SATA AHCI-IP SoC デモ手順書

Rev1.2J　2017/03/07

本ドキュメントは Intel(Altera)製 SoC 評価ボードにおいて SATA-IP コアおよび AHCI-IP コアを使って SATA デバイスを LinuxOS からアクセスするリファレンス・デザインを実機評価する手順について説明したものです。 評価可能なボードは CycloneV SX SoC、ArriaV ST SoC、Arria10SoC ボードです。 デバイスの制約から、Cyclone シリーズは SATA-Ⅱで、Arria シリーズは SATA-Ⅲでのみ動作可能です。本デモで使われる OS は Angstorm linux v2014.12 で、FPGA のコンフィグレーションとブートは microSD カードから実行されます。

1　ハードウエア環境

本 AHCI デモを実機評価するために図 1-1～図 1-3 に示すように、以下の環境を準備してください。

1) CycloneV SX/ArriaV ST/Arria10SoC 開発ボード (Intel 製)
2) DesignGateway 社製 SATA アダプタ基板
   - CycloneV SX SoC ボードの場合 AB11-HSMCSATA アダプタ基板
   - ArriaV ST/Arria10SoC ボードの場合 AB09-FMCRAID アダプタ基板
3) 2.5 インチ(あるいはアダプタケーブル経由にて別サイズのドライブも対応可)の SATA ドライブ
   - CycloneV SX ボードの場合 SATA-Ⅱまたは SATA-Ⅲドライブ（SATA-Ⅱモードで動作）
   - ArriaV ST/Arria10SoC ボードの場合 SATA-Ⅲドライブ(SATA-Ⅲモードで動作)
4) ブートイメージを格納した SD カード（DesignGateway 社 Web 登録サイトからユーザ登録の上ダウンロード可能）　http://www.dgway.com/SATA-IP_A.html
5) 評価ボード付属の USB ケーブルおよびシリアル・ターミナルソフトウエア導入済み PC

図 1-1: Cyclone V SX SoC ボードでの SATA AHCI-IP デモ環境

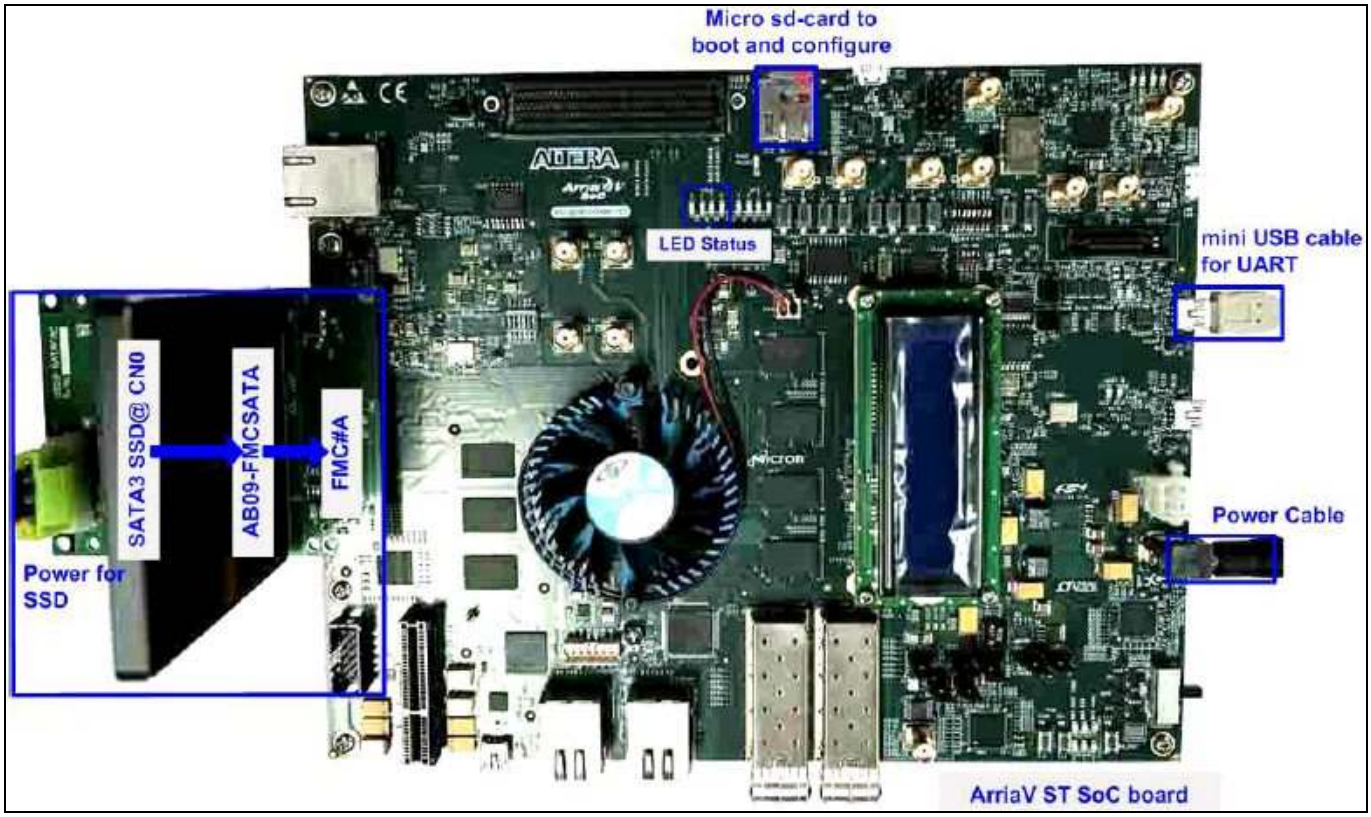

図 1-2: Arria V ST SoC ボードでの SATA AHCI-IP デモ環境

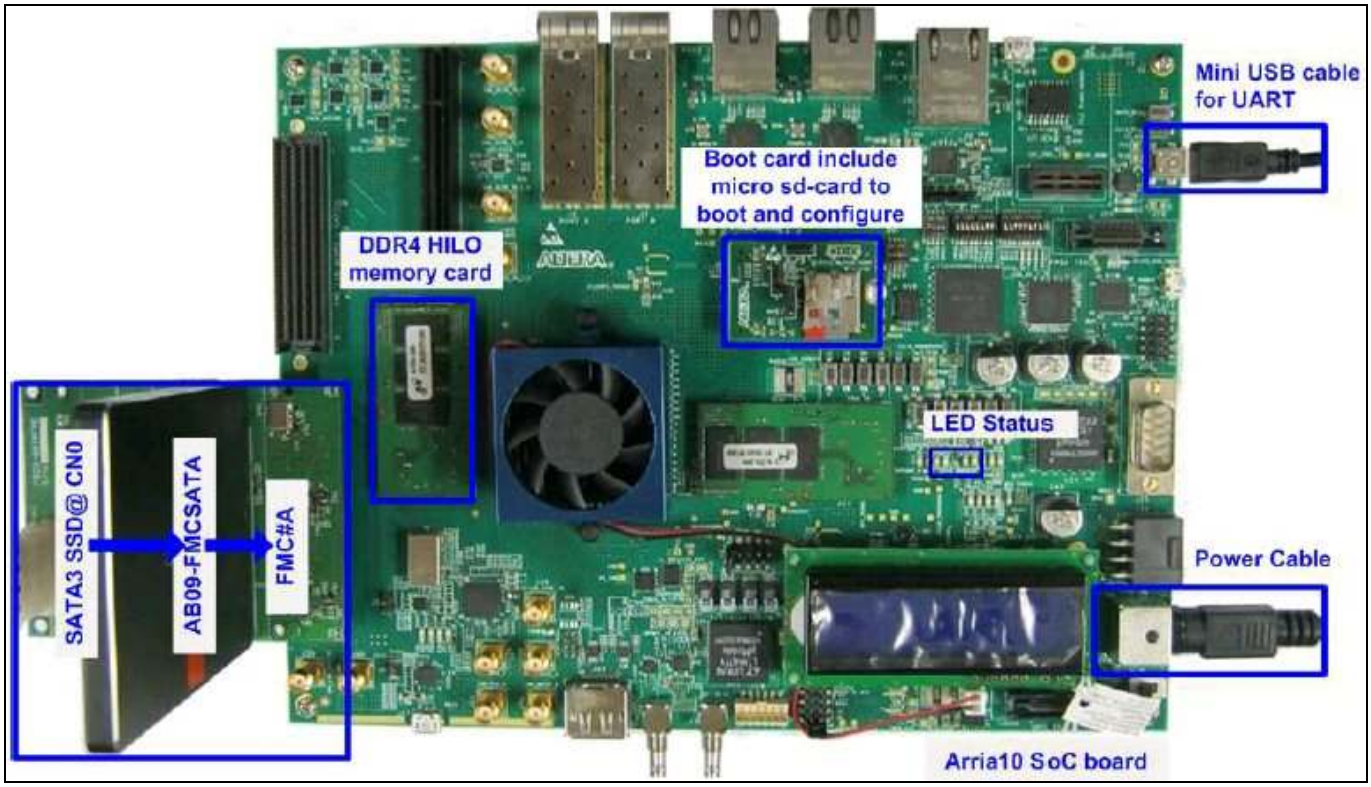

図 1-3: Arria10 SoC ボードでの SATA AHCI-IP デモ環境

2 ハードウエア設定

2.1 ブート用マイクロ SD カードの用意

- DesignGateway 社の登録ページよりダウンロードしたデモ用 SD カード・イメージ・ファイルを解凍します。解凍すると"sd_card_image_cyclone5/arria5/arria10.bin"ファイルが復元します。
- PC に microSD カードを接続し"Win32DiskImager"アプリケーションを起動します。
- "Win32DiskImager"にて図 2-1 のように"sd_card_image_cyclone5/arria5/arria10.bin"ファイルを microSD カードにコピーします。

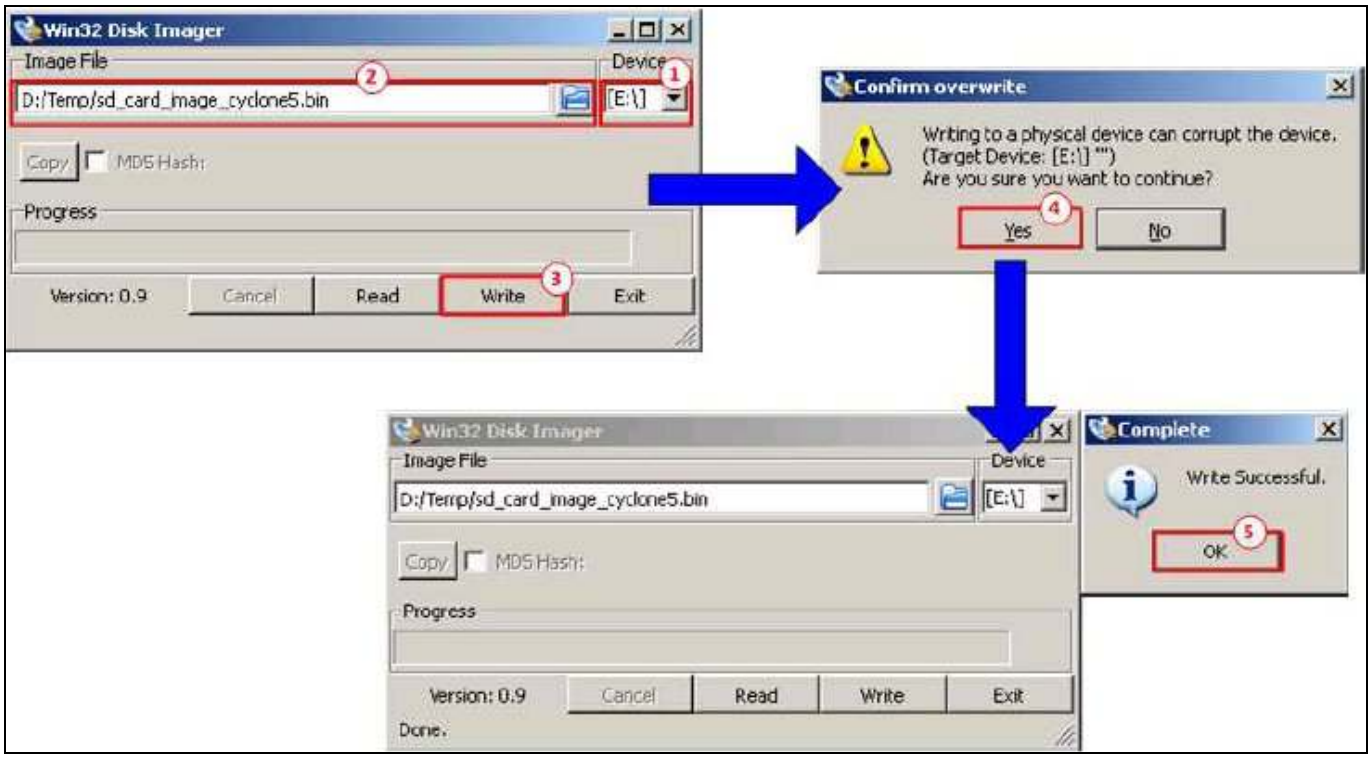

図 2-1: デモ用イメージ・ファイルの microSD カードへの書き込み

(1) microSD カード・ドライブを指定します
(2) ブラウズし"sd_card_image_cyclone5/arria5/arria10.bin"ファイルのあるパスを選択します
(3) "Write"ボタンを押下しデータを microSD へダンプします
(4) "Confirm overwrite"ウインドウが表示されるので"Yes"をクリックし書き込みを開始します
(5) 進捗ステータスが 0%から 100%になり完了メッセージが表示されるのを待ちます

2.2 FPGA ボードの設定

- FPGA 評価ボードとアダプタ基板の電源が OFF となっていることを確認します。
- ブート用 microSD カードを FPGA 評価ボードの SD カード・ソケットに挿入します。
- AB11-HSMCSATA アダプタまたは AB09-FMCRAID アダプタの CN0 に評価 SATA ドライブを装着します。
- 更にアダプタの 4 ピン電源コネクタに ATX 電源を接続します。
- アダプタを FPGA 評価ボードに接続します。
    - 注意：FPGA ひょ床ボードに複数の FMC/HMSC 拡張コネクタがある場合 port#A 側に接続します
- 評価ボード付属の USB ケーブルで FPGA 評価ボードと PC を接続します
- 評価ボードの JTAG チェーンを有効にします
  a) CycloneV SX/ArriaV ST SoC ボードの場合、図 2-2 のように SW4[1]/[2]/[4]を OFF として HPS,FPGA,MAX の JTAG チェーンをイネーブルにします

図 2-2: CycloneV SX/ArriaV ST SoC ボードの JTAG イネーブル設定

b) Arria10SoC ボードの場合、図 2-3 のように SW3[1]/[2]/[6]/[8]を OFF として HPS,FPGA,MAX の JTAG チェーンをイネーブルにします

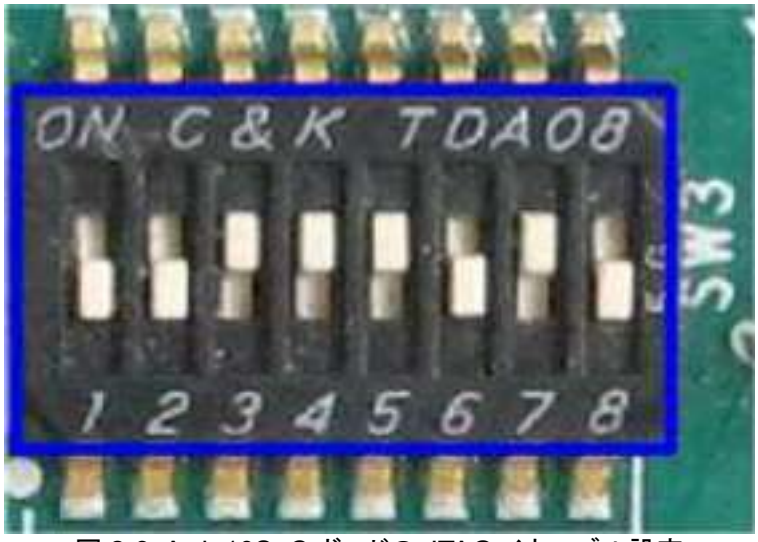

図 2-3: Arria10SoC ボードの JTAG イネーブル設定

- CycloneV SX SoC ボードの場合、プログラマブル・クロックをイネーブルにします。図 2-4 に示すように SW2 の bit[2]を ON とします

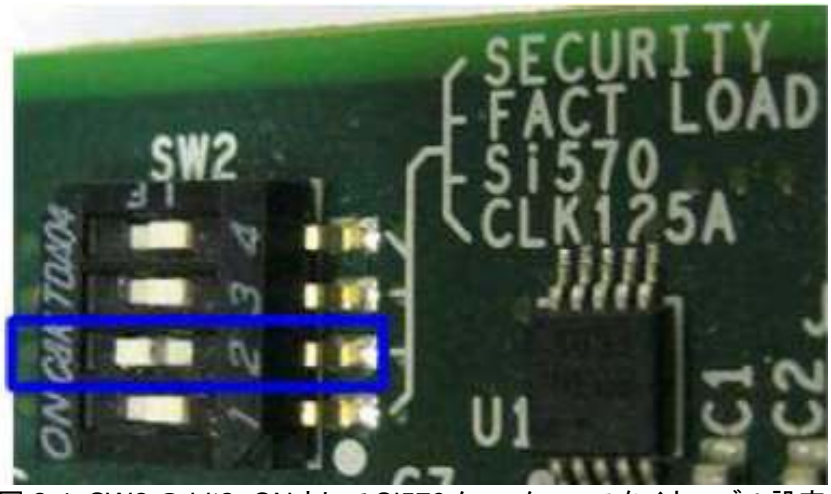

図 2-4: SW2 の bit2=ON として Si570 クロックソースをイネーブル設定

- Arria10SoC ボードの場合、HPS での FPGA コンフィグレーションをイネーブルにします。 図 2-5 に示すように SW4 の bit[2]-[4]を OFF とします

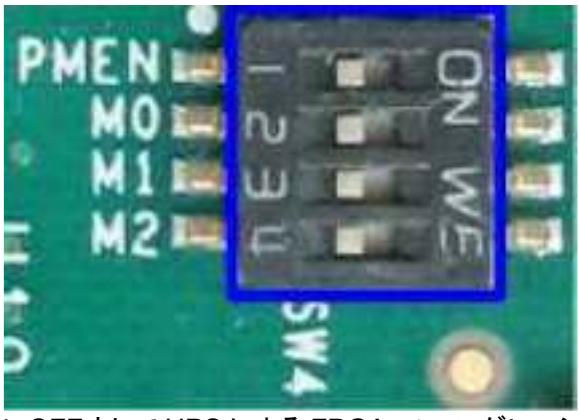

図 2-5: SW4 の bit2-4=OFF として HPS による FPGA コンフィグレーションをイネーブル設定

- FPGA 評価ボードの電源を投入します
- PC 側でシリアル・コンソール・ソフトウエア（例えば HyperTerminal など）を起動します。 ターミナルのシリアル通信設定は、ボーレート=115,200、データ=8 ビット、パリティなし、ストップビット=1 とします

- PC のシリアル・コンソールで Linux がブートし図 2-6 のようにログインの要求が表示されるまで待機します

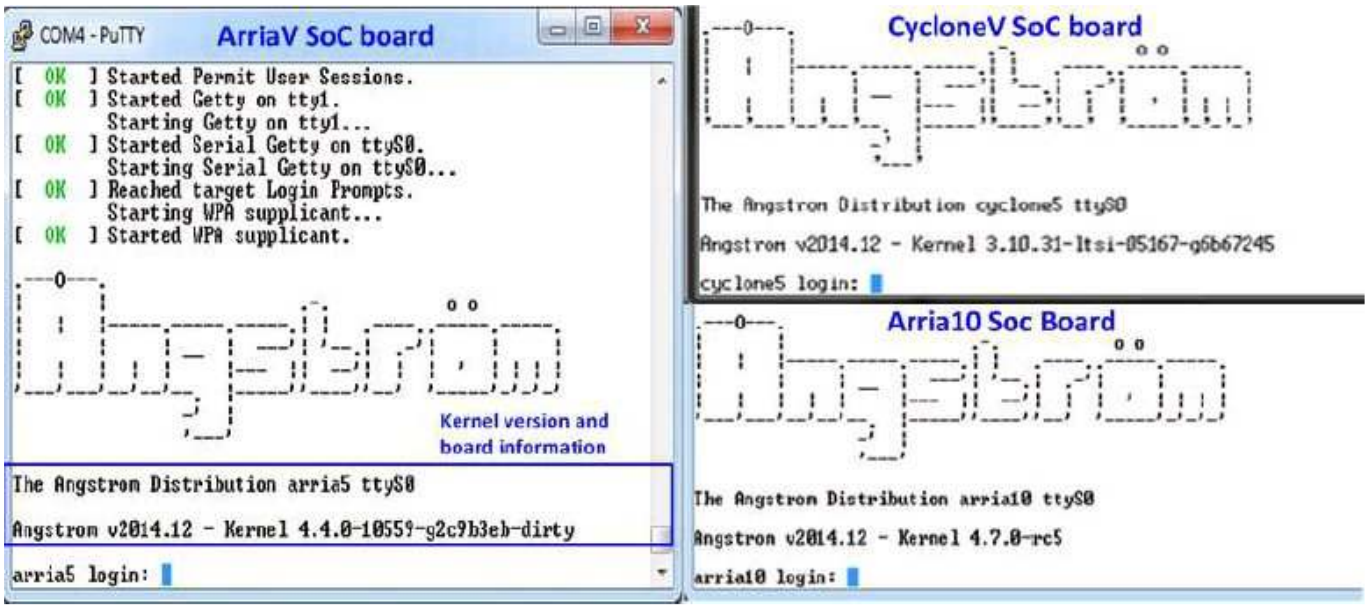

図 2-6: Linux のブート画面

- アダプタ基板の電源を投入します
  a) AB11-HSMCSATA アダプタを CycloneV SX/ボードに装着する場合、図 2-7 に示すように"Linux ブートが完了してから"AB11-HSMCSATA アダプタの電源スイッチを投入します。
  注意：Linux ブート完了前に SSD への電源を供給しないでください。SATA ドライブには Linux からプログラムする 150MHz のリファレンス・クロックが CycloneV SX ボード上で安定してから SATA 通信を開始する必要があるためです。

図 2-7:AB11-HSMCSATA 上の SATA ドライブ用電源スイッチ

b) AB09-FMCRAID アダプタを ArriaV ST/Arria10SOC ボードに装着する場合、図 2-8 のように接続した ATX 電源を投入してください。

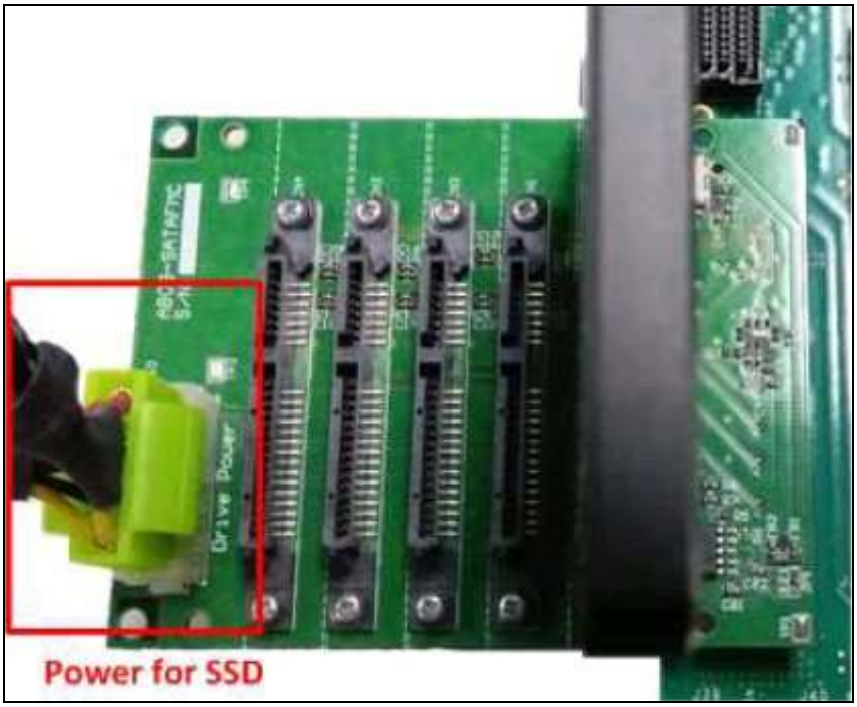

図 2-8:AB09-FMCRAID に供給する ATX 電源を投入

- FPGA 評価ボード上の LED を確認し、図 2-9 に示すように LED0-1 が点灯し LED2 は SATA 速度に応じた発光状態であることを確認します

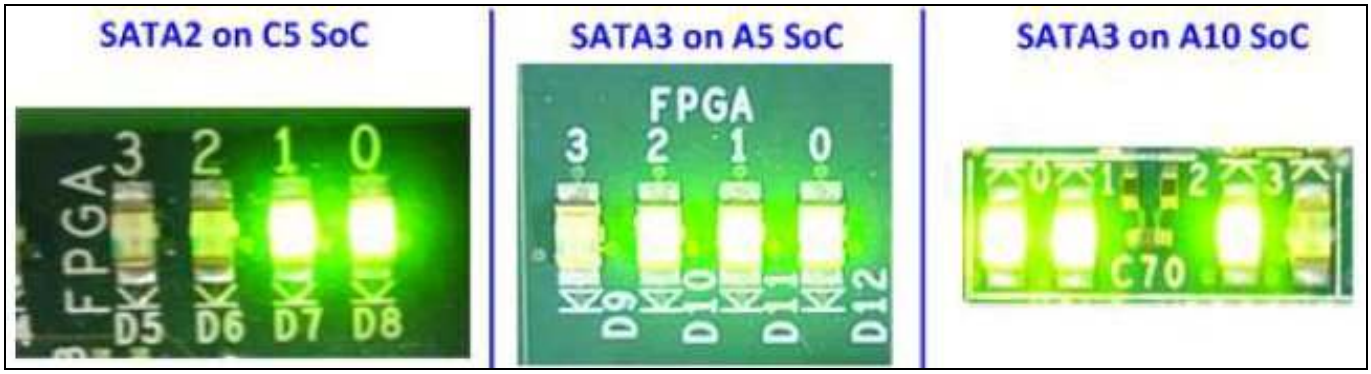

図 2-9: システム軌道後の LED 状態

LED	点灯	消灯
LEDL	OK	トランシーバ内の TXPLL がロックしません、150MHz クロックを確認してください。
LEDC	OK	SATA-IP コアが接続 SATA デバイスを認識できません。 SATA デバイスの接続や電源の供給状態を確認してください。
LEDR	SATA-III 接続	SATA-II 接続
LED0	AHCI-IP コアが処理実行中	AHCI-IP コアはアイドル(休止)中

表 2-1; AHCI リファレンス・デザインにおける LED 定義

3 Linux セットアップ

- 前ページのようにログイン画面まで進んだら以下のようにログイン名/パスワードとも"root"と入力してログインしてください。
  Login : root
  Password : root
  ログイン後、システムは図 3-1 に示すようにユーザからのコマンドが受け付け可能となります。

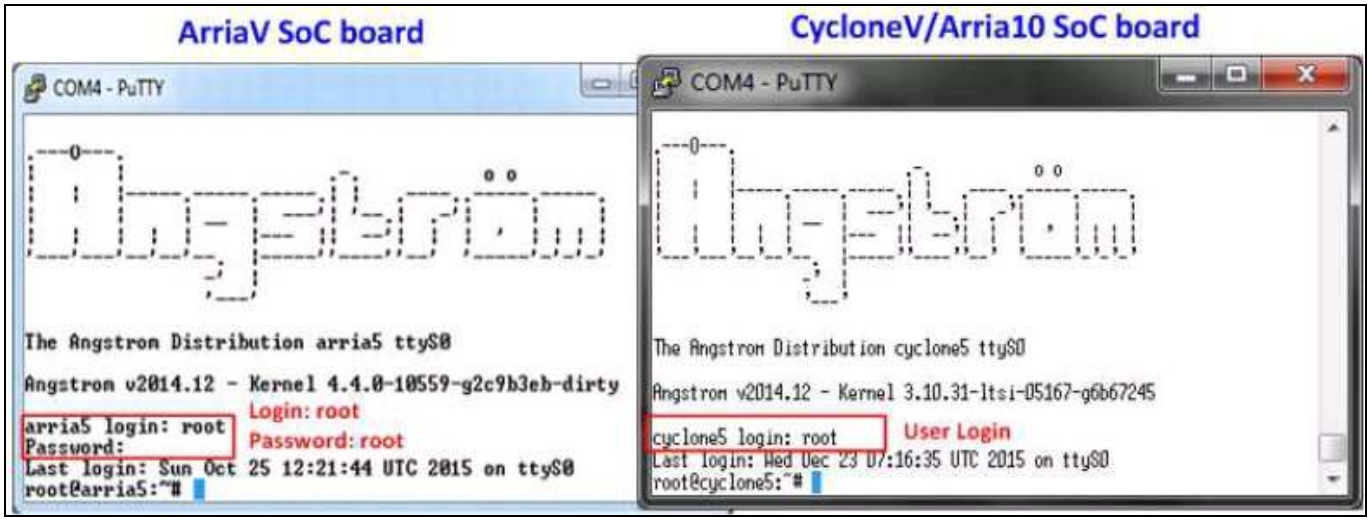

図 3-1: Linux のログイン

- AHCI ドライバを導入します
  - ✓ CycloneV SX SoC ボードにおいては libahci.ko (AHCI の共通ロウ・レベル関数)と dg_ahciDemo.ko(AHCI プラットフォーム・ドライバ)の 2 モジュールを導入する必要があります。
  - ✓ ArriaV ST/Arria10SOC ボードにおいては上記 2 ドライバに加え libahci_platform.ko も必要となるため、合計 3 ドライバの導入が必要です。
- 全てのドライバは"/home/root/driver"ホルダに格納されています。これらのモジュールを導入するため、以下のコマンド・ラインを実行してください。
  >> insmod dg_ahciDemo/driver/libahci.ko
  >> insmod dg_ahciDemo/driver/libahci_platform.ko
  (ArriaV ST/Arria10SoC の場合)
  >> insmod dg_ahciDemo/driver/dg_ahciDemo.ko
- 全てのドライバ・モジュールが導入されると図 3-2 に示すように接続ドライブの情報が表示されます。そして SATA デバイスが使用可能となります。

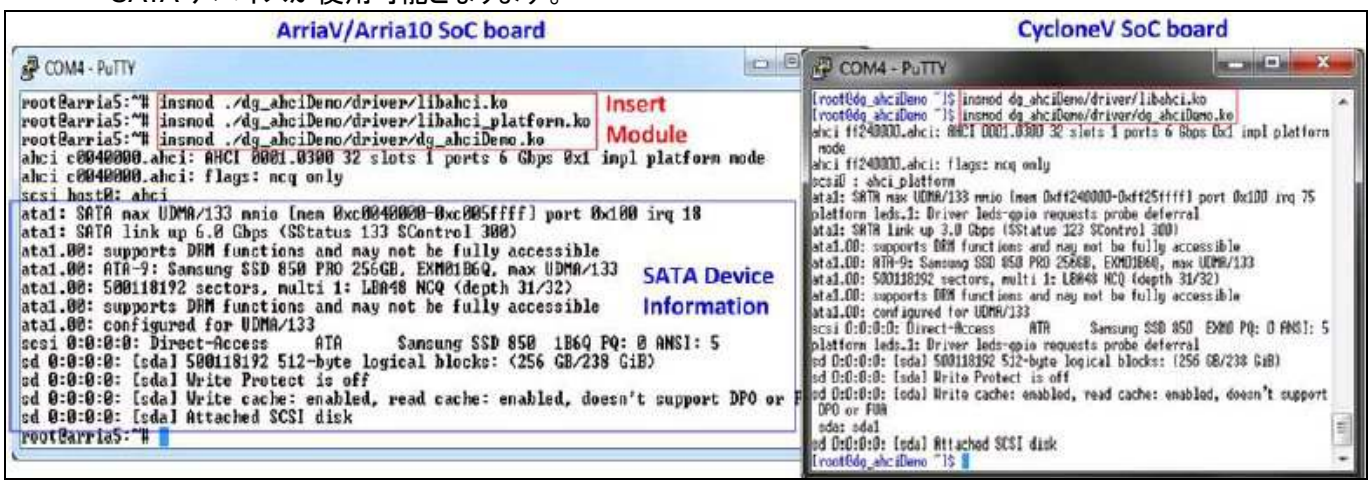

図 3-2: Linux カーネルへのモジュール導入

4 Linux コマンドの実行例

4.1 ディスク・パーティションの作成

接続ドライブに新たなパーティションを作成するには、図 4-1 に示すようなコマンド・ラインを実行します。

>> fdisk /dev/sda
→ ディスク・パーティションの管理ツール(fdisk)を起動します。

>> n
→ 新しいパーティションを作成します。

>> p
→プライマリ・パーティションを選択します。
→デフォルトのオプションを選択します、パーティション数=1、先頭シリンダ=1,シリンダ=31130 です。

>> w
→ テーブルをディスクに書き込みます。
→ この操作で"sda1"の名前で新しいパーティションがディスク上に作成されます。

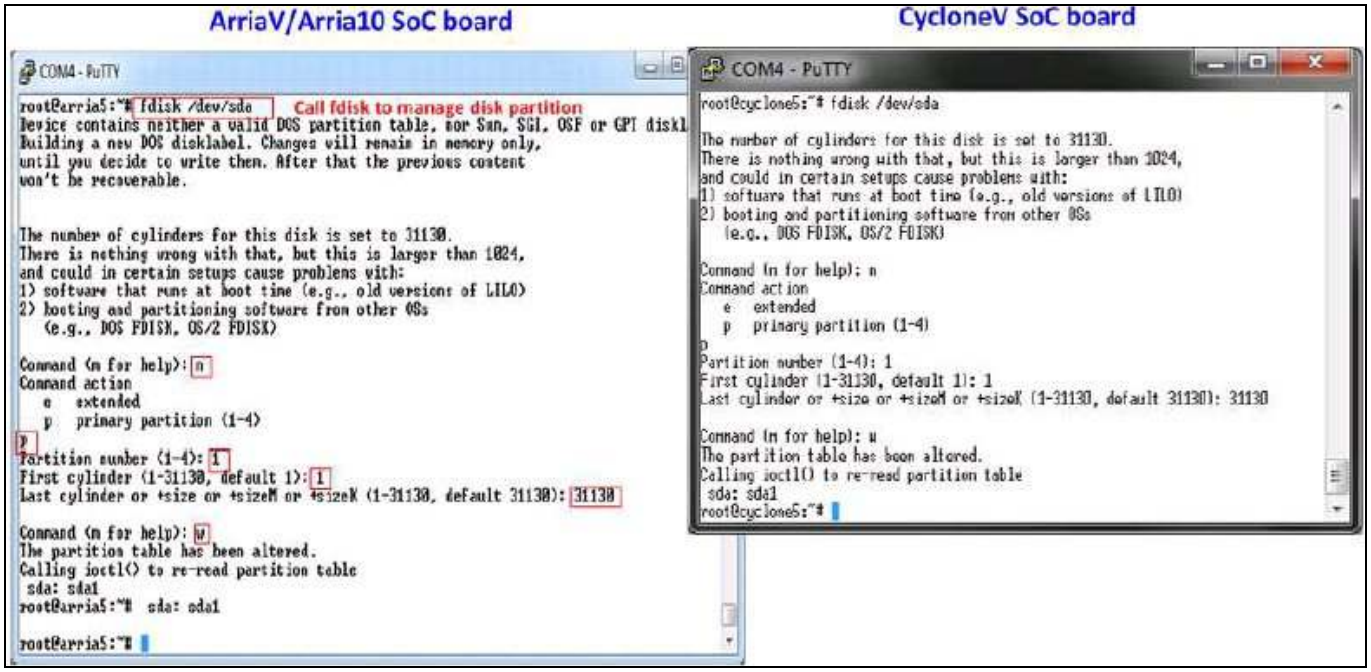

図 4-1: fdisk コマンドによるパーティションの作成例

注意: 'm'をタイプすることで fdisk コマンドにおける全てのオプションを表示できます。

4.2 ディスクのフォーマット

ディスクをフォーマットするには、ファイル・システム（例えば FAT や EXT4 等）を選択する必要があります。以下にコマンド・ラインでディスクを EXT4 でフォーマットする例を示します。

>> mkfs.ext4 /dev/sda1

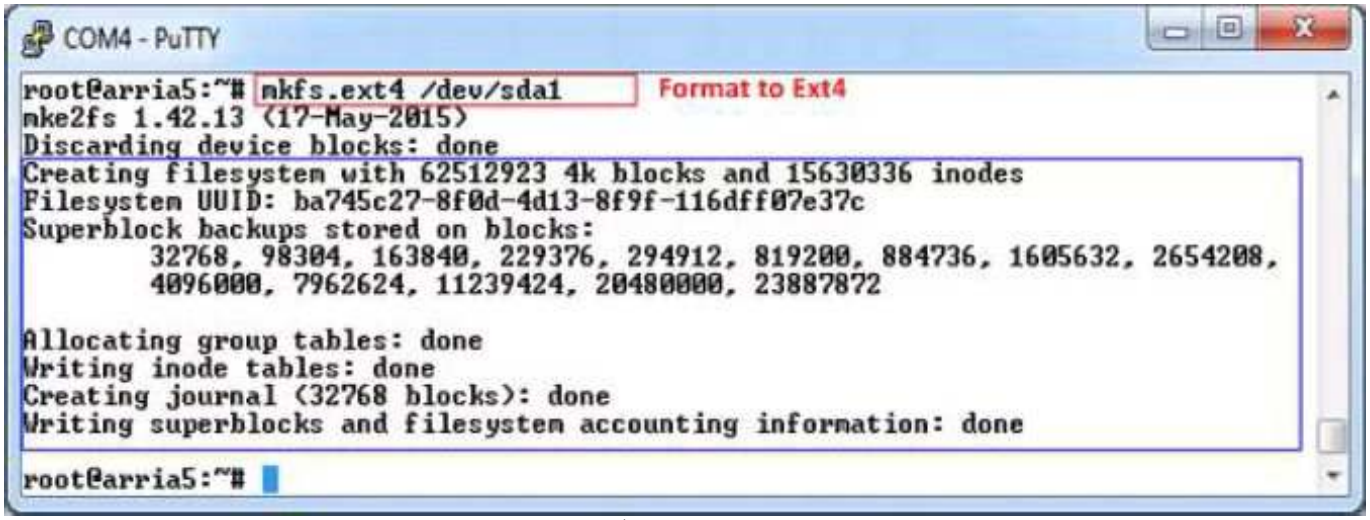

図 4-2: ディスクのフォーマット

4.3 ディスクのマウント

アプリケーション（Bonnie++や diskTestApp）からディスクをファイル・システム・モードでアクセスする場合、事前に、以下のコマンドでディスクをマウントを実行する必要があります。
>> mount /dev/sda1 /media/hdd

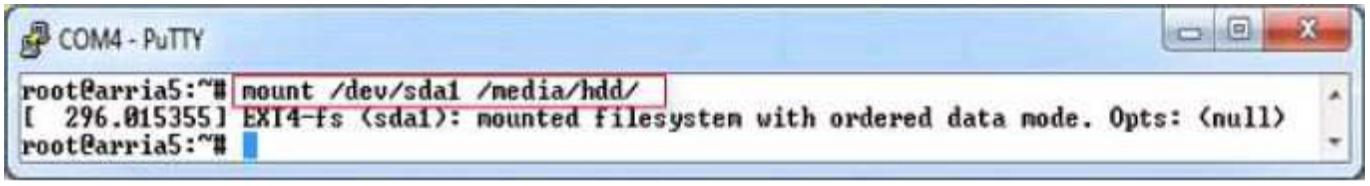

図 4-3: ディスクのマウント

5 パフォーマンス・テスト

この章ではディスクのパフォーマンスを測定するアプリケーションの例を説明します。ここでは 2 種類のテスト・アプリケーションである diskTestApp および Bonnie++について説明します。diskTestApp は DesignGateway 社で開発されたアプリケーションで、シーケンシャルにデータをライトする生データ・モードと、ファイル・システムでフォーマットされたデータ・モードの 2 モードでのライト/リード・パフォーマンスを実機確認できます。

5.1 diskTestApp によるパフォーマンス・テスト

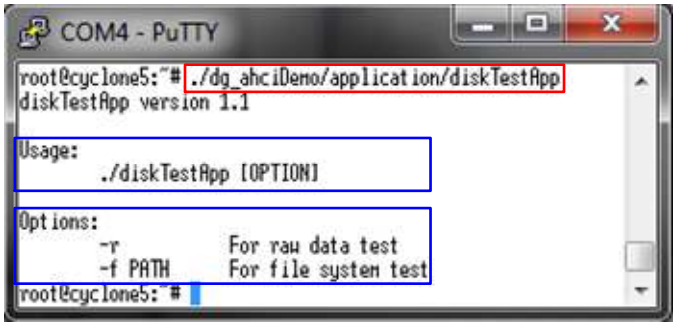

図 5-1: diskTestApp の使い方

図 5-1 に示すように、diskTestApp アプリケーションは生データあるいはファイル・システムの 2 モードで起動できます。
注意：生データ・モードでライトのテストを実行した場合、パーティション等のシステム情報は損壊されるため、ドライブのファイル・システムはアクセスできなくなりますのでご注意ください。

5.1.1 生データ・モード

生データ・モードでアプリケーションを起動するためには"./dg_ahciDemo/application/diskTestApp -r"をタイプしてください。起動時に以下 5 種類のパラメータ入力を求められます。

1) テスト対象のディスク選択
2) テスト種類: '0'：リード・テスト, '1'：ライト・テスト
3) データ・パターン
   '0'：ライトの場合ダミー・データ、リードの場合ベリファイなし
   '1'：32 ビット・インクリメンタル・パターンによるライトまたはリード&ベリファイ
   '2'：32 ビット・デクリメンタル・パターンによるライトまたはリード&ベリファイ
4) ディスク内のテスト開始オフセット: ライト/リードのテスト開始セクタ・アドレス。 先頭に' 0x' を入力すると 16 進数で、そうでない場合 10 進数でセクタ・アドレスを入力する
5) テスト長: ライト/リードのテスト・セクタ数(1 セクタ=512 バイト) 先頭に' 0x' を入力すると 16 進数で、そうでない場合 10 進数でセクタ数を入力する

図 5-2 および図 5-3 で生データ・モードにおけるダミー・データおよびインクリメンタル・データでのライト・テスト実行例を示します。 また図 5-4 および図 5-5 でベリファイなしおよびインクリメンタル・データでのリード&ベリファイのテスト実行例を示します。インクリメンタル/デクリメンタル・パターンの場合と比較すると、ダミー・データ(リード時はベリファイなし)を指定した場合、CPU 処理を必要とするテスト・データ生成やベリファイ動作などのオーバーヘッドがないため、より良いパフォーマンスが得られます。

ArriaV/Arria10 SoC board

```
COM4 - PuTTY
root@arria5:~# ./dg_ahciDemo/application/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 = 435.52 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): 0
```

CycloneV SoC board

```
COM4 - PuTTY
[root@dg_ahciDemo ~]$ ./dg_ahciDemo/application/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 = 233.57 MB/s

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

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

図 5-2: ダミー・データによる生データ・モードでのライト・パフォーマンス結果例

ArriaV/Arria10 SoC board

```
COM4 - PuTTY

// --- --- --- --- --- --- //
// 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): 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 = 99.36 MB/s
```

Write performance by 32 GB with 32-bit increment pattern

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

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

CycloneV SoC board

```
COM4 - PuTTY

// --- --- --- --- --- --- //
// 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): 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 = 77.64 MB/s

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

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

図 5-3:インクリメンタル・データによる生データ・モードでのライト・パフォーマンス結果例

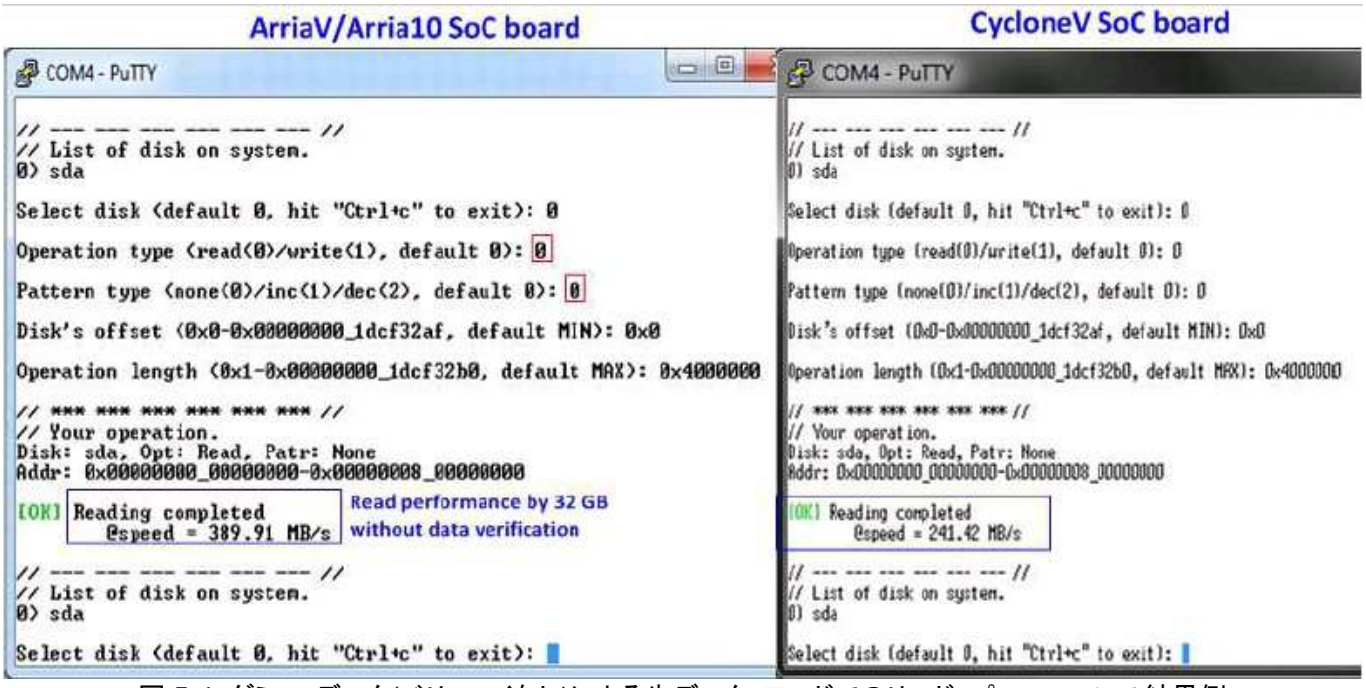

図 5-4: ダミー・データ(ベリファイなし)による生データ・モードでのリード・パフォーマンス結果例

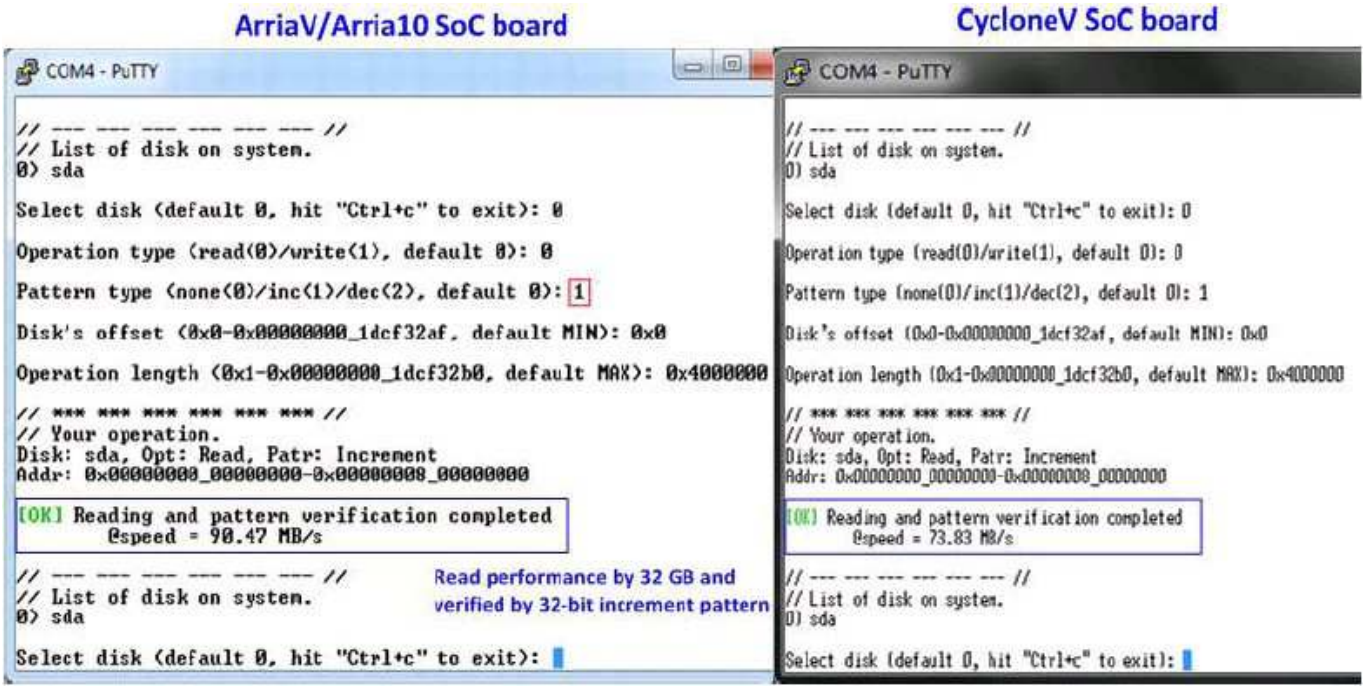

図 5-5:インクリメンタル・データによる生データ・モードでのリード&ベリファイ・パフォーマンス結果例

5.1.2 ファイル・システム・モード

ファイル・システム・モードでアプリケーションを起動する場合、それ以前にディスクにパーティションを作成、フォーマットし、マウントする必要があります。
起動コマンドとして"./dg_ahciDemo/application/diskTestApp -f"をタイプしてください。起動時に以下 5 種類のパラメータ入力を求められます。

1) ファイル名の入力: テストを実行するファイル名
2) テスト種類: '0' : リード・ファイル・テスト, '1' : ライト・ファイル・テスト
3) データ・パターン
   '0' : ライトの場合ダミー・データ、リードの場合ベリファイなし
   '1' : 32 ビット・インクリメンタル・パターンによるライトまたはリード&ベリファイ
   '2' : 32 ビット・デクリメンタル・パターンによるライトまたはリード&ベリファイ
4) ファイル数: ライト/リード・ファイル・テストを実行するファイル数
5) ファイル・サイズ: テストの 1 ファイル当たりのサイズをセクタ数 (1 セクタ=512 バイト) で指定。

生データ・モードと同様にダミー・データでのライト・テストやベリファイなしでのリード・テストでは図 5-6～図 5-9 に示す通りインクリメンタル・デクリメンタル・パターンによるテストと比較するとCPU 処理が入らないため良好なパフォーマンスが得られます。

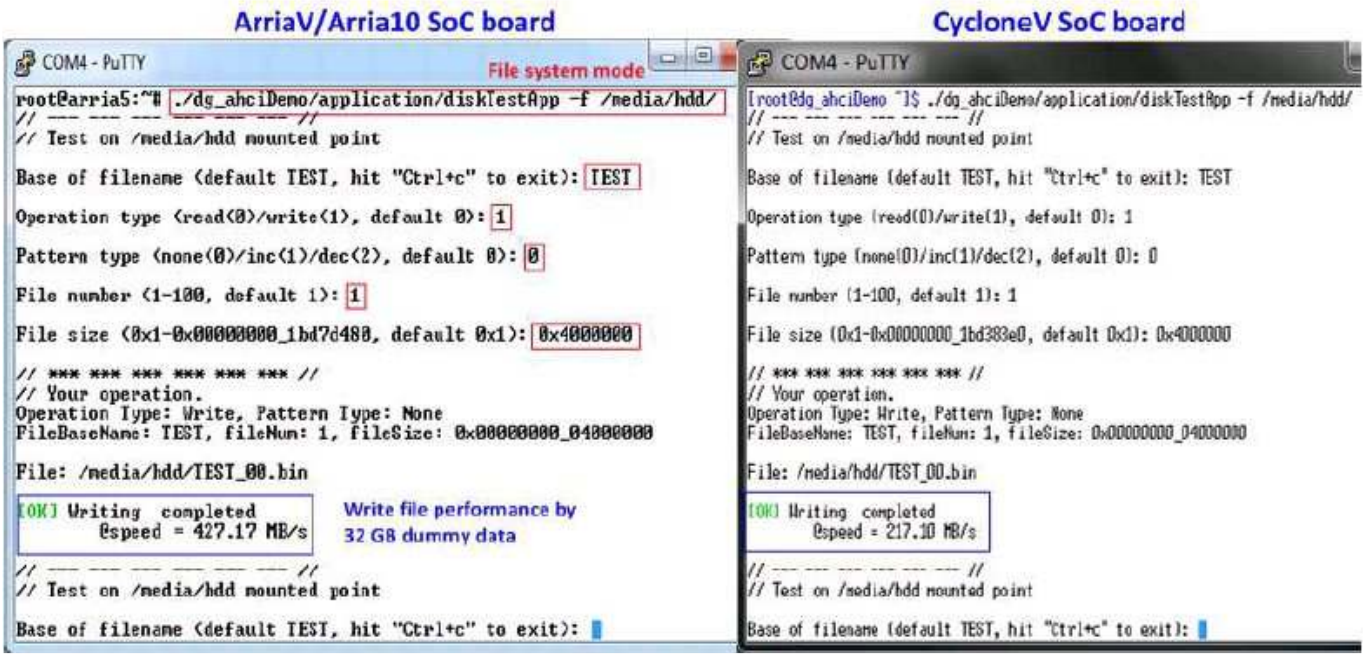

図 5-6: ダミー・データによるファイル・システム・モードでのライト・パフォーマンス結果例

ArriaV/Arria10 SoC board

```
COM4 - PuTTY
root@arria5:~# ./dg_ahciDemo/application/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): 1

File number (1-100, default 1): 1

File size (0x1-0x00000000_17d6d438, default 0x1): 0x4000000

// *** *** *** *** *** *** //
// Your operation.
Operation Type: Write, Pattern Type: Increment
FileBaseName: TEST, fileNum: 1, fileSize: 0x00000000_04000000

File: /media/hdd/TEST_00.bin

[OK] Writing verification completed
     @speed = 99.30 MB/s

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

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

Write file performance by 32 GB increment data

CycloneV SoC board

```
COM4 - PuTTY
[root@dg_ahciDemo ~]$ ./dg_ahciDemo/application/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): 1

File number (1-100, default 1): 1

File size (0x1-0x00000000_13d383d0, default 0x1): 0x4000000

// *** *** *** *** *** *** //
// Your operation.
Operation Type: Write, Pattern Type: Increment
FileBaseName: TEST, fileNum: 1, fileSize: 0x00000000_04000000

File: /media/hdd/TEST_00.bin

[OK] Writing verification completed
     @speed = 77.12 MB/s

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

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

図 5-7:インクリメンタル・データによるファイル・システム・モードでのライト・パフォーマンス結果例

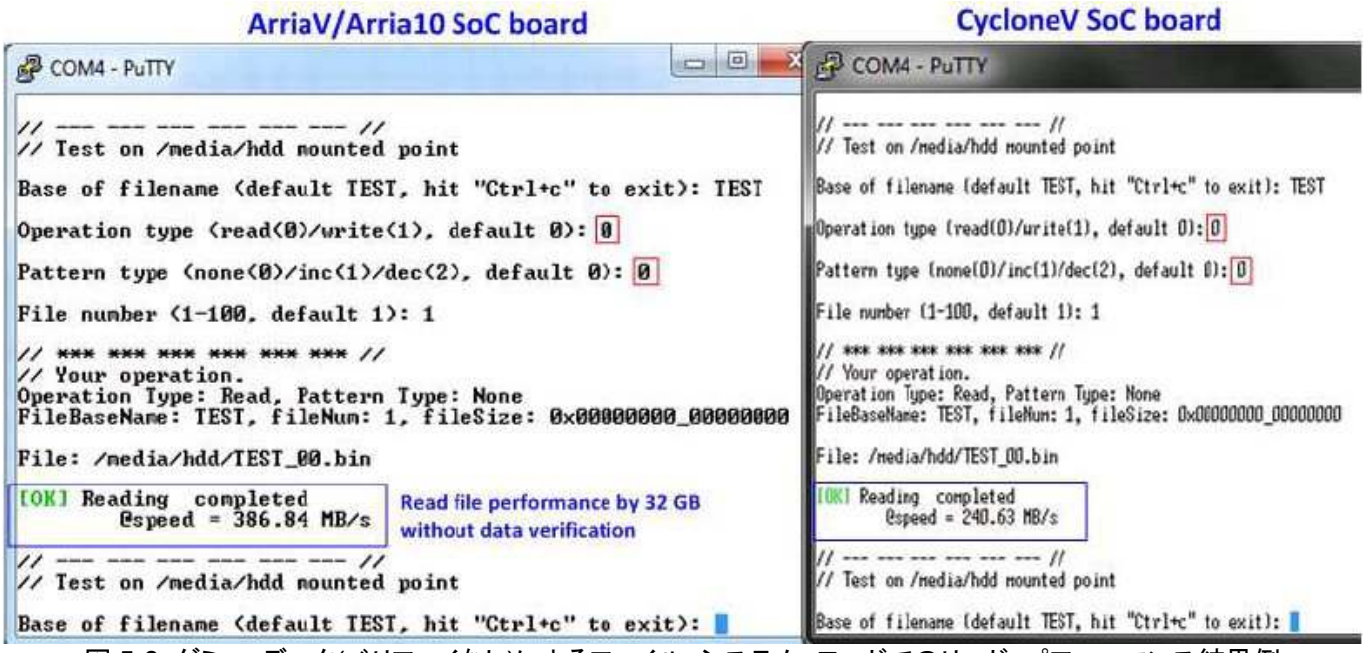

図 5-8: ダミー・データ(ベリファイなし)によるファイル・システム・モードでのリード・パフォーマンス結果例

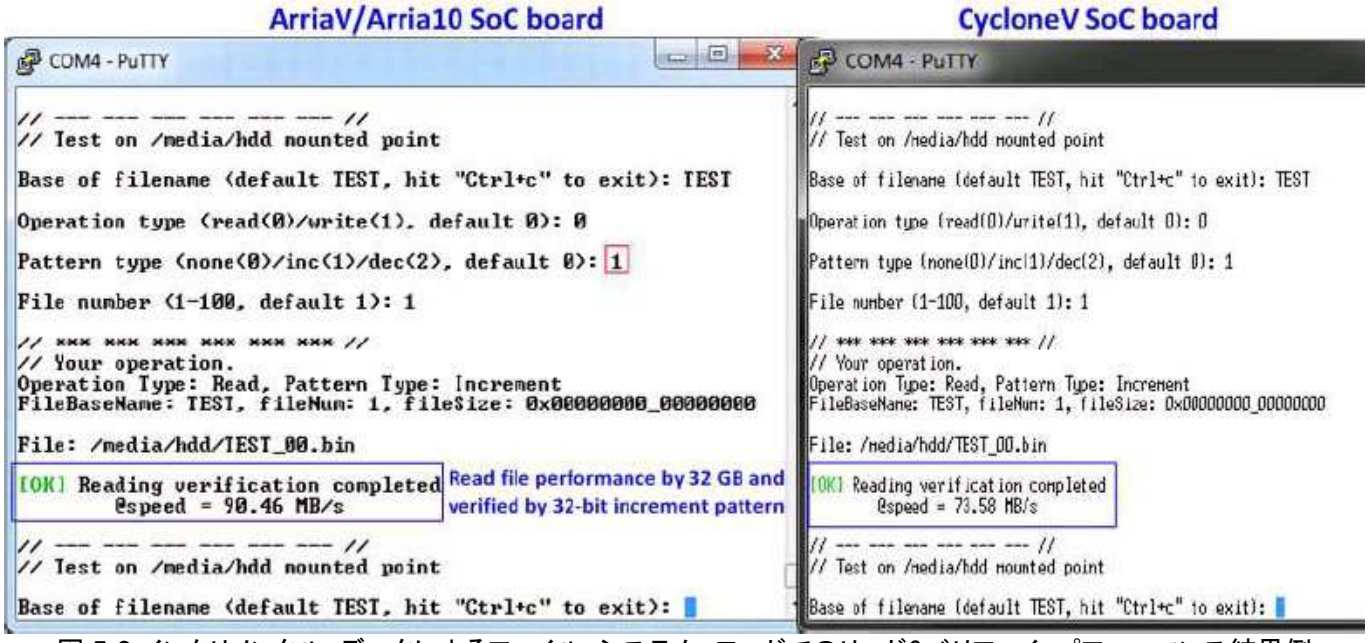

図 5-9:インクリメンタル・データによるファイル・システム・モードでのリード&ベリファイ・パフォーマンス結果例

6 Bonnie++ベンチマークによるパフォーマンス・テスト例

このトピックでは Bonnie++ベンチマークによるディスクのパフォーマンス評価の具体的な手順について説明します。Bonnie++ベンチマークのオプション・スイッチは以下となります。

```
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]
```

コマンド・ラインのオプション詳細については以下の URL を参照してください。
http://linux.die.net/man/8/bonnie++.

コマンド・ラインによるベンチマーク実行の一例を以下に示します。実行結果の例を図 5-10 に示します。

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

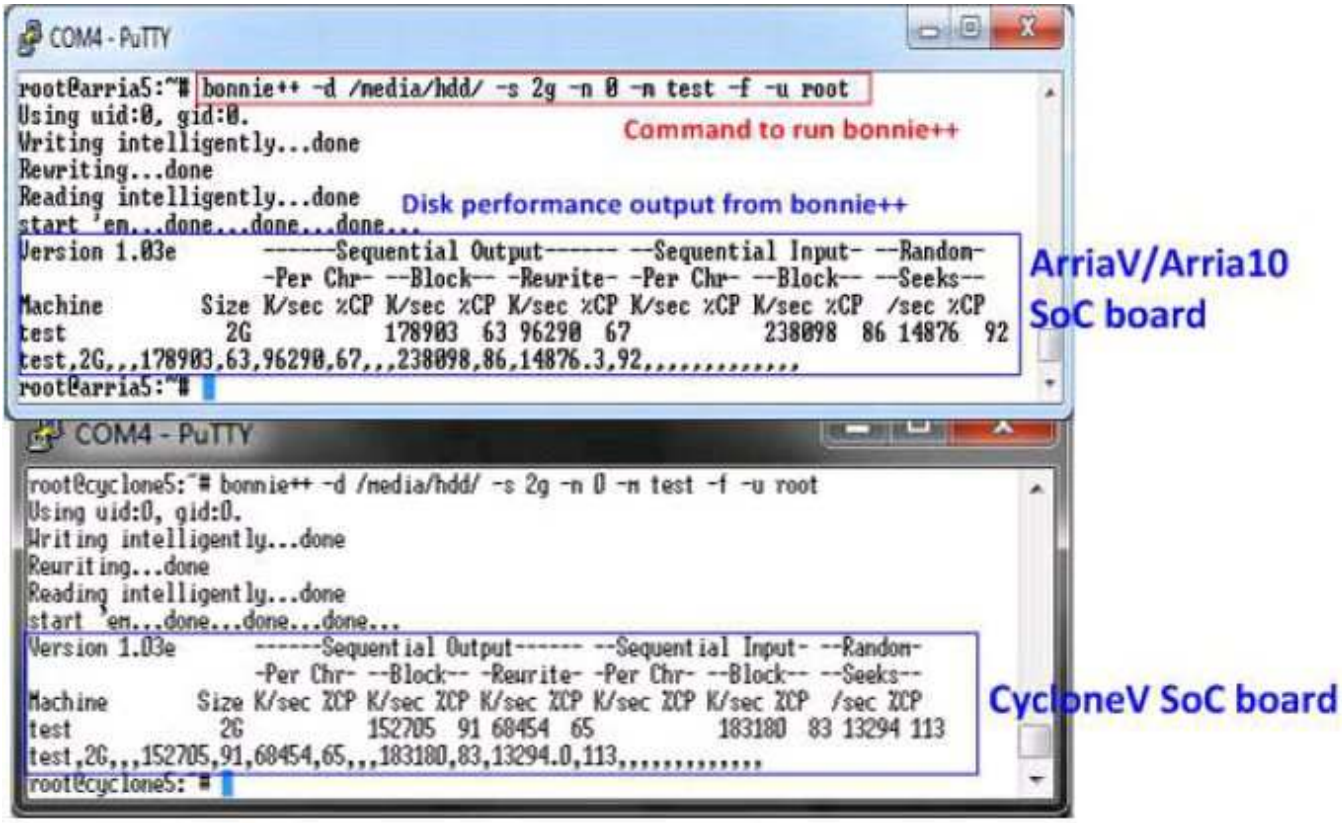

図 5-10: Bonnie++ ベンチマークの実行例

7　改版履歴

リビジョン	日付	説明
1.0	10-Novt-14	Initial version release (English version)
1.0J	2015/01/05	日本語版の初期バージョン作成
1.2J	2017/03/07	ArriaV SoC/Arria10SoC ボードのサポートに伴う更新

© 2014 Design Gateway Co,.Ltd.