



AOCL 4.2 Release Notes

© 2024 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

Contents3

Release Highlights4

 AOCL-Compression4

 AOCL-Cryptography4

 AOCL-LibMem5

 AOCL-BLAS5

 AOCL-LAPACK5

 AOCL-LibM6

 AOCL-Sparse.....6

 AOCL-ScaLAPACK7

 AOCL-RNG7

 AOCL-Utils7

Release Highlights

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

AOCL-Compression

- Single-threaded performance improvements for BZIP2, LZ4, LZ4HC, LZMA, Snappy, ZLIB, and ZSTD improvements
- Multi-threaded compression and decompression support for LZ4, Snappy, ZLIB, and ZSTD
- Support to select specific ISA for execution using environment variables
- Support for LOGGING mechanism using environment variables
- XZ utils interface support for basic APIs for LZMA method
- Improved unit testing of Lib by adding new test cases
- VALGRIND and ASAN tools added to detect memory related errors
- GCOV tool added to measure source code coverage
- Support to test native APIs for compression methods directly from test bench
- Test bench support to decompress large files and dumping of output to file with standard extension
- Added standard binaries name for the compression method supported using symbolic link
- AOCL-Compression’s ZLIB, LZMA, and ZSTD reference version (opensource) upgrade to version v1.3, SDK v22.01, and v1.5.5
- Refactoring and code alignment

AOCL-Cryptography

- Added Poly1305 MAC and ChaCha20 Stream Cipher algorithm support
- RSA encryption/decryption functions improved for 1024, 2048-bit key sizes. Also, added OAEP Padding support
- Architectural improvements in AES Cipher: Split algorithms into AEAD and Non-AEAD
- Performance improvements in:
 - AES cipher (GCM, CTR, XTS, CFB, and CBC) encrypt/decrypt algorithms for AMD “Zen4” architecture
 - Digest SHA2-512, SHA2-256, and dependent algorithms
- AES-XTS support added for block number-based encryption/decryption
- IPP and OpenSSL Compat support for more AEAD APIs
- IPP Compat support for MAC
- Extended IPP Compat support for XTS
- Improved dynamic dispatcher

AOCL-LibMem

- Added support for new string functions strncpy, strcmp, strncmp, and strlen
- Optimized memmove, memcmp, memset, and strcpy
- CTest-based functional validation.

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

AOCL-BLAS

- For LPGEMM:
 - Added uint8 output and zero-point support for int8 APIs
 - Added transA, transB support for bf16 API
 - Improved multithread performance
 - Fixed a few accuracy issues in post-ops, such as GELU_TANH and GELU_ERF
- Introduced AOCL_ENABLE_INSTRUCTIONS environment variable as an alternative to BLIS_ARCH_TYPE, but with slightly different semantics.
- Improved functionality of XERBLA error handling routine in AOCL-BLAS, giving options to control printing on error, stopping on error, and returning error value to the calling program.
- Performance optimizations for the following APIs:
 - DGEMM for tiny sizes
 - S/ZGEMM, D/ZTRSM, ZAXPBYV, Z/ZDSCALV, S/D/ZGEMV, and D/DZNRM2
- Following BLAS extension APIs have been added only for AMD “Zen” code paths:
 - sgemm_pack_get_size(), sgemm_pack(), and sgemm_compute()
 - dgemm_pack_get_size(), dgemm_pack(), and dgemm_compute()

AOCL-LAPACK

- Improved performance of the following APIs:
 - Double Precision SVD (DGESVD)
 - Factorization routines DGETRF and ZGETRF
 - Solver routines DGETRS and DGESV
- Option to link with AOCL-BLAS during build to enable invoking AOCL-BLAS internal APIs
- Applications using AOCL-LAPACK need to additionally link with AOCL-Utils library
- CMake improvements:
 - Ease of running tests using Ctest
 - ISA specific flags configurable during build

- Code coverage build option
- OpenMP parallelism enabled in the APIs {c,z}hetrd_hb2st, {s,d}sytrd_sb2st, and iparam2stage to match LAPACK 3.11 implementation.

Note: AOCL-Utills library has libstdc++ library dependency. As AOCL-LAPACK is dependent on AOCL-Utills, applications running on Linux must additionally link with libstdc++(-lstdc++).

AOCL-LibM

- Added new vector array functions:
 - vrsa_fmaxf, vrsa_fmaxfi, vrda_fmax and vrda_fmaxi
 - vrsa_fminf, vrsa_fminfi, vrda_fmin and vrda_fmini
- Added new functions to the fast scalar library, libalmfast::
 - pow
 - sin and sinf
 - cos and cosf
 - acosf
- Optimized the performance of:
 - log
 - vrd8_exp2
- Optimized the performance and improved the accuracy of:
 - tanhf, vr4s_tanhf, vr8s_tanhf and vr16s_tanhf
 - atan
- Added optimized versions of fmax, fmaxf, fmod, fmodf, fdim and fdimf.
- Improved the examples.

AOCL-Sparse

- Support for one-based indexing and complex data types
- New APIs added:
 - Level 1 APIs: aoclsparse_?gthr, aoclsparse_?gthrz, aoclsparse_?gthrs, aoclsparse_?sctr, aoclsparse_?sctrs, aoclsparse_?dotci, aoclsparse_?dotui, aoclsparse_?doti, aoclsparse_?roti, aoclsparse_?axpyi
 - Level 2 API: aoclsparse_?dotmv
 - Level 3 APIs: aoclsparse_?csrmm, aoclsparse_sp2m, aoclsparse_?add, aoclsparse_?trsm
 - Auxiliary APIs: aoclsparse_copy, aoclsparse_order_mat, aoclsparse_export_?csr, aoclsparse_export_?csc, aoclsparse_convert_csr
- Framework enhancements
- Integration with AOCL-Utills for detecting supported ISA

AOCL-ScaLAPACK

- Additional wrapper interfaces added to BLACS routines to support upper/lower case and with/without underscore suffix in the API name
- Tracing and Logging feature enabled for all double, single complex, and double complex precision APIs

AOCL-RNG

- AVX512 support for single precision SFMT Generator and Gaussian Distribution Generator
- AVX2 support for single precision Gaussian Distribution Generator
- Improved performance of double precision SFMT Generator and Gaussian Distribution Generator
- Sphinx-based documentation support

AOCL-Utills

- APIs to check the following cache features:
 - Number of logical processors sharing cache
 - Number of physical partitions
 - Fully associative
 - Self-initializing
 - Cache Inclusive/Exclusive
- Added BUILD_SHARED_LIBS flag to build shared library