

NVMe-IP Linux Demo Instruction

Rev1.0 19-Feb-18

This document describes the instruction to run NVMe-IP demo with Angstrom Linux 2014.12 OS on Arria10 SoC Development board. The demo is designed to get Identify data, write data, and read data with NVMe SSD. User controls test operation through Serial Console. To run the demo, FPGA configuration file as image file is prepared in microSD card. After that, install microSD card to FPGA board for configuration after power on board.

1 Environment Requirement

To demo NVMe-IP Linux demo on IntelFPGA board, please prepare the following hardware/software.

- 1) IntelFPGA board: Arria10 SoC Development board
- 2) PC with Serial console software, i.e. HyperTerminal
- 3) Power adapter of IntelFPGA board
- 4) NVMe SSD
- 5) 4 GB or bigger microSD card with demo image file, downloaded from http://www.dgway.com/NVMe-IP A E.html
- 6) miniUSB cable for Serial communication





Figure 1-1 NVMe-IP Linux Demo on Arria10 SoC Development Board



2 Demo setup

2.1 Prepare demo image file to microSD Card

The sequence to prepare image file is shown as follows.

- 1) Extract "sd_card_image_a10.img.tar.gz" to get output file "sd_card_image_a10.img".
- 2) Connect microSD card to PC by using card reader.
- 3) Open "Win32DiskImage" application.
- 4) Copy "sd_card_image_a10.img" to microSD by using Win32DiskImage, as shown in Figure 2-1.
 - a. Select drive to microSD card drive.
 - b. Browse to directory path of "sd_card_image_a10.img" file.
 - c. Click "Write" to start dump data.
 - d. "Confirm overwrite" window is displayed. Click "Yes" to confirm data dump.
 - e. Wait until progress status updated from 0% to 100%. "Complete" window with "Write Successful" is displayed. Click "OK" to complete this step.



Figure 2-1 Write demo image file to microSD Card



2.2 PCIe setup and Board setup

- 1) Power off system.
- 2) Insert microSD card to the socket on daughter cards.
- 3) Connect NVMe SSD to PCIe connector on FPGA board, as shown in Figure 2-2.



Figure 2-2 NVMe SSD connection on FPGA board

4) Connect mini USB cable from FPGA board to PC for Serial console, as shown in Figure 2-3.





5) Set SW3[1] /[2]/ [6]/ [8] = OFF position to enable JTAG of HPS, FPGA, and MAX, as shown in Figure 2-4.



Figure 2-4 JTAG Enable for Arria10 SoC board

- 6) Connect FPGA power adapter to FPGA board, as shown in Figure 2-5.
- 7) Power on FPGA development board.



Figure 2-5 Power on FPGA board

- 8) Open Serial console software such as HyperTerminal. Software setting is BaudRate=115,200, Data=8 bit, Non-Parity, and Stop=1.
- 9) On PC Serial console, please wait Linux boot-up until login required, as shown in Figure 2-6.





10)Check LED status on FPGA board. The description of LED is shown as follows.

	Table 1 LED I	Definition
GPIO LED	<u>ON</u>	OFF
0	Normal operation	System is in reset condition
1	System is busy	Idle status
2	Error detect	Normal operation
3	Data verification fail	Normal operation

.. . .



11)After programming completely, LED[0] and LED[1] are ON which mean that PCle initialization is processing. Then, LED[1] is OFF to show that PCle completes initialization process and system is ready to receive command from user.



Figure 2-8 LED status after program configuration file and PCIe initialization complete



3 Test Menu

Before running the test application, user must login to Linux system by login name as "root", as shown in Figure 3-1.



Figure 3-1 Login to Linux system

Next, insert module and running test application by typing following command. >> insmod dg_universal.ko >> ./dg_universal/application/dguTestApp

After that, main menu is displayed on Serial console, as shown in Figure 3-2.

COM4 - PuTTY root@arria10:~# insmod dg_universal/driver/dg_universal.ko root@arria10:~# ./dg_universal/application/dguTestApp	->Insert module ->Running test application
Main Menu [Version: 1.0] [0] : Identify SSD [1] : Write SSD [2] : Read SSD [3] : Dump SSD [9] : Exit Test Enter command:	
Figure 3-2 Insert module and running test applic	cation



3.1 Identify Device

Select '0' to send Identify command to NVMe SSD. When operation is completed, SSD capacity and model name are displayed on the console.

COM4 - PuTTY	×
Enter command: 0	Model name from
[OK] Model Number : Samsung SSD 960 PRO 512GB	identify data
[OK] SSD Capacity : 512 GB(s) SSD capacity output f	rom IP
Main Menu [Version: 1.0] [0] : Identify SSD [1] : Write SSD [2] : Read SSD [3] : Dump SSD [9] : Exit Test Enter command:	-
Figure 3-3 Result from Identify Device m	enu



3.2 Write SSD

Select '1' to send Write command to NVMe SSD. Three inputs are required for this menu.
1) Start LBA: Input start address of SSD in sector unit. The input is decimal unit when input only digit number. User can add "0x" to be prefix when input is hexadecimal unit.
2) Sector Count: Input total transfer size in sector unit. The input is decimal unit when input only digit number. User can add "0x" to be prefix when input is hexadecimal unit when input only digit number. User can add "0x" to be prefix when input is hexadecimal unit.
3) Test pattern: Select test pattern of test data for writing to SSD. Five types can be selected,

i.e. 32-bit increment, 32-bit decrement, all 0, all 1, and 32-bit LFSR counter.

As shown in Figure 3-4, if all inputs are valid, the operation will be started. During writing data, current transfer size is displayed on the console to show that system still run. Finally, test performance, total size, and total time usage are displayed on the console as test result.





125	-	i4-bi	t hea	der	of ea	ich s	ecto	\rightarrow									- 11	-	64-b	it hea	ader	of ea	ich s	ecto	\rightarrow								
	48	-bit I	BA	Addr	ess		0x0	000			2-bi	linci	eme	int di	ata			4	8-bit	LBA	Addr	ess		0x0	000			32-Ь	it LF	SR p	atte	m	
Offset	0	1	2	3	4	5	6	7	8	9	A	В	C	1D	E	F		0	1	2	3	4	5	6	7	8	9	A	В	c	D	E	F
0000000000	00	00	00	00	00	00	00	00	02	00	00	00	03	00	00	00		00	00	00	00	00	00	00	00	01	00	00	00	02	00	00	00
0000000010	04	00	00	00	05	00	00	00	06	00	00	00	07	00	00	00		04	00	00	00	09	00	00	00	12	00	00	00	24	00	00	00
0000000020	08	00	00	00	09	00	00	00	0A	00	00	00	0B	00	00	00		49	00	00	00	92	00	00	00	24	01	00	00	49	02	00	00
000000030	0C	00	00	00	OD	00	00	00	0E	00	00	00	OF	00	00	00		92	04	00	00	24	09	00	00	49	12	00	00	92	24	00	00
0000000040	10	00	00	00	11	00	00	00	12	00	00	00	13	00	00	00		24	49	00	00	49	92	00	00	92	24	01	00	24	49	02	00
0000000050	14	00	00	00	15	00	00	00	16	00	00	00	17	00	00	00		49	92	0.4	00	92	24	09	00	24	49	12	00	49	92	24	00
0000000060	18	00	00	00	19	00	00	00	1A	00	00	00	1B	00	00	00		93	24	49	00	27	49	92	00	4F	92	24	01	9E	24	49	02
0000000070	1C	00	00	00	1D	00	00	00	1E	00	00	00	1F	00	00	00		30	49	92	04	79	92	24	09	FЗ	24	49	12	E7	49	92	24
0000000080	20	00	00	00	21	00	00	00	22	00	00	00	23	00	00	00		CF	93	24	49	9E	27	49	92	ЗD	4F	92	24	7A	9E	24	49
0000000090	24	00	00	00	25	00	00	00	26	00	00	00	27	00	00	00		F5	ЗĊ	49	92	EB	79	92	24	D7	FЗ	24	49	AE	E7	49	92
0400000000	28	00	00	00	29	00	00	00	2A	00	00	00	2B	00	00	00		5D	CF	93	24	BA	9E	27	49	75	ЗD	4F	92	EB	7A	9E	24
0000000B0	2C	00	00	00	2D	00	00	00	2E	00	00	00	2F	00	00	00		D7	F5	3C	49	AE	EB	79	92	5C	D7	FЗ	24	B8	AE	E7	49
00000000000	30	00	00	00	31	00	00	00	32	00	00	00	33	00	00	00		70	5D	CF	93	E0	BA	9E	27	C1	75	3D	4F	83	EB	7A	9E
00000000D0	34	00	00	00	35	00	00	00	36	00	00	00	37	00	00	00		07	D7	F5	3C	0E	AE	EB	79	1D	5C	D7	FЗ	3B	B8	AE	E7
0000000E0	38	00	00	00	39	00	00	00	ЗA	00	00	00	3B	00	00	00		77	70	5D	CF	EE	EO	BA	9E	DC	C1	75	ЗD	B8	83	EB	7A
00000000F0	3C	00	00	00	3D	00	00	00	3E	00	00	00	ЗF	00	00	00		70	07	D7	F5	EO	OE	AE	EB	C1	1D	5C	D7	83	3B	B8	AE
0000000100	40	00	00	00	41	00	00	00	42	00	00	00	43	00	00	00		07	77	70	5D	0E	EE	EO	BA	1C	DC	C1	75	39	B8	83	EB
0000000110	44	00	00	00	45	00	00	00	46	00	00	00	47	00	00	00		73	70	07	D7	E6	EO	OE	AE	CD	C1	1D	5C	9A	83	3B	B8
0000000120	48	00	00	00	49	00	00	00	4A	00	00	00	4B	00	00	00		34	07	77	70	68	0E	EE	EO	D1	1C	DC	C1	A3	39	B8	83
0000000130	4C	00	00	00	4D	00	00	00	4E	00	00	00	4F	00	00	00		47	73	70	07	8E	E6	EO	0E	1D	CD	C1	1D	ЗA	9A	83	3B
0000000140	50	00	00	00	51	00	00	00	52	00	00	00	53	00	00	00		74	34	07	77	E9	68	OE	EE	DЗ	D1	1C	DC	A6	A3	39	B8
0000000150	54	00	00	00	55	00	00	00	56	00	00	00	57	00	00	00		4C	47	73	70	98	8E	E6	EO	31	1D	CD	C1	63	3A	9A	83
0000000160	58	00	00	00	59	00	00	00	5A	00	00	00	5B	00	00	00		C6	74	34	07	8D	E9	68	0E	1B	D3	D1	1C	37	A6	A3	39
0000000170	5C	00	00	00	5D	00	00	00	5E	00	00	00	5F	00	00	00		бE	4C	47	73	DC	98	8E	E6	B8	31	1D	CD	70	63	ЗA	9A
0000000180	60	00	00	00	61	00	00	00	62	00	00	00	63	00	00	00		E1	C6	74	34	C3	8D	E9	68	86	1B	D3	D1	OD	37	A6	A3
0000000190	64	00	00	00	65	00	00	00	66	00	00	00	67	00	00	00		1A	6E	4C	47	34	DC	98	8E	68	B8	31	1D	DO	70	63	ЗĂ
00000001A0	68	00	00	00	69	00	00	00	6A	00	00	00	6B	00	00	00		AO	E1	C6	74	41	C3	8D	E9	83	86	1B	D3	06	OD	37	Å6
00000001B0	6C	00	00	00	6D	00	00	00	6E	00	00	00	6F	00	00	00		0C	1À	6E	4C	18	34	DC	98	30	68	B8	31	60	DO	70	63
00000001C0	70	00	00	00	71	00	00	00	72	00	00	00	73	00	00	00		CO	AO	E1	C6	81	41	C3	8D	03	83	86	1B	07	06	OD	37
00000001D0	74	00	00	00	75	00	00	00	76	00	00	00	77	00	00	00		OF	0C	1A	6E	1F	18	34	DC	ЗF	30	68	B 8	7F	60	DO	70
00000001E0	78	00	00	00	79	00	00	00	7A	00	00	00	7B	00	00	00		FF	CO	AO	E1	FF	81	41	C3	FE	03	83	86	FD	07	06	OD
00000001F0	7C	00	00	00	7D	00	00	00	7E	00	00	00	7F	00	00	00		FA	OF	0C	1A	F4	1F	18	34	E9	ЗF	30	68	D3	7F	60	DO
0000000200	01	00	00	00	00	00	00	00	82	00	00	00	83	00	00	00		1	00	00	00	00	80	00	00	02	00	00	00	04	00	00	00
0000000210	84	00	00	00	85	00	00	00	86	00	00	00	87	00	00	00		09	00	00	00	12	00	00	00	24	00	00	00	49	00	00	00
0000000220	88	00	00	00	89	00	00	00	8A	00	00	00	8B	00	00	00		92	00	00	00	24	01	00	00	49	02	00	00	92	04	00	00
000000230	8C	00	00	00	8D	00	00	00	8E	00	00	00	8F	00	00	00		24	09	00	00	49	12	00	00	92	24	00	00	24	49	00	00
			64	-bit	head	ler														6	-bit	head	ler										

Figure 3-5 Example Test data in sector#0/#1 by increment/LFSR pattern

Test data of each sector has different 64-bit header which consists of 48-bit LBA address and 16-bit all 0 value. 48-bit LBA address is unique value for each sector. After that, the test pattern is filled following user selection such as 32-bit increment pattern (left window of Figure 3-5), 32-bit LFSR pattern (right window of Figure 3-5).



Figure 3-6 – Figure 3-8 shows error message when user input is invalid. "Input is not in range" is displayed on the console. Then, it returns to main menu to receive new command.

P COM4 - PuTTY	3
Enter command: 1 -> Write SSD Enter start LBA [0x0000000 - 0x3B9AC9FF] : 0x3b9aca00 [ER] Input is not in range Main Menu [Version: 1.0] [0] : Identify SSD [1] : Write SSD [2] : Read SSD [3] : Dump SSD [9] : Exit Test	
Enter command:	+

Figure 3-6 Invalid Start LBA input



Figure 3-7 Invalid Sector count input



Figure 3-8 Invalid Test pattern input



3.3 Read SSD

Select '2' to send Read command to NVMe SSD. Three inputs are required for this menu. 1) Start LBA: Input start address of SSD in sector unit. The input is decimal unit when input only digit number. User can add "0x" to be prefix when input is hexadecimal unit.

2) Sector Count: Input total transfer size in sector unit. The input is decimal unit when input only digit number. User can add "0x" to be prefix when input is hexadecimal unit.

3) Test pattern: Select test pattern to verify data from SSD. Test pattern must be matched with the test pattern which is used during write test. Five types can be selected, i.e. 32-bit increment, 32-bit decrement, all 0, all 1, and 32-bit LFSR counter.

Similar to write test if all inputs are valid, test system will read data from SSD. Test performance, total size, and total time usage are displayed after end of transfer. "Invalid input" will be displayed if some input values are out-of-range.





Figure 3-10 shows the error message when data verification is failed. "verification fail" is displayed with Read data and Expected data. User must wait until the test is completed.

- 0 X COM4 - PuTTY . Enter command: 2 -> Read SSD Enter start LBA [0x00000000 - 0x3B9AC9FF] : 0 [OK] Start LBA : 0x00000000 Enter sector count [0x0000001 - 0x3B9ACA00] : 0x4000000 [OK] Sector Count : 0x0400000 Enter pattern [0:Inc32, 1:Dec32, 2:ALL_0, 3:ALL_1, 4:LFSR] : 0 [OK] Selected Pattern : Inc32 Read data to SSD [ER] <verification fail> Read Data : 0x00000000_0000000_0000001_00000002 Expected Data: 0x00000000_00000000_0000002_0000003 14846140/33554432 kB(s> 3.23 GB/s Verify fail and Current transfer size are displayed - -X COM4 - PuTTY . Enter command: 2 -> Read SSD Enter start LBA [0x00000000 - 0x3B9AC9FF] : 0 [OK] Start LBA : 0x00000000 Enter sector count [0x0000001 - 0x3B9ACA00] : 0x4000000 [OK] Sector Count : 0x04000000 Enter pattern [0:Inc32, 1:Dec32, 2:ALL_0, 3:ALL_1, 4:LFSR] : 0 [OK] Selected Pattern : Inc32 Read data to SSD [ER] <verification fail> Read Data : 0x0000000_0000000_00000001_00000002 Read Data Total Transfer: 34 GB(s), Time: 10.86 s, Transfer Speed: 3.16 GB/s - Main Menu [Version: 1.0] ---Verify fail and Output Performance are displayed [0] : Identify SSD [1] : Write SSD [2] : Read SSD [3] : Dump SSD [9] : Exit Test Enter command:

Figure 3-10 Data verification is failed

3.4 Dump SSD

Select '3' to send Dump command to NVMe SSD. Only one input is required for this menu.

Start LBA: Input start address of SSD in sector unit. The input is decimal unit when input only digit number. User can add "0x" to be prefix when input is hexadecimal unit.

Similar to write and read test if the input is valid, test system will return data 1 sector (512 byte) from SSD.

P COM4 - PuTTY		
Entex command: 3		*
-> Dump SSD		
Enter start LBA [0x00000000 - 0x]	3B9AC9FF1 : Ø Input from user	
[OK] Start LBA : 0x00000000 64-bit	t header 512 byte data @	LBA=0
Dump data from SSD		0000
	00000003 00000004 00000005 00000006 000	00007
		00017
0060: 00000018 00000019 0000001A	0000001B 0000001C 0000001D 0000001E 000	0001F
0080: 00000020 00000021 00000022	00000023 00000024 00000025 00000026 000	00027
00A0: 00000028 00000029 0000002A	0000002B 0000002C 0000002D 0000002E 000	0002F
0000: 00000030 00000031 00000032	00000033 00000034 00000035 00000036 000	00037
0100- 00000038 00000039 0000003H	00000038 0000003C 0000003D 0000003F 000	0003F
0100. 0000040 0000041 0000042	00000043 00000044 00000045 00000046 000 00000048 0000004C 0000004D 0000004F 000	00047 0004F
0140: 00000050 00000051 00000052	00000053 00000054 00000055 00000056 000	00057
0160: 00000058 00000059 0000005A	0000005B 0000005C 0000005D 0000005E 000	0005F
0180: 00000060 00000061 00000062	00000063 00000064 00000065 00000066 000	00067
01A0: 00000068 00000069 0000006A	0000006B 0000006C 0000006D 0000006E 000	0006F
0100: 00000070 00000071 00000072		00077
DIED: 00000078 00000079 0000007H		11000
build area rillestons' rulest, ot.		

Figure 3-11 Dump menu example

After displaying one data sector, another input is required. Input 'p' to display the previous sector of Start LBA. Input 'n' to display the next sector of Start LBA. Input 'e' to go back to main menu.



As shown in Figure 3-12, when input 'n' to show the data of the next sector, Start LBA is increased by 1 (from 0 to 1).

Putty				
[OK] Start LBA Dump data from	: 0×00000000 64-b SSD	it header	512 byte dat	ta @ LBA=0 ^
	0000000 0000002	00000003 00000004 00000000 00000004	1 00000005 0000000 00000000 00000000	6 00000007 E 00000007
0040: 00000000	000000000000000000000000000000000000000		4 000000015 00000001	6 00000017
0060: 00000018	00000019 0000001A	0000001B 00000010	C 0000001D 0000001	E 0000001F
0080: 00000020	00000021 00000022	00000023 00000024	£ 00000025 0000002	6 00000027
0040: 00000028	00000027 0000002H	00000028 00000020	, 00000021 0000002 1 000000021 0000002	E 0000002F
00E0: 00000038	00000039 0000003A	0000003B 0000003C	C 0000003D 0000003	E 0000003F
0100: 00000040	00000041 00000042	00000043 00000044	1 00000045 0000004	6 00000047
0120: 00000048	00000049 0000004A	0000004B 00000040	C 0000004D 0000004	E 0000004F
0140: 00000050	00000051 00000052	00000053 00000054	1 00000055 0000005 000000055 0000005	5 00000057 F 0000005F
0180: 00000060	00000061 00000062	00000063 00000064	4 00000065 0000006	6 00000067
01A0: 00000068	00000069 000006A	0000006B 000006C	C 0000006D 000006	E 0000006F
01C0: 00000070	00000071 00000072	00000073 00000074	4 00000075 0000007	6 00000077
<u>Dump data [P]</u>	UUUUUYY UUUUUYH		<u>; 00000070 000007</u>	E 0000007F
[OK] Start LBA	: 0×00000001 64-b	it header	512 byte d	ata @ LBA=1
Dump data from	22D	00000023 00000004		6 0000002
0020: 00000088	00000089 00000084	00000088 0000080	C 0000008D 0000008	E 0000008F
0040: 00000090	00000091 00000092	00000093 00000094	1 00000095 0000009	6 00000097
0060: 00000098	00000099 0000009A	0000009B 0000090	: 0000009D 0000009	E 0000009F
0020: 000000000	00000001 000000002		1 000000A5 000000A	6 UUUUUUA?
0000 000000000000000000000000000000000	000000H7 000000HH 000000B1 000000B2	ANNANAB3 ANNANAB4	4 000000000000000000000000000000000000	6 000000HF
00E0: 000000B8	000000B9 000000BA	000000BB 000000BC	C 000000BD 000000B	E 000000BF
0100: 00000000	000000C1 000000C2	000000C3 000000C4	£ 000000C5 000000C	6 000000C7
0120: 000000C8	000000C9 000000CA	000000CB 000000CC	C 000000CD 000000C	E 000000CF
0140: 000000000	00000001 00000002		, 000000002 000000 00000002	E 00000007E
0180: 000000E0	000000E1 000000E2	000000E3 000000E4	4 000000E5 000000E	6 000000E7 -
01A0: 000000E8	000000E9 000000EA	000000EB 000000EC	: 000000ED 000000E	E 000000EF _
01CO: 000000F0	000000F1 000000F2	000000F3 000000F4	1 000000F5 000000F	6 000000F7
Dumn data [P]»	eujous, [N]ext, or	[Elvit :	<u>, anananth ananant</u>	E UUUUUUIFF
panp data frif	evidus, injekt, ur			

Figure 3-12 Example of Dump menu when input 'n'



As shown in Figure 3-13, when input 'p' to show the data of the previous sector, Start LBA is decreased by 1 (from 1 to 0).

Putty COM4 - Putty								x
[OK] Start LBA Dump data from	: 0×0000000	01 64-bi	t header		51	2 byte data	@ LBA=1	~
0000: 00000001	00000000 00	0000082	00000083	00000084	00000085	00000086	00000087	
0040: 00000090	00000089 0	000008H	00000088	00000080	00000080	00000086	0000008F 00000097	
0060: 00000098	00000099 00	000009A	0000009B	0000009C	0000009D	0000009E	0000009F	
0080: 000000A0	000000A1 00	00000A2	000000A3	000000A4	000000A5	000000A6	000000A7	
	000000H9 00	ооооонн Иииив2	OOOOOOOBB OOOOOOBB	ANNANAB4	OOOOOOODD OOOOOOOB5	000000HE	000000HF	
00E0: 000000B8	000000B9 00	00000BA	000000BB	000000BC	000000BD	000000BE	000000BF	
0100: 00000000	000000C1 00	00000C2	000000C3	000000C4	000000005	000000C6	000000C7	
0120: 00000000	000000003 00	00000CH	000000000	000000000	NUNNNNN	NUNNNN	000000CF 000000D7	
0160: 000000D8	000000D9 00	00000DA	000000DB	000000DC	000000DD	000000DE	000000DF	
0180: 000000E0	000000E1 00	00000E2	000000E3	000000E4	000000E5	000000E6	000000E7	
0180: 000000E8	00000019 00	UUUUULA GGGGGGE2	NNNNNNER NNNNNNER	000000EC	NNNNNED	000000EE	000000EF	
01E0: 000000F8	000000F9 0	00000FA	000000FB	000000FC	000000FD	000000FE	000000FF	
Dump data [P]r	euious [N]¢	evt om	[Elvit ·	22				
		<u>ex</u> , or	LEIXIC .	Input fr	om user			
[OK] Start LBA	: 0×000000	00 64-bit	t header	P Input fr	om user t	512 byte dat	a @ LBA=0	
[OK] Start LBA Dump data from 0000: 00000000	55D 00000000	00 64-bit	t header 00000003	00000004	om user 5 00000005	512 byte dat 00000006	a @ LBA=0 00000007	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000008	: 0×0000000 SSD 00000000 00 00000000 00	00000002 00000002	t header 00000003 0000000B	000000004 000000004	om user 600000005 60000000	512 byte dat 00000006 0000000E	a @ LBA=0 00000007 0000000F	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000008 0040: 00000010	(0.30)	00000002 00000002 0000000A 00000012	t header 00000003 00000008 00000013	00000004 00000004 0000000C 00000014	00000005 00000005 00000000 00000015	512 byte dat 00000006 0000000E 00000016	a @ LBA=0 00000007 0000000F 00000017	
[OK] Start LBA Dump data from 0000: 0000000 0020: 0000008 0040: 00000010 0060: 00000018 0080: 00000028	(0.30) = 0.000000000000000000000000000000000	00000002 00000000 00000000 00000012 00000010 00000010	theader 00000003 00000008 00000013 00000018 00000018	00000004 00000000 00000000 00000014 00000012 00000012	00000005 00000000 00000000 00000015 00000015 00000015	512 byte dat 00000006 0000000E 00000016 0000001E 0000001E	a @ LBA=0 00000007 00000000 000000017 0000001F 0000001F	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000008 0040: 00000010 0060: 00000018 0080: 00000020 00A0: 00000028	$\begin{array}{c} & 10000 \\ & 10000 \\ & 10000 \\ & 10000 \\ & 00000000 \\ & 000000000 \\ & 00000000$	60 64-bit 0000002 0000000 0000012 0000012 000001A 0000022 000002A	theader 00000003 0000000B 00000013 0000001B 0000001B 00000023 0000002B	00000004 00000000 000000014 0000001C 00000024 0000002C	00000005 00000000 000000015 00000015 00000010 00000025 00000020	512 byte dat 00000006 0000000E 00000016 0000001E 0000001E 00000026 0000002E	a @ LBA=0 00000007 0000000F 00000017 0000001F 0000001F 00000027 0000002F	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000008 0040: 00000010 0060: 00000018 0080: 00000020 00A0: 00000028 00C0: 00000030	$\begin{array}{c} & 10000 \\ & 00000000 \\ & 000000000 \\ & 00000000$	60 64-bit 0000002 0000000 0000012 0000012 0000010 0000022 0000022 0000022	theader 00000003 0000000B 00000013 0000001B 00000023 0000002B 00000023	00000004 0000000C 00000014 0000001C 00000024 0000002C 00000034	00000005 00000000 00000000 00000015 00000010 00000025 00000020 00000025	512 byte dat 00000006 0000000E 00000016 0000001E 00000026 0000002E 0000002E	a @ LBA=0 00000007 0000000F 00000017 00000017 00000027 0000002F 00000037	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000008 0040: 00000010 0060: 00000018 0080: 00000028 00A0: 00000028 00C0: 00000038 00C0: 00000038	$\begin{array}{c} & 10000 \\ & 10000 \\ & 00000000 \\ & 000000000 \\ & 00000000$	00000002 00000002 00000000 00000012 00000010 00000022 00000020 00000020 00000032 00000032	theader 00000003 0000000B 00000013 0000001B 00000023 0000002B 0000002B 00000033 00000033 0000003B	00000004 0000000C 00000014 0000001C 00000024 0000002C 00000034 00000034	00000005 00000000 00000015 00000015 00000025 00000025 00000025 00000035 00000035	512 byte dat 00000006 0000000E 00000016 0000001E 00000026 0000002E 00000036 0000003E	a @ LBA=0 00000007 0000000F 00000017 0000001F 00000027 0000002F 00000037 00000037	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000000 0040: 00000010 0060: 00000018 0080: 00000020 00A0: 00000028 00C0: 00000028 00E0: 00000030 00E0: 00000038 0100: 00000048	$\begin{array}{c} \begin{array}{c} & & & & & & \\ & & & & & \\ & & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & $	00000002 00000002 00000012 00000012 0000001A 00000022 00000022 00000032 0000003A 0000003A 0000003A 00000042	LEIXIC 00000003 00000003 00000013 00000013 00000023 00000028 00000033 00000033 00000033 00000033 00000043 00000043 00000043 00000043 00000043	00000004 0000000C 00000014 0000001C 00000024 0000002C 00000034 00000034 00000034 00000034	00000005 00000000 00000015 00000015 00000025 00000020 00000020 00000035 00000035 00000045 00000045	512 byte dat 00000006 0000000E 00000016 0000001E 00000026 0000002E 0000002E 00000036 0000003E 00000046	a @ LBA=0 00000007 0000000F 00000017 00000001F 000000027 00000002F 000000037 00000037 00000037 00000047 00000047	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000000 0040: 00000010 0060: 00000018 0080: 00000020 00A0: 00000028 00C0: 00000030 00E0: 00000030 00E0: 00000030 01E0: 00000048 0140: 00000050	$\begin{array}{c} \begin{array}{c} & & & & & & \\ & & & & & \\ & & & & $	0000002 0000002 00000012 0000012 0000014 0000022 0000020 0000032 0000032 0000032 0000034 0000042 0000042	theader 00000003 0000000B 00000013 00000018 00000023 00000028 00000028 00000033 00000033 00000043 00000043 00000048 00000053	D0000004 00000000 00000000 00000014 00000012 00000024 00000022 00000032 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000044 000000454	00000005 000000015 00000015 00000015 00000025 00000025 00000025 00000035 00000045 00000045 00000045	512 byte dat 00000006 00000016 0000001E 00000026 00000026 00000026 00000028 0000003E 0000003E 00000046 0000004E 0000004E	a @ LBA=0 00000007 000000017 00000017 00000027 00000027 00000027 00000037 0000003F 00000047 00000047 00000047	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000010 0060: 00000018 0080: 00000020 00A0: 00000028 00C0: 00000028 00E0: 00000038 01E0: 00000038 0120: 00000048 0140: 00000058 0160: 00000058	$\begin{array}{c} & & & & & & & & & & & & & & & & & & &$	0000002 0000000 00000012 0000012 0000022 0000022 0000020 0000023 0000032 0000032 0000032 0000034 0000042 0000042 0000042	theader 00000003 0000000B 00000013 00000023 00000023 00000028 0000003B 0000003B 00000043 00000048 00000053 0000005B	D0000004 00000000 00000000 00000014 00000024 00000024 00000032 00000033C 00000034 00000034 00000034 00000034 00000034 00000034 000000354 00000054 00000054 00000054	00000005 00000015 00000015 00000025 0000025 00000025 0000003D 0000003D 00000045 00000055 00000055	512 byte dat 00000006 00000016 0000001E 00000026 00000026 00000026 0000003E 0000003E 0000003E 00000046 00000046 00000056	a @ LBA=0 00000007 00000017 00000017 00000027 00000027 00000027 00000037 00000037 00000037 0000003F 00000047 000000457 00000055	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000000 0040: 00000010 0060: 00000018 0080: 00000020 00A0: 00000020 00C0: 00000030 00E0: 00000038 0100: 00000048 0140: 00000058 0180: 00000058 0180: 00000060 0160: 00000060	$\begin{array}{c} \vdots \\ 0 \times 0 & 0 & 0 & 0 \\ 0 \times 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 &$	00000002 00000000 00000012 00000012 00000022 00000022 00000032 00000032 00000032 00000032 00000032 00000040 00000052 00000052 00000052 00000052	theader 00000003 00000013 00000013 00000018 00000023 00000023 00000033 00000033 00000033 00000043 00000048 00000048 00000058 0000058 00000058 00000058 00000058 0000058 00000058 00000058 00000058 0000058 0000058 0000058 000058 000058 000058 000058 000058 000058 000058 000058 000058 000058 00058 00058 00058 00058 00058 00058 00058 00058 00058 00058 00058 00058 00058 0	D0000004 000000014 00000012 00000012 00000024 00000022 00000023 00000032 00000034 00000032 00000034 00000032 00000034 00000032 00000032 00000034 00000054 00000052 00000052 00000054 00000054 00000054 00000054 00000054 00000054 00000054 00000054 00000054	00000005 000000015 00000015 00000015 00000025 00000025 00000035 00000035 00000045 00000045 00000055 00000055 00000055 00000050	512 byte dat 00000006 00000016 00000016 0000001E 00000026 00000026 0000003E 0000003E 0000003E 0000004E 0000004E 0000005E 0000005E 0000005E	a @ LBA=0 00000007 00000017 00000017 00000017 00000027 00000027 00000037 00000037 00000037 00000037 00000047 00000047 0000004F 00000057 00000057 00000057	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000000 0040: 00000018 0080: 00000020 0080: 00000020 0080: 00000020 00E0: 00000038 01E0: 00000040 0120: 00000048 0140: 00000058 0180: 00000058 0180: 00000058 0180: 0000068 0160: 0000068 0160: 0000068	$\begin{array}{c} \vdots \\ 0 \times 0 & 0 & 0 & 0 \\ 0 \times 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 &$	00000002 00000000 00000000 00000012 00000022 00000022 00000022 00000032 00000032 00000032 00000032 00000040 00000052 00000050 0000050 0000062 0000060	theader 00000003 00000013 00000018 00000028 00000023 00000033 00000033 00000048 00000048 00000048 00000053 00000058 00000058 00000068 00000068 00000063	D Input fr 00000004 0000000C 00000014 00000014 00000024 00000024 00000034 00000034 00000044 00000044 0000005 00000044 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 00000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 0000005 00000005 0000005	00000005 000000015 00000015 00000015 00000020 00000025 00000035 00000045 00000045 00000045 00000055 00000050 00000050 00000060 00000065	512 byte dat 00000006 00000016 00000016 00000016 00000026 00000026 00000036 00000035 00000046 00000046 00000046 00000056 00000056 00000066 00000066	a @ LBA=0 00000007 0000000F 00000017 00000027 00000027 00000037 00000037 00000037 00000047 00000047 00000047 00000057 0000005F 00000067 00000067	
[OK] Start LBA Dump data from 0000: 00000000 0020: 00000008 0040: 00000018 0080: 00000028 0080: 00000028 00C0: 00000028 00C0: 00000038 0160: 00000040 0120: 00000048 0140: 00000050 0160: 00000050 0160: 00000068 0180: 00000068 0160: 00000078 0160: 00000078	$\begin{array}{c} & & & & & & & & \\ & & &$	00000002 00000002 00000012 0000012 00000012 00000022 00000022 00000032 00000032 00000042 00000042 00000052 00000052 0000052 0000052 0000052 0000062 0000062	theader 00000003 0000000B 00000013 00000023 0000002B 0000002B 0000003B 0000003B 0000004B 00000043 0000005B 0000005B 0000005B 0000006B 0000006B 00000073 0000007B	D0000004 00000000 0000000 00000014 00000012 00000024 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000034 00000054 00000052 00000064 00000064 00000074 00000072	00000005 000000015 00000015 00000025 00000025 00000020 00000035 00000045 00000045 00000045 00000055 00000055 00000055 00000055 000000	512 byte dat 00000006 00000016 0000001E 00000026 0000002E 00000036 0000003E 0000004E 0000004E 00000056 0000005E 0000005E 0000005E 0000005E 0000005E 0000005E	a @ LBA=0 00000007 00000017 00000017 00000027 00000027 00000037 00000037 0000004F 0000004F 00000057 00000057 00000057 0000006F 00000067 00000067	Ш

Figure 3-13 Example of Dump menu when input 'p'

Main menu is displayed when user input 'e' to exit Dump menu, as shown in Figure 3-14.

Dump	data [P]revious, [N]ext, or	[E]xit	: е	
1	lain Menu [Version: 1.0]			
[0] :	Identify SSD			
[1]:	Write SSD			
[2] :	Read SSD			
[3] :	Dump SSD			
[9] :	Exit Test			100
				-
Enter	• command:			-

Figure 3-14 Exit Dump SSD menu



3.5 Exit Test

Select '9' to exit test application.

1	Main Menu (Version: 1.0)	-
[0]	: Identify SSD	
[1] :	: Write SSD	
[2]	: Read SSD	
[3]	: Dump SSD	
[9]	: Exit Test	
		17

Figure 3-15 Exit test application



4 Revision History

Revision	Date	Description
1.0	19-Feb-18	Initial version release