

SATA AHCI-IP Demo Instruction

Rev1.2 9-Nov-16

This document describes the instruction to run SATA AHCI-IP and SATA-IP on ZC706/Zynq Mini-ITX 7Z100 for SATA-II/III device access by ARM CPU which runs PetaLinux2013.10 OS. In the demo, FPGA can boot and is configured by SD Card.

1 Environment Setup

To demo AHCI-IP on FPGA development board, please prepare following hardware.

- 1) Supported FPGA development board: ZC706/Zynq Mini-ITX 7Z100
- 2) PC with Serial port
- 3) AB09-FMCRAID board for ZC706 or SATA cable for Zynq Mini-ITX.
- 4) SATA-II/III device
- 5) Xilinx Power adapter for Xilinx board or ATX power supply for Zynq Mini-ITX board
- 6) mini/micro USB cable for Serial console connecting between FPGA board and PC
- 7) SD/microSD card with boot loader (BOOT.BIN) and LINUX image (image.ub), downloaded from http://www.dgway.com/SATA-IP_X_E.html

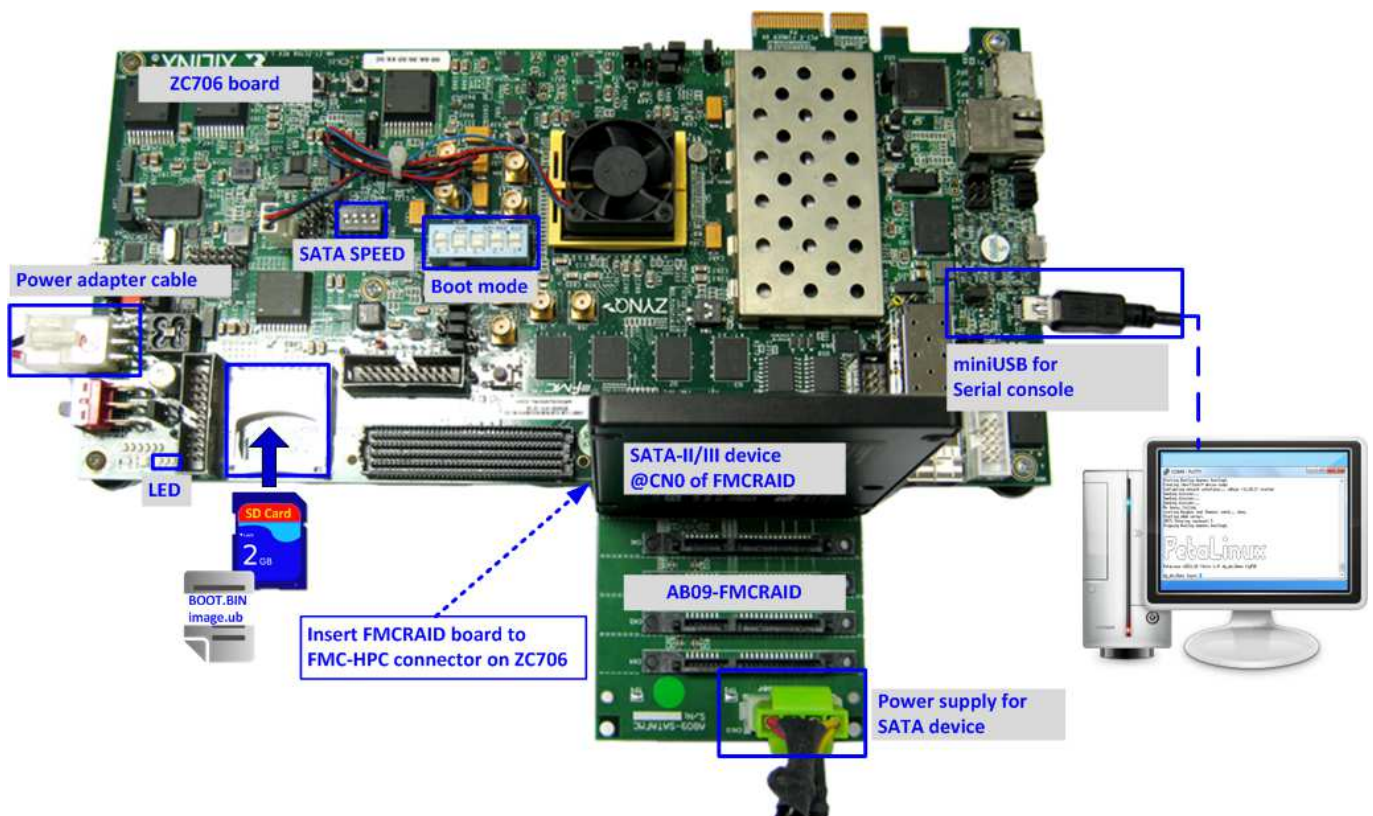


Figure 1-1 SATA AHCI-IP Demo Environment Setup on ZC706

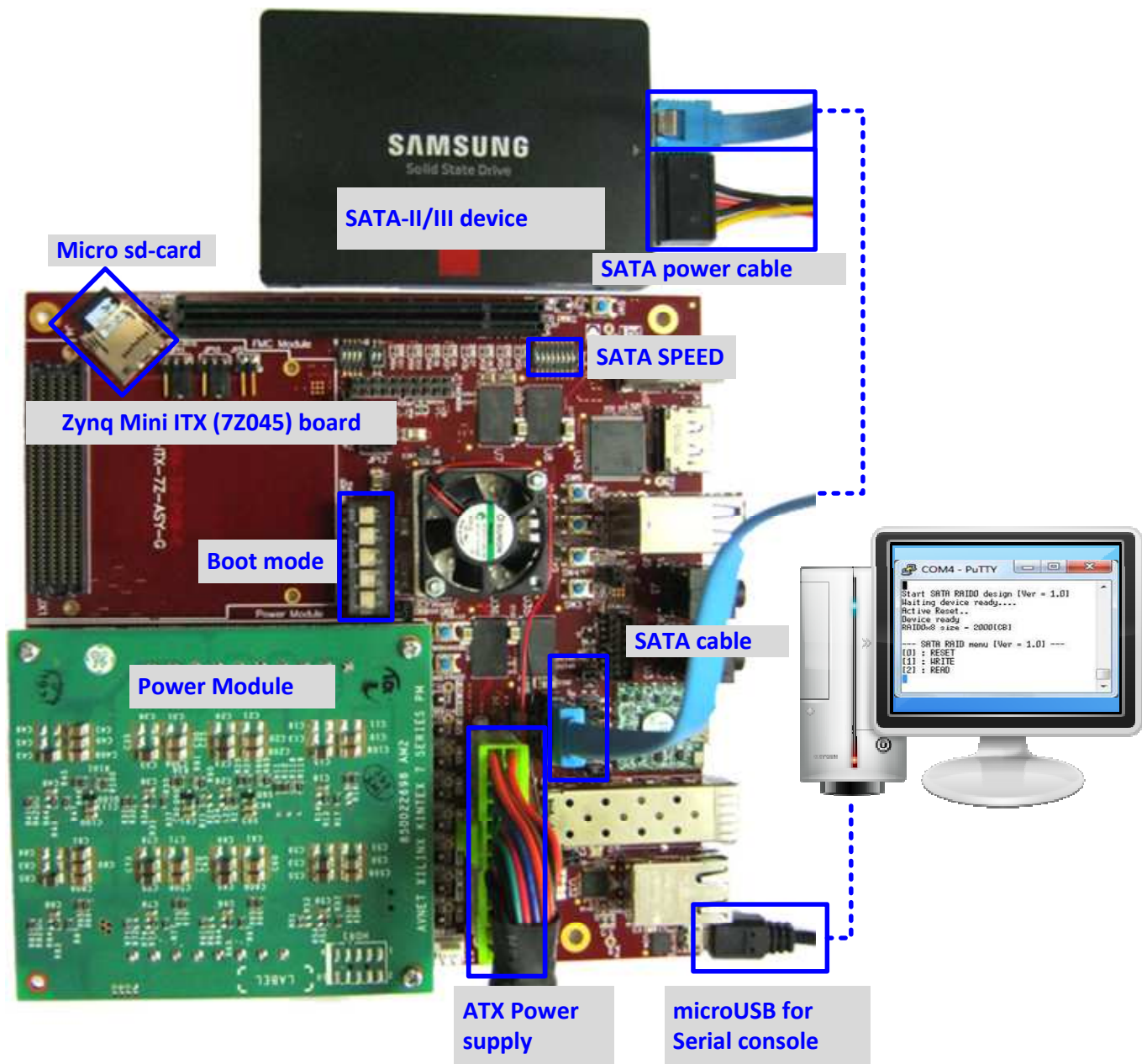


Figure 1-2 SATA AHCI-IP Demo Environment Setup on Zynq Mini-ITX 7Z100

2 Hardware setup

- 1) Copy BOOT.BIN and image.ub to SD card and insert to ZC706 /Zynq Mini-ITX 7Z100 board (SD Card is FAT32 format)

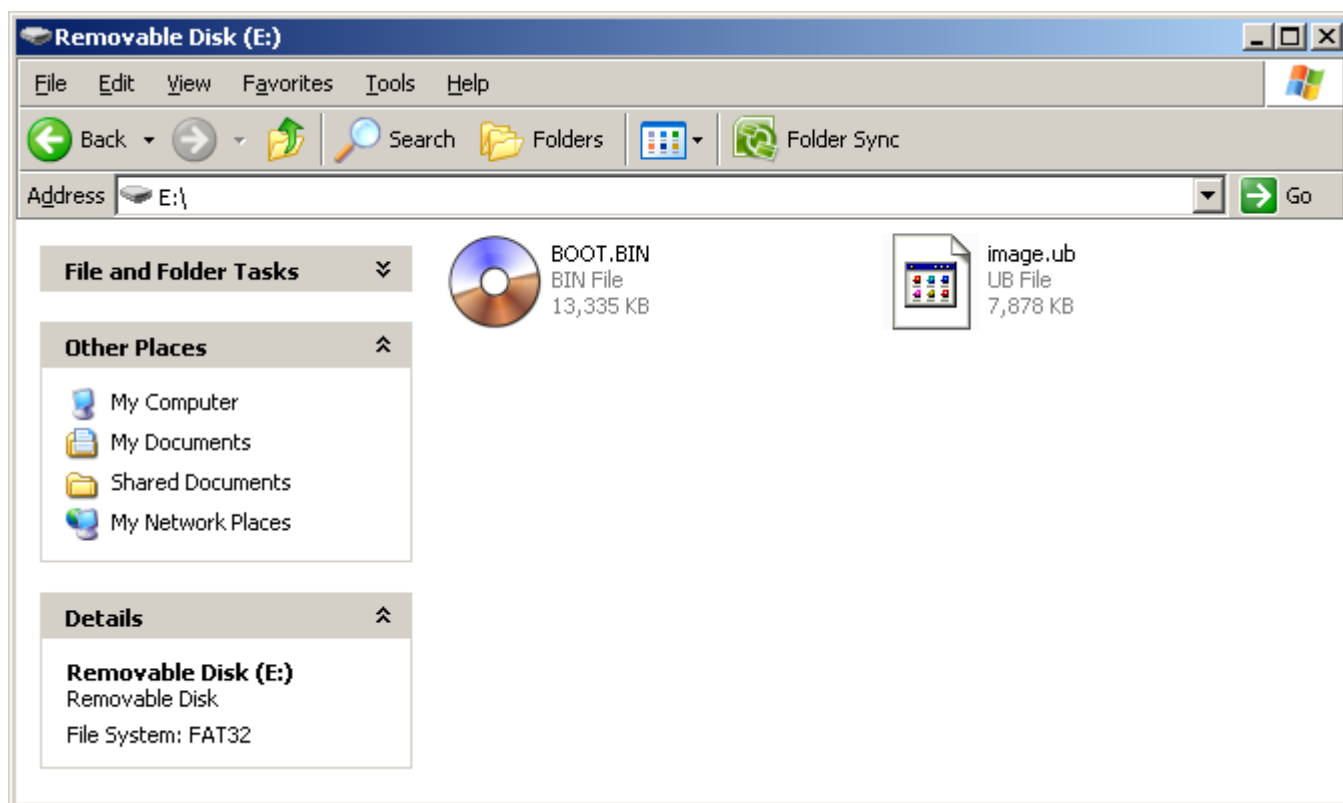


Figure 2-1 SD Card Image for SATA AHCI-IP Demo

- 2) Check board power is OFF
- 3) Set Boot mode (SW11 for ZC706 and SW7 for Zynq Mini-ITX board) = "00110" to select configuration option = SD mode, as shown in Figure 2-2

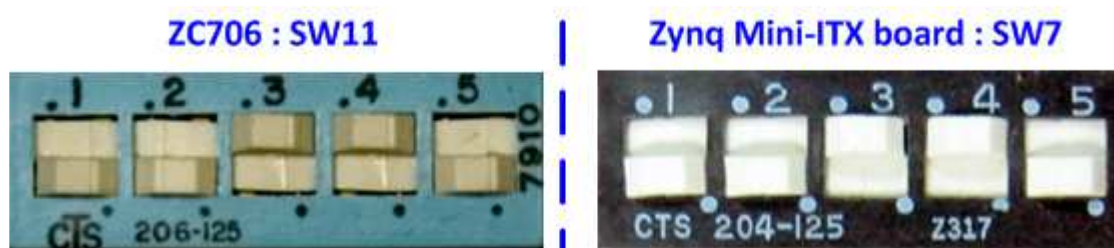


Figure 2-2 Boot mode Setting

- 4) Connect the SATA-II/III device to the board by the following steps.
 - a) For Xilinx development (ZC706),
 - Connect AB09-FMCRAID board to FMC(1)-HPC connector on Xilinx development board.
 - Connect SATA-II/III device to CN0 on FMCRAID board
 - Connect power to power connector on FMCRAID board.

The connections are shown in Figure 2-3

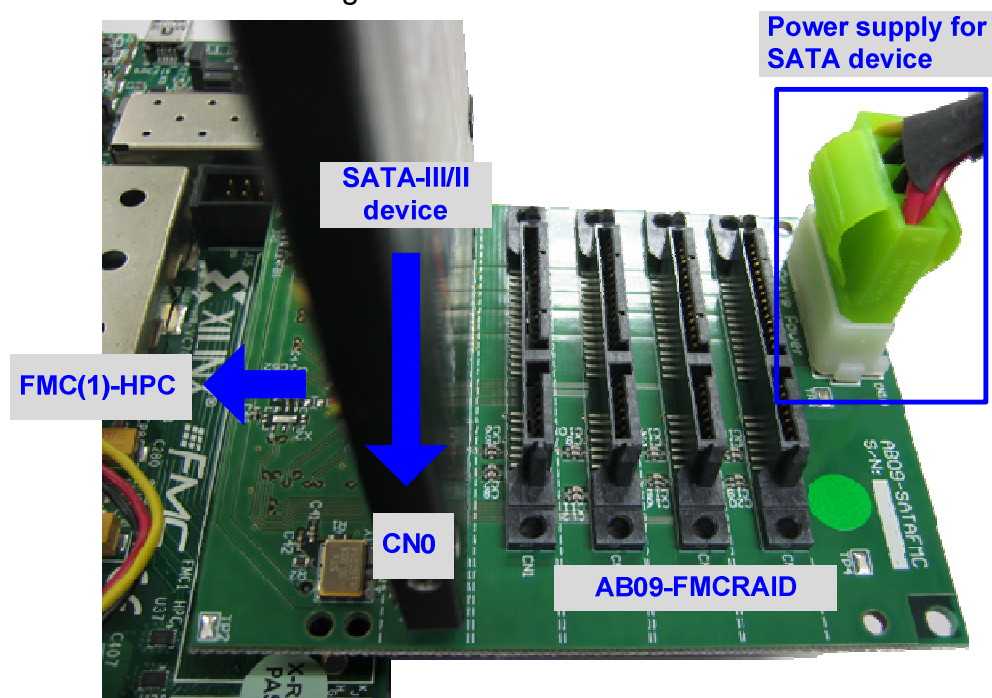


Figure 2-3 AB09-FMCRAID connection

- b) For Zynq Mini ITX board
 - Connect the device to the SATA connector (J12) on the board using SATA cable.
 - Connect SATA Power cable to the device

The connections are shown in Figure 2-4

SATA Cable between J12 and SATA Device

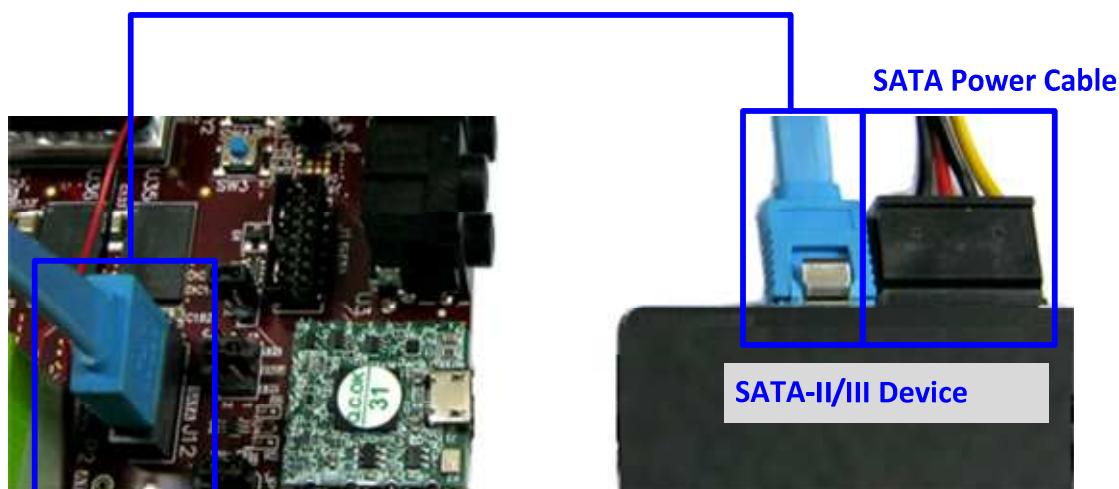


Figure 2-4 SATA-II/III device connection for Zynq Mini ITX

- 5) Set DIPSW bit 1-2 to select SATA speed mode.

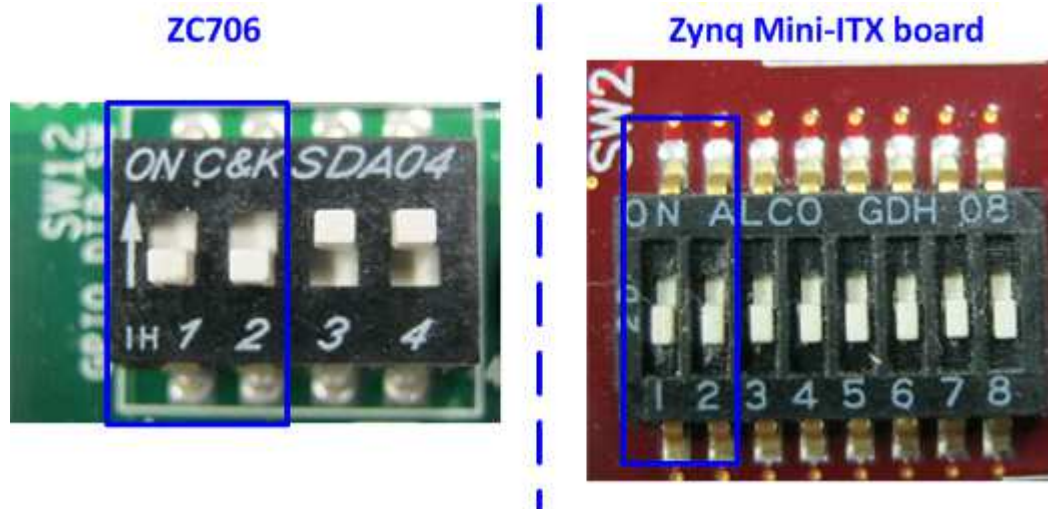


Figure 2-5 DIPSW to select SATA speed mode

DIPSW[2]	DIPSW[1]	Description
'1'	'1'	Fixed-speed at SATA3 (6.0 Gbps)
'1'	'0'	Fixed-speed at SATA2 (3.0 Gbps)
'0'	'X'	Auto-speed negotiation mode

Table 2-1 Description of DIPSW for SATA speed

- 6) Connect mini/micro USB cable from FPGA board to PC for Serial console.
- 7) Power on FPGA development board and power supply for SATA device. Now FPGA will start up the operation, and print the message to serial port.
- 8) Open Serial monitoring software such as HyperTerminal. Terminal settings should be (Baud Rate=115,200, Data=8 bit Non-Parity, Stop=1). If Serial console is opened too late, all boot messages will be printed out.

- 9) After disk initialization, check GPIO LEDs status on ZC706 board at LED0 and LED1. Both LEDs must be ON, as shown in Figure 2-6. Each LED description is described as follows.

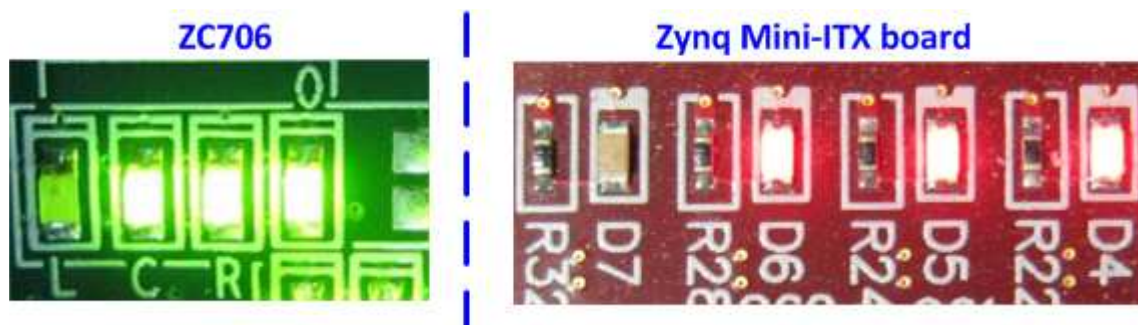


Figure 2-6 LED status after system set up complete on SATA-3 speed

LED	ON	OFF
0	OK	150 MHz of SATA clock on FMCRAID board cannot lock. Please check 150 MHz clock source on FMCRAID board.
1/R	OK	SATA-IP cannot detect SATA device. Please check SATA device and the connection.
2/C	SATA-III	SATA-II
3/L	SATA AHCI-IP in processing	SATA AHCI-IP in idle

Table 2-2 LED Status of AHCI reference design

- 10) On PC serial console, please wait PetaLinux boot-up until login required, as shown in Figure 2-7.

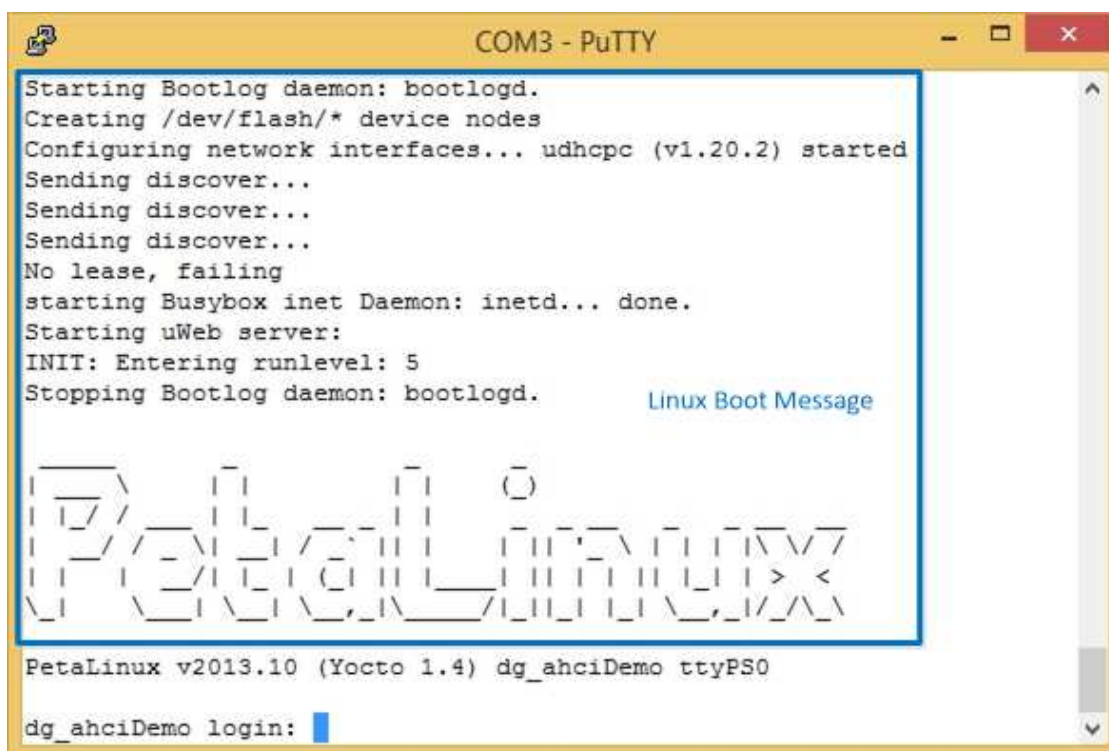


Figure 2-7 Linux Bootup

3 Linux Setup

- 1) User login on the demo is follows.

Login : root

Password : root

After login, system is ready to receive user command.



Figure 3-1 Linux Login

- 2) To run SATA AHCI-IP demo, two modules are required to insert, i.e libahci.ko (common AHCI SATA low-level routines) and dg_ahciDemo.ko (AHCI SATA platform driver). Both are stored in "/home/root/driver" directory. To insert module, use following command.

>> insmod /home/root/driver/libahci.ko skip_cpu_sync=1

>> insmod /home/root/driver/dg_ahciDemo.ko

- 3) After insert modules, disk information will be displayed, as shown in Figure 3-2. Now SATA device is ready to use.

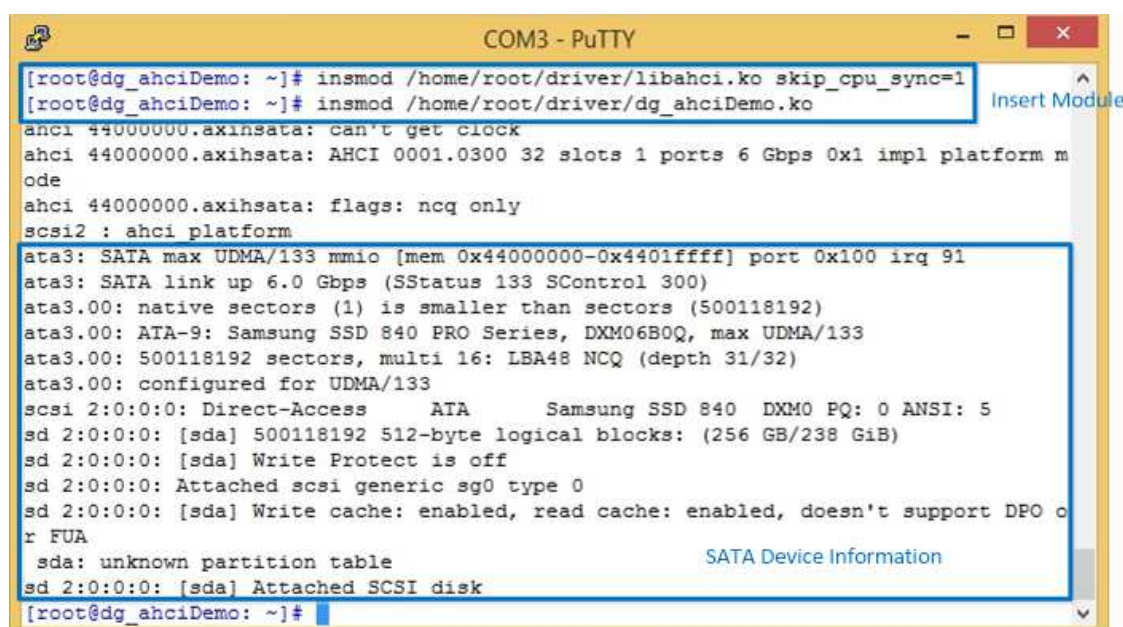


Figure 3-2 Insert module to linux kernel

4 Example Linux command

4.1 Create Disk Partition

To create new disk partition, user can follow the example shown in Figure 4-1.

```
>> fdisk /dev/sda
```

Call the tool to manage disk partition.

```
>> n
```

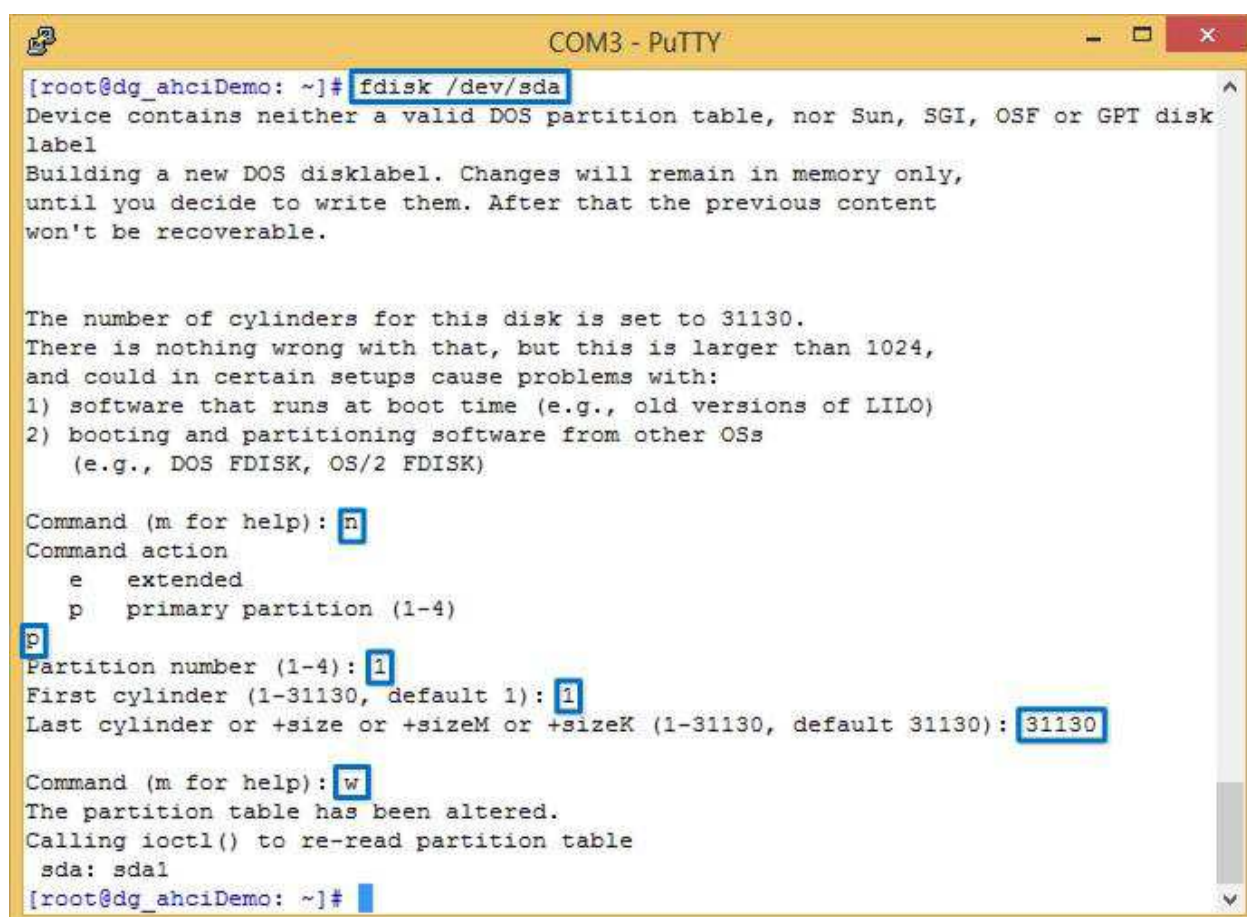
Create new partition.

Select the option following the recommended value in the tool.

```
>> w
```

Write table to the disk.

Now one partition named sda1 has been created in the disk.



```
COM3 - PuTTY
[root@dg_ahciDemo: ~]# fdisk /dev/sda
Device contains neither a valid DOS partition table, nor Sun, SGI, OSF or GPT disk
label
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that the previous content
won't be recoverable.

The number of cylinders for this disk is set to 31130.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
 1) software that runs at boot time (e.g., old versions of LILO)
 2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-31130, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-31130, default 31130): 31130

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table
sda: sda1
[root@dg_ahciDemo: ~]#
```

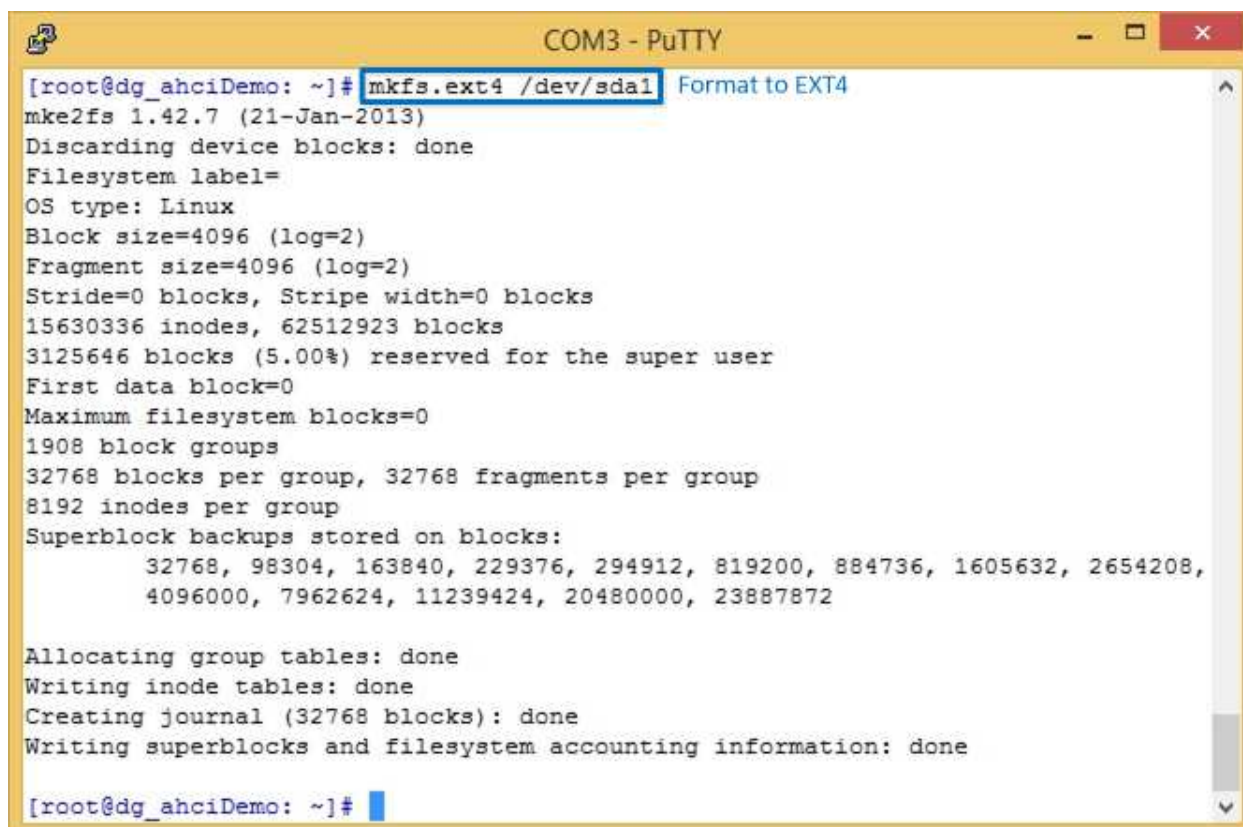
Figure 4-1 fdisk command

Note: User can type 'm' to show all fdisk options.

4.2 Format Disk

To format the disk, user needs to select file system type such as FAT, EXT4. This example shows only the command to format to EXT4 by typing following command.

```
>> mkfs.ext4 /dev/sda1
```



```
COM3 - PuTTY
[root@dg_ahciDemo: ~]# mkfs.ext4 /dev/sda1 Format to EXT4
mke2fs 1.42.7 (21-Jan-2013)
Discarding device blocks: done
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
15630336 inodes, 62512923 blocks
3125646 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=0
1908 block groups
32768 blocks per group, 32768 fragments per group
8192 inodes per group
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
    4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

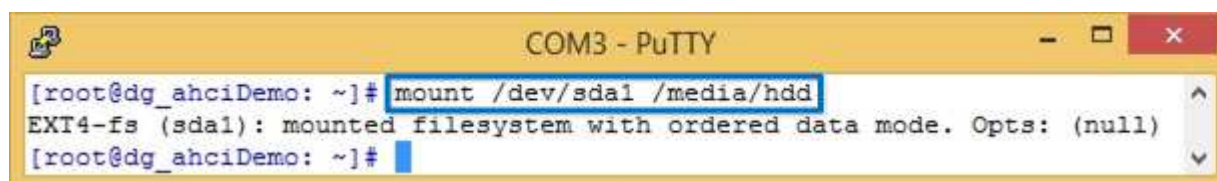
[root@dg_ahciDemo: ~]#
```

Figure 4-2 Format disk

4.3 Mount Disk

Before running any application to access the disk by file system such as Bonnie++, disk must be mounted firstly by following command.

```
>> mount /dev/sda1 /media/hdd
```



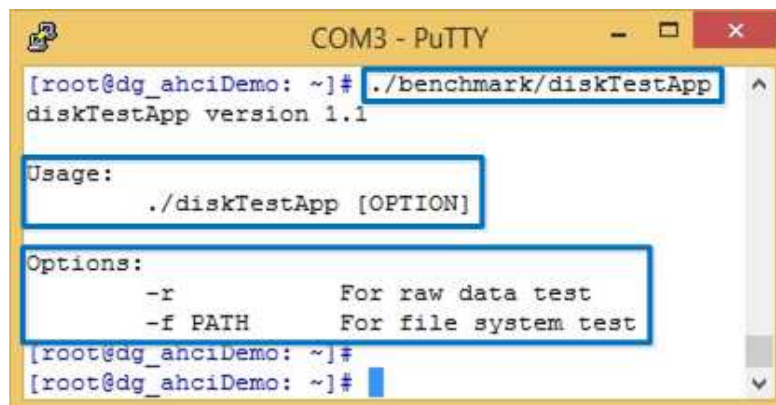
```
COM3 - PuTTY
[root@dg_ahciDemo: ~]# mount /dev/sda1 /media/hdd
EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
[root@dg_ahciDemo: ~]#
```

Figure 4-3 Mount disk

5 Performance test

This topic shows the example application to test disk performance. Two test applications are used, i.e. diskTestApp and Bonnie++. diskTestApp is the test application developed by Design Gateway to check write/read performance in both raw data format and file system.

5.1 Performance test by diskTestApp



```

COM3 - PuTTY
[root@dg_ahciDemo: ~]# ./benchmark/diskTestApp
diskTestApp version 1.1

Usage:
    ./diskTestApp [OPTION]

Options:
    -r          For raw data test
    -f PATH     For file system test
[root@dg_ahciDemo: ~]#
[root@dg_ahciDemo: ~]#

```

Figure 5-1 diskTestApp usage

As shown in Figure 5-1, diskTestApp can run in two data formats, i.e. raw data or file system.

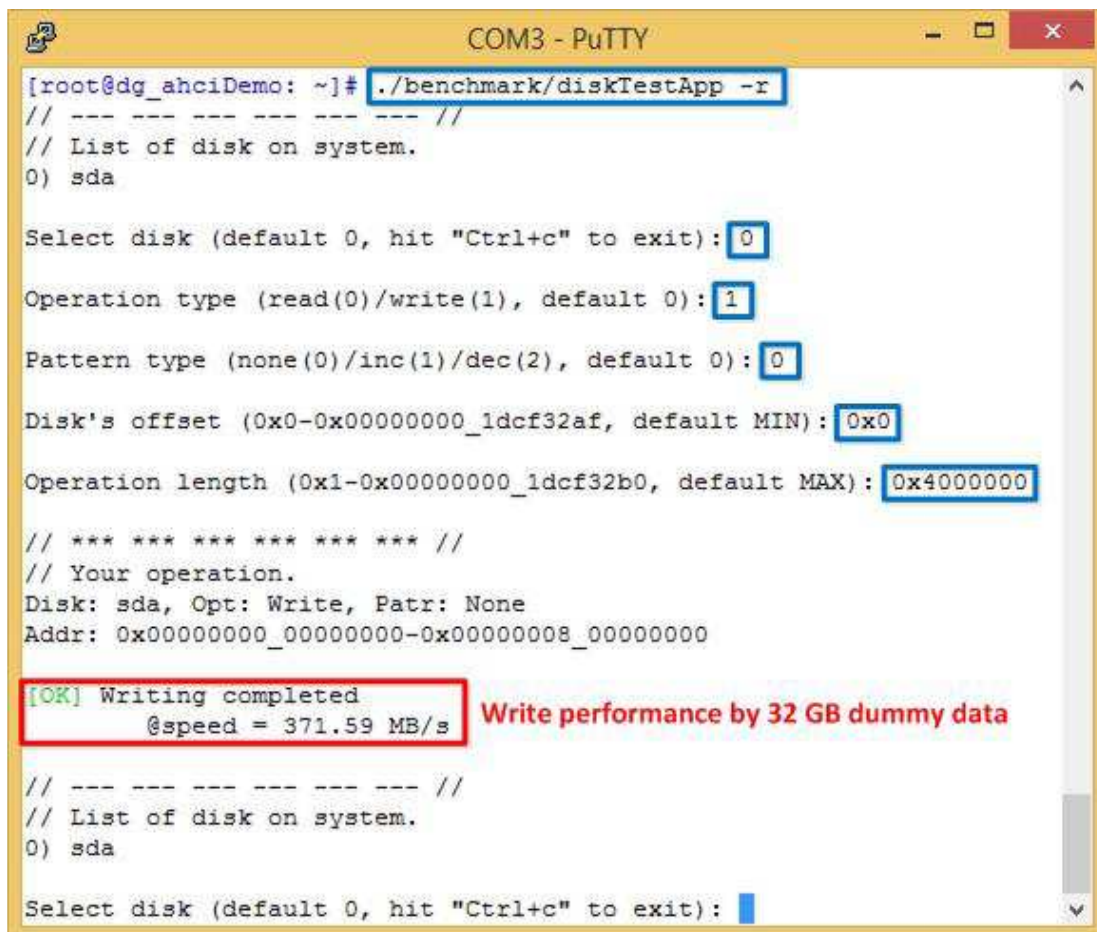
Warning: If running raw data test, file system in that disk partition will be lost.

5.1.1 Raw Data Mode

Type command “./benchmark/diskTestApp -r” to run test application in raw data format. Five input parameters are required, i.e.

- 1) Disk selection to select the disk to test performance
- 2) Operation type: ‘0’-Read disk test, ‘1’-Write disk test
- 3) Test pattern:
 - ‘0’: Write by dummy data or read without verification
 - ‘1’: Write or verify by 32-bit increment pattern
 - ‘2’: Write or verify by 32-bit decrement pattern
- 4) Disk offset: Disk start address in sector unit to write/read data. 0x prefix is added for hex unit input while default value without prefix is decimal unit.
- 5) Operation length: Transfer length in sector unit to write/read data. 0x prefix is added for hex unit input while default value without prefix is decimal unit.

Figure 5-2 and Figure 5-3 show the example of write test in raw data mode by dummy data and increment data. Figure 5-4 and Figure 5-5 show the example of read test in raw data mode without and with data verification. Comparing to increment/decrement pattern, using dummy mode for both write and read will achieve better performance because CPU resource is not used to fill or verify the data.



```

COM3 - PuTTY
[root@dg_ahciDemo: ~]# ./benchmark/diskTestApp -r
// --- --- --- --- --- //
// List of disk on system.
0) sda

Select disk (default 0, hit "Ctrl+c" to exit): 0
Operation type (read(0)/write(1), default 0): 1
Pattern type (none(0)/inc(1)/dec(2), default 0): 0
Disk's offset (0x0-0x00000000_1dcf32af, default MIN): 0x0
Operation length (0x1-0x00000000_1dcf32b0, default MAX): 0x4000000

// *** --- --- --- --- --- *** //
// Your operation.
Disk: sda, Opt: Write, Patr: None
Addr: 0x00000000_00000000-0x00000008_00000000

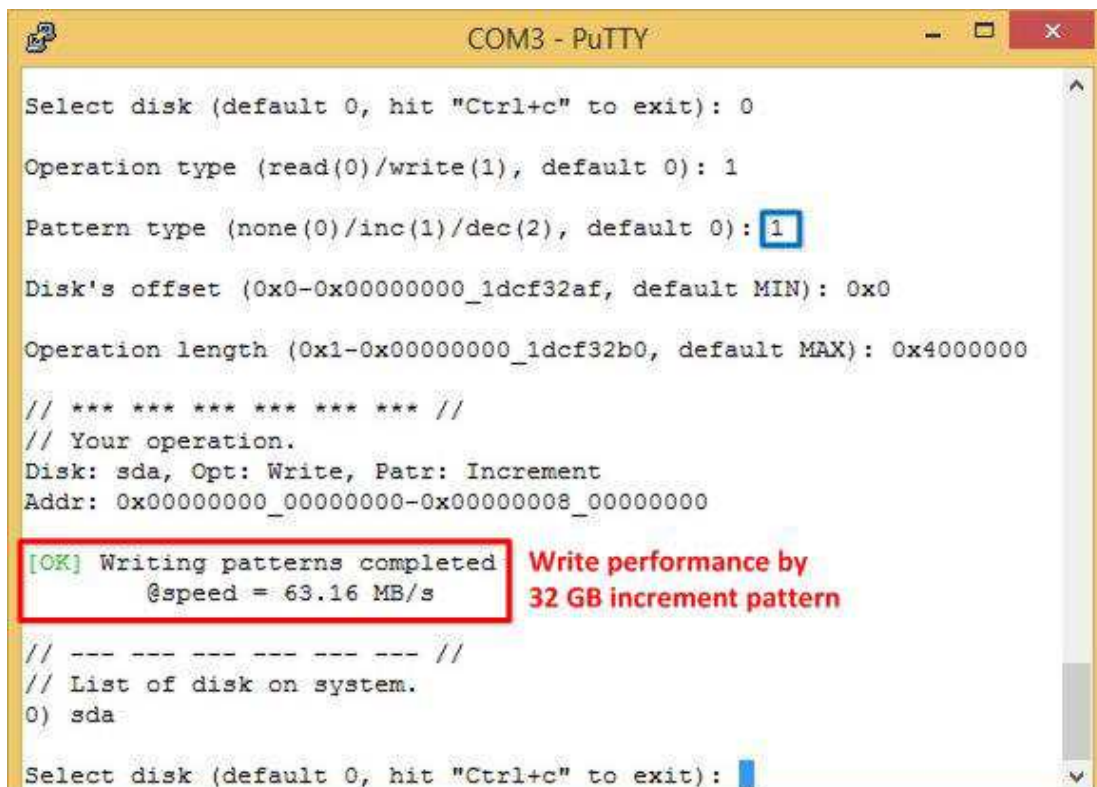
[OK] Writing completed
    @speed = 371.59 MB/s
Write performance by 32 GB dummy data

// --- --- --- --- --- //
// List of disk on system.
0) sda

Select disk (default 0, hit "Ctrl+c" to exit):

```

Figure 5-2 Write performance in raw data mode by dummy data



```

COM3 - PuTTY

Select disk (default 0, hit "Ctrl+c" to exit): 0
Operation type (read(0)/write(1), default 0): 1
Pattern type (none(0)/inc(1)/dec(2), default 0): 1
Disk's offset (0x0-0x00000000_1dcf32af, default MIN): 0x0
Operation length (0x1-0x00000000_1dcf32b0, default MAX): 0x4000000

// *** --- --- --- --- --- *** //
// Your operation.
Disk: sda, Opt: Write, Patr: Increment
Addr: 0x00000000_00000000-0x00000008_00000000

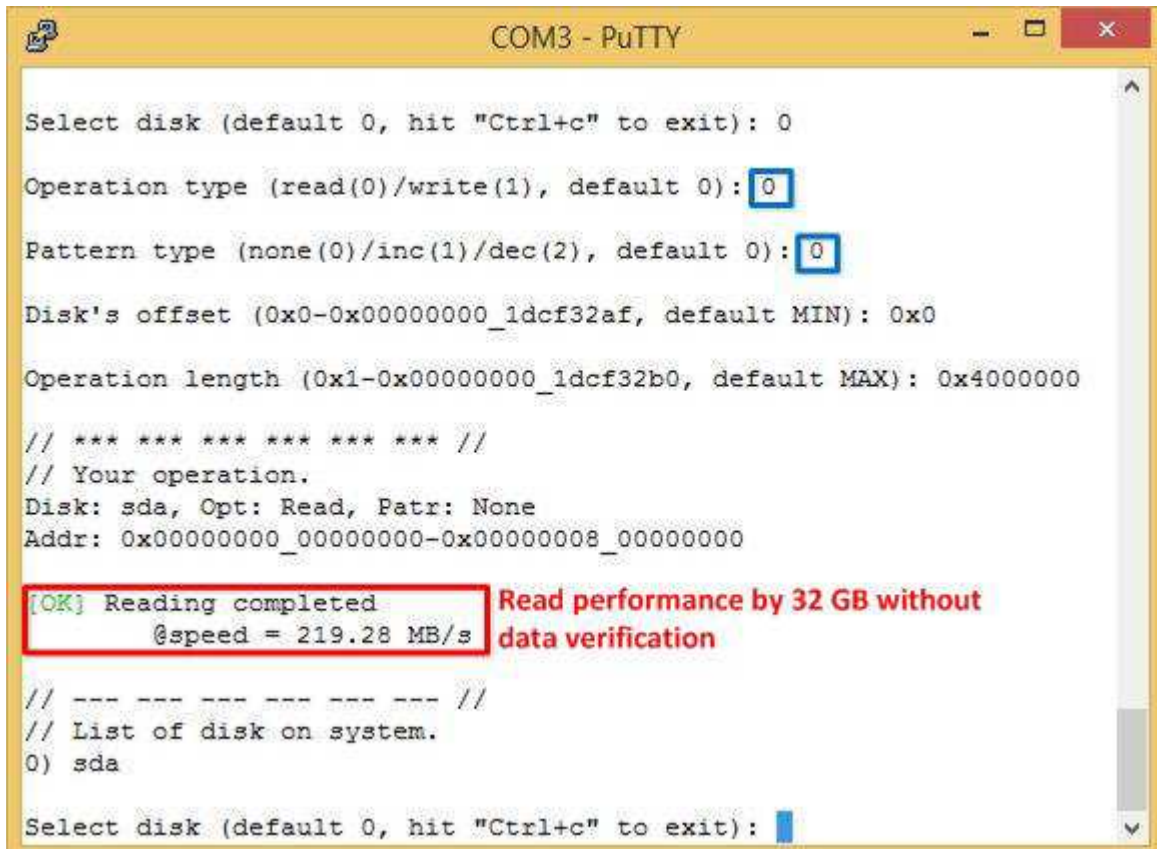
[OK] Writing patterns completed
    @speed = 63.16 MB/s
Write performance by
32 GB increment pattern

// --- --- --- --- --- //
// List of disk on system.
0) sda

Select disk (default 0, hit "Ctrl+c" to exit):

```

Figure 5-3 Write performance in raw data mode by 32-bit increment data



```

COM3 - PuTTY

Select disk (default 0, hit "Ctrl+c" to exit): 0

Operation type (read(0)/write(1), default 0): 0

Pattern type (none(0)/inc(1)/dec(2), default 0): 0

Disk's offset (0x0-0x00000000_1dcf32af, default MIN): 0x0

Operation length (0x1-0x00000000_1dcf32b0, default MAX): 0x4000000

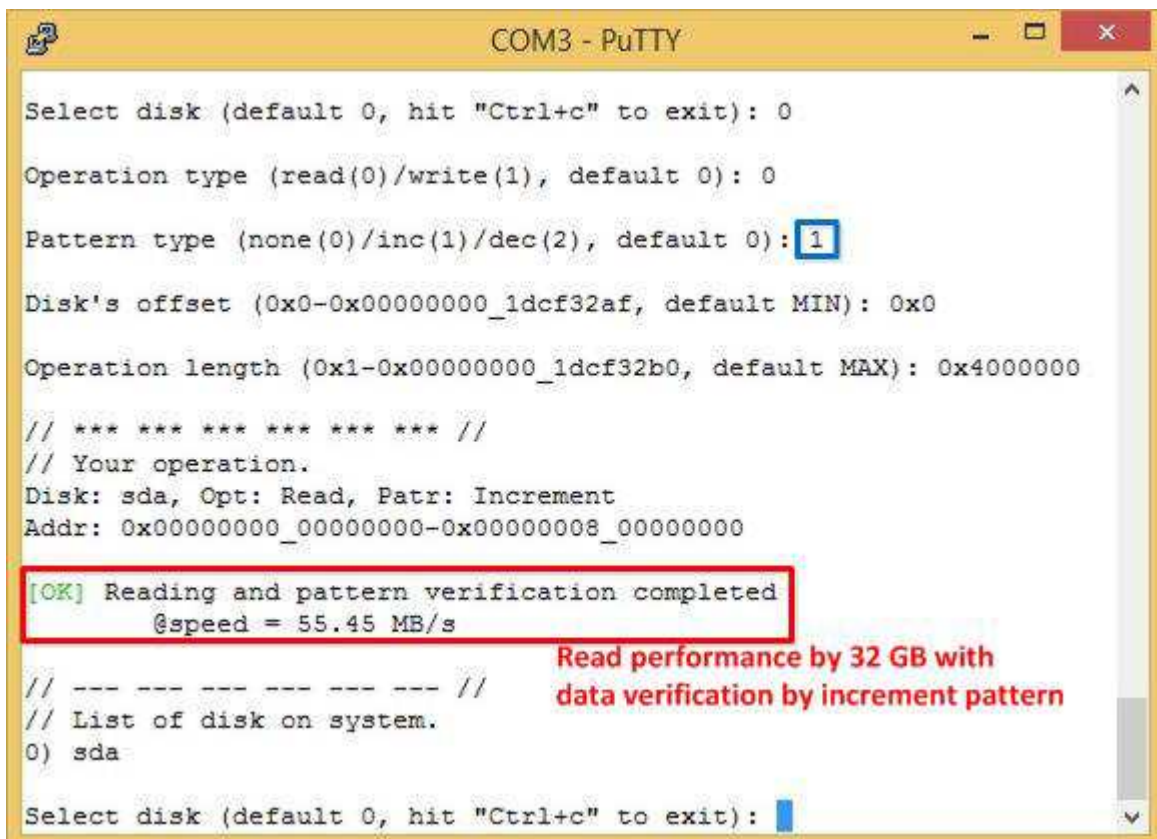
// *** *** *** *** *** //
// Your operation.
Disk: sda, Opt: Read, Patr: None
Addr: 0x00000000_00000000-0x00000008_00000000

[OK] Reading completed
    @speed = 219.28 MB/s

// --- --- --- --- --- //
// List of disk on system.
0) sda

Select disk (default 0, hit "Ctrl+c" to exit):
  
```

Figure 5-4 Read performance in raw data mode without data verification



```

COM3 - PuTTY

Select disk (default 0, hit "Ctrl+c" to exit): 0

Operation type (read(0)/write(1), default 0): 0

Pattern type (none(0)/inc(1)/dec(2), default 0): 1

Disk's offset (0x0-0x00000000_1dcf32af, default MIN): 0x0

Operation length (0x1-0x00000000_1dcf32b0, default MAX): 0x4000000

// *** *** *** *** *** //
// Your operation.
Disk: sda, Opt: Read, Patr: Increment
Addr: 0x00000000_00000000-0x00000008_00000000

[OK] Reading and pattern verification completed
    @speed = 55.45 MB/s

// --- --- --- --- --- //
// List of disk on system.
0) sda

Select disk (default 0, hit "Ctrl+c" to exit):
  
```

Figure 5-5 Read performance in raw data mode and verify by 32-bit increment data

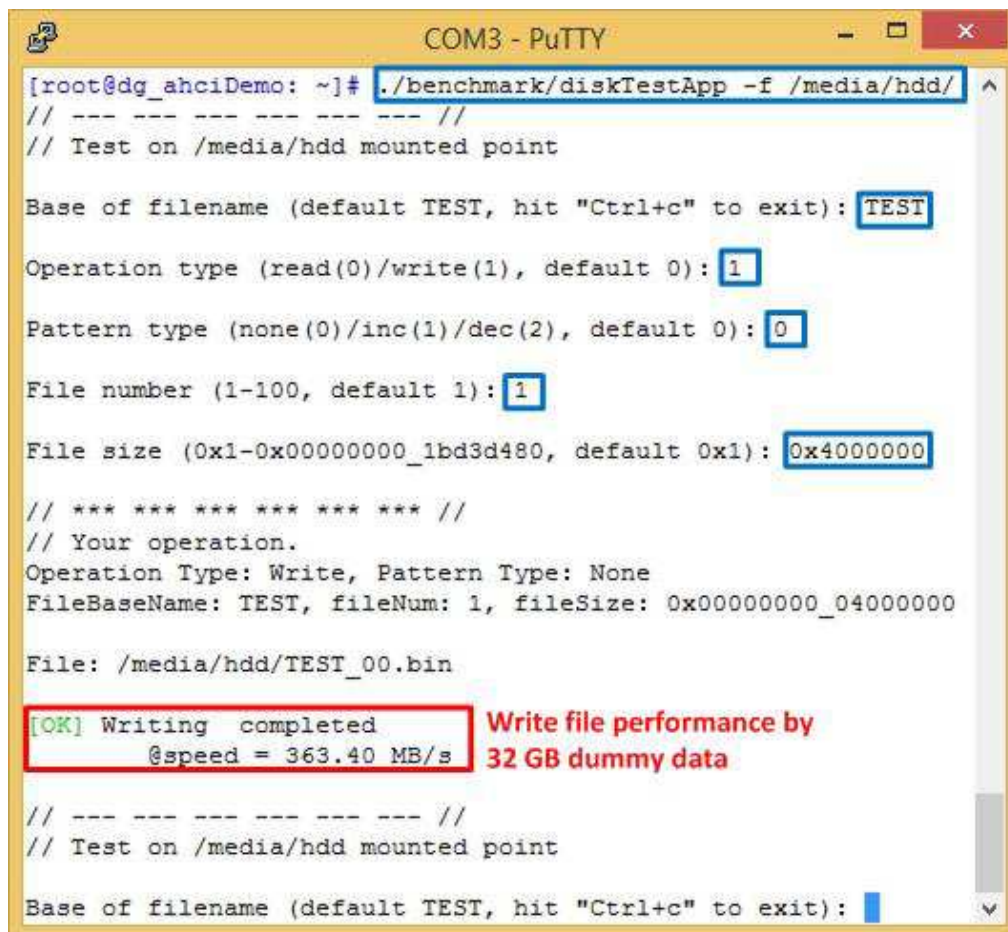
5.1.2 File System Mode

Before run test application in File system mode, user needs to create disk partition, format disk, and mount the disk.

Type “./benchmark/diskTestApp -f /media/hdd” to run the test in file system mode. Five input parameters are required, i.e.

- 1) File name input: File name to run the test
- 2) Operation type: ‘0’-Read file test, ‘1’-Write file test
- 3) Test pattern:
 - ‘0’: Write by dummy data or read without verification
 - ‘1’: Write or verify by 32-bit increment pattern
 - ‘2’: Write or verify by 32-bit decrement pattern
- 4) File number: Total number of files to run write/read file test
- 5) File size: Size of each file in sector unit to run write/read file test

Similar to raw data mode, when write by dummy pattern or read without data verification, performance will be better than increment/decrement pattern, as shown in Figure 5-6 - Figure 5-9.



```

COM3 - PuTTY
[root@dg_ahciDemo: ~]# ./benchmark/diskTestApp -f /media/hdd/
// --- --- --- --- --- //
// Test on /media/hdd mounted point

Base of filename (default TEST, hit "Ctrl+c" to exit): TEST
Operation type (read(0)/write(1), default 0): 1
Pattern type (none(0)/inc(1)/dec(2), default 0): 0
File number (1-100, default 1): 1
File size (0x1-0x000000000_1bd3d480, default 0x1): 0x40000000

// *** *** *** *** *** //
// Your operation.
Operation Type: Write, Pattern Type: None
FileName: TEST, fileNum: 1, fileSize: 0x000000000_040000000
File: /media/hdd/TEST_00.bin

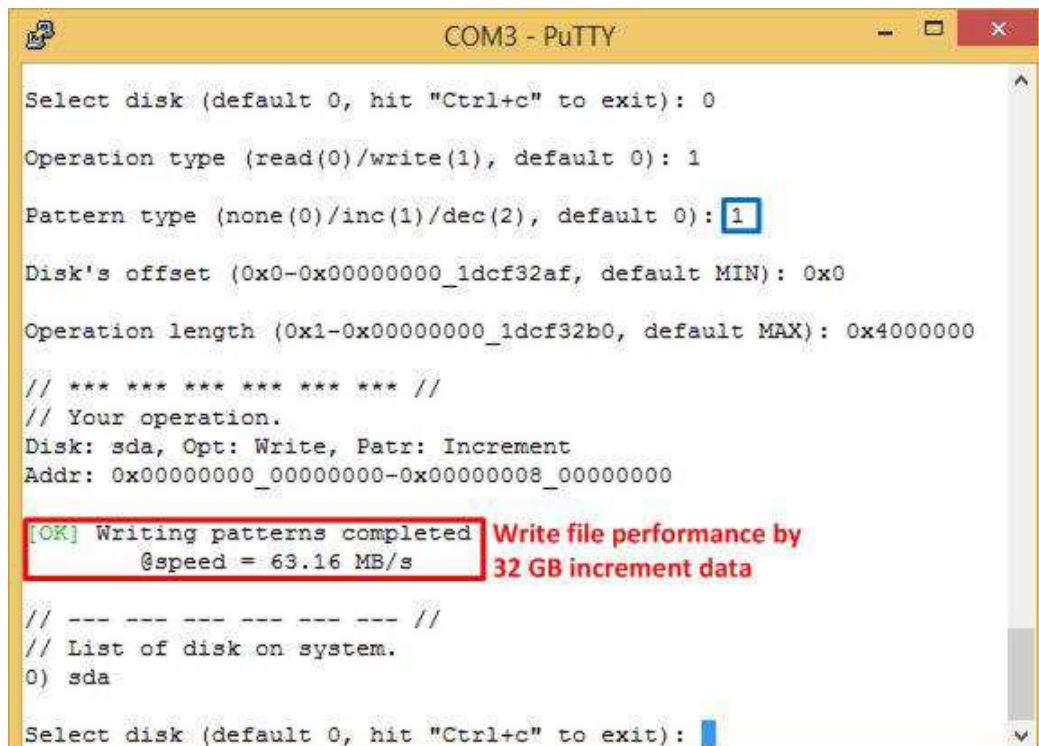
[OK] Writing completed      Write file performance by
    @speed = 363.40 MB/s    32 GB dummy data

// --- --- --- --- --- //
// Test on /media/hdd mounted point

Base of filename (default TEST, hit "Ctrl+c" to exit):

```

Figure 5-6 Write file performance by dummy data



```

COM3 - PuTTY

Select disk (default 0, hit "Ctrl+c" to exit): 0
Operation type (read(0)/write(1), default 0): 1
Pattern type (none(0)/inc(1)/dec(2), default 0): 1
Disk's offset (0x0-0x000000000_1dcf32af, default MIN): 0x0
Operation length (0x1-0x000000000_1dcf32b0, default MAX): 0x40000000

// *** *** *** *** *** //
// Your operation.
Disk: sda, Opt: Write, Patr: Increment
Addr: 0x000000000_000000000-0x000000008_000000000

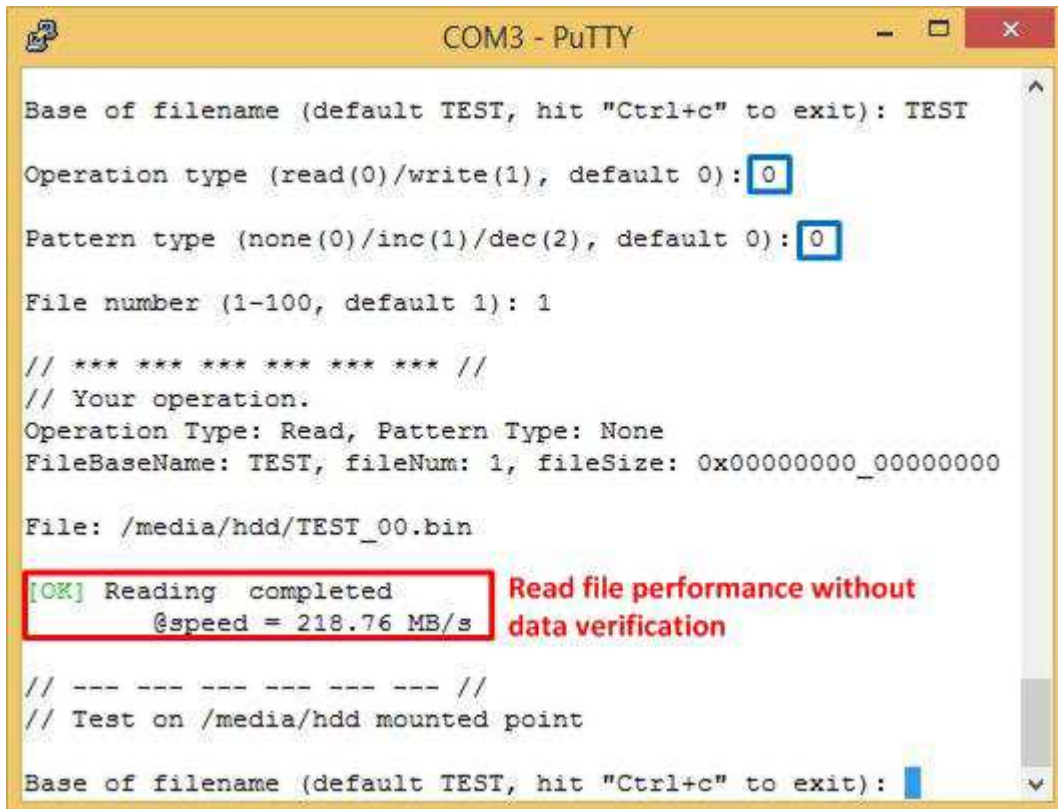
[OK] Writing patterns completed  Write file performance by
    @speed = 63.16 MB/s          32 GB increment data

// --- --- --- --- --- //
// List of disk on system.
0) sda

Select disk (default 0, hit "Ctrl+c" to exit):

```

Figure 5-7 Write file performance by 32-bit increment data



```

COM3 - PuTTY

Base of filename (default TEST, hit "Ctrl+c" to exit): TEST
Operation type (read(0)/write(1), default 0): 0
Pattern type (none(0)/inc(1)/dec(2), default 0): 0
File number (1-100, default 1): 1

// *** *** *** *** *** //
// Your operation.
Operation Type: Read, Pattern Type: None
FileBaseName: TEST, fileNum: 1, fileSize: 0x00000000_00000000
File: /media/hdd/TEST_00.bin

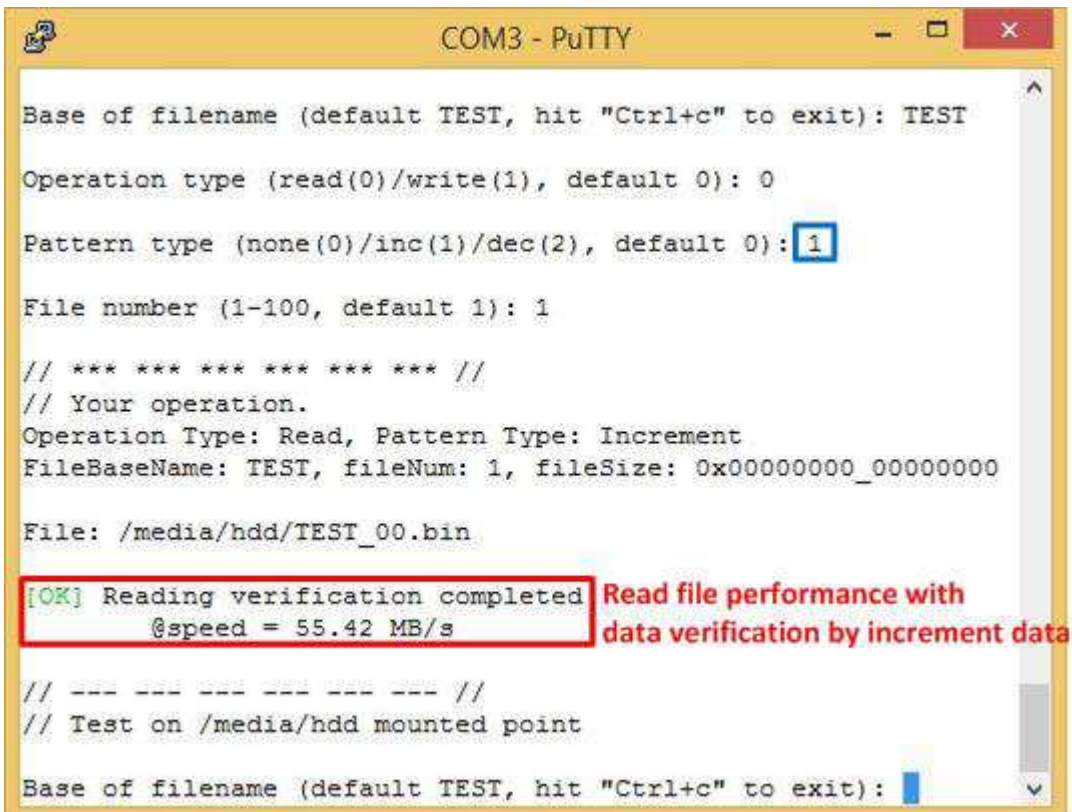
[OK] Reading completed          Read file performance without
    @speed = 218.76 MB/s        data verification

// --- --- --- --- --- //
// Test on /media/hdd mounted point

Base of filename (default TEST, hit "Ctrl+c" to exit):

```

Figure 5-8 Read file performance without data verification



```

COM3 - PuTTY

Base of filename (default TEST, hit "Ctrl+c" to exit): TEST
Operation type (read(0)/write(1), default 0): 0
Pattern type (none(0)/inc(1)/dec(2), default 0): 1
File number (1-100, default 1): 1

// *** *** *** *** *** //
// Your operation.
Operation Type: Read, Pattern Type: Increment
FileBaseName: TEST, fileNum: 1, fileSize: 0x00000000_00000000
File: /media/hdd/TEST_00.bin

[OK] Reading verification completed Read file performance with
    @speed = 55.42 MB/s             data verification by increment data

// --- --- --- --- --- //
// Test on /media/hdd mounted point

Base of filename (default TEST, hit "Ctrl+c" to exit):

```

Figure 5-9 Read file performance and verify by 32-bit increment data

5.2 Bonnie++ Software

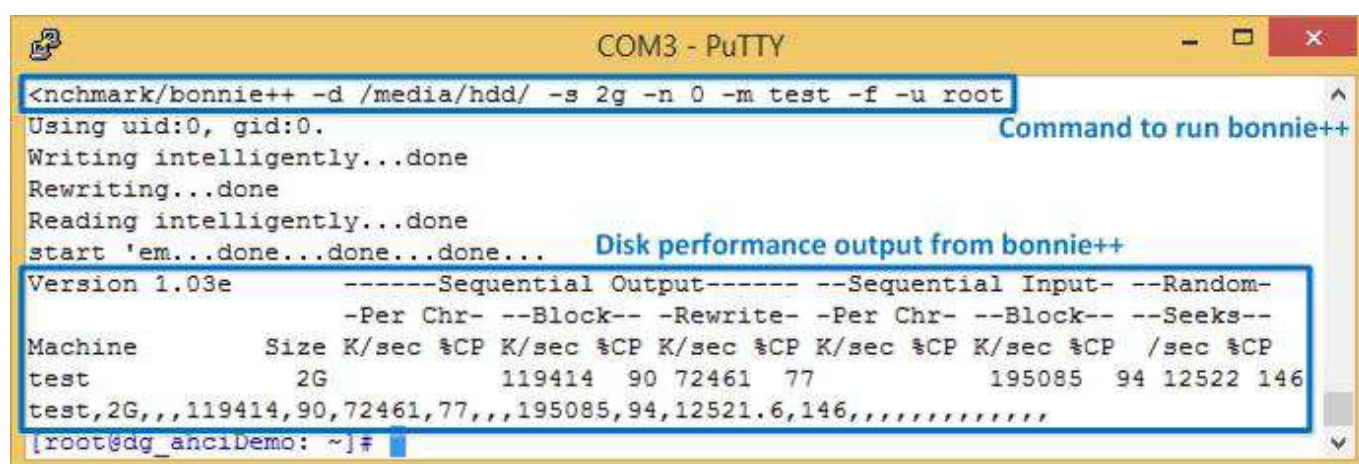
This topic shows how to test disk performance by using Bonnie++ software. The brief option of Bonnie++ software is belows.

```
bonnie++ [-d dir] [-s size(MB)[:chunk-size(b)]] [-n number-to-stat(*1024)
[:max-size[:min-size] [:num-directories]]] [-m machine-name] [-r ram-size-in-MB] [-x
number-of-tests] [-u uid-to-use:gid-to-use] [-g gid-to-use] [-q] [-f size-for-char-io]
[-b] [-D] [-p processes | -y p/s] [-z seed-num|-Z random-file]
```

More details about Bonnie++ user manual can be found from <http://linux.die.net/man/8/bonnie++>.

The example command to run Bonnie++ is follows.

```
>> ./benchmark/bonnie++ -d /media/hdd/ -s 2g -n 0 -m test -f -u root
```



```
COM3 - PuTTY
./benchmark/bonnie++ -d /media/hdd/ -s 2g -n 0 -m test -f -u root
Using uid:0, gid:0.
Writing intelligently...done
Rewriting...done
Reading intelligently...done
start 'em...done...done...done...
Version 1.03e
-----Sequential Output----- --Sequential Input- --Random-
-Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine      Size K/sec %CP K/sec %CP K/sec %CP K/sec %CP /sec %CP
test          2G      119414 90 72461 77      195085 94 12522 146
test,2G,,,119414,90,72461,77,,,195085,94,12521.6,146,,,,,
[root@edg_anc1Demo: ~]#
```

Figure 5-10 Test performance from Bonnie++ benchmark

6 Revision History

Revision	Date	Description
1.0	10-Nov-14	Initial version release
1.1	16-Jul-15	Add new test application
1.2	9-Nov-16	Zynq Mini-ITX 7Z100