第10章 Linuxレスのときに欠かせない! メモリ&レジスタに直接アクセスするためのMMU設定テクニック

Cortex-A9プロセッサを超高性能なワンチップ・マイコンのように使うヒント

萩本 良平

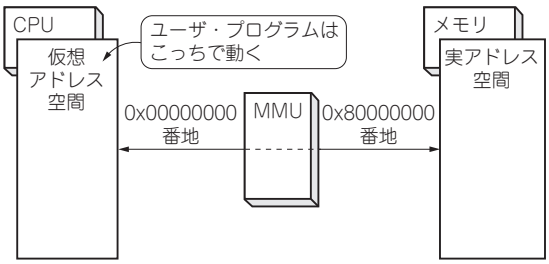

図1 プロセッサでLinuxを使わないときの問題…ユーザ・プログラムから物理アドレスに直接アクセスできないので非常に不便

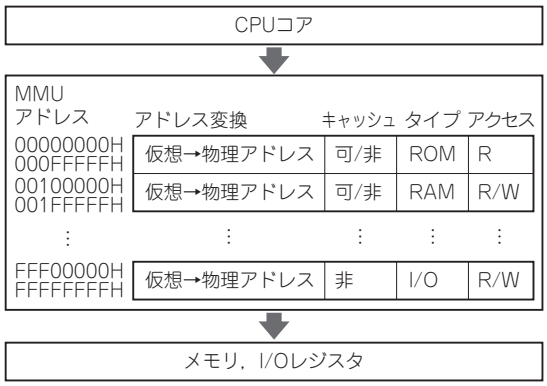

図2 メモリ管理ユニットMMUはキャッシュON/OFFやリード/ライト権限などが仮想アドレスの範囲で決まっている
Cortex-A9の例

本稿で目指すこと…仮想アドレス=物理アドレスとなるようにMMUに設定する

● プロセッサを汎用OSなしで使うときの問題…物理アドレスに直接アクセスできない

ARM Cortex-Aプロセッサを使用するにあたって，最初の壁となるのがメモリの設定だと思います．

プロセッサでは，CPU（プログラム）がメモリにアクセスするときに，仮想アドレス-物理アドレス変換器（メモリ管理ユニットMMU；Memory Management Unit）を経由します（**図1**）．

Linuxのような大規模OSを使う場合はメリットがありますが，OSなしで使ったり軽量リアルタイムOSで使ったりする場合は，物理アドレスが隠ぺいされてしまうので，希望のレジスタやメモリに直接アクセスすることができません．非常に不便です．

● MMUをOFFにすると性能がすごく（？）落ちる

仮想アドレスと物理アドレスを一致させ，直感的にわかりやすくアクセスできるようにする方法として最も簡単なのは，MMUをOFFにすることです（p.99，コラム2）．

しかし，MMUをOFFにしてしまうと，データ・キャッシュを利用することができなくなり（命令キャッシュは利用可能），Cortex-Aの実力を発揮できません[注1]．

● MMUがONのまま仮想アドレスと物理アドレスを一致させる

本稿では，MMUをONにしてキャッシュを生かした高性能処理を行えるにもかかわらず，物理アドレスにも簡単にアクセスできるように，MMUに仮想アドレスと物理アドレスを一致させる設定を行う方法を紹介します．

ターゲットはこれからの定番であるCortex-A9コア搭載プロセッサで，これをワンチップ・マイコンのように扱えるようになります．さらに詳細を知りたい場合は，文献(1)(2)などが参照できます．

注1：例えば，Cortex-A9搭載RZ/A1Hマイコン（ルネサス エレクトロニクス，400MHz動作）の内蔵SRAM上にプログラムやデータがあるときに，同一データのWMAデコード処理にかかる時間を計測したところ，性能が1/5以下になってしまいます．
L1データ・キャッシュ（32K）ON =1159.7ms
L1データ・キャッシュ（32K）OFF= 216.1ms
高速にアクセスできる内蔵RAMでこの数値です．外付けのSDRAMなどが対象の場合，さらに効果が大きくなります．