**Final Project Report:**

The priority queue is implemented using the minheap and the linked list patterns and simulated on the X86 and RISCV architectures on gem5.

First, the priority queue implementations on both patterns is translated into X86 assembly byte code, and simulated on the X86 architecture on gem5, to obtain the instruction mix of the program. The results are as follows,

Have to simulate again and attach the Instruction mix part of the stat.txt file…since the files were missing.

The instruction breakup of the X86 instructions for minheap implementation is as follows:

No\_OpClass 413398 0.99% 0.99% # Class of committed instruction. (Count)

IntAlu 32077035 76.63% 77.62% # Class of committed instruction. (Count)

IntMult 100738 0.24% 77.86% # Class of committed instruction. (Count)

IntDiv 28 0.00% 77.86% # Class of committed instruction. (Count)

FloatAdd 331994 0.79% 78.65% # Class of committed instruction. (Count)

FloatCmp 0 0.00% 78.65% # Class of committed instruction. (Count)

FloatCvt 0 0.00% 78.65% # Class of committed instruction. (Count)

FloatMult 0 0.00% 78.65% # Class of committed instruction. (Count)

FloatMultAcc 0 0.00% 78.65% # Class of committed instruction. (Count)

FloatDiv 0 0.00% 78.65% # Class of committed instruction. (Count)

FloatMisc 0 0.00% 78.65% # Class of committed instruction. (Count)

FloatSqrt 0 0.00% 78.65% # Class of committed instruction. (Count)

SimdAdd 8 0.00% 78.65% # Class of committed instruction. (Count)

SimdAddAcc 0 0.00% 78.65% # Class of committed instruction. (Count)

SimdAlu 180094 0.43% 79.08% # Class of committed instruction. (Count)

SimdCmp 0 0.00% 79.08% # Class of committed instruction. (Count)

SimdCvt 100046 0.24% 79.32% # Class of committed instruction. (Count)

SimdMisc 240258 0.57% 79.89% # Class of committed instruction. (Count)

SimdMult 0 0.00% 79.89% # Class of committed instruction. (Count)

SimdMultAcc 0 0.00% 79.89% # Class of committed instruction. (Count)

SimdMatMultAcc 0 0.00% 79.89% # Class of committed instruction. (Count)

SimdShift 0 0.00% 79.89% # Class of committed instruction. (Count)

SimdShiftAcc 0 0.00% 79.89% # Class of committed instruction. (Count)

SimdDiv 0 0.00% 79.89% # Class of committed instruction. (Count)

SimdSqrt 0 0.00% 79.89% # Class of committed instruction. (Count)

SimdFloatAdd 10000 0.02% 79.92% # Class of committed instruction. (Count)

SimdFloatAlu 0 0.00% 79.92% # Class of committed instruction. (Count)

SimdFloatCmp 0 0.00% 79.92% # Class of committed instruction. (Count)

SimdFloatCvt 259203 0.62% 80.54% # Class of committed instruction. (Count)

SimdFloatDiv 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdFloatMisc 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdFloatMult 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdFloatMultAcc 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdFloatMatMultAcc 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdFloatSqrt 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdReduceAdd 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdReduceAlu 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdReduceCmp 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdFloatReduceAdd 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdFloatReduceCmp 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdAes 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdAesMix 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdSha1Hash 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdSha1Hash2 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdSha256Hash 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdSha256Hash2 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdShaSigma2 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdShaSigma3 0 0.00% 80.54% # Class of committed instruction. (Count)

SimdPredAlu 0 0.00% 80.54% # Class of committed instruction. (Count)

Matrix 0 0.00% 80.54% # Class of committed instruction. (Count)

MatrixMov 0 0.00% 80.54% # Class of committed instruction. (Count)

MatrixOP 0 0.00% 80.54% # Class of committed instruction. (Count)

MemRead 4062106 9.70% 90.24% # Class of committed instruction. (Count)

MemWrite 2784908 6.65% 96.89% # Class of committed instruction. (Count)

FloatMemRead 789038 1.88% 98.78% # Class of committed instruction. (Count)

FloatMemWrite 512031 1.22% 100.00% # Class of committed instruction. (Count)

The instruction breakup of the X86 instructions for linked list implementation is as follows:

No\_OpClass 443408 0.14% 0.14% # Class of committed instruction. (Count)

IntAlu 235008823 73.35% 73.49% # Class of committed instruction. (Count)

IntMult 100740 0.03% 73.52% # Class of committed instruction. (Count)

IntDiv 28 0.00% 73.52% # Class of committed instruction. (Count)

FloatAdd 120214 0.04% 73.56% # Class of committed instruction. (Count)

FloatCmp 0 0.00% 73.56% # Class of committed instruction. (Count)

FloatCvt 0 0.00% 73.56% # Class of committed instruction. (Count)

FloatMult 0 0.00% 73.56% # Class of committed instruction. (Count)

FloatMultAcc 0 0.00% 73.56% # Class of committed instruction. (Count)

FloatDiv 0 0.00% 73.56% # Class of committed instruction. (Count)

FloatMisc 0 0.00% 73.56% # Class of committed instruction. (Count)

FloatSqrt 0 0.00% 73.56% # Class of committed instruction. (Count)

SimdAdd 8 0.00% 73.56% # Class of committed instruction. (Count)

SimdAddAcc 0 0.00% 73.56% # Class of committed instruction. (Count)

SimdAlu 181244 0.06% 73.62% # Class of committed instruction. (Count)

SimdCmp 0 0.00% 73.62% # Class of committed instruction. (Count)

SimdCvt 100046 0.03% 73.65% # Class of committed instruction. (Count)

SimdMisc 270254 0.08% 73.73% # Class of committed instruction. (Count)

SimdMult 0 0.00% 73.73% # Class of committed instruction. (Count)

SimdMultAcc 0 0.00% 73.73% # Class of committed instruction. (Count)

SimdMatMultAcc 0 0.00% 73.73% # Class of committed instruction. (Count)

SimdShift 0 0.00% 73.73% # Class of committed instruction. (Count)

SimdShiftAcc 0 0.00% 73.73% # Class of committed instruction. (Count)

SimdDiv 0 0.00% 73.73% # Class of committed instruction. (Count)

SimdSqrt 0 0.00% 73.73% # Class of committed instruction. (Count)

SimdFloatAdd 10000 0.00% 73.74% # Class of committed instruction. (Count)

SimdFloatAlu 0 0.00% 73.74% # Class of committed instruction. (Count)

SimdFloatCmp 0 0.00% 73.74% # Class of committed instruction. (Count)

SimdFloatCvt 25170416 7.86% 81.59% # Class of committed instruction. (Count)

SimdFloatDiv 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdFloatMisc 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdFloatMult 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdFloatMultAcc 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdFloatMatMultAcc 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdFloatSqrt 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdReduceAdd 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdReduceAlu 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdReduceCmp 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdFloatReduceAdd 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdFloatReduceCmp 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdAes 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdAesMix 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdSha1Hash 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdSha1Hash2 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdSha256Hash 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdSha256Hash2 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdShaSigma2 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdShaSigma3 0 0.00% 81.59% # Class of committed instruction. (Count)

SimdPredAlu 0 0.00% 81.59% # Class of committed instruction. (Count)

Matrix 0 0.00% 81.59% # Class of committed instruction. (Count)

MatrixMov 0 0.00% 81.59% # Class of committed instruction. (Count)

MatrixOP 0 0.00% 81.59% # Class of committed instruction. (Count)

MemRead 29980405 9.36% 90.95% # Class of committed instruction. (Count)

MemWrite 3347307 1.04% 92.00% # Class of committed instruction. (Count)

FloatMemRead 25360489 7.92% 99.91% # Class of committed instruction. (Count)

FloatMemWrite 281755 0.09% 100.00% # Class of committed instruction. (Count)

The stats of the sort loop in main.c, generated by the gem5 for X86 and RISCV for minheap implementation is as follows:

For X86:

Before sort loop’ instruction mix:

No\_OpClass 3316 0.36% 0.36% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::IntAlu 436748 47.00% 47.35% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::IntMult 0 0.00% 47.35% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::IntDiv 0 0.00% 47.35% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatAdd 36614 3.94% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatCmp 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatCvt 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatMult 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatMultAcc 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatDiv 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatMisc 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatSqrt 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdAdd 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdAddAcc 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdAlu 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdCmp 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdCvt 0 0.00% 51.29% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdMisc 1000 0.11% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdMult 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdMultAcc 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdMatMultAcc 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdShift 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdShiftAcc 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdDiv 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdSqrt 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatAdd 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatAlu 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatCmp 0 0.00% 51.40% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatCvt 17307 1.86% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatDiv 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatMisc 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatMult 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatMultAcc 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatMatMultAcc 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatSqrt 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdReduceAdd 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdReduceAlu 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdReduceCmp 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatReduceAdd 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdFloatReduceCmp 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdAes 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdAesMix 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdSha1Hash 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdSha1Hash2 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdSha256Hash 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdSha256Hash2 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdShaSigma2 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdShaSigma3 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::SimdPredAlu 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::Matrix 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::MatrixMov 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::MatrixOP 0 0.00% 53.26% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::MemRead 311013 33.47% 86.73% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::MemWrite 85729 9.22% 95.95% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatMemRead 35614 3.83% 99.78% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::FloatMemWrite 2003 0.22% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::IprAccess 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::InstPrefetch 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorUnitStrideLoad 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorUnitStrideStore 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorUnitStrideMaskLoad 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorUnitStrideMaskStore 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorStridedLoad 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorStridedStore 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorIndexedLoad 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorIndexedStore 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorUnitStrideFaultOnlyFirstLoad 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorWholeRegisterLoad 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorWholeRegisterStore 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorIntegerArith 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorFloatArith 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorFloatConvert 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorIntegerReduce 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorFloatReduce 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorMisc 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorIntegerExtension 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::VectorConfig 0 0.00% 100.00% # Class of committed instruction. (Count)

board.processor.cores0.core.commitStats0.committedInstType::total 929344 # Class of committed instruction. (Count)

**X86 sort loop instruction mix:**

.L3:

movl -48(%rbp), %eax

cltq

leaq 0(,%rax,8), %rdx

movq -40(%rbp), %rax

leaq (%rdx,%rax), %rbx

call drand48@PLT

movq %xmm0, %rax

movq %rax, (%rbx)

movl -48(%rbp), %eax

cltq

leaq 0(,%rax,8), %rcx

movl -48(%rbp), %eax

cltq

leaq 0(,%rax,8), %rdx

movq -32(%rbp), %rax

addq %rdx, %rax

movq -40(%rbp), %rdx

addq %rcx, %rdx

movq %rdx, (%rax)

addl $1, -48(%rbp)

.L2:

movl -48(%rbp), %eax

cmpl -44(%rbp), %eax

jl .L3

movl $0, -48(%rbp)

jmp .L4

.L5:

movl -48(%rbp), %eax

cltq

leaq 0(,%rax,8), %rdx

movq -32(%rbp), %rax

addq %rdx, %rax

movq (%rax), %rcx

movl -48(%rbp), %eax

cltq

leaq 0(,%rax,8), %rdx

movq -40(%rbp), %rax

addq %rdx, %rax

movq (%rax), %rdx

movq -24(%rbp), %rax

movq %rcx, %rsi

movq %rdx, %xmm0

movq %rax, %rdi

call pq\_push@PLT

addl $1, -48(%rbp)

.L4:

movl -48(%rbp), %eax

cmpl -44(%rbp), %eax

jl .L5

movl $0, -48(%rbp)

jmp .L6

.L7:

movl -48(%rbp), %eax

cltq

leaq 0(,%rax,8), %rdx

movq -32(%rbp), %rax

leaq (%rdx,%rax), %rbx

movq -24(%rbp), %rax

movq %rax, %rdi

call pq\_pop@PLT

movq %rax, (%rbx)

addl $1, -48(%rbp)

.L6:

movl -48(%rbp), %eax

cmpl -44(%rbp), %eax

jl .L7

movl $0, -48(%rbp)

jmp .L8

.L10:

movl -48(%rbp), %eax

cltq

leaq 0(,%rax,8), %rdx

movq -32(%rbp), %rax

addq %rdx, %rax

movq (%rax), %rax

testq %rax, %rax

je .L9

movl -48(%rbp), %eax

cltq

leaq 0(,%rax,8), %rdx

movq -32(%rbp), %rax

addq %rdx, %rax

movq (%rax), %rax

movq (%rax), %rax

movq %rax, %xmm0

leaq .LC0(%rip), %rax

movq %rax, %rdi

movl $1, %eax

call printf@PLT

.L9:

addl $1, -48(%rbp)

.L8:

movl -48(%rbp), %eax

cmpl -44(%rbp), %eax

jl .L10

movq -24(%rbp), %rax

movq %rax, %rdi

call pq\_destroy@PLT

movq -40(%rbp), %rax

movq %rax, %rdi

call free@PLT

movq -32(%rbp), %rax

movq %rax, %rdi

call free@PLT

movl $0, %eax

movq -8(%rbp), %rbx

leave

.cfi\_def\_cfa 7, 8

ret

.cfi\_endproc

**For RISCV:**

**Sort loop instructions**:

.L2:

call drand48

fsd fa0,0(s0)

sd s0,0(s3)

addi s0,s0,8

addi s3,s3,8

bne s0,s5,.L2

li a1,0

li a0,0

call m5\_dump\_reset\_stats

mv s0,s6

.L3:

ld a1,0(s0)

fld fa0,0(s2)

mv a0,s4

call pq\_push

addi s2,s2,8

addi s0,s0,8

bne s2,s5,.L3

addi s0,s6,80

.L4:

mv a0,s4

call pq\_pop

sd a0,0(s1)

addi s1,s1,8

bne s1,s0,.L4

li a1,0

li a0,0

call m5\_dump\_reset\_stats

mv a0,s4

call pq\_destroy

mv a0,s7

call free

mv a0,s6

call free

li a0,0

ld ra,72(sp)

.cfi\_restore 1

ld s0,64(sp)

.cfi\_restore 8

ld s1,56(sp)

.cfi\_restore 9

ld s2,48(sp)

.cfi\_restore 18

ld s3,40(sp)

.cfi\_restore 19

ld s4,32(sp)

.cfi\_restore 20

ld s5,24(sp)

.cfi\_restore 21

ld s6,16(sp)

.cfi\_restore 22

ld s7,8(sp)

.cfi\_restore 23

addi sp,sp,80

.cfi\_def\_cfa\_offset 0

jr ra

.cfi\_endproc

.LFE22:

.size main, .-main

.ident "GCC: (gc891d8dc23e) 13.2.0"

.section .note.GNU-stack,"",@progbits

The CPUs MINOR and O3 were built for RISCV and X86 respectively were built using the config script for the gem5:

The load was taken for 1000,10,10000 sizes for both min\_heap and linked\_list implementations with no optimizations of the compiler

In both the architectures, Considering the minheap implementation of the priority queue, as the memory locations are accessed with continguos memory addresses, this implementation exploits the spatial locality of the cache and generates fewer misses relative to the linked list implementation in which the addresses are scattered. This is not a surprising result.

O1, O2, O3 optimiation over O3 deriv CPU for X86 ISA:

Minheap:

n =10000:

A graph with red rectangles and numbers

Description automatically generated

N=1000:

A graph with red rectangles

Description automatically generated

N=10:

A graph with red rectangles

Description automatically generated

Linklist\_:

N=10000

A graph with red squares

Description automatically generated

N=1000:A graph with red squares

Description automatically generated

N:10:

A graph with red rectangles

Description automatically generated

As we can see from the above graphs, O3 and O2 optimizations execute in the same time where as O1 optimization drastically differs with a large input.

From the above results, and analysis. We can determine that the minheap implementation is faster than the linklist implementation. I believe the main cause is that, the cache accesses of the array addresses, are faster in a usual case because of spatial locality, while the address spaces of the link list are scattered all over the memory which cannot exploit spatial locality of the caches.