DirectFBで画面にペンギンが躍る!

第6章 I/Oアクセス方法やDMA転送ドライバ，各種ベンチマークの実行まで

Linuxデバドラ&アプリケーションの作成と性能評価

石原 ひでみ Hidemi Ishihara

ここでは，完成したハードウェア上で動作するアプリケーションの作成方法について解説します．ARMコアのメモリ空間上にマップしたI/Oにアクセスするには，Userspace I/Oとmmapという二つの方法があります．またデータ転送にはDMA転送を使いますが，ARMコアにはキャッシュがあるのでキャッシュ・コヒーレンシを考慮した転送が必要になります．最後にいくつかのベンチマークを走らせ，性能を評価してみます．

第4章までで，カスタムLinuxディストリビューションまでが整いました．次はIPコア・モジュールを制御するためのLinuxドライバとアプリケーションが必要です．ここではFPGAに実装したVGA表示回路を制御するLinuxドライバとアプリケーションを作成していきます．本章で作成するドライバとアプリケーションは次の二つになります．

- IPコア・モジュールにアクセスできるアプリケーション
- フレーム・バッファの確保とDMAを実行するLinuxドライバ

1. IPコア・モジュールにアクセスするアプリケーション

● I/Oアクセス方法3種

第3章でIPコア・モジュールを実装した際，各モジュールへ**表1**のようにアドレスを割り振りました．Linux上からこれらのアドレスにアクセスするには，次の三つの方法があります．

- FPGAモジュール専用のドライバを作成する
- Userspace I/Oを使用する
- mmapでレジスタ空間を開いて使用する

ここでは，実装したIPコア・モジュールにアクセスするアプリケーションとして，簡単に行える後者二つの方法を紹介します．

● Userspace I/O

Linux上にはUserspace I/Oというドライバが用意されています．このドライバは汎用のデバイス・ドライバになります．Device Treeで**リスト1**のように設定を行うと，一つのデバイスとして登録できます．Userspace I/Oを使用する利点は，割り込み信号を取得できる点です．

登録されているデバイスをopenし，read関数を実行すると割り込みを検出するまで待つことができます．read関数から返ってくると，変数countにはread関数から返ってくる検出した割り込み回数が入ってきます．

クリティカルな割り込みを使用しないケース，例えばGPIOでボタン押下の検出を割り込みで取得したいなどの場合，このUserspace I/Oを使用すると非常に便利です．

このデバイス・ファイルをオープンして，read関数で呼び出すと割り込み待ちを行えます．readには4バイト(Int型)のポインタを渡しておけば，read関数から戻ってきたときに，発生した割り込み回数が返ってきます．

Device Treeに記述したregの領域をIPコア・モジュールのレジスタとしてmmapで開けます．mmapで開ければ，IPコア・モジュールのレジスタにアクセスして値を取得することができます．

後述するmmapを使用する例では，/dev/memのメモリ領域をmmapしているので，mmapコマンドでオフセットを付加しています．Userspace I/Oの場合は，reg宣言のアドレスがオフセットになります．

Linuxカーネルが4.0になる前あたりで，Userspace I/Oのcompatible名が引数で与えられるように変わり

表1 IPコア・モジュールのアドレス

FPGAモジュール	DE1-SoC	ZYBO
VGAモジュール	0xFF20_0000	0x4000_0000
GPIOモジュール(スイッチ入力)	0xFF21_0000	0x4001_0000
GPIOモジュール(LED出力)	0xFF21_0010	0x4002_0000

リスト1 Device TreeのI/O設定例

```
uio@40010000 {
  compatible = "generic-uio";
  reg = <0x40010000 0x10000>;
  interrupt-parent = <0x3>;
  interrupts = <0x0 0x1d 0x4>;
};
```