第1章 実行コンテキストの退避/切り替え/復元

プログラムを切り替えるディスパッチャ

豊山 祐一

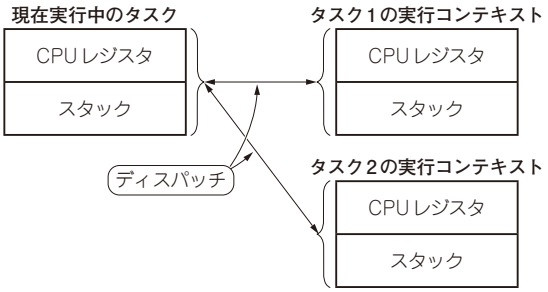

図1 ディスパッチャは実行コンテキストを切り替える(ディスパッチする)

第3部ではいよいよTry Kernelを作成していきます．まずRTOSの基本であるマルチタスクの機能を実現するため，第2部で作成したプログラムに，ディスパッチャ，スケジューラ，システム・タイマを順番に追加していきます．本章で作成するプログラムのファイルは，特に指定のない限りはkernelディレクトリに置きます．

プログラムを並行に実行する方法

● RTOSがディスパッチで複数のプログラムを切り替える

マルチタスクとは複数のプログラムを並行に実行する機能です．C言語のプログラムであれば，複数の関数が同時に実行されることをイメージすればよいでしょう．

ただし，実際にはマイコンのCPUコアは一時に1つのプログラムしか実行できませんので，RTOSが実行しているプログラムを切り替えることによって複数のプログラムを並行に実行します．この実行プログラムの切り替えをディスパッチと言います．そしてディスパッチを行うRTOSの内部のプログラムがディスパッチャです．

表1 Arm Cortex-M0+のCPUレジスタ

レジスタ名	機　能	内　容
R0～R12	汎用レジスタ	プログラム実行中の各種データ
R13 (SP)	スタック・ポインタ	スタックの現在の操作位置のアドレス
R14 (LR)	リンク・レジスタ	分岐先から戻るアドレス
R15 (PC)	プログラム・カウンタ	プログラムの実行アドレス
xPSR	ステータス・レジスタ	CPUの各種情報

● まずは簡易ディスパッチャを作る

本章では実際にC言語の関数を切り替えながら並行実行する簡易的なディスパッチャを作成して，その仕組みを説明します．本章で作成するディスパッチャは簡易ディスパッチャと呼ぶこととします．簡易ディスパッチャはC言語の関数をタスクとして並行実行します．

実行中のプログラムを切り替える具体的な方法

● ディスパッチャはコンテキストを切り替える

実行中のプログラムはその実行環境に関するさまざまな情報(現在実行中のプログラムの場所や，計算中の変数など)を持っています．これをプログラムの実行コンテキストと呼びます．タスクの場合はタスク・コンテキストとも呼ばれます．

ディスパッチャは，実行コンテキストを変更して実行中のプログラムを切り替えます(**図1**)．

実行コンテキストの実体は，主にCPUレジスタのデータとスタックです．ラズベリー・パイPico(以降，Pico)のCPUコアであるCortex-M0+は**表1**に示すレジスタを持っています．

PC(プログラム・カウンタ)レジスタは実行中のプログラムのアドレスを格納しています．PCレジスタの値を変更すれば，実行中のプログラムを変更できます．ただし，PCレジスタを変更しただけではプログラムは正常に切り替わりません．CPUの状態などは他のレジスタに保持されていますし，計算中のデータなどは汎用レジスタにあるからです．