BMU Design Spec

Version 1.0

2023/3/1

云海芯科微电子科技有限公司

CONTENTS

FIGURE

TABLE

REVISION HISTORY

Revision	Date	Modified By	Description
1.00	2023/03/1	James	Initial Release

Revision m.xy, where:

- m the first digit are incremented for major changes of substance, e.g., function changes.
- xy the second two digits are incremented when minor changes have been incorporated into the specification, i.e., enhancements, corrections, updates, etc.

1. INTRODUCTION

1.1. Document Scope

BMU is the Buffer Management Unit of high performance PCIe Gen5 SSD Controller. This Design Spec. would describe the design principles of the BMU, interface signal and registers in detail.

1.2. Reference

- NVMe CTRL Design Spec
- DRAM CTRL Design Spec
- ECC CTRL Design Spec
- NAND CTRL Design Spec
- Firmware AHB-like interface

1.3. Glossary

- PWT – Paradigm Works Training
- DUT – Device Under Test
- CPU – Central Processing Unit
- Packet – A custom datagram used to send and receive data from and to a device
- RO – Read Only
- RW – Read/Write
- W1C – Read/Write 1 to Clear
- RC – Read to Clear

1.4. Design Information

Table 1. Product List

Part Number	Application	Package Type	Package Size (mm)	Flash Die

1.5. Design Feature

- Adopting a special TDM architecture, in the case of a system frequency of 500MHz, it can provide four IPs of NVMe/DRAM/LDPC/NAND, and at the same time there is a 16GB/s bandwidth to access the SRAM data buffer
- The hardware manages the SRAM data buffer and automatically allocates and reclaims the 4KB data buffer in 4KB units
- The hardware manages DRAM data buffer, and the management method is exactly the same as that of SRAM
- The hardware records each LBA of 4KB data and automatically handles the same LBA
- Support Host sector size 512Bytes/4KBytes, H/W handle non 4KB align pattern
- Support T10 DIF/DIX Host Meta data (8/16/64/128Bytes per 4KB) and Host CRC check
- The Firmware Meta data size can be flexibly set (4B/8B/12B/16B/20B/24B/28B/32B)
- For different NAND specifications, LDPC parity size can be flexibly set
- Flexible support for different NAND channel settings (2channel/4channel/8channel) and when the number of channels is reduced, the frequency can be reduced proportionally
- All SRAMs support SECDED (Single Error Correct and Double Error Detect)

2. DESIGN DESCRIPTION

2.1. BMU Design Overview

BMU 是 Buffer Management Unit 的縮寫，是 SSD Controller 的核心 IP，所有其他 IP 的 Data 都經由 BMU 傳遞. BMU 主要負責管理 Data Buffer，包含 DRAM 10MB 和 SRAM 共 2MB，以 MAPU(Mapping Unit, 4.xKB)為單位，總共管理 3072 MAPU，這些 Buffer 可彈性分配給 CPU, RX/TX buffer, RAID parity. BMU 採用分時多工的架構，可保證每一個 IP 都能分配到 16GB/s 的頻寬，且不會互相干擾. BMU 自動記錄 buffer 存放 Data 的 L4K 及相關狀態，可快速搜尋，整合. BMU 同時內建 RAID5/6 的功能，可自動計算 RAID parity.

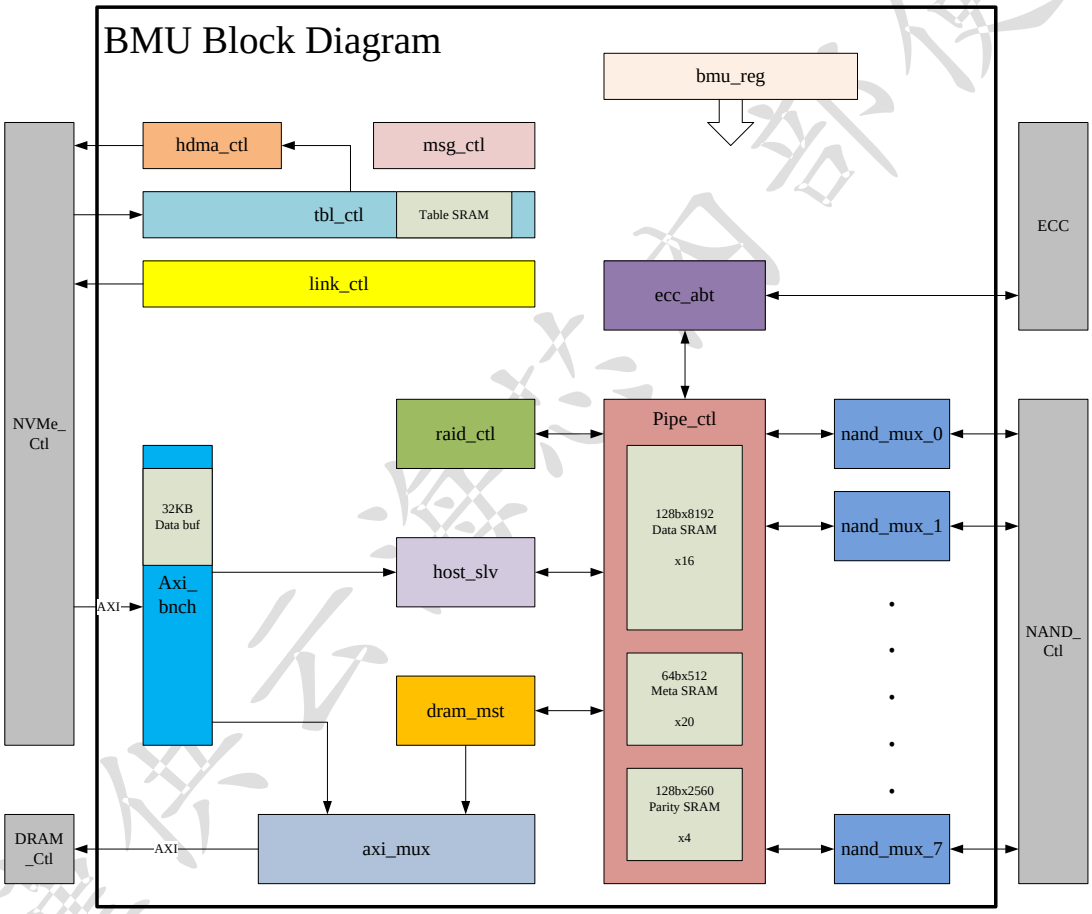

Figure 1. BMU Architecture/Function Block

如上圖一所示，BMU 主要是由 13 個 module 所組成. 每個 module 主要負責的 function 大致描述如下:

3. INTERFACES AND PROTOCOLS

3.1. AHB-like Interface

Pin Name	Dir	Description

3.2. Host Ctrl Interface

Pin Name	Dir	Description

3.3. DRAM Ctrl Interface

Pin Name	Dir	Description

4. PROGRAMMING MODEL

4.1. Control and Status Registers Listing Overview

4.1.1. Register Map

****特別注意，凡是 Type 是 RO，則代表 Read Only，因為並不存在實際的 register，所以並無 default value，讀到的值與當時的 status 有關. Type 是 RWC 則代表此 register 除了可以 Read/Write 外, H/W 會自動 Clear，也就是清成 0. W1C 則是代表寫 1, H/W 會清成 0**

Register Name	CPU_ACC_Reg
Register Address	*0x00~0x3F*

Osft	Type	Bit	Def	Description
0x0	RW	5	0	[0]: reg_cafp_req: cpu send request to get free buffer pointer for raid parity buffer or other reason, when reg_cafp_req = 0, imply request finish
			0	[2:1]: reg_cafp_typ: buffer type, 0: dram buffer pointer, 1: sram buffer pointer, 2: LDPC parity buffer pointer
			0	[4:3]: reg_cafp_mst: the master of get buffer, 0: host, 1: nand, 2: decw/dram, 3: cpua
0x1	RO	1	0	cafp_bok: get free buffer ok, when reg_cafp_req = 0, if(cafp_bok = 0), imply no free buffer
0x2	RW	12	0	reg_cafp_ptr: free buffer pointer
0x4	RW	1	0	reg_cawf_req: if cpu want to send data to host, cpu cand get free buffer and write data to this buffer. and cpu send “write buffer finish” info to BMU, BMU would add this cmd to host tx queue, when reg_cawf_req = 0, imply request finish
0x5	RW	2	0	[0]: reg_cawf_rls: when transfer finish, release buffer pointer [1]: reg_cawf_crd: compression data
0x6	RW	12	0	reg_cawf_bfp: buffer pointer
0x8	RW	18	0	reg_cawf_cid: command ID, if CPU receive error data(ECC fail), would get the CID of this data. When cpu correct this data(soft decode or RAID), would write CID to this register
0xc	W1C	1	0	reg_card_req: if H/W reveive error data, would set this register as 1 to tell CPU. When CPU finish this cmd, would set 1 to clear it

0xe	RW	12	0	reg_card_bfp: buffer pointer
0x10	RW	18	0	reg_card_cid0: command ID0, if receive compression data, would receive up to 4 CIDs
0x14	RW	18	0	reg_card_cid1: command ID1
0x18	RW	18	0	reg_card_cid2: command ID2
0x1c	RW	18	0	reg_card_cid3: command ID3
0x20	RW	1	0	reg_carl_req: cpu release buffer to BMU
0x22	RW	12	0	reg_carl_bfp: buffer pointer

Register Name	**BMU_CFG_Reg**
Register Address	*0x40~0x57*

Osft	**Type**	**Bit**	**Def**	**Description**
0x40	RW	2	0	reg_chan_typ: NAND channel type, 0:8channel, 1:4channel, 2:2channel
0x42	RW	4	10	reg_page_str: NAND page start location, reg_page_str = reg_frag_sft + reg_die_sft
0x43	RW	4	12	reg_page_sft : NAND page bit number, if 2304 page, need 12bit to record page number, so set reg_page_sft = 12
0x44	RW	1	0	reg_itbl_req: initial BMU table request, when reg_itbl_req = 0, imply initial finish
0x46	RW	12	4095	reg_itbl_dat: initial table data value, 4095 imply null entry
0x48	RW	5	0 0 0 1 0	[1:0]: reg_host_pri: [1]:host read/write force priority, [0]: write channel is high priority [2]: reg_prit_sel: priority type select, 1: strict priority, 0: round robin [3]: reg_ptbl_typ: program unit table algorithm [4]: reg_raid6_en: enable RAID6 function [5]: reg_raid5_en: enable RAID5 function
0x49	RW	2	0 0	[0]: reg_encr_rst: reset ECC encoder read link list [1]: reg_decr_rst: reset ECC decoder read link list
0x4a	RW	9	240	reg_eabt_thr: the arbiter threshold for ECC decoder_write
0x4c	RW	1	0	reg_ddma_req: cpu enable dram/sram dma
0x4d	RW	1	0	reg_ddma_wen: 1: transfer data from sram to dram, 0: transfer data

				from dram to sram
0x4e	RW	10	0	reg_ddma_len: transfer length, max 8KB length, 8byte unit
0x50	RW	32	0	reg_ddma_adr; dram start pointer, 16byte unit
0x54	RW	18	0	reg_ddma_sad: sram start pointer, 16byte unit

Register Name			**Buffer_THR_Reg**	
Register Address			*0x58~0x9F*	
Osft	**Type**	**Bit**	**Def**	**Description**
0x58	RW	12	0	reg_htdl_thr, free link threshold for host write data to dram buffer, when (dlnk_num < reg_htdl_thr), host can't use dram buffer
0x5a	RW	12	0	reg_htdb_thr, use buffer threshold for host write data to dram buffer, when (htdb_num > reg_htdb_thr), host can't use dram buffer
0x5c	RW	9	0	reg_htsl_thr, free link threshold for host write data to sram buffer
0x5e	RW	9	248	reg_htsb_thr, use buffer threshold for host write data to sram buffer
0x60	RW	8	24	reg_htpl_thr, free link threshold for host write data to ecc parity buffer
0x62	RW	8	216	reg_htpb_thr, use buffer threshold for host write data to ecc parity buffer
0x64	RW	8	8	reg_ndpl_thr, free link threshold for nctl write data to ecc parity buffer
0x66	RW	8	232	reg_ndpb_thr, use buffer threshold for nctl write data to ecc parity buffer
0x68	RW	9	0	reg_dwsl_thr, free link threshold for ecc_dec write data to sram buffer
0x6a	RW	9	128	reg_dwsb_thr, use buffer threshold for ecc_dec write data to sram buffer
0x6c	RW	8	0	reg_dwpl_thr, free link threshold for ecc_dec write data to ecc parity buffer
0x6e	RW	8	128	reg_dwpb_thr, use buffer threshold for ecc_dec write data to ecc parity buffer
0x70	RW	12	0	reg_dmdl_thr, free link threshold for gdma write data to dram buffer
0x72	RW	12	0	reg_dmdb_thr, use buffer threshold for gdma write data to dram

				buffer
0x74	RW	12	0	reg_cadl_thr, free link threshold for cpu write data to dram buffer
0x76	RW	12	128	reg_cadb_thr, use buffer threshold for cpu write data to dram buffer
0x78	RW	9	0	reg_casl_thr, free link threshold for cpu write data to sram buffer
0x7a	RW	9	128	reg_casb_thr, use buffer threshold for cpu write data to sram buffer
0x7c	RW	8	0	reg_capl_thr, free link threshold for cpu write data to ecc parity buffer
0x7e	RW	8	128	reg_capb_thr, use buffer threshold for cpu write data to ecc parity buffer
0x80	RO	12	2559	dlnk_num: dram free link number
0x82	RO	12	0	htdb_num: host use dram buffer number
0x84	RO	12	0	dmdb_num: gdma use dram buffer number
0x86	RO	12	0	cadb_num: cpu_agent use dram buffer number
0x88	RO	9	256	slnk_num: sram free link number
0x8a	RO	9	0	htsb_num: host use sram buffer number
0x8c	RO	9	0	dwsb_num: ecc_dec use sram buffer number
0x8e	RO	9	0	casb_num: cpu_agent use sram buffer number
0x90	RO	8	240	plnk_num: ecc parity free link number
0x92	RO	8	0	ndpb_num: nand_read use ecc parity buffer number
0x94	RO	8	0	htpb_num: host use ecc parity buffer number
0x96	RO	8	0	dwpb_num: ecc_dec use ecc parity buffer number
0x98	RO	8	0	capb_num: cpu_agent use ecc parity buffer number
0x9a	RO	12	0	htcq_num: host queue cmd number
0x9c	RO	12	0	dmcq_num: dram queue cmd number
0x9e	RO	12	0	cacq_num: cpu_agent queue cmd number