Intel® oneAPI Toolkits Installation Guide for Windows*

Contents

Intel® oneAPI Toolkits and Components Installation Guide for Windows*

1

This guide covers the installation of Intel® oneAPI toolkits and standalone components on Windows*.

Before You Begin

Check the Prerequisites information before installing the Intel oneAPI packages.

Installation

Install Intel oneAPI component and toolkit packages with one of the following options:

- Install with GUI
- Install with Command Line
- Install Using Package Managers

Next Steps

Get Intel oneAPI code samples and refer to the toolkit Get Started page for detailed usage instructions, examples, and more:

- Get Started with Intel® oneAPI Base Toolkit
- Get Started with Intel® HPC Toolkit
- Get Started with Intel® Rendering Toolkit

Prerequisites

Consider the following important information before installing the Intel oneAPI packages.

Check System Requirements

Refer to one of the following documents specific for your toolkit to learn more about compatibility details:

- Intel® oneAPI Base Toolkit Release Notes | System Requirements
- Intel® oneAPI System Bring-up Toolkit Release Notes | System Requirements
- Intel® Rendering Toolkit Release Notes | System Requirements

Install Microsoft* Visual Studio* C++ Workload

Before enabling integration into Microsoft* Visual Studio* via the oneAPI toolkit installer, install the **Desktop development with C++** workload into each Visual Studio instance.

For the list of the supported Visual Studio versions, refer to the toolkit System Requirements.

Set Up Your System for Intel GPU

If you are using Intel GPU, you need to install the GPU drivers and Adjust timeout detection and recovery setting.

Install Intel GPU Drivers

If you use Intel GPU, you need to install the OpenCL and Level 0 graphics drivers. Follow the instructions applicable for your device:

- Intel® Arc™ Graphics, 11th-13th Gen Intel® Core™ Processor Graphics.
- Xe Dedicated, 6th-10th Gen Intel® Core™ Processor Graphics, and related Intel Atom®, Pentium®, and Celeron® processors. Driver version varies depending on the Intel Graphics in the system.
- Intel® Data Center GPU Flex Series (ATS-M). Contact your original equipment manufacturer (OEM) representative for access to the Intel Registration Center.

GPU: Adjust Timeout Detection and Recovery Setting

Timeout Detection and Recovery (TDR) is a Windows* OS feature that detects and resets graphics cards with response problems. This feature freezes and reboots operating systems.

Windows resets GPUs if the compute kernels run longer than a few seconds. To avoid a device reset during long running kernels, increase the value of **TDRDelay** setting:

1. Open the **Run** dialog box by pressing **Windows** + **R** keys on your keyboard.
2. Type `regedit` and click **OK** to open the **Registry Editor**.
3. Navigate to the following path by expanding the folders on the left pane: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\GraphicsDrivers`
4.
   - If the `TdrDelay` key exists, double-click it and increase the value in the **Value** data field (such as 20 seconds or higher). Continue to Step 7.
   - If the `TdrDelay` key does not exist, click on an empty space on the right pane and select **New** > **DWORD (32-bit) Value**. Continue to Step 5.
5. Name the key `TdrDelay`.
6. Double-click the key and enter a desired value in the **Value** data field (such as 20 seconds or higher).
7. Click **OK** to save the changes.
8. Close the **Registry Editor** and restart your computer for the changes to take effect.

For more information, refer to TDRDelay documentation.

Installation

Toolkit Installation

> **Important** Some domain-specific toolkits require you to install the Intel oneAPI Base Toolkit first for full functionality.

Use one of the following options to get and install a toolkit package:

Toolkit Name	Binary Installer	Package Manager
Intel® oneAPI Base Toolkit	Online/offline installer	N/A

Toolkit Name	Binary Installer	Package Manager
Intel® HPC Toolkit (requires installation of the Intel oneAPI Base Toolkit)	Online/offline installer	N/A
Intel® Rendering Toolkit	Online/offline installer	N/A

Each of the binary installer types operates in various modes, which are covered later in this section.

Important For FPGA support in the online/offline installer, ensure that you choose to customize your installation and select the **FPGA Support Package for the Intel® oneAPI DCP++/C++ Compiler** component as part of your installation.

NOTE When using the offline installer, you can enable full offline mode by setting the environment variable `INTEL_SUPPRESS_INTERNET_CONNECTION=1`. When this mode is enabled:

- Installer does not send installation statistics
- Download-only mode is disabled
- Upgrade mode is disabled

Component Installation

You can install toolkit components as standalone via binary installer or package managers. Refer to the Single Component Downloads and Runtime Versions resource to locate a component package.

Get 32-bit Libraries

To get access to 32-bit binaries of the Intel® Integrated Performance Primitives (Intel® IPP) and Intel® oneAPI Math Kernel Library (Intel® oneMKL), install the Intel® oneAPI Base Toolkit (32-bit) add-on package from the Intel oneAPI Base Toolkit download page using the process described at Install with GUI.

To use 32-bit libraries from the Intel oneAPI Base Toolkit (32-bit) package, you need to have the same version of the Intel® oneAPI Base Toolkit or standalone libraries installed on your system.

FPGA Development Flow

For instructions on installing the Intel® oneAPI Base Toolkit components required for FPGA development, refer to Install Compiler Components for Altera FPGA Development Flows.

The following FPGA development flows are supported:

- SYCL* HLS Flow

  This flow targets an Intel® FPGA device to generate an RTL IP core that you integrate into your FPGA application design using the Quartus® Prime software.
- FPGA Acceleration Flow

  This flow targets FPGA acceleration boards and requires a third party vendor-provided BSP for deployment of your kernel.

  Depending on the type of FPGA compilation you perform, you might need different software.

Both FPGA development flows require Quartus® Prime software when compiling your design for FPGA simulation or generating an FPGA hardware image. Quartus® Prime software is not required when compiling your design for FPGA emulation or to obtain the FPGA Optimization Report.

For an outline of the steps required to set up your FPGA development environment, refer to Installing the FPGA Development Environment in the *Intel® oneAPI DPC++/C++ Compiler Handbook for Altera FPGAs*.

Install with GUI

Download an installation package using the offline installer option for the toolkit you wish to install from here.

After downloading the toolkit installation package, follow the steps below to install it with GUI.

1. Double-click the `*.exe` file to launch the GUI installer: `w_[Toolkit Name]Kit_[version].exe`
2. Follow the installer instructions.
3. Choose the required Microsoft* Visual Studio* version to enable integration into IDE (if supported) from the corresponding dialog. You need to have the **Desktop development with C++** workload installed into each Visual Studio instance beforehand.
4. Once the installation is complete, verify that your toolkit is installed to the correct installation directory `C:\Program Files (x86)\Intel\oneAPI`.

> **NOTE** By default, the installer GUI works via the OpenGL library. Alternatively, you can force running the installer GUI with the ANGLE library. To do this, set the `INTEL_USE_ANGLE` environment variable to 1.

Install with Command Line

1. Go to the directory where the installer is located.
2. Run the following command:

```
w_[Toolkit Name]Kit_[version]_offline.exe -s -a --silent --eula accept
```

   In silent mode, integration into Visual Studio* is installed by default (if supported). You need to have the **Desktop development with C++** workload installed into each Visual Studio instance beforehand. To skip Visual Studio* integration, pass the following arguments to the installation command: `–p=NEED_VSXXXX_INTEGRATION=0`, where `XXXX` is the Visual Studio version. For example, to install a toolkit and skip integration into Visual Studio* 2019, use the following installation command:

```
w_[Toolkit Name]Kit_[version]_offline.exe -s -a --silent --eula accept -
p=NEED_VS2019_INTEGRATION=0
```

   For the full list of supported installer options, refer to the Command Line Options section.
3. Once the installation is complete, verify that the toolkit is installed in the default directory: `C:\Program Files(x86)\Intel\oneAPI`.
4. If you are using GPU, you need to install the Intel GPU drivers separately.

Command Line Options

Options for Package Extraction Script

`-h, --help`	Show help for the package extraction script.
`-f, --extract-folder`	Point to the folder where the package content will be saved.
`-x, --extract-only`	Unpack the installation package without launching the installer.

`-r, --remove-extracted-files <yes|no>`	Remove extracted files after installation. This action cleans up the temporary package file location.
`-l, --log <log file>`	Log all package extraction actions to the specified file.
`-a <arguments>`	Pass arguments to the installer.

Arguments for Installer

Specify these arguments after the `-a` option.

Arguments for Installer

Option	Default value (if option is not passed)	Description
`-s, --silent`	N/A	Run the installer in non-interactive (silent) mode.
`--eula`	`decline`	Required. Accept or decline End User License Agreement (EULA), supported values: `accept` or `decline` (default).
`--action`	`install`	Specify one of the supported values below when the installer action is needed: • `install` (default) Install the product. Use the `--components` option to specify the list of components to be installed. If not specified, the default set of components is installed. • `remove` Uninstall the product. • `modify` Change the current set of components installed. List all the components you need using the `--components` option. Components that are already installed still must be in the list if remain relevant. • `downloadonly` Download an offline installation package without installing it. To customize the list of components to be included into a package, use the `--components` option. • `repair` Repair the currently installed product.
`--instance`	`default`	Specify an ID of an installation instance. For example: `w_[Toolkit Name]Kit_[version]_offline.exe -a --instance=<instance ID>`. This option enables side-by-side installation of oneAPI products. Each instance is a separate installation entity with its own isolated environment. Product installed in one instance is not visible in another instance. If omitted, installation is performed in default instance. To get the list of available instances, use the `--list-instances` option.

Option	Default value (if option is not passed)	Description
`--config`	N/A	Point to the configuration INI file with options. You can use this file as an alternative to passing options via the command line; mixed approach is also supported. Sample content of a configuration file: `s=eula=accept`.
`--components`	`default`	Specify components to perform an action on, supported values: `all`, `default`, custom components split by ':'. If you need the default components and some extra component(s), combine `default` with the name of the extra component(s) separated by ':'. For example: `--components default:<component_name>`.
`--list-products`	N/A	Get the list of downloaded products, their IDs, versions and statuses (installed/not installed). Use together with the `--instance` option to get the list of available products in a specific instance. For example: `w_[Toolkit Name]Kit_[version]_offline.exe -a --list-products --instance=<instance ID>`.
`--product-id`	N/A	Specify an ID of a product to perform an action on. Use this option with `--list-components` or `--action {install|remove|modify|repair}`.
`--product-ver`	N/A	Specify a product version to perform an action on. Use this option with `--list-components` or `--action {install|remove|modify|repair}`.
`--list-components`	N/A	Get the list of available components of the current package or of a product specified with `--product-id`. Use together with the `--instance` option to get the list of available components in a specific instance. For example: `w_[Toolkit Name]Kit_[version]_offline.exe -a --list-components --instance=<instance ID>`.
`--package-path`	N/A	Specify the directory of the package to install.
`--install-dir`	default installation directory	Customize the installation directory.
`--log-dir`	default log directory	Customize the directory to save the log file to.
`--proxy`	N/A	Specify proxy settings in the following format: `http://username:password@proxy-server.mycorp.com:3128`.
`--download-cache`	default download cache location	Point to the directory to store all downloaded and cached files.

Option	Default value (if option is not passed)	Description
`--download-dir`	default download directory	Customize the download directory, which is used in download-only mode.
`--intel-sw-improvement-program-consent`	`decline`	Accept or decline participation in Intel Software Improvement Program, supported values: `accept` or `decline` (default). To get the program description, use the `--show-intel-sw-improvement-program-consent` command.
`--show-intel-sw-improvement-program-consent`	N/A	Show the detailed description of the Intel Software Improvement Program.
`--ignore-errors`	N/A	Complete installation even if non-critical errors occur (like errors in pre-/post-install scripts).
`-h, --help`	N/A	Show the installer help.
`-p, --property`	N/A	Pass additional custom options. For example, the string `-p=option1=value -p option2=value` gives two additional options.

Command Line Options Usage Examples

- Display the list of already installed products and products included in the downloaded package:

```
w_[Toolkit Name]Kit_[version]_offline.exe -s -a --list-products
```

  Example of output:

```
ID Version Language Installed Name
================================================================================================
intel.oneapi.win.tbb.product 2021.1.1-129 false Intel® oneAPI Threading Building Blocks
```

- Display the list of components in product of current package:

```
w_[Toolkit Name]Kit_[version]_offline.exe -s -a --list-components
```

- Display the list of components of any installed product on the system:

```
w_[Toolkit Name]Kit_[version]_offline.exe -s -a --list-components --product-id
intel.oneapi.win.tbb.product --product-ver 2021.1.1-129
```

  Example of output:

```
ID Version Language Installed Name
================================================================================================
intel.oneapi.win.tbb.devel 2021.1.1-129 Intel® oneAPI Threading Building Blocks
```

- Install specific Intel oneAPI Toolkit products and components:

```
w_[Toolkit Name]Kit_[version]_offline.exe -s -a --silent --eula accept --components
intel.oneapi.win.tbb.devel
```

Install Using Package Managers

You can install Intel oneAPI packages from one of these repositories:

Conda

This page provides general instructions on installing the Intel® oneAPI component packages via the Conda* package manager.

For additional installation notes, refer to the Conda documentation.

To install a package, execute the following command:

- To install the latest version available:

```
conda install -c https://software.repos.intel.com/python/conda/ -c conda-forge <package_name>
```

  To get your package name, refer to the list of packages in the table below.
- To install a specific version:

```
conda install -c https://software.repos.intel.com/python/conda/ -c conda-forge
<package_name>==<version>
```

  For example: `conda install -c https://software.repos.intel.com/python/conda/ -c conda-forge mkl==2024.2.1`

List of Available Packages

Component Name	Package Name	Platform	Dependencies
Intel® MPI Library	`impi_rt` `impi-devel`	`win-x64`	N/A
Intel® Fortran Compiler and Intel® Fortran Compiler Classic	`intel-fortran-rt`	`win-x64` `win-x86`	Intel OpenMP* Runtime Library
Intel® CPU Runtime for OpenCL™ Applications	`intel-opencl-rt`	`win-x64`	oneTBB
Intel® oneAPI DPC++/C++ Compiler	`dpcpp-cpp-rt` `dpcpp_win-64`	`win-x64`	Intel® CPU Runtime for OpenCL™ Applications Intel OpenMP* Runtime Library
	`intel-sycl-rt`	`win-x64`	N/A
Intel OpenMP* Runtime Library	`intel-openmp`	`win-x64` `win-x86`	N/A
Intel® oneAPI DPC++ Library	`onedpl-devel`	`win-x64`	N/A

Component Name	Package Name	Platform	Dependencies
Intel® Graphics Offline Compiler for OpenCL™ Code (OCLOC)	ocloc	win-x64	N/A
Intel® oneAPI Threading Building Blocks (oneTBB)	tbb tbb-devel	win-x64 win-x86	N/A
	tbb4py	win-x64	N/A
Intel® oneAPI Data Analytics Library (oneDAL)	dal dal-static dal-devel dal-include	win-x64	oneTBB
	daal4py	win-x64	oneDAL
Intel® Extension for Scikit-learn*	scikit-learn-intelex	win-x64	oneDAL
Intel® Integrated Performance Primitives (Intel® IPP)	ipp ipp-static ipp-include ipp-devel	win-x64 win-x86	N/A
Intel® Integrated Performance Primitives Cryptography	ipp_crypto ipp_crypto_static ipp_crypto-include ipp_crypto-devel	win-x64 win-x86	N/A
Intel® oneAPI Math Kernel Library (oneMKL)	mkl mkl-devel mkl-static mkl-include	win-x64 win-x86	Intel OpenMP* Runtime Library oneTBB
	mkl-dpcpp mkl-devel-dpcpp onemkl-sycl-blas onemkl-sycl-lapack onemkl-sycl-dft	win-x64	Intel® oneAPI DPC++/C++ Compiler Runtime Intel® CPU Runtime for OpenCL™ Applications

Component Name	Package Name	Platform	Dependencies
	onemkl-sycl-sparse onemkl-sycl-vm onemkl-sycl-rng onemkl-sycl-stats onemkl-sycl-datafitting		
Intel® oneAPI Deep Neural Network Library (oneDNN)	onednn onednn-devel	win-x64	Intel® CPU Runtime for OpenCL™ Applications Intel® oneAPI DPC++/C++ Compiler Runtime
Thread Composability Manager (TCM)	tcm	win-x64 win-x86	N/A

PIP

This page provides general instructions on installing the Intel® oneAPI component packages from the Python* Package Index (PyPI).

For additional installation notes, refer to the PyPI documentation.

To install a package, execute the following command:

- To install the latest version available:

```
pip install <package_name>
```

  To get your package name, refer to the list of packages in the table below.
- To install a specific version:

```
pip install -c intel <package_name>==<version>
```

  For example: `pip install mkl==2021.1.1`

> **Important** For Intel® oneAPI Deep Neural Network Library (oneDNN), only packages of identical configuration can be installed into one environment. For example, you can install onednn-devel-cpu-vcomp with onednn-cpu-vcomp, but should avoid installing it with packages of other configurations, like cpu-iomp, cpu-tbb, cpu-dpcpp-gpu-dpcpp.

List of Available Packages

Component Name	Package Name	Platform	Dependencies
Intel® MPI Library	impi_rt	win-x64	N/A

Component Name	Package Name	Platform	Dependencies
	impi-devel		
Intel® Fortran Compiler and Intel® Fortran Compiler Classic	intel-fortran-rt	win-x64 win-x86	Intel® MPI Library Intel OpenMP* Runtime Library
Intel® CPU Runtime for OpenCL™ Applications	intel-opencl-rt	win-x64	oneTBB
Intel® oneAPI DPC++/C++ Compiler	dpcpp-cpp-rt	win-x64	Intel® CPU Runtime for OpenCL™ Applications Intel OpenMP* Runtime Library
	intel-sycl-rt	win-x64	N/A
Intel OpenMP* Runtime Library	intel-openmp	win-x64 win-x86	N/A
Intel® oneAPI Threading Building Blocks (oneTBB)	tbb tbb-devel	win-x64 win-x86	N/A
	tbb4py	win-x64	N/A
Intel® oneAPI Data Analytics Library (oneDAL)	daal daal-static daal-devel daal-include	win-x64	oneTBB
	daal4py	win-x64	oneDAL
Intel® Extension for Scikit-learn	scikit-learn-intelex	win-x64	oneDAL
Intel® Integrated Performance Primitives (Intel® IPP)	ipp ipp-static ipp-include ipp-devel	win-x64 win-x86	N/A

Component Name	Package Name	Platform	Dependencies
Intel® Integrated Performance Primitives Cryptography	ipp_crypto ipp_crypto_static ipp_crypto-include ipp_crypto-devel	win-x64 win-x86	N/A
Intel® oneAPI Math Kernel Library (oneMKL)	mkl mkl-devel mkl-static mkl-include	win-x64 win-x86	Intel OpenMP* Runtime Library oneTBB
	mkl-dpcpp mkl-devel-dpcpp onemkl-sycl-blas onemkl-sycl-lapack onemkl-sycl-dft onemkl-sycl-sparse onemkl-sycl-vm onemkl-sycl-rng onemkl-sycl-stats onemkl-sycl-datafitting	win-x64	Intel® oneAPI DPC++/C++ Compiler Runtime Intel® CPU Runtime for OpenCL™ Applications
Intel® oneAPI Deep Neural Network Library (oneDNN)	onednn onednn-devel	win-x64	Intel® CPU Runtime for OpenCL™ Applications Intel® oneAPI DPC++/C++ Compiler Runtime
Thread Composability Manager (TCM)	tcm	win-x64 win-x86	N/A

NuGet

This page provides general notes on how to install Intel® oneAPI components distributed via the NuGet channel. NuGet is a Microsoft-supported mechanism for sharing compiled code. It also defines how the packages are created, hosted and consumed, and it provides the tools for each of those roles. For more details on the installation process, please refer to the Microsoft* documentation.

Intel® oneAPI components distributed via NuGet include both development and runtime options.

For your convenience, the components are divided to *devel* and *static* packages corresponding to the different linking types (dynamic and static). Certain component packages are also split into x64 and x86 versions to reduce the overall package size.

Development Packages

The following table provides the full list of available packages:

Component Name	Package Name	Platform	Dependencies
Intel® MPI Library	`intelmpi.devel.<platform>`	`win-x64`	N/A
Intel OpenMP* Runtime Library	`intelopenmp.devel.<platform>`	`win`	N/A
Intel® oneAPI Threading Building Blocks (oneTBB)	`inteltbb.devel.<platform>`	`win`	N/A
Intel® oneAPI Data Analytics Library (oneDAL)	`inteldal.devel.<platform>`	`win-x64`	oneTBB
	`inteldal.static.<platform>`		
Intel® Integrated Performance Primitives (Intel® IPP)	`intelipp.devel.<platform>`	`win-x64`	N/A
	`intelipp.static.<platform>`	`win-x86`	
Intel® Integrated Performance Primitives Cryptography	`intelipp_crypto.devel.<platform>`	`win-x64`	N/A
	`intelipp_crypto.static.<platform>`	`win-x86`	
Intel® oneAPI Math Kernel Library (oneMKL)	`intelmkl.devel.<platform>`	`win-x64`	Intel® MPI Library Intel OpenMP* Runtime Library
	`intelmkl.static.<platform>`	`win-x86`	
Intel® oneAPI Math Kernel Library (Cluster Components)	`intelmkl.devel.cluster.<platform>`	`win-x64`	Intel OpenMP* Runtime Library oneMKL
	`intelmkl.static.cluster.<platform>`		

All the specified dependencies will be downloaded automatically by the NuGet Package Manager.

The devel packages on Windows* also have a dependency on the runtime redist packages (see the next section).

Runtime Packages

The runtime packages are runtime redistributable libraries that will automatically load optimizations specific to your Intel hardware (including, but not limited to, vectorization). They can be used by another NuGet package that depends on these runtimes.

Component Name	Package Name	Platform Availability
Intel® MPI Library	`intelmpi.redist.<platform>`	`win-x64`
Intel OpenMP* Runtime Library	`intelopenmp.redist.<platform>`	`win`
Intel® oneAPI Threading Building Blocks (oneTBB)	`inteltbb.redist.<platform>`	`win`
Intel® oneAPI Data Analytics Library (oneDAL)	`inteldal.redist.<platform>`	`win-x64`
Intel® Integrated Performance Primitives (Intel® IPP)	`intelipp.redist.<platform>`	`win-x64` `win-x86`
Intel® Integrated Performance Primitives Cryptography	`intelipp_crypto.redist.<platform>`	`win-x64` `win-x86`
Intel® oneAPI Math Kernel Library (oneMKL)	`intelmkl.redist.<platform>`	`win-x64`
Intel® oneAPI Math Kernel Library (Cluster Components)	`intelmkl.redist.cluster.<platform>`	`win-x64`

Install Compiler Components for Altera FPGA Development Flows

To use the Intel® oneAPI DPC++/C++ Compiler for development flows that target Altera FPGA devices, install the **FPGA Support Package for the Intel® oneAPI DPC++/C++ Compiler** (often referred to as the "FPGA Support Package"). This package adds FPGA-specific support to the Intel® oneAPI DPC++/C++ Compiler to enable FPGA code development.

At a high level, the steps to install the compiler components for FPGA development are as follows:

1. Obtain Compiler Components for FPGA Development.
2. Install compiler components in one of the following ways:
   - Install Compiler Components with GUI.
   - Install Compiler Components with Command Line.
3. (optional) Additional Software Requirements for FPGA Development.

Obtain Compiler Components for FPGA Development

Installation Type	Installation Packages Required
Online installation	• Intel® oneAPI Base Toolkit online installer
Offline installation	• Intel® oneAPI Base Toolkit offline installer • FPGA Support Package for the Intel® oneAPI DCP ++/C++ Compiler* offline installer

Installation Type	Installation Packages Required
	Download the FPGA Support Package offline installer from the Single Component Downloads and Runtime Versions page.

Install Compiler Components with GUI

Online Installation

1. Start the GUI installer according to the instructions in Install with GUI.
2. In the online installer GUI, customize your installation and select the optional **FPGA Support Package for the Intel® oneAPI DCP++/C++ Compiler** component as follows:
   a. Click **Customize**.
   b. Expand **Intel® oneAPI DCP++/C++ Compiler**.
   c. Under **Optional**, select **FPGA Support Package for the Intel® oneAPI DCP++/C++ Compiler**.

Offline Installation

1. Run the GUI installer for the Intel® oneAPI Base Toolkit according to the instructions in Install with GUI.
2. Run the GUI installer for the FPGA Support Package for the Intel® oneAPI Base Toolkit according to the instructions in Install with GUI.

Install Compiler Components with Command Line

Online Installation

Include the following option when you run the installation command (refer to Install with Command Line for details):

```
--components default:intel.oneapi.win.compiler.fpga
```

For example, to silently install the Intel oneAPI Base Toolkit and FPGA support package, run the following command:

```
w_BaseKit_[version]_p_offline.exe -a --silent --eula accept --components
default:intel.oneapi.win.compiler.fpga
```

Offline Installation

For an offline installation, run separate commands to install the Intel® oneAPI Base Toolkit and the FPGA Support Package for the oneAPI DCP++/C++ Compiler.

For example, to silently install the Intel oneAPI Base Toolkit and FPGA Support Package, run the following commands:

```
w_BaseKit_p_[version]_offline.exe -a --silent --eula accept
w_compiler-fpga-addon_p_[version]_offline.exe -a --silent --eula accept
```

Additional Software Requirements for FPGA Development

- Install Quartus® Prime Software.

  The Quartus® Prime software is required only when compiling your design for FPGA simulation or generating an FPGA hardware image. Quartus® Prime software is **not** required when compiling your design for FPGA emulation or to obtain the FPGA Optimization Report.
- Install FPGA Board Support Packages

Board support packages are required only when compiling to FPGA hardware as part of the FPGA acceleration flow. BSPs are not required for the SYCL* HLS flow.

Installation Topology Variations

When setting up your system, you can install the required components in a variety of topologies:

- **Set up a single system**

  In this method, you can use a single system acting as both the runtime and development system. Install the Intel oneAPI Base Toolkit, FPGA Support Package for the Intel® oneAPI DCP++/C++ Compiler, Intel® Quartus® Prime Software, and FPGA board support package on the same system.
- **Set up separate development and runtime systems**

  In this method, you install the custom platform/FPGA device on the runtime system and run only the design. On the development system, install the Intel® Quartus® Prime software to compile and generate the FPGA bitstream. Refer to Intel® Quartus® Prime Software, Install Intel® FPGA Board Packages sections for more information.
- **Set up a cloud on-premise**

  A cloud on-premise helps reduce the hardware cost necessary for development. In this workflow, you can set up two development systems, one for the FPGA development and the other for the Intel® Quartus® Prime software compilation. The runtime system can be different. After setting up your development systems, install the physical card on the runtime system. Refer to the Intel® oneAPI DPC++ /C++ Library System Requirements for FPGA requirements.

  - On the first development system with lower configurations (8 GB RAM), iterate over your designs using the emulation and report flow to verify code correctness. Install the Intel® oneAPI Base Toolkit and FPGA Support Package for the Intel® oneAPI DCP++/C++ Compiler packages on this system. For more information about emulation and report flow, refer to Types of SYCL* FPGA Compilation.
  - On the second development system with higher configurations (memory requirements dictated by the FPGA acceleration board or FPGA development kit that you use), install the Intel® Quartus® Prime Software. Perform Intel® Quartus® Prime compilation using either the hardware flow or the device link flow. For more information, refer to Types of SYCL* FPGA Compilation in the *Intel oneAPI DPC++/C++ Compiler Handbook for Altera FPGAs*.

Install Quartus® Prime Software

Quartus® Prime software includes everything you need to design for Intel® FPGAs, from design entry and synthesis to optimization, verification, and simulation. It contains the following features:

- Hybrid Placer & Global Router
- Timing Analyzer
- Physical Synthesis
- Incremental Fitter Optimization
- Interface Planner
- Synthesis Engine
- Platform Designer
- Partial Reconfiguration
- Block-Based (Hierarchical) Design

NOTE The Quartus® Prime software is not required for the FPGA development flow emulation or report generation stages. You can complete those stages with only the Intel® oneAPI DPC++/C++ Compiler (part of the Intel oneAPI Base Toolkit) and the FPGA Compiler Support Package for the Intel® oenAPI DPC++/C++ Compiler.

If you want to use the Quartus® Prime software (required for FPGA hardware and simulation flow) with the Intel® oneAPI DPC++/C++ Compiler, the edition and version of Quartus® Prime software that you need depends on your target device (either standalone device or the device on an acceleration board). Refer to the Intel® oneAPI DPC++ /C++ Library System Requirements for more information.

- If you want to install a version of the Quartus® Prime software, follow the instructions in the following documents:
  - Intel® FPGA Software Installation and Licensing
  - Quartus® Prime Pro Edition User Guide: Getting Started
  - Quartus® Prime Standard Edition User Guide: Getting Started
- If you already have a version of the Quartus® Prime software installed on your system and you want to use that version, then use one of the following methods to set up the environment:
  - Set `QUARTUS_ROOTDIR_OVERRIDE = <path_to_your_quartus_folder>`
  - Add the `bin` directory of the Quartus® Prime software to your `PATH` variable.
- If you have multiple versions of the Quartus® Prime software installed, Altera recommends setting the `QUARTUS_ROOTDIR_OVERRIDE` variable to point to the Quartus software path you want to use. Otherwise, you might end up using a version different than the one you expected. Ensure that you set the `QUARTUS_ROOTDIR_OVERRIDE` variable after running the `setvars` script, which can potentially override your setting.

Install FPGA Board Support Packages

To compile an executable that can run on an FPGA board, install a Board Support Package (BSP) that allows targeting compiles to that board. Intel does not ship BSPs with oneAPI. You must download and install BSPs from a third-party vendor. For more information, refer to the Intel® FPGA development flow page.

To use a third-party vendor-provided BSP with the Intel® oneAPI Base Toolkit, follow these instructions:

1. Follow vendor-specific instructions to download and install the board package.
2. Install a version of the Quartus® Prime software that is compatible with the BSP (as indicated by the vendor).
3. Follow instructions in Install the Quartus® Prime Software to ensure the Intel oneAPI DPC++/C++ Compiler is configured to run with the installed Quartus® Prime software.

Related Links

- Intel® oneAPI DPC++/C++ Compiler System Requirements
- Get Started with the Intel® oneAPI Base Toolkit for Windows
- Intel® oneAPI DPC++/C++ Compiler Handbook for Intel FPGAs
- Intel oneAPI FPGA Development Flow in the Intel® oneAPI DPC++/C++ Compiler Handbook for Altera FPGAs
- Intel® FPGA SDK for OpenCL™ Pro Edition: Custom Platform Toolkit User Guide
- Quartus® Prime Software User Guides

Uninstall oneAPI Toolkits and Components

Uninstall Using GUI

Use the **Add/Remove Programs** option from the Control Panel to uninstall oneAPI Toolkits.

Uninstall with Command Line

1. Display the list of the already installed products and products included in the downloadable package using the following command:

```
w_[Toolkit Name]Kit_[version].exe -s -a --list-products
```

2. Uninstall the selected product:

```
cd C:\Program Files\Intel\oneAPI\Installer
installer.exe -s --action remove --product-id intel.oneapi.win.tbb.product --product-ver
2021.1.1-129
```

Troubleshooting

Integration into Visual Studio* fails for oneAPI 2022.1 and lower versions of toolkits

During installation of 2022.1 and lower versions of toolkits, integration into Visual Studio* fails on systems with Visual Studio* 2022 installed.

Cause: Incompatibility between oneAPI installers and recent changes in the `Microsoft.VisualStudio.Setup.Configuration.Native.dll` library provided by Microsoft as part of the Visual Studio 2022 installation.

Solution: Upgrade to the latest version of the oneAPI toolkit. Alternatively, before installing 2022.1 and lower toolkit packages, remove Visual Studio* 2022 from your system. For more information about the available workarounds, refer to Known Microsoft* Visual Studio 2022 and oneAPI Toolkits Installation Issue.

OpenCL CPU Runtime Library fails to load Intel oneTBB

Intel OpenCL CPU Runtime Library fails to load the Intel oneTBB library that was installed via Conda.

Cause: No oneTBB registry key installed when Intel oneTBB is installed via Conda package manager.

Solution: Before completing the workaround steps below, make sure that you already configured the targeted device, otherwise, OpenCL Runtime will load two devices, and use the first one by default.

1. Locate the corresponding device configuration file in the same folder where the OpenCL Runtime Library file `intelocl64.dll` resides:
   - `cl.cfg` for CPU device
   - `cl.fpga_emu.cfg` for FPGA emulator
2. Open the configuration file for editing and add your oneTBB DLL path to the `CL_CONFIG_TBB_DLL_PATH` field:

```
CL_CONFIG_TBB_DLL_PATH = <tbb-install-dir>
```

NOTE

- If you configure both `cl.cfg` and `cl.fpga_emu.cfg`, the oneTBB DLL path in them should be the same value
- Add the value of the `CL_CONFIG_TBB_DLL_PATH` field without quotation marks

Integrity check fails

During installation, you may get an error message about failed integrity check of downloaded files.

Cause: Downloaded files are corrupted because of the hard drive corruption.

Solution: Check hard drive health and restart the installation.

2021.x installation overwrites existing 2022.x installer

If you launch the installer for the 2021.x version of a toolkit on a system with the 2022.x version installed, your existing 2022.x installer will be downgraded to the 2021.x version. The 2021.x installer does not recognize installed 2022.x packages.

Cause: Compatibility issue between 2022.x and 2021.x versions of the installer.

Solution: Restore the latest installer by launching the installer for the 2022.x version of the toolkit. In future, when you need to install 2021.x on a system with 2022.x installed, before launching the installer, back up the following installer directories by renaming them:

- 32-bit system: `C:\Program Files\Intel\oneAPI\Installer` and `C:\Program Files\Intel\PackageManager\1.0`
- 64-bit system: `C:\Program Files (x86)\Intel\oneAPI\Installer` and `C:\Program Files (x86)\Intel\PackageManager\1.0`

When you are done with 2021.x, you can change back the directory names to restore the 2022.x installer.

Notices and Disclaimers

Intel technologies may require enabled hardware, software or service activation.

No product or component can be absolutely secure.

Your costs and results may vary.