Renesas Synergy™ Platform

FileX® Source Module Guide

Introduction

The purpose of this document is to provide an easy reference for the FileX® source component in the Renesas Synergy™ e² studio ISDE. The properties are explained in greater detail than the footer comment supplied with each property. Context specific usage is included to help understand when to change the default values. This document should make it easier to use the FileX source component without having to cross reference with the *Express Logic FileX User Guide*, and help you get familiarized more quickly with FileX features.

Contents

1. When to Include the FileX® Source Component

Adding the FileX source component enables the developer in the Synergy configurator environment to customize the FileX library, change values from default settings, and enable or disable certain features. Otherwise they must use the prebuilt FileX library. In most projects beyond the simplest, the developer will typically want to customize their FileX environment. Note that the ThreadX® source component is automatically added when adding a higher level source component (like FileX, NetX, NetX Duo, GUIX, USBX).

Without adding the FileX source component, the e² studio configurator will use a prebuilt library with the FileX default settings.

2.   Adding the FileX® Source Component

In the e² studio configurator, add the ThreadX® source component by selecting any thread from the Threads list and pressing the **New Stack** button and navigating the menu to **X-Ware** -> **FileX** -> **FileX Source**. Often the FileX source is available as an option when a high-level framework is created. For example, the FileX source module is available as an option for the FileX on USB Mass Storage framework module as seen in the following thread stack diagram.
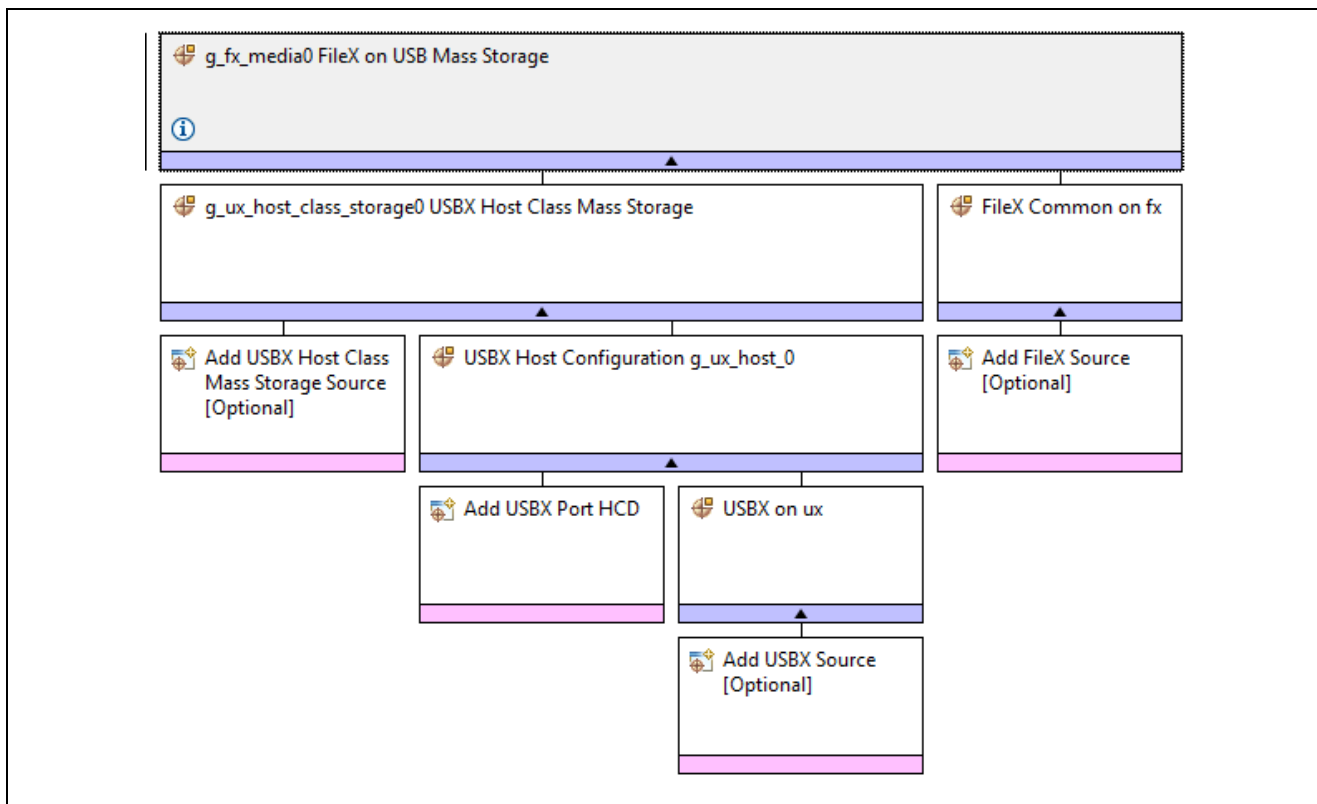

Figure 1   FileX Source Module Stack

3.   Changing FileX® Source Properties

After changing FileX property settings, the developer must click on the **Generate Project Content** button to update the project configurator in e² studio. Then the FileX library **must** be rebuilt and rebuild the project. Simply changing a property (or applying a #define in the preprocessor list) without rebuilding the project will not affect any change. e² studio will use the previously built library.

Default settings are based on use experience and are often the choice that will apply to the most common use cases.

4.   Configuring the FileX® Source Module

The properties of the FileX Source component are given in the order they appear in the properties window of the Synergy configurator.

Error Checking – default value enabled – It is generally enabled during development and debugging phase, and disabled when building a release version. When enabled, FileX include error checking services that check input and other parameters before calling the actual API. Some of the things it checks for are:

- NULL pointer input
- Invalid non-pointer parameters, such as an invalid file or directory names.
- Required configurable option must be enabled, for example, performance information must be enabled to call the get services.
- The data structure IDs must match what is expected, for example,
  — file_ptr -> fx_file_id != FX_FILE_ID /* check the file instance structure */
- Size of the data structure for example, the FileX file, matches the size of the data structure in the FileX library.

These last two checks guard against an application using a different version of the FileX library than the application is using.

Disabling the FileX error checking API results in improved performance (as much as 30%) and smaller code size.

Max Long Name Len – Maximum Long Name Length – default value not displayed, 33 characters is used – Defines the maximum size of long file names supported by FileX. The default value is 33. The minimum value is 13 and the maximum value is 256.

Max Last Name Len – Maximum Last Name Length – default value not displayed, 256 characters is used – Defines the maximum size of last opened file names supported by FileX. The default value is 256. The minimum value is 13 and the maximum value is 256. Must be as large as or larger than Max Long Name Len.

Max Sector Cache – Maximum Sector Cache Size - default value not displayed, 256 is used – Defines the maximum number of logical sectors that can be cached by FileX. The cache memory supplied to FileX at fx_media_open determines how many sectors can actually be cached. Minimum value is 2, all other values must be power of 2.

Fat Map Size – FAT Map Size - default value not displayed, 128 is used – Defines the size in bytes of the bit map used to update the secondary FAT sectors. The larger the value the less unnecessary secondary FAT sector writes. Minimum value is 1, no maximum value.

Max Fat Cache – Maximum FAT Cache - default value not displayed, 16 is used – Defines the number of entries in the internal FAT cache. The minimum value is 8, all values must be a power of 2.

Update Rate (seconds) - default value not displayed, 10 seconds used - Specifies rate at which the system time in FileX is adjusted. The default value 10 means that the FileX system time is updated every 10 seconds.

No Timer – default value disabled – When enabled, FileX is built without update to the time parameters. Eliminates the ThreadX® timer setup to update the FileX system time and date. Doing so causes default time and date to be placed on all file operations.

Single Thread – default value disabled – When enabled, FileX is running in a single-threaded environment and does not need thread protection. Eliminates ThreadX protection logic from the FileX source. It should be used if FileX is being used only from one thread or if FileX is being used without ThreadX.

Don't Update Open Files – default value disabled – When enabled, FileX does not update already opened files.

Media Search Cache – default value enabled – When enabled, a cache is used for optimization when searching for open media. Disabling this option will remove this optimization, reducing code size and memory footprint at the expense of performance.

Direct Data Read Cache Fill – default value enabled – When enabled, data sector reads are cached for faster access. Disabling this feature will reduce code size and memory footprint at the expense of performance.

Media Statistics – default value enabled – Determine if media statistics are kept and gathered. When disabled, no media statistics are available, this improves performance slightly and reduces code size; each instance of the FX_MEDIA_STRUCT structure is considerably smaller.

Single Open Legacy – default value disabled – When enabled, legacy single open logic for the same file is used. This may be necessary to make the fx_file_open behave in the same way as older versions of FileX, when migrating old FileX application code.

Rename Path Inherit – default value disabled – When enabled, renaming inherits path information. In other words, prepend the path in the new file name, then override the old file name with the new one on the renamed file.

No Local Path – default value disabled – When enabled, the local path logic is disabled. When disabled, a local path is kept for each thread; all operations performed with a relative path, will be relative to this local path.

Fault Tolerant Data – default value disabled – When enabled, data sector write requests are flushed immediately to the driver. This will increase the likelihood that data is not lost in case of power loss or a reset, but it will impact performance.

Fault Tolerant – default value disabled – When enabled, system sector write requests (including FAT and directory entry requests) are flushed immediately to the driver. This will increase the likelihood that data is not lost in case of power loss or a reset, but it will impact performance.

64-bit LBA – default value enabled – When enabled, 64-bits sector addresses are used in the I/O driver. This allows for bigger media and bigger files.

Fault Tolerant Service – default value disabled – Enables or disables the FileX fault tolerant service. The FileX Fault Tolerant Module is designed to prevent file system corruption caused by interruptions during the file or directory update. For example, when appending data to a file, FileX needs to update the content of the file, the directory entry,

RENESAS

and possibly the FAT entries. If this sequence of update is interrupted (such as power glitch, or the media is ejected in the middle of the update), the file system is in an inconsistent state, which may affect the integrity of the entire file system, leading towards corruption of other files.

Fault Tolerant Boot Index – default is 16 – Defines byte offset in the boot sector where the cluster for the fault tolerant log is. By default, this value is 116. This field takes 4 bytes. Bytes 116 through 119 are chosen because they are marked as reserved by FAT 12/16/32/exFAT specification.

Fault Tolerant Minimal Cluster Size – default is 3072 – Defines the requirement for minimal size of cluster in bytes. It must be multiple of sector size. The default value is 3072, which works with the worst case for long file rename.

5. FileX® Fault Tolerant Module

When an application writes data into a file, FileX updates both data clusters and system information. These updates must be completed as an atomic operation to keep information in the file system coherent. For example, when appending data to a file, FileX needs to find an available cluster in the media, update the FAT chain, update the length filed in the directory entry, and possibly update the starting cluster number in the directory entry. Either a power failure or media ejection can interrupt the sequence of updates, which will leave the file system in an inconsistent state. If the inconsistent state is not corrected, the data being updated can be lost, and because of damage to the system information, subsequent file system operation may damage other files or directories on the media.

The FileX Fault Tolerant Module works by journaling steps required to update a file before these steps are applied to the file system. If the file update is successful, these log entries are removed. However, if the file update is interrupted, the log entries are stored on the media. Next time the media is mounted, FileX will detect these log entries from the previous (unfinished) write operation. In such cases, FileX can recover from a failure by either rolling back the changes already made to the file system, or by reapplying the required changes to complete the previous operation. In this way, the FileX Fault Tolerant Module maintains file system integrity if the media loses power during an update operation.

Note: The FileX Fault Tolerant Module is not designed to prevent file system corruption caused by physical media corruption with valid data in it.

After the FileX Fault Tolerant module protects a media, the media must not be mounted by anything other than FileX with Fault Tolerant enabled. Doing so can cause the log entries in the file system to be inconsistent with system information on the media. If the FileX Fault Tolerant module attempts to process log entries after the media is updated by another file system, the recovery procedure may fail, leaving the entire file system in an unpredictable state.

The FileX Fault Tolerant feature is available to all FAT file systems supported by FileX, including FAT12, FAT16, FAT32, and exFAT. To enable the fault tolerant feature, FileX must be built with the option **Fault tolerant service** enabled. At run time, the application starts the fault tolerant service by calling `fx_fault_tolerant_enable()` immediately after the call to `fx_media_open`. After fault tolerant is enabled, all file write operations to the designated media are protected. By default, the fault tolerant module is not enabled.

The application needs to make sure the file system is not being accessed prior to `fx_fault_tolerant_enable()` being called. If application writes data to the file system prior to fault tolerant enable, the write operation could corrupt the media if prior write operations were not completed, and the file system.

The FileX fault tolerant log takes up one logical cluster in flash. The index to the starting cluster number of that cluster is recorded in the boot sector was not restored using fault tolerant log entries. For further details on the log format refer to the FileX user guide.

6. exFAT Support

The exFAT file system format is patented by Microsoft and a special license is necessary for it. For further information about licensing and accessing the FileX version with exFAT support, please contact a Renesas representative.

Website and Support

Visit the following vanity URLs to learn about key elements of the Synergy Platform, download components and related documentation, and get support.

Synergy Software   renesassynergy.com/software
    Synergy Software Package   renesassynergy.com/ssp
    Software add-ons   renesassynergy.com/addons
    Software glossary   renesassynergy.com/softwareglossary
    Development tools   renesassynergy.com/tools

Synergy Hardware   renesassynergy.com/hardware
    Microcontrollers   renesassynergy.com/mcus
    MCU glossary   renesassynergy.com/mcuglossary
    Parametric search   renesassynergy.com/parametric
    Kits   renesassynergy.com/kits

Synergy Solutions Gallery   renesassynergy.com/solutionsgallery
    Partner projects   renesassynergy.com/partnerprojects
    Application projects   renesassynergy.com/applicationprojects

Self-service support resources:
    Documentation   renesassynergy.com/docs
    Knowledgebase   renesassynergy.com/knowledgebase
    Forums   renesassynergy.com/forum
    Training   renesassynergy.com/training
    Videos   renesassynergy.com/videos
    Chat and web ticket   renesassynergy.com/support

Revision History

Rev.	Date	Description	
		Page	**Summary**
1.00	Jun 29, 2018	-	Initial release

RENESAS

Renesas Electronics Corporation

http://www.renesas.com