FTL NAND Media Driver for Micron MT29FxG08 with ECC User Guide

Version 1.30

For use with FTL NAND Media Driver for Micron® MT29FxG08 with Internal ECC versions 1.06 and above

Exported on 04/23/2018

Table of Contents

1 System Overview

This chapter contains the fundamental information for this module.

The component sections are as follows:

- Introduction – describes the main elements of the module.
- Feature Check – summarizes the main features of the module as bullet points.
- Device Description – summarizes the features of the supported devices.
- Packages and Documents – the *Packages* section lists the packages that you need in order to use this module. The *Documents* section lists the relevant user guides.
- Change History – lists the earlier versions of this manual, giving the software version that each manual describes.

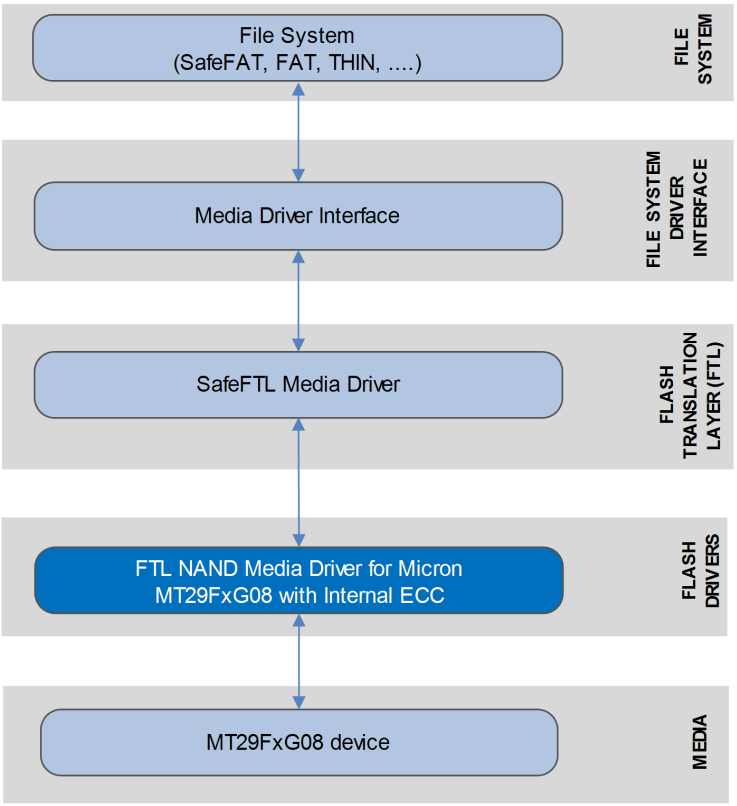
1.1 Introduction

This guide is for those who want to use HCC's FTL NAND Media Driver for Micron® MT29FxG08 flash devices that use internal Error Correcting Code (ECC). The driver supports MT29F1G08, MT29F2G08, and MT29F4G08 devices that use internal ECC.

> **Note:** If internal ECC is not being used, consult sales@hcc-embedded.com about using a different HCC Micron® flash driver.

This guide covers all aspects of configuration and use. This media driver conforms to the HCC Media Driver Interface Specification.

The diagram below shows a typical system architecture including a file system, media driver and media.



The media driver provides an interface for a file system to read from and write to a NAND Flash storage device. The file system handles all drives identically, regardless of their internal design features.

Note the following:

- The file system can be any HCC file system that addresses logical sector arrays (including SafeFAT, FAT, and THIN).
- The Flash Translation Layer (FTL) is the SafeFTL media driver. This has its own manual.
- The NAND flash driver is written specifically for the NAND flash controller (integrated with the MT29FxG08 microcontroller) and the specific NAND flash array used.

1.2 Feature Check

The main features of the media driver are the following:

- Conforms to the HCC Advanced Embedded Framework.
- Designed for integration with both RTOS and non-RTOS based systems.
- Conforms to the low level NAND flash interface defined by HCC's SafeFTL Flash Translation Layer.
- Supports Micron® MT29FxG08 flash drives that use internal Error Correcting Code (ECC).

1.3 Device Description

This table summarizes the features of the supported devices.

Note: If internal ECC is not being used, consult sales@hcc-embedded.com about using a different HCC Micron® flash driver.

	MT29F1G08	MT29F2G08	MT29F4G08
Size (Gb)	1	2	4
Width	8	8	8
Plane size	2 planes x 512 blocks per plane	2 planes x 1024 blocks per plane	2 planes x 2048 blocks per plane
Page size	2048 bytes + 64 spare bytes.	2048 bytes + 64 spare bytes.	2048 bytes + 64 spare bytes.
Blocks	1024	2048	4096
Block size	64 pages (128K + 4K bytes)	64 pages (128K + 4K bytes)	64 pages (128K + 4K bytes)
Internal ECC	4 bit	4 bit	4 bit
SLC	Yes	Yes	Yes

1.4 Packages and Documents

Packages

The table below lists the packages that you need in order to use this module:

Package	Description
hcc_base_doc	This contains the two guides that will help you get started.
media_drv_ftl_base	The base SafeFTL package.
media_drv_ftl_nand_mt29fxg08	The media driver package described in this document.

Documents

For an overview of HCC file systems and flash management technologies, see Product Information on the main HCC website.

Readers should note the points in the HCC Documentation Guidelines on the HCC documentation website.

HCC Firmware Quick Start Guide

This document describes how to install packages provided by HCC in the target development environment. Also follow the *Quick Start Guide* when HCC provides package updates.

HCC Source Tree Guide

This document describes the HCC source tree. It gives an overview of the system to make clear the logic behind its organization.

HCC Media Driver Interface Guide

This document describes the HCC Media Driver Interface Specification.

HCC SafeFTL User Guide

The user guide for SafeFTL.

HCC FTL NAND Media Driver for Micron® MT29FxG08 with ECC User Guide

This is this document.

1.5 Change History

This section describes past changes to this manual.

- To view or download manuals, see FTL Media Driver PDFs.
- For the history of changes made to the package code itself, see History: media_drv_ftl_nand_mt29fxg08.

The current version of this manual is 1.30. The full list of versions is as follows:

Manual version	Date	Software version	Reason for change
1.30	2018-04-23	1.06	Added MT29FXG08_ECC_SWITCH_ECC configuration option.
1.20	2017-08-11	1.05	Corrected *Packages* list.
1.10	2017-06-27	1.05	New *Change History* format.
1.00	2016-02-15	1.05	First online version.

2 Source File List

This section describes all the source code files included in the system. These files follow the HCC Embedded standard source tree system, described in the HCC Source Tree Guide. All references to file pathnames refer to locations within this standard source tree, not within the package you initially receive.

Note: Do not modify any files except the configuration files and PSP files.

2.1 API Header File

The file **src/api/api_ftl_nand_mt29fxg08_ecc.h** is the only file that should be included by an application using this module. For details of the single API function, see Application Programming Interface.

2.2 Configuration File

The file **src/config/config_ftl_nand_mt29fxg08_ecc.h** contains all the configurable parameters of the system. Configure these as required.

2.3 Source Code File

The file **src/media-drv/ftl/drivers/nand/micron/nand_mt29fxg08_ecc.c** holds the source code for the media driver. **This file should only be modified by HCC**.

2.4 Platform Support Package (PSP) Files

These files provide functions the core code needs to call, depending on the hardware. There are two sets of files, in the directories **src/psp_stm32f4xx/target/nand_mt29fxg08_ecc** and **src/psp_omapl138/target/nand_mt29fxg08_ecc**.

Note: You must modify these PSP implementations for your specific microcontroller and development board; see PSP Porting for details.

The files are as follows:

File	Description
psp_nand_mt29fxg08_ecc.c	Source code.
psp_nand_mt29fxg08_ecc.h	Header file.

2.5 Version File

The file **src/version/ver_ftl_nand_mt29fxg08_ecc.h** contains the version number of this module. This version number is checked by all modules that use a module to ensure system consistency over upgrades.

3 Configuration Options

Set the system configuration options in the file **src/config/config_ftl_nand_mt29fxg08_ecc.h**. This section lists the available configuration options and their default values.

MT29FXG08_ECC_ID

The NAND ID. The default is ((0x2C << 0) | (0xDC << 8) | (0x90 << 16) | (0x95 << 24)).

MT29FXG08_ECC_BLOCK_NUM

The number of erasable blocks in the target flash array. The default is 4096.

MT29FXG08_ECC_PAGE_DATA_SIZE

The data area in bytes available on one page. The default is 2048.

MT29FXG08_ECC_PAGE_TOTAL_SIZE

The total size of the page, including the data and spare areas. The default is 2112.

MT29FXG08_ECC_PAGE_PER_BLOCK

The number of pages per erasable block. The default is 64.

MT29FXG08_ECC_FREE_BLOCK_AVAILABLE

The number of free blocks. The value must not exceed MDRIVER_FTL_MAX_FREE_BLOCKS. The default is 99.

MT29FXG08_ECC_LOG_BLOCK_AVAILABLE

The number of log blocks. The default is 6; do not set it below this.

MT29FXG08_ECC_NUM_OF_DIF_MAPBLOCK

The number of blocks used for mapping in the system. The default is 2. The valid range is 1 to 16.

MT29FXG08_ECC_MAPBLOCK_SHADOW

The number of map shadow blocks. The default is 2.

The system may be more efficient if more map shadow blocks are used, but each additional block reduces the number of free blocks in the system.

MT29FXG08_ECC_RESERVED_BLOCKS

The number of reserved blocks, the blocks at the start of the flash area that driver should not use. The default is 0.

MT29FXG08_ECC_WEAR_STATIC_LIMIT

The maximum value that the difference between the maximum and minimum wear count can be. The default is 1024.

MT29FXG08_ECC_WEAR_STATIC_COUNT

The number of merge operations allowed before static wear checking must be run. The default is 128.

MT29FXG08_ECC_REWRITE_INTERVAL

The number of normal read/write operations allowed before a rewrite check must be run. The default is 512.

MT29FXG08_ECC_LL_TEST

Set this to 1 to enable driver testing. The default is 0.

MT29FXG08_ECC_SWITCH_ECC

There are two settings:

- 0 - ECC switching is not supported. It is switched on/off at time of manufacture and cannot be changed by software. Example: MT29F2G08ABAGAxx.
- 1 (the default) - ECC can be switched on/off by using the software (see Feature Address 90h, Array Operation Mode).

4 Application Programming Interface

This section describes the single function and the structure it uses.

When the media driver is used:

1. The file system calls the media driver's **nand_mt29fxg08_ecc_init()** function.
2. **nand_mt29fxg08_ecc_init()** returns a pointer to a *t_ftl_driver* structure containing a set of functions for accessing the media driver.

4.1 nand_mt29fxg08_ecc_init

This function initializes the driver for this device.

This is normally called automatically from SafeFTL, using its table of flash drives. Refer to the *HCC SafeFTL User Guide* for details.

Format

```
t_ftl_ret nand_mt29fxg08_ecc_init (
    uint32_t           drvnum,
    t_ftl_driver * *   pps_ftl_driver )
```

Arguments

Argument	Description	Type
drv_num	The number of the drive to initialize. The first drive is 0.	uint32_t
pps_ftl_driver	On return, a pointer to a *t_ftl_driver* structure defining the interface to that driver.	t_ftl_driver * *

Return values

Return value	Description
0	Successful execution.
1	Operation failed.

4.2 SafeFTL Flash Drive Structure Example

SafeFTL uses a flash drive structure containing all the available flash drives. Each available flash driver must have an entry in this table, specifying its initialization function and the parameter to be passed to it in that function. The flash drives are numbered from 0 to (MDRIVER_FTL_MAX_DRIVE-1). The index to this table is used to reference the flash drive.

This structure is held in the main SafeFTL package's **src/config/config_mdriver_ftl.c** file.

The following example shows how an MT29FxG08 drive would appear in this structure. In this case it is the only NAND drive, followed by two NOR drives.

```
t_ftldrive_init  as_ftldrive_init[MDRIVER_FTL_MAX_DRIVE] =
{
  { nand_mt29fxg08_ecc_init, 0U }
  , { ftl_nor_init, 0U }
  , { ftl_nor_init, 1U }
};
```

4.3 Error Codes

The possible return codes are shown in the table below:

Code	Value	Description
LL_OK	0	Successful execution.
LL_ERASED	1	Page is empty.
LL_ERROR	2	Other error.

4.4 t_ftl_driver

The **nand_mt29fxg08_ecc_init()** function returns a pointer to this *t_ftl_driver* structure.

Element	Type	Description
user_data	uint32_t	User-defined data.
pf_getphy	(* pf_getphy)	Pointer to **getphy()** function.
pf_read	(* pf_read)	Pointer to **read()** function.
pf_readpart	(* pf_readpart)	Pointer to **readpart()** function.
pf_write	(* pf_write)	Pointer to **write ()** function.
pf_writedouble	(* pf_writedouble)	Pointer to **writedouble()** function.
pf_erase	(* pf_erase)	Pointer to **erase()** function.
pf_isbadblock	(* pf_isbadblock)	Pointer to **isbadblock()** function.
pf_readonebyte	(* pf_readonebyte)	Pointer to **readonebyte()** function.

5 Integration

This section describes all aspects of the module that require integration with your target project. This includes porting and configuration of external resources.

5.1 PSP Porting

The Platform Support Package (PSP) is designed to hold all platform-specific functionality, either because it relies on specific features of a target system, or because this provides the most efficient or flexible solution for the developer.

The module makes use of the following standard PSP macros:

Macro	Package	Element	Description
PSP_RD_LE32	psp_base	psp_endianness	Reads a 32 bit value stored as little-endian from a memory location.
PSP_WR_LE32	psp_base	psp_endianness	Writes a 32 bit value to be stored as little-endian to a memory location.

The module makes use of the following PSP functions from the file **src/psp/target/nand_mt29fxg08_ecc/psp_nand_mt29fxg08_ecc.h**:

Function	Description
psp_nand_mt29fxg08_ecc_init()	Initializes the PSC, FCR register, and flash pins for NAND access.
psp_nand_mt29fxg08_ecc_wait_ready()	Waits for the flash to become ready. If the flash is not ready by the end of the time specified, an error is returned.

These functions are described in the following sections.

psp_nand_mt29fxg08_ecc_init

This function is provided by the PSP to initialize the device.

This initializes the PSC, FCR register, and flash pins for NAND access.

Format

```
uint32_t psp_nand_mt29fxg08_ecc_init ( void )
```

Arguments

None.

Return Values

Return value	Description
0	Successful execution.
Else	Operation failed.

psp_nand_mt29fxg08_ecc_wait_ready

This function is provided by the PSP to wait until the flash becomes ready or the timeout expires.

Format

```
uint32_t psp_nand_mt29fxg08_ecc_wait_ready ( uint32_t tick )
```

Arguments

Argument	Description	Type
tick	The timeout value in ticks.	uint32_t

Return Values

Return value	Description
0	The flash is ready.
Else	The timeout expired.