cādence®

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

GDDR5X
Palladium Memory Model
User Guide

Document Version: 1.9

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 4

1.1 RELATED PUBLICATIONS 4

GDDR5X MEMORY MODEL 5

1. INTRODUCTION 5
2. MODEL RELEASE LEVELS 6
3. CONFIGURATIONS 7
4. FEATURES 8
5. MODEL PARAMETER DESCRIPTIONS 9
6. MODEL BLOCK DIAGRAM 9
7. ADDRESS MAPPING 11
8. REGISTER DEFINITIONS 12
   - *8.1. Mode Register 0 .... 12*
   - *8.2. Mode Register 1 .... 13*
   - *8.3. Mode Register 2 .... 13*
   - *8.4. Mode Register 3 .... 14*
   - *8.5. Mode Register 4 .... 14*
   - *8.6. Mode Register 5 .... 15*
   - *8.7. Mode Register 6 .... 15*
   - *8.8. Mode Register 7 .... 15*
   - *8.9. Mode Register 8 .... 15*
   - *8.10. Mode Register 15 .... 16*
9. COMMANDS 16
10. INITIALIZATION SEQUENCE 17
11. SYNTHESIS AND COMPILATION OF THE MODEL 17
12. LIMITATIONS 18
13. QDR READ AND WRITE 20
14. HANDLING DQS IN PALLADIUM MEMORY MODELS 20
15. REVISION HISTORY 23

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

GDDR5X Memory Model

1. Introduction

The Cadence Palladium GDDR5X x16 and x32 models are based on the GDDR5X SGRAM Specification (V0.3) committee item number 1827.99

There are ten configurations with size ranging from 4Gb to 16Gb.

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 possible configurations. Not all configurations are available from all vendors. Please consult the appropriate vendor site for details on the parts they offer.

Memory Size	**4Gb**	**4Gb**	**6Gb**	**6Gb**	**8Gb**	**8Gb**	**12Gb**	**12Gb**	**16Gb**	**16Gb**
Data Width	**X16**	**X32**	**X16**	**X32**	**X16**	**X32**	**X16**	**X32**	**X16**	**X32**
Banks Address	BA[3:0]	BA[3:0]	BA[3:0]	BA[3:0]	BA[3:0]	BA[3:0]	BA[3:0]	BA[3:0]	BA[3:0]	BA[3:0]
Row Address	A[13:0]	A[12:0]	A[14:0]	A[13:0]	A[14:0]	A[13:0]	A[15:0]	A[14:0]	A[15:0]	A[14:0]
Column Address (GDDR5X)	A[5:0]	A[5:0]	A[5:0]	A[5:0]	A[5:0]	A[5:0]	A[5:0]	A[5:0]	A[5:0]	A[5:0]
Column Address (GDDR5)	A[6:0]	A[6:0]	A[6:0]	A[6:0]	A[6:0]	A[6:0]	A[6:0]	A[6:0]	A[6:0]	A[6:0]
Auto precharge	A[8]	A[8]	A[8]	A[8]	A[8]	A[8]	A[8]	A[8]	A[8]	A[8]

Note:

1. When Address Compability Mode is OFF, {A[15:12], A[9], A[7]} is used for upper DQ in GDDR5X mode and {A[15:12], A[9], A[7], A[6]} is used for upper DQ in GDDR5 mode.
2. In GDDR5 mode, the column address notation does not include the lower three address bits as the burst order is always fixed for READ and WRITE commands. Burst length is 8.
3. In GDDR5X mode, the column address notation does not include the lower four address bits as the burst order is always fixed for READ and WRITE commands. Burst length is 16.
4. Row address range with A[13:12] = 11 (x32 mode) or A[14:13] = 11 (x16 mode) is not present for 6G density. Row address range with A[14:13] = 11 (x32 mode) or A[15:14] = 11 (x16 mode) is not present for 12G density. ACT/RD/WR commands to these memory locations are illegal.

4. Features

The Cadence GDDR5X Palladium memory model is based upon GDDR5X SGRAM Specification (V0.3) Item 1827.99. The GDDR5X features and the level of support provided in the MMP memory model are listed below.

Features	Support
Single Data Rate (SDR) commands (CK)	Yes
Double Data Rate (DDR) addresses (CK)	Yes
GDDR5X mode: Quad Data Rate (QDR) data (WCK); 16n prefetch architecture with 512 bit per array read or write access; burst length 16	Yes
GDDR5 mode: Double Data Rate (DDR) data (WCK); 8n prefetch architecture with 256 bit per array read or write access; burst length 8	Yes
Write and Read lower/upper DQ on lower and upper addresses	Yes
Address compatibility mode	Yes
16 internal banks	Yes
4 bank groups for tCCDL = 3 tCK and 4 tCK	Yes
Programmable read latency: 5 to 20 tCK; Programmable write latency: 1 to 7 tCK	Yes
Write data mask function via address bus (single/double/quad byte mask)	Yes
Data bus inversion (DBI) & address bus inversion (ABI)	Yes
Input/output PLL/DLL	No
Address training: address input monitoring via DQ/DBI_n/EDC pins	No
WCK2CK clock training with phase information via EDC pins	No
Data read and write training via READ FIFO (depth = 6)	No
Direct write data load to READ FIFO via WRTR command	No
Consecutive read of READ FIFO via RDTR command	No
Read/write EDC on/off mode	Yes
Programmable EDC hold pattern for CDR	No
Read/write data transmission integrity secured by cyclic redundancy check (CRC-8)	Yes
Programmable CRC read latency = 1 to 4 tCK; programmable CRC write latency = 7 to 14 tCK	Yes
Low Power modes	No
RDQS mode on EDC pins	No
Temperature related functions	No
Auto & self refresh modes	Yes
Power down	Yes
Auto precharge option for each burst access	Yes
Vendor ID for device identification	Yes
Boundary scan test mode	No

5. Model Parameter Descriptions

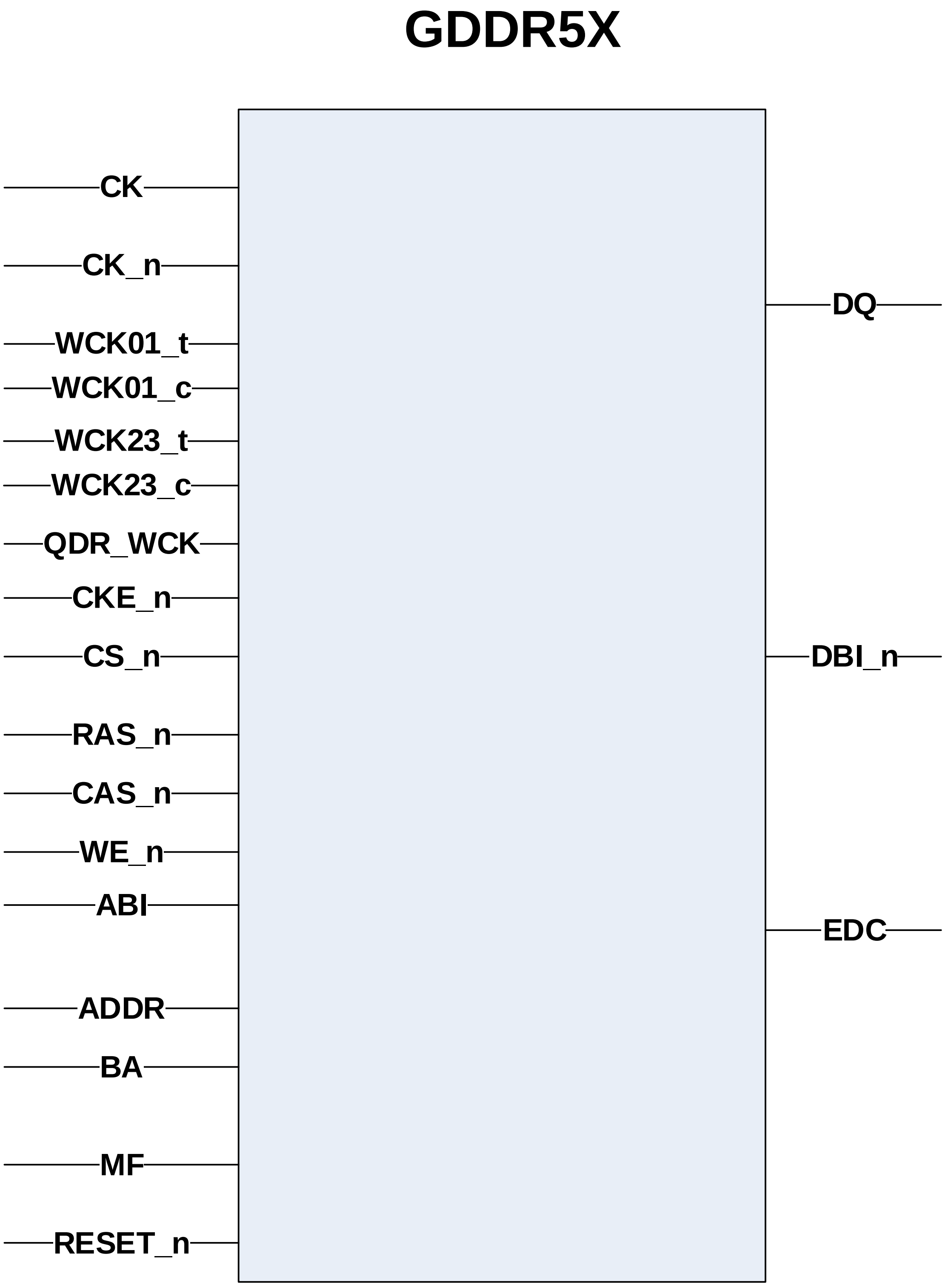
The following table provides details on the **user adjustable** parameters for the Palladium GDDR5X Memory Model. These parameters may be modified when instantiating a GDDR5X model or, if necessary, by modifying the HDL parameter declarations and default values which are exposed for access and debug visibility. Usually, users don't need to modify the parameters because Cadence has already provided several models in varying sizes. The values for these parameters follow the configurations listed in the Configurations section of this user guide.

User Adjustable Parameter	Default Value	Description
data_bits	32	Width of data bus
bnk_bits	4	Width of bank address for memory array
row_bits	16	Width of row address for memory array
colm_bits	7	Width of column address for memory array

6. Model Block Diagram

The width of the ADDR, BA, DBI_N, EDC, and DQ buses are dependent on the density and data width of the part being used.

NOTE: There is a special clock "QDR_WCK" in this model for supporting GDDR5X mode. The user MUST generate this QDR_WCK clock with Palladium and this clock is to be in 2:1 frequency relationship with and phase aligned with WCK01/WCK23. See section *Limitations* and section *QDR Read and Write* of this user guide to know the details. On the other hand, the GDDR5 legacy mode does not depend on the QDR_WCK.


GDDR5X
CK
CK_n
WCK01_t
WCK01_c
WCK23_t
WCK23_c
QDR_WCK
CKE_n
CS_n
RAS_n
CAS_n
WE_n
ABI
ADDR
BA
MF
RESET_n
DQ
DBI_n
EDC

7. Address mapping

The array of the GDDR5X model is mapped into the internal memory of the Palladium system. This array is a single two dimensional array. The mapping of bank, row and column addresses to the internal model array is as follows:

GDDR5 mode: ARRAY_ADDR = {BA, ROW, COL,3’b000} (COL={col[5:0],col[6]})
GDDR5X mode: ARRAY_ADDR = {BA, ROW, COL,4’b0000} (COL=col[5:0])

The following information is required if the memory needs to be preloaded with user data: the array names in the model hierarchy are: Array_L and Array_H for lower/upper DQ accessing. A xeDebug preload example will look as follows:

```
memory -load %readmemh tb_top.gddr5x_pd.Array_L -file lower.dat
memory -load %readmemh tb_top.gddr5x_pd.Array_H -file upper.dat
```

NOTE about Address Compatibility mode:
In this compatibility mode there are some known logical address to physical address mapping differences. These differences do not affect the functionality of the model with respect to the controller, as the address mapping from logical to physical is the same for writes vs reads. However, for backdoor load and dump functions the customer needs to take care. The difference is as follows: for GDDR5 the column address is 7 bits A[6:0]; in GDDR5X in compatibility mode it is **{**A[5:0], A[6]**}**. A[6] is the LSB within the page, not the MSB. If the user preloads the memory from a Palladium at runtime they will need to be aware of this difference.

8. Register Definitions

In the GDDR5X there are 16 Mode Registers (MR0 --- MR15). The Palladium GDDR5X model implements all the Mode Registers. However not all features are supported in the model.

8.1. Mode Register 0

The mode register 0 is implemented in the GDDR5X model. The RFU must be programmed to ‘0’ during MRS command, for all mode registers.

BA[3:0]	A[11:8]	A[7]	A[6:3]	A[2:0]
MR Select	Write Recovery	Test Mode	CAS Latency	Write Latency

BA[3:0]	MR Select
0000	MR0
0001	MR1
0010	MR2
0011	MR3
0100	MR4
0101	MR5
0110	MR6
0111	MR7
1111	MR15

Bit 11	Bit 10	Bit 9	Bit 8	Write Recovery
X	X	X	X	Not supported

Bit 7	Test Mode	
0	Normal	Supported
1	Test Mode	Not supported

MR8[0]	Bit 6	Bit 5	Bit 4	Bit 3	CAS Latency
0	0	0	0	0	5
0	0	0	0	1	6
0	0	0	1	0	7
0	0	0	1	1	8
0	0	1	0	0	9
0	0	1	0	1	10
0	0	1	1	0	11
0	0	1	1	1	12
0	1	0	0	0	13
0	1	0	0	1	14
0	1	0	1	0	15
0	1	0	1	1	16
0	1	1	0	0	17
0	1	1	0	1	18
0	1	1	1	0	19

0	1	1	1	1	20
1	0	0	0	0	21
1	0	0	0	0	22
…….					
1	1	1	1	0	35
1	1	1	1	1	36

Bit 2	**Bit 1**	**Bit 0**	**Write Latency**
0	0	0	Reserved
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

8.2. Mode Register 1

The mode register 1 is implemented in the GDDR5X model. The model supports the following parameter values.

BA[3:0]	**A[11]**	**A[10]**	**A[9]**	**A[8]**	**A[7]**	**A[6]**	**A[5:4]**	**A[3:2]**	**A[1:0]**
MR Select	RFU	ABI	WDBI	RDBI	RFU	Cal Upd	Addr / Cmd Termination	Data Termination	Drive Strength

Calibration Update, Addr / Cmd Termination, Data Termination, and Drive Strength are not applicable to the GDDR5X Palladium Model.

Bit 10	**ABI**
0	On
1	Off

Bit 9	**WDBI**
0	On
1	Off

Bit 8	**RDBI**
0	On
1	Off

8.3. Mode Register 2

The mode register 2 is implemented in the GDDR5X model. However, the model does not support any of the parameter values.

BA[3:0]	**A[11:9]**	**A[8:6]**	**A[5:3]**	**A[2:0]**

MR Select	Address/ Command Termination Offset	Data and WCK Termination Offset	OCD Pullup Driver Offset	OCD Pulldown Driver Offset

8.4. Mode Register 3

The mode register 3 is implemented in the GDDR5X model. The model does not support any content but is required to be present as it is part of the initialization sequence.

BA[3:0]	**A[11:10]**	**A[9:8]**	**A[7:6]**	**A5**	**A4**	**A3**	**A2**	**A[1:0]**
MR Select	Bank Groups	WCK Termination	DRAM Info	RDQS Mode	WCK2 CK	WCK23 INV	WCK01 INV	Self-Refresh

8.5. Mode Register 4

The mode register 4 is implemented in the GDDR5X model. The model supports the following parameter values, except EDC Inv and EDC Hold Pattern.

BA[3:0]	**A11**	**A10**	**A9**	**A[8:7]**	**A[6:4]**	**A[3:0]**
MR Select	EDC Inv	WR CRC	RD CRC	CRC Read Latency	CRC Write Latency	EDC Hold Pattern

Bit 10	**WR CRC**
0	On
1	Off

Bit 9	**RD CRC**
0	On
1	Off

Bit 8	**Bit 7**	**CRC Read Latency**
0	0	4
0	1	reserved
1	0	2
1	1	3

Bit 6	**Bit 5**	**Bit 4**	**CRC Write Latency**
0	0	0	reserved
0	0	1	8
0	1	0	9
0	1	1	10
1	0	0	11
1	0	1	12

1	1	0	13
1	1	1	14

8.6. Mode Register 5

The mode register 5 is implemented in the GDDR5X model. The model does not support any contents but is required to be present as it is part of the initialization sequence.

BA[3:0]	A[11:6]	A[5:3]	A2	A1	A0
MR Select	RAS	RFU	LP3	LP2	RFU

8.7. Mode Register 6

The mode register 6 is implemented in the GDDR5X model. The model does not support any contents but is required to be present as it is part of the initialization sequence.

BA[3:0]	A[11:8]	A[7:4]	A[3:1]	A[0]
MR Select	VREFD Offset Device ID 0 and 1	VREFD Offset Device ID 2 and 3	RFU	WCK PIN

8.8. Mode Register 7

The mode register 7 is implemented in the GDDR5X model. The model does not support any contents but is required to be present as it is part of the initialization sequence.

BA{3:0]	A[11:10]	A[9:8]	A[7]	A[6]	A[5]	A[4]	A[3]	A[2:0]
MR Select	DCC	VDD Range	Half VREFD	Temp Sense	DQ PreA	Auto Sync	LF Mode	RFU

8.9. Mode Register 8

The mode register 8 is implemented in the GDDR5X model. The model supports the following parameter values.

BA{3:0]	A[11:10]	A[9]	A[8]	A[7]	A[6]	A[5:4]	A[3]	A[2]	A[1]	A[0]
MR Select	CK Termination	GDDR5X Mode	Add Comp mode	VREFD monitor	VREFD C2D	PLL/DLL range	Hiber-nate	EDC High-z	WR MSB	RL MSB

Bit 9	G5X mode
0	Off
1	On

Bit 8	Add Comp mode
0	Off
1	On

Bit 1	WR MSB
0	0
1	1

Bit 0	RL MSB
0	0
1	1

8.10.Mode Register 15

The mode register 15 is implemented in the GDDR5X model. The model supports the following parameter values.

BA[3:0]	A[11:10]	A[9]	A[8]	A[7:0]
MR Select	Address Training	MRE MF1	MRE MF0	X

Bit 11	Bit 10	Address Training
X	0	Off
0	1	Train rising CK edge
1	1	Train rising /CK edge

Bit 9	MR Enable MF=1
0	Enable
1	Disable

Bit 8	MR Enable MF=0
0	Enable
1	Disable

9. Commands

The GDDR5X model accepts the following commands:

- Deselect
- NOP
- Mode Register Set
- Activate
- Read
- Read with AP
- Load Fifo
- Read Training
- Write
- Write with AP
- Write with Single Byte Mask
- Write with AP, Single Byte Mask
- Write with Double Byte Mask
- Write with AP, Double Byte Mask

- Write Training
- Precharge
- Precharge All
- Refresh

10. Initialization Sequence

The GDDR5X model requires that the memory controller follows the initialization sequence as documented in the specification. The sequence basically entails the following steps:

1. Assert RESET_N
2. De-assert RESET_N with DBI_N and MF set for the desired device ID
3. Start clocks CK and CK_N
4. Wait for CKE_N to asserted
5. Issue at least 2 NOP commands
6. Issue a Precharge All command followed by NOP commands
7. Issue MRS command to MR15 for address training mode (optional)
8. Start clocks WCK01/WCK23
9. Issue MRS commands to the Mode Registers in any order
10. Issue two Refresh commands followed by NOP commands

Generally there is no ordering required for step 9 but all mode registers need to be written. 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. The device ID chosen in step 2 determines the write mask mapping for x8 models.

11. Synthesis and Compilation of the Model

The model is provided as a protected RTL source file for the core memory along with a selection of wrapper files that correspond to various configurations. Please see section 3 Model Configurations. The gddr5x_pd.vp file need to be synthesized prior to compiling for Palladium, either with HDL-ICE in Classic ICE flow or with IXCOM flow.

An IXCOM compile example is provided below:

```
ixcom   -clean -ua \
        -timescale 1ps/1ps \
        +dut+gddr5x_pd +sv \
        -incdir ../../../utils/cdn_mmp_utils/sv \
        ../../../utils/cdn_mmp_utils/sv/cdn_mmp_utils.sv \
        +libext+.v+.sv+.vp \
        -y ../rtl \
        +incdir+../rtl
    . . .

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

The scripts below show two examples for Palladium classic ICE synthesis:

1)

```
hdlInputFile –add gddr5x_pd.vp
hdlImport -full -2001 -l qtref
hdlOutputFile -add -f Verilog gddr5x_pd.vg
hdlSynthesize -memory -keepRtlSymbol -keepAllFlipFlop gddr5x_pd
……
```

2)

```
vavlog      gddr5x_pd.vp
vaelab      -keepRtlSymbol -keepAllFlipFlop -outputVlog gddr5x_pd.vg
            gddr5x_pd
```

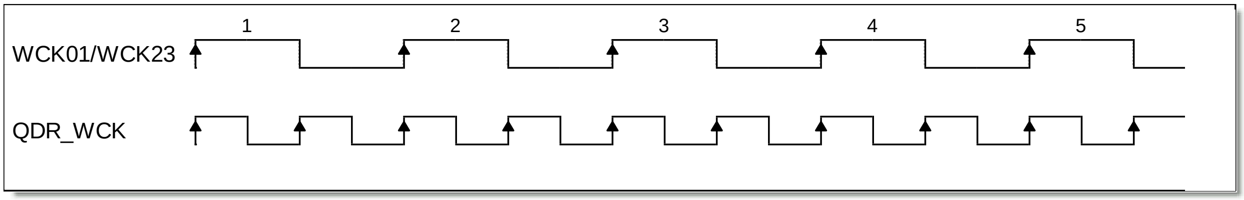
NOTE: It is common for Palladium flows to require –keepallFlipFlop since it removes optimizations that are in place by default.   For example, without –keepAllFlipFlop, HDL-ICE can remove flops with constant inputs and merge equivalent FF. The picture above is modified a bit when ICE ATB mode (–atb) 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.

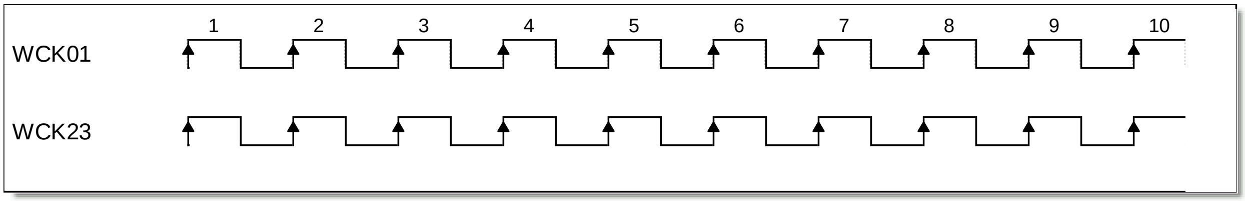
If the recommended compile script includes the aforementioned options, the user must include them to avoid affecting functionality of the design.

12. Limitations

1. Users MUST generate QDR_WCK by Palladium and in a 2:1 frequency relation to as well as phase aligned with WCK01/WCK23. The freuqnecy and phase relation is shown below. This clocking relationship supports the GDDR5X mode but is not required for legacy GDDR5 mode.

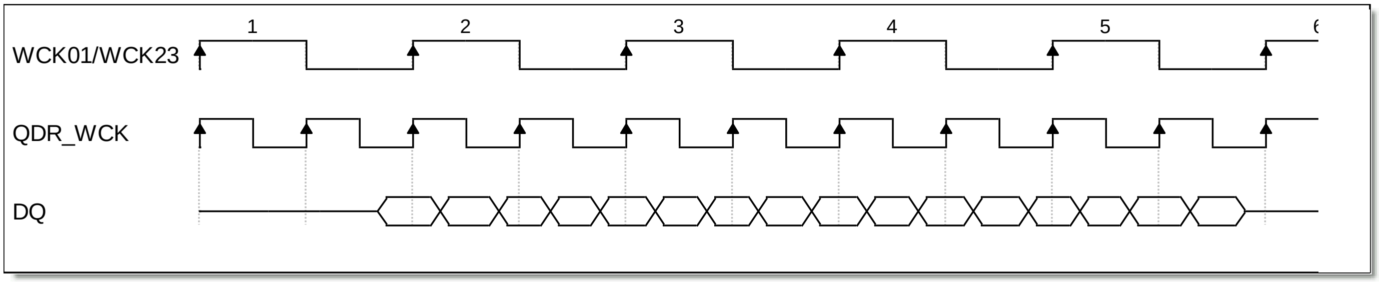


2. Emulation on Palladium is cycle based, so WCK01 and WCK23 shall be phase aligned.

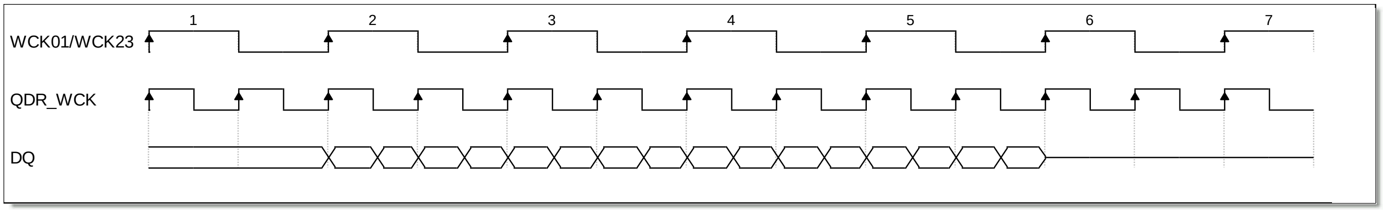

1
2
3
4
5
6
7
8
9
10
WCK01
WCK23

13. QDR Read and Write

The following figure shows QDR write with BL16 in GDDR5X mode.

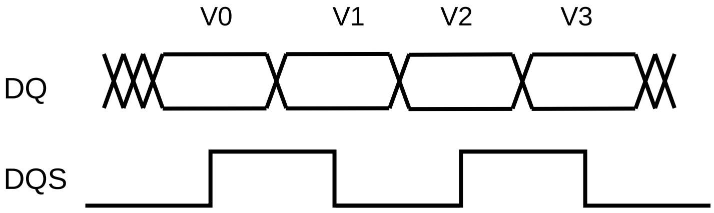


The following figure shows QDR read with BL16 in GDDR5X mode.

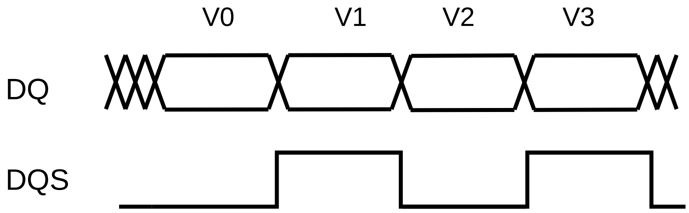


14. Handling DQS in Palladium Memory Models

For writes to a DDR memory, industry datasheets show each DQS edge centered within the corresponding valid period (v0, v1, v2, etc.) of DQ, as in the following diagram.

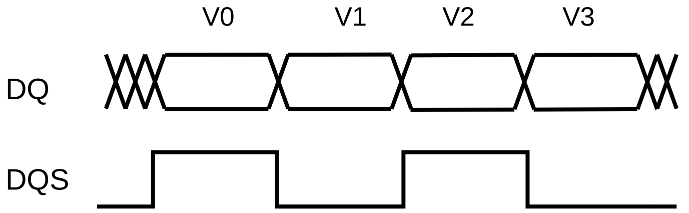


For DDR models provided by Cadence for Palladium, if the design drives DQ and DQS signals with the above timing, the DDR memory will behave correctly. However, to obtain this timing in Palladium, the fastest design clock must toggle twice as frequently as the DQS signal. If this faster clock is not needed for any other reason, the presence of the faster clock will usually cause an unnecessary 2X slowdown in emulation speed. To eliminate the need for a faster clock, you can have the design generate each DQS edge at the end of the corresponding DQ valid period (rather than the middle), as in the following diagram:



Note that the first DQS edge is at the *end* of first valid DQ, not at the beginning.

For reads from the DDR model, the DDR model will drive DQ and DQS with the first DQS edge at the *beginning* of the first valid data, not at the end:



The DDR model behaves this way to conform with industry datasheets for DDR memories. The design reading the data from the DDR model must delay the DQS signal, and use the delayed-DQS signal to sample the DQ. A delay of one Q_FDP0B should work fine, even in CAKE 1X mode. If you are using CAKE 1X mode and the DDR clock is the fastest design clock, the DQ signal will change twice per FCLK, and the Q_FDP0B delaying DQS will provide one-half FCLK delay, so that each delayed-DQS edge is at the end of the corresponding data valid period.

To delay the DQS signal, a commonly used approach is to create a special pad cell for DQS, that has a Q_FDP0B delay cell inserted on the path that leads from the DDR memory into the design.

The user may insert delays into pad cells (or elsewhere in the design) using the below code example which leverages ixc_pulse, an internal primitive that can be used to access FCLK and to create controlled delay, for IXCOM flow and leverages the Q_FDP0B primitive for delay generation in the Classic ICE flow. For more detailed information about ixc_pulse please reference the *UXE User Guide* section called *Generating Pulses*. There is no need for the user to define IXCOM_UXE for the Verilog macro; it is predefined for the user in IXCOM flow. Note that in UXE 13.1.0 and prior the equivalent pulse generating function was named axis_pulse.

```
// Flow independent delay cell
```

```
module pxp_fclk_delay (in, out_delay);
input in;
output out_delay;

reg out_delay;

`ifdef IXCOM_UXE
  wire VCC=1’b1;
  ixc_pulse #(1)(Fclk,VCC);
  always @(posedge Fclk)
    out_delay <= in;
`else
  Q_FDP0B fclk_dly (.D(in), .Q(out_delay));
`endif

endmodule
```

15. Revision History

The following table shows the revision history for this document

Date	Version	Revision
September 2015	1.0	Initial Release
November 2015	1.1	WCK01/WCK23 update
December 2015	1.2	Move GDDR5X model from IR (BETA) to ER (BETA)
December 2015	1.3	Update Mode registers
January 2016	1.4	Update for Palladium-Z1 and VXE
March 2016	1.5	Add note in address mapping section about address compatibility mode when preloading arrays
May 2016	1.6	Remove BETA watermark
July 2016	1.7	Remove hyphen in Palladium naming
January 2018	1.8	Modify header and footer
July 2018	1.9	Update for new utility library