



lib_logging: Debug Printing

Publication Date: 2025/8/8

Document Number: XM-006383-UG v3.3.2

IN THIS DOCUMENT

| | | |
|---|-------------------|---|
| 1 | Introduction | 2 |
| 2 | API | 2 |
| 3 | Debug units | 2 |
| 4 | Enabling printing | 2 |

1 Introduction

This library provides a lightweight printf function that can be enabled or disabled via configuration defines. Code can be declared to be within a “debug unit” (usually a library or application source base) and prints can be enabled/disabled per debug unit.

lib_logging is intended to be used with the [XCommon CMake](#), the XMOS application build and dependency management system.

2 API

To use this module, include **lib_logging** in the application’s **APP_DEPENDENT_MODULES** list and include the **debug_print.h** header file.

void **debug_printf**(char fmt[], ...)

A limited functionality version of printf that is low memory.

This function works like C-standard printf except that it only accepts d, x, s, u and c format specifiers with no conversions.

The p format specifier is treated the same as a x.

The capital version of each format specifier performs the same as the lower case equivalent.

Any alignment or padding characters are simply ignored.

The function uses the functions from **print.h** to do the underlying printing.

Unlike printf this function has no return value.

Whether the function does any output can be controlled via defines such as **DEBUG_PRINT_ENABLE** or **DEBUG_PRINT_ENABLE_[debug unit name]** in the application’s **debug_conf.h**

3 Debug units

A source file can be added to a debug unit by defining the **DEBUG_UNIT** macro before inclusion of **debug_print.h**. For example:

```
#define DEBUG_UNIT ETHERNET_MODULE
#include "debug_print.h"
```

To include all source files in a module in a particular debug unit, it is convenient to do it in the **lib_build_info.cmake** file of the module e.g.:

```
set(LIB_COMPILER_FLAGS ... -DDEBUG_UNIT=ETHERNET_MODULE ...)
```

If no **DEBUG_UNIT** is defined then the default debug unit is **APPLICATION**.

4 Enabling printing

By default, debug printing is turned *off*. To enable printing you need to pass the correct command line option to compilation. The following defines can be set by using the **-D**

option to the compiler. For example, the following in your application **CMakeLists.txt** will enable debug printing:

```
set(APP_COMPILER_FLAGS ... -DDEBUG_PRINT_ENABLE=1 ...)
```

The following defines can be set:

DEBUG_PRINT_ENABLE

Setting this define to 1 or 0 will control whether debug prints are output.

DEBUG_PRINT_ENABLE_[debug unit]

Enabling this define will cause printing to be enabled for a specific debug unit. If set to 1, this will override the default set by **DEBUG_PRINT_ENABLE**.

DEBUG_PRINT_DISABLE_[debug unit]

Enabling this define will cause printing to be disabled for a specific debug unit. If set to 1, this will override the default set by **DEBUG_PRINT_ENABLE**.



Copyright © 2025, All Rights Reserved.

XMOS Ltd. is the owner or licensee of this design, code, or Information (collectively, the "Information") and is providing it to you "AS IS" with no warranty of any kind, express or implied and shall have no liability in relation to its use. XMOS Ltd makes no representation that the Information, or any particular implementation thereof, is or will be free from any claims of infringement and again, shall have no liability in relation to any such claims.

XMOS, XCORE, VocalFusion and the XMOS logo are registered trademarks of XMOS Ltd. in the United Kingdom and other countries and may not be used without written permission. Company and product names mentioned in this document are the trademarks or registered trademarks of their respective owners.

