Avant LPDDR4 Memory Controller Driver API Reference

User Guide

FPGA-TN-02379-1.0

August 2024

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

Figures

Tables

Abbreviations in This Document

A list of aabreviations and abbreviations used in this document.

Abbreviations	Definition
LPDDR4	Low Power Double Data Rate Generation 4
SDRAM	Synchronous Dynamic Random Access Memory
SDK	Software Development Kit
API	Application Programming Interface
DDR PHY	Double Data Rate Physical Layer
CS	Chip Select
CA	Command and Address
FPGA	Field Programmable Gate Array
IP	Intellectual Property (FPGA)
DQS	Data Strobe
DQ	Data
CBT	Command Bus Training
JEDEC	Joint Electron Device Engineering Council
DBI	Data Bus Inversion
SCLK	System Clock

1. Introduction

The Lattice Semiconductor LPDDR4 Memory Controller for Avant™ Devices provides a turnkey solution consisting of a controller, DDR PHY, and associated clocking and training logic to interface with LPDDR4 SDRAM.

Refer to the Memory Controller IP Core for Avant Device User Guide (FPGA-IPUG-02208) for more details about the IP core and Avant DDR Memory PHY Module – Lattice Radiant™ Software (FPGA-IPUG-02195) for the register description.

1.1. Purpose

LPDDR4 and its SDK are a set of application programming interfaces (APIs) that provide access to specific Lattice hardware and software capabilities. We intend this document to serve as a reference guide for developers by giving details of 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 Avant devices. The technical guide assumes readers have expertise in embedded systems and FPGA technologies.

1.3. Driver and IP Compatibility

Driver version	IP version
24.01.00	2.2.0

Table 1.1. Quick Facts of Driver Tested Environment

Driver tested on HW Device	Avant Family	Avant-E70
Tool Version	Lattice Propel™ Builder	2024.1.2406150513_p
	Lattice Propel SDK	2024.1.2406150513_p
	Lattice Radiant Software	2024.1.0.34.2
	Radiant Programmer	2024.1.0.34.2

2. API Description

2.1. LPDDR4_init()

This API is used to initialize the LPDDR4 controller. This API takes the base address of LPDDR4 and initializes handle.

```
LPDDR_RET lpddr4_init(lpddr4 *instancePtr, unsigned int base_addr)
```

Table 2.1. LPDDR_init()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	"Enum LPDDR_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. LPDDR4_GetFeatureControlReg()

This API is used to read the configuration settings from the Feature Control Register. This register reflects the modes of operation specified according to the attributes selected during IP configuration and stores the result in the reg_data pointer. These attributes are set during IP configuration and cannot be modified during run-time. The CPU reads this register to identify the modes of operation. Refer to the IP user guide for the register description.

```
unsigned int lpddr4_GetFeatureControlReg(lpddr4 *instancePtr, unsigned int *reg_data)
```

Table 2.2. LPDDR4_GetFeatureControlReg()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	0: failure 1: success
Out	reg_data	Current values read from Feature Control Register.	

2.3. LPDDR4_GetSettingReg()

The Settings Register controls the LPDDR write and read latencies according to the attributes selected during IP configuration. This API is used to read latencies from settings register parameters. Refer to the IP user guide for the register description.

```
unsigned int lpddr4_GetSettingReg(lpddr4 *instancePtr, unsigned int *reg_data);
```

Table 2.3. LPDDR4_GetSettingReg()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	0: failure 1: success
Out	reg_data	Current values read from Settings Register.	

2.4. LPDDR4_GetPhyClockReg()

This API is used to read the RO parameters of PHY_CLOCK_REG. This register is used by the training CPU during clock frequency changes. The host CPU reads this register to determine the clock frequency and lock status and is not expected to write to this register. Refer to the IP user guide for the register description.

```
Unsigned int lpddr4_GetPhyClockReg(lpddr4 *InstancePtr, unsigned int *reg_data);
```

Table 2.4. LPDDR4_GetSettingReg()

In/Out	Parameter	Description	Returns
In	InstancePtr	Handle of the LPDDR4 structure.	0: failure 1: success
Out	reg_data	Current values read from PHY_CLOCK_REG.	

2.5. LPDDR4_TrainingErrorInterruptEnable()

This API enables training error interrupts. The Interrupt Enable Register lists all configurable interrupts within the Memory Controller IP. Refer to the IP user guide for the register description.

```
unsigned int lpddr4_TrainingErrorInterruptEnable(lpddr4 *instancePtr);
```

Table 2.5. LPDDR4_TrainingErrorInterruptEnable()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	0: failure 1: success

2.6. LPDDR4_TrainingErrorInterruptDisable()

This API disables training error interrupt. The Interrupt Enable Register lists all configurable interrupts within the Memory Controller IP. Refer to the IP user guide for the register description.

```
unsigned int lpddr4_TrainingErrorInterruptDisable(lpddr4 *instancePtr);
```

Table 2.6. LPDDR4_TrainingErrorInterruptDisable()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	0: failure 1: success

2.7. LPDDR4_TrainingDoneInterruptEnable()

This API enables training done interrupt. The Interrupt Enable Register lists all configurable interrupts within the Memory Controller IP. Refer to the IP user guide for the register description.

```
unsigned int lpddr4_TrainingDoneInterruptEnable(lpddr4 *instancePtr);
```

Table 2.7. LPDDR4_TrainingDoneInterruptEnable()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	0: failure 1: success

2.8. LPDDR4_TrainingDoneInterruptDisable()

This API disables training done interrupt. The Interrupt Enable Register lists all configurable interrupts within the Memory Controller IP. Refer to the IP user guide for the register description.

```
unsigned int lpddr4_TrainingDoneInterruptDisable(lpddr4 *instancePtr);
```

Table 2.8. LPDDR4_TrainingDoneInterruptDisable()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	0: failure 1: success

2.9. LPDDR4_GetTrainingOperationReg()

This API is used to read the configuration settings from the Training Operation Register. This register controls the memory initialization and training. Disable these registers (except write_lvl_en) during simulation to skip the lengthy initialization and training. There is no enable bit for self-calibrating logic because the logic is always enabled. Refer to the IP user guide for the register description.

```
unsigned int lpddr4_GetTrainingOperationReg(lpddr4 *instancePtr, unsigned int
*reg_data);
```

Table 2.9. LPDDR4_GetTrainingOperationReg()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	0: failure 1: success
Out	reg_data	Current values read from Training Operation Register.	

2.10. LPDDR4_GetStatusReg()

This API is used to read the status register. The memory controller writes to this register to communicate the status to the host CPU. Refer to the IP user guide for the register description.

```
unsigned int lpddr4_GetStatusReg(lpddr4 *instancePtr, unsigned int *reg_data);
```

Table 2.10. LPDDR4_GetStatusReg()

In/Out	Parameter	Description	Returns
In	instancePtr	Handle of the LPDDR4 structure.	0: failure 1: success
Out	reg_data	Current values read from Status Register.	

3. Function Call Flow Diagrams

3.1. LPDDR4_init()

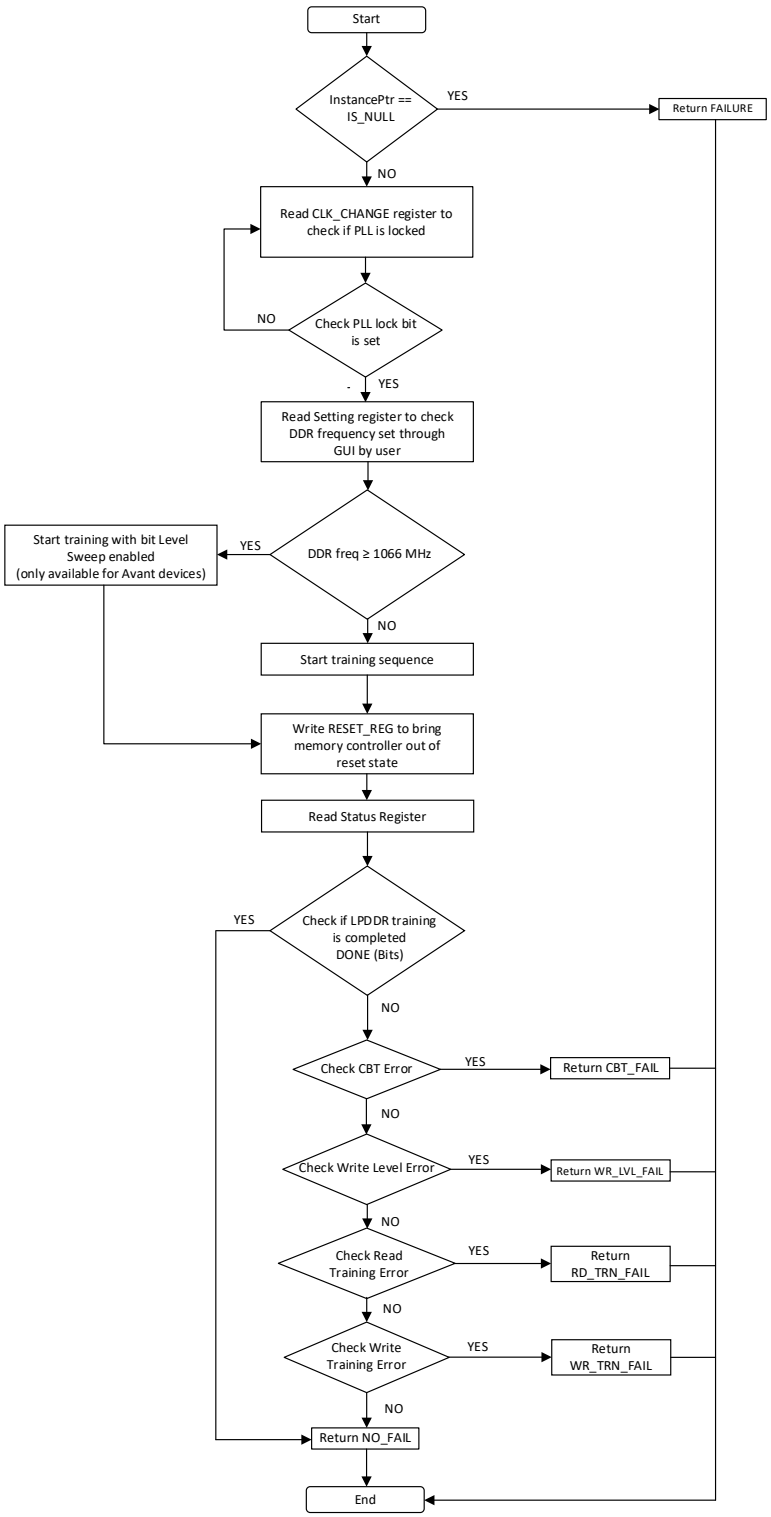

Figure 3.1. LPDDR4_init()

3.2. LPDDR4_GetFeatureControlReg()

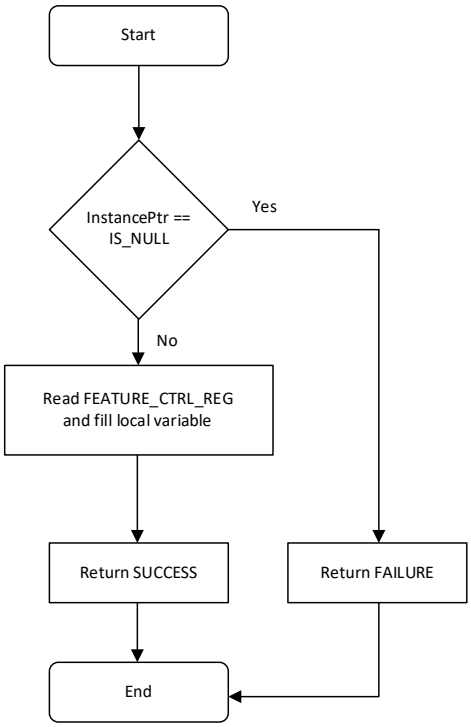

Figure 3.2. LPDDR4_GetFeatureControlReg()

3.3. LPDDR_GetSettingReg()

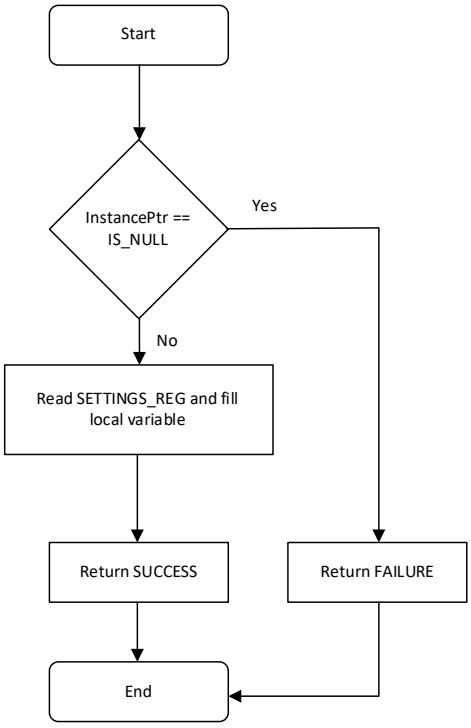

Figure 3.3. LPDDR4_GetSettingReg()

3.4. LPDDR_GetPhyClockReg()

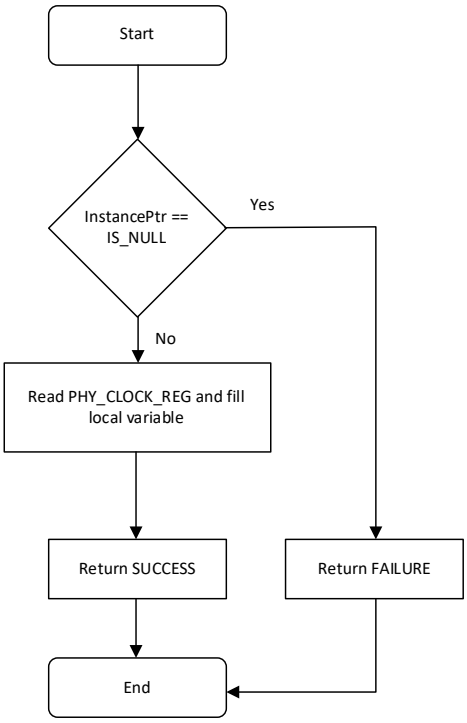

Figure 3.4. LPDDR4_GetPhyClockReg()

3.5. LPDDR4_TrainingErrorInterruptEnable()

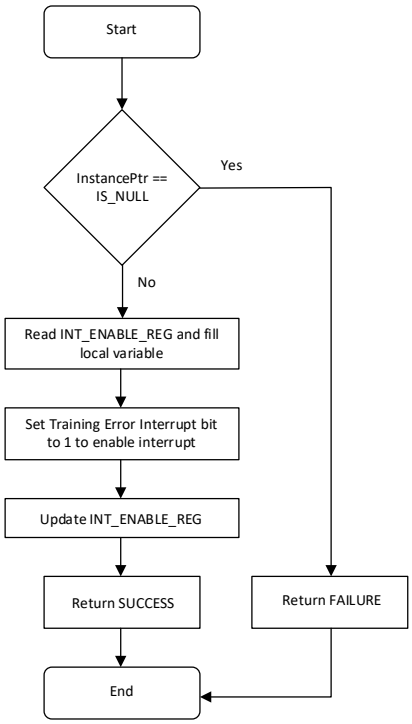

Figure 3.5. LPDDR4_TrainingErrorInterruptEnable()

3.6. LPDDR4_TrainingErrorInterruptDisable()

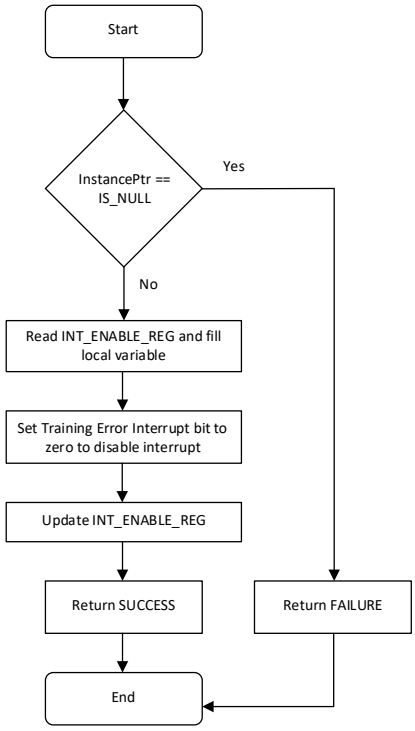

Figure 3.6. LPDDR4_TrainingErrorInterruptDisable()

3.7. LPDDR4_TrainingDoneInterruptEnable()

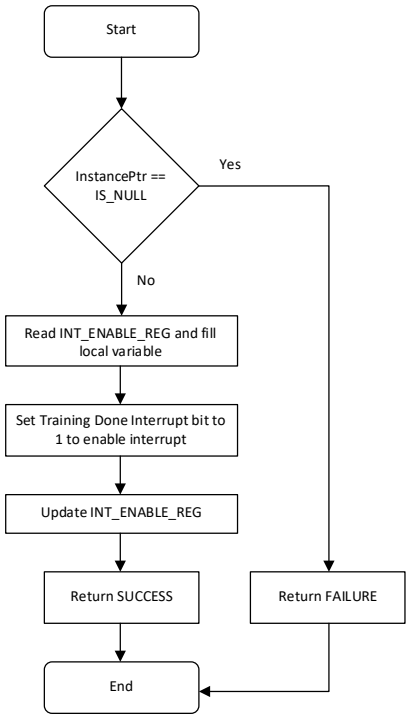

Figure 3.7. LPDDR4_TrainingDoneInterruptEnable()

3.8. LPDDR4_TrainingDoneInterruptDisable()

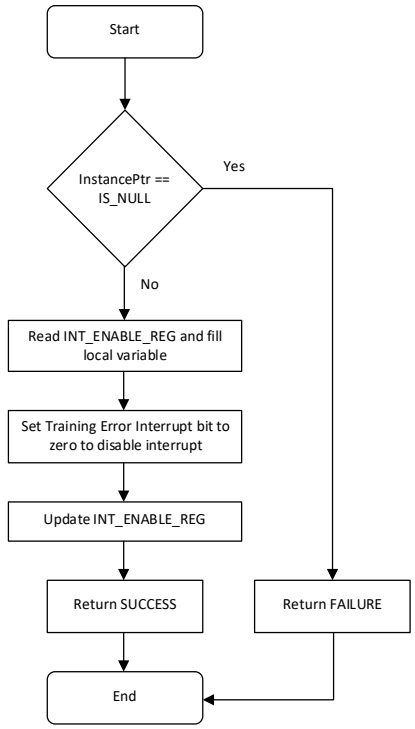

Figure 3.8. LPDDR4_TrainingDoneInterruptDisable()

3.9. LPDDR4_GetTrainingOperationReg()

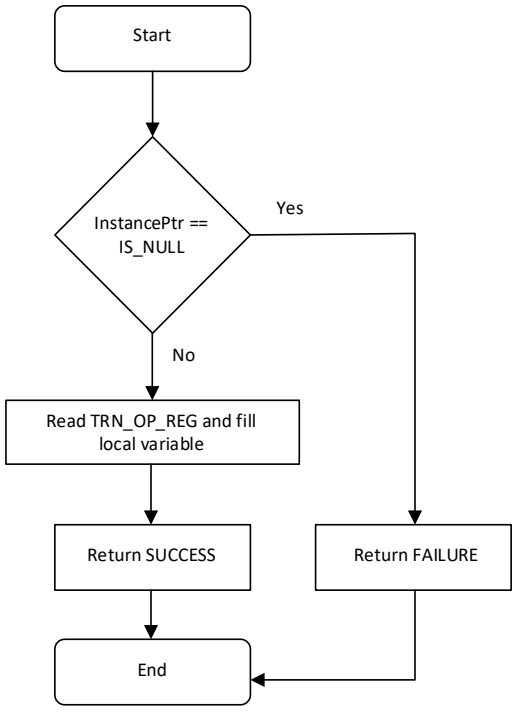

Figure 3.9. LPDDR4_GetTrainingOperationReg()

3.10. LPDDR4_GetStatusReg()

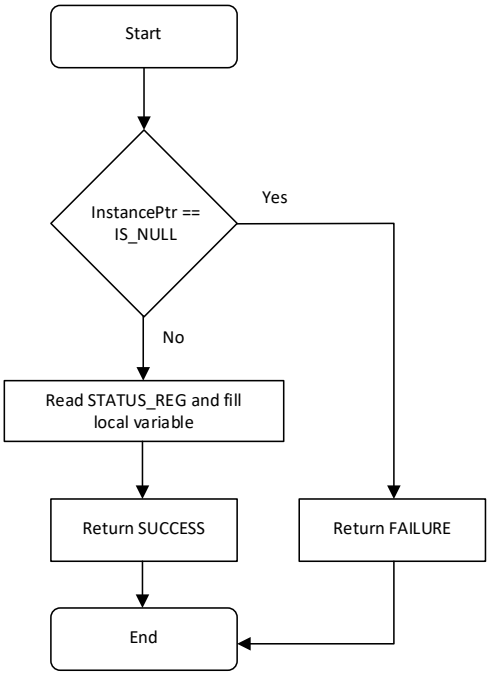

Figure 3.10. LPDDR4_GetStatusReg()

4. API Data Structures

4.1. Struct LPDDR4

Table 4.1. LPDDR4 Parameters

Data Type	Struct Member	Description
unsigned int	lpddr4_base_address	Base address of the LPDDR4 memory controller.

4.2. Union and Struct Feature_ctrl_reg_t

Refer to the IP user guide for the detailed register description.

Table 4.2. Feature_ctrl_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the feature_ctrl_reg_t register members.
unsigned int	reg	Variable to access the feature_ctrl_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	ecc_en	The ecc_en enables the sideband ECC function. This mode is based on the Enable Side-Band ECC attribute.
unsigned int	dbi_en	The dbi_en enables the data bus inversion function to reduce the toggling of DDR data signal in the board, thus improving the signal integrity and reducing dynamic power consumption. The DBI is only enabled on the read path and is always disabled on the write path to support masked write function. This mode is based on the Enable Read DBI attribute.
unsigned int	fea_rsvd_1	Reserved
unsigned int	gear_ratio	The gear_ratio specifies the ratio of clock frequency of DDR clock domain and system clock domain. This is based on the Gearing Ratio attribute. • 0 – 4:1 gearing: ddr_ck_o frequency = 2 x sclk_o frequency • 1 – 8:1 gearing: ddr_ck_o frequency = 4 x sclk_o frequency
unsigned int	fea_rsvd_2	Reserved
unsigned int	ddr_type	The ddr_type specifies the DDR standard implemented by the Memory Controller IP Core. This is based on the Interface Type attribute. • 3 – Interface Type = DDR3 • 4 – Interface Type = DDR4 • 12 – Interface Type = LPDDR4
unsigned int	ddr_width	The ddr_width specifies the bit width of the DDR data bus as follows: • 0 – DDR Bus Width = 8 bits • 1 – DDR Bus Width = 16 bits • 2 – Reserved • 3 – DDR Bus Width = 32 bits or 40 bits • 4 – 4'h6: reserved • 7 – DDR Bus Width = 64 bits or 72 bits When Enable Side-Band ECC is checked, the DDR Bus Width can be [40, 72] but the ddr_width is [3, 7]. The byte for ECC is not counted for this field.
unsigned int	num_ranks	The num_ranks specify the number of DDR ranks as follows: • 0 – Single rank • 1 – Dual rank

Data Type	Union Member	Description
unsigned int	fea_rsvd_3	Reserved
unsigned int	reset_init_by_phy	Enables the logic and routine that performs drives the ddr_reset_n_o and ddr_cke_o signals during memory initialization as well as the initial mode register writes and ZQ calibration right after the initialization. This is fixed to 1 for the first release. • 0 – Memory controller resets and initializes the memory • 1 – PHY resets and initializes the memory
unsigned int	fea_rsvd_4	Reserved

4.3. Union and Struct phy_clk_reg_t

Refer to the IP user guide for the detailed register description.

Table 4.3. Clk_chng_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the phy_clk_reg_t register members.
unsigned int	reg	Variable to access the phy_clk_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	disable_clkophy	Set to 1 to disable the PLL output clock going to the DDRPHY and ECLKSYNC hard IPs.
unsigned int	pll_lock	The cmd_freq reflects the DDR Memory Frequency attribute value. The training routine uses this value to trim initial delay settings of the DDR signals.
unsigned int	pll_reset	Used by the training CPU to reset the PLL during clock frequency change so that the PLL locks on the new frequency.
unsigned int	prim_rst_en_cfc	Enables the reset of the DDR primitives during clock frequency change.
unsigned int	pll_refclk	Returns the PLL reference clock frequency in MHz.
unsigned int	phy_clk_rsvd	Reserved

4.4. Union and Struct reset_reg_t

Refer to the IP user guide for the detailed register description.

Table 4.4. reset_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the reset_reg_t register members.
unsigned int	reg	Variable to access the reset_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	cpu_reset_n	Reset input to the training CPU. This register is on reset state at power-on. The host un-reset this register to start the memory initialization and training. Upon training completion, the internal CPU writes to this register to return the Memory Training Engine to reset state, thus saving power consumption. This register does not reset the CSR.
unsigned int	phy_reset	This is used by the training CPU to reset the DDRPHY foundation IP during initialization. Do not write 1 to this field from the APB interface
unsigned int	res_rsvd	Reserved

4.5. Union and Struct settings_reg_t

Refer to the IP user guide for the detailed register description.

Table 4.5. Settings_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the settings_reg_t register members.
unsigned int	reg	Variable to access the settings_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	write_latency	The write_latency specifies the number of DDR clock cycles from write command to the first write data.
unsigned int	read_latency	The read_latency specifies the number of DDR clock cycles from read command to the first read data.
unsigned int	cmd_freq	The cmd_freq reflects the DDR Memory Frequency attribute value. The training routine uses this value to trim initial delay settings of the DDR signals.
unsigned int	set_rsvd	Reserved

4.6. Union and Struct int_status_reg_t

Refer to the IP user guide for the detailed register description.

Table 4.6. Int_status_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the int_status_reg_t register members.
unsigned int	reg	Variable to access the int_status_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	trn_done_int	Training Done Interrupt. This Interrupt bit asserts when initialization and training is completed successfully. • 0 – No interrupt • 1 – Interrupt pending
unsigned int	trn_err_int	Training Error Interrupt. This Interrupt bit asserts when the Training Engine encounters an error during training. The user should read STATUS_REG to determine the specific error. • 0 – No interrupt • 1 – Interrupt pending
unsigned int	int_sts_rsvd	Reserved

4.7. Union and Struct status_reg_t

Refer to the IP user guide for the detailed register description.

Table 4.7. 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	The init_done asserts when the PHY initialization is complete.
unsigned int	cbt_done	Asserts when Command Bus Training has completed.
unsigned int	write_lvl_done	Asserts when write leveling has completed.
unsigned int	read_trn_done	Asserts when read training has completed.

Data Type	Union Member	Description
unsigned int	write_trn_done	Asserts when write training has completed.
unsigned int	in_self_refresh	Asserts when the memory is in self-refresh.
unsigned int	scl_done	The scl_done asserts when self-calibrating logic is complete successfully.
unsigned int	rank_0_done	Asserts when all trainings for Rank 0 are complete without error.
unsigned int	rank_1_done	Asserts when all trainings for Rank 1 are complete without error.
unsigned int	cbt_err	Asserts when a failure occurs during Command Bus Training.
unsigned int	write_lvl_err	Asserts when a failure occurs during write leveling.
unsigned int	read_trn_err	The read_trn_err asserts when there is a failure in read training.
unsigned int	write_trn_err	Asserts when a failure occurs during write training.
unsigned int	scl_err	The scl_err asserts when there is a failure in SCL.
unsigned int	trn_sts_rsvd1	Reserved
unsigned int	err_on_rank	The error_on_rank specifies the rank where the training error encountered. • 2'bx1 – Training error occurred on rank 0 • 2'b1x – Training error occurred on rank 1
unsigned int	bit_lvl_trim_sweep_done	Asserts when the bit-level trim sweep is complete.
unsigned int	trn_sts_rsvd2	Reserved
unsigned int	retraining_count	In case of training failure because of CBT error, Write Leveling error, or Read DQ bit leveling error, the Training CPU retrains with updated setting to find the Training Settings that pass the training. The maximum training retry is 5
unsigned int	trn_sts_rsvd3	Reserved

4.8. Union and Struct int_enable_reg_t

Refer to the IP user guide for the detailed Register description.

Table 4.8. int_enable_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the int_enable_reg_t register members.
unsigned int	reg	Variable to access the int_enable_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	trn_done_en	Training Done Interrupt Enable. • 0 – Interrupt disabled • 1 – Interrupt enabled
unsigned int	trn_err_en	Training Error Interrupt Enable. • 0 – Interrupt disabled • 1 – Interrupt enabled
unsigned int	int_en_rsvd	Reserved

4.9. Union and Struct trn_op_reg_t

Refer to the IP user guide for the detailed register description.

Table 4.9. trn_op_reg_t Parameters

Data Type	Union Member	Description
Struct	fields	Variable to access the trn_op_reg_tregister members.
unsigned int	reg	Variable to access the trn_op_reg_t register.
Data Type	**fields struct Members**	**Description**
unsigned int	init_en	Provides ability to shorten initialization for simulation purposes. • 0 – Initialization is greatly reduced. For example, reset time and CKE low time is greatly shortened. • 1 – Initialization is performed according to the LPDDR4 JEDEC Standard.
unsigned int	cbt_en	Enables Command Bus Training (CBT) during initialization and training. CBT performs CA_VREF programming and aligns the CS/CA and CK for high frequency operation. The command bus training is always performed as required by the LPDDR4 standard. This bit only disables the clock frequency change for faster simulation because this process requires long simulation time. • 0 – The clock frequency change is not performed during CBT. • 1 – The clock frequency change is performed during CBT.
unsigned int	write_lvl_en	The write_lvl_en enables the write leveling during initialization and training. The write leveling compensates for CK-DQS timing skews. See Write Leveling section for details in IP user guide. Write Leveling is always performed during training.
unsigned int	read_trn_en	The read_trn_en enables the read DQ bit leveling. This register optimizes the read DQ with respect to read DQS to improve the data valid window for reads. See Read DQ Bit Leveling section for details in IP user guide.
unsigned int	write_trn_en	The write_trn_en enables the write DQ bit leveling. The write training optimizes the write DQ delay with respect to the write DQS to improve the data valid window for writes. See Write DQ Bit Leveling section for details in IP user guide.
unsigned int	ca_vref_trn_en	The ca_vref_trn_en enables the memory's CA Vref training. When enabled, the DDRPHY performs CA training across multiple CA_Vref points and selects the optimal CA_Vref value. This bit is currently unused.
unsigned int	mc_verf_training_en	The mc_vref_trn_en enables the DDRPHY's DQ Vref training. When enabled, the DDRPHY performs write training across multiple Vref points of the FPGA's I/O and selects the optimal Vref value.
unsigned int	mem_vref_training_en	The mem_vref_trn_en enables the memory's DQ Vref training. When enabled, the DDRPHY performs write training across multiple memory Vref points and selects the optimal Vref value.
unsigned int	bit_lvl_trim_sweep_en	The bit_lvl_trim_sweep centers the DQ valid window to the DQS edge for each DQ bit. See Bit-Level Trim Sweep section for details in IP user guide.
unsigned int	trn_op_rsvd	Reserved

4.10. Union and Struct trim_settings_1_reg

Refer to the IP user guide for the detailed register description. The training CPU reads this register and set the corresponding DDR Hard IP registers. To try different settings on the board, you can write to this register before releasing the reset of the training CPU. This method is useful for fixing CBT error and write leveling error on the board.

Table 4.10. trim_settings_1_reg 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	ck_dly_val	Specifies the number of delay taps to be added/subtracted to the DRAM clock delay of the DDRPHY Hard IP.
unsigned int	ck_dly_val_incr	Specifies the sign when adding ck_dly_val to the DRAM clock delay of the DDRPHY Hard IP. • 0 – ck_dly_val is subtracted to the DRAM clock delay of the DDRPHY Hard IP. • 1 – ck_dly_val is added to the DRAM clock delay of the DDRPHY Hard IP.
unsigned int	trim_rsvd1	Reserved
unsigned int	adrctrl_dly_val	Specifies the number of delay taps to be added/subtracted to the address/control delay of the DDRPHY Hard IP
unsigned int	adrctrl_dly_val_inc	Specifies the sign when adding adrctrl_dly_val to the address/control delay of the DDRPHY Hard IP. • 0 – adrctrl_dly_val is subtracted to the address/control delay of the DDRPHY Hard IP. • 1 – adrctrl_dly_val is added to the address/control delay of the DDRPHY Hard IP.
unsigned int	trim_rsvd2	Reserved

4.11. Union and Struct odt_settings_reg

Refer to the IP user guide for the detailed register description. This register is currently unused for DDR3L and DDR4. The training CPU reads this register and sets the LPDDR4 memory mode register MR11 and MR22 during memory initialization.

Table 4.11. odt_settings_reg 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	ca_odt_val	Specifies the corresponding ODT settings to be written to LPDDR4 memory MR11.
unsigned int	odt_rsvd1	Reserved
unsigned int	dq_odt_val	Specifies the corresponding ODT settings to be written to LPDDR4 memory MR11.
unsigned int	soc_odt_val	Specifies the corresponding ODT settings to be written to LPDDR4 memory MR22.
unsigned int	odte_ck	
unsigned int	odte_cs	
unsigned int	odte_ca	
unsigned int	X8ODTD	
unsigned int	odt_rsvd2	Reserved

5. API Enum

5.1. Enum LPDDR_RET

Table 5.1. Qspi_reg_type_t 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 the LPDDR4 IP from the Lattice Propel Builder configuration table/sys_platform.h.

```
static lpddr4 *lpddr4_inst;
```

7. API Macros

Table 7.1. API Macros Description

Macro	Description
#define LPDDR4_DRV_VER	LPDDR4 Driver version.
#define FEATURE_CTRL_REG	Register address of Feature Control Register.
#define RESET_REG	Register address of Reset Register.
#define SETTINGS_REG	Register address of Settings Register.
#define TRN_EN	Training enable bit
#define INT_STATUS_REG	Register address of Interrupt Status Register.
#define INT_ENABLE_REG	Register address of Interrupt Enable Register.
#define INT_SET_REG	Register address of Interrupt Set Register. (for debug purpose only)
#define CLK_CHANGE	Register address of Clock Change Register.
#define TRN_OP_REG	Register address of Training Operation Register.
#define STATUS_REG	Register address of Status Register.
#define SUCCESS	Success.
#define FAILURE	Failure.
#define IS_NULL	Null value.
#define INIT_CTRL_REG	Register address of
#define PHY_CLOCK_REG	Register address of Initialization Control Register. Reserved for use by the internal CPU.
#define OUT_OF_RESET	Out of Reset Value for bringing training and CPU out of reset.
#define TIMEOUT_REG	Register address of Timeout Register.
#define TIMER_RELOAD_REG	Register address of Timer Reload Register.
#define MRW_CTRL_REG	Mode Register Write Control Register
#define SCRATCH_0_REG	Register address of SCRATCH_0_REG
#define SCRATCH_1_REG	Register address of SCRATCH_0_REG
#define TRIM_SETTINGS_1	Register address of Trim Settings 1 Register
#define ODT_SETTING	Register address of ODT Settings register
#define TRN_BIT_LVL_SWEEP_EN	Bit Level Sweep training enable
#define TRN_BIT_LVL_SWEEP_DIS	Bit Level Sweep training disable
#define DDR_FREQ	DDR frequency for Bit Level Sweep training
#define LPDDR_DONE_BITS	LPDDR initialization done bits.
#define LPDDR_ERR_DONE_BITS	LPDDR initialization done bits.
#define PLL_LOCK_BIT	LPDDR PLL lock bit

References

- Memory Controller IP Core for Avant Device User Guide (FPGA-IPUG-02208)
- Avant DDR Memory PHY Module – Lattice Radiant Software (FPGA-IPUG-02195)
- Lattice Avant Platform web page
- Lattice Avant-E web page
- Lattice Propel System Design Environment web page
- Lattice Propel Builder 2.1 User Guide (FPGA-UG-02143)
- 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 1.0, August 2024

Section	Change Summary
All	Initial release.

www.latticesemi.com