The OpenMP API for Parallel Programming

OpenMP®

OpenMP ARB Participants

Companies participating in the OpenMP Architecture Review Board (ARB) work to maintain and promote the OpenMP API.

- AMD
- Argonne National Laboratory
- Arm
- Barcelona Supercomputing Center
- Brookhaven National Laboratory
- cOMPunity
- EPCC
- Fujitsu
- Hewlett-Packard Enterprise
- IBM
- Inria
- Intel
- Lawrence Berkeley National Lab
- Lawrence Livermore National Lab
- Leibniz Supercomputing Centre
- Los Alamos National Laboratory
- Maui High Performance Computing Ctr.
- Mentor Graphics
- Micron
- NASA
- NEC
- NVIDIA
- Oak Ridge National Laboratory
- RWTH Aachen University
- Sandia National Laboratories
- Stony Brook University
- SUSE
- Texas Advanced Computing Center
- Univerist of Basel
- University of Bristol
- University of Delaware
- The University of Manchester
- University of Tennessee

Want to help shape OpenMP?

Join the OpenMP ARB

Contact us at **info@openmp.org**

The OpenMP® Application Program Interface (API) standardizes directive-based multi-language high-level parallelism that is performant, productive, and portable.

The OpenMP API

The OpenMP API supports parallel programming in C, C++, and Fortran on a range of platforms. It is jointly defined by a group of major computer hardware vendors, software vendors, software developers, and the OpenMP community. The OpenMP API provides a portable, scalable programming model that gives parallel programmers a simple and flexible interface for developing parallel applications for platforms ranging from shared-memory and multicore systems to embedded systems and coprocessor/accelerator devices.

The OpenMP ARB

The OpenMP Architecture Review Board (ARB) is the non-profit corporation that owns the OpenMP API, maintains existing specifications, and produces new versions. The ARB helps to promote the OpenMP API and to organize and fund conferences, workshops, tutorials, and other related events. Visit us online to read about OpenMP news, download the latest specification, get answers to questions about using the OpenMP API, and gain access to presentations, reference cards, and more. **www.openmp.org**

International Workshop on the OpenMP API (IWOMP)

The International Workshop on OpenMP (IWOMP) is an annual workshop to advance all aspects of parallel programming with the OpenMP API, including research ideas and results related to parallel programming using the OpenMP API. Since the first IWOMP workshop in 2005, these events have been held in many different countries around the world. **www.iwomp.org**

OpenMPCon: The OpenMP Users Convention

OpenMPCon is the convention by and for OpenMP users. Enjoy keynotes, inspirational talks, and a friendly atmosphere. Co-located with IWOMP, diverse technical tracks are designed to appeal to anyone, from the OpenMP novice to the seasoned expert. **www.openmpcon.org**

UK OpenMP User Conference

The annual UK OpenMP Users Conference provides two days of talks and workshops aimed at furthering collaboration and knowledge sharing among the UK community of expert and novice high-performance computing specialists using OpenMP. **www.ukopenmpusers.co.uk**

cOMPunity: The Community of OpenMP Users

cOMPunity is the community of OpenMP researchers and developers. The cOMPunity website is a repository of resources for learning about the OpenMP API, reading about the experiences of application developers who use the language, finding out about recent research activities and upcoming events, as well as obtaining OpenMP freeware and benchmarks.

Learn more at www.openmp.org

Getting Started With the OpenMP API

It's easy to add parallelism to your C, C++, or Fortran program using the OpenMP API. Typically you would use profiling tools first to determine which parts of the program could benefit from parallel execution using multiple threads. Next, simply use the appropriate `#pragma omp` C or C++ pragma or `!$OMP` Fortran directive to inform the compiler where the parallelism is. Next, simply use the appropriate C/C++ pragma or Fortran directive to inform performance hot spots.

In the examples below, the developer inserted the pragma to specify that this loop can be executed in parallel. That is all that is needed. The compiler generates the parallel code to create the threads and distribute the loop iterations over the threads to speed up the computation.

C/C++ example:

```
#pragma omp parallel for
for (i=0; i < n; i++) {
   y[i] = a * x[i] + y[i];
}
```

Fortran example:

```
!$omp parallel do
do i=1,n
   y(i) = a * x(i) + y(i)
end do
```

All that is required is a compiler that supports the OpenMP API. See **www.openmp.org/tools** for a list of open source and commercial OpenMP API compilers supplied by the members of the OpenMP ARB and other vendors.

Components of the OpenMP API

Directives: Parallel region definition and control, device and worksharing constructs, tasking, synchronization, offload to coprocessors/accelerators, and data-sharing attributes.

Runtime environment: Number of threads, thread ID, dynamic thread adjustment, nested parallelism, workload distribution schedule, active nesting levels, limit on number of threads, nesting level, ancestor thread ID, team size, wallclock timer, and locking.

Environment variables: Number of threads, scheduling type, dynamic thread adjustment, nested parallelism, thread stacksize, behavior of idle threads, active nesting levels, and limit on number of threads

OpenMP 5.1 API Features

Features in the 5.1 release of the OpenMP API include:

- The `assume` directive: Supports optimization hints based on invariants and promises to limit OpenMP usage to (optimizable) subsets.
- Loop transformation directives `tile` and `unroll`: Control the use of traditional sequential optimizations and ensure that loops are applied appropriately relative to parallelization.
- The `masked` construct supports filtering execution per thread.
- Extends `atomic` construct to support compare-and-swap, min and max.
- Adds many clauses and clause modifiers.
- Added extensions:
  - Full support for C11, C++11, C++14, C++17, C++20
  - Full support for Fortran 2008 and partial support for Fortran 2018
  - Extends directive syntax to C++ attribute specifiers
  - The `error` directive supports user-defined warnings and errors
  - The `scope` construct supports reductions within parallel regions
  - Support for mapping (translated) function pointers
  - Device-specific environment variables to control their ICVs
  - The `nothing` directive supports `metadirective` clarity and completeness

Learn more about OpenMP and download the specification at www.openmp.org

OpenMP at SC'20

The annual Supercomputing conference provides the high-performance computing commuity with technical programs that make it a yearly must-attend forum. Following is an overview of the OpenMP activities avaiable in conjunction with the virtual SC'20.

OpenMP SC20 Tutorials (November 9-10, 2020)

The OpenMP Common Core: A hands-on Introduction

In this tutorial, students use active learning through a carefully selected set of exercises to master the Common Core of the 21 foundational items from the OpenMP API and learn to apply them to their own problems.

Speakers: Tim Mattson, Yun (Helen) He, Alice Koniges, David Eder

Advanced OpenMP: Host Performance and 5.0 Features

This tutorial will focus on performance programming for multi-core architectures, including data/thread locality, false sharing, and exploitation of vector units.

Speakers: Christian Terboven, Michael Klemm, Ruud van der Pas, Bronis R. de Supinski.

Programming your GPU with OpenMP: A hands-on Introduction

This tutorial explores OpenMP target directives that map code and data onto a potentially diverse set of devices, as modern platforms are increasingly heterogeneous, with CPU cores, GPU cores, and other specialized accelerators..

Speakers: Simon McIntosh-Smith, Tom Deakin

OpenMP SC20 BOF

OpenMP API Version 5.1, released just before SC'20, is the next step of bringing the OpenMP API into the exascale era. The ARB undertook great effort to improve the features that Version 5.0 introduced, adding new, powerful features to support the heterogeneous computing world. In this BOF, hosted by OpenMP ARB member Jeff Larkin and OpenMP ARB CEO Michael Klemm, attendees will get first-hand information from OpenMP ARB members and have a chance to engage with the representatives of the OpenMP ARB, vendors, and users in an interactive discussion, ask questions, and provide their feedback as users.

OpenMP SC20 Booth Talks

When the SC show is live, OpenMP always has a booth where attendees can join us for casual conversation or to ask questions of our OpenMP API experts. We also have a series of Booth Talks that feature OpenMP innovations and tips. Download the SC20 Booth Talk videos and PDFs at link.openmp.org/sc20/

- OpenMP 5.1: Overview
- OpenMP 5.1 Features: The Dispatch Construct
- OpenMP 5.1 Features: The Interop Construct
- OpenMP 5.1 Features: The Assume Construct
- OpenMP 5.1 Features: Loop Transformation Constructs
- OpenMP ARB Membership
- Performance Portability of OpenMP on CPUs and GPUs
- Be Lazy and Get Good OpenMP Performance
- LB4OMP: A Load Balancing Portfolio for OpenMP
- Parallelware Analyzer: Data race detection for GPUs using OpenMP
- SOLLVE: OpenMP compiler optimizations in LLVM
- SOLLVE: Gauging OpenMP 5.0 Status Using SOLLVE V&V
- BOLT: A Lightweight and Highly Interoperable OpenMP Runtime
- Exascale Application: Port QMCPack for exascale using OpenMP
- Exascale Application: Port GESTS to Summit with OpenMP
- Exascale Application: Port GenASiS to accelerators with OpenMP
- Exascale Application: Port GAMESS to GPUs using OpenMP
- Exascale Library: Optimize SLATE using OpenMP tasks
- SOLLVE: Locality-sensitive Loop Scheduling in OpenMP

Learn more about OpenMP and download the specification at www.openmp.org