cadence

Cadence Combo PHY Core Driver Quick Start Guide

Product Version 1.0.0

April 2020

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Cadence Combo PHY Core Driver Quick Start Guide: User Guide

Cadence Design Systems, Inc.

Table of Contents

List of Tables

Chapter 1. Acronyms

API	Application Programming Interface.
Core Driver	Cadence Firmware component that provides IP programming abstraction.
CPS	Cadence Platform Services. A set of basic platform specific access functions acting as hardware abstraction layer for the core driver to operate.

Chapter 2. Document Purpose

This document is intended to serve as an overview and a reference for customers looking to use the Cadence Combo PHY Core Driver for the Cadence Cadence Combo PHY Core.

Chapter 3. Description

The Core Driver provided by Cadence is:

1. Production Code: Code built to production standards. i.e. it is fully verified with positive and negative testing.

3.1. Core Driver Features

The Cadence Combo PHY Core Driver has extended support for low speed version, enabled by adding the switch CCP_MODE=1. By default, driver will be in full version mode unless specified by CCP_MODE=1.

The *Init* function configures a number of parameters on creation of the driver instance, e.g. register base address.

Detailed data structures and APIs are listed in the later chapters of the document.

3.2. Naming conventions

The Cadence Combo PHY Core Driver API will use the prefix CCP_ and CCPEXT_ for public top-level names of functions and data objects/definitions.

Chapter 4. Core Driver Usage Model

4.1. Description

In order to support multiple instances and provide the minimum name space pollution, the Core Driver is implemented as a structure that contains a function pointer table.

A Common public header is provided to define the API object structure and the data structures common for both low speed and full version mode. The file is contained in the include directory of the distribution as *ccp_obj_if.h*. Whereas, *ccpEXT_obj_if.h* holds the extra API's required for full version mode functioning with prefix CCPEXT_
.

The API requires a private data pointer that provides the driver with instance information. It is critical that the upper software layer abides by the programming model.

Every API call apart from *probe* requires the private data address as the first parameter, to identify the driver instance and provide access to the internal state of that instance.

There is no resource protection mechanism provided in the Combo PHY Core Driver. It is expected that the higher level software stack will be responsible for protecting calls to the core driver where appropriate.

4.2. Probing the Hardware

The client code must obtain the object pointer to the core driver. It can be retrieved by issuing the *CCP_GetInstance()* function. This provides access to all of the API functions, but initially only *probe* may be called.

Before initialising the core driver, the client must first invoke the *probe* function. The *probe* function returns the dynamic memory requirements for the driver, depending on the configuration specified. The configuration information is passed to *probe* in an *CCP_Config* struct, although at this stage only those fields which affect the requirements need to be set, as indicated in the struct description. The client may repeat *probe* calls to customise configuration depending on available system resources.

The client code must then allocate the memory to the sizes returned by the probe function.

```
...
#include "ccp_structs_if.h"

CCP_SysReq sysReq = { 0 };
CCP_Config cfg = { 0 };


/* Initialise config structure fields required by probe */

uint32_t result = CCP_Probe(&cfg, &req);
if (result != CDN_EOK) {
    printf("Error: Probe function failed\n");
    return result;
}
printf("privateData object requires %u bytes\n", req.privDataSize);
...
```

4.3. Initialisation

After this stage, the client must reserve the necessary memory areas for the driver private data and hardware configuration (if required). The CCP_Config struct should be filled out with the values for all initial configuration fields.

The driver can now be initialised with an *init* call, which will:

- initialise the driver instance

- configure the hardware as specified in the *config* parameter fields

e.g. *init* example, following on from *probe* example above:

```
uint32_t readPhyReg(uint32_t address) {
    // return a value for a given Cadence Combo PHY register
}

void writePhyReg(uint32_t address, uint32_t value) {
    // write a given Cadence Combo PHY
}
...
cfg.readPhyReg = readPhyReg;
cfg.writePhyReg = writePhyReg;
CCP_PrivateData *privData = calloc(1, sysReq.privDataSize);
...
result = CCP_Init(pD, &cfg);
if (result != CDN_EOK) {
    printf("Error: Init function failed\n");
    return result;
}
```

4.4. Configuring PHY

After this stage, the client can call required CCP driver API functions according to required configuration.

```
...
CCP_PhyDqTimingReg config = { 0 };
status = CCP_GetPhyDqTimingReg(pD, &config);
CHECK_STATUS(status);

config.ioMaskAlwaysOn = 0;
config.ioMaskEnd = 1;
config.ioMaskStart = 0;
config.dataSelectOeEnd = 2;

status = CCP_SetPhyDqTimingReg(pD, &config);
CHECK_STATUS(status);
return status;
...
```

Chapter 5. Configuration and Hardware Operation Information

5.1. Introduction

The following definitions specify the driver operation environment that is defined by hardware configuration or client code.

These defines are located in the header file of the core driver.

Table 5.1. Configuration and Hardware Operation Information

#define	value	description
CCP_MAGIC_NUMBER	0x6182U	Magic number value.

Chapter 6. Dynamic Data Structures

6.1. Introduction

This section defines the data structures used by the driver to provide hardware information, modification and dynamic operation of the driver.

These data structures are defined in the header file of the core driver and utilized by the API.

6.2. CCP_SysReq_s

Type: struct

Description

"The structure that contains the resource requirements for driver operation."

Fields

`privDataSize` "@details The number of bytes required for driver operation."

6.3. CCP_Config_s

Type: struct

Description

"Configuration parameters passed to probe and init."

Fields

`readPhyReg` PHY Registry read access callback.

`writePhyReg` PHY Registry write access callback.

6.4. CCP_PrivateData_s

Type: struct

Fields

`regs`

`cfg` "Config info supplied to init."

6.5. CCP_PhyDqTimingReg_s

Type: struct

Description

Controls the DQ related timing.

Fields

`ioMaskAlwaysOn`

`ioMaskEnd`

`ioMaskStart`

`dataClkperiodDelay`

`dataSelectTselStart`	Defines the DQ pad dynamic termination select enable time.
`dataSelectTselEnd`	Adjusts the starting point of the DQ pad output enable window.
`dataSelectOeStart`	Defines the DQ pad dynamic termination select disable time.
`dataSelectOeEnd`	Adjusts the starting point of the DQ pad output enable window.

6.6. CCP_PhyDqsTimingReg_s

Type: struct

Description

Controls the DQS related timing.

Fields

`dqsClkPeriodDelay`	Defines additional latency on the write DQS path.
`useExtLpbkDqs`	Field to choose lpbk_dqs to capture data for reads.
`useLpbkDqs`	Field choose lpbk_dqs to capture data for reads.
`usePhonyDqs`	Use Phony DQS.
`usePhonyDqsCmd`	Field to choose phony DQS cmd.
`phonyDqsSel`	Phony DQS select.
`dqsSelectTselStart`	Defines the DQ pad dynamic termination select enable time.
`dqsSelectTselEnd`	Defines the DQ pad dynamic termination select disable time.
`dqsSelectOeStart`	Adjusts the starting point of the DQS pad output enable window.
`dqsSelectOeEnd`	Adjusts the ending point of the DQS pad output enable window.

6.7. CCP_PhyGateLpbkCtrlReg_s

Type: struct

Description

Controls the gate and loopback control related timing.

All the reserved fields must be set to 0.

Fields

`syncMethod`	Defines the method of transfering the data from DQS domain flops to the clk_phy clock domain.
`swDqsPhaseBypass`	DQS Phase bypass.
`enSwHalfCycle`	Enables the software half cycle shift.
`swHalfCycleShift`	Software half cycle shift effect.
`paramPhaseDetectSelOe`	DLL Phase Detect Selector for DQS OE generation.
`rdDelSel`	Defines the read data delay.
`rdDelSelEmpty`	Defines the read data delay for the empty signal generated based on the incoming DQS strobes.
`lpbkErrCheckTiming`	Sets the cycle delay between the LFSR and loopback error check logic.
`lpbkFailMuxsel`	Selects data output type for phy_obs_reg_0[23:8].
`loopbackControl`	Loopback control.
`lpbk_internal`	Controls the loopback read multiplexer.
`lpbkEn`	Controls internal write multiplexer.
`gateCfgAlwaysOn`	This parameter cause the gate to be always on.
`gateCfgClose`	Extend the closing of the DQS gate.
`gateCfg`	Coarse adjust of gate open time.

6.8. CCP_PhyDllMasterCtrlReg_s

Type: struct

Description

This register holds the control for the Master DLL logic.

Fields

`paramDllBypassMode`	This register holds the control for the Master DLL logic.
`paramPhaseDetectSel`	Selects the number of delay elements to be inserted between the phase detect flip-flops.

`paramDllLockNum` Holds the number of consecutive increment or decrement .indications.

`paramDllStartPoint` This value is the initial delay value for the DLL.

6.9. CCP_PhyDllSlaveCtrlReg_s

Type: struct

Description

This register holds the control for the slave DLL logic.

All reserved fields must be set to 0.

Fields

`readDqsCmdDelay` Controls the read command DQS delay.

`clkWrdqsDelay` Controls the clk_wrdqs delay line.

`clkWrDelay` Controls the clk_wr delay line.

`readDqsDelay` Controls the read DQS delay.

6.10. CCP_PhyIeTimingReg_s

Type: struct

Description

This register controls the DQS related timing.

Fields

`ieAlwaysOn` Forces the input enable(s) to be on always.

`dqIeStart` Define the start position for the DQ input enable.

`dqIeStop` Define the stop position for the DQ input enable.

`dqsIeStart` Define the start position for the DQS input enable.

`dqsIeStop` Define the stop position for the DQS input enable.

`rddataEnIeDly` Specifies the number of clocks of delay for the dfi_rddata_en signal.

6.11. CCP_PhyObsReg0_s

Type: struct

Description

This register holds the following observable points in the PHY.

Fields

`dqsCmdOverflow`	CMD Status signal to indicate that the logic gate was closed.
`dqsCmdUnderrun`	CMD Status signal to indicate that the logic gate had to be forced closed.
`dqsOverflow`	Status signal to indicate that the logic gate was closed too late.
`dqsUnderrun`	Status signal to indicate that the logic gate had to be forced closed.
`lpbkDqData`	Loopback DQ errors data.
`lpbkStatus`	Loopback status.

6.12. CCP_PhyDllObsReg0_s

Type: struct

Description

This register holds the following observable points in the PHY.

Fields

`lockIncDbg`	State of the cumulative dll_lock_inc register.
`lockDecDbg`	State of the cumulative dll_lock_dec register.
`dllLockValue`	Reports the number of delay elements.
`dllUnlockCount`	Number of master DLL consecutive inc/dec.
`dllLockedMode`	Indicates status of DLL. Defines the mode in which the DLL has achieved the lock.
`dllLock`	Indicates status of DLL - locked/unlocked.

6.13. CCP_PhyDllObsReg1_s

Type: struct

Description

This register holds the following observable points in the PHY.

Fields

`decoderOutWr`	Holds the encoded value for the clk_wr delay line for this slice.
`decoderOutRdCmd`	Holds the encoded value for the CMD read delay line for this slice.

`decoderOutRd` Holds the encoded value for the read delay line for this slice.

6.14. CCP_PhyDllObsReg2_s

Type: struct

Description

This register holds the following observable points in the PHY.

Fields

`decoderOutWrdqs` Holds the encoded value for the clk_wrdqs delay line for this slice.

6.15. CCP_PhyStaticToggReg_s

Type: struct

Description

This register holds the following observable points in the PHY.

Fields

`readDqsToggEnable` Enables the toggling for the active part of the read_dqs delay line in idle state.

`staticToggEnable` Holds the encoded value for the CMD read delay line for this slice.

`staticToggGlobalEnable` Global control to enable the toggle signal during static activity.

`staticTogClkDiv` Clock divider to create toggle signal.

6.16. CCP_PhyWrDeskewReg_s

Type: struct

Description

This struct holds the values of delay of each DQ bit on the write path.

Fields

`wrDq0DeskwDelay` Deskew delay for DQ bit 0.

`wrDq1DeskwDelay` Deskew delay for DQ bit 1.

`wrDq2DeskwDelay` Deskew delay for DQ bit 2.

`wrDq3DeskwDelay` Deskew delay for DQ bit 3.

`wrDq4DeskwDelay` Deskew delay for DQ bit 4.

`wrDq5DeskwDelay` Deskew delay for DQ bit 5.

`wrDq6DeskwDelay` Deskew delay for DQ bit 6.

`wrDq7DeskwDelay` Deskew delay for DQ bit 7.

6.17. CCP_PhyRdDeskewReg_s

Type: struct

Description

This struct holds the values of delay of each DQ bit on the read path.

Fields

`rdDq0DeskwDelay` Deskew delay for DQ bit 0.

`rdDq1DeskwDelay` Deskew delay for DQ bit 1.

`rdDq2DeskwDelay` Deskew delay for DQ bit 2.

`rdDq3DeskwDelay` Deskew delay for DQ bit 3.

`rdDq4DeskwDelay` Deskew delay for DQ bit 4.

`rdDq5DeskwDelay` Deskew delay for DQ bit 5.

`rdDq6DeskwDelay` Deskew delay for DQ bit 6.

`rdDq7DeskwDelay` Deskew delay for DQ bit 7.

6.18. CCP_PhyWrRdDeskewCmd_s

Type: struct

Description

This structs holds the values of delay of CMD bit on the write and read path path as well as the values of phase detect block for CMD bit on the write path.

Fields

`cmdClkperiodDelay` Defines additional latency on the CMD signal.

`cmdSwDqPhaseBypass` CMD SW DQ phase bypass.

`cmdEnSwHalfCycle` Enables the software half cycle shift.

`cmdSwHalfCycleShift` CMD SW half cycle shift.

`cmdPhaseDetectSel` DLL Phase Detect Selector for CMD generation to handle the clock domain crossing between the clock and clk_wr signal.

6.19. CCP_PhyWrDeskewPdCtrl0Dq0_s

Type: struct

Description

This struct holds the values of phase detect block for each DQ DQ bit on the write path.

Fields

`dq0SwDqPhaseBypass`	DQ0 SW Phase Bypass.
`dq0EnSwHalfCycle`	DQ0 EN SW half cycle.
`dq0SwHalfCycleShift`	DQ0 SW half cycle shift.
`dq0PhaseDetectSel`	DQ0 phase detect sel.

6.20. CCP_PhyFeaturesReg_s

Type: struct

Description

This struct holds the available hardware features.

Fields

`asfSup`	Support for Automotive Safety Feature.
`pllSup`	Support for PLL.
`jtagSup`	Support for JTAG muxes.
`extLpbkDqs`	Support for external LPBK_DQS io pad.
`regIntf`	SFR interface type.0 - DFI, 1 - APB.
`perBitDeskew`	Support for per-bit deskew.
`dfiClockRatio`	Support for clock ratio on DFI interface.
	0 - 1:1, 1 - 1:2
`aging`	Support for aging in delay lines.
`dllTapNum`	Number of taps in I/O delay lines.
	0 - 256, 1 - 512.
`bankNum`	Maximum number of banks supported by hardware.
`sdEmmc`	Support for SD/eMMC.

`xspi`	Support for XSPI.
`sdr16Bit`	Support for 16bit in ONFI SDR work mode.
`onfi41`	Support for ONFI4.1 - NAND Flash.
`onfi40`	Support for ONFI4.0 - NAND Flash.

6.21. CCP_PhyCtrlMain_s

Type: struct

Description

This register handles the global control settings for the PHY.

Please make sure all the reserved fields are set to 0.

Fields

`sdrDqsValue`	The value that should be driven on the DQS pin while SDR operations are in progress.
`phonyDqsTiming`	The timing of assertion of phony DQS to the data slices.
`ctrlClkperiodDelay`	Defines additional latency on the control signals ALE/CLE/WE/RE/CE/WP.

6.22. CCP_PhyTselReg_s

Type: struct

Description

This register handles the global control settings for the termination selects for reads.

Please make sure all the reserved fields are set to 0.

Fields

`tselOffValueData`	Termination select off value for the data.
`tselRdValueData`	Termination select read value for the data.
`tselOffValueDqs`	Termination select off value for the data strobe.
`tselRdValueDqs`	Termination select read value for the data strobe.

6.23. CCPEXT_PhyWrDeskewReg_s

Type: struct

Description

This struct holds the values of delay of each DQ bit on the write path.

Fields

`wrDq0DeskwDelay` Deskew delay for DQ bit 0.

`wrDq1DeskwDelay` Deskew delay for DQ bit 1.

`wrDq2DeskwDelay` Deskew delay for DQ bit 2.

`wrDq3DeskwDelay` Deskew delay for DQ bit 3.

`wrDq4DeskwDelay` Deskew delay for DQ bit 4.

`wrDq5DeskwDelay` Deskew delay for DQ bit 5.

`wrDq6DeskwDelay` Deskew delay for DQ bit 6.

`wrDq7DeskwDelay` Deskew delay for DQ bit 7.

6.24. CCPEXT_PhyRdDeskewReg_s

Type: struct

Description

This struct holds the values of delay of each DQ bit on the read path.

Fields

`rdDq0DeskwDelay` Deskew delay for DQ bit 0.

`rdDq1DeskwDelay` Deskew delay for DQ bit 1.

`rdDq2DeskwDelay` Deskew delay for DQ bit 2.

`rdDq3DeskwDelay` Deskew delay for DQ bit 3.

`rdDq4DeskwDelay` Deskew delay for DQ bit 4.

`rdDq5DeskwDelay` Deskew delay for DQ bit 5.

`rdDq6DeskwDelay` Deskew delay for DQ bit 6.

`rdDq7DeskwDelay` Deskew delay for DQ bit 7.

6.25. CCPEXT_PhyWrDeskewPdCtrl0Dqx_s

Type: struct

Description

This struct holds the values of phase detect block for each DQ DQ bit on the write path.

Fields

`dq3SwDqPhaseBypass` DQ3 SW Phase Bypass.

`dq3EnSwHalfCycle` DQ3 EN SW half cycle.

`dq3SwHalfCycleShift` DQ3 SW half cycle shift.

`dq3PhaseDetectSel` DQ3 phase detect sel.

`dq2SwDqPhaseBypass` DQ2 SW Phase Bypass.

`dq2EnSwHalfCycle` DQ2 EN SW half cycle.

`dq2SwHalfCycleShift` DQ2 SW half cycle shift.

`dq2PhaseDetectSel` DQ2 phase detect sel.

`dq1SwDqPhaseBypass` DQ1 SW Phase Bypass.

`dq1EnSwHalfCycle` DQ1 EN SW half cycle.

`dq1SwHalfCycleShift` DQ1 SW half cycle shift.

`dq1PhaseDetectSel` DQ1 phase detect sel.

6.26. CCPEXT_PhyWrDeskewPdCtrl1Reg_s

Type: struct

Description

This struct holds the values of phase detect block for each DQ DQ bit on the write path.

Fields

`dq7SwDqPhaseBypass` DQ7 SW Phase Bypass.

`dq7EnSwHalfCycle` DQ7 EN SW half cycle.

`dq7SwHalfCycleShift` DQ7 SW half cycle shift.

`dq7PhaseDetectSel` DQ7 phase detect sel.

`dq6SwDqPhaseBypass` DQ6 SW Phase Bypass.

`dq6EnSwHalfCycle` DQ6 EN SW half cycle.

`dq6SwHalfCycleShift` DQ6 SW half cycle shift.

`dq6PhaseDetectSel` DQ6 phase detect sel.

`dq5SwDqPhaseBypass` DQ5 SW Phase Bypass.

`dq5EnSwHalfCycle` DQ5 EN SW half cycle.

`dq5SwHalfCycleShift` DQ5 SW half cycle shift.

`dq5PhaseDetectSel` DQ5 phase detect sel.

`dq4SwDqPhaseBypass` DQ4 SW Phase Bypass.

`dq4EnSwHalfCycle` DQ4 EN SW half cycle.

`dq4SwHalfCycleShift` DQ4 SW half cycle shift.

`dq4PhaseDetectSel` DQ4 phase detect sel.

6.27. CCPEXT_PhyCtrlLowFreqSel_s

Type: struct

Description

This register handles the global control settings for the PHY.

Please make sure all the reserved fields are set to 0.

Fields

`lowFreqSel` If this field is set high the DFI interface is synchronous to the falling edge of the clock.

6.28. CCPEXT_PhyWrRdDeskewCmdExt_s

Type: struct

Description

This structs holds the values of delay of CMD bit on the write and read path path as well as the values of phase detect block for CMD bit on the write path.

Fields

`wrCmdDeskewDelay` Deskew delay (wrCmd).

`rdCmdDeskewDelay` Deskew delay (rdCmd).

6.29. CCP_ReadPhyReg

Type: typedef

This is a callback event type. A function with matching prototype needs to be registered with the `callbacks` structure at `init` to enable event signaling.

Function Declaration

```
uint32_t(* CCP_ReadPhyReg )(uint32_t address)
```

Description

Reads the requested PHY register value.

6.30. CCP_WritePhyReg

Type: typedef

This is a callback event type. A function with matching prototype needs to be registered with the `callbacks` structure at `init` to enable event signaling.

Function Declaration

```
void(* CCP_WritePhyReg )(uint32_t address, uint32_t value)
```

Description

Write the requested PHY register with the value.

Chapter 7. Driver Function API

7.1. Introduction

Prototypes for the driver API functions.

The user application can link statically to the necessary API functions and call them directly.

7.2. CCP_Probe

Function Declaration

`uint32_t CCP_Probe(`*`config`*`, `*`memReq`*`);`

```
const CCP_Config *config
```

```
CCP_SysReq *memReq
```

Description

"Get the driver memory requirements to support the given configuration.

" @param[in] config "Driver/hardware configuration required." @param[out] memReq "Returns the size of memory allocations required."

Return Value

CDN_EOK on success

CDN_EINVAL for invalid pointer

7.3. CCP_Init

Function Declaration

`uint32_t CCP_Init(`*`pD`*`, `*`config`*`);`

```
CCP_PrivateData *pD
```

```
const CCP_Config *config
```

Description

"Initialize the driver.

Must be called before all other access APIs. The init function will retain the default value in the hardware." @param[in] pD "Memory pointer to the uninitialized private data of the size specified by probe." @param[in] config "Specifies driver/hardware configuration."

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.4. CCP_SetPhyDqTimingReg

Function Declaration

`uint32_t CCP_SetPhyDqTimingReg(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD
```

```
const CCP_PhyDqTimingReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyDqTimingReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.5. CCP_GetPhyDqTimingReg

Function Declaration

`uint32_t CCP_GetPhyDqTimingReg(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCP_PhyDqTimingReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for phyDqTimingReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.6. CCP_SetPhyDqsTimingReg

Function Declaration

`uint32_t CCP_SetPhyDqsTimingReg(`*`pD`*`, `*`value`*`);`

`CCP_PrivateData *pD`

`const CCP_PhyDqsTimingReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyDqsTimingReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.7. CCP_GetPhyDqsTimingReg

Function Declaration

`uint32_t CCP_GetPhyDqsTimingReg(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCP_PhyDqsTimingReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for phyDqsTimingReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.8. CCP_SetPhyGateLpbkCtrlReg

Function Declaration

`uint32_t CCP_SetPhyGateLpbkCtrlReg(`*`pD`*`, `*`value`*`);`

`CCP_PrivateData *pD`

`const CCP_PhyGateLpbkCtrlReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyGateLpbkCtrlReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.9. CCP_GetPhyGateLpbkCtrlReg

Function Declaration

`uint32_t CCP_GetPhyGateLpbkCtrlReg(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCP_PhyGateLpbkCtrlReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyGateLpbkCtrlReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.10. CCP_SetPhyDllMasterCtrlReg

Function Declaration

`uint32_t CCP_SetPhyDllMasterCtrlReg(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD
```

```
const CCP_PhyDllMasterCtrlReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyDllMasterCtrlReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.11. CCP_GetPhyDllMasterCtrlReg

Function Declaration

`uint32_t CCP_GetPhyDllMasterCtrlReg(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD

CCP_PhyDllMasterCtrlReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyDllMasterCtrlReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.12. CCP_SetPhyDllSlaveCtrlReg

Function Declaration

`uint32_t CCP_SetPhyDllSlaveCtrlReg(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD

const CCP_PhyDllSlaveCtrlReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyDllSlaveCtrlReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.13. CCP_GetPhyDllSlaveCtrlReg

Function Declaration

`uint32_t CCP_GetPhyDllSlaveCtrlReg(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCP_PhyDllSlaveCtrlReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyDllSlaveCtrlReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.14. CCP_SetPhyIeTimingReg

Function Declaration

`uint32_t CCP_SetPhyIeTimingReg(`*`pD`*`, `*`value`*`);`

`CCP_PrivateData *pD`

`const CCP_PhyIeTimingReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyIeTimingReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.15. CCP_GetPhyIeTimingReg

Function Declaration

`uint32_t CCP_GetPhyIeTimingReg(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCP_PhyIeTimingReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyIeTimingReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.16. CCP_GetPhyObsReg0

Function Declaration

`uint32_t CCP_GetPhyObsReg0(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCP_PhyObsReg0 *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for phyObsReg0

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.17. CCP_GetPhyDllObsReg0

Function Declaration

`uint32_t CCP_GetPhyDllObsReg0(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCP_PhyDllObsReg0 *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for phyDllObsReg0

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.18. CCP_GetPhyDllObsReg1

Function Declaration

`uint32_t CCP_GetPhyDllObsReg1(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCP_PhyDllObsReg1 *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for phyDllObsReg1

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.19. CCP_GetPhyDllObsReg2

Function Declaration

`uint32_t CCP_GetPhyDllObsReg2(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCP_PhyDllObsReg2 *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for phyDllObsReg2

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.20. CCP_SetPhyStaticToggReg

Function Declaration

`uint32_t CCP_SetPhyStaticToggReg(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD

const CCP_PhyStaticToggReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyStaticToggReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.21. CCP_GetPhyStaticToggReg

Function Declaration

`uint32_t CCP_GetPhyStaticToggReg(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD

CCP_PhyStaticToggReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyStaticToggReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.22. CCP_SetPhyWrRdDeskewCmd

Function Declaration

`uint32_t CCP_SetPhyWrRdDeskewCmd(`*`pD`*`, `*`value`*`);`

`CCP_PrivateData *pD`

`const CCP_PhyWrRdDeskewCmd *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for PhyWrRdDeskewCmd

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.23. CCP_GetPhyWrRdDeskewCmd

Function Declaration

`uint32_t CCP_GetPhyWrRdDeskewCmd(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCP_PhyWrRdDeskewCmd *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyWrRdDeskewCmd

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.24. CCP_SetPhyWrDeskewPdCtrl0Dq0

Function Declaration

`uint32_t CCP_SetPhyWrDeskewPdCtrl0Dq0(`*`pD`*`, `*`value`*`);`

`CCP_PrivateData *pD`

`const CCP_PhyWrDeskewPdCtrl0Dq0 *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for PhyWrDeskewPdCtrl0Dq0

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.25. CCP_GetPhyWrDeskewPdCtrl0Dq0

Function Declaration

`uint32_t CCP_GetPhyWrDeskewPdCtrl0Dq0(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCP_PhyWrDeskewPdCtrl0Dq0 *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyWrDeskewPdCtrl0Dq0

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.26. CCP_SetPhyCtrlMain

Function Declaration

`uint32_t CCP_SetPhyCtrlMain(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD
```

```
const CCP_PhyCtrlMain *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyCtrlMain

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.27. CCP_GetPhyCtrlMain

Function Declaration

`uint32_t CCP_GetPhyCtrlMain(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCP_PhyCtrlMain *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyCtrlMain

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.28. CCP_SetPhyTselReg

Function Declaration

`uint32_t CCP_SetPhyTselReg(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD
```

```
const CCP_PhyTselReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for phyTselReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.29. CCP_GetPhyTselReg

Function Declaration

`uint32_t CCP_GetPhyTselReg(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCP_PhyTselReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyTselReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.30. CCP_SetPhyGpioCtrl0

Function Declaration

`uint32_t CCP_SetPhyGpioCtrl0(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`uint32_t value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input value

Description

setter for PhyGpioCtrl0

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.31. CCP_SetPhyGpioCtrl1

Function Declaration

`uint32_t CCP_SetPhyGpioCtrl1(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
uint32_t value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input value

Description

setter for PhyGpioCtrl1

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.32. CCP_GetPhyGpioStatus0

Function Declaration

`uint32_t CCP_GetPhyGpioStatus0(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
uint32_t *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

getter for PhyGpioStatus0

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.33. CCP_GetPhyGpioStatus1

Function Declaration

`uint32_t CCP_GetPhyGpioStatus1(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
uint32_t *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

getter for PhyGpioStatus1

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.34. CCP_WaitForDllLock

Function Declaration

`uint32_t CCP_WaitForDllLock(`*`pD`*`, `*`locked`*`, `*`delayNs`*`);`

```
const CCP_PrivateData *pD
```

```
uint32_t *locked
```

```
uint32_t delayNs
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`locked` [out]

PHY DLL lock status

`delayNs` [in]

Delay in nanoseconds

Description

Waits until Phy DLL locked bit is set.

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.35. CCP_GetPhyFeaturesReg

Function Declaration

`uint32_t CCP_GetPhyFeaturesReg(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCP_PhyFeaturesReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyFeaturesReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.36. CCPEXT_SetPhyWrDeskewPdCtrl0Dqx

Function Declaration

`uint32_t CCPEXT_SetPhyWrDeskewPdCtrl0Dqx(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD

const CCPEXT_PhyWrDeskewPdCtrl0Dqx *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for PhyWrDeskewPdCtrl0Dqx

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.37. CCPEXT_GetPhyWrDeskewPdCtrl0Dqx

Function Declaration

`uint32_t CCPEXT_GetPhyWrDeskewPdCtrl0Dqx(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD

CCPEXT_PhyWrDeskewPdCtrl0Dqx *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyWrDeskewPdCtrl0Dqx

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.38. CCPEXT_SetPhyWrDeskewReg

Function Declaration

`uint32_t CCPEXT_SetPhyWrDeskewReg(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD
```

```
const CCPEXT_PhyWrDeskewReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for PhyWrDeskewReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.39. CCPEXT_GetPhyWrDeskewReg

Function Declaration

`uint32_t CCPEXT_GetPhyWrDeskewReg(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCPEXT_PhyWrDeskewReg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyWrDeskewReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.40. CCPEXT_SetPhyRdDeskewReg

Function Declaration

`uint32_t CCPEXT_SetPhyRdDeskewReg(`*`pD`*`, `*`value`*`);`

`CCP_PrivateData *pD`

`const CCPEXT_PhyRdDeskewReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for PhyRdDeskewReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.41. CCPEXT_GetPhyRdDeskewReg

Function Declaration

`uint32_t CCPEXT_GetPhyRdDeskewReg(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCPEXT_PhyRdDeskewReg *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyRdDeskewReg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.42. CCPEXT_SetPhyWrDeskewPdCtrl1Reg

Function Declaration

`uint32_t CCPEXT_SetPhyWrDeskewPdCtrl1Reg(`*`pD`*`, `*`value`*`);`

```
CCP_PrivateData *pD
```

```
const CCPEXT_PhyWrDeskewPdCtrl1Reg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for PhyWrDeskewPdCtrl1Reg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.43. CCPEXT_GetPhyWrDeskewPdCtrl1Reg

Function Declaration

`uint32_t CCPEXT_GetPhyWrDeskewPdCtrl1Reg(`*`pD`*`, `*`value`*`);`

```
const CCP_PrivateData *pD
```

```
CCPEXT_PhyWrDeskewPdCtrl1Reg *value
```

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyWrDeskewPdCtrl1Reg

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.44. CCPEXT_SetPhyCtrlLowFreqSel

Function Declaration

`uint32_t CCPEXT_SetPhyCtrlLowFreqSel(`*`pD`*`, `*`value`*`);`

`CCP_PrivateData *pD`

`const CCPEXT_PhyCtrlLowFreqSel *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for PhyCtrlLowFreqSel

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.45. CCPEXT_GetPhyCtrlLowFreqSel

Function Declaration

`uint32_t CCPEXT_GetPhyCtrlLowFreqSel(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCPEXT_PhyCtrlLowFreqSel *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyCtrlLowFreqSel

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.46. CCPEXT_SetPhyWrRdDeskewCmdExt

Function Declaration

`uint32_t CCPEXT_SetPhyWrRdDeskewCmdExt(`*`pD`*`, `*`value`*`);`

`CCP_PrivateData *pD`

`const CCPEXT_PhyWrRdDeskewCmdExt *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [in]

input struct

Description

setter for PhyWrRdDeskewCmdExt

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL

7.47. CCPEXT_GetPhyWrRdDeskewCmdExt

Function Declaration

`uint32_t CCPEXT_GetPhyWrRdDeskewCmdExt(`*`pD`*`, `*`value`*`);`

`const CCP_PrivateData *pD`

`CCPEXT_PhyWrRdDeskewCmdExt *value`

Parameter List

`pD` [in]

Pointer to the private data initialized by init

`value` [out]

output struct

Description

getter for PhyWrRdDeskewCmdExt

Return Value

CDN_EOK on success

CDN_EINVAL if any pointer parameters are NULL