最適実装機能を有した組み込みシステム向けリアルタイムＯＳの開発

堤　　大祐，山本　　寧

Development of Real-time Operating System for Small-scale Embedded Systems

Daisuke TSUTSUMI, Yasushi YAMAMOTO

抄　録

組み込みシステムはCPUの内蔵メモリで動作させる小規模な用途からイーサネットの通信インターフェースを持つCPUと大容量の外部メモリを用いた大規模な用途までその規模が様々である。そのため，多様なハードウェア環境に合わせてアプリケーション・ソフトウェアを開発する必要がある。今回，メモリ内蔵CPUを用いた組み込みシステムを対象にしたリアルタイムＯＳを開発した。さらに，CPUと外部メモリを併用した組み込みシステムにおいて，CPUの内蔵メモリに今回開発したリアルタイムＯＳをあらかじめROM化して実装することにより，より簡単にリアルタイムＯＳを用いたアプリケーション・ソフトウェア開発が行える教育用評価用マイコンボードを商品化した。

キーワード：リアルタイムOS，μITRON，マイコンボード

Abstruct

The embedded systems have various scales of hardware, from a small system which has a small CPU and a built-in memory to a large system has a high performance CPU and a large external memory. Therefore, we developed a real-time OS for the small-scale embedded systems with the small CPU. And furthermore, for a system which used the small CPU and an external memory, we developed CPU cards which are installed the real-time OS in the built-in memory in advance. Consequently, those cards facilitate the development of the application software with the real-time OS.

KEY-WORDS：Real-time OS, μITRON, CPU card

１．はじめに

組み込みシステムはモータの回転やスイッチの入力に対する応答など時間的制約を伴うリアルタイム処理が求められる。さらに，CPUの高速化と高機能化により，組み込みシステムの性能が向上しているため，相対的にシステム全体に占めるソフトウェアの比重が大きくなっており，開発するソフトウェアの規模も大きい[1)]。このため，アプリケーション・ソフトウェアはリアルタイムＯＳをベースに開発することが一般化しており，複数の処理を同時並行的に実行できるリアルタイムＯＳの必要性が高まっている。

組み込みシステムはその対象によって規模や要求仕様が異なる。たとえば，メモリを内蔵したCPUを用いた小規模または単機能な制御アプリケーションから，イーサネットの通信インターフェースを持ったCPUと大容量メモリを用いた制御アプリケーションまで様々である。

メモリを内蔵したCPUではROM/RAM共にその容量が少なく，アプリケーション・ソフトウェア開発において厳しい

事業名：一般試験研究
課題名：最適実装機能を有した組み込みシステム向けリアルタイムＯＳの開発

制約となる。また，外部メモリを併用する場合においては，内蔵メモリは外部メモリに比べてアクセスタイムが短く，高速処理が可能である。このため，内蔵メモリと外部メモリの両方を効率的に使用することが求められる。

本研究ではメモリ内蔵CPUを用いた組み込みシステムにおいて，リアルタイムＯＳの機能を選択することによって，リアルタイムＯＳとアプリケーション・ソフトウェアを少ない内蔵メモリに実装できる機能（以下，選択実装）とメモリ内蔵CPUと外部メモリを併用した組み込みシステムにおいて，あらかじめリアルタイムＯＳを内蔵メモリに，アプリケーション・ソフトウェアを外部メモリにそれぞれ分離してROM化することにより，メモリ資源を有効活用できる機能（以下，分離実装）を開発した。これら選択実装と分離実装の2種類の最適実装機能をμITRON3.0仕様[2-3)]のリアルタイムＯＳに適用し，実際の制御機器に組み込んで評価し，商品化した。

2．選択実装機能の開発

2.1 選択実装可能なリアルタイムＯＳの開発

リアルタイムＯＳを用いたアプリケーション・ソフトウェアは実行したい処理を複数のタスクと呼ばれる処理単位に分割して開発する。タスクはリアルタイムＯＳの機能を呼び出す時にシステムコールを発行する。リアルタイムＯＳはそのシステムコールによってタスクを切り換えて実行する。

リアルタイムＯＳとタスクの関係を図1に示す。カーネルとはリアルタイムＯＳの中心的な部分で，タスクの切り換えや管理テーブルの管理を行う。管理テーブルはリアルタイムＯＳの機能とタスクの状態を管理するものであり，リアルタイムＯＳの機能とタスクの状態やそれらの関係を管理する。表1にタスク管理テーブルの例を示す。

2種類の最適実装機能の開発は，以前当場で開発したμITRON3.0仕様のリアルタイムＯＳをベース行った[4)]。

選択実装で用いられるメモリを内蔵したCPUではROM/RAM共にその容量が少なく，ROMが32KB～128KB，RAMが4KB～32KB程度である。このような動作条件が厳しい組み込みシステムではソフトウェア全体のサイズがメモリの量より大きくなってしまう場合がある。このとき，リアルタイムＯＳを用いたアプリケーション・ソフトウェア開発において以下のような方策が考えられる。

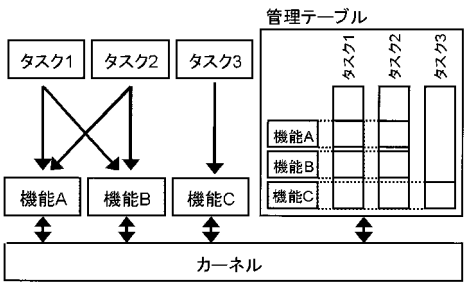


図1　リアルタイムＯＳとタスク

表1　タスク管理テーブルの例

項目	説明
タスクＩＤ	タスクの番号（固定）
現在のスタックポインタ	レジスタなどを待避させる
タスクの状態	DORMANT、WAITなど
現在の優先度	接続されるレディキュー
待ち状態	時間待ち状態など
属しているキュー	レディキューなど
その他	

ａ）使用するタスクの数を減らす。

ｂ）使用するシステムコールを減らす。

ｃ）管理テーブルの管理する項目を減らす。

以上の方策を併用することにより，アプリケーション・ソフトウェアに応じて，使用するシステムコールを選択し，管理テーブルを構成することにより，リアルタイムＯＳが使用するROM/RAMの容量を少なくできる。リアルタイムＯＳの機能選択の項目は以下の通りである。

・タイムアウト機能の有無
・セマフォ機能の有無
・メイルボックス機能の有無
・スリープ機能の有無
・サスペンド機能の有無
・チャネルごとの通信バッファサイズ

2.2 リアルタイムＯＳの機能の選択方法

開発したμITRON3.0仕様のリアルタイムＯＳは使用するタスク数やイベントフラグ数などをあらかじめ設定する必要がある。そのため，タスクの数，タスクの優先度，スタックサイズやイベントフラグの数などリアルタイムＯＳを用いた組み込みシステムの実行に必要な初期化ファイルを生成するためコンフィグレータを使用する[4)]。

コンフィグレータはテキストファイルに一定のフォーマットに従って記述することにより，タスク管理テーブルなどのコントロールブロックの初期状態を定義し，初期化を行うプログラムを自動的に生成するソフトウェアである。

タスクの初期設定には，タスク番号（tskid）と実行するタスク（task），スタックの大きさ（stksz），タスクの優先度（itskpri）などを指定する。これらの指定はテキストファイルに記述し，コンフィグレータによって，初期化に必要なファイルを生成する。タスクの初期状態の指定例を図2に，初期化ファイルを図3にそれぞれ示す。

```
taskcreat = {
    tskid = SHELLTASK;/* タスクID */
    task = ShellTask;/* 実行するタスク */
    stksz = 2048;/* ユーザスタックサイズ */
    itskpri;/* タスク起動時優先度 */
};
```

図2　タスクの初期状態の設定例

```
tcbtbl[0].tskid = SHELLTASK;
tcbtbl[0].task = (FP) ShellTask;
tcbtbl[0].itskpri = 1;
tcbtbl[0].isp = (VH*) &UsrStk1[2048];
```

図3　生成された初期化ファイルの例

リアルタイムＯＳの機能の選択は図4のようにリアルタイムＯＳの機能選択ファイルを作成して行う。このファイルによって，コンフィグレータはシステムに応じたコントロールブロックを生成する。

```
#define SLEEP_OFF
#define TIMEOUT_OFF
#define SCI0READBUFSIZ 32
```

図4　リアルタイムＯＳの機能の選択例

選択実装時における組み込みシステムの開発フローは以下のようになり，組み込みシステムに必要なシステムコールと管理テーブルを有したコンパクトなリアルタイムＯＳを構成することができる。

1．タスク，初期化ファイル作成
2．リアルタイムＯＳの機能選択ファイル作成
3．選択されたリアルタイムＯＳの機能と管理テーブルを有したリアルタイムＯＳのオブジェクト作成
4．初期化情報を含んだタスクのオブジェクト作成
5．リアルタイムＯＳとタスクのオブジェクトをリンク
6．内蔵メモリなどにROM化して実行

2.3　選択実装機能の適用事例

図5に示すような酪農機器の通信制御部分に開発したリアルタイムＯＳを適用した。通信制御部分はＨ８シリーズ[5]のCPUを使用したマイコンボードを用いた。図6に使用したマイコンボードを示す。本装置は牛の個体識別装置から得た情報を個体情報データベースに参照することで，個体にあわせた給餌装置の制御と，外部ハンディターミナルを用いて機器全体の動作の監視を行う装置である。複数の通信チャネル

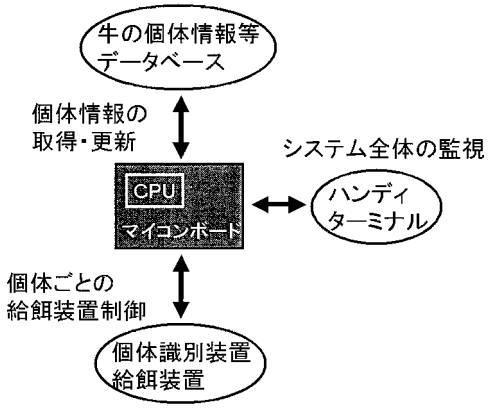


図5　酪農機器の通信制御

図6　酪農機器の通信制御に用いたマイコンボード

を切り換えて情報伝達を行うため，リアルタイムＯＳの機能を選択して内蔵メモリのみで動作させる。これによって，通信制御が高速に行うことができる。これについては実機上で評価中である。

3．分離実装機能の開発

3.1　分離実装可能なリアルタイムＯＳの開発

分離実装において，内蔵メモリは外部メモリに比べてアクセスタイムが速く高速処理が可能である。このため，内蔵メモリと外部メモリの両方を効率的に使用することが求められる。タスクの切り換えなどリアルタイムＯＳの処理を高速に行うことにより，システム全体の処理を効率的に行うことができる。すなわち，リアルタイムＯＳをアクセスタイムの速い内蔵のROMにあらかじめ搭載し，アプリケーション・ソフトウェアを外部ROMに書き込むことにより，メモリ資源を有効に活用できる。この場合のシステムの構成とリアルタイムＯＳの実装を図7に示す。

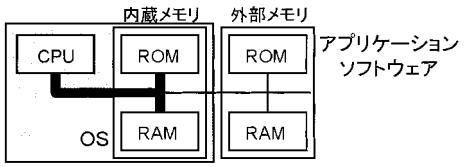


図7　内蔵メモリに実装したリアルタイムＯＳ

リアルタイムＯＳとアプリケーション・ソフトウェアは別々にコンパイル・リンクして実装する。リアルタイムＯＳのみをあらかじめ内蔵メモリに実装するため，アプリケーション・ソフトウェアはリアルタイムＯＳのシステムコール・アドレス情報と一緒にリンクして，外部メモリに書き込む。必要なシステムコール・アドレス情報はあらかじめリアルタイムＯＳを実装したときに取得する。

システムコール・アドレス情報はリンカから出力されるレポートファイルから必要な情報を抽出し，ヘッダファイルと

して提供する。レポートファイルの一例を図８に，このファイルから生成されたシステムコール・アドレス情報ファイルの例を図９に示す。レポートファイルからシステムコール・アドレス情報を生成する専用のジェネレータを開発したので，この変換は自動的に行うことができる。

```
_dly_tsk     H'00005672  ENT
_slp_tsk     H'00004896  ENT
_twai_flg    H'00005BEE  ENT
```

図８　レポートファイルの例

```
#define DLY_TSK_ADD      0x00005672
#define SLP_TSK_ADD      0x00004896
#define TWAI_FLG_ADD     0x00005BEE
```

図９　システムコール・アドレス情報ファイルの例

リアルタイムＯＳをあらかじめROM化して内蔵メモリに実装する場合の開発フローを以下に示す。

１．リアルタイムＯＳのオブジェクト作成
２．内蔵メモリにリアルタイムＯＳをROM化
３．システムコール・アドレス情報を生成
４．タスク，初期化ファイル作成
５．初期化情報を含んだタスクのオブジェクト作成
６．システムコール・アドレス情報とタスクのオブジェクトをリンク
７．外部メモリなどにROM化して実行

3.2　分離実装機能の適用例１（リアルタイムＯＳ搭載16bitマイコンボード）

分離実装機能を適用し，初心者でも簡単にリアルタイムＯＳを使用したアプリケーション・ソフトウェアの開発を行えるマイコンボードを開発した。

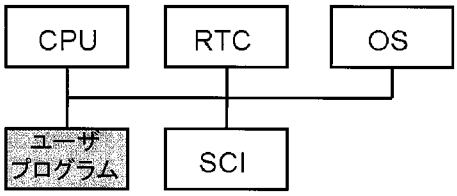
従来，リアルタイムＯＳを用いたソフトウェア開発は以下に示すフローとなる。リアルタイムＯＳはライブラリまたはソースコードで配布されアプリケーション・ソフトウェア開発者の開発環境でコンパイルおよびリンクを行いROM化するのが一般的である。

１．タスク，初期化ファイル作成
２．リアルタイムＯＳのオブジェクト作成
３．初期化情報を含んだタスクのオブジェクト作成
４．リアルタイムＯＳとタスクのオブジェクトをリンク
５．ROM化して実行

分離実装機能を適用すると，アプリケーション・ソフトウェア開発は以下に示すとおり，リアルタイムＯＳとは別にコンパイルして外部メモリにROM化する。そのため，リアルタイムＯＳのコンパイルなど開発環境に関する知識は必要とせず，初心者でも簡単にリアルタイムＯＳが使える環境を実現することができる。

１．タスク，初期化ファイル作成
２．初期化情報を含んだタスクのオブジェクト作成
３．システムコール・アドレス情報とタスクのオブジェクトをリンク
４．外部メモリなどにROM化して実行
（リアルタイムＯＳは内蔵メモリにROM化）

アプリケーション・ソフトウェアからみると，あらかじめROM化したリアルタイムＯＳは，図10のような周辺ハードウェアと同様にアクセスすることと等価となり，容易にリアルタイムＯＳを使用したアプリケーション・ソフトウェア開発を行うことができる。



RTC : Real Time Clock
SCI : Serial Communication Interface

図10　あらかじめROM化して実装したリアルタイムＯＳへのアクセス

H8Sシリーズ[6]のCPUにあらかじめリアルタイムＯＳをROM化して実装した教育用・評価用マイコンボードを図11に示す。

図11　リアルタイムＯＳ搭載16bitマイコンボード

3.3　分離実装機能の適用例２（リアルタイムＯＳ搭載32bitマイコンボード）

インターネットの普及により組み込みシステムにおいてもネットワーク接続に対する要望が多くなってきた。そのため，イーサネットコントローラを内蔵したSH2シリーズ[7]のCPU

を使用した教育用・評価用マイコンボードを開発した。このマイコンボードにおいてもリアルタイムＯＳをあらかじめROM化して搭載し，2MBのROMと1MBのRAMの大容量のメモリを実装した。これによって，制御システムへの適用の他，高性能なCPUとイーサネットコントローラを用いたネットワーク接続用途に使用可能となった。

図12に開発したリアルタイムＯＳ搭載32bitマイコンボードを示す。

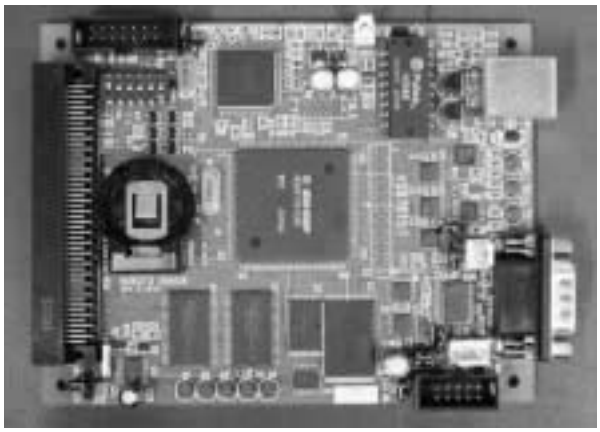


図12　リアルタイムＯＳ搭載32bitマイコンボード

3.3.1　リアルタイムＯＳ搭載32bitマイコンボードの応用例

本ボードとPCまたはRTC搭載マイコンボードを図13のようにイーサネットで接続し，本ボードがPCなどより現在時刻を1秒ごとに取得するネットワーク対応の通信ミドルウェアを開発した。図14に通信して時刻を取得している結果を示す。複数のマイコンやPCがネットワーク接続して使用される場合が今後増加すると思われる。このように，PCまたはRTCを搭載したマイコンボードが1台あれば時刻を共有することができる。

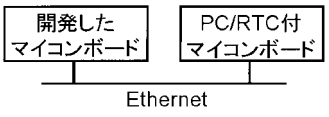


図13　イーサネット接続による時刻取得例

```
TCPtime Start
Fri Mar 15 17:34:42 2002
Fri Mar 15 17:34:43 2002
Fri Mar 15 17:34:44 2002
Fri Mar 15 17:34:45 2002
Fri Mar 15 17:34:46 2002
Fri Mar 15 17:34:47 2002
Fri Mar 15 17:34:48 2002
Fri Mar 15 17:34:49 2002
Fri Mar 15 17:34:50 2002
Fri Mar 15 17:34:51 2002
Fri Mar 15 17:34:52 2002
```

図14　時刻取得結果

4．ま と め

選択実装と分離実装の2種類の最適実装機能の開発した。これにより，小規模な組み込みシステムのメモリ資源を有効に活用でき，効率的なアプリケーション・ソフトウェアの開発が可能になった。

選択実装機能を用いることにより，リアルタイムＯＳが使用するROM/RAMの量を低減することができる。そのため，アプリケーション・ソフトウェアの開発により多くのメモリを使用することができ，組み込みシステムに最適な実装が行うことができる。分離実装機能を用いることにより，リアルタイムＯＳがあらかじめ内蔵メモリにROM化することができた。これによって，リアルタイムＯＳに関する深い知識がなくても容易にアプリケーション・ソフトウェアの開発が行えるようになった。

選択実装においては酪農機器の通信制御部に適用して現在評価中である。また，分離実装においては，教育用・評価用マイコンボードとして商品化された。

謝　辞

最適実装機能を有したリアルタイムＯＳの機能評価に協力していただいた北原電牧株式会社の深谷政廣氏，株式会社北斗電子の中野隆司氏，株式会社土谷特殊農機具製作所の土谷賢一氏に感謝の意を表します。

引用文献

[1] http://www.itron.gr.jp/survey2001/result-j.html
[2] μITRON3.0標準ハンドブック，パーソナルメディア，1993
[3] ITRON標準ガイドブック2，パーソナルメディア，1994
[4] 堤　大祐ほか6名：組み込みシステム向けリアルタイムＯＳの開発，北海道立工業試験場報告，No.299，pp 111-117，2000
[5] H8/3067シリーズハードウェアマニュアル，日立製作所，1997
[6] H8S2148シリーズハードウェアマニュアル，日立製作所，1997
[7] SH7040シリーズハードウェアマニュアル，日立製作所，1996