# Xilinx Standalone Library Documentation

## *XilFFS Library v4.3*

**UG1032 (v2020.1) June 3, 2020**

XILINX®

# Table of Contents

Chapter 1

# XilFFS Library API Reference

The Xilinx fat file system (FFS) library consists of a file system and a glue layer. This FAT file system can be used with an interface supported in the glue layer. The file system code is open source and is used as it is. Currently, the Glue layer implementation supports the SD/eMMC interface and a RAM based file system. Application should make use of APIs provided in ff.h. These file system APIs access the driver functions through the glue layer.

The file system supports FAT16, FAT32, and exFAT (optional). The APIs are standard file system APIs. For more information, see the http://elm-chan.org/fsw/ff/00index_e.html.

***Note*:** The XilFFS library uses Revision R0.13b of the generic FAT filesystem module.

### Library Files

The table below lists the file system files.

| File | Description |
|---|---|
| ff.c | Implements all the file system APIs |
| ff.h | File system header |
| ffconf.h | File system configuration header – File system configurations such as READ_ONLY, MINIMAL, can be set here. This library uses FF_FS_MINIMIZE and FF_FS_TINY and Read/Write (NOT read only) |

The table below lists the glue layer files.

| File | Description |
|---|---|
| diskio.c | Glue layer – implements the function used by file system to call the driver APIs |
| ff.h | File system header |
| diskio.h | Glue layer header |

## Selecting a File System with an SD Interface

To select a file system with an SD interface:

1. Click **File → New → Platform Project**.

2. Click **Specify** to create a new hardware platform specification.
3. Provide a new name for the domain in the **Project name** field if you wish to override the default value.
4. Select the location for the board support project files. To use the default location, as displayed in the **Location** field, leave the **Use default location** check box selected. Otherwise, deselect the checkbox and then type or browse to the directory location.
5. From the **Hardware Platform** drop-down, choose the appropriate platform for your application or click the **New** button to browse to an existing hardware platform.
6. Select the target CPU from the drop-down list.
7. From the **Board Support Package OS** list box, select the type of board support package to create. A description of the platform types displays in the box below the drop-down list.
8. Click **Finish**. The wizard creates a new software platform and displays it in the Vitis Navigator pane.
9. Select **Project → Build Automatically** to automatically build the board support package. The **Board Support Package Settings** dialog box opens. Here you can customize the settings for the domain.
10. Click **OK** to accept the settings, build the platform, and close the dialog box.
11. From the Explorer, double-click `platform.spr` file and select the appropriate domain/board support package. The **Overview** page opens.
12. In the overview page, click **Modify BSP Settings**.
13. Using the Board Support Package Settings page, you can select the OS version and which of the supported libraries are to be enabled in this domain/BSP.
14. Select the **xilffs** library from the list of **Supported Libraries**.
15. Expand the **Overview** tree and select **xilffs**. The configuration options for xilffs are listed.
16. Configure the xilffs by setting the `fs_interface = 1` to select the `SD/eMMC`. This is the default value. Ensure that the SD/eMMC interface is available, prior to selecting the `fs_interface = 1` option.
17. Build the bsp and the application to use the file system with SD/eMMC. SD or eMMC will be recognized by the low level driver.

# Selecting a RAM Based File System

To select a RAM based file system:

1. Click **File → New → Platform Project**.
2. Click **Specify** to create a new hardware platform specification.

3. Provide a new name for the domain in the **Project name** field if you wish to override the default value.
4. Select the location for the board support project files. To use the default location, as displayed in the **Location** field, leave the **Use default location** check box selected. Otherwise, deselect the checkbox and then type or browse to the directory location.
5. From the **Hardware Platform** drop-down, choose the appropriate platform for your application or click the **New** button to browse to an existing hardware platform.
6. Select the target CPU from the drop-down list.
7. From the **Board Support Package OS** list box, select the type of board support package to create. A description of the platform types displays in the box below the drop-down list.
8. Click **Finish**. The wizard creates a new software platform and displays it in the Vitis Navigator pane.
9. Select **Project → Build Automatically** to automatically build the board support package. The **Board Support Package Settings** dialog box opens. Here you can customize the settings for the domain.
10. Click **OK** to accept the settings, build the platform, and close the dialog box.
11. From the Explorer, double-click `platform.spr` file and select the appropriate domain/board support package. The **Overview** page opens.
12. In the **Overview** page, click **Modify BSP Settings**.
13. Using the Board Support Package Settings page, you can select the OS version and which of the supported libraries are to be enabled in this domain/BSP.
14. Select the **xilffs** library from the list of **Supported Libraries**.
15. Expand the **Overview** tree and select **xilffs**. The configuration options for xilffs are listed.
16. Configure the xilffs by setting the `fs_interface = 2` to select the `RAM`.
17. As this project is used by LWIP based application, select **lwip library** and configure according to your requirements. For more information, see the LwIP Library API Reference documentation.
18. Use any lwip application that requires a RAM based file system - TCP/UDP performance test apps or tftp or webserver examples.
19. Build the bsp and the application to use the RAM based file system.

# Library Parameters in MSS File

XilFFS Library can be integrated with a system using the following code snippet in the Microprocessor Software Specification (MSS) file:

```
BEGIN LIBRARY
    PARAMETER LIBRARY_NAME = xilffs
    PARAMETER LIBRARY_VER = 4.3
    PARAMETER fs_interface = 1
    PARAMETER read_only = false
    PARAMETER use_lfn = 0
    PARAMETER enable_multi_partition = false
    PARAMETER num_logical_vol = 2
    PARAMETER use_mkfs = true
    PARAMETER use_strfunc = 0
    PARAMETER set_fs_rpath = 0
    PARAMETER enable_exfat = false
    PARAMETER word_access = true
    PARAMETER use_chmod = false
END
```

The table below describes the libgen customization parameters.

| Parameter | Default Value | Description |
|---|---|---|
| LIBRARY_NAME | xilffs | Specifies the library name. |
| LIBRARY_VER | 4.3 | Specifies the library version. |
| fs_interface | 1 for SD/eMMC<br>2 for RAM | File system interface. SD/eMMC and RAM based file system are supported. |
| read_only | False | Enables the file system in Read Only mode, if true. Default is false. For Zynq UltraScale+ MPSoC devices, sets this option as true. |
| use_lfn | 0 | Enables the Long File Name(LFN) support if non-zero. 0: Disabled (Default) 1: LFN with static working buffer 2 (on stack) or 3 (on heap): Dynamic working buffer |
| enable_multi_partitio | False | Enables the multi partition support, if true. |
| num_logical_vol | 2 | Number of volumes (logical drives, from 1 to 10) to be used. |
| use_mkfs | True | Enables the mkfs support, if true. For Zynq UltraScale+ MPSoC devices, set this option as false. |

| Parameter | Default Value | Description |
|---|---|---|
| use_strfunc | 0 | Enables the string functions (valid values 0 to 2). Default is 0. |
| set_fs_rpath | 0 | Configures relative path feature (valid values 0 to 2). Default is 0. |
| ramfs_size | 3145728 | Ram FS size is applicable only when RAM based file system is selected. |
| ramfs_start_addr | 0x10000000 | RAM FS start address is applicable only when RAM based file system is selected. |
| enable_exfat | false | Enables support for exFAT file system. 0: Disable exFAT 1: Enable exFAT(Also Enables LFN) |
| word_access | True | Enables word access for misaligned memory access platform. |
| use_chmod | false | Enables use of CHMOD functionality for changing attributes (valid only with read_only set to false). |

XILINX

*Appendix A*

# Additional Resources and Legal Notices

## Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see Xilinx Support.

## Documentation Navigator and Design Hubs

Xilinx® Documentation Navigator (DocNav) provides access to Xilinx documents, videos, and support resources, which you can filter and search to find information. To open DocNav:

- From the Vivado® IDE, select **Help → Documentation and Tutorials**.
- On Windows, select **Start → All Programs → Xilinx Design Tools → DocNav**.
- At the Linux command prompt, enter `docnav`.

Xilinx Design Hubs provide links to documentation organized by design tasks and other topics, which you can use to learn key concepts and address frequently asked questions. To access the Design Hubs:

- In DocNav, click the **Design Hubs View** tab.
- On the Xilinx website, see the Design Hubs page.

***Note:*** For more information on DocNav, see the Documentation Navigator page on the Xilinx website.

# Please Read: Important Legal Notices

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at https://www.xilinx.com/legal.htm#tos.

### AUTOMOTIVE APPLICATIONS DISCLAIMER

AUTOMOTIVE PRODUCTS (IDENTIFIED AS "XA" IN THE PART NUMBER) ARE NOT WARRANTED FOR USE IN THE DEPLOYMENT OF AIRBAGS OR FOR USE IN APPLICATIONS THAT AFFECT CONTROL OF A VEHICLE ("SAFETY APPLICATION") UNLESS THERE IS A SAFETY CONCEPT OR REDUNDANCY FEATURE CONSISTENT WITH THE ISO 26262 AUTOMOTIVE SAFETY STANDARD ("SAFETY DESIGN"). CUSTOMER SHALL, PRIOR TO USING OR DISTRIBUTING ANY SYSTEMS THAT INCORPORATE PRODUCTS, THOROUGHLY TEST SUCH SYSTEMS FOR SAFETY PURPOSES. USE OF PRODUCTS IN A SAFETY APPLICATION WITHOUT A SAFETY DESIGN IS FULLY AT THE RISK OF CUSTOMER, SUBJECT ONLY TO APPLICABLE LAWS AND REGULATIONS GOVERNING LIMITATIONS ON PRODUCT LIABILITY.

### Copyright