形式手法による組み込みソフトウェア開発

第9回 リアルタイムOSを使ったときのプログラムの構造

藤倉 俊幸

OSなしの場合とリアルタイムOSを使った場合とではプログラムの構造が変わってきます．本稿では，リアルタイムOSを使うときのタスクの割り当てやタスク間インターフェース，グローバル変数の排他制御などについて解説します．　（編集部）

表1　本連載で設計してきたカップラーメン・タイマの要求仕様

ID	内　容
Spec01	電源がONの間（プログラムが動いている間），LED1の点灯・消灯を1秒間隔で繰り返す
Spec02	SW1がONになると設定時間を30秒にして，タイマを起動する
Spec04	タイマが動いている間にSW8がONになると，設定時間を30秒延長する
Spec05	タイマが動いている間は，10秒間隔で，LED4を2回点滅させる．LED4の2回点滅は，点灯・消灯を0.25秒間隔で2回繰り返すことで行う
Spec06	設定時間が経過すると，LED4を15秒間点滅させた後，消灯する．LED4の点滅は，点灯・消灯を0.25秒間隔で繰り返すことで行う
Spec07	SW8が押されたらLED2を点灯する

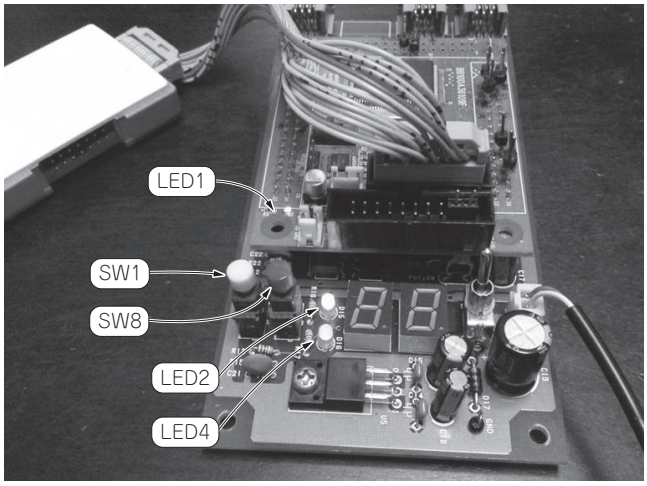

写真1　リアルタイムOSを使ったカップラーメン・タイマの実装を試したH8マイコン基板

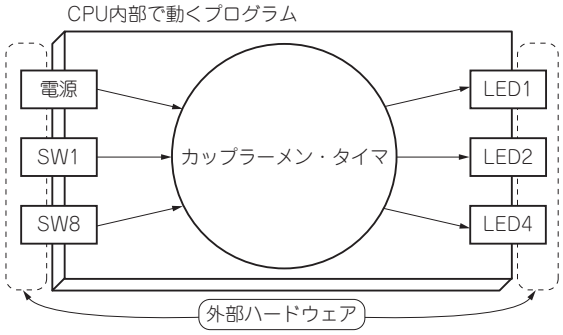

図1　カップラーメン・タイマのシステム構成

本連載では，同一の要求仕様を満たすカップラーメン・タイマなどのプログラムを，さまざまなタイプの環境に実装しながら，形式手法による開発の特徴を示してきました．

カップラーメン・タイマの要求仕様は単純で，リアルタイムOS（以下RTOS）を使用しなくても実装できます．実際今までOSなしで動作させてきました．しかし，ボタンを押されるなどの外部環境の変化に直ちに反応するイベント・ドリブンなプログラムを作成するには，RTOSが必要になります．

そこで本稿では，カップラーメン・タイマを例に，RTOSを使った際の検証の仕方を説明したいと思います．

今回使用するのは，H8マイコンとH8マイコンに対応したSmalightOS(1)という軽量なITRON-APIを持ったRTOSの組み合わせです．

今回は移植方法を，次回は検証方法を説明します．

RTOS用のプログラム構造に直す

● 今回のシステム構成

本連載で設計してきたカップラーメン・タイマの要求仕様を**表1**に，システム構成を**図1**に示します．

使用するH8マイコン・ボードを**写真1**に示します．今回のカップラーメン・タイマを実装するには，ボタンまたは