ステップ・バイ・ステップでメカニズムを確認！

シェルからのハード操作超入門

中村 和敬

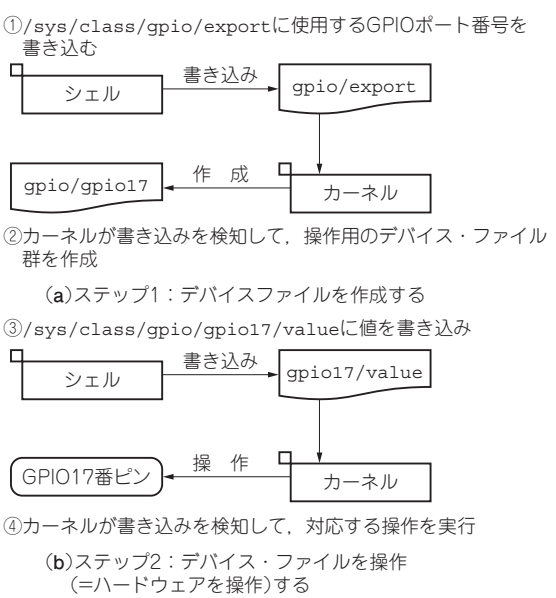

図1　デバイス・ファイルを通じたGPIOポート制御の仕組み

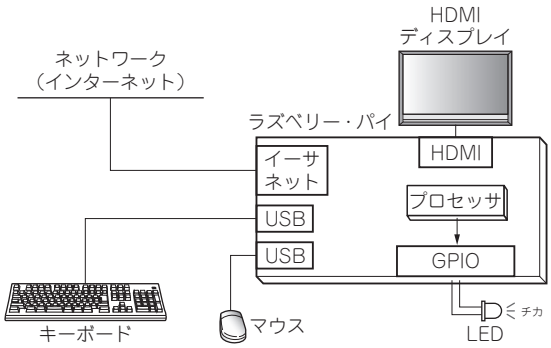

図2　シェルを使ったハードウェア操作のメカニズムをステップ・バイ・ステップで確かめてみる

汎用OSボード・コンピュータの特徴

●マイコンと比べてソフト開発の比率が高い

Linuxが動作する代表的なボード・コンピュータとして，ラズベリー・パイがあります．GPIOやI²Cといった組み込み向けのインターフェースから，イーサネットやUSBといったPC向けのインターフェースまでそろっていて，ちょっとしたガジェットを作る際のベースとして，定番になった感があります．

ラズベリー・パイのようなLinuxベースのボード・コンピュータを利用する場合，マイコンを用いた機器開発と比べてソフトウェア開発の比率が格段に高まります．

● ムダにたくさん書く必要はない…ハードの操作はむしろ簡単

OSが載っているためにできることが制限されたように思うことがあるかもしれません．

しかし，ラズベリー・パイの標準のOS，Raspbianは，ちょっと気の利いた小物を開発する際に便利なさまざまな機能を提供しています．シェルを通じてそれらを組み合わせることで，初心者でも簡単にさまざまな機能を持った機器を作ることができます．

また，Raspbian上からも，簡単な手順でGPIOなどの低レベル・ポートを直接操作することができます．こういった操作はシェルを通じて行うことができるので，対話的に周辺機器を操作して動作を確認しながら開発を進めることができます．

つまりシェルの使い方を学ぶことで，一からプログラムを書かなければならない場合と比べてはるかに簡単に開発を進めることができます．

UNIXハード制御の基本メカニズム

●ファイルの読み書きでデバイスを制御できる

UNIXでは仮想ファイルにデータを書き込むことによって，さまざまなデバイスの制御を行うことができます．UNIXは仮想ファイルへの書き込みをデバイスの操作と解釈し，内部でファイルに対応するデバイス・ドライバを呼び出し，対応する操作を行います(**図1**)．今回はLEDの点滅ということで，出力の例を示します．入力についても同じようにファイルを読み