cādence®

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

HyperRam Palladium Memory Model User Guide

Document Version: 1.4

Document Date: July 2018

Contents

List of Figures

List of Tables

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

HyperRam Memory Model

1. Introduction

The Cadence Palladium HyperRam Palladium memory model is based upon the *CYPRESS HyperRam* datasheet *(001-97964 Rev. *G, September 08, 2016).*

The model is initially only available in a few configurations based on the CYPRESS HyperRam specification. 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. Acronyms

Table 1: HyperRam Model Acronyms

NAME	Descriptions
ADDR	Address
AS	Address Space
BL	Burst Length
CA	Command-Address
CFG	Configuration
CK	Clock
ID	Device Identification
MFR	Manufacturer
POR	Power-On-Reset
RWDS	Read Write Data Strobe

4. Features

The HyperRam Palladium memory model is based upon the *CYPRESS HyperRam* datasheet. Below, Table 2: Feature List of HyperRam Model lists which features are supported and which are unsupported. Please refer to the *CYPRESS HyperRam spec* for detailed information.

Table 2: Feature List of HyperRam Model

FEATURE	SUPPORT	NOTE
Memory Array Read (Legacy Wrapped Burst)	Yes	
Memory Array Read (Hybrid Wrapped Burst)	Yes	
Memory Array Read (Linear Burst)	Yes	
Memory Array Write (Legacy Wrapped Burst)	Yes	
Memory Array Write (Hybrid Wrapped Burst)	Yes	
Memory Array Write (Linear Burst)	Yes	
Device Identification Registers	Yes	
Configuration Register	Yes	Drive Strength is not supported
Hybrid Burst	Yes	
Self-Refresh	Yes	A free-running 100Mhz clock is required
Power-On-Reset	No	

5. Configurations

Table 3: HyperRam Model Configurations 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.

Table 3: HyperRam Model Configurations

Model Size	Row Count	Row Size(Kbyte)
8 Mbyte	8192	1
16 Mbyte	16384	1
32 Mbyte	32768	1

6. Model Block Diagram

The following figure shows the HyperRam Model block diagram, which has a low pin count bus interface. The VCC power and ground pins are not supported. The CK_100M pin is added to support the self-refresh feature; when using this feature a free-running 100Mhz clock is required.

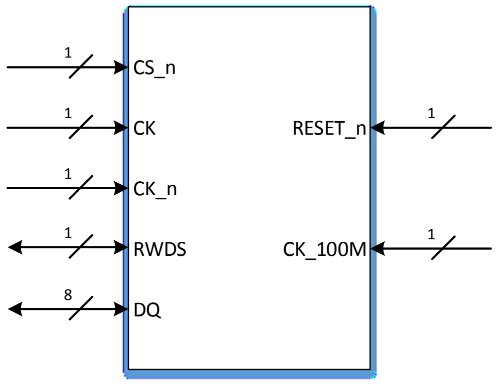


Figure 1: HyperRam Model Block Diagram

7. I/O Signal Description

The table below lists and describes the model I/O signals.

Table 4: HyperRam Model I/O Signals

NAME	TYPE	DESCRIPTION
CS_n	Input	**Chip Select.** Bus transactions are initiated with a High to Low transition. Bus transactions are terminated with a Low to High transition. The master device has a separate CS_n for each slave.
CK, CK_n	Input	**Differential Clock**. Command, address, and data information is output with respect to the crossing of the CK and CK# signals. The clock is not required to be free-running.
RWDS	Input/Output	**Read Write Data Strobe.** During the Command/Address portion of all bus transactions RWDS is a slave output and indicates whether additional initial latency is required. Slave output during read data transfer, data is edge-aligned with RWDS. Slave input during data transfer in write transactions to function as a data mask.(High = additional latency, Low = no additional latency).
DQ[7:0]	Input/Output	**Data Input/Output.** Command, Address, and Data information is transferred on these signals during Read and Write transactions.
RESET_n	Input	**Hardware RESET**. When Low the slave device will self initialize and return to the Standby state. RWDS and DQ[7:0] are placed into the High-Z state when RESET_n is Low. The slave RESET_n input includes a weak pull-up, if RESET_n is left unconnected it will be pulled up to the High state.
CK_100M	Input	**Self-Refresh Clock**. A free-running 100Mhz clock is need to support Self-Refresh.

8. Model Parameter Descriptions

The following table provides details on the user adjustable parameters for the Palladium HyperRam Memory Model. These parameters may be modified when instantiating a HyperRam wrapper or, if necessary, by modifying the HDL parameter declarations and default values which are exposed for access and debug visibility.

Table 5: User Adjustable Parameters

User Adjustable Parameter	Default Value	Description
addr_bits	22	The address width of HyperRam

The following table provides some information about exposed parameters and localparams that are NOT user adjustable. On rare occasion the user may find one of these parameters or localparam needs adjusting for their configuration. If this case arises, please contact Cadence emulation or MMP support.

Table 6: Visible Non-User-Adjustable Parameters & Localparams

Parameter / Localparam	Default Value	Description
data_bits	8	The data bus width of HyperRam
mem_depth	2**addr_bits	The memory depth of HyperRam
MFR	0001	Manufacturer of HyperRam 0000 - Reserved 0001 - Cypress 0010 to 1111 - Reserved
DEVICE_TYPE	0000	Device type 0000 - HyperRAM 0001 to 1111 - Reserved
ROW_ADDR_BIT_CNT	01100	Row Address Bit Count 00000 - One Row address bit ... 11111 - Thirty-two row address bits
COL_ADDR_BIT_CNT	1000	0000 - One column address bit ... 1111 - Sixteen column address bits
REFRESH_INTERVAL_CNT	400	Distributed Refresh Interval Count (Time/10ns)
REFRESH_TIME_CNT	8	Distributed Refresh Time Count (Time/10ns)

Note that there are additional exposed localparams in the model HDL that are not described here nor intended to be described here. These additional localparams are exposed for debugging purposes only and will not be described herein.

9. Address mapping

The array of the HyperRam model is mapped into the internal memory of the Palladium system. This array is a signal two dimensional array. The array width is 16bit and the address of the memory array is word address.

The array name in the model is: memcore

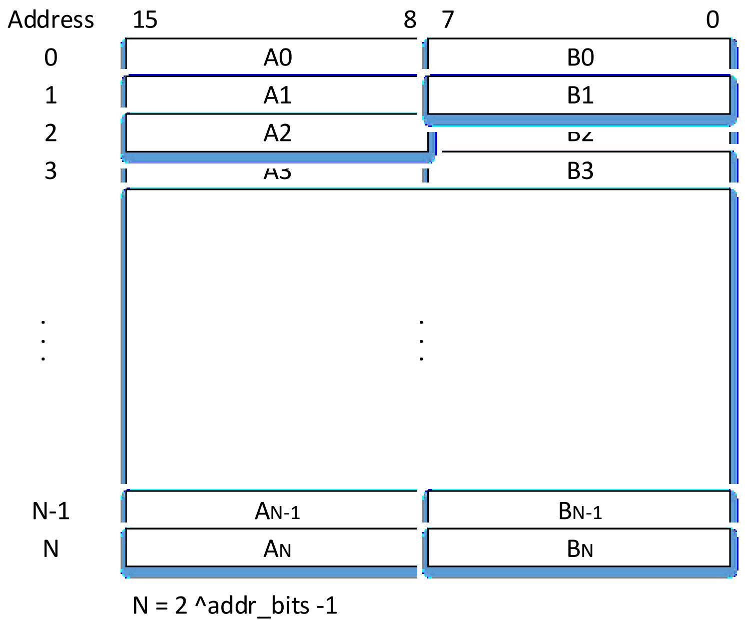


Figure 2: HyperRam Address Mapping

10. HyperRam Commands

The HyperRam memory model supports the commands listed in Table 7: Command List of HyperRam Model. Please refer to the *CYPRESS HyperRam* specification for details about the timing and bus cycles for each command.

Table 7: Command List of HyperRam Model

Command Sequence	SUPPORT
Memory Array Read (Wrapped Burst)	Yes
Memory Array Read (Linear Burst)	Yes
Memory Array Write (Wrapped Burst)	Yes
Memory Array Write (Linear Burst)	Yes
Register Read	Yes
Register Write	Yes

11. HyperRam Initialization Sequence

The HyperRam Memory Model requires that the memory controller first issue a hardware reset before issuing any commands.

12. Limitations

1. Unsupported features are listed in Table 2: Features List of HyperRam Model.

13. Compile and Emulation

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

```
ixcom -64bit -ua +sv +dut+s27ks0641 \
    ./s27ks0641.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 example for Palladium classic ICE synthesis:

```
1)
hdlInputFile s27ks0641.vp
hdlImport -full -2001 -l qtref
hdlOutputFile -add -f verilog s27ks0641.vg
hdlSynthesize -memory -keepVhdlCase -keepRtlSymbol -keepAllFlipFlop  s27ks0641
……

2)
```

```
vavlog s27ks0641.vp

vaelab -keepRtlSymbol -keepAllFlipFlop -outputVlog s27ks0641.vg s27ks0641
……
```

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.

14. Debugging

The HyperRam model has several debugging options techniques and tips that may assist the user may use in isolating a problem.

- For issues that are may not be HyperRam specific please review the *Memory Model Portfolio FAQ for All Models User Guide.*

- **Golden waveform:** A package with a reference waveform is available which shows the following command sequence:

  (1) Hardware reset
  (2) ID Check
  ID0 Register Read
  ID1 Register Read
  (3) Configuration Register R/W Test
  CFG0 Register Write 16’h55AA
  CFG1 Register Write 16’h9966
  CFG0 Register Read Back Check
  CFG1 Register Read Back Check
  (4) Memory R/W Test
  CFG0 Register Write, Set Legacy Wrap Burst, BL=32
  Memory Write, start address = 0x0000_0003, Burst Size = 64
  Memory Read, start address = 0x0000_0003, Burst Size = 64

- **Debug Display:** The Palladium HyperRam memory model has available a built-in debug methodology called MMP Debug Display that is based on the Verilog system task

$display. Please see the *Palladium Memory Model Debug Display User Guide* in the release docs directory for additional information.

- **Manual Configuring of this MMP Model Family**

This MMP model supports manual configuration by accompanying the model mode register or configuration register declarations with synthesis directives, such as keep_net directives, that instruct the compiler to ensure that the relevant nets remain available for runtime forcing. For a general description of this support please see the user guide in the MMP release with path and filename *docs/MMP_FAQ_for_All_Models.pdf*.

While MMP strongly recommends following protocol-based commands to configure MMP models, MMP recognizes that the design test environment may desire to trade off the risks inherent in streamlining or circumventing the initialization sequence part of the protocol in order to better support some testing environments.

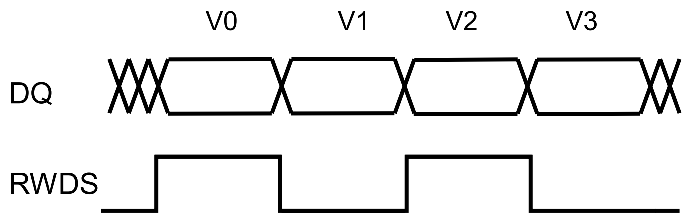
The following table lists the internal register path and naming along with the specification or datasheet naming for model mode registers or configuration registers that are accompanied by keep_net synthesis directives in support of such manual configuration. ONLY writeable configuration registers or fields are supported thusly. Please read the relevant datasheet for details about individual register behavior and mapping to fields.

Table 8: Writeable Mode Register / Configuration Register Info

Hierarchical RTL Naming for Writeable Configuration Related Registers & Signals	Specification or Vendor Datasheet Naming for Configuration Related Registers	Access
<model_name>.reg_cr0	Configuration Register 0	R/W
<model_name>.reg_cr1	Configuration Register 1	R/W

15. Handling RWDS in Palladium HyperRam Memory Model

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



The HyperRam model behaves this way to conform to CYPRESS HyperRam spec. The design reading the data from the HyperRam model must delay the RWDS signal, and use the delayed-RWDS 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 HyperRam clock is the fastest design clock, the DQ signal will change twice per FCLK, and the Q_FDP0B delaying RWDS will provide one-half FCLK delay, so that each delayed-RWDS edge is at the end of the corresponding data valid period.

To delay the RWDS signal, a commonly used approach is to create a special pad cell for RWDS that has a Q_FDP0B delay cell inserted on the path that leads from the HyperRam 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
```

Revision History

The following table shows the revision history for this document

Date	Version	Revision
November 2016	1.0	Initial release.
March 2017	1.1	Change model name to lower case
January 2018	1.2	Modify header and footer
May 2018	1.3	Move Model to MR non-Beta Level Add section for Manual configuration
July 2018	1.4	Update for new utility library