Cypress® Low Level Driver User Guide

Release 15.2.1

Doc. # 002-00329 Rev. *E

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone (USA): 800.858.1810
Phone (Intnl): +1.408.943.2600
www.cypress.com

Flash Code Protection

Cypress products meet the specifications contained in their particular Cypress Datasheets. Cypress believes that its family of products is one of the most secure families of its kind on the market today, regardless of how they are used. There may be methods, unknown to Cypress, that can breach the code protection features. Any of these methods, to our knowledge, would be dishonest and possibly illegal. Neither Cypress nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

Cypress is willing to work with the customer who is concerned about the integrity of their code. Code protection is constantly evolving. We at Cypress are committed to continuously improving the code protection features of our products.

Contents

1. Introduction

The Low Level Driver (LLD) software from Cypress is an API that provides the most basic set of functions required to communicate with a Cypress flash memory device. In most cases, there is a one to one correspondence between commands listed in the data sheet and the LLD Commands. Very little customization is necessary to make the LLD work in your system. The integration of the LLD into your system will greatly reduce your flash driver development time.

This document describes the general Cypress flash device functionalities. Not all functions are applicable to your device. Please refer to the device data sheet for applicable functions.

1.1 Files

The LLD consists of several folders for each flash device family, e.g., \S29GLxxxS\S29ALxxxj\S29ALxxxD. Each folder includes three files as follows:

- lld_xxx.c — The 'xxx' is the flash device family name, e.g., lld_S29GLxxxS.c. This file contains the Common Commands. You should not need to change this file.
- xxx.h — The 'xxx' is the flash device family name, e.g., S29GLxxxs.h. This file contains the external prototypes and the command macros. Include this file wherever you use LLD functions. You should not need to change this file.
- lld_target_specific.h – This file requires changes to work in your system.

We provide the trace.c / trace.h modules that allow you to enable the software traces, which helps a lot during debug phases.

1.2 Making the LLD Work in Your Environment

The LLD was written to support various architectures.

The file lld_target_specific.h does require modification in order to work in your environment.

In lld_target_specific.h:

1. Select the include header file for the device that you are using. For example, if the device that you are using is S29GL512S, then select S29GLxxxS.h header file.
2. Define the LLD flash chip configuration by setting the LLD_CONFIGURATION to a value that matches your system. For example, if you are using two WS256Ns (interleaved), then set the LLD_CONFIGURATION to X16_AS_X32.

```
#define LLD_CONFIGURATION_X16_AS_X16    /* no-interleaving, a single x16 device in x16 mode */
#define LLD_CONFIGURATION_X8X16_AS_X16   /* no-interleaving, a single x8/x16 device in x16 mode */
#define LLD_CONFIGURATION_X8X16_AS_X8    /* no-interleaving, a single x8/x16 device in x8 mode */
#define LLD_CONFIGURATION_X16_AS_X32     /* two x16 devices interleaved to form x32 */
#define LLD_CONFIGURATION_X8X16_AS_X32   /* two x8/x16 devices interleaved to form x32 */
#define LLD_CONFIGURATION_X8_AS_X8       /* no-interleaving, a single x8 device in x8 mode
#define LLD_CONFIGURATION_X8_AS_X32      /* special case when four X8X16 devices in X8 mode interleaving to form X32 */
#define LLD_CONFIGURATION_X8_AS_X16      /* special case when two X8X16 devices in X8 mode interleaving to form X16 */
```

```
#define LLD_CONFIGURATION_X32_AS_X32   /* no-interleaving, a single x32 device in x32 mode */
```

3. Define how the LLD will do memory reads and writes in your system. Define the macros FLASH_RD and FLASH_WR. The default macro should work for most systems.
4. The DelayMicroseconds() functions in lld_xxx.c and are based on the macro DELAY_1µs. If you choose to use the default lld_xxx.c delay functions, put a value in DELAY_1µs that will give a one-microsecond delay.
5. Define the macro PAUSE_BETWEEN_ERASE_SUSPENDS if you are using the erase suspends in your system and the time between suspends is less that 10 milliseconds and the total number of erase suspends can exceed 5000.

2. API Specification

2.1 Nomenclature, Arguments, and Typedefs

Bank
A bank (flash bank) is like a separate device. Some Cypress devices have multiple banks, thus allowing for simultaneous read (in one bank), while programming (in another bank).

Cascade
Cascade is a term used to describe a multiple flash configuration where the additional flash devices are used to increase the number of addressable locations.

Command (Cmd)
Command refers to the software implementation of a specific data sheet command.

DYB
Dynamic protection Bit. Volatile protection bit for a sector.

Interleaved
Flash is said to be interleaved when identical multiple devices are used to match the data bus size of a processor. For example, two 16-bit devices are combined to match a 32-bit system bus. On interleaved flash, some API will be executed on all multiple devices, e.g. API lld_ChipEraseOp() will erase all multiple devices.

LLD
Low Level Driver. The low level driver is the most basic set of flash functions.

Operation (Op)
An operation is defined as one or more commands combined to provide a more complete capability.

OTP
One Time Programmable. A memory area that can be programmed once and cannot be erased.

Page
The largest programmable unit for Write Buffered Programming. Pages are located on boundaries determined by the size of the page. For devices with 32 word write buffers, the page size is 32 words. In this case, pages start at addresses in which the lower five address bits are zero. Write Buffered Programming can only write to locations within a page.

PPB
Persistent Protection Bit. A non-volatile bit used to protect a sector or a sector group.

Word
Word is used to describe the smallest accessible unit of flash in your system. In a system with a single 16-bit flash, a word would be 16 bits (two bytes). In a system with four interleaved 8 bit flash devices, a word would be 32 bits (four bytes).

2.2 Arguments

base_addr
The base_addr is the starting address of the bank/device being manipulated.

offset
Offset is a measure of distance in words from the beginning of the device. For command cycles defined in the data sheet, it correlates to the “Addr” field.

2.3 Typdefs

ADDRESS
A variable type used in the code to hold addresses and offsets. Defined in xxx.h.

DEVSTATUS
A variable type used in the code to describe the flash status after an API executed. It is defined in xxx.h. For example, if the return value of API ‘lld_ProgramOp()’ is ‘DEV_NOT_BUSY’, it means flash device return status is ‘not busy’ and program completed without error.

```
typedef enum {
 DEV_STATUS_UNKNOWN = 0,
 DEV_NOT_BUSY,
 DEV_BUSY,
 DEV_EXCEEDED_TIME_LIMITS,
 DEV_SUSPEND,
 DEV_WRITE_BUFFER_ABORT,
 DEV_STATUS_GET_PROBLEM,
 DEV_VERIFY_ERROR,
 DEV_BYTES_PER_OP_WRONG,
DEV_SECTOR_LOCK,
DEV_PROGRAM_SUSPEND,
DEV_PROGRAM_SUSPEND_ERROR,
DEV_ERASE_SUSPEND,
DEV_ERASE_SUSPEND_ERROR,
DEV_BUSY_IN_OTHER_BANK,
DEV_CONTINUITY_CHECK_PATTERN_ERROR,
DEV_CONTINUITY_CHECK_NO_PATTERN_ERROR, DEV_CONTINUITY_CHECK_PATTERN_DETECTED
} DEVSTATUS;
```

FLASHDATA
A variable type used in the code to hold the smallest unit of data in your system. Its size is determined by the macro LLD_CONFIGURATION (in lld_target_specific.h. FLASHDATA is defined in xxx.h).

POLLING_TYPE
POLLING_TYPE is a type of variable used to identify the operation to the polling routine.

```
typedef enum
{
```

LLD_P_POLL_PGM = 1,

LLD_P_POLL_WRT_BUF_PGM,

LLD_P_POLL_SEC_ERS,

LLD_P_POLL_CHIP_ERS,

LLD_P_POLL_RESUME

LLD_P_POLL_BLANK

}POLLING_TYPE;

2.4 Common APIs

Notice some of the APIs listed below share the same names, but with different parameters. For instance, lld_GetDeviceID() has two forms of parameter list. The first one requires a base address while the second one needs to pass both a base address and offset address. To decide which form of APIs to use, the users need to refer to the data sheet of the specific device or related documents for more details.

2.4.1 Basic Operations

The Command API is a set of functions common to all Cypress flash devices. As we mentioned earlier, there is basically a one to one correlation between Common API functions and the commands listed in the flash data sheet. This API consists of a set of basic functions (Basic Operations) and a set of building blocks (Basic Commands).

The Basic Operations are a set of functions that provide a level of operation one step above the Basic Commands. The operation performs the desired function and poll for completion. The return value is used to determine the status of the operation.

Most of the function names of the Basic Operations end with “Op”.

Note: In systems that cannot wait for programming or erasing to finish, you will need to either implement another solution or develop non-blocking code based on our Basic Commands. In LLD, two support functions, DelayMilliseconds() and DelayMicroseconds(), have been implemented as examples. Users need to re-examine or re-implement the functions based on their own particular platforms so more accurate time delay can be achieved.

2.4.1.1 lld_GetVersion

Description:

This command is used to return LLD version number.

Returns: Version number returned in given array.

Parameters:

Name	Type	Description
versionStr[]	LLD_CHAR *	Empty char array for receiving LLD version number.

Behavior:

Note: The size of the given char array has to be at least 9 in order to avoid buffer overflow.

Related Commands: n/a

Example Code:

```
    LLD_ChAR versionStr[9];
    lld_GetVersion(versionStr);
    printf(" LLD Release Version: %s", versionStr);
```

2.4.1.2 lld_Poll – Using DQ Toggling

Description:

This function is used to poll the status of the flash after program and erase operations. In the event of a device error, this function will record the error, reset the flash (software reset) and return the status to the caller.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
Offset	ADDRESS	Offset to the location being manipulated.
exp_data_ptr	FLASHDATA *	A pointer to a variable containing the expected data.
act_data_ptr	FLASHDATA *	A pointer to a variable to store the actual data.
polling_type	POLLING_TYPE	An indication of the type of operation being performed.

Behavior:

This function will continue to poll until the operation completes or an error is detected.

Related Commands: lld_StatusGet

Example Code:

```
lld_ProgramBufferToFlashCmd(base_addr, last_loaded_addr);

  status = lld_Poll(base_addr, last_loaded_addr, &write_data,
                    &read_data, LLD_P_POLL_WRT_BUF_PGM);
  return(status);
```

2.4.1.3 lld_Poll – Using Status Register

Description:

This function is used to poll the status of the flash after program and erase operations. It will return the value of the status register. The caller routine need to check the status register bit to determine the operation result is succeed or failed.

Returns: FLASHDATA

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
Offset	ADDRESS	Offset to the location being manipulated.

Behavior:

This function will continue to poll until the operation completes or an error is detected.

Related Commands:

Example Code:

```
lld_ProgramBufferToFlashCmd(base_addr, offset);

status_reg = lld_Poll(base_addr, offset);
return(status_reg);
```

2.4.1.4 lld_StatusGetReg

Description:

This function writes the status register read command sequence to flash and reads the current value of the status register.

Returns: value of status register

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	Offset to the location being manipulated.

Behavior:

n/a

Related Commands: lld_StatusGetReg

Example Code:

```
status_reg = lld_StatusGetReg (base_addr, offset);
```

2.4.1.5 lld_StatusGet

Description:

Unlike lld_Poll, lld_StatusGet tests the status and returns immediately. This function would be a good choice in situations where non-blocking functions were required.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	Offset to the location being manipulated.

Behavior:

n/a

Related Commands: lld_Poll

Example Code:

```
do
{
    dev_status = lld_StatusGet(base_addr, offset);
}
while(dev_status == DEV_BUSY);
```

2.4.1.6 lld_StatusClear (CMD1)

Description:

This function clears the status register of the flash.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_StatusGetReg

Example Code:

```
lld_StatusClear (base_addr);
```

2.4.1.7 lld_StatusClear (CMD2)

Description:

This function clears the status register of the flash.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	Offset to the location being manipulated.

Behavior:

n/a

Related Commands: lld_StatusGetReg

Example Code:

```
lld_StatusClear (base_addr, offset);
```

2.4.1.8 lld_ProgramOp

Description:

This function programs a single word in flash and poll the status for completion. This API is implemented by lld_WriteBufferProgramOp() and with less arguments. If user want to program multiple words in one program sequence, please use lld_WriteBufferProgramOp() for better performance.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index into the flash of the location to program.
write_data	FLASHDATA	The value to program into flash.

Behavior:

If return status of this function is not ‘DEV_NOT_BUSY’, it means program process failed. Under this condition, device must be reset to return to read array mode. Program suspend will not work with this function, since this function will not return until the process is finished. Returns the device to read array mode.

Related Commands: lld_ProgramCmd, lld_Poll

Example Code:

```
status = lld_ProgramOp(addr, offset, write_data);
  printf("status = %s\n", get_status_str(status));
```

2.4.1.9 lld_WriteBufferProgramOp

Description:

This function programs words in the specified flash page and polls status for completion.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being programmed.
offset	ADDRESS	An index into the flash page.

Name	Type	Description
word_count	WORDCOUNT	Number of words (FLASHDATA elements) to program.
data_buf	FLASHDATA *	Pointer to the data to program to flash.

Behavior:

You must be familiar enough with your platform to know what the page boundaries and page sizes are for this function. Page sizes are based on the maximum number of words that can be written in Write Buffered Programming (check the data sheet) and based on your architecture's flash interleaving (check with the designer/schematics).

User can check the return status of this function to get program result.

DEV_NOT_BUSY — program process is completed and no error

DEV_SECTOR_LOCK — the sector which program address located is locked and cannot be programmed

DEV_PROGRAM_ERROR — program process is failed

Restrictions: Each Write Buffered Programming operation can only write data within a single page and can only write a maximum of LLD_BUFFER_SIZE words.

Program suspend will not work with this function, since this function will not return until the process is finished.

Related Commands: lld_WriteToBufferCmd, lld_ProgramBufferToFlashCmd

Example Code:

```
status = lld_WriteBufferProgramOp(addr, offset, word_cnt,
                                   (FLASHDATA *)source);
printf("status = %s\n", get_status_str(status));
```

2.4.1.10 lld_ChipEraseOp

Description:

This function erases the entire chip and polls for completion. In the case of interleaved devices, all are erased.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be erased.

Behavior:

Returns the device to read array mode.

Note: This command takes a long time (minutes) to complete!

Related Commands: lld_ChipEraseCmd, lld_Poll

Example Code:

```
status = lld_ChipEraseOp(addr);
  printf("status = %s\n", get_status_str(status));
```

2.4.1.11 lld_SectorEraseOp

Description:

This command erases the specified sector and waits for the process to end.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being erased.
Offset	ADDRESS	An index into the flash sector to be erased.

Behavior:

This command takes some time to complete (seconds).

Related Commands: lld_SectorEraseCmd, lld_Poll

Example Code:

```
status = lld_SectorEraseOp(addr, offset);
  printf("status = %s\n", get_status_str(status));
```

2.4.1.12 lld_ReadOp

Description:

This function reads the specified word.

Since the flash is usually memory mapped, you can read it without any special commands (its just memory). However, by funneling all the reads through this function a more consistent code base is developed. Also, some higher-level Cypress layers may require it.

Returns: FLASHDATA (Word read)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the bank/device being read.
offset	ADDRESS	An index to the location to be read.

Behavior:

No special behavior.

Related Commands: n/a

Example Code:

```
data_read = lld_ReadOp(addr, offset);
printf("%8.8X\n", data_read);
```

2.4.1.13 lld_PageReadOp

Description:

This function reads multiple words within one operation.

The read count is defined by flash device page size. Please refer to device spec for more detail.

Returns: NA

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the bank/device being read.
offset	ADDRESS	An index to the location to be read.
read_buf	FLASHDATA *	The read data buffer
cnt	FLASHDATA	The read byte/word count

Behavior:

No special behavior.

Related Commands: n/a

Example Code:

```
lld_PageReadOp(base_addr, offset, read_buf, cnt);
for (i=0;i<cnt;i++)
  printf("%8.8X\n", data_buf[i]);
```

2.4.1.14 lld_EraseSuspendOp (CMD1)

Description:

This function suspends the erase operation. The erase resume command will resume the erase operation.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be erased.

Behavior:

The device will be in erase suspend mode.

Related Commands: lld_EraseSuspendCmd, lld_EraseResumeCmd, lld_Poll

Example Code:

```
status = lld_EraseSuspendOp(base_addr);
printf("status = %s\n", get_status_str(status));
```

2.4.1.15 lld_EraseSuspendOp (CMD2)

Description:

This function suspends the erase operation. The erase resume command will resume the erase operation.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be erased.
offset	ADDRESS	An index to the flash sector to be suspended.

Behavior:

The device will be in erase suspend mode.

Related Commands: lld_EraseSuspendCmd, lld_EraseResumeCmd, lld_Poll

Example Code:

```
  status = lld_EraseSuspendOp( base_addr, offset);
  printf ( "status = %s\n", get_status_str(status) );
```

2.4.1.16 lld_ProgramSuspendOp (CMD1)

Description:

This function suspends the program operation. The program resume command will resume the program operation.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be suspended.

Behavior:

The device will be in program suspend mode.

Related Commands: lld_ProgramSuspendCmd, lld_ProgramResumeCmd, lld_Poll

Example Code:

```
status = lld_ProgramSuspendOp(base_addr);
printf("status = %s\n", get_status_str(status));
```

2.4.1.17 lld_ProgramSuspendOp (CMD2)

Description:

This function suspends the program operation. The program resume command will resume the program operation.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be checked.
offset	ADDRESS	An index to the flash sector to be suspended.

Behavior:

The device will be in program suspend mode.

Related Commands: lld_ProgramSuspendCmd, lld_ProgramResumeCmd, lld_Poll

Example Code:

```
  status = lld_ProgramSuspendOp( base_addr, offset);
  printf ( "status = %s\n", get_status_str(status) );
```

2.4.1.18 lld_BlankCheckOp

Description:

This function checks the if specified sector is blank. This function is not supported on all flash devices. Please refer flash device data sheet before using this function.

Returns: DEVSTATUS (DEV_ERASE_ERROR = no blank; DEV_NOT_BUSY = blank)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being erased.
offset	ADDRESS	An index to the flash sector to be checked.

Behavior:

This command takes some time to complete.

Related Commands: lld_SectorEraseCmd, lld_Poll

Example Code:

```
status = lld_BlankCheckOp(base_addr, offset);
  printf("status = %s\n", get_status_str(status));
```

2.4.1.19 lld_memcpy

Description:

The lld_memcpy function was added to simplify Write Buffer Programming. It is used to program memory like the lld_WriteBufferProgramOp, but the caller does not have to understand flash page sizes, boundaries, etc.

Returns: DEVSTATUS

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being programmed.
Offset	ADDRESS	An index into the flash of the first location to be programmed.
word_cnt	WORDCOUNT	Number of words to program.
data_buf	FLASHDATA *	The location of the source data.

Behavior:

The device is put into read array mode when finished. This can take a long time when there is a great deal of data. This command cannot span banks/devices.

Related Commands: lld_WriteBufferProgramOp

Example Code:

```
status = lld_memcpy(addr, offset, word_count, source);
printf("status = %s\n", get_status_str(status));
```

2.4.1.20 lld_GetDeviceId

Description:

This function reads the device ID from CFI region.

Returns: unsigned int   deviceID

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being read.

Behavior:

No special behavior.

Related Commands: n/a

Example Code:

```
data_read = lld_GetDeviceId(base_addr );
  printf("%8.8X\n", data_read);
```

2.4.1.21 lld_GetDeviceId (Device with Address Space Overlay Mode)

Description:

This function reads the device ID from CFI region.

Returns: unsigned int   deviceID

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being read.
offset	ADDRESS	Specify sector offset for ASO (Address Space Overlay).

Behavior:

No special behavior.

Related Commands: n/a

Example Code:

```
data_read = lld_GetDeviceId(base_addr, offset);
  printf("%8.8X\n", data_read);
```

2.4.2 Basic Commands

2.4.2.1 lld_ResetCmd

Description:

This command is used to return the flash to the read array mode. It is normally not necessary after programming or erase, since the flash returns to read array mode automatically when there are no problems. However, if a program or erase operation encounters an error, it will be necessary to issue an lld_ResetCmd to return the device to read array mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being reset.

Behavior:

Note: The flash Reset command, implemented by lld_ResetCmd, does not invoke a hardware reset of the flash.

Related Commands: n/a

Example Code:

```
FLASH_WR(base_addr, LLD_UNLOCK_ADDR1, NOR_CFI_QUERY_CMD); /*CFI mode*/
  data  = FLASH_RD(base_addr, offset);  /* Read CFI data */
  lld_ResetCmd(base_addr); /* return flash to read array mode */
  return(data);
```

2.4.2.2 lld_ProgramCmd

Description:

This command is used to program a single word.

Note: On devices that support Write Buffer Programming, you are expected to use Write Buffer Programming. It is possible that future Cypress flash devices will not support the Program command. If you are developing code to run on future parts AND the current part supports Write Buffer Programming, you should program the flash with the Write Buffer Program commands.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being programmed.
offset	ADDRESS	An index into the flash that correlates to the flash array element to be programmed.
pgm_data_ptr	FLASHDATA *	A pointer to the data to be used for programming.

Behavior:

When issued, this command will begin the programming process. The flash will no longer be in read array mode during programming. Typically, this command is followed by a status polling routine to determine the state of the flash.

Related Commands: lld_Poll, lld_StatusGet

Example Code:

```
lld_ProgramCmd(base_addr, offset, &write_data);
```

2.4.2.3 lld_WriteToBufferCmd

Description:

This command is used to start the Write Buffer Program sequence. It must be followed by other commands to perform Write Buffer Programming.

Note: Write Buffer Programming is faster than the legacy lld_ProgramCmd programming method, and it is the recommended way to program flash in devices that support this feature.

Returns: n/a

Parameters:

Name	**Type**	**Description**
base_addr	FLASHDATA *	The base address of the flash device being programmed.
offset	ADDRESS	An index into the flash that correlates to the Addr. field in the Command Table of the data sheet.

Behavior:

Write Buffer Programming is more complicated to use than the older lld_ProgramCmd. Make sure you read the data sheet section on Write Buffer Programming before coding. The lld_WriteBufferProgramOp or the lld_memcpy might be easier, since they are complete implementations of Write Buffer Programming.

Restrictions: Each Write Buffer Programming operation can only write data within a single page and can only write a maximum of LLD_BUFFER_SIZE words.

Related Commands: lld_ProgramBufferToFlashCmd, lld_WriteBufferProgramOp, lld_memcpy, lld_Poll, lld_StatusGet

Example Code:

```
/* Issue Load Write Buffer Command Sequence */
  lld_WriteToBufferCmd(base_addr, offset);

  /* Write # of locations to program */
  wcount *= LLD_DEV_MULTIPLIER;

  FLASH_WR(base_addr, offset, wcount);
```

2.4.2.4 lld_ProgramBufferToFlashCmd

Description:

This command is used in conjunction with the lld_WriteToBufferCmd. It is the last command issued in the Write Buffer Programming sequence.

Returns: n/a

Parameters:

Name	**Type**	**Description**
base_addr	FLASHDATA *	The base address of the flash device being programmed.
offset	ADDRESS	An index into the flash that correlates to the Addr. field in the Command Table of the data sheet.

Behavior:

The device will no longer be in read array mode during the execution of Write Buffered Programming.

Related Commands: lld_WriteToBufferCmd, lld_WriteBufferProgramOp, lld_memcpy, lld_Poll, lld_StatusGet

Example Code:

```
/* Issue Program Buffer to Flash command */
  lld_ProgramBufferToFlashCmd(base_addr, last_loaded_addr);
```

2.4.2.5 lld_WriteBufferAbortResetCmd

Description:

This command is used to reset the device to read array mode after an error causes a write buffer abort condition. See your device data sheet for a description of what conditions will cause the write buffer abort condition.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being reset.

Behavior:

n/a

Related Commands: lld_ProgramBufferToFlashCmd, lld_WriteBufferProgramOp, lld_memcpy, lld_Poll, lld_StatusGet

Example Code:

```
if(dev_status != DEV_NOT_BUSY)
  {
    if(dev_status == DEV_WRITE_BUFFER_ABORT)
    {
      lld_WriteBufferAbortResetCmd(base_addr);
    }
  }
```

2.4.2.6 lld_ChipEraseCmd

Description:

The command begins the Chip Erase process. This can take quite a while. During this process the device is not in read array mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being erased.

Behavior:

The device is not in read array mode while the erase is in progress.

Related Commands: lld_SectorEraseCmd, lld_Poll, lldStatusGet

Example Code:

```
lld_ChipEraseCmd(base_addr);
```

2.4.2.7 lld_SectorEraseCmd

Description:

This command begins a sector erase process. In terms of CPU cycles, this command will take a little time. During that time, the device will not be in read array mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being erased.
offset	ADDRESS	An index into the sector to be erased.

Behavior:

The flash will not be in read array mode during this process. This process can be suspended.

Related Commands: lld_EraseSuspendCmd, lld_EraseResumeCmd, lld_Poll, lld_StatusGet, lld_SectorEraseOp

Example Code:

```
lld_SectorEraseCmd(base_addr, offset);
```

2.4.2.8 lld_EraseSuspendCmd (CMD1)

Description:

This command is used to suspend the erase process. It is useful when reading/programming other sectors is necessary.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being suspended.
offset	ADDRESS	An index into the sector being erased.

Behavior:

A DQ polling is inserted before issue suspend command for GL-P device.

Related Commands: lld_EraseResumeCmd, lld_SectorEraseCmd

Example Code:

```
lld_EraseSuspendCmd(base_addr, offset);
```

2.4.2.9 lld_EraseSuspendCmd (CMD2)

Description:

This command is used to suspend the erase process. It is useful when reading/programming other sectors is necessary.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being suspended.

Behavior:

n/a

Related Commands: lld_EraseResumeCmd, lld_SectorEraseCmd

Example Code:

```
lld_EraseSuspendCmd(base_addr);
```

2.4.2.10 lld_EraseResumeCmd (CMD1)

Description:

This command resumes the erase process on a suspended sector.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index into the sector where the erase needs to be restarted.

Behavior:

The device will not be in read array mode any longer.

Related Commands: lld_EraseSuspendCmd, lld_SectorEraseCmd

Example Code:

```
lld_EraseSuspendCmd(base_addr, offset);
```

2.4.2.11 lld_EraseResumeCmd (CMD2)

Description:

This command resumes the erase process on a suspended sector.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

The device will not be in read array mode any longer.

Related Commands: lld_EraseSuspendCmd, lld_SectorEraseCmd

Example Code:

```
lld_EraseSuspendCmd( base_addr );
```

2.4.2.12 lld_ProgramSuspendCmd (CMD1)

Description:

This command is used to suspend the programming process. It is useful when reading other sectors is necessary.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being suspended.
offset	ADDRESS	An index to the location being programmed.

Behavior:

n/a

Related Commands: lld_ProgramResumeCmd, lld_ProgramBufferToFlashCmd

Example Code:

```
lld_ProgramSuspendCmd(base_addr, offset);
```

2.4.2.13 lld_ProgramSuspendCmd (CMD2)

Description:

This command is used to suspend the programming process. It is useful when reading/programming other sectors is necessary.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being suspended.

Behavior:

n/a

Related Commands: lld_ProgramResumeCmd, lld_ProgramBufferToFlashCmd

Example Code:

```
lld_ProgramSuspendCmd( base_addr );
```

2.4.2.14 lld_ProgramResumeCmd (CMD1)

Description:

This command resumes the programming process on the suspended location.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index into the location where the programming was occurring.

Behavior:

The device will not be in read array mode any longer.

Related Commands: lld_ProgramSuspendCmd, lld_ProgramBufferToFlashCmd

Example Code:

```
lld_ProgramSuspendCmd(base_addr, offset);
```

2.4.2.15 lld_ProgramResumeCmd (CMD2)

Description:

This command resumes the program process on the suspended location.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

The device will not be in read array mode any longer.

Related Commands: lld_ProgramSuspendCmd, lld_ProgramBufferToFlashCmd

Example Code:

```
lld_ProgramSuspendCmd(base_addr);
```

2.4.2.16 lld_StatusRegReadCmd (CMD1)

Description:

This command reads the current value of the status register.

Returns: value of status register

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a.

Related Commands: lld_StatusRegClearCmd

Example Code:

```
status_reg = lld_StatusRegReadCmd( base_addr);
```

2.4.2.17 lld_StatusRegReadCmd (CMD2)

Description:

This command reads the current value of the status register.

Returns: value of status register

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	The status read corresponds to the location specified by the offset.

Behavior:

n/a.

Related Commands: lld_StatusRegClearCmd

Example Code:

```
status_reg = lld_StatusRegReadCmd(base_addr, offset);
```

2.4.2.18 lld_StatusRegClearCmd (CMD1)

Description:

This command clears the current value of the status register.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a.

Related Commands: lld_StatusRegReadCmd

Example Code:

```
    status = lld_ProgramSuspendOp( base_addr, offset);
    lld_StatusRegClearCmd( base_addr);
```

2.4.2.19 lld_StatusRegClearCmd (CMD2)

Description:

This command clears the current value of the status register.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	The status register is cleared for the offset specified.

Behavior:

n/a.

Related Commands: lld_StatusRegReadCmd

Example Code:

```
lld_StatusRegClearCmd(base_addr, offset);
```

2.4.2.20 lld_BlankCheckCmd

Description:

This command checks if a sector is blank or not.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index into the location where to do blank check.

Behavior:

Blank check can only be issued while in array mode not in program or erase suspend mode. Reads to the array while in blank check mode is not allowed and will return unknown data.

Related Commands: lld_SectorEraseCmd

Example Code:

```
lld_BlankCheckCmd(base_addr, offset);
```

2.5 CFI Query APIs

2.5.1 CFI Query Operation

2.5.1.1 lld_ReadCfiWord

Description:

This function reads the CFI data register

Returns: FLASHDATA (Word CFI Register)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index into the flash sector to be read.

Behavior:

n/a

Related Commands: lld_CfiEntryCmd, lld_CfiExitCmd

Example Code:

```
data = lld_ReadCfiWord(base_addr, offset);
  printf("%8.8X\n", data);
```

2.5.2 CFI Query Commands

2.5.2.1 lld_CfiEntryCmd

Description:

This command causes the CFI data to be available in the first sector of the specified bank.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

The first sector of the device specified in the base_addr parameter will be replaced with the CFI data. Use the lld_CfiExitCmd to return to read array mode.

Related Commands: lld_CfiExitCmd

Example Code:

```
lld_CfiEntryCmd(base_addr);
```

2.5.2.2 lld_CfiEntryCmd

Description:

This command causes the CFI data to be available in the specified sector of the specified bank.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	Sector offset for ASO (Address Space Overlay).

Behavior:

The first sector of the device specified in the base_addr parameter will be replaced with the CFI data. Use the lld_CfiExitCmd to return to read array mode.

Related Commands: lld_CfiExitCmd

Example Code:

```
lld_CfiEntryCmd(base_addr, offset);
```

2.5.2.3 lld_CfiExitCmd

Description:

This command exits CFI mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

Returns the device to read array mode.

Related Commands: lld_CfiEntryCmd

Example Code:

```
lld_CfiExitCmd(base_addr);
```

2.6 Autoselect APIs

2.6.1 Autoselect Commands

2.6.1.1 lld_AutoselectEntryCmd

Description:

This command replaces the first sector with the Autoselect information.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

The device will no longer be in read array mode.

Related Commands: lld_AutoselectExitCmd

Example Code:

```
lld_AutoselectEntryCmd(base_addr);
```

2.6.1.2 lld_AutoselectEntryCmd (Device with Address Space Overlay Mode)

Description:

This command replaces the specified sector with the Autoselect information.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	Sector offset for ASO (Address Space Overlay).

Behavior:

The device will no longer be in read array mode.

Related Commands: lld_AutoselectExitCmd

Example Code:

```
lld_AutoselectEntryCmd(base_addr, offset);
```

2.6.1.3 lld_AutoselectExitCmd

Description:

This command returns the device/bank to read array mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

Returns the device to read array mode.

Related Commands: lld_AutoselectEntryCmd

Example Code:

```
lld_AutoselectExitCmd(base_addr);
```

2.7 Unlock Bypass APIs

2.7.1 Unlock Bypass Commands

2.7.1.1 lld_UnlockBypassEntryCmd

Description:

This command puts the flash state machine in a mode where it will accept minimum cycle commands.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

Use only Unlock Bypass Commands while in this mode.

Related Commands: lld_UnlockBypassExitCmd, lld_UnlockBypassProgramCmd

Example Code:

```
lld_UnlockBypassEntryCmd(addr);
```

2.7.1.2 lld_UnlockBypassProgramCmd

Description:

This command is a faster version of the lld_ProgramCmd.

Note: Like lld_ProgramCmd, lld_UnlockBypassProgramCmd should not be used in systems that provide Write Buffer Programming.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be programmed.
offset	ADDRESS	An index to the location to be programmed.
pgm_data_ptr	FLASHDATA *	Pointer to the data to program.

Behavior:

This command can only be used in Unlock Bypass mode.

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassExitCmd

Example Code:

```
lld_UnlockBypassProgramCmd(addr, offset, &data);
```

2.7.1.3 lld_UnlockBypassResetCmd

Description:

This command returns the flash state machine to the standard (non-Unlock Bypass) mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassProgramCmd

Example Code:

```
lld_UnlockBypassResetCmd(base_addr);
```

2.7.1.4 lld_UnlockBypassWriteToBufferCmd

Description:

This command is a fast version of Write To Buffer

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	An index to the location to be programmed.
word_count	WORDCOUNT	Number of words (not bytes) to program.
data_buf	FLASHDATA*	Pointer to the data to program.

Behavior:

This command can only be used in Unlock Bypass mode.

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassResetCmd

Example Code:

```
lld_UnlockBypassWriteToBufferCmd(base_addr, offset, word_count, &data_buf);
```

2.7.1.5 lld_UnlockBypassProgramBufferToFlashCmd

Description:

This command is a fast version of Program Buffer to Flash

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	An index to the location to be programmed.

Behavior:

This command can only be used in Unlock Bypass mode.

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassResetCmd

Example Code:

```
lld_UnlockBypassProgramBufferToFlashCmd (base_addr, offset);
```

2.7.1.6 lld_UnlockBypassWriteToBufferAbortResetCmd

Description:

This command is a fast version of Write To Buffer Abort Reset

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

This command can only be used in Unlock Bypass mode.

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassResetCmd

Example Code:

```
lld_UnlockBypassWriteToBufferAbortResetCmd (base_addr);
```

2.7.1.7 lld_UnlockBypassSectorEraseCmd

Description:

This command is a fast version of Sector Erase

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	An index to the location to be programmed.

Behavior:

This command can only be used in Unlock Bypass mode.

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassResetCmd

Example Code:

```
lld_UnlockBypassSectorEraseCmd (base_addr, offset);
```

2.7.1.8 lld_UnlockBypassChipEraseCmd

Description:

This command is a fast version of Chip Erase

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

This command can only be used in Unlock Bypass mode.

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassResetCmd

Example Code:

```
lld_UnlockBypassChipEraseCmd (base_addr);
```

2.7.2 Unlock Bypass Operations

2.7.2.1 lld_UnlockBypassProgramOp

Description:

This command is a fast version of Program Op

Returns: DEVSTATUS (Program Complete, Program Error)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	An index to the location to be programmed.
write_data	FLASHDATA	The data to program.

Behavior:

This command can only be used in Unlock Bypass mode.

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassResetCmd

Example Code:

```
lld_UnlockBypassEntryCmd(base_addr);
status = lld_ UnlockBypassProgramOp (base_addr, offset, write_data);
printf("status = %s\n", get_status_str(status));
lld_UnlockBypassResetCmd(base_addr);
```

2.7.2.2 lld_UnlockBypassBufferWriteProgramOp

Description:

This command is a fast version of Buffer Write Program Op

Returns: DEVSTATUS (Program Complete, Program Error)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	An index to the location to be programmed.
word_count	WORDCOUNT	Number of words (not bytes) to program.
data_buf	FLASHDATA*	Pointer to the data to program.

Behavior:

This command can only be used in Unlock Bypass mode.

Related Commands: lld_UnlockBypassEntryCmd, lld_UnlockBypassResetCmd

Example Code:

```
lld_UnlockBypassEntryCmd(base_addr);
status = lld_UnlockBypassBufferWriteProgramOp (base_addr, offset, word_count, &data_buf);
```

2.8 Sector Protection APIs

2.8.1 SecSi Sector Commands

Upon SecSiSectorEntryCmd, the first sector is replaced by the SecSi sector. Once in this mode, use the standard programming and reading commands.

2.8.1.1 lld_SecSiSectorEntryCmd

Description:

This command grants access to the SecSi sector.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

You should read and understand the data sheet section on the SecSi sector before writing to this OTP area.

Related Commands: lld_SecSiSectorExitCmd

Example Code:

```
lld_SecSiSectorEntryCmd(addr);
```

2.8.1.2 lld_SecSiSectorEntryCmd (Device with Address Space Overlay Mode)

Description:

This command grants access to the SecSi sector.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	sector offset for ASO (Address Space Overlay).

Behavior:

You should read and understand the data sheet section on the SecSi sector before writing to this OTP area.

Related Commands: lld_SecSiSectorExitCmd

Example Code:

```
lld_SecSiSectorEntryCmd(base_addr, offset);
```

2.8.1.3 lld_SecSiSectorExitCmd

Description:

This command restores the first sector with read array data (from SecSi sector data)

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_SecSiSectorEntryCmd

Example Code:

```
lld_SecSiSectorExitCmd(base_addr);
```

2.8.2 Lock Register Operations

2.8.2.1 lld_LockRegBitsReadOp

Description:

This function returns the value of Lock Register.

Returns: FLASHDATA (Lock Register Word)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_LockRegBitsProgramCmd, lld_LockRegEntryCmd, lld_LockRegExitCmd

Example Code:

```
data = lld_LockRegBitsReadOp(addr);
printf("%8.8X\n", data);
```

2.8.2.2 lld_SSRLockRegBitsReadOp

Description:

This function reads the Secure Silicon Region (SSR) lock register.

Returns: FLASHDATA (Word Lock Register)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being read.
offset	ADDRESS	An index into the flash sector to be read.

Behavior:

n/a

Related Commands: lld_SSRLockRegEntryCmd, lld_SSRLockRegExitCmd, lld_SSRLockRegBitsProgramCmd

Example Code:

```
data = lld_SSRLockRegBitsReadOp(base_addr, offset);
printf("%8.8X\n", data);
```

2.8.2.3 lld_LockRegBitsProgramOp

Description:

This function programs the Lock Register with a value. Refer to the data sheet for bit definitions.

Returns: 0 = Operation successful: 1 = Operation failed

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
value	FLASHDATA	The Lock Register value to be programmed.

Behavior:

n/a

Related Commands: lld_LockRegEntryCmd, lld_LockRegBitsReadCmd, lld_LockRegExitCmd

Example Code:

```
lld_LockRegBitsProgramOp(addr, value);
```

2.8.2.4 lld_SSRLockRegBitsProgramOp

Description:

This function programs the Secure Silicon Region (SSR) lock register.

Returns: DEVSTATUS (0 = Program Completed; DEV_PROGRAM_ERROR = Program Error)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being programmed.
offset	ADDRESS	An index into the flash sector to be programmed.
write_data	FLASHDATA	The value to program into flash.

Behavior:

n/a

Related Commands: lld_SSRLockRegEntryCmd, lld_SSRLockRegExitCmd, lld_LockRegBitsReadCmd

Example Code:

```
status = lld_SSRLockRegBitsProgramOp(addr, offset, write_data);
printf("status = %s\n", get_status_str(status));
```

2.8.2.5 lld_PpbAllEraseOp

Description:

This function un-protects all the PPB bits for sectors/sector groups.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

All the PPB bits are erased at once.

Related Commands: lld_PpbEntryCmd, lld_PpbExitCmd, lld_Poll

Example Code:

```
lld_PpbAllEraseOp(addr);
```

2.8.2.6 lld_PpbProgramOp

Description:

This function sets the PPB protection for a sector/sector group. When set, the PPB Status Read will return 0 (protected), otherwise it will return 1 (unprotected).

Returns: DEVSTATUS (0 = Program Completed; -1 = Program Error)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index to the flash sector to be protected.

Behavior:

n/a

Related Commands: lld_PpbEntryCmd, lld_PpbExitCmd, lld_PpbLockBitReadOp, lld_Poll

Example Code:

```
op_status = lld_PpbProgramOp(addr, offset);
printf("%X\n", op_status);
```

2.8.2.7 lld_PpbStatusReadOp

Description:

This function reads the status of the PPB Protection Bit for the addressed sector/sector group.

Returns: FLASHDATA (0 = protected, 1 = unprotected)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index to the sector/sector group.

Behavior:

n/a

Related Commands: lld_PpbEntryCmd, lld_PpbExitCmd, lld_PpbAllEraseCmd, lld_PpbProgramCmd

Example Code:

```
data = lld_PpbStatusReadOp(addr, offset);
  printf("%8.8X\n", data);
```

2.8.3 Lock Register Commands

2.8.3.1 lld_LockRegEntryCmd

Description:

This mode of operation is used to read and program the Lock Register Bits. Non-Lock Register commands should not be used while in this mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_LockRegBitsProgramCmd, lld_LockRegBitsReadCmd, lld_LockRegExitCmd

Example Code:

```
lld_LockRegEntryCmd(addr);
```

2.8.3.2 lld_SSRLockRegEntryCmd (Device with Address Space Overlay Mode)

Description:

This mode of operation is used to read and program the Lock Register Bits.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
Offset	ADDRESS	sector offset for ASO (Address Space Overlay).

Behavior:

n/a

Related Commands: lld_SSRLockRegBitsProgramCmd, lld_SSRLockRegBitsReadCmd, lld_SSRLockRegExitCmd

Example Code:

```
lld_SSRLockRegEntryCmd(base_addr, offset);
```

2.8.3.3 lld_LockRegBitsProgramCmd

Description:

Programs the Lock Register with a value. Refer to the data sheet for bit definitions.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
Value	FLASHDATA	The Lock Register value to be programmed.

Behavior:

n/a

Related Commands: lld_LockRegEntryCmd, lld_LockRegBitsReadCmd, lld_LockRegExitCmd

Example Code:

```
lld_LockRegBitsProgramCmd(addr, value);
```

2.8.3.4 lld_SSRLockRegBitsProgramCmd (Device with Address Space Overlay Mode)

Description:

Programs the Lock Register with a value. Refer to the data sheet for bit definitions.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	sector offset for ASO (Address Space Overlay).
value	FLASHDATA	The Lock Register value to be programmed.

Behavior:

n/a

Related Commands: lld_SSRLockRegEntryCmd, lld_SSRLockRegBitsReadCmd, lld_SSRLockRegExitCmd

Example Code:

```
lld_SSRLockRegBitsProgramCmd(base_addr, offset,  value);
```

2.8.3.5 lld_LockRegBitsReadCmd

Description:

This command returns the value of the Lock Register.

Returns: FLASHDATA (Lock Register Word)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_LockRegBitsProgramCmd, lld_LockRegEntryCmd, lld_LockRegExitCmd

Example Code:

```
data = lld_LockRegBitsReadCmd(addr);
printf("%8.8X\n", data);
```

2.8.3.6 lld_SSRLockRegBitsReadCmd (Device with Address Space Overlay Mode)

Description:

This command returns the value of the Lock Register.

Returns: FLASHDATA (Lock Register Word)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	sector offset for ASO (Address Space Overlay).

Behavior:

n/a

Related Commands: lld_SSRLockRegBitsProgramCmd, lld_SSRLockRegEntryCmd, lld_SSRLockRegExitCmd

Example Code:

```
data = lld_SSRLockRegBitsReadCmd(base_addr, offset);
  printf("%8.8X\n", data);
```

2.8.3.7 lld_LockRegExitCmd

Description:

This command exits the Lock Register mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_LockRegBitsProgramCmd, lld_LockRegBitsReadCmd, lld_LockRegEntryCmd

Example Code:

```
lld_LockRegExitCmd(addr);
```

2.8.3.8 lld_SSRLockRegExitCmd (Device with Address Space Overlay Mode)

Description:

This command exits the SSR Lock Register mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior: n/a

Related Commands: lld_SSRLockRegBitsProgramCmd, lld_SSRLockRegBitsReadCmd, lld_SSRLockRegEntryCmd,

Example Code:

```
lld_SSRLockRegExitCmd(addr);
```

2.8.4 Password Protection Mode Commands

2.8.4.1 lld_PasswordProtectionEntryCmd

Description:

This command puts the state machine in Password Protection Modification mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

Only Password Protection Mode commands should be issued while in this mode.

Related Commands: lld_PasswordProtectionProgramCmd, lld_PasswordProtectionReadCmd, lld_PasswordProtectionUnlockCmd, lld_PasswordProtectionExitCmd

Example Code:

```
lld_PasswordProtectionEntryCmd(addr);
```

2.8.4.2 lld_PasswordProtectionProgramCmd

Description:

This command is used to program the password once the device is in the Password Protection Modification mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	The offset of the password to write. There are four separate passwords (offsets 0 - 3).
pwd	FLASHDATA	Password word.

Behavior:

You should read the data sheet about this feature, because this is OTP memory - you only get one chance.

Related Commands: lld_PasswordProtectionEntryCmd, lld_PasswordProtectionReadCmd, lld_PasswordProtectionUnlockCmd, lld_PasswordProtectionExitCmd

Example Code:

```
lld_PasswordProtectionProgramCmd(addr, 0, pwd1);
lld_PasswordProtectionProgramCmd(addr, 1, pwd2);
lld_PasswordProtectionProgramCmd(addr, 2, pwd3);
lld_PasswordProtectionProgramCmd(addr, 3, pwd4);
```

2.8.4.3 lld_PasswordProtectionReadCmd

Description:

This function issues the read password command.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
pwd0	FLASHDATA *	A pointer to a location to store the first password word.
pwd1	FLASHDATA *	A pointer to a location to store the second password word.
pwd2	FLASHDATA *	A pointer to a location to store the third password word.
pwd3	FLASHDATA *	A pointer to a location to store the forth password word.

Behavior:

This command will not return the password after Password Protection mode is committed.

Related Commands: lld_PasswordProtectionProgramCmd, lld_PasswordProtectionEntryCmd, lld_PasswordProtectionUnlockCmd, lld_PasswordProtectionExitCmd

Example Code:

```
lld_PasswordProtectionReadCmd(addr, &pwd0, &pwd1, &pwd2, &pwd3);
  printf("%8.8X %8.8X %8.8X %8.8X\n", pwd0, pwd1, pwd2, pwd3);
```

2.8.4.4 lld_PasswordProtectionUnlockCmd

Description:

This command presents the password to the flash. There is no indication of success.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
pwd0	FLASHDATA	The first word of the password.
pwd1	FLASHDATA	The second word of the password.
pwd2	FLASHDATA	The third word of the password.
pwd3	FLASHDATA	The forth word of the password.

Behavior:

n/a

Related Commands: lld_PasswordProtectionProgramCmd, lld_PasswordProtectionReadCmd, lld_PasswordProtectionEntryCmd, lld_PasswordProtectionExitCmd

Example Code:

```
lld_PasswordProtectionUnlockCmd(addr, pwd0, pwd1, pwd2, pwd3);
```

2.8.4.5 lld_PasswordProtectionExitCmd

Description:

This command exits the Password Protection Manipulation Mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

Resets the device to read array mode.

Related Commands: lld_PasswordProtectionProgramCmd, lld_PasswordProtectionReadCmd, lld_PasswordProtectionUnlockCmd, lld_PasswordProtectionEntryCmd

Example Code:

```
lld_PasswordProtectionExitCmd(addr);
```

2.8.5 PPB Commands

2.8.5.1 lld_PpbEntryCmd

Description:

This command put the flash into PPB Command Set Mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_PpbStatusReadCmd, lld_PpbExitCmd, lld_PpbAllEraseCmd, lld_PpbProgramCmd

Example Code:

```
lld_PpbEntryCmd(addr);
```

2.8.5.2 lld_PpbProgramCmd

Description:

This command sets the PPB protection for a sector/sector group. When set, the PPB Status Read will return 0 (protected), otherwise it will return 1 (unprotected).

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index to the flash sector to be protected.

Behavior:

n/a

Related Commands: lld_PpbEntryCmd, lld_PpbExitCmd, lld_PpbAllEraseCmd, lld_PpbStatusReadCmd

Example Code:

```
lld_PpbProgramCmd(addr, offset);
```

2.8.5.3 lld_PpbAllEraseCmd

Description:

This command un-protects all the PPB bits for sectors/sector groups.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

All the PPB bits are erased at once.

Related Commands: lld_PpbEntryCmd, lld_PpbExitCmd, lld_PpbStatusReadCmd, lld_PpbProgramCmd

Example Code:

```
lld_PpbAllEraseCmd(addr);
```

2.8.5.4 lld_PpbStatusReadCmd

Description:

Reads the status of the PPB Protection Bit for the addressed sector/sector group.

Returns: FLASHDATA (0 = protected, 1 = unprotected)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index to the sector/sector group.

Behavior:

n/a

Related Commands: lld_PpbEntryCmd, lld_PpbExitCmd, lld_PpbAllEraseCmd, lld_PpbProgramCmd

Example Code:

```
data = lld_PpbStatusReadCmd(addr, offset);
  printf("%8.8X\n", data);
```

2.8.5.5 lld_PpbExitCmd

Description:

This command exits the PPB Command Set Mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_PpbEntryCmd, lld_PpbStatusReadCmd, lld_PpbAllEraseCmd, lld_PpbProgramCmd

Example Code:

```
lld_PpbExitCmd(addr);
```

2.8.5.6 lld_PpbSAProtectStatusCmd

Description:

This command is used to Read PPB SA Protect Status.

Returns: FLASHDATA (Word array data)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	Sector offset for ASO (Address Space Overlay).

Behavior: n/a

Related Commands: lld_PpbEntryCmd, lld_PpbProgramCmd, lld_PpbAllEraseCmd,

lld_PpbStatusReadCmd, lld_PpbExitCmd

Example Code:

```
sa_protect_status = lld_PpbSAProtectStatusCmd (base_addr, offset);
```

2.8.6 DYB Commands

2.8.6.1 lld_DybEntryCmd

Description:

This command enters the DYB Protection Command Set Mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_DybSetCmd, lld_DybClrCmd, lld_DybReadCmd, lld_DybExitCmd

Example Code:

```
lld_DybEntryCmd(addr);
```

2.8.6.2 lld_DybSetCmd

Description:

This command sets the DYB to 0 (protected).

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index to the sector to be protected.

Behavior:

n/a

Related Commands: lld_DybEntryCmd, lld_DybClrCmd, lld_DybReadCmd, lld_DybExitCmd

Example Code:

```
lld_DybSetCmd(addr, offset);
```

2.8.6.3 lld_DybClrCmd

Description:

This command un-protects the appropriate DYB.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index to the sector to be un-protected.

Behavior:

n/a

Related Commands: lld_DybEntryCmd, lld_DybSetCmd, lld_DybReadCmd, lld_DybExitCmd

Example Code:

```
lld_DybClrCmd(addr, offset);
```

2.8.6.4 lld_DybReadCmd

Description:

This command reads the value of the sector's DYB bit.

Returns: FLASHDATA (0=protected, 1=un-protected)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	An index to the appropriate sector.

Behavior:

n/a

Related Commands: lld_DybEntryCmd, lld_DybSetCmd, lld_DybClrCmd, lld_DybExitCmd

Example Code:

```
data   = lld_DybReadCmd(addr, offset);
  printf("%8.8X\n", data);
```

2.8.6.5 lld_DybExitCmd

Description:

This commands exits the DYB Protection Command Set Mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_DybEntryCmd, lld_DybSetCmd, lld_DybClrCmd, lld_DybReadCmd

Example Code:

```
lld_DybExitCmd(addr);
```

2.8.6.6 2.8.6.6 lld_DybSAProtectStatusCmd

Description:

This command is used to Read DYB SA Protect Status.

Returns: FLASHDATA (Word array data)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	ADDRESS	Sector offset for ASO (Address Space Overlay).

Behavior: n/a

Related Commands: lld_DybEntryCmd, lld_DybSetCmd, lld_DybClrCmd, lld_DybReadCmd,

lld_DybExitCmd

Example Code:

```
sa_protect_status = lld_DybSAProtectStatusCmd (base_addr, offset);
```

2.8.7 PPB Lock Bit Operation

2.8.7.1 lld_PpbLockBitReadOp

Description:

This function reads the value of the PPB Lock Bit.

Returns: FLASHDATA (0=PPB Protection selected, 1=not selected)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Behavior:

n/a

Related Commands: lld_PpbLockBitEntryCmd, lld_PpbLockBitExitCmd

Example Code:

```
data = lld_PpbLockBitReadOp(addr);
  printf("%8.8X\n", data);
```

2.8.7.2 lld_PpbLockBitSetOp

Description:

This function sets the Flash Protection Mode to PPB Mode (as opposed to Password Protection Mode).

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_PpbLockBitEntryCmd, lld_PpbLockBitExitCmd, lld_Poll

Example Code:

```
lld_PpbLockBitSetOp(addr);
```

2.8.8 PPB Lock Bit Commands

2.8.8.1 lld_PpbLockBitEntryCmd

Description:

This command enters the PPB Lock Bit Manipulation Mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_PpbLockBitSetCmd, lld_PpbLockBitReadCmd, lld_PpbLockBitExitCmd

Example Code:

```
lld_PpbLockBitEntryCmd(addr);
```

2.8.8.2 lld_PpbLockBitSetCmd

Description:

This command sets the Flash Protection Mode to PPB Mode (as opposed to Password Protection Mode).

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_PpbLockBitEntryCmd, lld_PpbLockBitReadCmd, lld_PpbLockBitExitCmd

Example Code:

```
lld_PpbLockBitSetCmd(addr);
```

2.8.8.3 lld_PpbLockBitReadCmd

Description:

This command read the value of the PPB Lock Bit.

Returns: FLASHDATA (0=PPB Protection selected, 1=not selected)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_PpbLockBitEntryCmd, lld_PpbLockBitSetCmd, lld_PpbLockBitExitCmd

Example Code:

```
data = lld_PpbLockBitReadCmd(addr);
  printf("%8.8X\n", data);
```

2.8.8.4 lld_PpbLockBitExitCmd

Description:

This command exits the PPB Lock Bit Manipulation Mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

n/a

Related Commands: lld_PpbLockBitEntryCmd, lld_PpbLockBitSetCmd, lld_PpbLockBitReadCmd

Example Code:

```
lld_PpbLockBitExitCmd(addr);
```

2.8.9 Sector Protection Commands

2.8.9.1 lld_SectorLockCmd

Description:

This command locks and protects all sectors.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.

Behavior:

All sectors will be locked for writing.

Related Commands: lld_SectorUnlockCmd, lld_SectorLockRangeCmd

Example Code:

```
lld_SectorLockCmd(base_addr;
```

2.8.9.2 lld_SectorUnlockCmd

Description:

This command unlocks and un-protects a sector.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
offset	FLASHDATA	An index into the location where the sector to unlock.

Behavior:

The specified sector will be unlocked for writing.

Related Commands: lld_SectorUnlockCmd, lld_SectorLockRangeCmd

Example Code:

```
lld_SectorUnlockCmd(base_addr, offset);
```

2.8.9.3 lld_SectorLockRangeCmd

Description:

This command locks and protects a range of sectors.

Returns: -1 for error, 0 for success

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device being manipulated.
StartSec	ADDRESS	An index into the location where the start sector to lock.
StopSec	ADDRESS	An index into the location where the stop sector to lock.

Behavior:

The specified range of sectors will be lock for writing.

Related Commands: lld_SectorUnlockCmd, lld_SectorLockRangeCmd

Example Code:

```
lld_SectorLockRangeCmd(base_addr, StartSec, StopSec);
```

2.9 Miscellaneous APIs

2.9.1 Miscellaneous Commands

2.9.1.1 lld_SetConfigRegCmd

Description:

This command is used to set the Configuration Register. Refer to the data sheet for specific bit information.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash to be manipulated.
value	FLASHDATA	The configuration data to be written.

Behavior:

n/a

Related Commands: lld_ReadConfigRegCmd

Example Code:

```
lld_SetConfigRegCmd(addr, value);
```

2.9.1.2 lld_SetConfigRegCmd (Device with Address Space Overlay Mode)

Description:

This command is used to set the Configuration Register. Refer to the data sheet for specific bit information.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash to be manipulated.
offset	ADDRESS	sector offset for ASO (Address Space Overlay).
value	FLASHDATA	The configuration data to be written.

Behavior:

n/a

Related Commands: lld_ReadConfigRegCmd

Example Code:

```
lld_SetConfigRegCmd(base_addr, offset, value);
```

2.9.1.3 lld_SetConfigRegCmd (WS-P Device)

Description:

The command is used to set the Configuration Registers for WS-P devices. Refer to the data sheets for specific bit information.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
value	FLASHDATA	The data to be written to Configuration Register 0.
Value1	FLASHDATA	The data to be written to Configuration Register 1.

Behavior:

n/a

Related Commands: lld_ReadConfigRegCmd

Example Code:

```
lld_SetConfigRegCmd( base_addr, value, value1);
```

2.9.1.4 lld_ReadConfigRegCmd

Description:

This command reads the Configuration Register word.

Returns: Configuration Register Word

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_SetConfigRegCmd

Example Code:

```
data = lld_ReadConfigRegCmd(addr);
printf("%8.8X\n", data);
```

2.9.1.5 lld_ReadConfigRegCmd (Device with Address Space Overlay Mode)

Description:

This command reads the Configuration Register word.

Returns: Configuration Register Word

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	Sector offset to read.

Behavior:

n/a

Related Commands: lld_SetConfigRegCmd

Example Code:

```
data = lld_ReadConfigRegCmd(base_addr, offset);
  printf("%8.8X\n", data);
```

2.10 Deep Power-Down APIs

2.10.1 Deep Power-Down Commands

2.10.1.1 lld_EnterDeepPowerDownCmd

Description:

This command is used to enter Deep Power Down (DPD) mode.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_ReleaseDeepPowerDownCmd

Example Code:

```
lld_EnterDeepPowerDownCmd(base_addr);
```

2.10.1.2 lld_ReleaseDeepPowerDown

Description:

This command is used to exit Deep Power Down (DPD) mode. Exiting the DPD mode is accomplished with the assertion of CS# during any read or write transaction (CS# LOW for at least four clock cycles). During the t_{DPD} (300 µs) period the device will ignore command sequences (read and write transactions will not be processed) and RDS will not toggle during an attempted read transaction. A “dummy” write transaction is the preferred method for exiting DPD. Driving the RESET# input LOW (for the minimum t_{RP} time) will also cause the device to exit the DPD mode. The device will take t_{DPD} to return to the idle state.

This API provides 2 methods to exit deep power down: a dummy write and a read array. User may also issue a hardware reset operation to exit DPD.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

mode	FLASHDATA	2 methods to exit deep power down:
		mode = 0: issue “dummy” write
		mode = 1: issue read array

Behavior:

n/a

Related Commands: lld_EnterDeepPowerDownCmd()

Example Code:

```
lld_ReleaseDeepPowerDownCmd(base_addr, mode);
```

2.11 Temperature Sensor APIs

2.11.1 Temperature Sensor Commands

2.11.1.1 lld_MeasureTemperatureCmd

Description:

This command is used to measure temperature of device.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_ReadTemperatureRegCmd

Example Code:

```
lld_MeasureTemperatureCmd(base_addr);
```

2.11.1.2 lld_ReadTemperatureRegCmd

Description:

This command is used to read temperature register.

Returns: FLASHDATA (Word Temperature Register Result)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_MeasureTemperatureCmd

Example Code:

```
temp_reg = lld_ReadTemperatureRegCmd(base_addr);
printf("Temperature Register = 0x%04X\n", temp_reg);
```

2.11.1.3 lld_MeasureTemperatureRegOp

Description:

This operation is used to measure temperature and read temperature register.

Returns: DEVSTATUS (DEV_BUSY = Measure temperature failed; DEV_NO_BUSY = Measure temperature completed)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
temperature_reg	FLASHDATA *	The pointer of temperature register value.

Behavior:

When operation return status is “DEV_NO_BUSY”, the operation read temperature register value from device and store into output parameter “temperature_reg”; when operation return status is “DEV_BUSY”, the operation doesn't read temperature register.

Related Commands: lld_MeasureTemperatureCmd, lld_ReadTemperatureRegCmd

Example Code:

```
status = lld_MeasureTemperatureRegOp(base_addr, temperature_reg);
if (status != DEV_NO_BUSY)
  printf("Measure Temperature Failed!\n");
else
  printf("Measure Temperature Registers = %X\n", &temperature_reg);
```

2.12 Power-On Reset Timer APIs

2.12.1 Power-On Reset Timer Commands

2.12.1.1 lld_ProgramPORTimerRegCmd

Description:

This command is used to program Power On Reset Timer Register.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
portime	FLASHDATA	The data to be used for programming.

Behavior:

n/a

Related Commands: lld_ReadPORTimerRegCmd

Example Code:

```
lld_ProgramPORTimerRegCmd(base_addr, wr_por_reg);
```

2.12.1.2 lld_ReadPORTimerRegCmd

Description:

This command is used to read Power On Reset Timer Register.

Returns: FLASHDATA (Word Power On Reset Timer Result)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_ProgramPORTimerRegCmd

Example Code:

```
rd_por_reg = lld_ReadPORTimerRegCmd(base_addr);
printf("POR Register = 0x%04X\n", rd_por_reg);
```

2.13 Interrupt APIs

2.13.1 Interrupt Commands

2.13.1.1 lld_LoadInterruptConfigRegCmd

Description:

This command is used to load Interrupt Configuration Register.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
icr	FLASHDATA	The data to be used for programming.

Behavior:

n/a

Related Commands: lld_ReadInterruptConfigRegCmd

Example Code:

```
lld_LoadInterruptConfigRegCmd(base_addr, load_icr);
```

2.13.1.2 lld_ReadInterruptConfigRegCmd

Description:

This command is used to read Interrupt Configuration Register.

Returns: FLASHDATA (Word Interrupt Configuration Register)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_LoadInterruptConfigRegCmd

Example Code:

```
read_icr = lld_ReadInterruptConfigRegCmd(base_addr);
printf("Interrupt Configuration Register = 0x%04X\n", read_icr);
```

2.13.1.3 lld_LoadInterruptStatusRegCmd

Description:

This command is used to load Interrupt Status Register.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
isr	FLASHDATA	The data to be used for programming.

Behavior:

n/a

Related Commands: lld_ReadInterruptStatusRegCmd

Example Code:

```
lld_LoadInterruptStatusRegCmd(base_addr, load_isr);
```

2.13.1.4 lld_ReadInterruptStatusRegCmd

Description:

This command is used to read Interrupt Status Register.

Returns: FLASHDATA (Word Interrupt Status Register)

Parameters:

Behavior:

n/a

Related Commands: lld_LoadInterruptStatusRegCmd

Example Code:

```
read_isr = lld_ReadInterruptStatusRegCmd(base_addr);
printf("Interrupt Configuration Register = 0x%04X\n", read_isr);
```

2.14 Volatile and Non-Volatile Configuration Register APIs

2.14.1 Configuration Register Operation

2.14.1.1 lld_ProgramNonVolatileConfigRegOp

Description:

This function is used to program non-volatile configuration register and poll the status for completion

Returns: DEVSTATUS (DEV_NOT_BUSY = Program Completed; DEV_PROGRAM_ERROR = Program Error; DEV_ERASE_ERROR = Erase Error)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
nvcr	FLASHDATA	The data to be used for programming.

Behavior:

n/a

Related Commands: lld_ProgramNonVolatileConfigRegCmd, lld_EraseNonVolatileConfigRegCmd, lld_ReadNonVolatileConfigRegCmd

Example Code:

```
dev_status = lld_ProgramNonVolatileConfigRegOp (base_addr, pgm_nvcr);
printf("status = %s\n", get_status_str(dev_status));
```

2.14.2 Configuration Register Commands

2.14.2.1 lld_LoadVolatileConfigRegCmd

Description:

This command is used to load Volatile Configuration Register.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
vcr	FLASHDATA	The data to be used for write.

Behavior:

n/a

Related Commands: lld_ReadVolatileConfigRegCmd

Example Code:

```
lld_LoadVolatileConfigRegCmd(base_addr, load_vcr);
```

2.14.2.2 lld_ReadVolatileConfigRegCmd

Description:

This command is used to read Volatile Configuration Register.

Returns: FLASHDATA (Word Volatile Configuration Register)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_LoadInterruptConfigRegCmd

Example Code:

```
read_vcr = lld_ReadVolatileConfigRegCmd(base_addr);
printf("Interrupt Configuration Register = 0x%04X\n", read_vcr);
```

2.14.2.3 lld_ProgramNonVolatileConfigRegCmd

Description:

This command is used to Program Non-Volatile Configuration Register.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
nvcr	FLASHDATA	The data to be used for programming.

Behavior:

When issued, this command will begin the programming process. The flash will no longer be in read array mode during programming. Typically, this command is followed by a status polling routine to determine the state of the flash.

Related Commands: lld_EraseNonVolatileConfigRegCmd, lld_ReadNonVolatileConfigRegCmd

Example Code:

```
lld_ProgramNonVolatileConfigRegCmd(base_addr, pgm_nvcr);
```

2.14.2.4 lld_ReadNonVolatileConfigRegCmd

Description:

This command is used to read Non-Volatile Configuration Register.

Returns: FLASHDATA (Word Non-Volatile Configuration Register)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_ EraseNonVolatileConfigRegCmd, lld_ProgramNonVolatileConfigRegCmd

Example Code:

```
read_nvcr = lld_ReadVolatileConfigRegCmd(base_addr);
printf("Interrupt Configuration Register = 0x%04X\n", read_nvcr);
```

2.14.2.5 lld_EraseNonVolatileConfigRegCmd

Description:

This command is used to Erase Non-Volatile Configuration Register.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_ProgramNonVolatileConfigRegCmd, lld_ReadNonVolatileConfigReg

Example Code:

```
lld_EraseNonVolatileConfigRegCmd (base_addr);
```

2.15 Evaluate Erase Status APIs

2.15.1 Evaluate Erase Status Commands

2.15.1.1 lld_EvaluateEraseStatusCmd

Description:

The Evaluate Erase Status (EES) command verifies that the last erase operation on the addressed sector was completed successfully. The EES command can be used to detect erase operations that failed due to loss of power, reset, or failure during an erase operation. Please read device data sheet for more detail.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
offset	ADDRESS	Sector offset for SO (Address Space Overlay).

Behavior:

n/a

Related Commands: n/a

Example Code:

```
lld_EvaluateEraseStatusCmd (base_addr, offset);
```

2.16 CRC APIs

2.16.1 CRC Commands

2.16.1.1 lld_CRCEnterCmd

Description:

This command is used to enter CRC ASO.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash to be manipulated.

Behavior:

n/a

Related Commands: n/a

Example Code:

```
lld_CRCEnterCmd (base_addr);
```

2.16.1.2 lld_LoadCRCStartAddrCmd

Description:

This command is used to load CRC start address.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
bl	ADDRESS	Beginning location of checkvalue calculation.

Behavior:

n/a

Related Commands: lld_CRCEnterCmd, lld_CRCExitCmd

Example Code:

```
lld_LoadCRCStartAddrCmd (base_addr, bl);
```

2.16.1.3 lld_LoadCRCEndAddrCmd

Description:

This command is used to load CRC end address.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.
el	ADDRESS	Ending location of checkvalue calculation.

Behavior:

n/a

Related Commands: lld_CRCEnterCmd, lld_CRCExitCmd

Example Code:

```
lld_LoadCRCEndAddrCmd (base_addr, el);
```

2.16.1.4 lld_CRCSuspendCmd

Description:

This command is used to suspend CRC calculation.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_CRCEnterCmd, lld_CRCExitCmd

Example Code:

```
lld_CRCSuspendCmd (base_addr);
```

2.16.1.5 lld_CRCResumeCmd

Description:

This command is used to resume CRC calculation.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_CRCEnterCmd, lld_CRCExitCmd, lld_CRCSuspendCmd

Example Code:

```
lld_CRCResumeCmd (base_addr);
```

2.16.1.6 lld_ReadCheckvalueLowResultRegCmd

Description:

This command is used to read CheckValue Low Result.

Returns: FLASHDATA (Word CheckValue Low Result)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_CRCEnterCmd, lld_CRCExitCmd, lld_LoadCRCStartAddrCmd, lld_LoadCRCEndAddrCmd

Example Code:

```
low = lld_ReadCheckvalueLowResultRegCmd (base_addr);
```

2.16.1.7 lld_ReadCheckvalueHighResultRegCmd

Description:

This command is used to read CheckValue High Result.

Returns: FLASHDATA (Word CheckValue High Result)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_CRCEnterCmd, lld_CRCExitCmd, lld_LoadCRCStartAddrCmd, lld_LoadCRCEndAddrCmd

Example Code:

```
high = lld_ReadCheckvalueHighResultRegCmd (base_addr);
```

2.16.1.8 lld_CRCExitCmd

Description:

This command is used to exit CRC ASO.

Returns: n/a

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior:

n/a

Related Commands: lld_CRCEnterCmd

Example Code:

```
lld_CRCExitCmd (base_addr);
```

2.17 Continuity Check APIs

2.17.1 Continuity Check Operation

2.17.1.1 lld_ContinuityCheckOp

Description:

The Continuity Check provides a basic test of connectivity from package connectors to each die pad and to each individual die in a DDP.

Returns: DEVSTATUS (Continuity Check pattern detected, Continuity Check Error)

Parameters:

Name	Type	Description
base_addr	FLASHDATA *	The base address of the flash device to be manipulated.

Behavior: n/a

Example Code:

```
continuity_check_status = lld_ContinuityCheckOp (base_addr);
printf("continuity check status = %s\n", get_status_str(continuity_check_status));
```

Revision History

Cypress® Low Level Driver User Guide Revision History

Document Title: Cypress® Low Level Driver User Guide Document Number: 002-00329			
Revision	**Issue Date**	**Origin of Change**	**Description of Change**
**	01/07/2010	–	Initial revision
*A	02/24/2011	–	Files: Added sentence about trace.c/trace.h Making the LLD Work in Your Environment: Changed S29GL512R and S29GLxxxR to S29GL512S and S29GLxxxS
*B	09/28/2011	–	Global: Rearranged the order of the sections to improve usability of document Common API: Added paragraph under section heading Basic Operations: Modified note LLD Cleanup: Moved lld_GetVersion to of “Basic Operations” section Changed the name of lld_EraseSuspendnOp to lld_EraseSuspendnOp(CMD1) Changed the name of lld_ProgramSuspendOp to lld_ProgramSuspendOp(CMD1) and edited the table description Edited the table description of lld_BlandkCheckOP Changed the name of lld_StatusRegReadCmd to lld_StatusRegReadCmd(CMD2) Edited the example code of lld_StatusRegReadCmd Changed the name of lld_StatusRegClearCmd to lld_StatusRegClearCmd(CMD2) Removed: lld_BitfieldProgrammingOp, lld_BitFieldCmd APIs, lld_SecSiSectorExitCmd Added: lld_StatusGetReg, lld_StatusClear(CMD1) lld_StatusClear(CMD2), lld_EraseSuspendOp(CMD2), lld_ProgramSuspendOp(CMD2), lld_GetVersion API, lld_StatusRegReadCmd(CMD1), lld_StatusRegClearCmd(CMD1), lld_AutoselectEntryCmd(Device with Address Space Overlay Mode), lld_LockReg2EntryCmd, lld_SetConfigRegCmd(WS-P Device)

Cypress® Low Level Driver User Guide Revision History (Continued)

Document Title: Cypress® Low Level Driver User Guide Document Number: 002-00329			
Revision	**Issue Date**	**Origin of Change**	**Description of Change**
*C	12/16/2014	–	General: Updated from Release 11.3.2 to Release 14.4.1 Deep Power-Down APIs: Added section Temperature Sensor APIs: Added section Power-On Reset Timer APIs: Added section Interrupt APIs: Added section Volatile and Non-Volatile Configuration Register APIs: Added section Evaluate Erase Status APIs: Added section ECC APIs: Added section CRC APIs: Added section SA Protect Status APIs: Added two sections: PPB SA Protect Status Commands and DYB SA Protect Status Commands
*D	06/30/2015	–	General: Updated from Release 14.4.1 to Release 15.2.1. Changed flash bank to flash device throughout Introduction: Updated section API Specification: Nomenclature, Arguments, and Typedefs: updated Interleaved Typdefs: updated section Common APIs Example Code: updated throughout Sector Protection APIs: lld_PpbLockBitReadOp, lld_PpbLockBitSetOp: moved sections lld_LockReg2EntryCmd: deleted section ECC APIs: Removed section
*E	10/06/2015	BACD	Updated to the Cypress template