Nexus DDR3 Memory Controller Driver API Reference

User Guide

FPGA-TN-02401-1.1

July 2025

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS, with all faults, and all associated risk is the responsibility entirely of the Buyer. The information provided herein is for informational purposes only and may contain technical inaccuracies or omissions, and may be otherwise rendered inaccurate for many reasons, and Lattice assumes no obligation to update or otherwise correct or revise this information. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. LATTICE PRODUCTS AND SERVICES ARE NOT DESIGNED, MANUFACTURED, OR TESTED FOR USE IN LIFE OR SAFETY CRITICAL SYSTEMS, HAZARDOUS ENVIRONMENTS, OR ANY OTHER ENVIRONMENTS REQUIRING FAIL-SAFE PERFORMANCE, INCLUDING ANY APPLICATION IN WHICH THE FAILURE OF THE PRODUCT OR SERVICE COULD LEAD TO DEATH, PERSONAL INJURY, SEVERE PROPERTY DAMAGE OR ENVIRONMENTAL HARM (COLLECTIVELY, "HIGH-RISK USES"). FURTHER, BUYER MUST TAKE PRUDENT STEPS TO PROTECT AGAINST PRODUCT AND SERVICE FAILURES, INCLUDING PROVIDING APPROPRIATE REDUNDANCIES, FAIL-SAFE FEATURES, AND/OR SHUT-DOWN MECHANISMS. LATTICE EXPRESSLY DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY OF FITNESS OF THE PRODUCTS OR SERVICES FOR HIGH-RISK USES. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Inclusive Language

This document was created consistent with Lattice Semiconductor’s inclusive language policy. In some cases, the language in underlying tools and other items may not yet have been updated. Please refer to Lattice’s inclusive language FAQ 6878 for a cross reference of terms. Note in some cases such as register names and state names it has been necessary to continue to utilize older terminology for compatibility.

Contents

Contents 3
Abbreviations in This Document 6
1. Introduction 7
1.1. Purpose 7
1.2. Audience 7
1.3. Driver and IP Compatibility 7
2. API Description 8
2.1. DDR3_init() 8
2.2. ddr3_PowerDownModeEnable() 8
2.3. ddr3_PowerDownModeDisable() 8
2.4. ddr3_SelfRefreshModeEnable() 9
2.5. ddr3_SelfRefreshModeDisable() 9
2.6. ddr3_ZQcalibrationShortEnable() 9
2.7. ddr3_ZQcalibrationLongEnable() 9
3. Function Call Flow Diagrams 10
3.1. DDR3_init() 10
3.2. ddr3_PowerDownModeEnable() 11
3.3. ddr3_PowerDownModeDisable() 12
3.4. ddr3_SelfRefreshModeEnable() 13
3.5. ddr3_SelfRefreshModeDisable() 14
3.6. ddr3_ZQcalibrationShortEnable() 15
3.7. ddr3_ZQcalibrationLongEnable() 16
4. API Data Structures 17
4.1. Struct DDR3 17
4.2. Union and Struct config_ctrl_reg_t 17
4.3. Union and Struct mode_settings_reg_t 17
4.4. Union and Struct status_reg_t 18
5. API Enum 19
5.1. Enum DDR_RET 19
6. API Variables 20
7. API Macros 21
References 22
Technical Support Assistance 23
Revision History 24

Figures

Figure 3.1. DDR3_init() 10
Figure 3.2. ddr3_PowerDownModeEnable() 11
Figure 3.3. ddr3_PowerDownModeDisable() 12
Figure 3.4. ddr3_SelfRefreshModeEnable() 13
Figure 3.5. ddr3_SelfRefreshModeDisable() 14
Figure 3.6. ddr3_ZQcalibrationShortEnable() 15
Figure 3.7. ddr3_ZQcalibrationLongEnable() 16

Tables

Table 1.1. Driver and IP Version 7
Table 1.2. Quick Facts of Driver Tested Environment 7
Table 2.1. DDR3_init() 8
Table 2.2. DDR3_PowerDownModeEnable() 8
Table 2.3. DDR3_PowerDownModeDisable() 8
Table 2.4. ddr3_SelfRefreshModeEnable() 9
Table 2.5. ddr3_SelfRefreshModeDisable() 9
Table 2.6. ddr3_ZQcalibrationShortEnable() 9
Table 2.7. ddr3_ZQcalibrationLongEnable() 9
Table 4.1. DDR3 Parameters 17
Table 4.2. config_ctrl_reg_t Parameters 17
Table 4.3. mode_settings_reg_t Parameters 17
Table 4.4. status_reg_t Parameters 18
Table 5.1. DDR_RET Variables 19
Table 7.1. API Macros Description 21

Abbreviations in This Document

A list of abbreviations used in this document.

Abbreviations	Definition
API	Application Programming Interface
FPGA	Field Programmable Gate Array
IP	Intellectual Property (FPGA)
JEDEC	Joint Electron Device Engineering Council
DDR3	Double Data Rate Generation 3
SCLK	System Clock
SDRAM	Synchronous Dynamic Random Access Memory
SDK	Software Development Kit

1. Introduction

The Lattice Semiconductor DDR3 Memory Controller for Nexus Devices provides a complete solution that includes a controller, DDR PHY, and the necessary clocking and training logic to interface with DDR3 SDRAM.

For further information about the IP core and its register descriptions, refer to the DDR3 SDRAM Controller IP for Nexus Devices (FPGA-IPUG-02086).

1.1. Purpose

DDR3 and SDK provide a set of application programming interfaces (APIs) that grant access to specific Lattice hardware and software capabilities. This document serves as a reference guide for developers by detailing the C language driver APIs and function call flows.

1.2. Audience

The intended audience for this document includes embedded system designers and embedded software developers using Lattice Certus™-NX devices. The technical guide assumes readers have expertise in embedded systems and FPGA technologies.

1.3. Driver and IP Compatibility

Table 1.1. Driver and IP Version

Driver version	IP version
25.01.00	2.2.0

Refer to the DDR3-SDRAM-Controller-IP-Release-Notes (FPGA-RN-02032) for more information on the driver and IP versions.

Table 1.2. Quick Facts of Driver Tested Environment

Driver tested on HW Device	Nexus Family	Certus-NX
Tool Version	Lattice Propel Builder	2023.2, 2024.1.2406150513_p, 2025.1.2505271020
	Lattice Propel SDK	2023.2, 2024.1.2406150513_p, 2025.1.2505271020
	Lattice Radiant Software	2023.2, 2024.1.0.34.2, 2024.1.1.259.1, 2024.2.1.334.0
	Radiant Programmer	2023.2, 2024.1.0.34.2, 2024.1.1.259.1, 2024.2.1.334.0

2. API Description

2.1. DDR3_init()

This API is used to initialize the DDR3 controller. This API takes the base address of DDR3 and initializes the handle.

```
DDR_RET ddr3_init(ddr3 *InstancePtr, unsigned int base_addr)
```

Table 2.1. DDR3_init()

In/Out	Parameter	Description	Returns
In	InstancePtr	Handle the DDR3 structure.	"Enum DDR_RET state" NO_FAIL = 0, CBT_FAIL, WR_LVL_FAIL, RD_TRN_FAIL, WR_TRN_FAIL, OTHER_FAIL
In	base_addr	Base address to be assigned to controller.	

2.2. ddr3_PowerDownModeEnable()

This API enables power-down configuration settings from the Mode Control Register. Users can be set these attributes manually during run-time. Once the DDR3 engine enters power-down mode, it remains in this mode until user issues a power-down exit command manually. The controller discards the Power Down Entry command if the memory is already in Power Down mode. Refer to the IP user guide for a register description/user command.

```
unsigned int ddr3_PowerDownModeEnable(ddr3 *InstancePtr)
```

Table 2.2. DDR3_PowerDownModeEnable()

In/Out	Parameter	Description	Returns
In	InstancePtr	Handle the DDR3 structure.	0: failure 1: success

2.3. ddr3_PowerDownModeDisable()

This API disables power-down configuration settings from the Mode Control Register. Users can set these attributes manually during run-time. Once the DDR3 engine enters power-down mode, this command brings the DDR3 controller out of power-down mode. The controller discards the Power Down Exit command if the memory is not already in Power Down mode. Refer to the IP user guide for a register description/user command.

```
unsigned int ddr3_PowerDownModeDisable(ddr3 *InstancePtr)
```

Table 2.3. DDR3_PowerDownModeDisable()

In/Out	Parameter	Description	Returns
In	InstancePtr	Handle the DDR3 structure.	0: failure 1: success

2.4. ddr3_SelfRefreshModeEnable()

This API enables self-refresh mode. The controller discards self-refresh entry command if the memory is already in self-refresh mode. Refer to the IP user guide for the register description/user commands.

```
unsigned int ddr3_SelfRefreshModeEnable(ddr3 *InstancePtr);
```

Table 2.4. ddr3_SelfRefreshModeEnable()

In/Out	Parameter	Description	Returns
In	InstancePtr	Handle the DDR3 structure.	0: failure 1: success

2.5. ddr3_SelfRefreshModeDisable()

This API disables self-refresh mode. The controller discards self-refresh exit command if the memory is not already in self-refresh mode. Refer to the IP user guide for the register description/user commands.

```
unsigned int ddr3_SelfRefreshModeDisable(ddr3 *InstancePtr);
```

Table 2.5. ddr3_SelfRefreshModeDisable()

In/Out	Parameter	Description	Returns
In	InstancePtr	Handle the DDR3 structure.	0: failure 1: success

2.6. ddr3_ZQcalibrationShortEnable()

This API enables short ZQ calibration during runtime. Users can manually perform calibrations to account for voltage and temperature variations. Refer to the IP user guide for the register description/user commands.

```
unsigned int ddr3_ZQcalibrationShortEnable(ddr3 *InstancePtr);
```

Table 2.6. ddr3_ZQcalibrationShortEnable()

In/Out	Parameter	Description	Returns
In	InstancePtr	Handle the DDR3 structure.	0: failure 1: success

2.7. ddr3_ZQcalibrationLongEnable()

This API enables Long ZQ calibration during runtime. This API performs full calibration. Usually, long ZQ calibration triggers during the reset/Initialization phase. Users must use ZQ short calibration during runtime for quick calibration. Refer to the IP user guide for the register description/user commands.

```
unsigned int ddr3_ZQcalibrationLongEnable(ddr3 *InstancePtr);
```

Table 2.7. ddr3_ZQcalibrationLongEnable()

In/Out	Parameter	Description	Returns
In	InstancePtr	Handle the DDR3 structure.	0: failure 1: success

3. Function Call Flow Diagrams

3.1. DDR3_init()

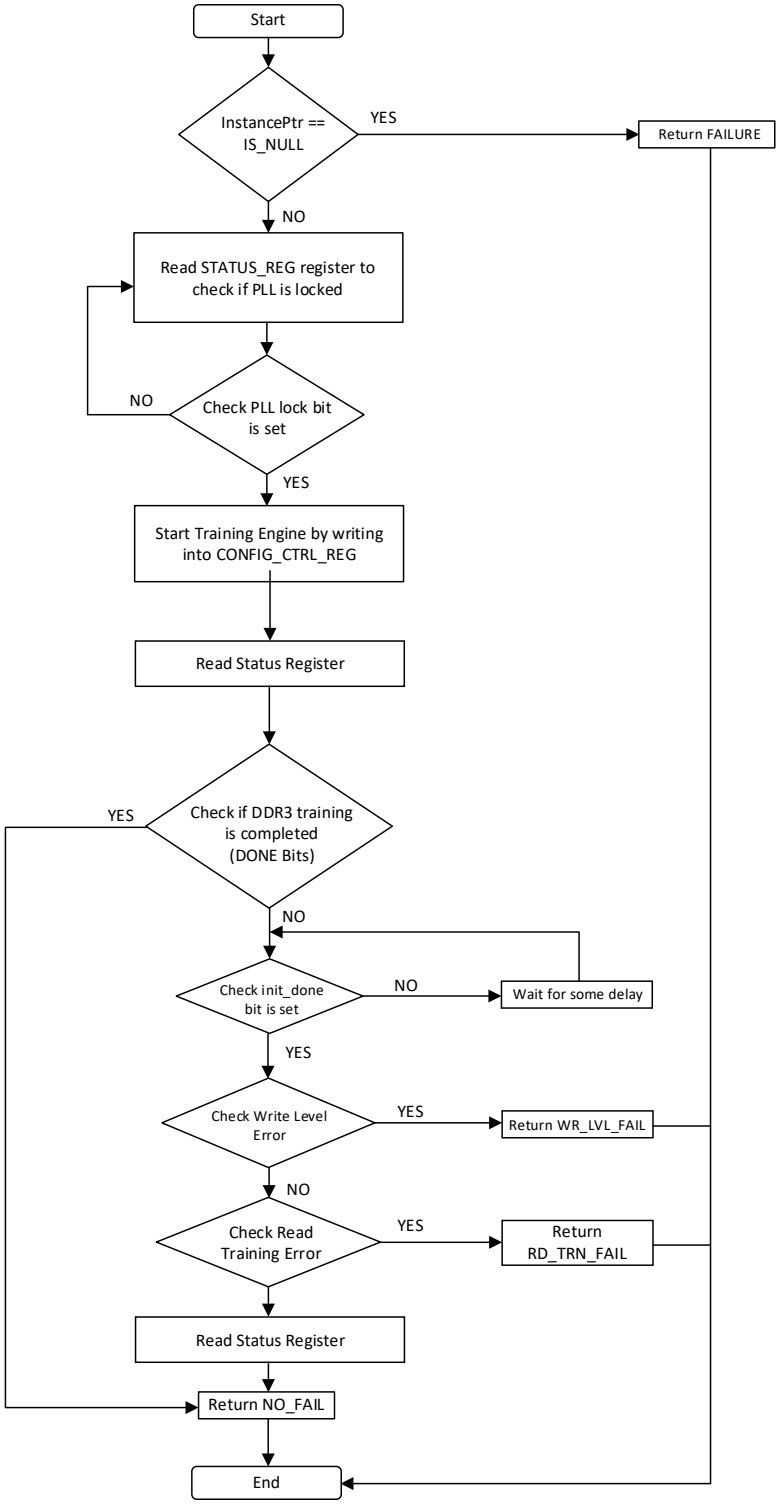

Figure 3.1. DDR3_init()

3.2. ddr3_PowerDownModeEnable()

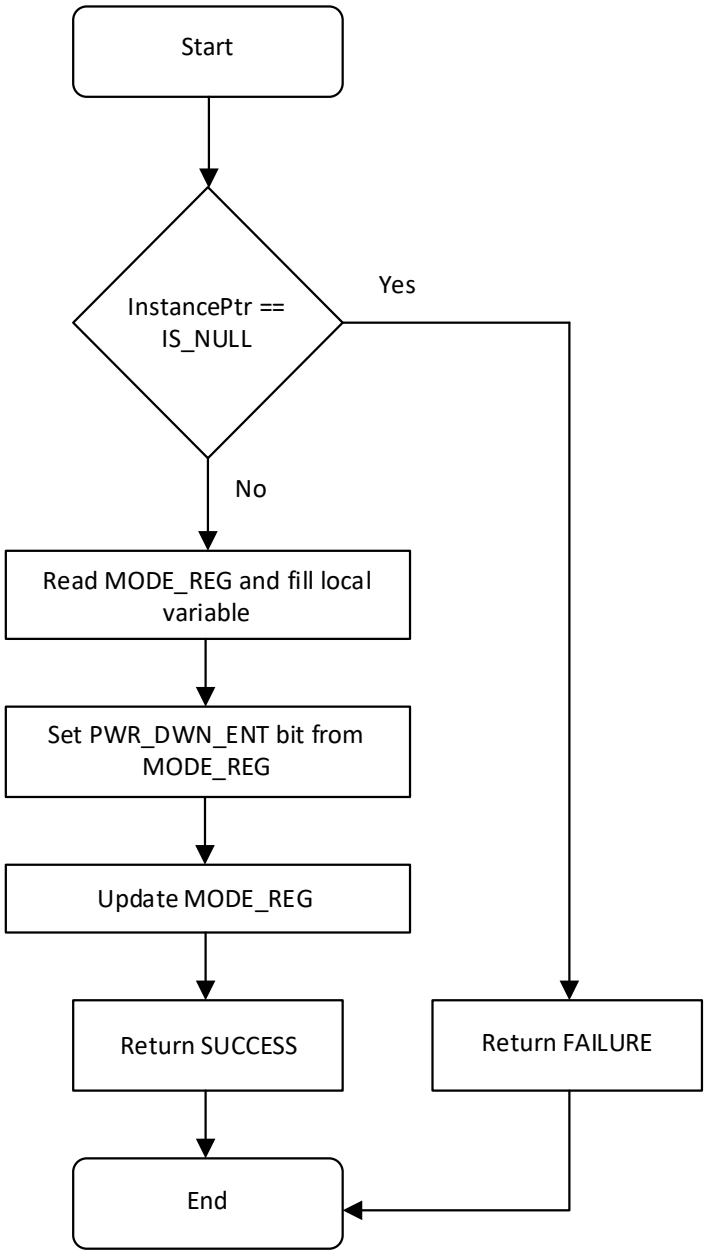

Figure 3.2. ddr3_PowerDownModeEnable()

3.3. ddr3_PowerDownModeDisable()

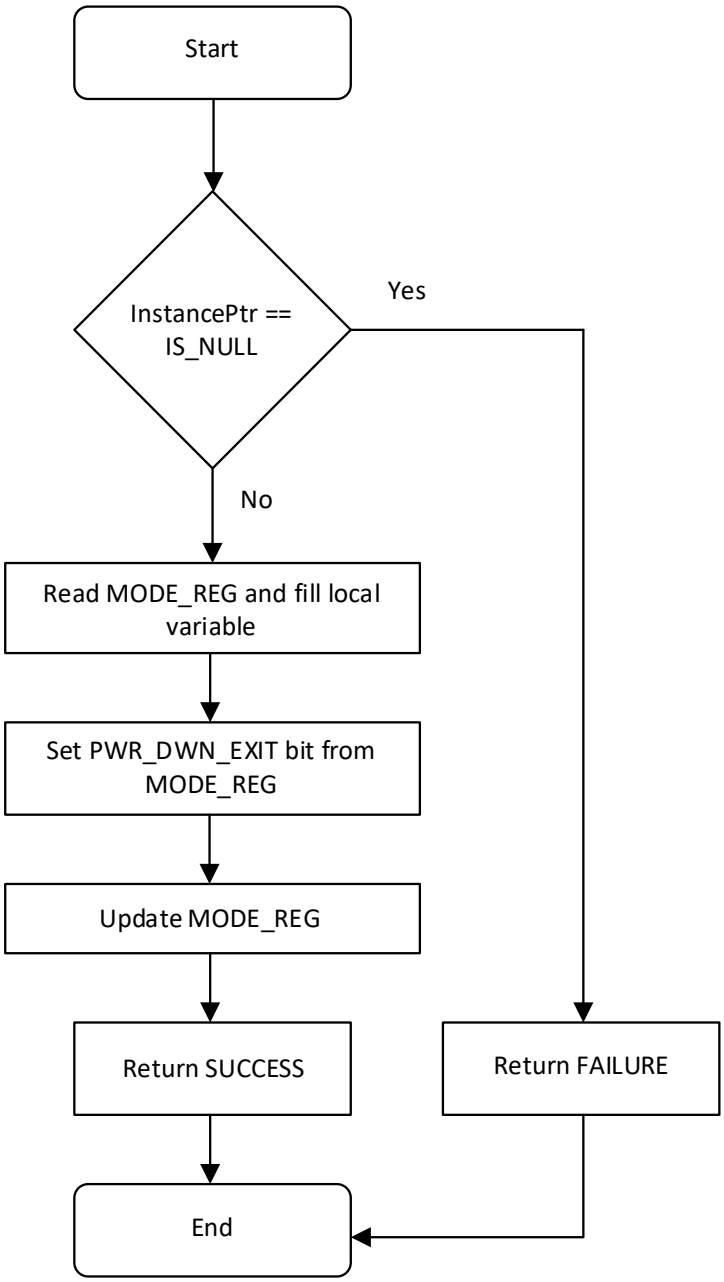

Figure 3.3. ddr3_PowerDownModeDisable()

3.4. ddr3_SelfRefreshModeEnable()

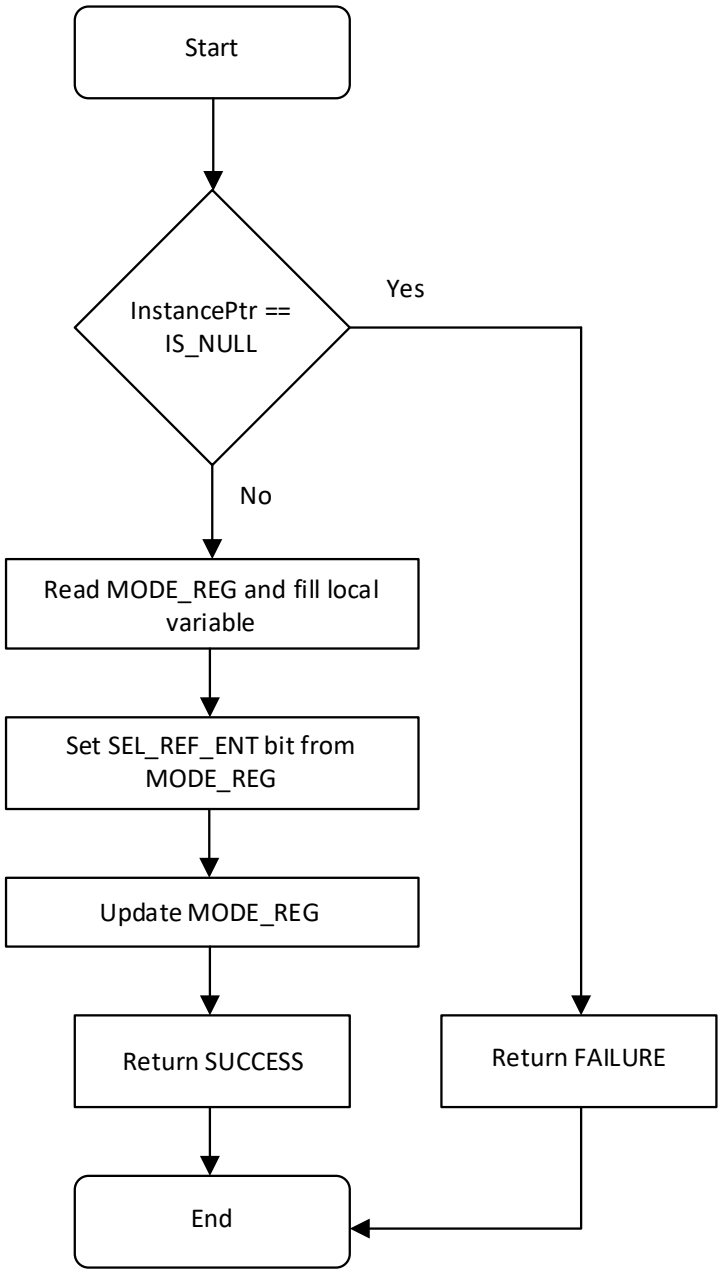

Figure 3.4. ddr3_SelfRefreshModeEnable()

3.5. ddr3_SelfRefreshModeDisable()

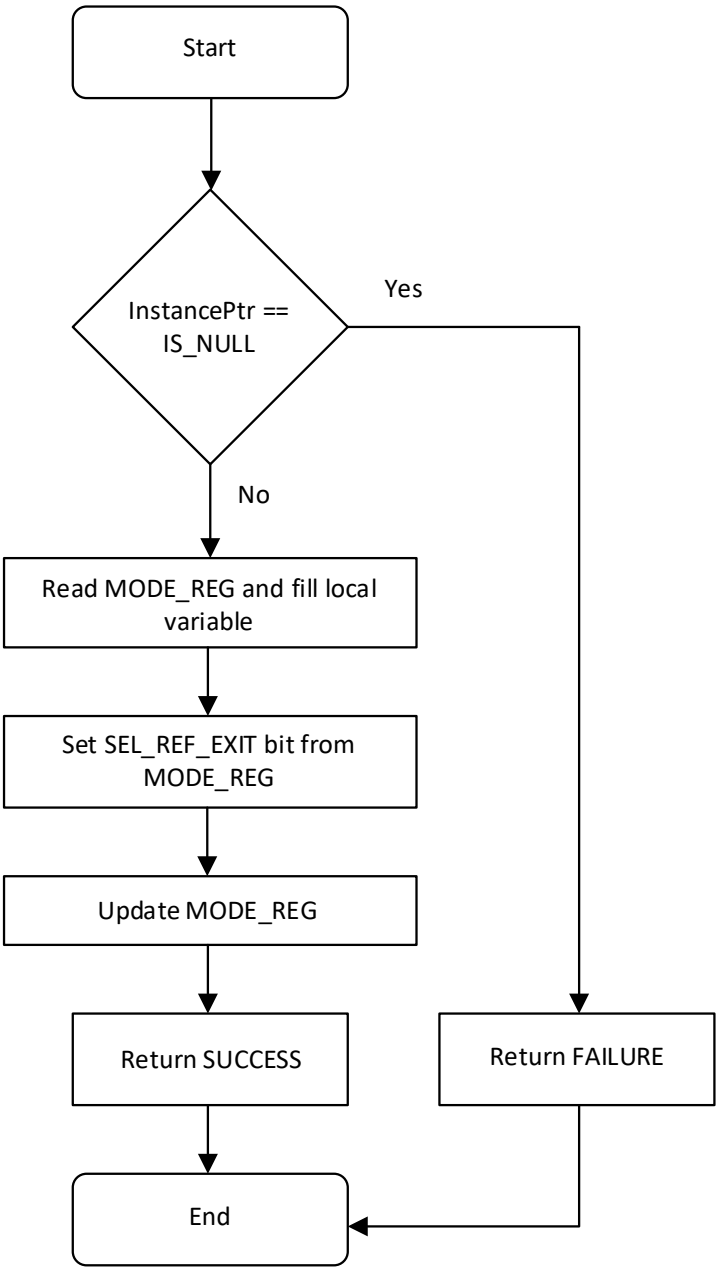

Figure 3.5. ddr3_SelfRefreshModeDisable()

3.6. ddr3_ZQcalibrationShortEnable()

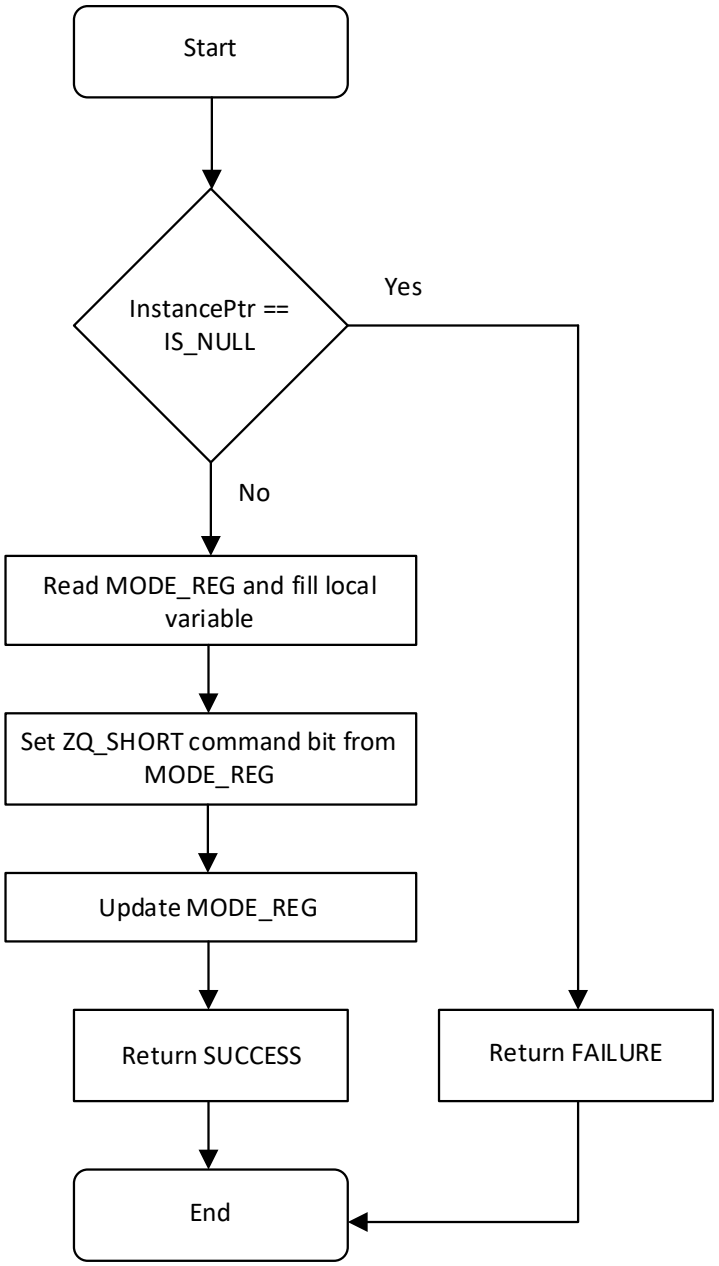

Figure 3.6. ddr3_ZQcalibrationShortEnable()

3.7. ddr3_ZQcalibrationLongEnable()

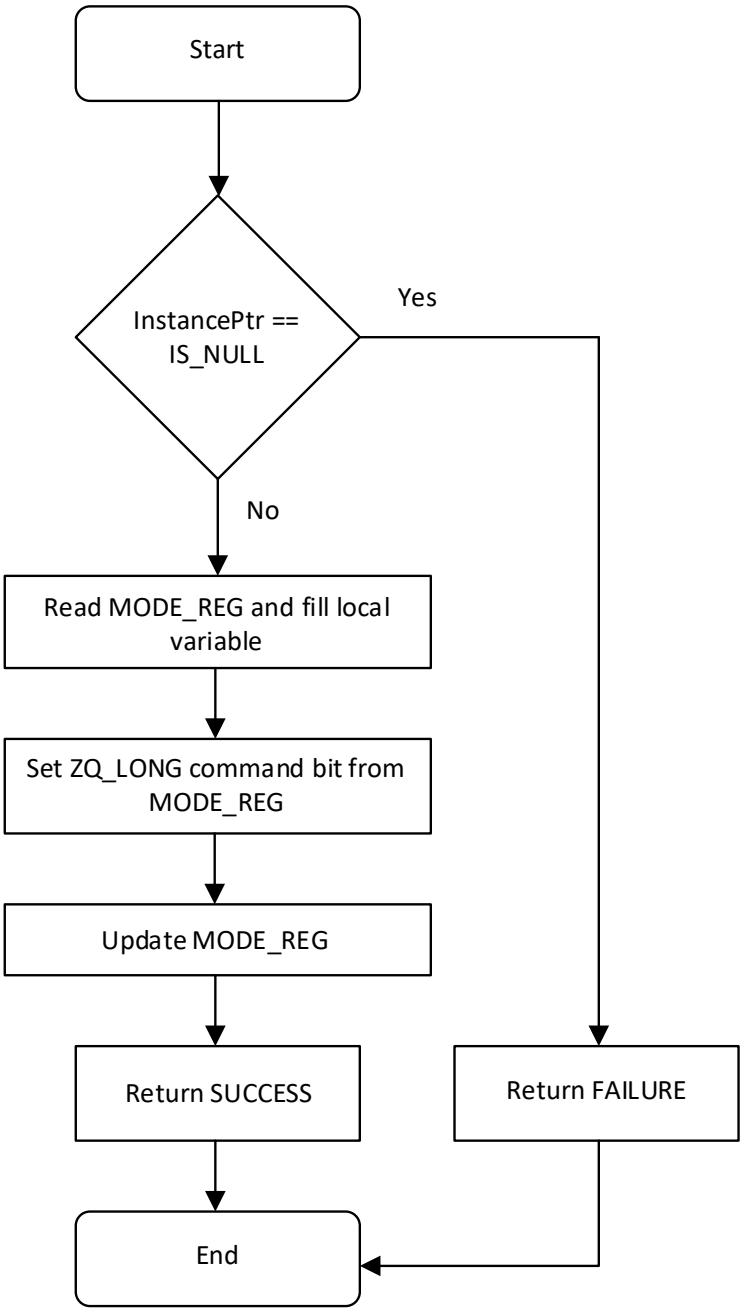

Figure 3.7. ddr3_ZQcalibrationLongEnable()

4. API Data Structures

4.1. Struct DDR3

Table 4.1. DDR3 Parameters

Data Type	Struct Member	Description
unsigned int	ddr3_base_address	Base address of the DDR3 memory controller.

4.2. Union and Struct config_ctrl_reg_t

Refer to the DDR3 SDRAM Controller IP for Nexus Devices (FPGA-IPUG-02086) IP user guide for the detailed register description.

Table 4.2. config_ctrl_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the config_ctrl_reg_t register members.
unsigned int	reg	Variable to access the config_ctrl_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	init_start	Write a '1' to this bit to begin initialization. This is a strobe bit.
unsigned int	config_rsvd1	Reserved
unsigned int	start_wl	Write a '1' to this bit to begin write levelling. This is a strobe bit.
unsigned int	config_data_en	Write a '1' to this bit to enable configuration access mode. This bit is sticky. To return to data access mode, write a '0' to this bit.
unsigned int	config_rsvd2	Reserved

4.3. Union and Struct mode_settings_reg_t

Refer to the DDR3 SDRAM Controller IP for Nexus Devices (FPGA-IPUG-02086) IP user guide for the detailed register description.

Table 4.3. mode_settings_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the mode_settings_reg_t register members.
unsigned int	reg	Variable to access the mode_settings_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	user_cmd	Supported user commands: • Power down entry • Self-Refresh entry • Self-Refresh exit • Power down exit • ZQ Calibration long • ZQ Calibration short Refer to the User Commands section in the DDR3 SDRAM Controller IP for Nexus Devices (FPGA-IPUG-02086) for the complete list of supported user commands via Native interface.
unsigned int	mode_reg_rsvd_1	Reserved

4.4. Union and Struct status_reg_t

Refer to the DDR3 SDRAM Controller IP for Nexus Devices (FPGA-IPUG-02086) IP user guide for the detailed register description.

Table 4.4. status_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the status_reg_t register members.
unsigned int	reg	Variable to access the status_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	init_done	1: Initialization done 0: Initialization not done
unsigned int	write_lvl_done	1: Write levelling done 0: Write levelling not done
unsigned int	write_lvl_err	1: Write levelling failed 0: Write levelling successful
unsigned int	read_trn_err	1: Read training failed 0: Read training successful
unsigned int	clk_good	1: Clock is ready 0: Clock not ready
unsigned int	cmd_ready	1: Ready to accept commands 0: Not ready to accept commands
unsigned int	pll_lock	1: PLL Lock asserted 0: PLL Lock not asserted
unsigned int	self_ref_mode	1: Self-Refresh mode 0: Out of Self-Refresh mode
unsigned int	pwr_down_mode	1: In power down mode 0: Out of power down mode
unsigned int	sts_rsvd1	Reserved

5. API Enum

5.1. Enum DDR_RET

Table 5.1. DDR_RET Variables

Enum Member	Enum Decimal Value
NO_FAIL	0
CBT_FAIL	1
WR_LVL_FAIL	2
RD_TRN_FAIL	3
WR_TRN_FAIL	4
OTHER_FAIL	5

6. API Variables

This is used to access the base address of DDR3 IP from the Propel Builder Configuration table/sys_platform.h.

```
static ddr3 *ddr3_inst;
```

7. API Macros

Table 7.1. API Macros Description

Macro	Description
#define DDR3_DRV_VER	DDR3 Driver version
#define CONFIG_CTRL_REG	Register address of Config Control Register
#define STATUS_REG	Register address of Status Register
#define MODE_REG	Register address of Mode Settings Register
#define DDR3_DONE_BITS	DDR3 initialization done bits
#define DDR3_ERR_DONE_BITS	DDR3 initialization error bits
#define PLL_LOCK_CLK_GOOD_BIT	DDR3 PLL lock bit
#define TRN_EN	DDR3 training enable bits
#define DATA ACCCESS_EN	Data Access Enable
#define TWO_US_DELAY	2us delay
#define PWR_DWN_ENT	Power Down Entry
#define PWR_DWN_EXIT	Power Down Exit
#define SUCCESS	Success
#define FAILURE	Failure
#define IS_NULL	Null value
#define SEL_REF_ENT	Self-Refresh Entry
#define SEL_REF_EXIT	Self-Refresh Exit
#define ZQ_SHORT	ZQ Calibration Short
#define ZQ_LONG	ZQ Calibration Long

References

- DDR3 SDRAM Controller IP for Nexus Devices (FPGA-IPUG-02086)
- DDR3-SDRAM-Controller-IP-Release-Notes (FPGA-RN-02032)
- Lattice Propel 2024.1 Builder User Guide (FPGA-UG-02212)
- Lattice Solutions IP Cores web page
- Lattice Propel Design Environment web page
- Lattice Radiant Software 2024.1 User Guide
- Lattice Radiant FPGA design software
- Lattice Insights for Lattice Semiconductor training courses and learning plans

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

For frequently asked questions, please refer to the Lattice Answer Database at www.latticesemi.com/Support/AnswerDatabase.

Revision History

Revision 1.1, July 2025

Section	Change Summary
All	Minor editorial fixes.
Introduction	• Updated the driver version to *25.01.00*. • Updated the IP version to *2.2.0*.
Function Call Flow Diagrams	Updated Figure 3.1. DDR3_init().
API Data Structures	Added the DDR3 SDRAM Controller IP for Nexus Devices (FPGA-IPUG-02086) reference in Table 4.3. mode_settings_reg_t Parameters.
API Macros	Added DATA_ACCESS_EN API Macro.

Revision 1.0, March 2025

Section	Change Summary
All	Initial release.

www.latticesemi.com