Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.



THIS SPEC IS OBSOLETE

Spec No: 001-60081

Spec Title: AN60081 - INTERFACING SAMSUNG MUXONENAND TO CYPRESS USB CONTROLLERS

Sunset Owner: Rama Sai Krishna Vakkantula (RSKV)

Replaced By: NONE

AN60081

Interfacing Samsung MuxOneNand to Cypress USB Controllers

Author: Basil P Varghese
Associated Project: No
Associated Part Family: CY7C68013A
Software Version: None
Related Application Notes: None

AN60081 explains how to interface Cypress EZ-USB® controller to a Samsung MuxOneNand. It includes a complete API library with source code, and helps to integrate MuxOneNand support to any Cypress High Speed USB controller-based systems.

OBSOLETE

Introduction

The EZ-USB FX2LP™ is a versatile, USB 2.0 compliant High Speed USB controller from Cypress. FX2LP has an 8051 core and a highly configurable interface (GPIF) to external memory devices and other peripherals or controllers. Cypress also provides a configurable Firmware Frameworks written in C, which implements a fully compliant USB interface for FX2LP.

The MuxOneNand is a single-chip solution from Samsung Semiconductors comprising a NAND Flash array, NOR Flash interface, 4 KB data buffer, and 1 KB boot buffer. The NOR interface with a dual buffering enables fast read speeds and the NAND core of MuxOneNand provides fast write speeds. The MuxOneNand also has an integrated 4-bit Error Correction.

This application note explains how to interface the Cypress EZ-USB controller to a MuxOneNand.

MuxOneNand Memory Interface

Figure 1 shows the MuxOneNand Memory Interface. A brief description of the interface is given in Table 1.

Figure 1. MuxOneNand Interface

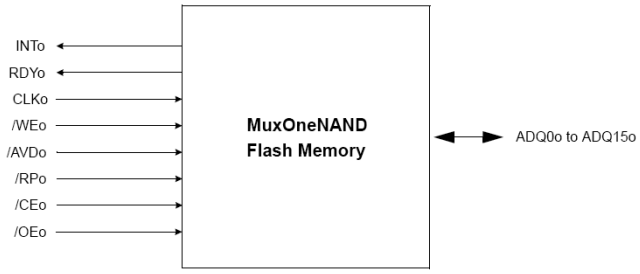


Table 1. MuxOneNand Pin Descriptions

Pin	Description
ADQ[15:0]	Multiplexed Address Inputs/Data Bus. This is the multiplexed 16 bit address and data bus. The address of registers and data are transmitted over this bus.
INT	Interrupt line from MuxOneNand to the controller. The MuxOneNand interrupts the controller on completion of read, write, erase, and reset operation.
CLK	The clock input line to the MuxOneNand from the controller.
WE	The WRITE ENABLE input to the MuxOneNand. This line is low during a data and register input to the MuxOneNand.
OE	The OUTPUT ENABLE input to the MuxOneNand. This line is low when the controller reads data from the MuxOneNand.
AVD	The Address Valid Input. This line is low when writing the register address or the buffer address.
CE	Chip Enable input to the MuxOneNand.
RDY	An output from MuxOneNand to the controller. This line is low when the MuxOneNand chip is busy.

Firmware - API Library

The library provided with this application note includes a comprehensive set of APIs to quickly integrate MuxOneNand support to any Cypress High Speed USB controller based systems. The APIs for the following functionalities are available; they are implemented for GPIF configured for 48 MHz.

- Synchronous Register Read/Write
- Nand Initialization
- Nand Page Program
- Nand Block Internal Move
- Nand Page Read
- Nand Block Erase

Three files are also included:

- cymuxonenand.h
- syncdly.h
- cymuxonenand.c

cymuxonenand.h

This header file has the required definitions for the MuxOneNand registers and commands. The controller GPIOs connecting the NAND is defined in this header file. It can be modified to match the actual interface of the target application.

```
/****************************************
/

//MuxOneNand Interface GPIOS

/****************************************
/

#define CE PC0 //Chip Enable. Port C-Pin0

#define RP PC3 //Reset. Port C-Pin3

#define INT PC4 //Interrupt. Port C-Pin3

#define WE PA7 //Write Enable. Port A-Pin7

#define OE PA6 //Output Enable. Port A-Pin6
```

You can modify this section of the header file to match the hardware connection of the target system.

syncdly.h

This file has definition of the SYNCDELAY macro, which is required to generate synchronization delay when you write to some registers of the EZ-USB controller.

cymuxonenand.c

This file has definitions for all the APIs. Here are the API descriptions:

- CyMuxOneNandInit

This function initializes the MuxOneNand. After initialization, the MuxOneNand is available for read, write, and erase operations in the following configuration:

  - Synchronous Read and Write
  - Burst Read, Write Latency = 4 Clock Cycles
  - All Block Unlocked for program and erase operations

`BYTE CyMuxOneNandInit(void)`

Parameters

None

Returns

- CY_MUXONENAND_DEVICE_NOT_FOUND

  A MuxOneNand device with the provided Device Id is not found
- CY_MUXONENAND_UNLOCK_ERROR

  Unlock operation failed

- CyMuxOneNandSyncReadRegister

This function returns the value of a register from the NAND.

`WORD CyMuxOneNandSyncReadRegister(WORD RegAddress)`

Parameters

- RegAddres - Register Address

Returns

- Register Contents

- CyMuxOneNandWriteRegister

This function writes to a NAND register.

`void OneNandSyncWriteRegister(WORD RegAddress, WORD Value)`

Parameters

- RegAddress - Register Address
- Value - Value written to the register

Returns

None

OBSOLETE

- CyMuxOneNandReadPages

This function uses three global variables - `glTargetPageNumber`, `glTargetBlockAddress`, and `glPageCount`. This function reads `glPageCount` number of pages starting from `glTargetPageNumber` of `glTargetBlockAddress` block. The appropriate values for these variables must be set before calling this function. When continuously reading from the NAND if a block boundary is reached, data page '0' of the next block is read.

The pages are read into EndPoint 6 of the controller. The EndPoint 6 is configured to Auto IN mode with automatic commit length of 512 bytes. If this configuration needs to be modified, follow the comments in the source code.

`void CyMuxOneNandReadPages(void)`

Parameters

None

Returns

None

Dependencies

- glTargetPageNumber
- glTargetBlockAddress
- glPageCount

- CyMuxOneNandWritePages

This function uses three global variables - `glTargetPageNumber`, `glTargetBlockAddress`, and `glPageCount`. This function writes to `glPageCount` number of pages starting from `glTargetPageNumber` of `glTargetBlockAddress` block. Set appropriate values for these variables before calling this function. This is implemented as a read-modify write. Values of the page not modified in the target block are not affected. For the purpose of this read-modify write, a staging block (block number 0x0FFE) is used; therefore, the target application must not use this block for data.

When writing continuously to the NAND, if a block boundary is reached, data is continuously written to the next block.

This function writes data from EndPoint 2 of the controller. The function waits until data is available in EndPoint 2 to write to the NAND. For the purpose of continuous write, EndPoint 2 is put into AUTO OUT mode in this function.

`void CyMuxOneNandWritePages(void)`

Parameters

None

Returns

None

Dependencies

- glTargetPageNumber
- glTargetBlockAddress
- glPageCount

- CyMuxOneNandCopyBlock

This function copies data from one block to another block internally in the NAND. Erase the target block before copying data.

`BYTE OneNandCopyBlock(WORD targetBlockNumber, WORD sourceBlockNumber, BYTE startPageNumber, BYTE endPageNumber)`

Parameters

- sourceBlockNumber - Source block from where data is copied
- targetBlockNumber - Target block to which data is copied
- startPageNumber - Data starting from this page is copied to the same page on target block
- endPageNumber - Data from pages until this page is copied from source block to target block

Returns

- CY_MUXONENAND_LOAD_ERROR - Error loading page in Nand
- CY_MUXONENAND_PROGRAM_ERROR - Error programming page in Nand

- CyMuxOneNandBlockErase

A call to this function erases the specified block.

`BYTE OneNandEraseBlock(WORD BlockAddress)`

Parameters

- BlockAddress - Block to be erased

Returns

- CY_MUXONENAND_ERASE_ERROR - Internal Error while erasing a block

OBSOLETE

Integrating APIs into Project

Integrating this library to a project is a two step process:

1. Modify *cymuxonenand.h* to match the target system hardware interface.
2. Include *cymuxonenand.h*, *syncdly.h*, and *cymuxonenand.c* files to the project.

Modifying *the cymuxonenand.h*

Two sections must be modified before this library is used to implement the application.

```
/*************************************/
//MuxOneNand Interface GPIOS
//*************************************/
#define CE PC0 //Chip Enable. Port C-Pin0
#define RP PC3 //Reset. Port C-Pin3
#define INT PC4 //Interrupt. Port C-Pin3
#define WE PA7 //Write Enable. Port A-Pin7
#define OE PA6 //Output Enable. Port A-Pin6
```

This section of the header file is modified to match the actual target interface. The default GPIOs used are those used to test this library.

```
/*************************************/
//Device Constants - IDs etc.
/*************************************/
#define ONENAND_DEVICE_ID 0x00EC
```

This section should be modified to match the correct ID from the data sheet.

Summary

You can use the library provided in this application note to quickly integrate MuxOneNand support to any application built around Cypress EZ-USB controllers. You can build complete systems such as a USB mass storage device using MuxOneNand using this API library.

OBSOLETE

Document History

Document Title: AN60081 - Interfacing Samsung MuxOneNand to Cypress USB Controllers

Document Number: 001-60081

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	2892506	PBV	03/15/10	New Application Note
*A	3943502	RSKV	03/25/2013	Updated document template.
*B	4184440	RSKV	11/06/2013	Obsolete document.

OBSOLETE

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Automotive	cypress.com/go/automotive
Clocks & Buffers	cypress.com/go/clocks
Interface	cypress.com/go/interface
Lighting & Power Control	cypress.com/go/powerpsoc cypress.com/go/plc
Memory	cypress.com/go/memory
PSoC	cypress.com/go/psoc
Touch Sensing	cypress.com/go/touch
USB Controllers	cypress.com/go/usb
Wireless/RF	cypress.com/go/wireless

PSoC® Solutions

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 5LP

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/go/support

OBSOLETE

EZ-USB is a registered trademark and FX2LP is a trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

© Cypress Semiconductor Corporation, 2010-2013. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

This Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.