1. **RISC-V Core**



Figure 1. RI5CY Core Diagram

RI5CY is an in-order, single-issue core with 4 pipeline stages and it has an IPC close to 1, full support for the base integer instruction set (RV32I), compressed instructions (RV32C) and multiplication instruction set extension (RV32M). It can be configured to have single-precision floating-point instruction set extension (RV32F). It implements several ISA extensions such as: hardware loops, post-incrementing load and store instructions, bit-manipulation instructions, MAC operations, support fixed-point operations, packed-SIMD instructions and the dot product. It has been designed to increase the energy efficiency of in ultra-low-power signal processing applications. RI5CY implementes a subset of the 1.9 privileged specification.

|  |  |  |
| --- | --- | --- |
|  | RI5CY without FPU | RI5CY with FPU |
| RV32I (Integer) | × | × |
| RV32C (Compact) | × | × |
| RV32M (Multiply/divide) | × | × |
| Single Precision FP |  | × |
| Debug | Run, Control, Inspect | Run, Control, Inspect |

Table 1. RI5CY core features

Enhanced supported extensions:

* Post-incrementing load and stores
* Multiply-accumulate extensions
* ALU extensions
* Hardware loops

Floating-point support in the form of IEEE-754 single precision can be enabled by setting the parameter FPU of the toplevel file “riscv\_core” to one. This will instantiate the FPU in the execution stage, and also extend the register file to host floating-point operands and extend the ALU to support the floating-point comparisons and classifications. The FPU extension is capable of performing all RISC-V floating-point operations that are defined in the FV32F ISA extensions.

|  |  |  |  |
| --- | --- | --- | --- |
| Instruction | Latency | Throughput | Architecture |
| Addition/Subtraction | 2 | 1 | Fully pipelined |
| I2F | 2 | 1 | Fully pipelined |
| F2I | 2 | 1 | Fully pipelined |
| Multiplication | 2 | 1 | Fully pipelined |
| FMAC | 3 | 1/3 | Iterative |
| DIV/SQRT | 8 | 1/7 | Iterative |

Table 2. FPU performance

RI5CY core is a very efficient core for DSP applications in higher frequencies, meanwhile RISCY+FPU core offer an IEEE-754 single precision FPU without much sacrificing clock speed.

1. **RISC-V Processor system**

The RISC-V Processor is a single-core SoC built based on RI5CY core. It uses separate single-port data and instruction RAMs. It includes a boot ROM that contains a boot loader that can load/program a program via SPI from/to an external flash device.

As shown in its block diagram, the SoC uses a AXI as its main interconnect with a bridge to APB for siple peripherals. Both the AXI and the APB buses feature 32-bit wide data channels. For debugging purposes, the SoC includes an advanced debug unit which enables access to core registers, the two RAMs and memory-mapped IO via JTAG. Both RAMs are connected to the AXI bus via bus adapters.



Figure 2. RISC-V SoC Diagram

The SoC also has a set of standard peripherals to ease deployment. These includes:

* Timer
* Event/Interrupt Unit
* SPI Master
* SPI Slave
* I2C
* UART
* GPIO
* Debug Unit (JTAG)

|  |  |  |
| --- | --- | --- |
| 0x0000 0000  0x0000 7FFF | 32kB RAM | Instruction Memory |
|  | Reserved |  |
| 0x0000 8000  0x0000 9FFF | 8kB ROM | Boot ROM (r/o) |
|  | Reserved |  |
| 0x0010 0000  0x0010 8000 | 32kB RAM | Data Memory |
|  | Reserved |  |
| 0x1A10 0000 | UART | Peripherals |
| 0x1A10 1000 | GPIO |
| 0x1A10 2000 | SPI Master |
| 0x1A10 3000 | Timer |
| 0x1A10 4000 | Event/Interrupt Unit |
| 0x1A10 5000 | I2C |
| 0x1A10 6000 | FLL |
| 0x1A10 7000 | SoC Control |
| 0x1A11 0000 | Debug |

Table 3. SoC memory map

1. **Software development**

Standard development environment includes:

* Custom RISCV GCC
* C/C++ libraries
* Adruino library
* Assembler, linker, disassembler
* Interactive software debug
* Instruction and cycle accurate simulators

In order to interactively debug software via gdb, the JTAG bridge is provided, which take advantage of FT2232 IC to send/receive JTAG commands to advanced debug unit inside the core.

To program the SoC, the SPI slave or the advanced debug unit can be used. Both have accesses to the whole memory map and can control the SoC. The SoC can also be used standalone without any external host that controls it. In this mode, it starts by using the boot loader contained in the boot ROM in order to load its program from an external SPI flash.