定番ZYBOボードによるハードウェア制御…その③

オリジナル回路用 デバイス・ドライバの動作メカニズム

鳥海 佳孝

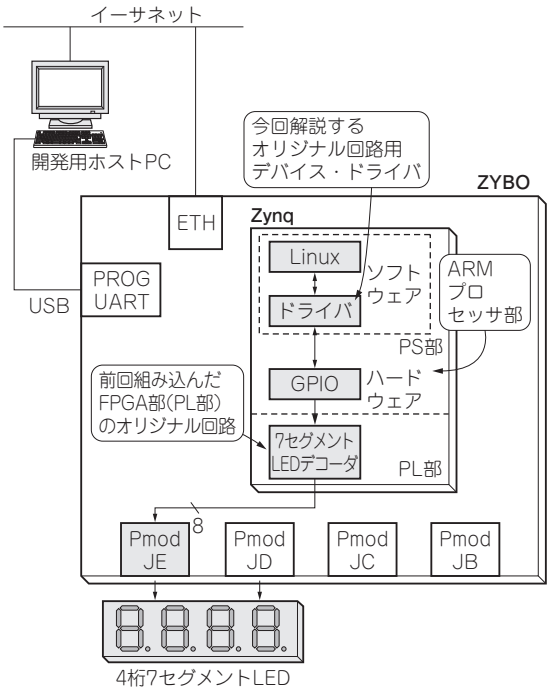

図1　今回解説すること…FPGA部オリジナル回路を動かすためのデバイス・ドライバ（GPIO）の動作メカニズム

このコーナでは，ARMプロセッサとFPGA（Field Programmable Gate Array）が1チップになったザイリンクスのZynqと，アルテラのSoC（Cyclone SoCやStratix SoC）を対象に，うまく使う方法やさまざまな話題を取り上げていきます．

2016年4月号と5月号の本コーナでは，FPGA部を使ったオリジナル回路の制御に，ダウンロードしたGPIOのデバイス・ドライバを使用していました（**図1**）．今回は，Linuxからオリジナル回路を動かすためのデバイス・ドライバの動作メカニズムについて説明します．Linuxデバイス・ドライバの全てを一から解説していては膨大な誌面が必要になるので，ここではまず本コーナで使用した基本的なGPIOのデバイス・ドライバに注目します．（編集部）

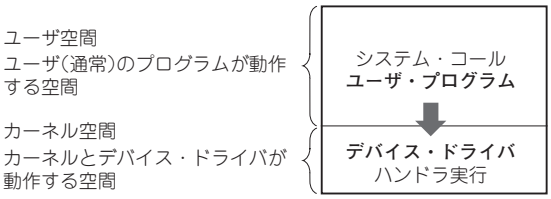

図2　デバイス・ドライバによるI/Oアクセス
ユーザ・プログラムによって実行されるシステム・コールに従ってカーネル空間で動作するデバイス・ドライバのハンドラが動作する

動作の仕組み

LinuxのようなOSにおいては，アドレス・マッピングされたI/Oを直接アクセスすることができないような仕掛けになっています．マイコン・システムとの，大きな違いの一つです．

● I/Oアクセスはデバイス・ドライバが基本

Linuxシステムにおける I/Oアクセスは，次のような方法が考えられます．

- mmapシステム・コールを用いて/dev/memを使ってI/Oアクセス
- デバイス・ドライバ使用してI/Oアクセス（**図2**）

mmapシステム・コールを用いた場合には，

- rootでないと実行できない
- 割り込みが使えない

などの問題があります．従って，LinuxでI/Oアクセスを行うときには，デバイス・ドライバを用いるのが王道です．

● デバイス・ドライバの呼び出し

Linuxシステムの場合，メモリ空間がユーザ空間とカーネル空間に分かれています．デバイス・ドライバはカーネル空間の中で動作しています．ユーザ空間で動作するプログラムがI/Oアクセスを行う場合は，実行されたシステム・コールに従って，デバイス・ドライバが動作します．この際，カーネルが該当するデバ

ARM FPGAザイリンクスZynqのLinux開発環境（2016年2月号）
Cでも書けるARM FPGAの回路設計環境（2016年3月号）
ARM FPGA用Linuxにデバドラを組み込む（2016年4月号）