CV32E40P Verification

Name: shiva karthik

1. Tool Versions

- GCC (RISC-V): riscv32-unknown-elf-gcc 15.2.0
- Verilator: 5.044
- Python: 3.10.12
- GTKWave: 3.3.104
- Git: 2.34.1

2. RTL Source Explanation

The RTL source code for the CV32E40P processor is not included directly in the core-v-verif repository because the project follows a modular and reusable verification methodology. The core-v-verif repository is designed to provide a common verification framework and testbench infrastructure that can be reused for multiple CORE-V processor implementations. Keeping the RTL in a separate repository allows independent development, version control, and maintenance of the processor design. During simulation, the testbench gains access to the RTL by automatically cloning the official CV32E40P repository from GitHub using scripts defined in the Makefile. The required version of the RTL is checked out and integrated into the simulation environment through file lists and build scripts, enabling the testbench to compile and verify the processor without permanently storing the RTL source code in the verification repository.

3. Friction Point and Resolution

During the setup and validation of the CORE-V-VERIF environment, several technical challenges were encountered. Initially, only the riscv64-unknown-elf-gcc toolchain was available, while the CV32E40P verification environment required the riscv32-unknown-elf-gcc compiler. This mismatch caused compilation failures and required rebuilding the RISC-V GNU toolchain with Newlib support for the RV32 architecture. While building Newlib, permission errors occurred when installing to /opt/riscv32, which were resolved by configuring proper installation paths and environment variables. Additionally, an outdated version of Verilator initially prevented proper waveform generation and tracing, which was fixed by upgrading to Verilator version 5.044. Another major issue occurred when attempting to run the custom mscratch_test, where the simulation continued to execute the default hello-world program instead of the intended test. This problem was traced to missing or incorrectly configured test.yaml files and environment variables, causing the build system to fall back to the default test configuration. After properly defining the test metadata and toolchain variables, the custom test was successfully compiled and executed.

4. Waveform Generation

Waveforms were generated using WAVES=1 flag. The generated .vcd file was viewed using GTKWave to analyze instruction fetch behavior.

5. mscratch Test Description

A custom program was written to write and read 32-bit values to the mscratch CSR and verify correctness.

Code mscratch_test.c

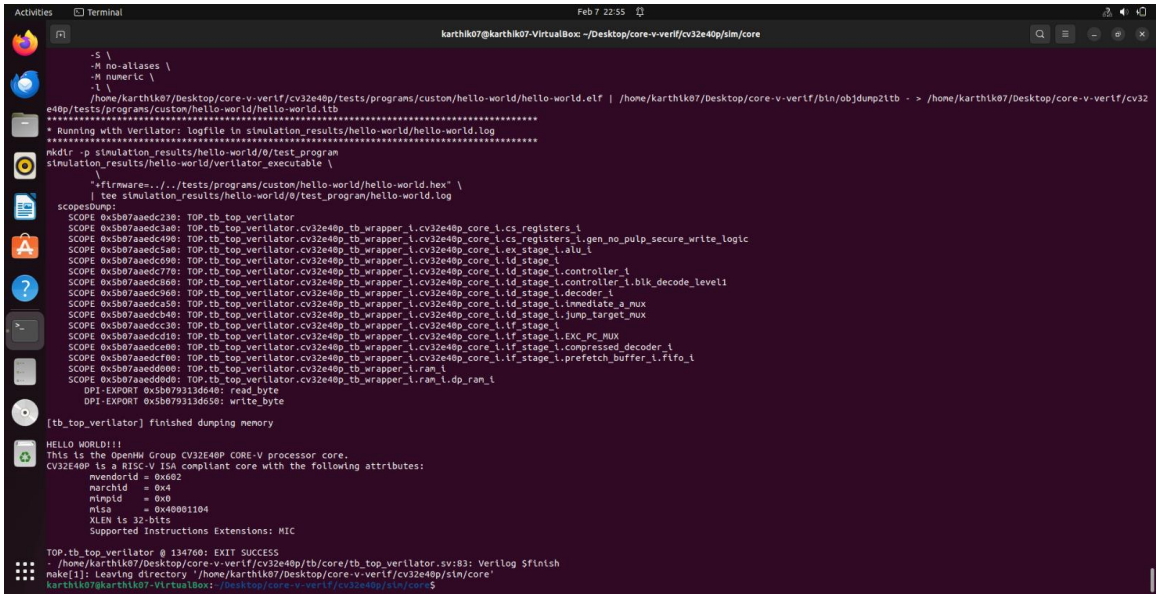
```c
#include <stdint.h>
#include <stdio.h>

static inline void write_csr(uint32_t val) {
    asm volatile ("csrw mscratch, %0" :: "r"(val));
}

static inline uint32_t read_csr(void) {
    uint32_t val;
    asm volatile ("csrr %0, mscratch" : "=r"(val));
    return val;
}

int main() {
    uint32_t patterns[] = {
        0x00000000,
        0xFFFFFFFF,
        0xA5A5A5A5,
        0x5A5A5A5A
    };

    printf("Starting mscratch test...\n");

    for (int i = 0; i < 4; i++) {
        write_csr(patterns[i]);
        uint32_t r = read_csr();

        printf("Write: 0x%08X Read: 0x%08X\n",
               patterns[i], r);

        if (r != patterns[i]) {
            printf("FAIL at index %d\n", i);
            while (1);   // stop here
        }
    }

    printf("MSCRATCH TEST PASSED\n");

    return 0;
}
```

test.yaml

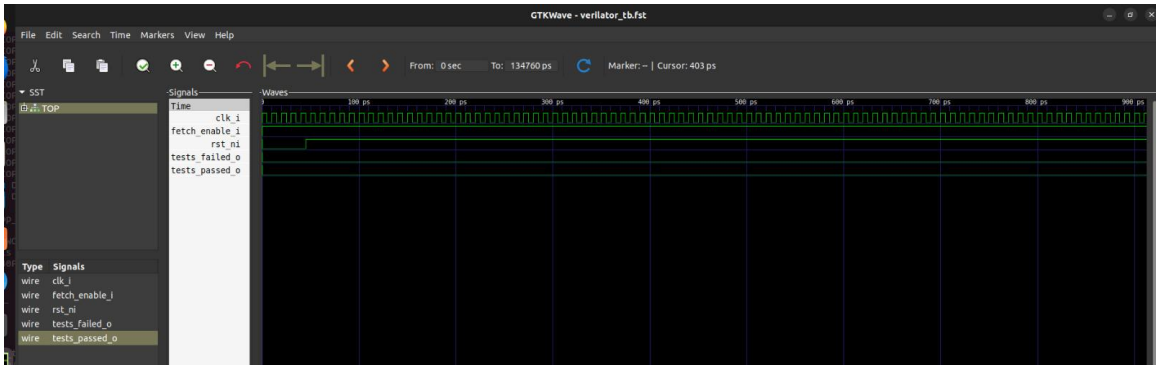
```yaml
name: mscratch_test

description: Test full 32-bit read/write of mscratch CSR

uvm_test: corev_direct_test

program: mscratch_test.c

build:
  gcc_opts: "-march=rv32imc_zicsr -mabi=ilp32"

sim:
  timeout: 200000
```

6. Validation Evidence

HELLO WORLD TERMINAL OUTPUT



WAVEFORM VIEWED



Mscratch 32 bit read and write terminal output