



AOCL 4.1 Release Notes

© 2023 Advanced Micro Devices, Inc. All rights reserved.

The information contained herein is for informational purposes only, and is subject to change without notice. While every precaution has been taken in the preparation of this document, it may contain technical inaccuracies, omissions and typographical errors, and AMD is under no obligation to update or otherwise correct this information. Advanced Micro Devices, Inc. makes no representations or warranties with respect to the accuracy or completeness of the contents of this document, and assumes no liability of any kind, including the implied warranties of noninfringement, merchantability or fitness for particular purposes, with respect to the operation or use of AMD hardware, software or other products described herein. No license, including implied or arising by estoppel, to any intellectual property rights is granted by this document. Terms and limitations applicable to the purchase or use of AMD's products are as set forth in a signed agreement between the parties or in AMD's Standard Terms and Conditions of Sale. Any unauthorized copying, alteration, distribution, transmission, performance, display or other use of this material is prohibited.

Trademarks

AMD, the AMD Arrow logo, AMD AllDay, AMD Virtualization, AMD-V, PowerPlay, Vari-Bright, and combinations thereof are trademarks of Advanced Micro Devices, Inc. Other product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

Dolby is a trademark of Dolby Laboratories.

HDMI is a trademark of HDMI Licensing, LLC.

HyperTransport is a licensed trademark of the HyperTransport Technology Consortium.

Microsoft, Windows, Windows Vista, and DirectX are registered trademarks of Microsoft Corporation in the US and/or other countries.

MMX is a trademark of Intel Corporation.

OpenCL is a trademark of Apple Inc. used by permission by Khronos.

PCIe is a registered trademark of PCI-Special Interest Group (PCI-SIG).

USB Type-C® and USB-C® are registered trademarks of USB Implementers Forum.

Reverse engineering or disassembly is prohibited.

USE OF THIS PRODUCT IN ANY MANNER THAT COMPLIES WITH THE MPEG ACTUAL OR DE FACTO VIDEO AND/OR AUDIO STANDARDS IS EXPRESSLY PROHIBITED WITHOUT ALL NECESSARY LICENSES UNDER APPLICABLE PATENTS. SUCH LICENSES MAY BE ACQUIRED FROM VARIOUS THIRD PARTIES INCLUDING, BUT NOT LIMITED TO, IN THE MPEG PATENT PORTFOLIO, WHICH LICENSE IS AVAILABLE FROM MPEG LA, L.L.C., 6312 S. FIDDLERS GREEN CIRCLE, SUITE 400E, GREENWOOD VILLAGE, COLORADO 80111.

Contents

Contents	3
Release Highlights	4
AOCL-LibM	4
AOCL-FFTW	4
AOCL-BLAS	5
AOCL-LAPACK	6
AOCL-ScaLAPACK	6
AOCL-Sparse	6
AOCL-RNG	6
AOCL-Cryptography	7
AOCL-LibMem	7
AOCL-Compression	7
AOCL-Utils	8

Release Highlights

In this release, all the AOCL libraries support AMD “Zen4” configuration.

AOCL-LibM

- Added erf function
- Added vector array arithmetic functions:
 - vrsa_addf, vrsa_addfi, vrda_add, and vrda_addi
 - vrsa_subf, vrsa_subfi, vrda_sub, and vrda_subi
 - vrsa_mulf, vrsa_mulfi, vrda_mul, and vrda_muli
 - vrsa_divf, vrsa_divfi, vrda_div, and vrda_divi
- Added vector functions:
 - vrd2_erf, vrd4_erf and vrd8_erf
 - vrs16_erff
 - vrd4_sincos and vrd8_sincos
 - vrs16_tanhf, vrs16_cosf, and vrs16_acosf
- Optimized the performance of:
 - vrd2_pow, vrd4_pow and vrd8_pow
 - exp2 and log2
- Improved the accuracy of atan and cos
- Added optimized versions of fminf, fmin, remainderf, remainder, nearbyintf, nearbyint, and cbrt
- Added support for the AOCL-Utills library
- Fast scalar variants:
 - Added fasterf
 - Improved fasterff and fastpowf

Note: For more information on fast scalar variants, refer to the AOCC 4.1 user guide.

AOCL-FFTW

Dynamic dispatch for AOCC on Linux

AOCL-BLAS

- Following MatMul API's for INT8 and F32 added and optimized with post-ops support:
 - `aocl_gemm_s8s8s32os32()` using AVX512-VNNI
 - `aocl_gemm_s8s8s32os8()` using AVX-512-VNNI
 - `aocl_gemm_s8s8s16os16()` using AVX2
 - `aocl_gemm_s8s8s16os8()` using AVX2
 - `aocl_gemm_f32f32f32of32()` using AVX512
- Post-ops GELU_TANH, GELU_ERF, and CLIP added and optimized with a provision to call multiple element-wise post-ops per GEMM call.
- Following utility functions added and optimized for AVX2 and AVX512:
 - `aocl_gemm_tanh_gelu_f32()`
 - `aocl_gemm_tanh_erf_f32()`
 - `aocl_gemm_softmax_f32()`
- Improved performance of all the APIs in `aocl_gemm` add-on
- Dynamic dispatch and `amdzen` configuration support added to `aocl_gemm` add-on
- Dynamic dispatch supports different models within the AMD "Zen3" and "Zen4" configurations. For these architecture configurations, the selected model can be controlled using the environment variable `BLIS_MODEL_TYPE`. Differences in model code paths are limited at present but may increase in future releases.
- Dynamic dispatch will now select the AMD "Zen3" or AMD "Zen4" code paths (if enabled) automatically on future AMD processors (those not directly supported in this release), and on suitable Intel processors based on AVX2 or AVX512 instruction support. No specific optimizations are done for the non-Zen3/4 processors, but these code paths should perform better than the fallback "generic" code path.
- AVX 512-based optimizations for AMD "Zen4" platform:
 - SGEMM, DGEMM, and ZGEMM
 - DTRSM
 - D/ZAXPY, ZGEMV, DDOTV, and D/ZSCALV
- Improved support for OpenMP nested parallelism, that is, calling AOCL-BLAS routines within an OpenMP parallel region and controlling threads through OpenMP mechanisms only. BLIS-specific threading control mechanisms remain an option.

AOCL-LAPACK

- Upgraded to Netlib LAPACK 3.11.0 specification that includes 14 new APIs, bug fixes, and improvements to the existing APIs
- Improved performance of the following APIs:
 - Double Precision SVD (DGESVD) for small matrices
 - Factorization routines (DGEQP3, SGETRF, ZGETRF, and DGETRF)
 - Double Precision solver API DGESV
- Framework changes to support automatic selection of optimized routines based on target CPU
- Unified CMake build system supporting Linux and Windows
- Build system cleanup and minor bug fixes

***Note:** AOCL-LAPACK 4.1 has dependency on libstdc++ library. On Linux, you must link libstdc++ (-lstdc++) when the AOCL-LAPACK library is used.*

AOCL-ScaLAPACK

- Trace and Logging feature enabled for real double precision APIs
- Runtime enablement of Trace, Logging, and API Progress tracking features through environment variables
- Build system cleanup and minor bug fixes

AOCL-Sparse

- Performance improvement on SpMV AVX512 kernels
- Dynamic dispatch for SpMV on AMD “Zen” architectures (AMD “Zen1”, AMD “Zen2”, AMD “Zen3”, and AMD “Zen4”)
- APIs to support HPCG
- Reusable kernel templates to enable quick design and implementation of level 1 and level 2 kernels
- Integration with AOCL-BLAS and AOCL-LAPACK for level 1 APIs
- Minimum CMake requirement upgraded to 3.11
- Minor bug fixes

AOCL-RNG

- AVX512 support for double precision SFMT Generator
- Improved performance of Gaussian Distribution Generator
- AVX512 and AVX2 support for Double Precision Gaussian Distribution Generator
- Framework improvements for selection of the optimized path based on the target CPU instruction set for Mersenne Twister, SFMT, and Gaussian Distribution routines

AOCL-Cryptography

- Improvements in AES-GCM encrypt/decrypt routines for AMD “Zen” architecture
- Improved SHA3 digest routines including SHAKE128 and 256 schemes
- AES CCM encrypt/decrypt routines
- HMAC SHA2 and SHA3 routines
- AES CMAC routines for key sizes (128, 192, and 256)
- AES SIV routines
- ECDH x25519 key exchange functions: Generate Public Key and Compute Secret Key
- RSA Functions: Encrypt Text with public key and Decrypt with private key.

Note: Only non-padded mode is supported in this release.

- Dynamic dispatch on different x86-64 ISA architectures support
- Windows support

AOCL-LibMem

- Added AVX-512-based optimizations
- Added support for strcpy function
- Added support for static binary build

Note: Different binaries are available for AVX-512 and AVX2. For more information, refer to the [AOCL-LibMem](#) web page.

AOCL-Compression

- Performance optimizations of all the AOCL-Compression supported methods ZLIB, ZSTD, LZ4, BZIP2, LZMA, LZ4HC, and Snappy
- Dynamic dispatch support
- AVX512 CPU feature detection
- Doxygen-based documentation covering library's API level details
- Benchmarking of IPP library methods library (LZ4, LZ4HC, ZLIB, and BZIP2) supported in the AOCL-compression test framework
- GTest-based unit testing for all the compression methods
- CTest-based test suite that runs GTest unit tests and AOCL Test Bench for Silesia, Calgary, and Canterbury datasets
- Python-based performance benchmarking automation script
- Custom build options to exclude the unnecessary compression methods from the library build to achieve a lower code footprint
- Compiler flags for enforcing C and C++ standards
- Minor bug fixes

AOCL-Utills

- APIs to get Instruction Set Architecture (ISA) details
- API to check following CPU features:
 - SHA, AES, and VAES availability
 - RDSEED and RDRAND availability
 - AVX2 availability
 - AVX512 foundation and sub-feature flags
- APIs for cache topology
- Linux and Windows support