第3章

レジスタ・セット，MOVE命令，ロード/ストア命令編

Arm & RISC-V アセンブリ言語リファレンス

中森 章

1 レジスタ・セット

まずは，データの保持や演算を行うレジスタの仕様を知っておくことが必要です．少なくとも使えるレジスタの本数を知らないと存在しないレジスタを指定する恐れがあります．

Arm32（AArch32）の場合

図1[(1)]にAArch32（Armv7-A）のレジスタ・セットを示します．通常のプログラミングにおいては，「アプリケーション・レベル・ビュー」のレジスタを見ておけば十分です．R0からR15までの16本のレジスタがあります．このうち，R15はPC（プログラム・カウンタ）です．R15にアドレス値を書き込むと，そのアドレスにジャンプできます．また，ロード命令のアドレシングでベース・レジスタに使用すればPC相対アドレシングを行うことができます．

R15以外は基本的に汎用のレジスタです．「基本的に」の意味は，R14はサブルーチン・コールで戻りアドレスが自動的に設定されたり，R13はスタック・ポインタとして用途が限定されたりしている（別にそれを守らないといけない訳ではありませんが）ので，CPUの都合で値が変化する可能性があるということです．APSRはCPUの動作状態を決定するステータス・レジスタです．APSRへのアクセスは特殊な命令（MSR/MRS）を使って行います．割り込み処理とかを扱わない場合は無視して構いません．

Arm64（AArch64）の場合

図2[(2)]にAArch64（Armv8-A）のレジスタ・セットを示します．R0からR30までの31本の汎用レジスタが存在し，それに加えて，SP（スタック・ポインタ），ZR（ゼロ・レジスタ），PC（プログラム・カウンタ）が存在します．AArch32とは異なり，PCにプログラムで直接アクセスはできません．分岐命令の分岐先が書き込まれ，そこから命令フェッチが開始されるレジスタがCPU内部に存在するという以上の意味はありません．

SPとZRは，命令のエンコード的にはR31に相当します．それが使用される場面に応じて，SPとして機能したり，ZRとして機能したりする不思議なレジスタです．SPも，普通のMOVE命令や演算のソース/デスティネーションに指定できるので，汎用レジスタとみなすこともできます．つまり，AArch64の汎用レジスタの本数は32本であるといっても差し支えありません．

なお，Rn（n=0，…，30）という名称は仮想的なものです．アセンブリ言語の記述では，XnまたはWnを使用します．Xnは64ビット・レジスタ，Wnは32ビット・レジスタを示します．物理的にはXnの下位32ビットがWnとなります．また，SPやZRは，64ビット長と32ビット長で，それぞれ，XSP，WSP，XZR，WZRと記述します．また，サブルーチン・コールで戻りアドレスが格納されるレジスタはX30/W30です．

ところで，Armv8-Aでは，AArch64の他にも，AArch32をサポートします．この場合，AArch64のW0からW14がAArch32のR0からR14としてみなされます．それ以外のAArch64の汎用レジスタからAArch32の汎用レジスタの割り当ては**図3**[(3)]のよう