# LibXil Flash (v4.0)

## Overview

The XilFlash library provides read/write/erase/lock/unlock features to access a parallel flash device. Flash device family specific functionality are also supported by the library. This library requires the underlying hardware platform to contain the following:

- axi_emc or similar core for accessing the flash.

This library implements the functionality for flash memory devices that conform to the "Common Flash Interface" (CFI) standard. CFI allows a single flash library to be used for an entire family of parts.This library supports Intel and AMD CFI compliant flash memory devices.

All the calls in the library are blocking in nature in that the control is returned back to user only after the current operation is completed successfully or an error is reported.

The following common APIs are supported for all flash devices:

- Initialize
- Read
- Write
- Erase
- Lock
- UnLock
- IsReady
- Reset
- Device specific control

You must call the "int XFlash_Initialize (XFlash *InstancePtr, u32 BaseAddress, u8 BusWidth, int IsPlatformFlash)" API before calling any other API in this library.

## XilFlash Library APIs

This section provides a linked summary and detailed descriptions of the LibXil Flash library APIs.

### API Summary

The following is a summary list of APIs provided by the LibXil Flash library. The list is linked to the API description. Click on the API name to go to the description.

int XFlash_Initialize (XFlash *InstancePtr, u32 BaseAddress, u8 BusWidth, int IsPlatformFlash)
int XFlash_Reset (XFlash *InstancePtr)
int XFlash_Read (XFlash *InstancePtr, u32 Offset, u32 Bytes, void *DestPtr)
int XFlash_Write (XFlash *InstancePtr, u32 Offset, u32Bytes, void *SrcPtr)
int XFlash_Erase (XFlash *InstancePtr, u32 Offset, u32 Bytes)
int XFlash_Lock (XFlash *InstancePtr, u32 Offset, u32 Bytes)
int XFlash_UnLock (XFlash *InstancePtr, u32 Offset, u32 Bytes)
int XFlash_DeviceControl (XFlash *InstancePtr, u32 Command, DeviceControl *Parameters)
int XFlash_IsReady (XFlash *InstancePtr)

# XilFlash Library API Descriptions

`int` **`XFlash_Initialize`** `(XFlash *InstancePtr, u32 BaseAddress, u8 BusWidth, int IsPlatformFlash)`

| | |
|---|---|
| Parameters | *InstancePtr* is a pointer to XFlash Instance.<br>BaseAddress is the base address of the Flash memory.<br>BusWidth is the total width of the Flash memory, in bytes.<br>IsPlatformFlash specifies whether the Flash memory is a Xilinx® Platform Flash configuration memory device. |
| Returns | XST_SUCCESS if successful.<br>XFLASH_PART_NOT_SUPPORTED if the command set algorithm or the layout is not supported by any flash family compiled into the system.<br>XFLASH_CFI_QUERY_ERROR if the device would not enter the CFI query mode. Either device doesn't support CFI or unsupported part layout exists or a hardware problem exists. |
| Description | Initializes a specific XFlash Instance.<br>The initialization entails:<br>• Issuing the CFI query command<br>• Identifying the Flash family and layout from CFI data<br>• Setting the default options for the instance<br>• Setting up the VTable<br>• Initialize the Xilinx Platform Flash XL to Async mode if the user selects to use the Platform Flash XL. The Platform Flash XL is an Intel CFI complaint device. |
| Includes | xilflash.h<br>xilflash_cfi.h<br>xilflash_intel.h<br>xilflash_amd.h |

---

`int` **`XFlash_Reset`** `(XFlash *InstancePtr)`

| | |
|---|---|
| Parameters | *InstancePtr* is a pointer to XFlash Instance. |
| Returns | XST_SUCCESS if Successful.<br>XFLASH_BUSY if the flash devices were in the middle of an operation and could not be reset.<br>XFLASH_ERROR if the device has experienced an internal error during the operation. XFlash_DeviceControl() must be used to access the cause of the device specific error condition. |
| Description | Resets the flash device and places it in read mode. |
| Includes | xilflash.h<br>xilflash_cfi.h<br>xilflash_intel.h<br>xilflash_amd.h |

`int` **`XFlash_Read`** `(XFlash *InstancePtr, u32 Offset, u32 Bytes, void *DestPtr)`

| | |
|---|---|
| Parameters | *InstancePtr* is a pointer to XFlash Instance.<br>*Offset* is the offset into the devices address space from which to read.<br>*Bytes* is the number of bytes to read.<br>*DestPtr* is the destination Address to copy data to. |
| Returns | XST_SUCCESS if successful.<br>XFLASH_ADDRESS_ERROR if the source address did not start within the addressable areas of the device. |
| Description | This API reads the data from the flash device and copies it into the specified user buffer. The source and destination addresses can be on any alignment supported by the processor. |
| Includes | xilflash.h<br>xilflash_cfi.h<br>xilflash_intel.h<br>xilflash_amd.h |

---

`int` **`XFlash_Write`** `(XFlash *InstancePtr, u32 Offset, u32Bytes, void *SrcPtr)`

| | |
|---|---|
| Parameters | *InstancePtr* is a pointer to XFlash Instance.<br>*Offset* is the offset into the devices address space from which to begin programming.<br>*Bytes* is the number of bytes to Program.<br>*SrcPtr* is the Source Address containing data to be programmed. |
| Returns | `XST_SUCCESS` if successful.<br>`XFLASH_ERROR` if a write error has occurred. The error is usually device specific. Use `XFlash_DeviceControl()` to retrieve specific error conditions. When this error is returned, it is possible that the target address range was only partially programmed. |
| Description | Programs the flash device with the data specified in the user buffer. The source and destination addresses must be aligned to the width of the flash data bus.<br>If the processor supports unaligned access, then the source address does not need to be aligned to the flash width; however, this library is generic, and because some processors (such as MicroBlaze™ processors) do not support unaligned access, this API requires that the source address be aligned. |
| Includes | `xilflash.h`<br>`xilflash_cfi.h`<br>`xilflash_intel.h`<br>`xilflash_amd.h` |

`int` **`XFlash_Erase`** `(XFlash *`*`InstancePtr`*`, u32 `*`Offset`*`, u32 `*`Bytes`*`)`

| | |
|---|---|
| Parameters | *`InstancePtr`* is a pointer to XFlash Instance.<br>*`Offset`* is the offset into the devices address space from which to begin erasure.<br>*`Bytes`* is the number of bytes to Erase. |
| Returns | `XST_SUCCESS` if successful.<br>`XFLASH_ADDRESS_ERROR` if the destination address range is not completely within the addressable areas of the device. |
| Description | This API erases the specified address range in the flash device. The number of bytes to erase can be any number as long as it is within the bounds of the devices. |
| Includes | `xilflash.h`<br>`xilflash_cfi.h`<br>`xilflash_intel.h`<br>`xilflash_amd.h` |

---

`int` **`XFlash_Lock`** `(XFlash *`*`InstancePtr`*`, u32 `*`Offset`*`, u32 `*`Bytes`*`)`

| | |
|---|---|
| Parameters | *`InstancePtr`* is a pointer to XFlash instance.<br>*`Offset`* is the offset of the block address into the devices address space which need to be locked.<br>*`Bytes`* is the number of bytes to be locked. |
| Returns | `XST_SUCCESS` if successful.<br>`XFLASH_ADDRESS_ERROR` if the destination address range is not completely within the addressable areas of the device. |
| Description | Locks a block in the flash device. |
| Includes | `xilflash.h`<br>`xilflash_cfi.h`<br>`xilflash_intel.h`<br>`xilflash_amd.h` |

`int` **`XFlash_UnLock`** `(XFlash *InstancePtr, u32 Offset, u32 Bytes)`

| | |
|---|---|
| Parameters | *InstancePtr* is a pointer to XFlash Instance.<br>*Offset* is the offset of the block address into the devices address space which need to be unlocked.<br>*Bytes* is the number of bytes to be unlocked. |
| Returns | XST_SUCCESS if successful.<br>XFLASH_ADDRESS_ERROR if the destination address range is not completely within the addressable areas of the device. |
| Description | Unlocks previously locked blocks that are locked. |
| Includes | xilflash.h<br>xilflash_cfi.h<br>xilflash_intel.h<br>xilflash_amd.h |

---

`int` **`XFlash_DeviceControl`** `(XFlash *InstancePtr, u32 Command, DeviceControl *Parameters)`

| | |
|---|---|
| Parameters | *InstancePtr* is a pointer to XFlash Instance.<br>*Command* is the device specific command to issue.<br>*Parameters* specifies the arguments passed to the device control function. |
| Returns | XST_SUCCESS if successful.<br>XFLASH_NOT_SUPPORTED if the command is not supported by the device. |
| Description | Executes device specific commands. |
| Includes | xilflash.h<br>xilflash_cfi.h<br>xilflash_intel.h<br>xilflash_amd.h |

---

`int` **`XFlash_IsReady`** `(XFlash *InstancePtr)`

| | |
|---|---|
| Parameters | *InstancePtr* is a pointer to XFlash instance. |
| Returns | TRUE if the device has been initialized; otherwise, FALSE. |
| Description | Checks the device readiness, signifying successful initialization. |
| Includes | xilflash.h<br>xilflash_cfi.h<br>xilflash_intel.h<br>xilflash_amd.h |

## Libgen Customization

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

```
BEGIN LIBRARY
PARAMETER LIBRARY_NAME = xilflash
PARAMETER LIBRARY_VER = 4.0
PARAMETER PROC_INSTANCE = microblaze_0
PARAMETER ENABLE_INTEL = true
PARAMETER ENABLE_AMD = false
END
```

Where:

- `LIBRARY_NAME`—Is the library name (`xilflash`).
- `LIBRARY_VER`—Is the library version (`4.0`).
- `PROC_INSTANCE`—Is the processor name (`microblaze_0`).
- `ENABLE_INTEL`—Enables or disables the Intel flash device family (`true|false`).
- `ENABLE_AMD`—Enables or disables the AMD flash device family (`true|false`).