cādence®

Hardware System Verification (HSV)
Vertical Solutions Engineering (VSE)

NAND Flash
Palladium Memory Model
User Guide

Document Version: 2.3

Document Date: July 2018

Disclaimer: Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

Contents

General Information

The Cadence Memory Model Portfolio provides memory device models for the Cadence Palladium XP, Palladium XP II and Palladium Z1 series systems. Optimizing the acceleration and/or emulation flow on these platforms for MMP memory models may require information outside the scope of the MMP user guides and related MMP documentation.

1.1 Related Publications

For basic information regarding emulation and acceleration, please refer to the following documents:

For Palladium XP and Palladium XP II:

UXE User Guide
UXE Library Developer’s Guide
UXE Known Problems and Solutions
UXE Command Reference Manual
Palladium XP Planning and Installation Guide
Palladium Target System Developer’s Guide
What’s New in UXE

For Palladium Z1:

VXE User Guide
VXE Library Developer’s Guide
VXE Known Problems and Solutions
VXE Command Reference Manual
Palladium Z1 Planning and Installation Guide
Palladium Target System Developer’s Guide
What’s New in VXE

NAND Flash Palladium Memory Models

1. Introduction

The Cadence Palladium NAND Flash Models are based on the following data sheets of 1Gb, 2Gb, 4Gb, 8Gb, and 16Gb devices:

Micron 1Gb: x8/x16 NAND Flash Memory
Micron 2, 4, 8Gb: x8/x16 NAND Flash Memory
Numonyx 4Gb, 8Gb NAND flash memories
Numonyx 8Gb, 16Gb NAND flash memory
Samsung 512Mb x8 NAND Flash Memory
Samsung 2Gb x8/x16 NAND Flash Memory
Samsung 4Gb x8, 8Gb x8 NAND Flash Memory
Samsung 4Gb x8/x16, 8Gb x8/x16 NAND Flash Memory
Toshiba 2Gb x8/x16 CMOS NAND EEPROM

The models are available in several configurations with model sizes to match real devices manufactured by the following vendors listed above.

Currently only a few different sizes are available, please consult the memory model catalog for the current available list.

2. Model Release Levels

All models in the Memory Model Portfolio are graded with a release level. This release level informs users of the current maturity and status of the model. All families in the library are graded at one of these levels.

The different levels give an overall indication of the amount of testing, level of quality and feature availability in the model. For details on supported features check the User Guide for that particular model family.

There are three release levels for models in the MMP release.

Release Level		Model Status	Available in Release	Listed in Catalog	Requires Beta Agreement
Mainstream Release	MR	Fully released and available in the catalog for all customers to use.	Yes	Yes	No
Emerging Release	ER	Model has successfully completed Beta engagement(s). Most, but not all features have been tested. Documentation is available.	No	Yes	Yes
Initial Release	IR	Model has completed initial development and has been released to Beta customer(s). The model may have missing features, may not be fully tested and may not have documentation. Model may contain defects.	No	Yes	Yes

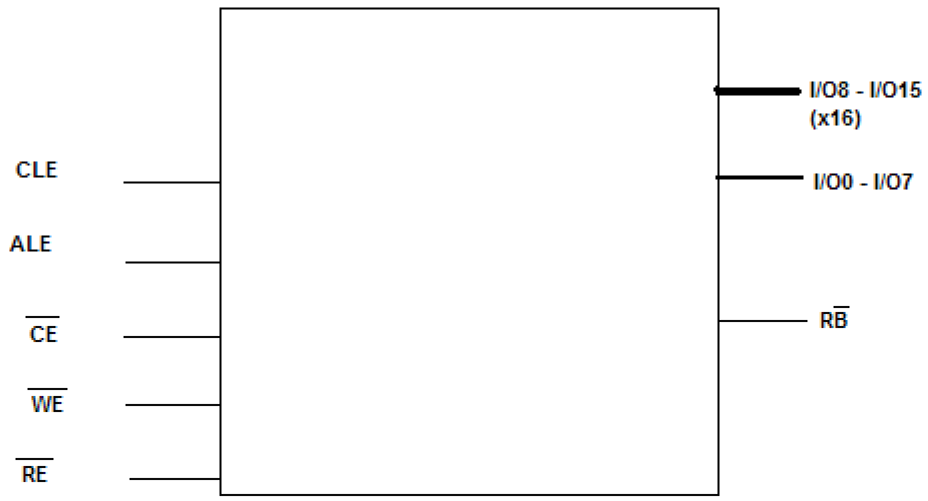
Access to Initial Release and Emerging Release versions of the models will require a Beta Agreement to be signed before the model can be delivered.

3. Configurations

The following table lists the configurations specified in the data sheet listed above. It is possible that not all configurations are available from all vendors. Please consult the appropriate vendor site for details on the parts they offer.

Model number	Density	I/O Bus Width	Page size	Block size in pages	Blocks in real part	Blocks in MMP model	Vendor
mt29f1g08a	1Gb	8	2112	64	1024	1024	Micron
mt29f1g16a	1Gb	16	1056	64	1024	1024	Micron
mt29f2g08	2Gb	8	2112	64	2048	32	Micron
mt29f2g16	2Gb	16	1056	64	2048	32	Micron
mt29f4g08	4Gb	8	2112	64	4096	32	Micron
mt29f4g16	4Gb	16	1056	64	4096	32	Micron
mt29f8g08	8Gb	8	2112	64	8192	32	Micron
mt29f8g16	8Gb	16	1056	64	8192	32	Micron
nand04gw3f2e	4Gb	8	4224	64	2048	32	Numonyx
nand04gw4f2e	4Gb	16	2112	64	2048	32	Numonyx
nand08gw3f2e	8Gb	8	4224	64	4096	32	Numonyx
nand08gw4f2e	8Gb	16	2112	64	4096	32	Numonyx
nand08gw3d2a	8Gb	8	4224	128	2048	32	Numonyx
nand16gw3d2a	16Gb	8	4224	128	4096	32	Numonyx
k9f1g08u0m	1Gb	8	2112	64	1024	1024	Samsung
K9f1208r0b	512Mb	8	528	32	4096	4096	Samsung
k9f2g08u0m	2Gb	8	2112	64	2048	32	Samsung
k9f2g16u0m	2Gb	16	1056	64	2048	32	Samsung
k9f4g08u0m	4Gb	8	2112	64	4096	32	Samsung
k9k8g08u1m	8Gb	8	2112	64	8192	32	Samsung
kf94g08q	4Gb	8	4224	64	2048	32	Samsung
kf94g16q	4Gb	16	2112	64	2048	32	Samsung
kf88g08q	8Gb	8	4224	64	4096	32	Samsung
kf88g16q	8Gb	16	2112	64	4096	32	Samsung
K9g8g08u0c	8Gb	8	8628	128	1038	32	Samsung
tc58nvg1s3b	2Gb	8	2112	64	2048	32	Toshiba
tc58nvg1s8b	2Gb	16	1056	64	2048	32	Toshiba

4. Model Logic Diagram



Note: Commands and addresses are input on I/O[7:0] only. For devices with a x16 interface, I/O[15:8] must be written with zeros when issuing a command or an address.

5. Address Mapping

The array of the NAND Flash model is mapped into the internal memory of the Palladium system. This array is a single two dimensional array. The mapping of block, page and column addresses to the internal model array is as follows:

ARRAY_ADDR = {BA, PA, CA}

6. Spare Area and Preloading memory array

There is spare area available in each page. For example in a page with size 2112 the main data area is 2048 and the spare area is 64, and for page size of 4224 the main data area is 4096 and the spare area is 128. Due to the spare area the memory becomes fragmented. In the example the memory address range for the first page is 0 to 2111. The second page starts at 4096, instead of 2112. Addresses 2112 to 4095 do not exist in the device. Please adjust the addresses in the data file accordingly when preloading the memory. Here is a data file example:

```
//==== block 0
```

```
//  Page 0 data
@0 0f 0e 0d 0c 0b 0a f9 f8 f7 f6 f5 f4 f3 f2 f1 f0
// Page 0 Spare
@800 40 41 42 43 44 45 46 47 48 49 4a

// Page 1 data
@1000 50 51 52 53 54 55 56 57 58 59 51 5b 5c 5d 5e 5f
// Page 1 Spare
@1800 80 81 82 83 84 85
```

Use one of the following example commands to load or dump the memory content:

```
    memory –load %readmemh mem_model_inst.i0.mem –file init.dat
    memory –dump %pd_h mem_model_inst.i0.mem –file mem.dat
```

The following table lists the internal memory array name or path to use for loading data.

Model number	Path within model instance
mt29f1g08a	L11.mem_array, L11.param_page
mt29f1g16a	L11.mem_array, L11.param_page
mt29f2g08	L11.mem_array
mt29f2g16	L11.mem_array
mt29f4g08	L11.mem_array
mt29f4g16	L11.mem_array
mt29f8g08	L11.mem_array, L12.mem_array
mt29f8g16	L11.mem_array, L12.mem_array
nand04gw3f2e	L0.mem_array, L0.param_page
nand04gw4f2e	L0.mem_array, L0.param_page
nand08gw3f2e	L0.mem_array, L1.mem_array, L0.param_page, L1.param_page
nand08gw4f2e	L0.mem_array, L1.mem_array, L0.param_page, L1.param_page
nand08gw3d2a	mem_array
nand16gw3d2a	mem_array
k9f1g08u0m	L11.mem_array
K9f1208r0b	L11.mem_array
k9f2g08u0m	i0.mem
k9f2g16u0m	i0.mem
k9f4g08u0m	L0.mem_array
k9k8g08u1m	L0.mem_array, L1.mem_array
kf94g08q	mem
kf94g16q	mem
kf88g08q	mem
kf88g16q	mem
K9g8g08u0c	L0.mem_array
tc58nvg1s3b	i0.mem
tc58nvg1s8b	i0.mem

7. ID Operation

7.1. ID Read

The Manufacturer and Device codes have been hardcoded into each model. Therefore user data file is not required.

7.2. Read ONFI Signature and Parameter Page

Some of the NAND flash models support ONFI 1.0 commands.

8. Commands

The NAND flash models accept the following commands. However some models may not support all the commands. Obviously single plane devices do not support multiplane commands. Please check each device's data sheet for supported commands and model limitations listed in section 11 below and in each model's header.

- Page Read
- Read for Copy Back
- Read ID
- Read1
- Read2
- Reset
- Page Program
- Multiplane Page Program
- Multiplane Read
- Copy Back Program
- Multiplane Copy Back Program
- Multiplane Copy Back Read
- Block Erase
- Multiplane Block Erase
- Read Status Register
- Random Data Input
- Random Data Output
- Multiplane Random Data Output
- Page Read Cache Mode
- Program for Internal Data Move
- Program Page Cache Mode
- Read for Internal Data Move
- Read Parameter Page

Status register read during program and erase operations is supported. The R/B# pin can be monitored for Ready/Busy status.

9. Initialization Sequence

The NAND Flash model does not require a special initialization sequence. Issuing the Reset command first is recommended. When R/B# is high the model is ready for normal operation.

10. Model Size

For models larger than 1Gb, to reduce memory utilization each model size is only 32 blocks. If larger size is needed please contact Customer Support. No size reduction in models of size 1Gb or less.

11. Limitations

1. Error detection and correction are not supported. See Section 14 MMP and ECC (Error Correcting Code) below for additional information and potential work-arounds.
2. Block Lock, OTP, and Write protection are not supported.
3. Model does not check illegal sequence of command cycles.
4. Model does not support timing parameters, except tRHOH.
5. The time to program a page or erase a block is 1 fclk (fast clock) per address location.

12. Timing Parameter tRHOH

The model does not generally support timing parameters because it is cycle based. However tRHOH (an asynchronous mode parameter that controls RE# HIGH to output hold) is supported in number of fclks (fast clocks). The user may define a macro at compile time to hold the data by a number of fclks after RE# goes high. Use this macro only if data output is not already held long enough. The model holds the data valid for 2 fclks by default. The data can be held 3 additional fclks by using the following example command line option in ixcom flow:

```
vlan +define+tRHOH=3
```

13. Emulation

The memory models are currently provided in one format: an encrypted RTL file(s) (*.vp) that targets use in the IXCOM flows or in the ICE flow. The encrypted RTL (*.vp) file(s) must be synthesized along with other design code prior to acceleration / emulation.

An example of the command for compilation (including synthesis) of this model in IXCOM flow is shown below:

```
ixcom -64bit +sv -ua +dut+nand04gw3f2e \
../tb.v \
../src/nand04gw3f2e.vp \
-incdir ../../../utils/cdn_mmp_utils/sv \
../../../utils/cdn_mmp_utils/sv/cdn_mmp_utils.sv \
……

xeDebug -64 --ncsim \
-sv_lib ../../../utils/cdn_mmp_utils/lib/64bit/libMMP_utils.so -- \
-input auto_xedebug.tcl
```

Note that +sv switch is needed.

ICE flow synthesis commands:

```
vavlog ../src/ nand04gw3f2e.vp

vaelab –keepRtlSymbol –keepAllFlipFlop –outputVlog nand04gw3f2e.vgp
nand04gw3f2e
```

NOTE: It is common for Palladium flows to require –keepallFlipFlop since it removes optimizations that are in place by default. For example, without –keepAllFlipFlop, HDL-ICE can remove flops with constant inputs and merge equivalent FF. The picture above is modified a bit when ICE ATB mode (–atb) is used since then a constant input FF is only optimized out when there is no initial value for it or the initial value is the same as the constant input value.

It is also common for Palladium flows to require –keepRtlSymbol. This option enables the HDL Compiler to keep original VHDL RTL symbols, such as ".", whenever possible. In other words, it maps VHDL RTL signal name a.b to the netlist entry, \a.b. Without this modifier, the signal name would otherwise be converted to a_b in the netlist.

If the recommended compile script includes the aforementioned options, the user must include them to avoid affecting functionality of the design.

13.1. Model file list

<model_name>.vp - model file.

14. MMP and ECC (Error Correcting Code)

MMP models do not support Error Correcting Code (ECC) functionality. ECC functions, if they are present in a memory device, are typically found in the NAND and DDRx families. The MMP product does not have any plans to provide such functions in the models. MMP models are provided as system level emulation models and not as verification IP. The below sections discuss work-arounds that enable the user to deal with some ECC scenarios. Note that ECC means different things to different device families.

14.1. NAND FLASH and ECC (Error Correcting Code)

MMP NAND models do not support Error Correcting Code (ECC) functionality. There will not be an ECC error in a Palladium MMP flash model; the data stored in a MMP model should not need to be corrected because the model does not degrade over time like the real device. The data returned is always correct. The paragraphs below provide details about host ECC in relation to MMP NAND Flash.

For NAND Flash devices, ECC means that the internal engine in the Flash device calculates the ECC when programming and writes the resulting value into the spare array. The low level details of this operation are in the device specification. The Flash then re-calculates the ECC on reads and compares with the value stored in the spare array. If non-equivalence is found, bit error is indicated, and the device corrects and/or flags an error. There are several cases:

- If the controller relies on ECC generation internal to the Flash device, then the model needs to do nothing. This has worked for all users so far.
  - To support this scenario, model parameters can be modified to indicate that ECC is enabled. The controller is then happy. NOTE: the model will NOT actually do the ECC calculation.

- If the controller uses its own ECC and manually writes to the spare array in the device, then again the MMP model does not need to do anything.
  - There is a spare area is implemented in the NAND model for the host to store ECCs. This spare area allows the host to do data correction.

- If the controller relies on ECC generation internal to the memory device AND the controller reads and examines the spare calculation itself, then the MMP model will not work.
  - There is no MMP plan to enhance NAND FLASH MMP models to support this case. It is a large effort.

Occasionally, an issue may be seen due to the parameter page setting for the available number of bits of ECC correction. According to the ONFI standard this setting is handled by byte 112 of the parameter page. See the figure below for an example entry from the standard. Problems may occur with some controllers when byte 112 of the parameter page is set to the value ‘0’. If the controller requires some positive value for the *Number of bits ECC correctability*, then the user may need to change the setting to a value of ‘1’.

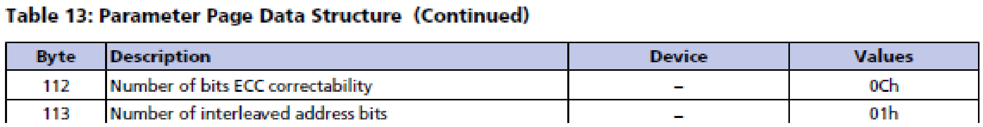


Table 13: Parameter Page Data Structure (Continued)

Byte	Description	Device	Values
112	Number of bits ECC correctability	–	0Ch
113	Number of interleaved address bits	–	01h

Revision History

The following table shows the revision history for this document

Date	Version	Revision
May 2011	1.0	Initial version
June 2011	1.1	Updated with “Related Documents” information
January 2013	1.2	Renaming
July 2014	1.3	Repaired doc title property. Added revision history. Updated legal. Added a missing model number to tables.
September 2014	1.4	Remove version from UG file name. Update UXE / IXE documentation reference titles.
November 2014	1.5	Remove emulation capacity info.
January 2015	1.6	Added MMP and ECC section. Update related publications list.
March 2015	1.7	Add timing parameter limitation and a section on tRHOH.
July 2015	1.8	Update Cadence naming on front page
September 2015	1.9	Added Emulation section with compile examples. Removed ‘Model does not check attempts to program a bit to 1’ from Limitations list.
January 2016	2.0	Update for Palladium-Z1 and VXE
July 2016	2.1	Remove hyphen in Palladium naming
January 2018	2.2	Modify header and footer
July 2018	2.3	Update for new utility library