AMD EPYC™ 7xx2-series Processors
Compiler Options Quick Reference Guide

AOCC compiler (with Flang - Fortran Front-End)

Latest release: 2.1, Nov 2019

https://developer.amd.com/amd-aocc/

Architecture	
Generate instructions that run on 2nd Gen EPYC/ RYZEN	-march=znver2
Generate instructions for the local machine	-march=native

Optimization Levels	
Disable all optimizations	-O0
Minimal level speed and code optimization	-O1
Moderate level optimization (default)	-O2/ -O
Aggressive optimizations	-O3
Maximize performance	-Ofast
Enable Link Time Optimization	-flto
Enable unrolling	-funroll-loops
Enable aggressive loop optimizations	-enable-loop-versioning-licm -enable-partial-unswitch -unroll-aggressive
Enable aggressive inline optimizations	-function-specialize -finline-aggressive
Enable aggressive vectorization	-enable-strided-vectorization -enable-epilog-vectorization
Enable memory layout optimizations	-fremap-arrays (use with -flto)
Profile Guided optimizations	-fprofile-instr-generate (1st invocation) -fprofile-instr-use (2nd invocation)
OpenMP®	-fopenmp
For enabling streaming stores, memory bandwidth workloads	-fnt-store
Enable removal of un-used array computation	-reduce-array-computation=3

Other options	
Enable faster, less precise math operations	-ffast-math -freciprocal-math
OpenMP® threads and affinity (N number of cores)	export OMP_NUM_THREADS=N export GOMP_CPU_AFFINITY="0-{N-1}"
Enabling Vector library	-vector-library=LIBMVEC
Link to Vector library	-L/libm-install-dir/lib -lmvec
Link to AMD library	-L/libm-install-dir/lib -lamdlibm

For Fortran workloads	
Compile free form FORTRAN	-ffree-form
Enable precise math operations in FORTRAN	-kieee

AMD Optimized Libraries

Latest release: 2.1, Jan 2020

https://developer.amd.com/amd-aocl/

AMD µProf (Performance & Power Profiler)

Latest release: 3.2, Nov 2019

https://developer.amd.com/amd-uprof/

AMD EPYC™ 7xx2-series Processors
Compiler Options Quick Reference Guide

GNU compiler collection (gcc, g++, gfortran)

Latest release: 9.3, Mar 2020

Recommended version : 9.3

http://gcc.gnu.org

Architecture	
Generate instructions that run on 2nd Gen EPYC/ RYZEN	-march=znver2
Generate instructions for the local machine	-march=native

Optimization Levels	
Disable all optimizations (default)	-O0
Minimal level speed and code optimizations	-O1/ -O
Moderate level optimizations	-O2
Aggressive optimizations	-O3
Maximize performance	-Ofast

Additional Optimizations	
Link time optimization	-flto
Enable unrolling	-funroll-all-loops
Generate memory preload instructions	-fprefetch-loop-arrays --param prefetch-latency=300
Profile-guided optimization	-fprofile-generate (1st invocation) -fprofile-use (2nd invocation)
OpenMP®	-fopenmp

Other options	
Enable generation of code that follows IEEE arithmetic	-mieee-fp
Enable faster, less precise math operations	-ffast-math
Compile free form FORTRAN	-ffree-form
OpenMP® threads and affinity (N number of cores)	export OMP_NUM_THREADS=N export GOMP_CPU_AFFINITY="0-{N-1}"
Link to AMD library	-L/libm-install-dir/lib -lamdlibm

GlibC

Latest release: 2.31, Jan 2020

Recommendation : 2.26 or later

https://www.gnu.org/software/libc/

Binutils

Recommendation: 2.26.1 or later

https://www.gnu.org/software/binutils/

Microsoft Visual Studio 2019

Latest stable release : 16.5, Mar 2020

https://www.visualstudio.com/

User Guide

Architecture	
Generate instructions that run on 2nd Gen EPYC/ RYZEN	/arch:[AVX|AVX2]
Optimize for 64-bit AMD processors	/favor:AMD64 /d2vzeroupper

Optimization Levels	
Disable optimizations	/Od
Maximum optimizations (favor space)	/O1
Maximum optimizations (favor speed)	/O2
[link.exe] Eliminate unreferenced function and/ or data	/OPT:REF
[link.exe] Perform identical COMDAT folding	/OPT:ICF
Output an informational message for loops that are auto-vectorized	/Qvec-report:[1|2]
Enable automatic parallelization of loops, used in conjunction with #pragma loop() directive	/Qpar
Output an informational message for loops that are auto-parallelized	/Qpar-report:[1|2]

Additional Optimizations	
Maintain the precision for floating-point operations through proper rounding	/fp:precise
Optimize floating-point code for speed at the expense of floating-point accuracy and correctness	/fp:fast
Whole Program Optimization (link-time code generation)	/GL
Profile-guided optimization	LTCG:PGI and /LTCG:PGO

Intel compilers (icc, icpc, ifort)

Latest release: 19.1

http://software.intel.com

Architecture	
Generate instructions that run on 2nd Gen EPYC/RYZEN	-march=core-avx2 (preferred) **OR** -axCORE-AVX2
Optimization Levels	
Disable all optimizations	-O0
Speed optimization without code growth	-O1
Enable optimization for speed including vectorization	-O2
Aggressive optimization	-O3
Maximize performance	-Ofast
Additional Optimizations	
Aggressive unrolling	-unroll-aggressive
Disable improved precision floating divides	-no-prec-div
Enable vectorization	-vec
Inter procedural Optimization	-ipo
OpenMP®	-qopenmp
Prefetch optimization	-qopt-prefetch
Profile generated optimization	-prof-gen and -prof-use
Use optimized header definitions	-use-intel-optimized-headers
Other options	
Floating point accuracy tuning	-fp-model
Compile free form FORTRAN	-free