

# 高信頼かつ高機能を実現する組込み向けOS共存システム

阿部 剛・酒井 淳嗣・鳥居 淳

## 要 旨

高度な情報処理を要求する将来の組込み機器では、従来からの高信頼性や高応答性に加え、その複雑な処理に対応するための高機能性も重要になります。これらを共存させる方式として、組込み用リアルタイムOSとサーバ用SMP OSを統合したデュアルOSを提案、組込み向けマルチコアプロセッサMPCore SoC上に実装し、その効果を実証しました。

## キーワード

●マルチプロセッサ ●組込みシステム ●リアルタイムOS ●高機能OS ●応答性 ●信頼性 ●機能性

## 1. はじめに

近年発展している携帯電話や車載システムなどの組込みシステムは複雑な情報処理能力を求めため、従来から組込みシステムに求められてきた応答性や信頼性だけでなく、高い機能性も要求します。しかしながら、ITRONなどに代表されるリアルタイムOSは、機器制御系処理を主要対象とした最小限の機能提供により応答性能/信頼性を確保しているため、要求されている高い機能性を備えていません。一方、高度な情報処理を主要対象としているSolarisやLinux、Windowsなどの高機能OSは、リアルタイムOSと比して十分な応答性や信頼性を確保できていません。なぜなら、OSの高機能化は実装の複雑化と予測性の低下を招き、組込みシステムが要求する高い応答性と信頼性を達成できなくなるためです。そのため、現在、高機能性を要求する組込みシステムに適したOSは存在していません(図1)。

したがって、現実的に高機能性を備えた組込みシステムを実現するためには、この性質の異なった複数のOSを1つのシステム内で両立させる技術を検討する必要があります。

## 2. OS共存手段

1つのシステムでOSを複数動作させるための課題は、リソース競合の解消です。リソースにはCPU、キャッシュを含むメモリ階層、バス、ペリフェラルに接続された各デバイスが含まれます。これらの調停方法により、OS共存の実現手段

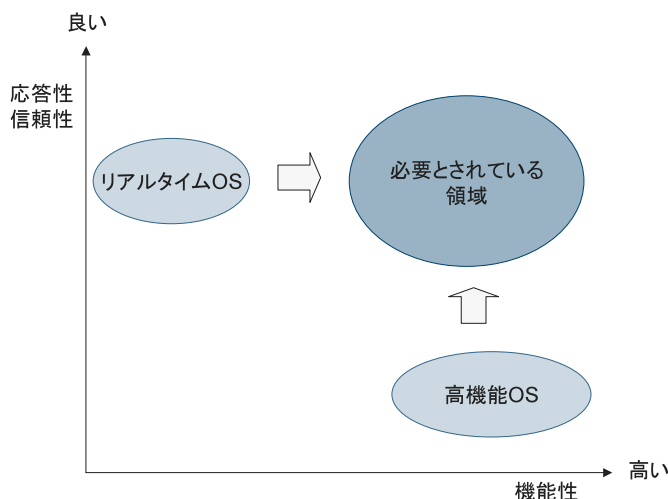


図1 組込みシステム要求

は大きく3種類に分類できます。1) 1つのOS上で残りすべてのOSを動作させるハイブリッド手段、2) OSに対してハードウェアを多重化するミドルウェアを用いる仮想化手段、そして3) マルチコア導入および各OSへの調停用修正によって実現可能なネイティブ手段です(図2)。

表に各手法の特徴をまとめました。性能では各OSから直接リソースアクセス可能なネイティブ手法、実装コストではリソース競合管理に既存OSを使用するハイブリッド手法、汎用性ではハードウェア層を隠ぺいする仮想化手法が優れています。誤動作が人命に関わるような組込みシステムにおいて

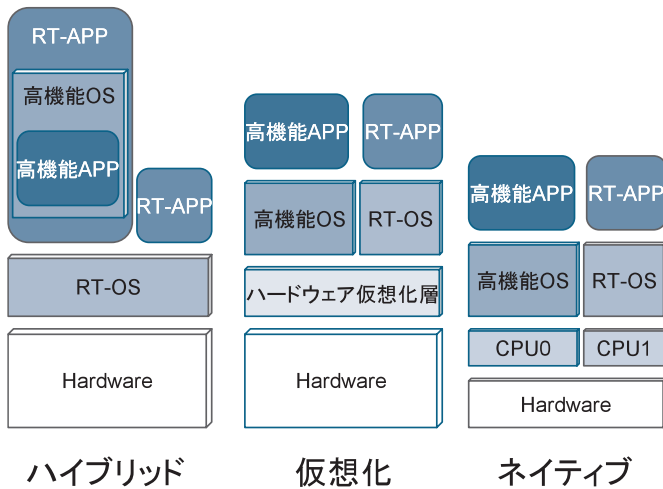


図2 OS共存手段

表 OS共存手法比較

| 手法     | 利点                           | 欠点                                    |
|--------|------------------------------|---------------------------------------|
| ハイブリッド | 既存OSによる競合リソース一括管理で実現コストに優れる。 | 管理OSの能力がシステム全体の性能と信頼性を決定。組込まれたOS性能低下。 |
| 仮想化    | 競合回避を下位層で実現するため高い汎用性を持つ。     | ハードウェア多重化を実現するソフトウェア処理によって性能と信頼性が低下。  |
| ネイティブ  | 直接リソースアクセスするため性能劣化が少ない。      | マルチプロセッサが必要。ハードウェアへの依存度が高く、汎用性が低い。    |

は、ソフトウェアによるOS分離技術は信頼性低下要因となります。NECでは、以上の共存手法の組込みシステムへの適性と半導体集積技術の進展で組込み向けシステムLSIでも複数プロセッサを搭載することが現実的になってきていることから、上記のうちネイティブ手段が最も組込みシステムに相当であると判断しました。

### 3. 組込み向けOS共存SoC環境

ネイティブ手法を用いて現実的な組込みシステムを構築するためには、OSを共存させる技術だけでなく、組込みの特徴である高い応答性と信頼性を維持するための技術も必要となります。NECではこれらを次のような課題と捕らえ、ソフトウェアとハードウェアの両面から解決を図りました。

まず、OSを共存させるための課題は1) 1つの物理リソースを共有するデバイスの調停、2) オンチップ上での高速な

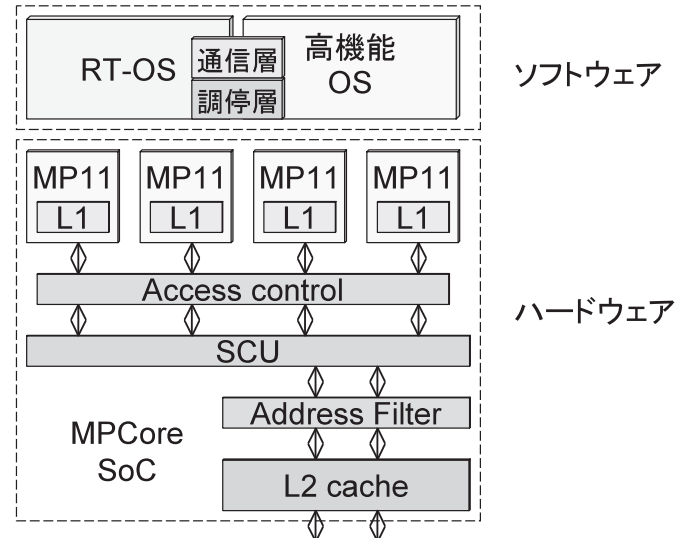


図3 OS共存システム構成

OS間通信機構の提供、となります。これら課題についてはソフトウェア主導の解決策をとりました(図3上部)。

ここでは、OSを目的に応じて3つの機構に分けました。調停層には、課題1) のデバイス調停を解決させるために、共有デバイスの初期化、アクセス制限ハードウェアの設定、各OSの起動処理、OS間調停機構を、通信層には課題2) を解決するためのOS間通信機構を実装しました。ここでは、マルチコアに搭載されるプロセッサ間通信機構を利用することで、高速な通信を実現しています。そして、調停層を利用して共有デバイス競合を解消するよう修正を加えたOSを各CPUに配置することにより全体としてOS共存の課題を解決しました。

一方、組込みシステムとしての要求を満たすための課題は、3) OS間干渉による性能低減の防止、4) OS暴走時の信頼性確保が挙げられます。これらの課題に関してはハードウェア主導の解決策をとりました(図3下部)。課題3) には、組込み向けマルチプロセッサMPCoreが備える割り込み分散機構や柔軟なキャッシュ設定を活用して解決を図りました。これによりOSごとに独立した割り込み設定やL1キャッシュコヒーレンシ設定が可能となり、OS相互間の性能影響を低減できます。課題4) に関しては、MPCore内部の特殊レジスタアクセスをCPUごとに制限可能なアクセスコントロール機能とCPUごとにアクセス可能なアドレス空間を制限するアドレスフィルターを提案することによって解決しました。

## 高信頼かつ高機能を実現する組込み向けOS共存システム

これらの手法により、隣のCPUで動作していた高機能OSが暴走しても、リアルタイムOSの高い応答性と信頼性が揺らぐことのないシステムを構築することが可能となりました。

NECでは以上のような組込み向けシステムアーキテクチャを提唱し、NECエレクトロニクスがARMとの協力の下、組込み向けのMPCore SoCを設計、実装しました。

## 4. 検証

NECでは、TOPPERS/JSPとSMP LinuxによるDualOS環境を構築し、この環境上に車載システム向けデモシステムを構築しました。そして、本組込み向けOS共存環境により応答性、信頼性、機能性の共存が可能であることを実証しました。

デモシステムは、図4のような構成をとっています。道路状況の認識アプリケーションを、TOPPERS/JSP上の画像処理とLinux上の認識処理を連携させながら動作させています。また、現在走行地点の地図と関連情報をLinux上のブラウザで表示し続けています。さらに性能表示と異常監視システムとして機能する性能モニタを両OS上に配置しました。この構築により現実的にOS共存が可能であることを示しました。

さらに、リアルタイムOS側の信頼性に関しては、性能モニタに実装した相互OS監視機能を利用し、Linux側の異常負荷時およびデッドロックを検知させ、Linux側が使用しているCPUを強制終了させる試みを行いました。その結果、Linux側が異常動作中でもTOPPERS/JSP側は安定性を損なうことが無いこと、Linux側を強制終了させてもTOPPERS/JSP側に影響が出ないことを確認できました。

また、無負荷時に平均3usec/最悪値6usecであった応答時間(割り込み発生から割り込みハンドラ起動までの時間)は、

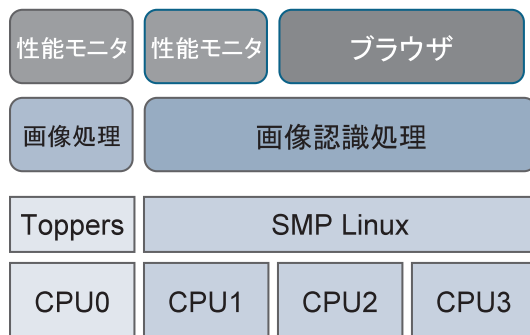


図4 デモシステム構成



図5 デモの様子

両OS間で競合するバスに最大負荷(帯域100%)をかけた状態でさえ平均6usec/最悪値15usecを維持しており、高い応答性を確保できていることを確認できました。さらに、一定負荷以上では最悪値の傾向が変化しないことから、各CPUからバスへの要求調停が公平(ラウンドロビン)に行われる機構により、飽和状態でも一定の性能が維持されることも確認できました(図5)。

## 5. むすび

今回は、次世代組込みシステムに求められている要件を概観し、それを実現するためのマルチコアベースのSoCシステム技術に関するNECの取り組みについて説明しました。

NECでは、これまで培ったハードウェア、およびソフトウェア技術を生かし、今後もより高信頼/高性能/高機能な組込みシステムをお客様に提案してまいります。

## 執筆者プロフィール

阿部 剛  
システムIPコア研究所  
SoCデザインTG 担当

酒井 淳嗣  
システムIPコア研究所  
SoCデザインTG 主任研究員  
情報処理学会、IEEE CS各会員

鳥居 淳  
システムIPコア研究所  
SoCデザインTG 主任研究員  
情報処理学会会員