デュアルCPUコアでデュアルOS!

最小限の機能でFPGAを高速起動してからLinuxを立ち上げる Zynqで試す！リアルタイムOS & Linuxのデュアルブート方法

石原 ひでみ Hidemi Ishihara

ZynqやSoC FPGAにはデュアルコア・プロセッサが内蔵されています．そのため，それぞれのCPUコアで別々のOSを走らせることができます．豊富なソフトウェア資産が使えるLinuxを使いつつ，クリティカルな部分にはリアルタイムOSで処理させることができます．ここではリアルタイムOSとしてFreeRTOSを使い，リアルタイムOSを起動させてからLinuxを起動する方法について解説します．

1 デュアルOSを採用する理由

● ARM＋FPGAシステムのOSはLinuxが多い

ZynqやSoC FPGAに内蔵されているCPUには，ARM Cortex-A9が2つ実装されています．今後登場する，Xilinx社Zynq UltraScale+やIntel社Stratix 10 SoCは，ARM Cortex-A53が4つ実装されています（Zynq UltraScale+ではさらにARM Cortex-R5も実装されている）．

これからますます，マルチコアなプロセッサが内蔵されたFPGAが登場してきます．これらのARM＋FPGAを搭載したシステムで，皆さんはどのようなOSを稼働させているでしょうか．おそらく，Linuxが一番多く使われているのではないのでしょうか？

● リアルタイムOS＋Linuxという手もある

複数CPUがあるので，クリティカルな処理はリアルタイムOSにまかせ，GUIなどはLinuxに役割分担させたいというケースも少なくないでしょう．筆者は仕事で画像＋通信処理系を主に扱うので，画像処理や通信処理のクリティカルな処理にはリアルタイムOS＋FPGAに担当させ，ユーザ・インターフェースに当たるGUIなどの部分はLinuxに担当させる構成を採用しています．

● Linuxは起動が遅い！

もう1つ実現したいことは，システムの高速起動です．Linuxが稼働しているシステムでは，電源投入から最終的なアプリケーションの起動までに，数十秒以上かかるケースが一般的です．しかし，電源を入れてから1，2秒で動き出してほしいという要求もあります．システムが稼働するのを数十秒も待っていられないケースがあるのです．

ユーザ・インターフェースは後から起動してもよいから，コアな機能部分は先に動いてほしいという場合であれば，今回紹介する方法で，先にリアルタイムOSを走らせてからLinuxを起動することができます．

● 複数のOSを走らせる方法

CPUがARMの場合，複数のOSを起動させるには，大きく分けると次の3つの方法があります．

- 仮想化技術ハイパ・バイザで実装
- CPU内蔵セキュリティ機能TrustZoneで実現
- 単に各CPUにOSを実装

前者2つは，ソフトウェア（ハイパ・バイザ）またはハードウェア（TrustZone）を用いて，主に複数のOS間のセキュリティを保全し，複数のOSを安全に稼働させるシステムを構築するときに用いられます．OS間のセキュリティとは例えば，OS間でメモリを参照できないようにして，各OSの安全性を高めることです．

また，前者2つは一般的には有償であり高い技術力も必要になってくるので，使用するのにちゅうちょする方も多いのも事実です．

● 今回は単に各CPUにOSを実装する

残る最後の，単に各CPUにOSを実装するというのが，今回実現しているリアルタイムOS＋Linux環境になります．**図1**のようにCPU0でリアルタイムOSを，CPU1でLinuxを起動する構成になっています．この構成では単に各CPUでOSを稼働させるだけなので，OS間でのメモリ保護は行っていません．**図2**のようにCPU0で動作しているOSのアプリケーションが，CPU1で動作しているOSのメモリ空間を参照することもできます．違うOSのメモリを参照できるということは，違うOSのメモリを破壊することができるということになります．

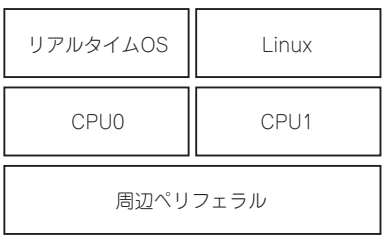

図1 CPU0でRTOSを，CPU1でLinuxを走らせる