

The Trireme Platform: Basin Processor Demo

November 29, 2022

1 Introduction

The Basin Processor is a dual core RV32i system with a cache hierarchy and an external memory interface that can be connected to an FPGA/board specific memory controller (not included in the Trireme Platform). It is meant to provide a good example to become familiar with the Trireme Platform's multi-core programming features and cache hierarchy configuration. Figure 1 shows a high-level block diagram of the Basin Processor. Parameters for each L1 and L2 cache can be configured in the basin processor test bench (**tb_basin_processor**) located in the rtl directory. Parameters for each L1 cache are provided as a list of concatenated 32-bit integers. The main memory connects an FPGA BRAM to the L2 cache.

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 Basin 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_basin_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_basin_processor** test bench.

3 Demo Programs

The modelsim/binaries directory stores binaries used in test bench simulation. One compiled multi-programmed workload example binary named gcd_hanoi.vmh is included. Both applications in the workload should complete in under 60 seconds. Changing the PROGRAM parameter in the tb_basin_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 the gcd_hanoi workload, in addition to the elf file output by the RISC-V toolchain, a binary disassembly (.dump) and a copy of the .vmh file.

The program was compiled with the following compiler wrapper command. Additional programs can be compiled by downloading the Trireme Platform's software stack and providing different C code as input.

```

PROGRAM=gcd_hanoi
./tirieme_gcc -o applications/binaries/$PROGRAM \
  --vmh applications/binaries/$PROGRAM.vmh \
  --dump applications/binaries/$PROGRAM.dump \
  --num-cores 2 \
  --ram-size 262144 --start-addr 0 \
  --stack-addr 262144 --stack-size 4096 --heap-size 4096 \
  applications/src/$PROGRAM.c

```

4 Appendix

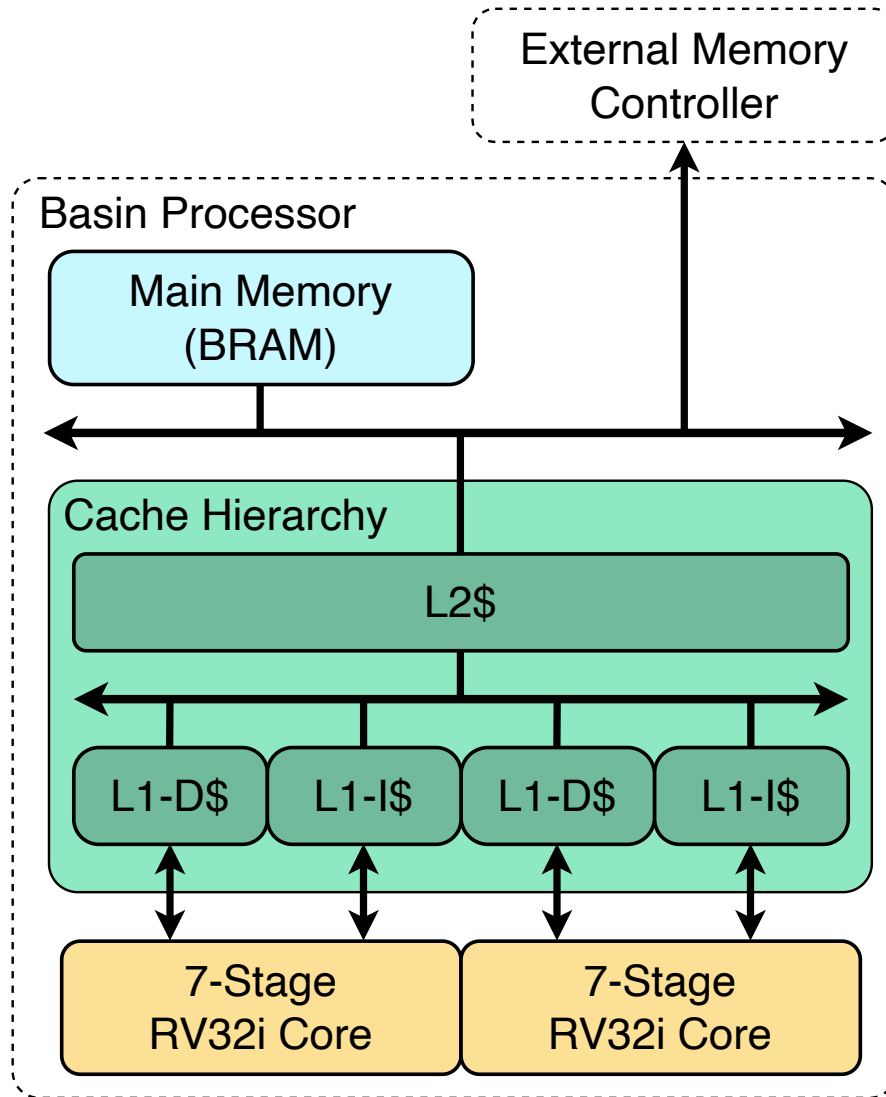


Figure 1: A high-level block diagram of the Basin Processor.

```

# *****
# * The Trireme Platform - Basin Processor Test Bench
# *****
#
# Use the PROGRAM parameter to execute different program binaries
# Loading ./binaries/gcd_hanoi.vmh
#
#
# Core 0 finished. Run Time (cycles):          553
# Dumping core 0 reg file states:
# Reg Index, Value
#      0: 00000000
#      1: 000000c0
#      2: 00040000
#      3: 00000000
#      4: 00000000
#      5: 00000000
#      6: 00000000
#      7: 00000000
#      8: 00000000
#      9: 00000010
#     10: 00000010
#     11: 00000010
#     12: 00000000
#     13: 00000000
#     14: 00000010
#     15: 00000010
#     16: 00000000
#     17: 00000000
#     18: 00000000
#     19: 00000000
#     20: 00000000
#     21: 00000000
#     22: 00000000
#     23: 00000000
#     24: 00000000
#     25: 00000000
#     26: 00000000
#     27: 00000000
#     28: 00000000
#     29: 00000000
#     30: 00000000
#     31: 00000000
#
#

```

```

# Core 1 finished. Run Time (cycles):      815
# Dumping core 1 reg file states:
# Reg Index, Value
#      0: 00000000
#      1: 00000170
#      2: 00041000
#      3: 00000000
#      4: 00000000
#      5: 00000000
#      6: 00000000
#      7: 00000000
#      8: 00000000
#      9: 0000000f
#     10: 0000000f
#     11: 00000002
#     12: 00000001
#     13: 00000003
#     14: 00000007
#     15: 0000000f
#     16: 00000000
#     17: 00000000
#     18: 00000000
#     19: 00000000
#     20: 00000000
#     21: 00000000
#     22: 00000000
#     23: 00000000
#     24: 00000000
#     25: 00000000
#     26: 00000000
#     27: 00000000
#     28: 00000000
#     29: 00000000
#     30: 00000000
#     31: 00000000
#
#
# tb_basin_processor Test Complete!

```

Listing 5: Default output for the tb_basin_processor test bench.