DPDK

DATA PLANE DEVELOPMENT KIT

# Getting Started Guide for Windows

***Release 19.11.14***

**Dec 15, 2022**

CHAPTER

ONE

# INTRODUCTION

This document contains instructions for installing and configuring the Data Plane Development Kit (DPDK) software. The document describes how to compile and run a DPDK application in a Windows* OS application environment, without going deeply into detail.

*Other names and brands may be claimed as the property of others.

CHAPTER

TWO

# LIMITATIONS

DPDK for Windows is currently a work in progress. Not all DPDK source files compile. Support is being added in pieces so as to limit the overall scope of any individual patch series. The goal is to be able to run any DPDK application natively on Windows.

CHAPTER

THREE

# COMPILING THE DPDK TARGET FROM SOURCE

## 3.1 System Requirements

Building the DPDK and its applications requires one of the following environments:

- The Clang-LLVM C compiler and Microsoft MSVC linker.
- The MinGW-w64 toolchain (either native or cross).

The Meson Build system is used to prepare the sources for compilation with the Ninja backend. The installation of these tools is covered in this section.

## 3.2 Option 1. Clang-LLVM C Compiler and Microsoft MSVC Linker

### 3.2.1 Install the Compiler

Download and install the clang compiler from LLVM website. For example, Clang-LLVM direct download link:

```
http://releases.llvm.org/7.0.1/LLVM-7.0.1-win64.exe
```

### 3.2.2 Install the Linker

Download and install the Build Tools for Visual Studio to link and build the files on windows, from Microsoft website. When installing build tools, select the "Visual C++ build tools" option and ensure the Windows SDK is selected.

## 3.3 Option 2. MinGW-w64 Toolchain

Obtain the latest version from MinGW-w64 website. On Windows, install to a folder without spaces in its name, like `C:\MinGW`. This path is assumed for the rest of this guide.

Version 4.0.4 for Ubuntu 16.04 cannot be used due to a MinGW-w64 bug.

## 3.4 Install the Build System

Download and install the build system from Meson website. A good option to choose is the MSI installer for both meson and ninja together:

```
http://mesonbuild.com/Getting-meson.html#installing-meson-and-ninja-with-the-msi-installer%22
```

Recommended version is either Meson 0.47.1 (baseline) or the latest release.

## 3.5 Install the Backend

If using Ninja, download and install the backend from Ninja website or install along with the meson build system.

## 3.6 Build the code

The build environment is setup to build the EAL and the helloworld example by default.

### 3.6.1 Option 1. Native Build on Windows

When using Clang-LLVM, specifying the compiler might be required to complete the meson command:

```
set CC=clang
```

When using MinGW-w64, it is sufficient to have toolchain executables in PATH:

```
set PATH=C:\MinGW\mingw64\bin;%PATH%
```

To compile the examples, the flag `-Dexamples` is required.

```
cd C:\Users\me\dpdk
meson -Dexamples=helloworld build
ninja -C build
```

CHAPTER

FOUR

# RUN THE HELLOWORLD EXAMPLE

Navigate to the examples in the build directory and run *dpdk-helloworld.exe*.

```
cd C:\Users\me\dpdk\build\examples
dpdk-helloworld.exe
hello from core 1
hello from core 3
hello from core 0
hello from core 2
```

Note for MinGW-w64: applications are linked to `libwinpthread-1.dll` by default. To run the example, either add toolchain executables directory to the PATH or copy the library to the working directory. Alternatively, static linking may be used (mind the LGPLv2.1 license).