

The Trireme Platform: Cove Processor Demo

September 11, 2022

1 Introduction

The Cove Processor is an RV64im microcontroller with support for a subset of the RISC-V privilege specification. Supported privilege modes include M (machine), S (supervisor), and U (user). The ECALL instruction as well as timer and software interrupts are supported. Future versions of the **seven_stage_priv_core** used by the Cove Processor will continue to build support for additional privilege specification features, including virtual memory. Figure 1 presents a high level block diagram of the Cove Processor. The Cove Processor uses the **dual_port_BRAM_memory_subsystem** as a main memory and does not include a cache hierarchy. Several additional memory mapped peripherals are supported, including:

- A software interrupt register
- A timer with MTIME and MTIMECMP registers
- A UART configured with a default configuration of 8n1 at 115200 bits/second

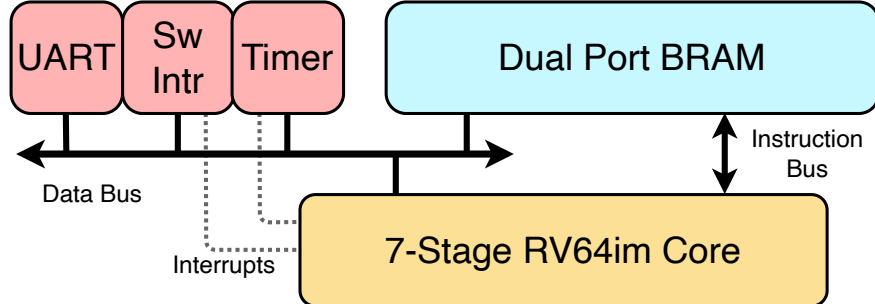


Figure 1: A high-level block diagram of the Cove Processor.

2 Running a Simulation

These directions assume you have a version of Modelsim (or Questasim) correctly installed. The simulation scripts included with the Trireme Platform require that Modelsim is added to your PATH variable. By default, Modelsim installs itself into your home directory. To add it to your PATH, add something like the following to your .bashrc file:

```
# For Modelsim Altera Starter Edition installed with Quartus II 15.0
export PATH=$PATH:~/altera/15.0/modelsim_ase/bin
# For Modelsim Altera Starter Edition installed with Quartus Prime 18.1
export PATH=$PATH:~/intelFPGA_lite/18.1/modelsim_ase/bin
# For Questasim Intel FPGA Edition installed with Quartus Prime Pro 21.3
export PATH=$PATH:~/intelFPGA_pro/21.3/questa_fse/bin
```

To run the top level Cove Processor test bench, execute the `run_test` script from the `modelsim` directory. The `run_test` script will simulate one or more verilog test benches whose module names are provided as an argument.

```
./run_test tb_cove_processor
```

When you run the simulation script, the verilog modules are compiled and the simulator runs the specified test bench(s). Listing 4 in the Appendix shows the default output for the **tb_cove_processor** test bench.

3 Demo Programs

The `modelsim/binaries` directory stores binaries used in test bench simulation. Two compiled example binaries named `gcd64_262144.vmh` and `short_mandelbrot64_262144.vmh` are included. The GCD application should complete in under 60 seconds while the Mandelbrot application will take 5-10 minutes. Changing the `PROGRAM` parameter in the `tb_cove_processor.v` file in the `rtl` directory will change which program the test bench executes. The `example_programs` directory includes the source C code for each application, in addition to the elf file output by the RISC-V toolchain, a binary disassembly (`.dump`) and a copy of the `.vmh` file.

The programs were compiled with the following compiler wrapper command. The outputs were manually renamed to reflect the 64-bit toolchain and the stack address of 262144 Bytes. Additional programs can be compiled by downloading the Trireme Platform's software stack and providing different C code as input.

```
PROGRAM=gcd
./trireme_gcc -o applications/binaries/$PROGRAM \
--vmh applications/binaries/$PROGRAM.vmh \
--dump applications/binaries/$PROGRAM.dump \
--ram-size 262144 --start-addr 0 \
--stack-addr 262144 --stack-size 4096 --heap-size 4096 --64-bit \
applications/src/$PROGRAM.c
```

4 Apendix

```
# *****
# * The Trireme Platform - Cove Processor Test Bench
# *****
#
# Use the PROGRAM parameter to execute different program binaries
# Loading ./binaries/gcd64_262144.vmh
#
#
# Run Time (cycles):          367
# Dumping reg file states:
# Reg Index, Value
#      0: 0000000000000000
#      1: 00000000000000a8
#      2: 0000000000040000
#      3: 0000000000000000
#      4: 0000000000000000
#      5: 0000000000000000
#      6: 0000000000000000
#      7: 0000000000000000
#      8: 0000000000000000
#      9: 0000000000000010
#     10: 0000000000000010
#     11: 0000000000000010
#     12: 0000000000000000
#     13: 0000000000000000
#     14: 0000000000000010
#     15: 0000000000000010
#     16: 0000000000000000
#     17: 0000000000000000
#     18: 0000000000000000
#     19: 0000000000000000
#     20: 0000000000000000
#     21: 0000000000000000
#     22: 0000000000000000
#     23: 0000000000000000
#     24: 0000000000000000
#     25: 0000000000000000
#     26: 0000000000000000
#     27: 0000000000000000
#     28: 0000000000000000
#     29: 0000000000000000
#     30: 0000000000000000
#     31: 0000000000000000
#
#
# tb_cove_processor test complete!
#
```

Listing 4: Default output for the tb_cove_processor test bench.