



Denglin Compute Compiler V2

User Guide

DL-DG/SW-032D-02

2024-09-05

Copyright©苏州登临科技有限公司，2019 - 2025，版权所有。

未经苏州登临科技有限公司事先书面同意，不得以任何形式或方式复制或传播本文件的任何部分。

商标和许可



和其它苏州登临科技有限公司的其它登临科技的图标为苏州登临科技有限公司的商标。本手册中提及的所有其他商标均为其各自所有者的财产。

通知

所购买的产品、服务和特性由苏州登临科技有限公司与客户签订的合同规定。本文件中描述的所有或部分产品、服务和特性可能不在采购范围或使用范围内。除非合同中另有规定，本文件中的所有声明、信息和建议均按“原样”提供，无任何明示或暗示的保证或陈述。

本手册中的信息如有更改，恕不另行通知。本文件在编制过程中已尽一切努力确保内容的准确性，本文件中的所有声明、信息和建议不构成任何明示或暗示的保证。

苏州登临科技有限公司

苏州工业园区扬富路11号南岸新地一期商务楼5号1101室，江苏，中国

<http://www.denglin.ai>

email : support@denglin.ai

Change History

Version	Change description
02	add windows build linker options.
01	Initial version.

Table of Contents

[Table of Contents](#)

[1 Overview](#)

[2 Usage](#)

[2.1 Major Options](#)

[2.2 Other Options](#)

[3 Double Data Type Error Report](#)

[4 Extra Linker Options](#)

1 Overview

dlcc (DengLin Compute Compiler) V2 is Denglin's CUDA & OpenCL compilation tool, based on LLVM/clang.

2 Usage

```
dlcc [options] <inputs>
```

2.1 Major Options

Option	Description
<code>--cuda-gpu-arch=<value></code>	CUDA GPU architecture. May be specified more than once. Value: <code>dlgput64</code> Example: <code>--cuda-gpu-arch=dlgput64</code>
<code>-xllc --maxrregcount=<value></code>	Specify the maximum amount of registers that GPU functions can use. Value range: <code>32</code> , <code>64</code> , <code>128</code> Example: <code>-xllc --maxrregcount=128</code>
<code>-use_fast_math</code>	Allow aggressive, lossy floating-point optimizations.
<code>-D <macro>=<value></code>	Define <code><macro></code> to <code><value></code> (or <code>1</code> if <code><value></code> omitted). Example: <code>-DMACRO=1</code>
<code>-U <macro></code>	Undefine <code><macro></code> .
<code>-std=<value></code>	Language standard to compile for (<code>c++98</code> , <code>c++11</code> , <code>c++14</code>). Value range: <code>c++98</code> , <code>c++11</code> , <code>c++14</code> .
<code>-w</code>	Suppress all warnings.
<code>-W <warning></code>	Enable the specified warning.
<code>--cuda-device-only</code>	Compile CUDA code for device only.
<code>-c</code>	Only run preprocess, compile, and assemble steps.
<code>-o <file></code>	Write output to <code><file></code> .
<code>-I <dir></code>	Add directory to include search path.
<code>-L <dir></code>	Add directory to library search path.
<code>-x cuda</code>	Treat subsequent input files as having cuda file type.
<code>-v</code>	Show commands to run and use verbose output.

2.2 Other Options

Option	Description
<code>-save-temps</code>	Save intermediate compilation results.
<code>-shared</code>	Generate Dynamic link library (hybrid compilation).
<code>-fPIC</code>	Generate Position-Independent-Code in compile stage (hybrid compilation).
<code>-wdouble-promotion</code>	Warning for implicit conversion from float to double.
<code>-Werror</code>	Treat warning as error.

3 Double Data Type Error Report

For the current version, dlcc does not support double data type.

Example error information:

```
error: <unknown>:0:0: in function XXXXXX : unsupported call to function __nv_log2
```

```
error: <unknown>:0:0: in function XXXXXX : unsupported call to function __nv_ceil
```

Note:

A CUDA float built-in function's name ends with `f`, for example, `__nv_log2f` is a float built-in function, and `__nv_log2` is a double built-in function.

User can use option `-wdouble-promotion` to find the codes containing double data type in a file.

4 Extra Linker Options

we must add 2 extra linker options when we build cuda-c/c++ project with dlcc in windows platform.

eg:we can add options in cmake command like this: `-DCMAKE_EXE_LINKER_FLAGS="-OPT:NOICF -OPT:NOREF"`