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

Toggle DDR NAND Flash
Palladium Memory Model
User Guide

Document Version: 2.0

Document Date: July 2018

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

Toggle DDR NAND Flash Palladium Memory Models

1. Introduction

The Cadence Palladium Toggle DDR NAND Flash Models are based on data sheet specifications of the following Samsung devices:

32Gb M-die Toggle NAND Flash with MLC technology
16Gb M-die Toggle NAND Flash with SLC technology

Please note that these models support the Toggle Mode DDR interface only, even though the data sheet mentions Legacy Asynchronous SDR and Synchronous DDR also. Please check the ID Definition Tables in the data sheet for more details.

The models are available in several configurations with model sizes to match real devices manufactured by the following vendor: Samsung.

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, may not have documentation. Model may contain defects.	No	Yes	Yes

Access to Initial 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 two data sheets 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	Density	Level	# of Die	# of CE#	# of R/B#	Interface
K9LCGD8X1M	**64Gb**	**MLC**	**2**	**2**	**2**	**Toggle DDR**
K9HDGD8X5M	**128Gb**	**MLC**	**4**	**4**	**4**	**Toggle DDR**
K9PFGD8X7M	**256Gb**	**MLC**	**8**	**8**	**4**	**Toggle DDR**
K9PFGD8X5M	**256Gb**	**MLC**	**8**	**4**	**4**	**Toggle DDR**
K9KBGD8X1M	**32Gb**	**SLC**	**2**	**2**	**2**	**Toggle DDR**
K9WCGD8X5M	**64Gb**	**SLC**	**4**	**4**	**4**	**Toggle DDR**
K9QDGD8X5M	**128Gb**	**SLC**	**8**	**4**	**4**	**Toggle DDR**

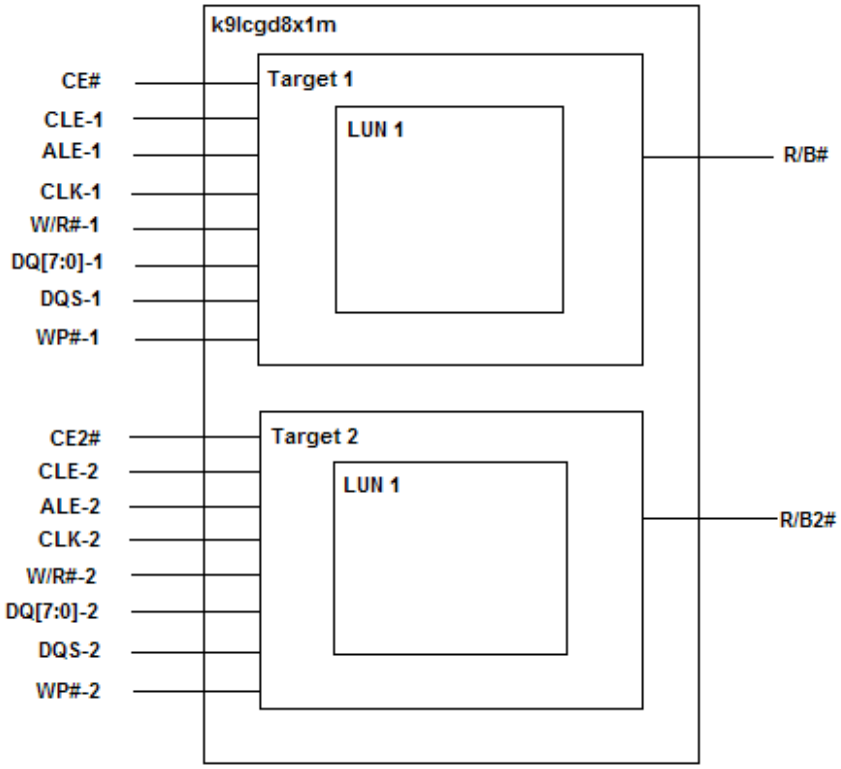
Notes: MLC = 2bits/cell
SLC = 1bit/cell
All 7 models have 2 sets of I/O pins
(pin set: ALE,CLE,DQ,DQS,RE#,WE#,WP#)
Chip, Die and LUN are used interchangeably
Target and CE# are used interchangeably
A target may have one or two die

4. Model Block Diagram

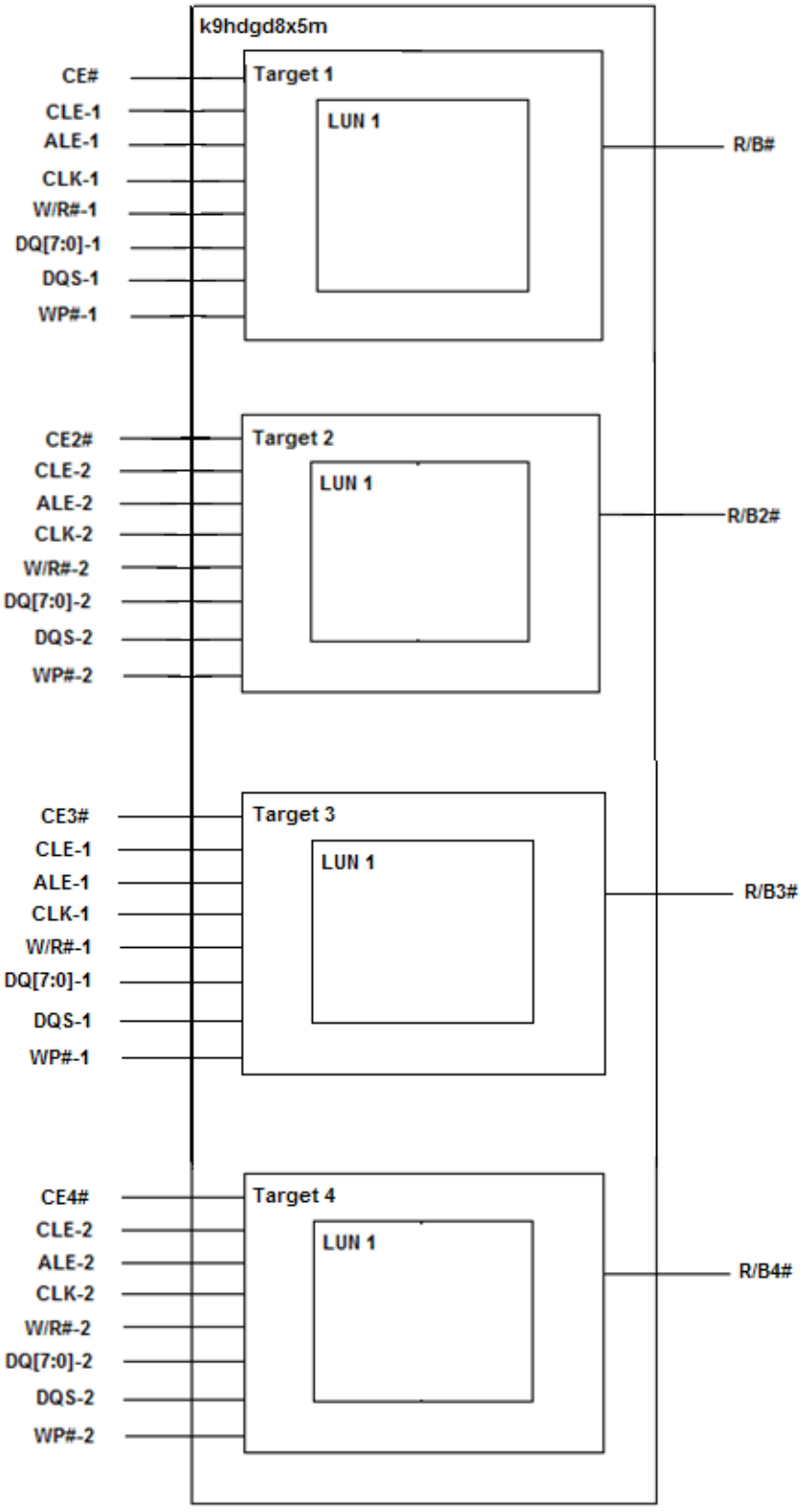
These models are implemented modularly based on the die core, which is instantiated as many times as needed within each model. The user does not need to instantiate the core directly. The user instantiate the model based on the model's I/O pin sets.

Block diagrams of various models are shown below.
For models that have 2 CE#s, Target 1 uses pin set 1, Target 2 uses pin set 2.
For models that have 4 CE#s, Targets 1 and 3 share pin set 1, Targets 2 and 4 share pin set 2.
For models that have 8 CE#s, Targets 1, 3,5,7 share pin set 1, Targets 2,4,6,8 share pin set 2.
For model k9pfgd8x7m, Targets 1 and 5 share R/B# 1, Targets 2 and 6 share R/B# 2, Targets 3 and 7 share R/B# 3, Targets 4 and 8 share R/B# 4.

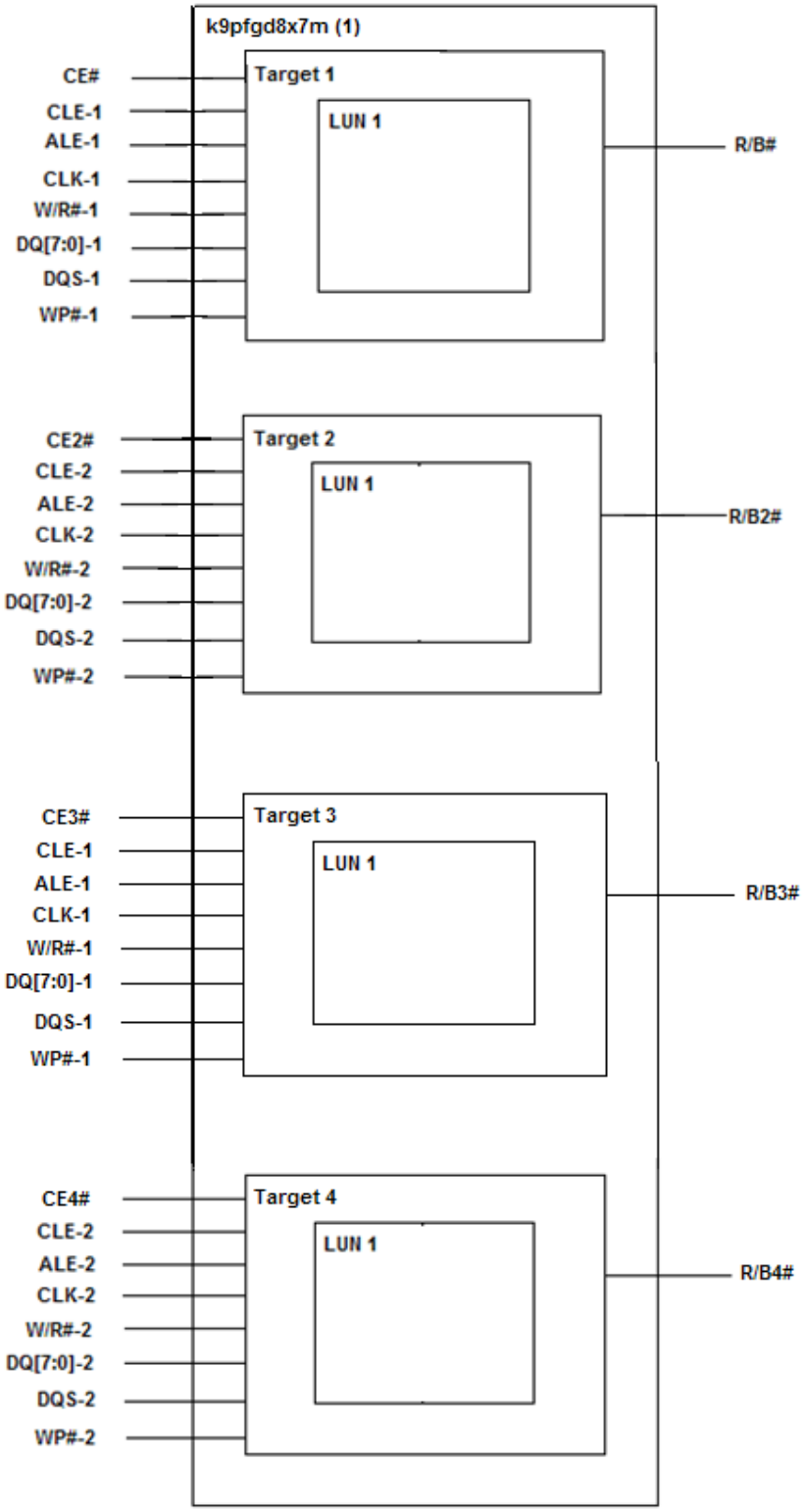
The following block diagram shows the k9lcgd8x1m model which has 2 LUNs, 2 CE#s, 2 R/B#s and two sets of I/O pins.

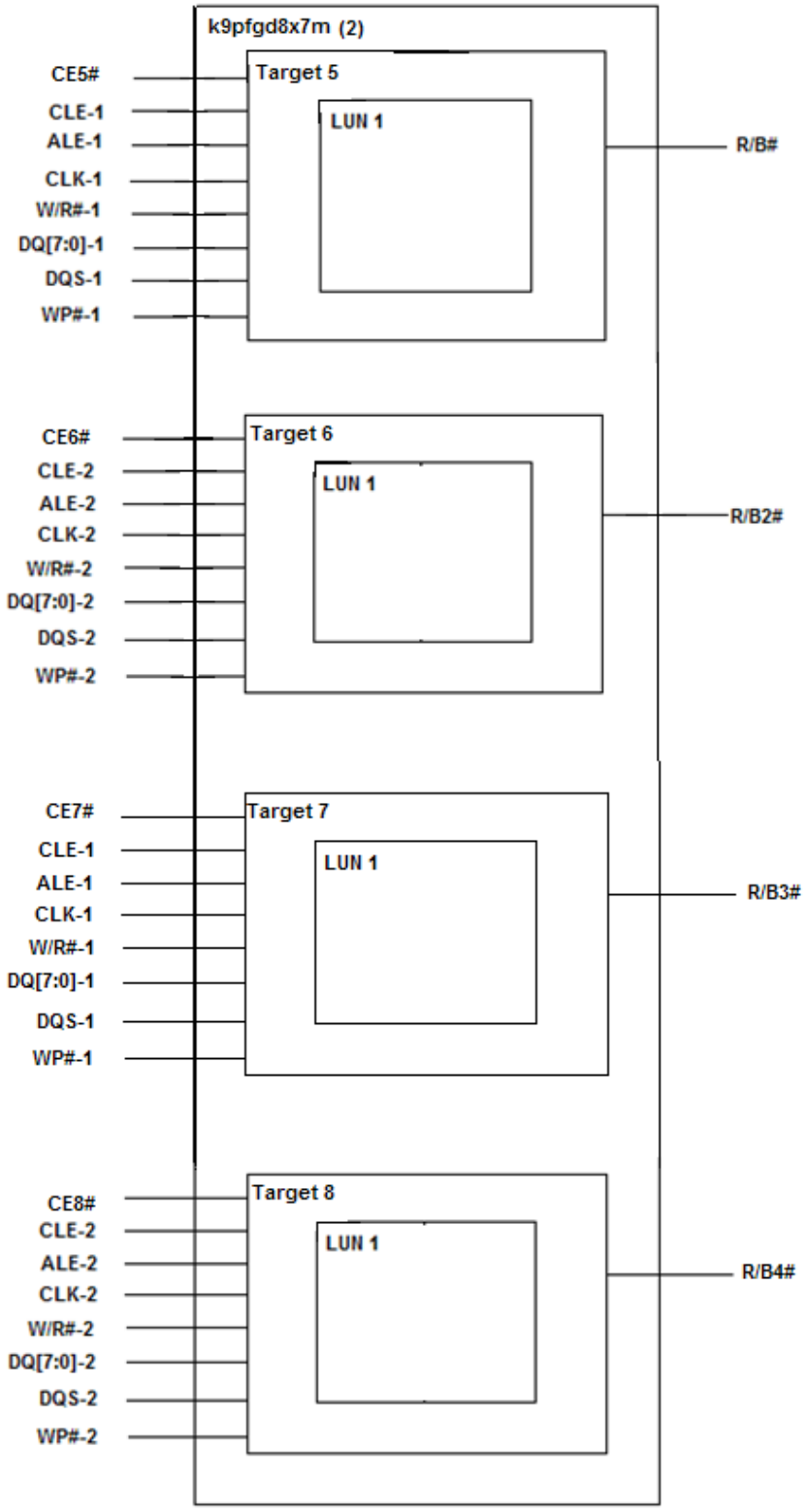


The following block diagram shows the k9hdgd8x5m model which has 4 LUNs, 4 CE#s, 2 R/B#s and two sets of I/O pins.

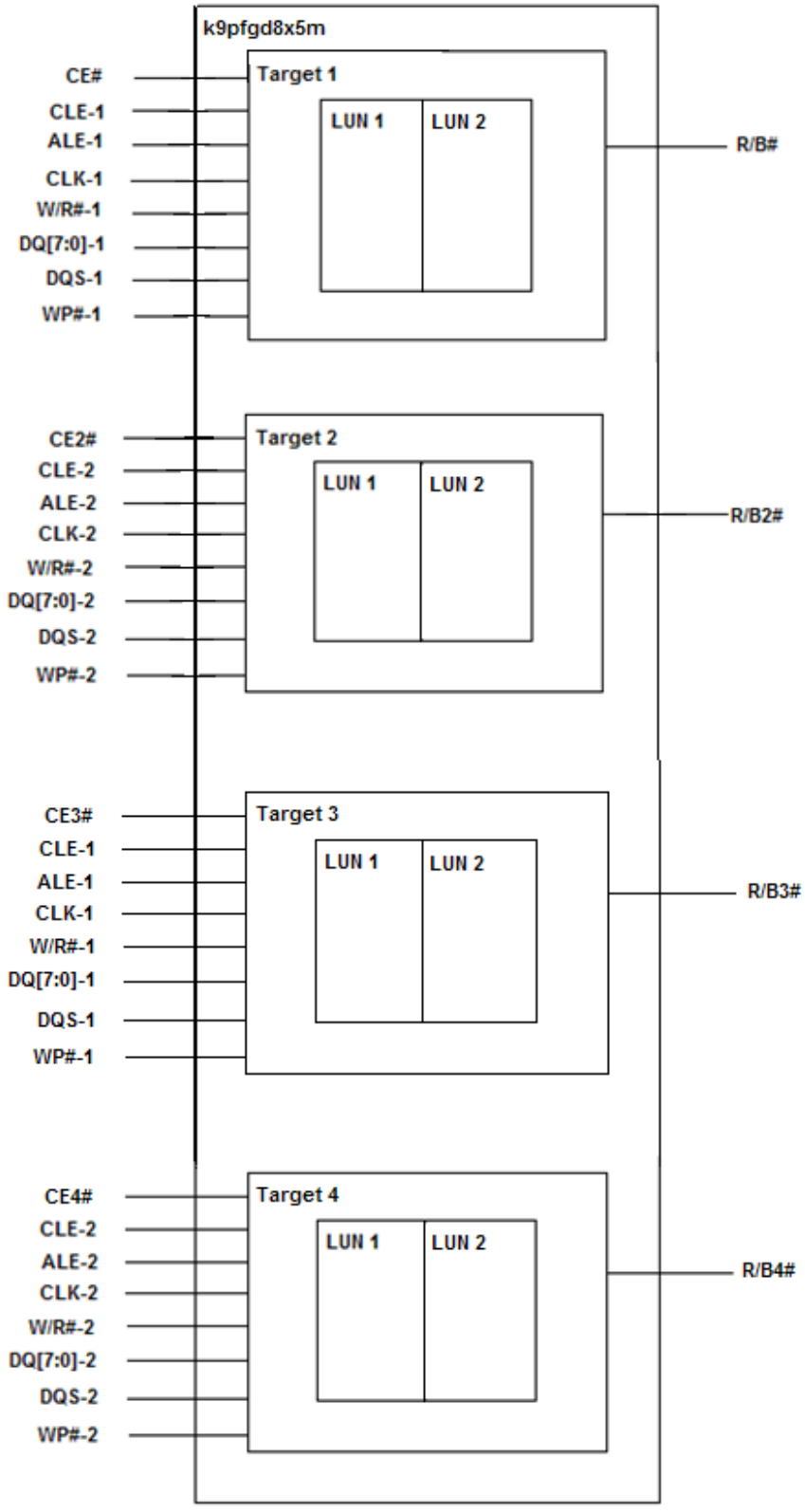


The following two-part block diagram shows the k9pfgd8x7m model which has 8 LUNs, 8 CE#s, 4 R/B#s and two sets of I/O pins.





The following block diagram shows the k9pfgd8x5m model which has 8 LUNs, 4 CE#s, 4 R/B#s and two sets of I/O pins.



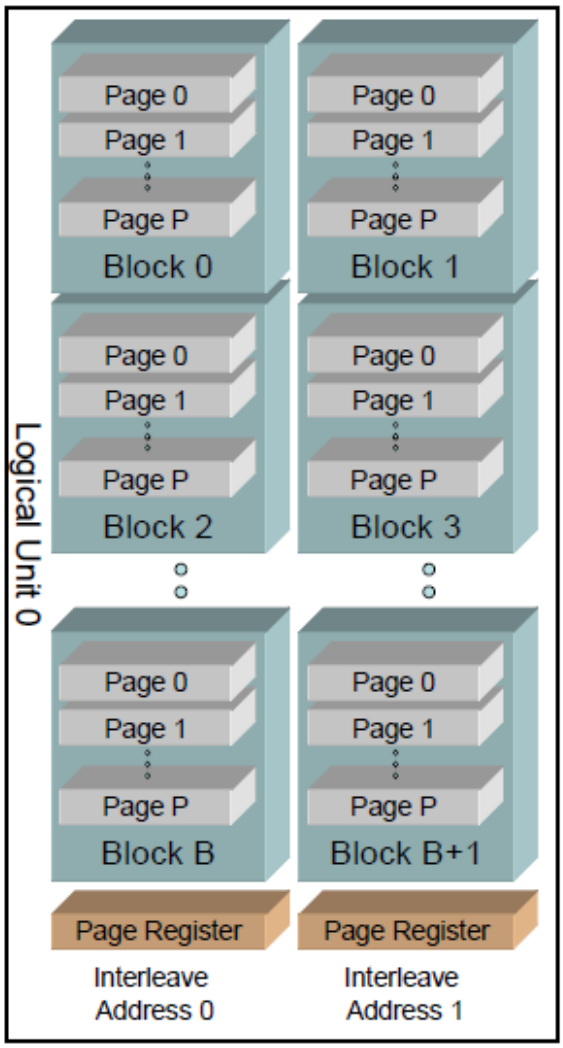
5. Address mapping

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 lun, block, page and column addresses to the internal model array is as follows:

ARRAY_ADDR = { LA, BA, PA, CA}

This information is required if the memory needs to be preloaded with user data. For models with only 1 die, LA should be set to 0. Here are the array organization and addressing cycle table for MLC models, followed by SLC models. Note that SLC models have one less page address bit.

Array Organization for MLC Array



1 Page = (8K + 512)Bytes
1 Block = (8K + 512)Bytes x 128 Pages
= (1M + 64k)Bytes
1 Device = (8K + 512)Bytes x 128 Pages x 4152 Blocks
= 36,139,008 Kbits

Plane 0 contains blocks 0, 2, 4, ..., 4150
Plane 1 contains blocks 1, 3, 5, ..., 4151

Blocks 0 to 4095 are main blocks
Blocks 4096 to 4151 are spare blocks

MLC Array Organization per Logical Unit (LUN) or die.
credit: diagram taken from ONFI 2.0 Specification
dated 27-Feb-2008 page 32 figure 16

Address Cycle Table for MLC Array

Cycle	I/O 0	I/O 1	I/O 2	I/O 3	I/O 4	I/O 5	I/O 6	I/O 7
First	A0	A1	A2	A3	A4	A5	A6	A7
Second	A8	A9	A10	A11	A12	A13	Low	Low
Third	A14	A15	A16	A17	A18	A19	A20	A21
Fourth	A22	A23	A24	A25	A26	A27	A28	A29
Fifth	A30	A31	A32	A33	A34	Low	Low	Low

Notes from Samsung data sheet:
A0 ~ A13 = column address, A14 ~ A20 = page address, A21~ A33 = block address, A34 = LUN or Chip address. The page address, block address, and LUN address are collectively called the row address.

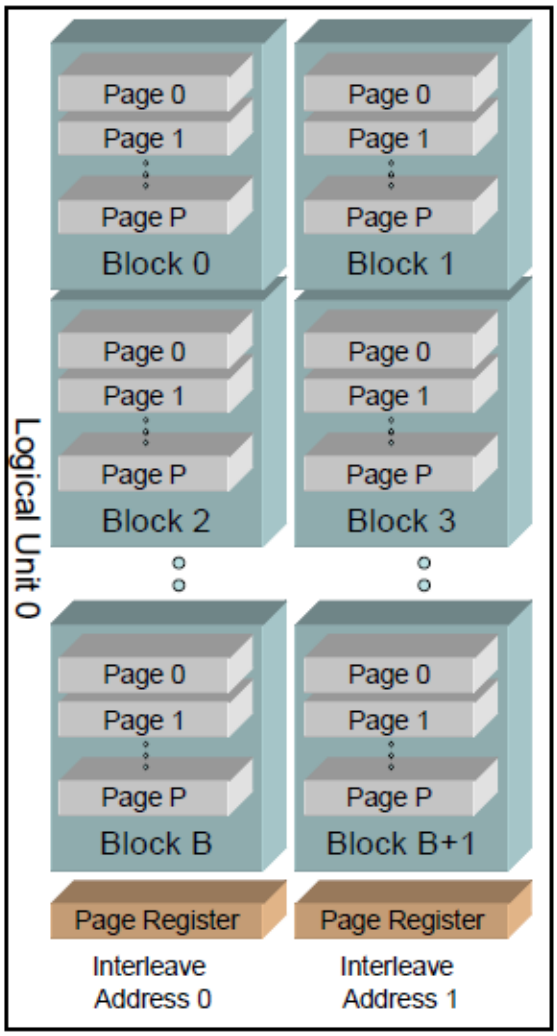
When using the Toggle DDR interface, A0 is forced to 0 internally; one data cycle always returns one even byte and one odd byte.

Column addresses 8704 (2200h) through 16383 (3FFFh) are invalid, out of bounds, do not exist in the device, and cannot be addressed.

A21 are the plane-select bits:
Plane 0: A21 = 0
Plane 1: A21 = 1

A34 is the LUN-select bit. It is present only when two LUNs are shared on the target; otherwise, it should be held LOW.
LUN 1: A34 = 0
LUN 2: A34 = 1

Array Organization for SLC Array



1 Page = (8K + 640)Bytes
1 Block = (8K +640)Bytes x 64 Pages
+ (512K + 40K)Bytes
1 Device = (8K + 640)Bytes x 64 pages x 4152 Blocks
= 18,069,504 Kbits

Plane 0 contains blocks 0, 2, 4, ..., 4150
Plane 1 contains blocks 1, 3, 5, ..., 4151

Blocks 0 to 4095 are main blocks
Blocks 4096 to 4151 are spare blocks

SLC Array Organization per Logical Unit (LUN) or die.
credit: diagram take from ONFI 2.0 Specification
dated 27-Feb-2008 page 32 figure 16

Address Cycle Table for SLC Array

Cycle	I/O 0	I/O 1	I/O 2	I/O 3	I/O 4	I/O 5	I/O 6	I/O 7
First	A0	A1	A2	A3	A4	A5	A6	A7
Second	A8	A9	A10	A11	A12	A13	Low	Low
Third	A14	A15	A16	A17	A18	A19	A20	A21
Fourth	A22	A23	A24	A25	A26	A27	A28	A29
Fifth	A30	A31	A32	A33	Low	Low	Low	Low

Notes from Samsung data sheet:
A0 ~ A13 = column address, A14 ~ A19 = page address, A20 ~ A32 = block address, A33 = LUN or Chip address. A20 is the plane address also. The page address, block address, and LUN address are collectively called the row address.

When using the Toggle DDR interface, A0 is forced to 0 internally; one data cycle always returns one even byte and one odd byte.

Column addresses 8832 (2280h) through 16383 (3FFFh) are invalid, out of bounds, do not exist in the device, and cannot be addressed.

A20 are the plane-select bits:
Plane 0: A20 = 0
Plane 1: A20 = 1

A33 is the LUN-select bit. It is present only when two LUNs are shared on the target; otherwise, it should be held LOW.
LUN 1: A33 = 0
LUN 2: A33 = 1

6. Feature Address Definition

Since the feature address is 8 bits there can be up to 256 feature addresses defined – a 256 x 32 array. However signal drive strength at address 10h is the only parameter that can be set by the SET FEATURE (EFh) command, therefore this command is not supported in the Palladium models. The following table shows the driver strength definitions as described in the Samsung data sheet.

B0 Value	Driver Strength
00h~01h	Reserved
02h	Driver Multiplier : underdriver1
03h	Reserved
04h	Driver Multiplier : 1 (Default)
05h	Reserved
06h	Driver Multiplier : overdriver1
07h	Reserved
08h	Driver Multiplier : overdriver2
09h	Reserved
0Ah ~ FFh	Reserved

7. ID Operations

7.1. READ ID

The READ ID parameters for addresses 00h and 40h have been hardcoded into each model. Therefore user data file is not required.

7.2. READ Device ID Table

The data for the device id table is provided in the <model_name>.dat file. This data file should be preloaded into all LUNs in the model if the user wants to read Device ID information from the model. The instance names are L<CE><LUN>. Using the k9qdgd8x5m model as an example, there are 4 CEs and 2 LUNs per CE. The path to each LUN's device id table or parameter page is as follows:

<path.to.model.inst>.L11.param_page
<path.to.model.inst>.L12.param_page
<path.to.model.inst>.L21.param_page
<path.to.model.inst>.L22.param_page
<path.to.model.inst>.L31.param_page
<path.to.model.inst>.L32.param_page
<path.to.model.inst>.L41.param_page
<path.to.model.inst>.L42.param_page

8. Commands

The NAND flash model accepts the following commands:

- Read
- Read for Copy-Back
- Read ID
- Device Identification Table Read
- Reset
- Page Program
- Copy-Back Program
- Block Erase
- Random Data Input
- Random Data Output
- Read Status
- Chip1 Status
- Chip2 Status
- Two-Plane Read
- Two-Plane Read for Copy-Back
- Two-Plane Random Data Output
- Two-Plane Page Program
- Two-Plane Copy-Back Program
- Two-Plane Block Erase

The following table shows the command set as described in the Samsung data sheet.

Function	1st Set	2nd Set	Acceptable Command during busy	Notes
Read	00h	30h		
Read for Copy-Back	00h	35h		
Read ID	90h	-		1
Device Identification Table Read	ECh	-		
Reset	FFh	-	Yes	
Page Program	80h	10h		
Copy-Back Program	85h	10h		
Block Erase	60h	D0h		
Random Data Input	85h	-		2
Random Data Output	05h	E0h		2
Read Status	70h	-	Yes	
Chip1 Status	F1h	-	Yes	
Chip2 Status	F2h	-	Yes	4
Set Feature	EFh	-		5
Get Feature	EEh	-		5, 6
Two-Plane Read	60h-60h	30h		
Two-Plane Read for Copy-Back	60h-60h	35h		
Two-Plane Random Data Output	00h-05h	E0h		
Two-Plane Page Program	80h-11h	81h-10h		3
Two-Plane Copy-Back Program	85h-11h	81h-10h		3
Two-Plane Block Erase	60h-60h	D0h		

Notes from Samsung data sheet:

1. Two sets of ID bytes are defined depending on following address which is either 00h (Samsung Legacy) or 40h (JEDEC) after Read ID.
2. Random Data Input/Output can be executed in a page.
3. Any command between 11h and 80h/81h/85h is prohibited except 70h/F1h/F2h and FFh.
4. Chip2 Status Command is supported by k9pfgd8x5m and k9qdgd8x5m (models with 2 LUNs per target) only.

Notes specific to Palladium models:

5. These commands are not supported by the Palladium models.
6. This command is listed in the SLC data sheet only.

9. 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.

9.1. 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

10. Compile and Emulation

The model is provided as a protected RTL file(s) (*.vp). The file(s) need to be synthesized prior to the back-end Palladium compilation. An example of the command for compilation (including synthesis) and run of this model in the IXCOM flow is shown below.

```
ixcom -64 +sv -ua +dut+<model_name> \
    ./<model_name>.vp \
```

```
    ./k9e_32.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
```

The script below shows two examples for Palladium classic ICE synthesis:

```
1)
hdlInputFile <model_name>.vp
hdlImport -full -2001 -l qtref
hdlOutputFile -add -f verilog <model_name>.vg
hdlSynthesize -memory -keepVhdlCase -keepRtlSymbol -keepAllFlipFlop
<model_name>
……

2)
vavlog <model_name>.vp

vaelab -keepRtlSymbol -keepAllFlipFlop -outputVlog <model_name>.vg
<model_name>
……
```

NOTE: It is common for Palladium flows to require –keepallFlipFlop since it removes optimizations which 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.

11. Initialization Sequence

The NAND Flash model requires that the memory controller follows the initialization sequence as documented in the specification. The sequence basically entails the following steps:

1. The RESET (FFh) command must be the first command issued to all targets (CE#s). The RESET busy time can be monitored by polling R/B#.
2. When R/B# is high the model is now initialized and ready for normal operation.

The model requires that these steps are performed in the correct sequence in order to complete initialization. The model will not respond to any others commands unless this sequence is completed.

12. Model Size

To reduce memory utilization each LUN has only 32 blocks but the actual device has 4152 blocks. Each MLC block is 1 MB or 8 Mb. If larger size is needed please contact Customer Support.

13. Limitations

1. Set Feature and Get Feature commands are not supported.
2. Model does not check illegal sequence of command cycles.
3. Model does not check for attempts to program a bit to 1, user should make sure the block is erased before program.

Revision History

The following table shows the revision history for this document

Date	Version	Revision
July 2010	1.0	Initial version
June 2011	1.1	Updated with “Related Documents” information
July 2014	1.2	Repaired doc title property. Added revision history. Updated legal.
September 2014	1.3	Remove version from UG file name. Update UXE / IXE documentation reference titles.
November 2014	1.4	Remove emulation capacity info. Update related publications list.
July 2015	1.5	Update Cadence naming on front page
January 2016	1.6	Update for Palladium-Z1 and VXE
July 2016	1.7	Remove hyphen in Palladium naming
January 2018	1.8	Modify header and footer
February 2018	1.9	Ported ECC section from NAND UG to ONFI and TDDR UGs
July 2018	2.0	Update for new utility library