第2章 なるほどそうやって動くのか！ リセット直後から丸はだか！

知ってたらハナタカ！

完全理解！ラズベリー・パイの32/64ビットLinux起動シーケンス

原山 みや

表1 ラズベリー・パイ・シリーズのCPUコア

名 称	プロセッサ	コア名	コア数	コアのクロック周波数
ラズベリー・パイ	BCM2835	ARM1176JZF-S	1	700MHz
ラズベリー・パイ2	BCM2836	Cortex-A7	4	900MHz
ラズベリー・パイ3	BCM2837	Cortex-A53	4	1.2GHz

ラズベリー・パイ3では，CPUコアが64ビット対応ARM Cortex-A53になりました(**表1**)．当然，搭載されるLinuxも64ビットだと思っていましたが，なんと，32ビットLinuxのままでした．

ラズベリー・パイのエコシステムを考えると，Linuxを64ビットにする必要はなく，Cortex-A53コアをクロック周波数が900MHzから1.2GHzになった高速な32ビットARMv7コアとして動作させた方がよいと判断したのでしょう．そうすることで，クロック周波数が速くなった恩恵を受けて，33%程度の性能向上が見込めるのですから．

しかし，Cortex-A53コアを64ビットで動かせれば32ビットのときよりもDhrystone値で20%程度速くなります[1]．ということは，ラズベリー・パイ3で64ビットLinuxが動けば，ラズベリー・パイ2(32ビットLinux)に対して，60%（＝1.2/0.9×1.2）も速くなります．ラズベリー・パイ3でもっと速くプログラムを動かすには，64ビットLinuxを動かせればよいのです．

32ビット/64ビットLinuxを使い分ける前に

● リセット後最初に実行されるプログラム「ブート・スタブ」とは

ラズベリー・パイ/2/3と各ブート・スタブの対応を**表2**に示します．CPUコアのアーキテクチャごとにブート・スタブがあります．

ここでいうブート・スタブとは，CPUコアのリセットが解除された後に実行される最初のプログラムのテンプレートを考えてよいと思います．テンプレートなのでこのブート・スタブにコードを追加したり，既にあるコードを修正したりすることで独自のブート部を作れるということを意味しています．

表2 リセット解除後最初に動くプログラム「ブート・スタブ」はCPUコアのアーキテクチャごとに用意されている

名称	CPUコアのアーキテクチャ	ビット幅	ソースコード名
ラズベリー・パイ	ARMv6	32ビット	armstub.S
ラズベリー・パイ2	ARMv7	32ビット	armstub7.S
ラズベリー・パイ3	ARMv8	32ビット	armstub7.S
		64ビット	armstub8.S

Cortex-A53(ARMv8)ですが，32ビットのときはラズベリー・パイ2と同じブート・スタブ(armstub7.S)を，64ビットのときは別の新しいブート・スタブ(armstub8.S)を使います．

ラズベリー・パイ・シリーズのCPUコアのブート部は，ソースコードが公開されていないブート・プログラムの一部に含まれていて，ユーザがその部分を変更することができませんでした．しかしラズベリー・パイの公式ツールの一部としてスタブ・コードが公開されたことで，ユーザが独自のブート部を作れるようになりました．また，最新のブート・プログラムでは，ラズベリー・パイ/2/3にあったデフォルトのブート部が組み込まれているようです．

● 32ビットに64ビット…使いたいモード用のブート・スタブで起動する

どうすれば，64ビットLinuxが動くかという問いに対しては，armstub8.Sを使うことで，64ビットLinuxを起動することができるようになりました．「はい，これで目的は達成できました．よかったですね？」

Linux上でアプリケーションを動作させるだけであれば，ここまでで終わりです．ブート・スタブの詳細を知る必要はありません．

しかし，組み込みソフトウェアに本格的に取り組み