組込み向けデュアルOS実行システムDARMAの開発†

4B－3

齊藤雅彦*　加藤直**　大野洋**　中村智明**　上脇正*　井上太郎***
*(株)日立製作所日立研究所　**同新事業推進本部　***同システム開発研究所

1. はじめに

組込み向けマイクロプロセッサSH4を搭載するハードウェアプラットホーム上でWindows® CEとμITRONを共存させるDARMA(Dependable Autonomous Hard Realtime Management)システムを開発した．μITRON上で既存アプリケーションを動作させ，Windows® CE上でオープンソフトウェアを実行する．本システムは，既存ソフトウェア資産を継承しつつ，情報サービスなどの新しいオープンアプリケーションを追加していこうという分野，例えば，車載ナビゲーションシステムへの適用に有効である．

本稿では，リアルタイム性，小型化という観点から導入した二つの機構「優先順位統一モデル」「デバイス共有機能」を中心に説明する．

2. 組込み向けDARMAシステムの構成

2.1 DARMAシステムの特徴

弊社では，PC/ATハードウェア上でWindows NT®と独自リアルタイムOSを共存させるDARMAシステムを開発および製品化している[1][2][3]．我々はこの技術を組込みシステムに適用し，Windows® CEとμITRONを共存させるプロトタイプシステムを構築した(図 1)．

DAMRA技術における共通の特徴として，次の三点がある．

(1) 資源分割機能

メモリ，入出力機器を分割して各OSに割り付ける．プロセッサは時分割，タイマは仮想化して，各OSで共用する．

(2) OS間連携機能

異なるOS上のプロセス間で共有メモリ，メッセージ通信，セマフォ(排他制御)を使用できる．

(3) 障害監視，回復機能

OSの動作状況を監視する機能，および，OSに障害が発生した場合，該当OSのみをリスタートさせる機能などを実現する．

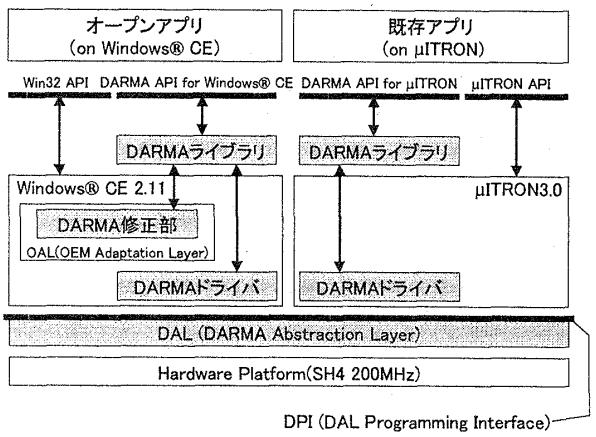

図 1　DARMAシステムアーキテクチャ

上記三つの機能を実現するため，DAL(DARMA Abstraction Layer)というソフトウェア階層を実現し，これに対応するライブラリ・デバイスドライバを両OS上に実装した．DALは32KBのテキスト領域および32KB程度のデータ領域から構成される(データ領域のサイズはシステム内の最大タスク数に依存する)．DARMAライブラリ・デバイスドライバは両OSとも5KB程度のプログラムである．DALとOSとの間の連携はDPI(DAL Programming Interface)と呼ぶインタフェースで規定されており，DPIにしたがったライブラリ・デバイスドライバを作成すれば，他OSを搭載することが可能である．

DALは主要なハードウェアのみを管理する．これ以外のハードウェアは，基本的には，DARMAシステム上で動作するいずれかのOSに占有される．

---

†)Development of Dual OS Execution Environment "DARMA" for Embedded Applications
Masahiko SAITO, Naoshi KATO, Hiroshi ONO, Tadashi KAMIWAKI, Tomoaki NAKAMURA, Taro INOUE
miyabi@hrl.hitachi.co.jp
Hitachi, Ltd.

2．2　組込みシステムへの展開

図 2に，DARMA システムの応用例として，車載向け DARMA システムの構成を示す．

既存ナビゲーション機能を既存 OS(μITRON)で処理し，GUI(Graphical User Interface)，オープンな情報サービスをオープン OS(Windows® CE)で処理する最適な構成が可能となる．

以下，この構成を実現するために導入した主要機能である「優先順位統一モデル」「デバイス共有機能」について述べる．

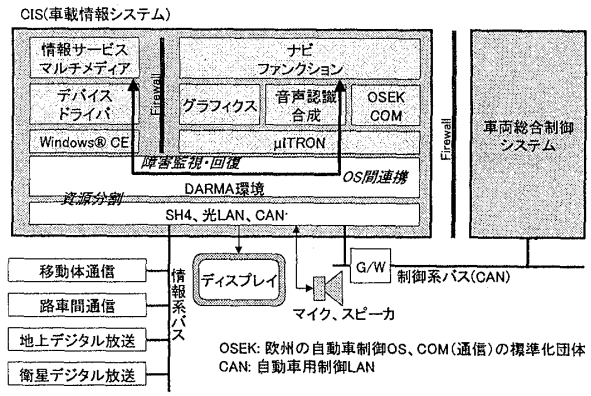

図 2　車載向け DARMA システム

2．2．1　優先順位統一モデル

各 OS へのプロセッサ割当てに関して最も単純な方法は，既存 OS が idle 状態に移行した場合のみオープン OS に切り替えるという方式である．しかしながら，図 2に示したシステム例を考えると，経路探索などのように時間の掛かる処理を実行している最中には，GUI の応答性が低下するといった問題点が生じる．すなわち，既存アプリケーションのリアルタイム性は達成されるが，オープンソフトウェアの応答性が保証されないという問題点がある．このため，優先順位の高いタスクを実行している OS を優先して動作させるという OS スケジューリング方式を導入する．

ここで，異なる OS 間では一般に異なる優先順位体系を有しており，かつ，それぞれの OS に課せられた役割自体も異なる．

例えば，μITRON では最多 255 レベルの優先順位を使用することができるが，Windows® CE 2.11 では，8 レベルの優先順位でアプリケーションを制御する(Windows® CE 3.0 では 256 レベルの優先順位を使用することができる)．また，OS の種類によって，下記のように「優先順位」と「優先順位の数値」の関連が異なっている．

- "The smaller, the higher" Model:優先順位の数値が小さいほど，優先順位が高い．
- "The greater, the higher" Model:優先順位の数値が大きいほど，優先順位が高い．

このような差違を吸収するため，組込み向け DARMA システムでは，正規化優先順位(Normalized Priority)と呼ぶ概念を導入した(図 3参照)．正規化優先順位は，OS 間で共通の優先順位体系である．各 OS における優先順位を一旦正規化優先順位に変換して比較することにより，より重要性の高いタスクを実行している OS を優先して動作させることができる．

例えば，図 3の例では，正規化優先順位体系上では，経路探索タスクよりもボタン制御タスクが高い．このため，ボタン制御タスクが動作している場合には Windows® CE を動作させ，経路探索タスクのみが動作している場合にはμITRON を動作させる．

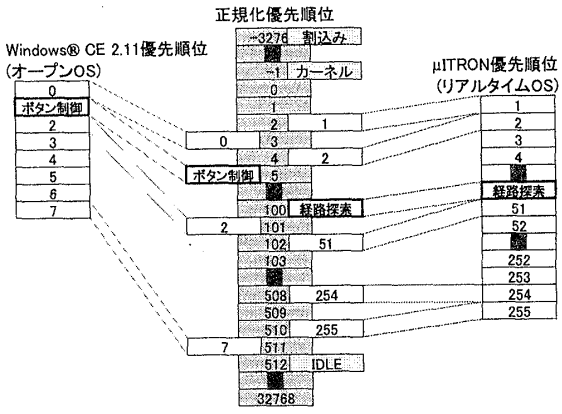

図 3　正規化優先順位

正規化優先順位を導入し，この体系下で比較を行って OS 切替えを行う方法を，ここでは優先順位統一モデル(Priority Unification Model)と仮称している．優先順位統一モデルを搭載した車載向け DARMA システムのアーキテクチャを図 4に示す．本システムでは，タスク切替え時に正規化優先順位を通知する必要があ

るが，公開されているタスク切替え時のフック機能を利用した．

DAL は各 OS から通知された正規化優先順位を比較する．必要があれば，より高い正規化優先順位のタスクを動作させている OS に切り替えて処理を実行させる．

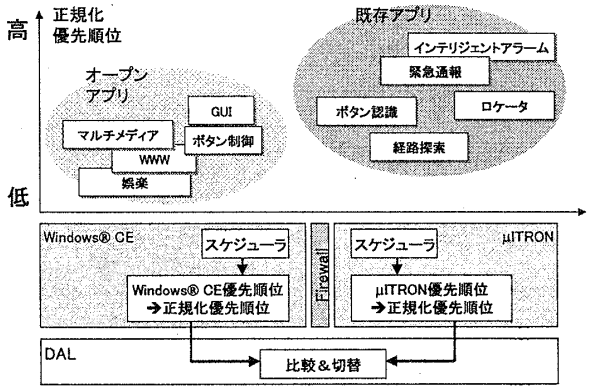

図 4　優先順位統一モデル

2．2．2　デバイス共有機能

前述のように，DAL は最低限のハードウェアのみを管理するため，ディスプレイなどの入出力機器はいずれか一方の OS に占有される．しかしながら，車載システム等の例を考えると，既存アプリケーション，オープンアプリケーションの両者が画面表示を必要とする，すなわち，Windows® CE，μITRON の両者からディスプレイの制御を行わなければならないが，小型化という観点から見れば，両 OS にそれぞれ一つのディスプレイを割り当てることはできない．このため，下記に示す機能を一連のデバイス共有機能として提供する．

- 動的割込み割り振り機能
- OS 間連携機能
  - ➢ OS 間共有メモリ
  - ➢ OS 間メッセージ通信
  - ➢ OS 間セマフォ

一つのハードウェアデバイスからの割込みは，通常，いずれか一つの OS に割り振られる．DARMA システムでは，割込みの種類毎に割り振りを決定できる．動的割込み割り振り機能は，この割込みの割り振りをアプリケーション・デバイスドライバの要求にしたがって変更する機能である．各 OS からの要求により，ある割込みを該当 OS に割り振ったり，別 OS に強制的に割り振ったりすることができる．

OS 間連携機能の詳細な説明は本稿では省略する．なお，ここでは，前節で導入した正規化優先順位との関係上，OS 間セマフォに正規化優先順位による優先順位継承機能を搭載した．これにより，OS 間に跨る優先順位逆転現象を解決することができる．

割込みの動的割り振り機能と OS 間セマフォを使用してディスプレイを共用する例を図 5に示す．

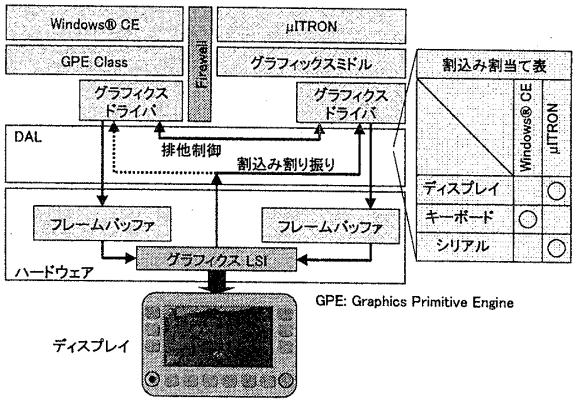

図 5　デバイス共有機能

この例では，両 OS にディスプレイを制御するデバイスドライバを実装している．これとは逆に，一方の OS のみにデバイスドライバを実装し，別 OS からのデバイス入出力は，OS 間メッセージ通信を利用して依頼するという構成方法も可能である．

ここに示したように，デバイス共有機能は，ハードウェアデバイスを仮想化するものではない．デバイスの仮想化によってデバイスドライバの作成を容易なものにすることは，リアルタイム性とのトレードオフの関係にある．DAL は，リアルタイム性重視のため，割込み発生時には最低限の処理を行って各 OS の割込みハンドラに制御を移行させる．共用デバイスの排他制御等は両 OS のデバイスドライバが保証する必要がある．

3．　性能評価

組込み向け小型マイクロプロセッサ SH4 上で本システムの性能評価を行った．SH4 は 200MHz クロックで

動作する.

ここでは,リアルタイム性を測定するため,DAL が割込みを受け付けてからμITRON の割込みハンドラが起動するまでの時間を測定した.割込み発生から割込みハンドラ起動までの処理フローは下記に示すものとなる.

(1) DAL が割込みを受理する.
(2) いずれの OS への割込みであるかを判定する.
(3) 動作中 OS と割込み対象 OS とが異なっていれば,割込み対象 OS に切り替える.
(4) 割込み対象 OS に制御を移行させる.
(5) OS 内で割込み要因を判定し,対応する割込みハンドラに分岐する.

上記(1)～(4)が DAL 処理時間,(5)が OS 処理時間である.二つの処理時間をそれぞれ測定した.

測定条件は下記の二種類である.これらの条件下でμITRON の割込みを発生させた.各 OS ではプロセッサに負荷を掛けるプログラムを動作させている.

- Windows® CE 動作中
- μITRON 動作中

Windows® CE 動作中にμITRON の割込みが発生すると,OS 切替えが発生する.両測定条件における DAL 処理時間の差違は OS 切替えを行うか否かの違いである.OS 切替えを行う場合,動作中 OS のレジスタ等をメモリ上に退避し,割込み対象 OS のレジスタをメモリから復旧しなければならない.

測定結果の最小値・平均値を表 1に示す.

表 1　割込み処理時間測定結果

割込み処理時間		DAL 処理	OS 処理	合計
Windows® CE 動作中	最小値	1.33μs	0.97 μs	2,30 μs
	平均値	1.40 μs	1.05 μs	2.46 μs
μITRON 動作中	最小値	0.36 μs	0.73 μs	1.21 μs
	平均値	0.44 μs	0.93 μs	1.37 μs

平均的に,2μ秒～3μ秒程度の割込み処理時間を達成している.なお,割込み処理時間には,各 OS が割込みを禁止している時間は含まれていない.割込み応答時間を導出するためには,このような割込み禁止時間を加算しなければならない.

4.　おわりに

オープン OS と既存 OS を単一プロセッサ上で共存させる DARMA 技術を組込みシステム向け,特に,車載ナビゲーションシステム向けに適用させた例について説明した.本システムでは,リアルタイム性・小型化を達成するための「優先順位統一モデル」「デバイス共有機能」を新しく搭載している.

200MHz 動作の SH4 マイクロプロセッサでは,割込み処理時間 2μ～3μ秒を達成した.

【参考文献】

[1] 新井ほか:異種 OS 共存技術「DARMA」の開発と制御システムへの適用,計測技術,pp.39-44, Vol.27, No.07, 1999.
[2] 新井ほか:ナノカーネル方式による異種 OS 共存技術「DARMA」の提案,情報処理学会第 59 回全国大会(本大会)一般講演論文集,1999.
[3] 佐藤ほか:ナノカーネル方式による異種 OS 共存技術「DARMA」の実装,情報処理学会第 59 回全国大会(本大会)一般講演論文集,1999.