

Computer Architecture

Final Project Part 2

**Enhance FP1 Design with Vector Coprocessors and Custom
Accelerators On Near-DRAM Platform**

Group:25

Member:

313512041 朱宥勳

313512035 侯書維

313512085 張嘉麟

313512071 陳柏皓

A. Index.c

To improve the performance of RNA sequence indexing, we implemented a hardware-accelerated k-mer hashing approach using custom RISC-V instructions. Specifically, we used the accelerator_reduce_sum instruction to process each k-mer as two 64-bit integers, enabling fast hashing for mapping the k-mer to its corresponding transcript. The results are stored in a HashMap for later use in quantification.

a. Algorithm and Instruction Usage

1. Accelerator Hashing

A core component of our design is the use of the custom RISC-V instruction accelerator_reduce_sum. This instruction performs low-level bitwise operations on two 64-bit operands to quickly produce a hash-like result. Each k-mer is split into two 64-bit words and passed to the instruction, whose output serves as a computational anchor to simulate hash processing and trigger accelerator activity for evaluation purposes.

2. HashMap Insertion

Every k-mer extracted from the input sequences is converted to a null-terminated string and passed to accelerator_hash() for hardware-based processing. The resulting k-mer is then inserted into a HashMap along with the transcript index. This design preserves high-level programming clarity while leveraging hardware-level acceleration to optimize performance.

b. Performance Result

1. Part1

```
root@1a4699c286c:/home/CA-FP1/evaluation# make run_index
/home/gen5/build/RISCV/gen5.opt /home/gen5/configs/example/se.py -c index.o -o "15" --cpu-type=MinorCPU --cpu-clock=800MHz --mem-type=Ramulator --ramulator-config=/home/gen5/configs/ramulator/LDDR4-config.cfg --mem-size=256MB --caches --l1i_size=32kB --l1i_assoc=8 --l1d_size=32kB --l1d_assoc=8 --cacheline=32
gen5 Simulator System. http://gen5.org
gen5 is copyrighted software; use the --copyright option for details.

gen5 version 28.1.0.0
gen5 compiled Feb 7 2022 09:12:04
gen5 started May 20 2025 13:00:58
gen5 executing e1a4699c286c, pid 2775
command line: /home/gen5/build/RISCV/gen5.opt /home/gen5/configs/example/se.py -c index.o -o 15 --cpu-type=MinorCPU --cpu-clock=800MHz --mem-type=Ramulator --ramulator-config=/home/gen5/configs/ramulator/LDDR4-config.cfg --mem-size=256MB --caches --l1i_size=32kB --l1i_assoc=8 --l1d_size=32kB --l1d_assoc=8 --cacheline=32
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'.
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'.
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'.
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'.
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'.
Ramulator system configuration file = /home/gen5/configs/ramulator/LDDR4-config.cfg
MEMORY TYPE:: Ramulator
Ramulator
warn: membus.master is deprecated. 'master' is now called 'mem_side_ports'
Global Frequency set at 1000000000000 ticks per second
using riscv64 ISAs
warn: Unknown operating system; assuming Linux.
0: system.remote_gdb: listening for remote gdb on port 7000
using LPDDR4X 4260MHz
*****[REDACTED]*****
Info: Entering event queue @ 0. Starting simulation...
reset tick: 14287500
dump tick: 67507382500
Simulated Time : 0.067553 seconds
Exiting @ tick 80530253750 because exiting with last active thread context
rm -rf index_result/m5out
mv m5out index_result/
root@1a4699c286c:/home/CA-FP1/evaluation# vi distrib.cpp
```

2. Part2

```
[root@794ea3f3e15:/home/CA-FP2/evaluation]# gcc -march=rv32imc -O2 index.c -I /home/gen5/include -o index.o index.c -I /home/gen5/include/riscv -I /home/gen5/include/lmb5a -I /home/gen5/include/lscv -I /home/gen5/include/lscv/out/lmb5a.o
[root@794ea3f3e15:/home/CA-FP2/evaluation]# make run_index
make[1]: Entering directory '/home/CA-FP2/evaluation'
# Ensure Index.result directory exists for output
mkdir -p Index.result
# Run Gems simulation
/home/gen5/build/RISCV/gem5.opt /home/gen5/configs/example/se.py -c index.o -o "i5" --cpu-type=MinorCPU --cpu-clock=800MHz --mem-type=Ramulator --ramulator-config=/home/gen5/configs/ramulator/LPDDR4-Config.gem5
gem5 Simulator System. http://gem5.org
gem5 is copyrighted software; use the --copyright option for details.

gem5 version 20.1.0.0
gem5 compiled Jun 20 2025 05:59:35
gem5 started Jun 20 2025 06:25:16
gem5 executing on a794ea3f3e15, pid 249
command line: /home/gen5/build/RISCV/gem5.opt /home/gen5/configs/example/se.py -c index.o -o i5 --cpu-type=MinorCPU --cpu-clock=800MHz --mem-type=Ramulator --ramulator-config=/home/gen5/configs/ramulator/LPDDR4-Config.cfg --mem-size=256M --caches --l1_size=32kB --l1_assoc=8 --l1_size=32kB --l1_assoc=8 --cacheline=32
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: membus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
Ramulator system configuration file = /home/gen5/configs/ramulator/LPDDR4-Config.cfg
MEMORY TYPE: Ramulator
Ramulator
warn: membus.master is deprecated. 'master' is now called 'mem_side_ports'
Global Frequency set at 1000000000000 ticks per second
using riscv15A
warning: using operating system; assuming Linux.
0: system.remote.gdb: listening for remote gdb on port 7000
using LPDDR4X_4260MHz
**** REAL SIMULATION ****
Warning: queue @ 0. Starting simulation...
reset tick: 9350000
dump tick: 28822620250
Simulated Time : 0.028813 seconds
Exiting @ tick 28822620250 because exiting with last active thread context
executing previous command and move new one
rm -rf index.result/nsout
mv nsout index.result
[root@794ea3f3e15:/home/CA-FP2/evaluation]# make make_quantify
```

B. Quantify.c

To accelerate RNA sequence quantification, we integrated a custom RISC-V instruction, `vector_compare`, into the k-mer comparison logic. This allows for byte-level equality checking between two 64-bit words in a single hardware instruction, significantly improving performance compared to traditional per-character comparisons in C.

a. Algorithm and Accelerator Usage

1. Vectorized String Comparison (Vector Compare)

The core optimization in this project lies in replacing software-based k-mer comparison with a custom instruction, `vector_compare`. Each k-mer is split into two 64-bit words, and the instruction compares corresponding bytes between them. If all bytes match, the result is a mask of `0xFFFFFFFFFFFFFFFFF`. This transforms a multi-iteration character comparison into a single-cycle instruction, significantly reducing CPU workload and branching overhead.

2. Matching Strategy with HashMap

Matching still leverages a `HashMap` lookup to retrieve potential transcript indices. For each match, we increment the count using minimal memory writes and avoid redundant updates using a round-based update scheme, ensuring both efficiency and correctness in determining the best-matched transcript.

b. Performance Result

1. Part1

```
root@1a4699c286c:/home/CA-FP1_quantify_new/evaluation# make run_quantify
/home/gen5/build/RISCV/gen5_opt /home/gen5/configs/example/se.py -c quantify.o -o "15" --cpu-type=MinorCPU --cpu-clock=800MHz --mem-type=Ramulator --ramulator-config=/home/gen5/conf
gs/ramulator/LDDR4-config.cfg --mem-size=250MB --caches -l1_size=32kB -l1_assoc=8 -l1d_size=32kB -l1d_assoc=8 --cacheline=32
gen5 Simulator System. http://gen5.org
gen5 is copyrighted software; use the --copyright option for details.

gen5 version 20.1.0-0
gen5 compiled Feb 7 2022 09:12:04
gen5 started May 20 2025 13:02:15
gen5 executing on e1a4699c286c, pid 2788
command line: /home/gen5/build/RISCV/gen5_opt /home/gen5/configs/example/se.py -c quantify.o -o 15 --cpu-type=MinorCPU --cpu-clock=800MHz --mem-type=Ramulator --ramulator-config=/ho
me/gen5/configs/ramulator/LDDR4-config.cfg --mem-size=250MB --caches -l1_size=32kB -l1_assoc=8 -l1d_size=32kB -l1d_assoc=8 --cacheline=32

warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
MEMORY TYPE:: Ramulator
RAMULATOR system configuration file = /home/gen5/configs/ramulator/LDDR4-config.cfg
RAMULATOR
warn: nembus.master is deprecated. 'master' is now called 'mem_side_ports'
Global frequency set at 1000000000000 ticks per second
using riscv64 ISA
warn: Unknown operating system; assuming Linux.
0: system.remote_gdb: listening for remote gdb on port 7001
using LPDDR4 426MHz
***** REAL SIMULATION *****
Info: Entering event queue @ 0. Starting simulation...
Info: Increasing stack size by one page.
Info: Increasing stack size by one page.
reset tick: 10199331250
dump tick: 10199375000
Simulated Time: 0.136885 seconds
Exiting @ tick 250355948750 because exiting with last active thread context
rm -rf quantify_result/m5out
mv m5out quantify_result/
root@1a4699c286c:/home/CA-FP1_quantify_new/evaluation#
```

2. Part2

```
root@794ea3f3e15:/home/CA-FP2/evaluations# make run_quantify
/opt/riscv-ca-fp2/bin/riscv64-unknown-elf-gcc -static -I /home/gen5/include -o quantify.o quantify.c /home/gen5/util/n5/build/riscv/out/lbm5.a
# Ensure quantify_result directory exists for output
mkdir -p quantify_result
# Run quantifying
/home/gen5/build/RISCV/gen5_opt /home/gen5/configs/example/se.py -c quantify.o -o 15 --cpu-type=MinorCPU --cpu-clock=800MHz --mem-type=Ramulator --ramulator-config=/home/gen5/configs/ramulator/LDDR4-co
nfig.cfg --mem-size=250MB --caches -l1_size=32kB -l1_assoc=8 -l1d_size=32kB -l1d_assoc=8 --cacheline=32
gen5 Simulator System. http://gen5.org
gen5 is copyrighted software; use the --copyright option for details.

gen5 version 20.1.0-0
gen5 compiled Jun 29 2025 05:59:35
gen5 started Jun 29 2025 05:59:41
gen5 executing on 794ea3f3e15, pid 271
command line: /home/gen5/build/RISCV/gen5_opt /home/gen5/configs/example/se.py -c quantify.o -o 15 --cpu-type=MinorCPU --cpu-clock=800MHz --mem-type=Ramulator --ramulator-config=/home/gen5/configs/ramula
tor/LDDR4-config.cfg --mem-size=250MB --caches -l1_size=32kB -l1_assoc=8 -l1d_size=32kB -l1d_assoc=8 --cacheline=32

warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
warn: nembus.slave is deprecated. 'slave' is now called 'cpu_side_ports'
MEMORY TYPE:: Ramulator
RAMULATOR system configuration file = /home/gen5/configs/ramulator/LDDR4-config.cfg
RAMULATOR
warn: nembus.master is deprecated. 'master' is now called 'mem_side_ports'
Global frequency set at 1000000000000 ticks per second
using riscv64 ISA
warn: Unknown operating system; assuming Linux.
0: system.remote_gdb: listening for remote gdb on port 7000
using LPDDR4 426MHz
***** REAL SIMULATION *****
Info: Entering event queue @ 0. Starting simulation...
Info: Increasing stack size by one page.
Info: Increasing stack size by one page.
reset tick: 10049957850
dump tick: 10049957850
Simulated Time: 0.129920 seconds
Exiting @ tick 103591137500 because exiting with last active thread context
rm -rf quantify_result/m5out
mv m5out quantify_result/
root@794ea3f3e15:/home/CA-FP2/evaluations cd /home/gen5/src/arch/riscv/isa
```

C. Decoder.isa

To accelerate k-mer hashing and comparison in RNA quantification, we implemented two custom RISC-V instructions within the decoder.isa file: accelerator_reduce_sum and vector_compare. These instructions are tightly coupled with the logic in index.c and quantify.c, respectively, to support hardware-level optimization for hashing and byte-wise comparison.

a. Instruction Design

1. Accelerator reduce sum

In our design, accelerator_reduce_sum is used to compute a fast hash over two 64-bit integers that represent the binary encoding of a k-mer. Instead of summing each byte, we use a simple bitwise XOR operation:

$$Rd = Rs1 \wedge Rs2;$$

This approach provides fast execution and simple hardware implementation. It is consistent with the accelerator_hash() function in index.c, which expects a quick and deterministic hash to index k-mers into a map.

2. Vector compare

The vector_compare instruction performs a byte-wise equality check between Rs1 and Rs2. For each matching byte, the corresponding byte in Rd is set to 0xFF; otherwise, it remains 0x00:

for each byte i:

if byte_i(Rs1) == byte_i(Rs2):

Rd.byte[i] = 0xFF

else:

Rd.byte[i] = 0x00

This encoding is directly compatible with the logic in vector_kmer_equal() from quantify.c, which considers two k-mers equal only if the result is 0xFFFFFFFFFFFFFFFULL.