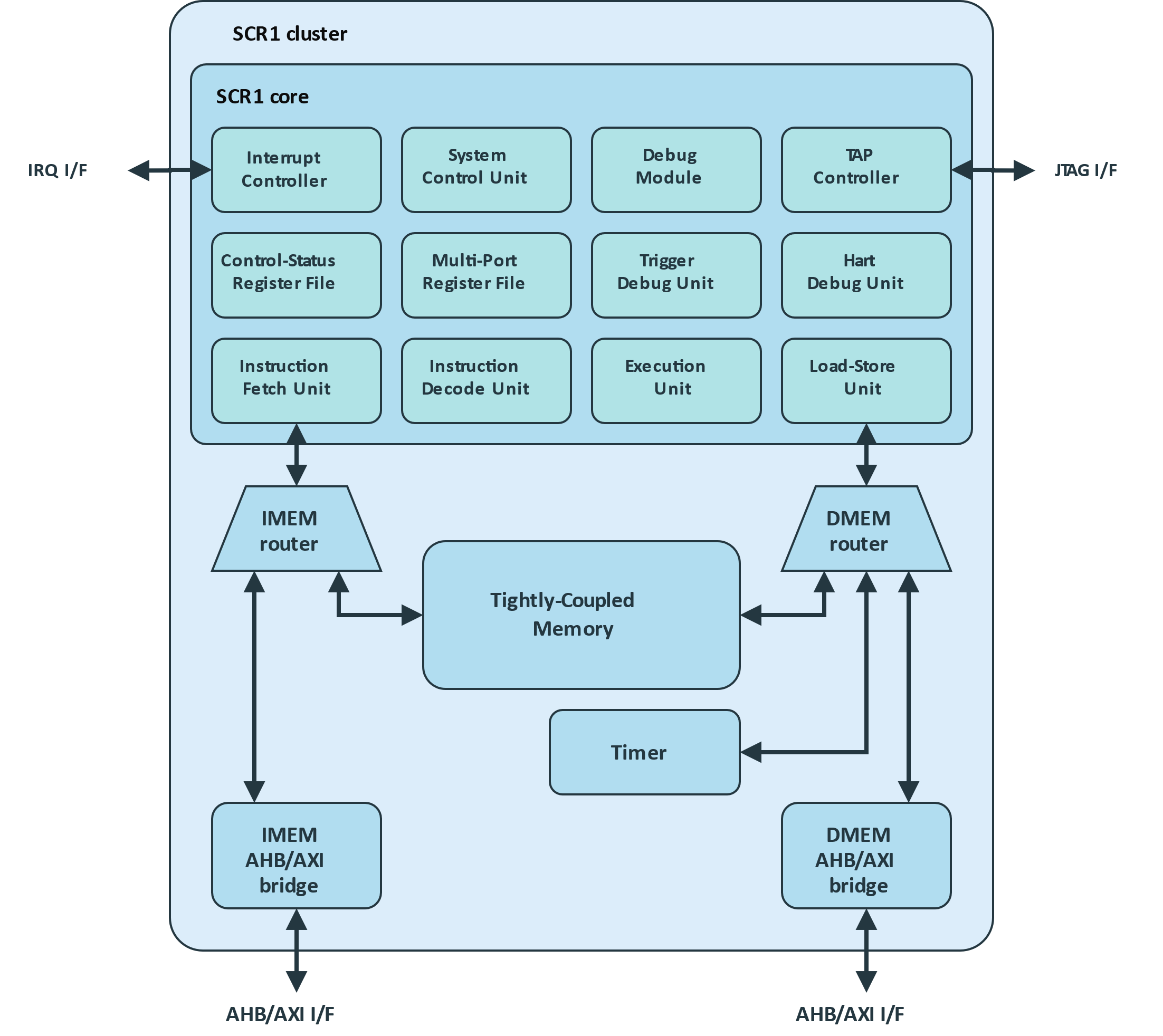
# Key features

* Open sourced under Apache-2.0 License (see LICENSE file) - unrestricted commercial use allowed.
* industry-grade and silicon-proven Open-Source RISC-V core from syntacore
* industry-graded and silicon-proven 8-bit SDRAM controller
* Quad SPI Master
* Wishbone compatible design
* Written in System Verilog
* Open-source tool set
* simulation - iverilog
* synthesis - yosys
* backend/sta - openlane tool set
* Verification suite provided.

# Sub IP features

## RISC V Core

YiFive SOC Integrated Syntacore SCR1 Open-source RISV-V compatible MCU-class core. It is industry-grade and silicon-proven IP. Git link: <https://github.com/syntacore/scr1>



### RISC V Core Key feature

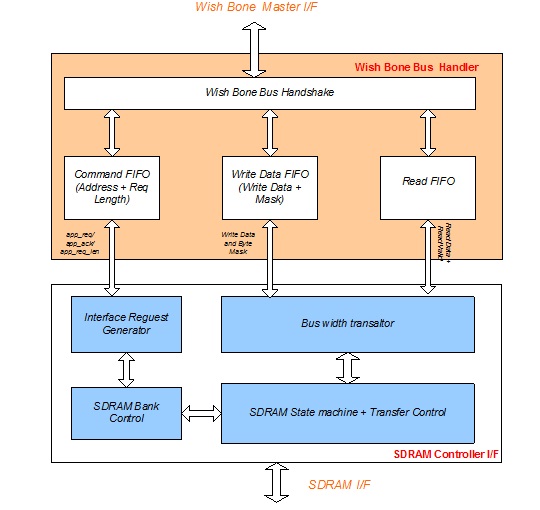
* RV32I or RV32E ISA base + optional RVM and RVC standard extensions
* Machine privilege mode only
* 2 to 4 stage pipeline
* Optional Integrated Programmable Interrupt Controller with 16 IRQ lines
* Optional RISC-V Debug subsystem with JTAG interface
* Optional on-chip Tightly-Coupled Memory

### RISC V core customization YiFive SOC

* **Update** : Modified some of the system verilog syntax to basic verilog syntax to compile/synthesis in open source tool like simulator (iverilog) and synthesis (yosys).
* **Modification**: Modified the AXI/AHB interface to wishbone interface towards instruction & data memory interface

## 8bit SDRAM Controller

Due to number of pin limitation in carvel shuttle, YiFive SOC integrate 8bit SDRAM controller. This is a silicon proven IP. IP Link: <https://opencores.org/projects/sdr_ctrl>



### SDRAM Controller key Feature

* + 8/16/32 Configurable SDRAM data width
  + Wish Bone compatible
  + Application clock and SDRAM clock can be async
  + Programmable column address
  + Support for industry-standard SDRAM devices and modules
  + Supports all standard SDRAM functions.
  + Fully Synchronous; All signals registered on positive edge of system clock.
  + One chip-select signals
  + Support SDRAM with four banks
  + Programmable CAS latency
  + Data mask signals for partial write operations
  + Bank management architecture, which minimizes latency.
  + Automatic controlled refresh

# SOC Pin Mapping

Carvel SOC provides 38 GPIO pins for user functionality. YiFive SOC GPIO Pin Mapping as follows

|  |  |  |  |
| --- | --- | --- | --- |
| **GPIO Pin Number** | **Direction** | **Pad Name** | **Block Name** |
| gpio[7:0] | Inout | SDRAM Data [7:0] | SDRAM |
| gpio[20:8] | Output | SDRAM Address [12:0] | SDRAM |
| gpio[22:21] | Output | SDRAM Bank Select [1:0] | SDRAM |
| gpio[23] | Output | SDRAM Byte Mask | SDRAM |
| gpio[24] | Output | SDRAM Write Enable | SDRAM |
| gpio[25] | Output | SDRAM CAS | SDRAM |
| gpio[26] | Output | SDRAM RAS | SDRAM |
| gpio[27] | Output | SDRAM Chip Select | SDRAM |
| gpio[28] | Output | SDRAM CKE | SDRAM |
| gpio[29] | Inout | SDRAM Clock | SDRAM |
| gpio[30] | Output | SPI Clock | SPI |
| gpio[31] | Output | SPI Chip Select | SPI |
| gpio[35:32] | Inout | SPI Data | SPI |
| gpio[36] | Output | Uart TX | UART |
| gpio[37] | Output | Uart RX | UART |

Memory Map

|  |  |  |  |
| --- | --- | --- | --- |
| RISC IMEM | RISC DMEM | EXT MAP | IP |
| 0x0000\_0000 to 0x0FFF\_FFFF | 0x0000\_0000 to 0x0FFF\_FFFF | 0x4000\_0000 to 0x4FFF\_FFFF | SPI FLASH MEMORY |
| 0x1000\_0000 to 0x1000\_00FF | 0x1000\_0000 to 0x1000\_00FF | 0x5000\_0000 to 0x5000\_00FF | SPI REGISTER |
| 0x2000\_0000 to 0x2FFF\_FFFF | 0x2000\_0000 to 0x2FFF\_FFFF | 0x6000\_0000 to 0x6FFF\_FFFF | SDRAM |
| 0x3000\_0000 to 0x3000\_00FF | 0x3000\_0000 to 0x3000\_00FF | 0x3000\_0000 to 0x3000\_00FF | GLOBAL REGISTER |

Register Documentation

Global Register:

Table: Global Control Register (0x0) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 0 | Cpu\_rst\_n | R/W | 0 | 0- Keep RISC core under reset  1- Remove RISC core from reset |
| 1 | Spi\_rst\_n | R/W | 0 | 0 - Keep SPI Master under reset  1 - Remove SPI Master from reset |
| 2 | Sdram\_rst\_n | R/W | 0 | 0 - Keep SDRAM under reset  1 - Remove SDRAM from reset |
| 3 | Sdram\_clk\_div\_enb | R/W | 0 | 0 – Bypass SDRAM clock division, same as system clock  1 – SDRAM clock will be Div-2 of system clock |
| 31:4 | Reserved | R |  |  |

Table: CPU device id (0x4) = 0xA55AA55A

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 31:0 | Device\_id | R/W | 0xA55AA55A | RISV Core Device ID |

Table: CPU fuse id (0x8) = 0xAABBCCDD

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 31:0 | Device\_id | R/W | 0xAAABBCCDD | RISV Core Fuse ID |

Table: CPU interrupt control (0xC) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 15:0 | Irq\_lines | R/W | 0x00 | RISC V Hardware Interrupts |
| 16 | Soft\_irq | R/W | 0x00 | RISC V Software Interrupts |

Table: SDRAM configuration1 (0x10) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 1:0 | cfg\_sdr\_width | R/W | 0x00 | 2'b00 - 32 Bit SDR, 2'b01 - 16 Bit SDR, 2'b1x - 8 Bit  Recommended value: 0x10 |
| 3:2 | cfg\_colbits | R/W | 0x00 | 00 - 8 Bit column address  01 – 9 bit column address  10 – 10 bit column address  11 – 11 bit column address  Recommended value: 0x0 |
| 7:4 | cfg\_sdr\_tras\_d | R/W | 0x00 | SDRAM active to precharge, specified in clocks; Recommended value: 0x4 |
| 11:8 | cfg\_sdr\_trp\_d | R/W | 0x00 | SDRAM active to precharge, specified in clocks; Recommended value: 0x2 |
| 15:12 | cfg\_sdr\_trcd\_d | R/W | 0x00 | SDRAM active to read or write delay (tRCD), specified in clocks. Recommended value: 0x2 |
| 19:16 | cfg\_sdr\_trcar\_d | R/W | 0x00 | SDRAM active to active / auto-refresh command period (tRC), specified in clocks.  Recommended value: 0x7 |
| 23:29 | cfg\_sdr\_twr\_d | R/W | 0x00 | SDRAM write recovery time (tWR), specified in clocks  Recommended value: 0x1 |
| 25:24 | cfg\_req\_depth | R/W | 0x00 | Maximum Request accepted by SDRAM controller.  Recommended value: 0x3 |
| 28:26 | cfg\_sdr\_cas | R/W | 0x00 | SDRAM CAS latency, specified in clocks  Recommended value: 0x3 |
| 29 | cfg\_sdr\_en | R/W | 0x0 | 1 – Enable SDRAM Controller |
| 30 | sdr\_init\_done | R | - | 1 – SDRAM Initialization done. |
| 30 | Reserved |  |  |  |

Table: SDRAM configuration2 (0x14 -0x3C) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 31:0 | cfg\_sdr\_rfmax | R/W | 0x00 | Maximum number of rows to be refreshed at a time (tRFSH). Recommended value: 0x6 |
| 15:3 | cfg\_sdr\_mode\_reg | R/W | 0x00 | SDRAM Mode Register.  Recommended value: 0x33 |
| 27:16 | cfg\_sdr\_rfsh | R/W |  | Period between auto-refresh commands issued by the controller, specified in clocks.  Recommended value: 0x100 |

Table: Software Reg (0x18 to 0x3C) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 31:0 | Software register | R/W | 0x00 | These software register useful for core to core communication |

SPI Master Register:

Table: SPI Global Control (0x00) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 0 | spi\_rd | R/W | 0x00 | SPI Read Access |
| 1 | spi\_wr | R/W | 0x00 | SPI Write Access |
| 2 | spi\_qrd | R/W | 0x00 | SPI Quad Read Access |
| 3 | spi\_qwr | R/W | 0x00 | SPI Quad Write Access |
| 4 | spi\_drd | R/W | 0x00 | SPI Dual Read Access |
| 5 | spi\_dwr | R/W | 0x00 | SPI Dual Write Access |
| 6 | spi\_swrst | R/W | 0x00 | SPI FSM Reset |
| 7 | Reserved | R/W | 0x00 | - |
| 11:8 | reg2spi\_csreg | R/W | 0x00 | SPI Chip Select |
| 31:12 | Reserved |  |  |  |

Table: SPI Clock Ctrl (0x04) = 0x02

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 7:0 | spi\_clk\_div | R/W | 0x00 | SPI Clock Div |
| 31:8 | Reserved | R/W | 0x00 | - |

Table: SPI COMMAND (0x08) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 7:0 | spi\_cmd | R/W | 0x00 | SPI Command Register |
| 15:8 | spi\_mode | R/W | 0x00 | SPI Mode Register |
| 31:16 | Reserved | R/W | 0x00 |  |

Table: SPI ADDRESS (0x0C) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 31:0 | spi\_addr | R/W | 0x00 | SPI Address |

Table: SPI LENGTH CTRL (0x10) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 5:0 | spi\_cmd\_len | R/W | 0x00 | SPI CMD Length in bit |
| 6 | reg2spi\_mode\_enb | R/W | 0x00 | SPI Mode Enable |
| 7 | Reserved | R/W |  |  |
| 13:8 | spi\_addr\_len | R/W | 0x00 | SPI Address Length in bit |
| 15:14 | Reserved | R/W |  |  |
| 21:16 | spi\_data\_len | R/W | 0x00 | SPI Data Length in bit |
| 31:22 | Reserved | R/W |  |  |

Table: SPI DUMMY CTRL (0x14) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 15:0 | spi\_dummy\_rd\_len | R/W | 0x00 | SPI Dummy Read cycles in bit |
| 31:16 | spi\_dummy\_wr\_len | R/W | 0x00 | SPI Dummy Write Cycle in bit |

Table: SPI WRITE DATA (0x18) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 31:0 | spi\_wdata | R/W | 0x00 | SPI WRITE DATA |

Table: SPI READ DATA (0x1C) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 31:0 | spi\_rdata | R | 0x00 | SPI READ DATA |

Table: SPI STATUS (0x1C) = 0x00

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bit(s) | Name | Type | Default | Function |
| 8:0 | Spi\_status | R | 0x00 | [0] – SPI IDLE PHASE  [1] – SPI CMD PHASE  [2] – SPI ADDRESS PHASE  [3] – SPI MODE PHASE  [4] – SPI DUMMY TX PHASE  [5] – SPI DUMMY RX PHASE  [6] – SPI TX DATA PHASE  [7] – SPI RX DATA PHASE  [8] – WAIT for CLK EDGE WAIT PHASE |

# Prerequisites

* Docker (ensure docker daemon is running) -- tested with version 19.03.12, but any recent version should suffice.

## Environment setting

export CARAVEL\_ROOT=<Carvel Installed Path>

export OPENLANE\_ROOT=<OpenLane Installed Path>

export PDK\_ROOT=<PDK Installed Path>

export IMAGE\_NAME=efabless/openlane:rc7

# Test’s preparation

The simulation package includes the following tests:

* **risc\_boot** - Simple User Risc core boot
* **wb\_port** - User Wishbone validation
* **user\_risc\_boot** - Standalone User Risc core boot

# Running Simulation

Examples:

make verify-wb\_port

make verify-risc\_hello

# Tool Sets

YiFive Soc flow uses Openlane tool sets.

1. **Synthesis**
   1. yosys - Performs RTL synthesis
   2. abc - Performs technology mapping
   3. OpenSTA - Pefroms static timing analysis on the resulting netlist to generate timing reports
2. **Floorplan and PDN**
   1. init\_fp - Defines the core area for the macro as well as the rows (used for placement) and the tracks (used for routing)
   2. ioplacer - Places the macro input and output ports
   3. pdn - Generates the power distribution network
   4. tapcell - Inserts welltap and decap cells in the floorplan
3. **Placement**
   1. RePLace - Performs global placement
   2. Resizer - Performs optional optimizations on the design
   3. OpenPhySyn - Performs timing optimizations on the design
   4. OpenDP - Perfroms detailed placement to legalize the globally placed components
4. **CTS**
   1. TritonCTS - Synthesizes the clock distribution network (the clock tree)
5. **Routing**
   1. FastRoute - Performs global routing to generate a guide file for the detailed router
   2. CU-GR - Another option for performing global routing.
   3. TritonRoute - Performs detailed routing
   4. SPEF-Extractor - Performs SPEF extraction
6. **GDSII Generation**
   1. Magic - Streams out the final GDSII layout file from the routed def
   2. Klayout - Streams out the final GDSII layout file from the routed def as a back-up
7. **Checks**
   1. Magic - Performs DRC Checks & Antenna Checks
   2. Klayout - Performs DRC Checks
   3. Netgen - Performs LVS Checks
   4. CVC - Performs Circuit Validity Checks

## **important Note**

Following tools in openlane docker is older version, we need to update these tool set.

* Icarus Verilog version 12.0 (devel) (s20150603-1107-ga446c34d)
* Yosys 0.9+4081 (git sha1 b6721aa9, clang 10.0.0-4ubuntu1 -fPIC -Os)

## Contacts

Report an issue: <https://github.com/dineshannayya/yifive_r0/issues>

# Documentation

* **Syntacore Link** - <https://github.com/syntacore/scr1>
* **SDRAM Controller** - <https://opencores.org/projects/sdr_ctrl>