**AQL Tech Solutions, Islamabad**

****

**Project Report**

Compressed Extension of RISCV Core

**Group Members**

Soban Ahmed

Ariba Zahid

Nadeem Asghar

**Project Owners**

Aamir Sultan

Haseeb Ahmed

**Dated:** 1st June, 2022

# **Compressed Extension of RISCV Core**

# **Introduction**

Compressed instructions are a set of instructions which has length 16 bit with respect to RISC-V architecture. The Compressed instruction are represented by the character "C" in RISC-V specifications. So, in this project we would design and verify compressed extension of RISCV Core. For design and verification of compressed extension of RISCV core firstly we have studied the related literature. Then we have distributed our tasks and start working on tasks. Following we would discuss our tasks in details.

# **RTL Design**

For RTL design firstly we have tried to understand the existing RTL flow for this firstly we have tried to understand the RTL HDL coding. Basically, we were provided design of RISCVIM with 3 stage pipeline support. So, firstly we have designed its block diagram keeping our background knowledge and what we have to modify in previous design.

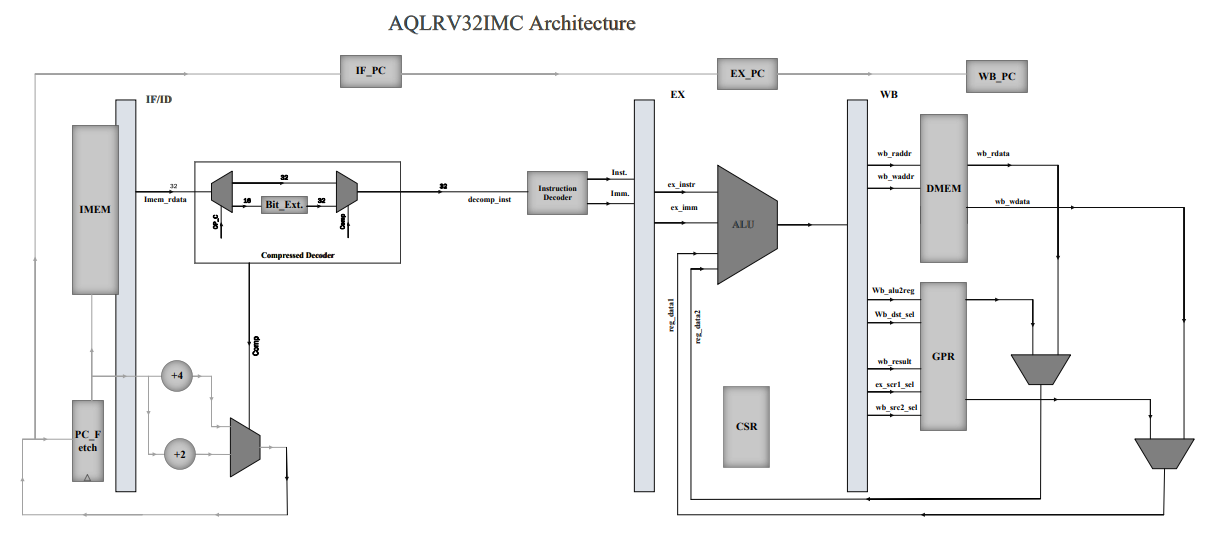


Fig.1 Block Diagram of RISCV IMC

After detail analysis we come to know that we have to update IF/ID stage of RISCV IM to convert into RISCV IMC. Therefore, we have started working on it and designed the compressed decoder module as shown in figure.1. Firstly, we have converted and map the compressed RISCV instructions into respective 32-bit instructions and pass them into next stage. Firstly, there is a DEMUX which decides whether the instruction is compressed or not based on opcode. Based on opcode it passes the respective instructions to MUX. If instruction is compressed then bit extension is done on the instruction and then forwarded to the next stage. Then we have used a flag in mux which passes compressed or uncompressed instruction depending upon the compressed flag. Second part of our design is PC optimization for compressed instructions. For normal flow PC should be update with +4 for each instruction but for compressed instruction we have updated PC value with +2 and for this purpose a MUX is used and selection between these combinations is done via using compressed flag from compressed decode as shown figure.1. So, this was all about our block diagram and remaining work flow of system remains the same.

Then we have consulted the RTL schematics and try to develop the understanding of flow execution before starting our Verilog coding. Following is the schematic of our RISCV IM module.

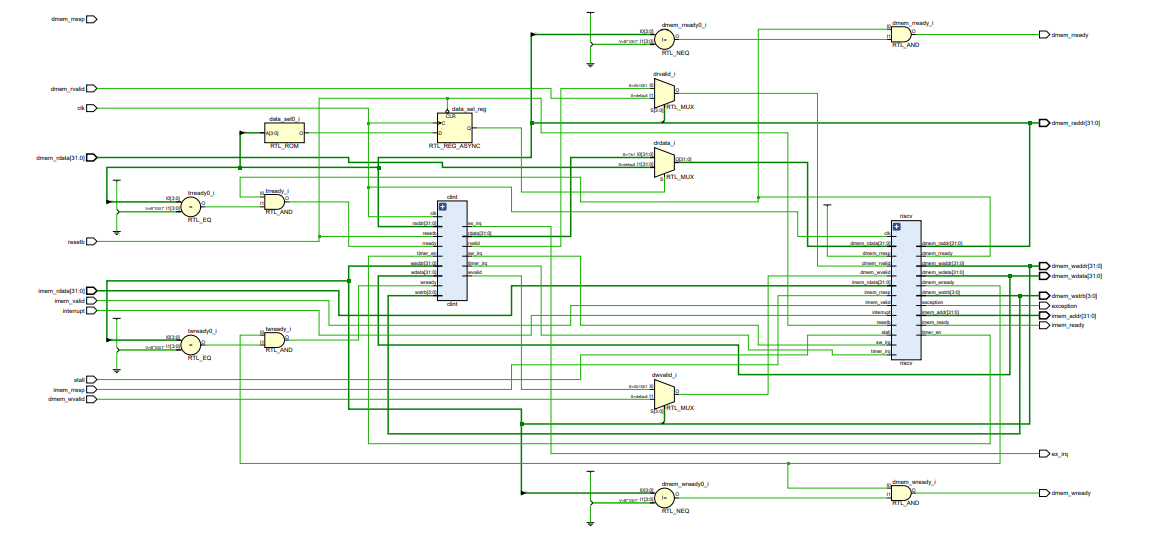
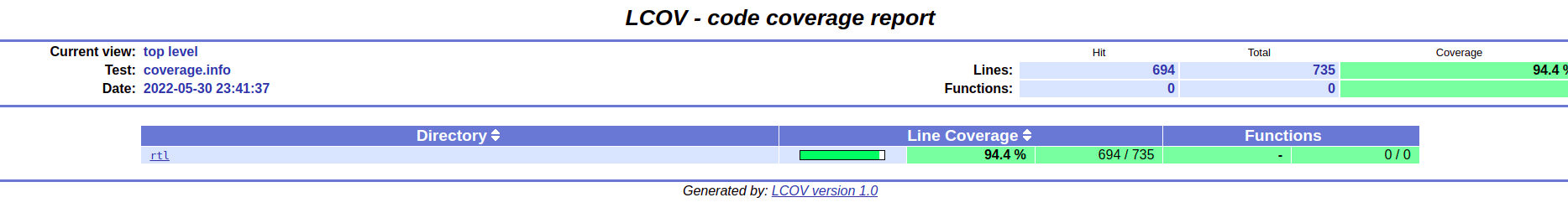


Fig.2 Schematic of RISCVIM

This RTL schematic is obtained by using Xilinx Vivado. After that we have start working on Verilog coding of compressed decoder and write the code for the compressed decoder based upon instruction manual of RISCV compressed instructions. The mapping compressed instructions on respective RISCVI instructions is based on RISCV green card and the instruction manual of RISCVC.

# **Verification**

Fig.2 Schematic of RISCVIM

