第3章　アラーム・ハンドラ，周期ハンドラ，排他制御，セマフォ，ミューテックス

タスクと各種ハンドラの使い方を理解する

杉本 明加

本章では，処理を実行する主体（処理単位）となるタスクと各種ハンドラの使い方を中心に解説する．なお，本章の内容はTOPPERS/ASPで動作確認しているが，μITRON 4.0仕様互換の機能を使っている個所については，コンフィギュレーション・ファイルの記法やソース・ファイル中のデータ型を変更すれば，μITRON 4.0仕様準拠のリアルタイムOS上で動作する．（筆者）

第3章～第5章では，アプリケーションを開発する際のシチュエーション別に，リアルタイムOS（RTOS）の機能をどのように実装するのかを解説します．理解しやすい小さなパターンを紹介して，アプリケーション要求に対して，それらを組み合わせられるように進めていきます．

1．時間ベースの処理 ―― 時間経過で動作する処理を記述する

組み込みシステムにとって，時間の経過によって行う処理は必須です．時間に同期した処理の代表的なパターンを下記に示します．

1. 指定時間経過後，1回だけ処理を行う
2. 指定時間周期で，繰り返し処理を行う
   （a）固定周期
   （b）可変周期

この3種類のパターンについてそれぞれ実現方法を見ていきましょう．

● 指定時間経過後に1度だけ処理を行う

・タイムアウト処理の実現

外部の機器と通信を行うプログラムの場合，相手側機器の故障や通信路の断線による障害発生を検知できるようにソフトウェアを設計する必要があります（**図1**）．つまり，ある一定期間，相手側の機器から応答が返ってこない場合を検出できるしくみ「タイムアウト処理」が必要です．TOPPERS/ASPを使って，これを実現してみましょう．

図1　タイムアウト処理を行わないと，いつまでも断線に気づかない

タイムアウト処理はTOPPERS/ASPが提供しているアラーム・ハンドラという機能で簡単に実現できます．アラーム・ハンドラとは，指定時刻になったらOSが指定した関数を呼び出す機能です．

図2に示すように，あるタスクからカーネル（OS）に対してアラーム・ハンドラの起動要求を行います．ここでは，アラーム時間として500msを指定しています．カーネルはタイマ割り込みハンドラから，タイムティックと呼ばれる「時間の刻み」を受け取ることで時間の経過を認識します．そして，アラーム要求の500msが経過すると，カーネルはアラーム・ハンドラに対して起動要求を出します．アラーム・ハンドラの起動をカーネルに要求することで，指定時間経過後にアラーム・ハンドラが実行されます．

・アラーム・ハンドラの作成方法と使用方法

アラーム・ハンドラは，コンフィギュレーション・ファイルに静的APIを記述して作成します．

```
CRE_ALM(ID almid , ATR almatr ,
    intptr_t exinf , ALMHDR almhdr );
```

almid：アラーム・ハンドラID
almatr：アラーム・ハンドラ属性．TA_NULLもしくはTA_ACTを指定
exinf：アラーム・ハンドラの引数
almhdr：アラーム・ハンドラの名称（C言語の関数名）ALM_TMOUTという名前のアラーム・ハンド