報告番号	※甲　　第　　号

主　論　文　の　要　旨

論文題目　組込みリアルタイムアプリケーション統合のための階層型スケジューリング

氏　　名　松原　豊

論　文　内　容　の　要　旨

近年，ハードリアルタイム性を要求される組込みシステムの分野においても，ソフトウェアの大規模化，複雑化が著しく，その信頼性をどのように確保・向上させるかが大きな課題となっている．ハードリアルタイム性を要求される代表的な組込みシステムである自動車制御システムも，低燃費化，走行性能の向上，排気ガス規制への対応などの目的から，大規模化，複雑化が進んでいる．自動車に搭載される制御用コンピュータをECU（Electronic Control Unit；電子制御ユニット）と呼ぶが，1台の自動車に搭載されるECUの数は，多いもので，1998年では30個程度であったが，2003年には70個，2010年には100個程度まで増加しているといわれている．ECUの数は，さらに増加する傾向にあり，コストの増加やECU搭載スペースの不足という問題を引き起こしている．

自動車に搭載されるECUの数が増加する理由の1つとして，エンジン，ステアリング，ブレーキといった制御対象毎に別々のECUが用いられ，その統合が進んでいないことが挙げられる．それに加えて，自動車制御システムに新しい機能を追加する場合にも，サプライヤから提供されるECUを，自動車メーカが制御システムにその都度追加する開発プロセスも1つの要因である．

自動車に搭載されるECUの数を減らすためには，複数のECUを1つのECUに統合する必要がある．しかし，ECUで動作するソフトウェア（アプリケーションと呼ぶ）が大規模，複雑化することで，ECUを統合する開発プロセスも複雑化することや，アプリケーションの開発，検証コストが増加することなどの理由により，統合が進んでいない．その背景にある技術的な問題としては，ECUに使われているOSが保護機能を持っていないことが挙げられる．OSに保護機能がないと，複数のECUで動作しているアプリケーションを1つのECUに統合する場合に，あるアプリケーションの不具合により，別のアプリケーションが障害を引き起こす可

能性がある．その結果，アプリケーションの信頼性が低下するだけでなく，統合した状況において発生した障害の切り分けが困難になる．

これまで，組込みリアルタイムアプリケーションの統合を容易に実現するための保護機能を持つリアルタイム OS（RTOS と呼ぶ）が開発されている．RTOS の保護機能には，メモリ保護に代表される，資源へのアクセス保護機能と，時間多重される資源に対する時間保護機能がある．これまで提案された時間保護機能は，統合後のアプリケーションのリアルタイム性を保証しないものが多く，アプリケーション統合における検証の負荷を低減させる標準的な技術にはなっていない．

本論文では，分散制御システムにおいて，個別のプロセッサで統合前に固定優先度ベーススケジューリングでスケジュール可能であったアプリケーションを，1 つのプロセッサに統合して動作させる状況を想定し，アプリケーション単位でプロセッサ時間を保護する階層型スケジューリングアルゴリズムを提案する．提案するアルゴリズムを，既存の RTOS に実装して処理時間を評価することで，提案アルゴリズムの実現可能性と実用性を明らかにする．

本論文の具体的な内容は，次の 4 つである．

1 つ目に，時間保護の機能要件を明確にするため，スケジューリングアルゴリズムが満たすべき要件を 3 つに整理し，それらの要件を満たす階層型スケジューリングアルゴリズム（時間保護アルゴリズム 1 と呼ぶ）を提案する．時間保護アルゴリズム 1 は，2 階層のスケジューリングアルゴリズムである．各アプリケーションのタスクをスケジュールするローカルスケジューラは，固定優先度ベーススケジューリングを，アプリケーションをスケジュールするグローバルスケジューラは EDF（Earliest Deadline First）を採用している．このアルゴリズムは，タスクの起動時刻のタイミングで，デッドラインまでに使用可能なプロセッサ時間（バジェットと呼ぶ）を割り当てる．したがって，タスクの正確な実行時間の情報を必要とせず，タスクの起動時刻と相対デッドラインの 2 つの情報が得られることを前提とする．時間保護アルゴリズム 1 の詳細と，具体的なスケジュール例を示し，時間保護の 3 つの要件を満たすことを証明する．

2 つ目に，タスクの起動時刻が不明なアプリケーションにも対応するために，BSS（Bandwidth Sharing Server）アルゴリズムをベースに時間保護アルゴリズム 1 を拡張したアルゴリズム（時間保護アルゴリズム 2 と呼ぶ）を提案する．時間保護アルゴリズム 2 は，時間保護アルゴリズム 1 のローカルスケジューラに対してタスクの起動を遅延する仕組みを加え，グローバルスケジューラに対して BSS アルゴリズムをベースにしたバジェット管理手法を導入したものである．タスクの起動時刻が不明なアプリケーションにも適用できるため，時間保護アルゴリズム 1 と比べて，より多くのアプリケーションに時間保護機能を適用できる．しかしながら，実行時の CPU 負荷により，実行時間が変化するよう QoS 制御されたタスクを含むア

プリケーションに対しては，時間保護を実現できないという制限がある．時間保護アルゴリズム 2 を用いてアプリケーションを統合することで，統合前に固定優先度スケジューリングによりスケジュール可能なアプリケーションは，タスクの優先度を変更することなく，統合後にもスケジュール可能であることを証明する．

3 つ目に，提案した階層型スケジューリングアルゴリズムの実現可能性を確認し，性能を評価するための柔軟なスケジューリングフレームワークを開発する．満たすべき時間要件や，統合段階で既知であるパラメータはアプリケーションごとに異なる場合が多い．そのため，単一の階層型スケジューリングで多様なアプリケーションに対応することは困難である．このフレームワークでは，提案した階層型スケジューラの共通点を整理し，ローカルスケジューラとグローバルスケジューラ間の標準インタフェースを定義する．このインタフェースを用いると，複数のスケジューリングアルゴリズムを同一の RTOS に実装できるようになる．RTOS に複数のアルゴリズムを実装できると，アプリケーション開発者は，アプリケーションごとに適切なアルゴリズムを選択できるようになる．提案フレームワークを用いて，時間保護アルゴリズム 1 に加えて，アプリケーションを周期的に起動するアルゴリズムの 2 種類を実装する．性能評価として，同一アプリケーション内でのタスク切替時間と，異なるアプリケーション間でのタスク切替時間を測定する．その結果，提案フレームワークを用いて実装した RTOS が実用可能な範囲のオーバヘッドで，実装可能であることを明らかにする．

4 つ目に，これまで提案した階層型スケジューリングに対して，容易に追加可能な割込み処理スケジューリングアルゴリズムを 2 つ提案する．これらのアルゴリズムにより，タスクに加えて，割込み処理が含まれるアプリケーションにも対応できるようになる．1 つ目のアルゴリズムは，割込み処理をスケジュールする際に，予め設定された，割込み処理の優先度を，すべての実行可能な処理（すなわち，割込み処理とタスク）と比較するアルゴリズム（グローバル固定優先度スケジューリングと呼ぶ）である．2 つ目は，割込み処理の優先度を，所属するアプリケーションの実行可能な処理に限定して比較するアルゴリズム（ローカル固定優先度スケジューリングと呼ぶ）である．既存プロセッサのもつ割込みコントローラでは，発生した割込みの優先度が，実行中の処理の優先度よりも高い場合に，即座に割込み処理を実行するものが多い．したがって，グローバル固定優先度スケジューリングは，既存の割込みコントローラを用いて容易に実装可能である．それに対して，ローカル固定優先度スケジューリングをそのまま実装するのは難しい．そこで，ローカル固定優先度スケジューリングを既存の割込みコントローラでも容易に実装できるようにするため，割込み処理を，割込みハンドラと割込みサービスタスクの 2 つに分割した割込み処理モデルを提案する．このモデルを使用することで，割込みハンドラの処理時間を短く抑えることができ，割込み処理の本質的な

部分である割込みサービスタスクを通常のタスクと同様に扱うことが可能となる．提案する 2 つのアルゴリズムをスケジューリングフレームワークを用いて実装し，割込み応答性と，割込み処理起動時のオーバヘッドを評価する．そして，これら 2 つの手法を，アプリケーションの独立性，実装容易性，割込み応答性能等の観点で比較する．さらに，割込み禁止区間を含むアプリケーションを統合する場合のスケジュール可能性を解析する手法を提案する．この解析手法を用いることで，システムの設計段階で，アプリケーションの詳細な情報を知ることなく，アプリケーション統合の可否を判定することができる．

本研究では，まず，ハードリアルタイム性を要求されるアプリケーションを統合する際の課題を指摘し，それらを解決する時間保護の要件を明確に定義した．それらの要件を満たす階層型スケジューリングアルゴリズムとして，時間保護アルゴリズム 1 と 2 を提案し，時間保護の要件を満たすことを証明した．さらに，提案したアルゴリズムを含め，多様なアルゴリズムを同一の RTOS に実装するためのスケジューリングフレームワークと，割込み処理を含むアプリケーションに対応するためのスケジューリングアルゴリズムを提案した．提案したアルゴリズムを実際に実装し，実用性の高いアルゴリズム及び RTOS を実現可能であることを明らかにした．これらの成果が，組込みリアルタイムアプリケーションの統合を促進することにつながり，さらに RTOS 技術の発展に貢献することを期待する．