



UG409020

User Guide

Verilog Behavioral Model for M29W160EB/T Flash Memory

This user guide describes the Verilog behavioral model for M29W160EB and M29W160ET Flash memory devices. The M29W160EB and M29W160ET memories will be referred to as M29W160EB/T throughout the document unless otherwise specified.

Contents

Verilog Model Delivery Package Architecture	3
Device Description	4
Verilog Behavioral Model	5
Modules and libraries	5
Modules	5
Libraries	6
Verilog Test Bench and Stimuli files.....	6
Simulation guidelines	7
Launching a Simulation	7
Simulation timings	7
memory_file file format	7
Other initialization files	7
CFImemory_B.vmf, CFImemory_T.vmf	7
GroupProtect_BT.vmf.....	7
Customize model and simulation.....	8
Verilog types used in model ports.....	8
Revision history	9
Legal Disclaimer	10

Verilog Model Delivery Package Architecture

The VERILOG Model Delivery Package, NU_M29W160E_VGxx.tar.gz, is organized into a main directory, named NU_M29W160E_VGxx, containing the following subdirectories with their related files.

NU_M29W160E_VGxx //main directory

- readme.txt
- run_ncsim
- run_modelsim
- code** //code subdirectory: contains code source files
 - I__ M29W160E.v Verilog model
- include** //include subdirectory: contains the library source files
 - I__ BankLib.h
 - CUIcommandData.h
 - data.h
 - def.h
 - ProtectLib.h
 - TimingData.h
 - UserData.h
- doc** //doc subdirectory contains documentation (user guide)
 - I__ UserGuide.pdf
- sim** //sim subdirectory: contains simulation initialization files
 - I__ memory_BT.vmf
 - CFImemory_B.vmf
 - CFImemory_T.vmf
 - GroupProtect_BT.vmf
- stim** //stim subdirectory: contains stimuli files for simulation
 - I__
 - I__ x16 AllTask.h
 - Various stimuli written
 - In Verilog language
 - I__ x8 AllTask.h
 - Various stimuli written
 - In Verilog language
- top** //top subdirectory: contains testbench file for simulation
 - I__ top.v testbench file

Device Description

The M29W160E is a 16 Mbit (2Mb x8 or 1Mb x16) non-volatile memory that can be read, erased and reprogrammed. These operations can be performed using a single low voltage (2.7 to 3.6V) supply. On power-up the memory defaults to its Read mode where it can be read in the same way as a ROM or EPROM.

The memory is divided into blocks that can be erased independently so it is possible to preserve valid data while old data is erased. Each block can be protected independently to prevent accidental Program or Erase commands from modifying the memory. Program and Erase commands are written to the Command Interface of the memory. An on-chip Program/Erase Controller simplifies the process of programming or erasing the memory by taking care of all of the special operations that are required to update the memory contents.

The end of a program or erase operation can be detected and any error conditions identified. The command set required to control the memory is consistent with JEDEC standards.

The blocks in the memory are asymmetrically arranged.

The first or last 64 KBytes have been divided into four additional blocks. The 16 KByte Boot Block can be used for small initialization code to start the microprocessor, the two 8 KByte Parameter Blocks can be used for parameter storage and the remaining 32K is a small Main Block where the application may be stored.

Chip Enable, Output Enable and Write Enable signals control the bus operation of the memory. They allow simple connection to most microprocessors, often without additional logic.

The memory is supplied with all the bits erased (set to '1').

Verilog Behavioral Model

The M29W160EB/T Verilog behavioral model is contained in the M29W160E.v file of the code subdirectory. It includes a set of modules that implement all the device functions listed in the device datasheet. See Section 2.1 for the description of modules and libraries.

Note: Please check the Numonyx web site or contact your local Numonyx Sales office for the most recent version of the device datasheet. Please refer to readme.txt file in the main package directory for reference datasheet used during model development and validation.

This model was validated using a Cadence NC-SIM 5.7 simulator. The use of this model with other simulators is not guaranteed.

Modules and libraries

The M29W160EB/T Verilog modules and libraries are described in the following sections. The code/M29W160E.v Verilog file and libraries code files must be compiled in the same order as described below and specified into run_ncsim file.

Modules

m29w160e module

This is the Core of the model (the top level module which make use of all others modules).

TimingCheckModule

This module checks for all timing constraints.

CFIQueryModule

This module manages the CFI memory.

CUIdecoderModule

This module decodes all one cycle commands.

CUIdecoder_busy Module

This Module decodes other commands.

CUIdecoder_busyCCSModule

This Module decodes certain specific commands.

CUIdecoderCCS module

This module decodes the AMD-like sequences.

EraseModule

This module implements the Erase Commands of the CUI.

MemoryModule

This module manages all operations on the memory array.

OutputBufferModule

This module manages all operations on the Data Bus.

ProgramModule

This module implements the Program Commands of the CUI.

ReadModule

This module implements the Read Memory Array Command.

Verilog Types Used in Model Ports

StatusRegModule

This module manages the Status Register and implements the Read Status Register Command and the Clear Status Register Command.

KernelModule

This module is the kernel of the model (manages the status of the model and the output messages).

BlockProtectModule

This module manage the Block Protection.

SignatureModule

This module implements the read electronic signature functions.

Libraries

BankLib.h

This Library contains general procedures and functions for representing and managing the banks and the blocks of the device.

CUIcommandData.h

This Library contains commands code and states of Command User Interface (CUI).

TimingData.h

This Library contains the definition of constants related with the timing constraints of the device.

UserData.h

This Library contains the definition of constants that can be changed by users like access time (70 ns, 80 ns, 90 ns), TimingChecks(on or off). This is the only library that can be modified by the user. For more information, refer to Section 3.5: Customize model and simulation.

data.h

This Library contains the definition of constants, types and procedures related with memory characteristics,

def.h

This Library contains the definition of constants of types.

ProtectLib.h

This Library contains task and functions for Block Protection.

Verilog Test Bench and Stimuli files

The top subdirectory of the VERILOG Model Delivery Package contains a testbench file, top.v.

Stimuli files in VERILOG format are available in the stim subdirectory. The stimuli files cover many operational conditions of the device, and in particular, the Command User Interface (CUI) commands.

The testbench and the stimuli files are written using the standard Verilog version.

Simulation guidelines

Launching a Simulation

The script, `run_ncsim`, is an example that allows launching the Cadence NC-SIM simulation. It is located in the main directory. This file compiles and elaborates the Verilog model file and the stimuli files contained into the `stim` directory.

Simulation timings

To reduce the simulation time, the user can reduce certain program and erase times in the Verilog simulation model.

memory_file file format

To facilitate testing of the memory array behavior and model functions, the memory array can be loaded with specific data at power-up.

The format of the `memory_file.vmf`, located in the `sim` subdirectory, must be as follows:

```
@hex_address
hex_data
hex_data_1
.....
```

where `hex_data` is stored at location `hex_address`, `hex_data_1` at the location `hex_address + 1`, and so on. The following is an example:

```
@07FFFF
C14B
129A
.....
```

Comments in memory file lines are allowed using the notation: `// comment`. The model is delivered with a template memory file in the `sim` directory called `memory_file.vmf`.

The user must write the filename of `memory_file` into the `UserData.h` library file, defining the alias called `FILENAME_mem`:

```
`define FILENAME_mem "filename"
```

If the user does not provide the initialization file (`memory_file:= " "`), all the memory bits are loaded with '1', therefore the entire array is erased.

Other initialization files

This section describes other initialization files included in the `sim` subdirectory.

CFImemory_B.vmf, CFImemory_T.vmf

This files defines the content of the CFI memory area, and not must be changed by the user.

GroupProtect_BT.vmf

This file defines the locking status of protection groups. Each line define the status of one protection group: the first line is used for the first protection group, the second line for the second protection group, and so on. The status unlocked is encoded with the value 0, the status locked with the value 1. This file is customizable by the user.

Verilog Types Used in Model Ports

Customize model and simulation

Certain features and characteristics of the model and the simulation process are customizable by the user. The parameters that the user can modify are contained in the UserData.h library file. The parameters whose value can be changed are:

- device name definition: This parameter is the first definition contained in UserData.h, and specifies which device is described by the model (each device is characterized by specific block organization).
- FILENAME_mem: This alias specifies the file that defines data loaded in the memory.
- TimingChecks: If this string is set to "on" value, all timing checks will be performed during the simulation. Otherwise, if it is set to "off", no timing checks are performed.
- t_access: This parameter defines the read access time of the device. Available values are 70, 80 and 90 (ns).

Verilog types used in model ports

The port section of M29W160E Verilog model defines the name and the related type for each signal of the device.

Table 1: Model Ports

Port	Type	Description
A0 to A20	Input (20 downto 0)	Address Inputs
DQ0-DQ14	Input/Output (14 downto 0)	Data Input /Output
DQ15A-1	Input/Output	Data Input /Output or Address Input
BYTE	Input	Byte/Word Organization Select
E_N	Input	Chip Enable
G_N	Input	Output Enable
W_N	Input	Write Enable
RP_N	Input	Reset
RB	Input	Ready/Busy Output
Vcc	Input [35:0] ¹	Supply Voltage
Vccq	Input [35:0]	Supply Voltage

¹ VCC and VCCQ voltage signals represented with a 36-bit binary vector, whose decimal value corresponds to voltage value in millivolts.

Revision history

Table 9. Document revision history

Date	Revision	Changes
2-Nov-2009	1	Initial release

Legal Disclaimer

Please Read Carefully:

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH NUMONYX™ PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN NUMONYX'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, NUMONYX ASSUMES NO LIABILITY WHATSOEVER, AND NUMONYX DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF NUMONYX PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Numonyx products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Numonyx may make changes to specifications and product descriptions at any time, without notice.

Numonyx, B.V. may have patents or pending patent applications, trademarks, copyrights, or other intellectual property rights that relate to the presented subject matter. The furnishing of documents and other materials and information does not provide any license, express or implied, by estoppel or otherwise, to any such patents, trademarks, copyrights, or other intellectual property rights.

Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Numonyx reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Contact your local Numonyx sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Numonyx literature may be obtained by visiting Numonyx's website at <http://www.numonyx.com>.

Numonyx StrataFlash is a trademark or registered trademark of Numonyx or its subsidiaries in the United States and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2009, Numonyx, B.V., All Rights Reserved.