

NAND SLC large page memory devices VHDL model

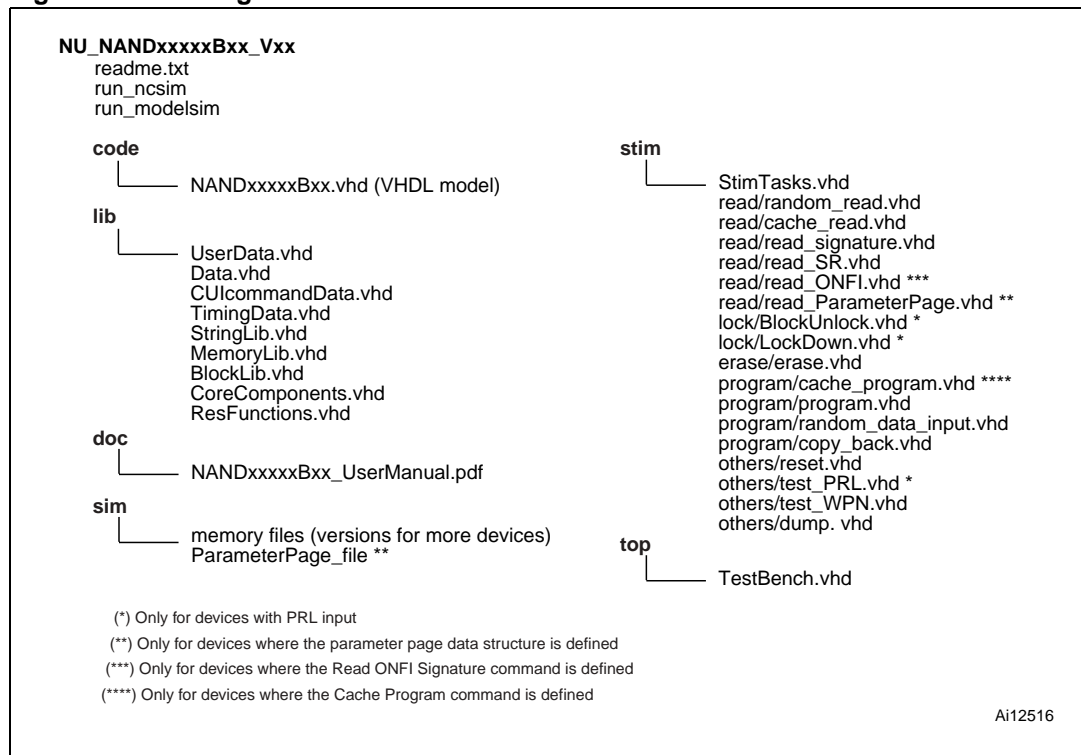
This user manual describes the VHDL behavioral model for NANDxxxxxBxx flash memory devices.

Organization of the VHDL model delivery package

The VHDL model delivery package, *NU_NANDxxxxxBxx_Vxx.zip*, is organized into a main directory, named *NU_NANDxxxxxBxx_Vxx*, containing six subdirectories with their related files (*Figure 1*).

- **code** subdirectory: contains the code source files
- **lib** subdirectory: contains the library source files
- **top** subdirectory: contains the testbench file used for simulation
- **stim** subdirectory: contains the stimuli files used for simulation
- **sim** subdirectory: contains the simulation initialization files
- **doc** subdirectory: contains model documentation

Figure 1. Package architecture



Note: See the *readme.txt* file for the complete list of files contained in each folder.

Contents

- 1 Device description 3**

- 2 VHDL behavioral model 4**
 - 2.1 Model libraries 4
 - 2.2 VHDL Testbench and Stimuli files 5

- 3 Simulation guidelines 6**
 - 3.1 Launching a simulation 6
 - 3.2 memory_file format 6
 - 3.3 Customize model and simulation 6
 - 3.4 Simulation timings 7

- 4 VHDL types used in model ports 8**

- 5 Revision history 9**

1 Device description

The NAND flash 2112-byte/1056-word page is a family of non-volatile flash memories that use NAND cell technology. The devices range from 512 Mbits to 8 Gbits and operate with either a 1.8 V or 3 V voltage supply. The size of a page is either 2112 bytes (2048 + 64 spare) or 1056 words (1024 + 32 spare) depending on whether the device has a x8 or x16 bus width.

The address lines are multiplexed with the Data Input/Output signals on a multiplexed x8 or x16 input/output bus. This interface reduces the pin count and makes it possible to migrate to other densities without changing the footprint.

Each block can be programmed and erased up to 100,000 cycles. To extend the lifetime of NAND flash devices, the implementation of an error correction code (ECC) is mandatory.

The devices have hardware and software security features:

- A write protect pin is available to provide a hardware protection against program and erase operations.
- A block locking scheme is available to provide user code and/or data protection (in devices with PRL input).

The devices feature an open-drain ready/busy output that can be used to identify if the program/erase/read controller is currently active. The use of an open-drain output allows the ready/busy pins from several memories to be connected to a single pull-up resistor.

A Copy Back Program command is available to optimize the management of defective blocks. When a page program operation fails, the data can be programmed in another page without having to resend the data to be programmed. Cache program and cache read features improve the program and read throughputs for large files.

During cache programming, the device loads the data in a cache register while the previous data is transferred to the page buffer and programmed into the memory array.

During cache reading, the device loads the data in a cache register while the previous data is transferred to the I/O buffers to be read.

All devices have the Chip Enable 'don't care' feature, which allows code to be directly downloaded by a microcontroller, as Chip Enable transitions during the latency time do not stop the read operation.

2 VHDL behavioral model

The NANDxxxxxBxx VHDL behavioral model is contained in the *NANDxxxxxBxx.vhd* file of the **code** subdirectory. It includes a set of entities that implement all the device functions listed in the device datasheet. See [Section 2.1](#) for the description of the libraries.

Please refer to *readme.txt* file in the main package directory for reference datasheet used during model development and validation.

Please check the Numonyx web site or contact your local Numonyx sales office for the most recent version of the device datasheet.

This model was validated using a Cadence NC-SIM 5.4 simulator. The use of this model with other simulators is not guaranteed.

2.1 Model libraries

The libraries in the **lib** subdirectory contains constants definitions and utility functions and tasks, used in the VHDL entities.

The **code/NANDxxxxxBxx.vhd** file and libraries files must be compiled in the same order as specified into *run_ncsim* file. The libraries files are the following:

UserData.vhd

Contains the definition of constants that can be changed by users: *DeviceName*, *memory_file*, *isDebug* and *timingCheck_on*. This is the only library that can be modified by the user and is described in [Section 3.3: Customize model and simulation](#).

Data.vhd

Contains the definition of constants, types and procedures related with memory characteristics.

CUIcommandData.vhd

Contains commands code and states of Command User Interface (CUI).

TimingData.vhd

This library contains the definition of the constants related to the timing constraints.

StringLib.vhd

This library contains the utilities used for string management.

MemoryLib.vhd

Contains general procedures for managing the memory array.

BlockLib.vhd

Contains general procedures and functions for representing and managing the blocks of the device.

CoreComponents.vhd

Contains definitions of components used in the model.

ResFunctions.vhd

Contains resolution functions used in the model.

2.2 VHDL Testbench and Stimuli files

The **top** subdirectory of the VHDL model delivery package contains a testbench file, *TestBench.vhd*. Stimuli files in VHDL 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 VHDL language.

3 Simulation guidelines

3.1 Launching a simulation

run_ncsim is an example of script that launches the Cadence NC-SIM simulation. It is located in the main directory. This file compiles and elaborates the VHDL model file and the stimuli file contained in the **stim** directory.

3.2 memory_file format

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

The format of the memory file must be as follows:

```
hex_address / hex_data
```

For example:

```
07FFFF / 7FFF
```

The user must write the filename of *memory_file* into the *UserData.vhd* library file, setting the constant called *memory_file*:

```
constant memory_file : string := "path / filename" ;
```

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

The **stim** subdirectory (which is the current path if *run_ncsim* script is used) contains some examples of memory files. Each of these examples of memory file works only with specific devices, as specified by the name of the file.

3.3 Customize model and simulation

The model and the simulation process are customizable by the user in some features and characteristics. The parameters that the user can modify are contained in the *UserData.vhd* library file.

The parameters whose value can be changed are:

- **currentDevice**: specifies which device is described by the model (each device is characterized by memory size, bus width and supply voltage); available devices are defined in the enumeration type *DeviceName* (this enumeration type is defined in *UserData.vhd* library file).
- **memory_file**: constant used for specifying the file that defines data loaded in the memory.
- **isDebug**: if this parameter is set to true value, output messages will be printed in console during simulation.
- **timingCheck_on**: if this parameter is set to True, all timing checks will be performed during the simulation. Otherwise if set to False, no timing checks will be performed during the simulation.

3.4 Simulation timings

To reduce simulation time, the user can reduce the values of certain latency times. These values can be defined by setting variables in the *TimingData.vhd* library file.

The following variables can be defined by the user:

- CACHE_READ_delay
- READ_BUSY_time
- PROGRAM_time
- ERASE_time
- RESET_time

Default values of these variables are the values specified in the datasheet, but can be reduced by the user. For example, if the user wishes to change the PROGRAM_time to 100 ns, he can redefine PROGRAM_time constant in the *TimingData.vhd* library file, as follows:

```
constant PROGRAM_time : time := 100 ns ;
```

4 VHDL types used in model ports

The port section of *NANDxxxxxBxx entity* (defined in `code/NANDxxxxxBxx.vhd` file) defines the name and the related type for each signal of the device as shown in [Table 1](#).

Table 1. Model ports

Port	Type	Description
I/O	Standard Logic (7 down to 0) Or Standard Logic (15 down to 0) ⁽¹⁾	Data Input/Output and Address Input
E_N	Standard Logic	Chip Enable
R_N	Standard Logic	Read Enable
W_N	Standard Logic	Write Enable
AL	Standard Logic	Address Latch Enable
CL	Standard Logic	Command Latch Enable
WP_N	Standard Logic	Write Protect
PRL ⁽²⁾	Standard Logic	Power-up read enable, Lock/Unlock Enable
RB_N	Standard Logic	Ready/Busy Output
V _{DD}	Real	Supply voltage
V _{SS}	Real	Ground

1. Depending on whether bus width of device is 8 or 16 bits.
2. This port is present only for the device models with PRL input.

5 Revision history

Table 2. Document revision history

Date	Revision	Changes
11-Oct-2006	1	Initial release.
30-Apr-2007	2	Updates relating to new features introduced in the version 3.3 of the model.
16-Oct-2008	3	Applied Numonyx branding. Removed Figure 2: Logic block diagram and all the references to the version of the model. Modified Figure 1: Package architecture and added a variable in Section 3.4: Simulation timings .

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 © 11/5/7, Numonyx, B.V., All Rights Reserved.