

● Full vendor support	● Indirect, but comprehensive support, by vendor	★ Limited, probably indirect support – but at least some
■ Vendor support, but not (yet) entirely comprehensive	▲ Comprehensive support, but not by vendor	✍ No direct support available, but of course one could ISO-C-bind your way through it or directly link the libraries

	CUDA		HIP		SYCL		OpenACC		OpenMP		Standard		Kokkos		ALPAKA		Python
	C++	Fortran	C++	Fortran	C++	Fortran	C++	Fortran	C++	Fortran	C++	Fortran	C++	Fortran	C++	Fortran	Python
NVIDIA	●1	●2	●3	▲4	▲5	✍6	●7	●8	■9	■10	●11	●12	▲13	★14	▲15	✍16	■▲17
AMD	●18	★19	●20	✍4	●21	✍6	▲22	★23	●24	●24	✍25	✍25	▲26	★14	▲27	✍16	★28
Intel	●29	✍30	★31	✍4	●32	✍6	★33	★33	●34	●34	✍35	■36	▲37	★14	★38	✍16	★39

- 1: CUDA C/C++, supported through CUDA Toolkit
- 2: CUDA Fortran, proprietary Fortran extension supported by NVIDIA HPC SDK
- 3: HIP programs can directly use NVIDIA GPUs via a CUDA backend; HIP is maintained by AMD
- 4: No such thing like HIP for Fortran
- 5: SYCL can be used on NVIDIA GPUs with *experimental* support either in SYCL directly or in DPC++, or via [hipSYCL](#)
- 6: No such thing like SYCL for Fortran
- 7: OpenACC C/C++ supported on NVIDIA GPUs directly (and best) through NVIDIA HPC SDK; additional, somewhat limited support by GCC C compiler and Clacc
- 8: OpenACC Fortran supported on NVIDIA GPUs directly (and best) through NVIDIA HPC SDK; additional, somewhat limited support by GCC Fortran compiler and [Flacc](#)
- 9: OpenMP in C supported on NVIDIA GPUs through NVIDIA HPC SDK (but not full OpenMP feature set available), by GCC, and Clang
- 10: OpenMP in Fortran supported on NVIDIA GPUs through NVIDIA HPC SDK (but not full OpenMP feature set available), by GCC, and Flang
- 11: pSTL features supported on NVIDIA GPUs through NVIDIA HPC SDK
- 12: Standard Language parallel features supported on NVIDIA GPUs through NVIDIA HPC SDK
- 13: Kokkos supports NVIDIA GPUs by calling CUDA as part of the compilation process
- 14: Kokkos is a C++ model, but at least the authors provided an ISO C Binding example for Fortran
- 15: Alpaka supports NVIDIA GPUs by calling CUDA as part of the compilation process
- 16: Alpaka is a C++ model
- 17: There is a vast community of offloading Python code to NVIDIA GPUs, like CuPy, Numba, cuNumeric, and many others; NVIDIA actively supports a lot of them, but has no direct product like CUDA for Python; so, the status is somewhere in between
- 18: [hipify](#) by AMD can translate CUDA calls to HIP calls which runs natively on AMD GPUs
- 19: AMD offers a Source-to-Source translator to convert some CUDA Fortran functionality to OpenMP for AMD GPUs ([gpubfort](#)); in addition, there are ROCm library bindings for Fortran in [hipfort](#)
- 20: HIP is the preferred native programming model for AMD GPUs
- 21: SYCL can use AMD GPUs, for example with [hipSYCL](#) or [DPC++ for HIP AMD](#)
- 22: OpenACC C/C++ can be used on AMD GPUs via GCC or Clacc; also, [Intel's OpenACC to OpenMP Source-to-Source translator](#) can be used to generate OpenMP directives from OpenACC directives
- 23: OpenACC Fortran can be used on AMD GPUs via GCC; also, AMD's [gpubfort](#) Source-to-Source translator can move OpenACC Fortran code to OpenMP Fortran code, and also Intel's translator can work
- 24: AMD offers a dedicated, Clang-based compiler for using OpenMP on AMD GPUs: AOMP; it supports both C/C++ (Clang) and Fortran (Flang, [example](#))
- 25: Currently, no (known) way to launch Standard-based parallel algorithms on AMD GPUs
- 26: Kokkos supports AMD GPUs through HIP
- 27: Alpaka supports AMD GPUs through HIP
- 28: AMD does not officially support GPU programming with Python (also not semi-officially like NVIDIA), but third-party support is available, for example through [Numba](#) or a [HIP version of CuPy](#)
- 29: [SYCLomatic](#) translates CUDA code to SYCL code, allowing it to run on Intel GPUs; also, Intel's [DPC++ Compatability Tool](#) can transform CUDA to SYCL
- 30: No direct support, only via ISO C bindings, but at least an example can be [found on GitHub](#); it's pretty scarce and not by Intel itself, though
- 31: [CHIP-SPV](#) supports mapping CUDA and HIP to OpenCL and Intel's Level Zero, making it run on Intel GPUs
- 32: SYCL is the prime programming model for Intel GPUs; actually, SYCL is only a standard, while Intel's implementation of it is called DPC++ (*Data Parallel C++*), which extends the SYCL standard in places
- 33: OpenACC can be used on Intel GPUs by translating the code to OpenMP with [Intel's Source-to-Source translator](#)
- 34: Intel has [extensive support for OpenMP](#) through their latest compilers
- 35: Currently, no (known) way to launch Standard-based parallel algorithms on Intel GPUs
- 36: With [Intel oneAPI 2022.3](#), Intel supports DO CONCURRENT with GPU offloading
- 37: Kokkos supports Intel GPUs through SYCL
- 38: [Alpaka v0.9.0](#) introduces experimental SYCL support
- 39: Not a lot of support available at the moment, but notably [DPNP](#), a SYCL-based drop-in replacement for Numpy