2008－EMB－10 (3)
2008／11／27

動作記録を用いたスケジューラ切替え機構の開発

黒井崇史†，早川栄一‡
†拓殖大学大学院工学研究科 〒193-0985 東京都八王子市館町815-1
‡拓殖大学工学部 〒193-0985 東京都八王子市館町815-1
E-Mail： †kuroi@os.cs.takushoku-u.ac.jp，‡hayakawa@cs.takushoku-u.ac.jp

本稿では，既存のスケジューラの動作記録を基にした切替え条件によって複数のリアルタイムスケジューラを切り替える機構の開発について述べる．ターゲットに複雑化に伴い，ターゲットに適応できるリアルタイムスケジューラを簡易に開発したいという要求がある．そこで本システムでは，OS動作のロギングにより得られた組込みシステムの動作記録を基にして，リアルタイムスケジューラの切替え条件を生成するスクリプトライブラリならびに，既存のリアルタイムスケジューラを切り替える機構の開発を行った．また，この機構を利用したリアルタイムスケジューラの開発についても述べる．

Development of a Scheduler Switching Mechanism using Execution Histories

Takashi KUROI†, Eiichi HAYAKAWA‡
†Graduate School of Engineering, Takushoku University 815-1 Tate-machi, Hachouji City, Tokyo, 193-0985 Japan
‡Faculty of Engineering, Takushoku University 815-1 Tate-machi, Hachiouji City, Tokyo, 193-0985 Japan
E-Mail： †kuroi@os.cs.takushoku-u.ac.jp，‡hayakawa@cs.takushoku-u.ac.jp

This paper describes the development of a real-time scheduler switching mechanism based on execution histories of existing schedulers. Increasing complex real-time applications, simple development of adaptive real-time scheduler is required. Therefore script libraries and a scheduler selection mechanism that is based on the logged execution histories of the operating system are developed in this system. In addition, the paper describes the development of a real-time scheduler using the mechanism.

1. はじめに

組込み機器に特化させたシステムを設計する際，重点を置く事柄としてリアルタイム処理や低消費電力化といったものが挙げられる．組込みシステムに求められる処理は増大しつつあるが，現在組込みシステムには，我々が普段利用している計算機と同様の性能や機能が要求される．しかし，組込み機器は普段利用する計算機とCPUの処理速度や記憶装置の容量，消費電力が異なることから，従来と同様の設計方法ではリアルタイム処理や低消費電力化を実現するは難しい．

従来，このようなリアルタイム処理を必要とする場合のスケジューリング方法には，アダプティブ（適応）・スケジューリングと呼ばれるシステムの実行中に優先度の変更を行うという手法[4][5]が採られてきた．しかし，この手法には，実行中のパラメータ再計算に伴うオーバヘッドの発生があり，タスクの実行の最に余裕が見込まれない場合，再計算オーバヘッドが致命的な原因になるおそれがある．そのため，パラメータの再計算をシステムの実行とは切り離して行うことでタスクの実行に余裕がない場合でも，リアルタイム処理を実現できると考えられる．

そこで本研究では，ソフトウェア特にリアルタイムOS（以下 RTOS）のスケジューラに着目して実時間の保証を実現しようと考えた．しかし，スケジューラそのものの開発では開発規模も開発時間も大きくなってしまうため，コストの削減というものを考えたとき現実的とはいえない．本研究では，先行研究として開発を行った実際に利用されている既存のリアルタイムスケジューラを複数用いて，それらのスケジューラのリアルタイム性能の評価を取るシステム[1]を用いて，動作結果を基にスケジューラを切り替えて動作をさせる機構の開発を行った．

2. 問題分析

リアルタイムスケジューラを用いた場合のスケジューリング可能性の検証として，シミュレータを用いた方式がある（Cheddar[2]，[3]）．この方式には，計算機一台用意するだけでよく，検証に必要な機器を用意しなくても良いという特徴があるが，次に挙げる問題点が存在する．

(1) 限定的な条件下での実験

マルチメディア用の組込みシステム向けスケジューリング手法の研究[3]では，ターゲットを事前にマルチメディアアプリケーションに絞っており，[2]を用いたシミュレーションでもタスクの遷移だけで，組込みシステムの設計で重要な I/O の動作を考慮した実験を行っていない．また，切り替えるスケジューラと切替えの条件についても，Earliest Deadline First スケジューリングと固定優先度スケジューリングだけの切替えとなっており，条件の設定の理由についても言及していない．先に挙げた二つのスケジューラだけでは最適とはいえない上，切替えの条件を明示的に設定する必要がある．

また，組込みシステムで時間保護を実現するためのスケジューリングアルゴリズムとしては[4]をはじめとして過去の実行状態や時間を基にして，動的にパラメータを変えてスケジューリングを行うという手法がとられている．しかし，これには次のような問題点が考えられる．

(2) 再スケジュールのオーバヘッド

従来の過去の結果や実行予測をするアダプティブ・スケジューリングは，実行中に最悪実行時間（Worst-Case Execution Time，WCET）の見積もりや直前のタスク実行時間，デッドラインまでの余裕時間からパラメータの再計算をして優先度の変更を行い，スケジューリングを行っている[5]．単純な処理や高性能な CPU を用いる場合であれば問題ないが，処理能力の乏しい機器や複数の処理を同時に実行する必要がある場合は再スケジュールの処理自体が元でスケジューリング不能になりかねない．そのため，スケジューラは可能な限り，動的に優先度が変わらないほうが望ましいと考えられる．

3. 設計方針

上記の二つの問題分析より，より実際の環境に近い環境でリアルタイムスケジューラの評価実験と，スケジューラ切替えによって時間保護を実現するために，次の方針を定める．

(1) 切替え条件の明示

本研究では，[1]を用いることで明示的にスケジューラの切替え条件を示す．スケジューラを切替える条件を明示することで，システム設計において，どの入出力動作やタスクの実行が，システムの遅延の原因になっているかを知ることができ，設計手法の改善に役立てることもできる．

(2) パラメータ再計算のないスケジューラ

スケジュール時にパラメータが変わることは，パラメータの設定のために再計算が必要になり，タスク実行時間に余裕がないシステムでは再計算自体がオーバヘッドになる．動作させるスケジューラはパラメータが実行中に変わらないことが望ましいので，事前に起こりうる状態を実験し記録をとっておく．そのため，RTOS の再スケジュール時にパラメータの再計算が必要なスケジューラを用いなくても，実時間の保証を可能とする．

4. 全体構成

システムの構成は，個々のスケジューラで性能評価実験を行い，結果をログファイルに出力する機構とログファイルを基にスケジューラ切替え条件を生成するスクリプト（図 1），スケジューラの切替えを行う機構（図 2）の三つから成り立つ．

4.1. 動作ログ解析機構の構成

本研究のスケジューラ切替え機構は[1]の動作ログ解析機構による動作ログの取得を前提としている．この動作ログを解析専用のスクリプトを用いて解析して，実時間保証のできなくなる原因の発生時の状態を導きだし，スケジュールを切り替える条件を生成するために必要な情報をそろえる．動作ログ解析によって生成する解析内容を次に示す．

i．優先度逆転問題：発生時刻，高優先度タスク，実行タスク，高優先度タスク起動時刻
ii．デッドロック：発生時刻，発生タスク，発生タスク起動時刻，クリティカルセクション
iii．デッドラインミス：発生時刻，発生タスク，発生タスク起動時刻

4.2. スケジューラ切替え機構の構成

(1) スケジューラ切替え条件生成

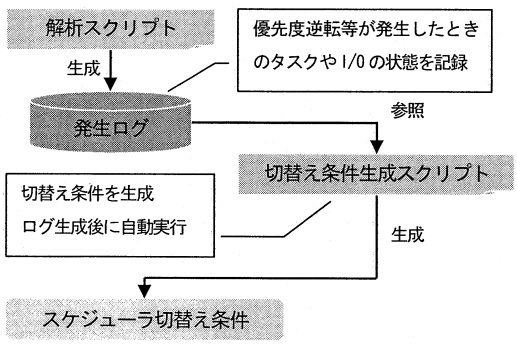

図 1　切替え条件生成の流れ

スケジューラ切替え条件の生成は，生成用のスクリプトに発生ログに読み込ませる．生成用のスクリプトは，実験動作ログ取得解析機構の動作ログ解析スクリプトと連動していて，発生ログの生成が終わると同時にスケジューラ切替えの条件の記述されたファイルを生成する．生成までの

流れを図1に記す.

切替え条件生成スクリプトは，動作ログ解析スクリプトが生成したスケジューリング不能の発生ログと動作ログを読み込んで，基本とするスケジューラの決定とスケジューラの切替える条件，切り替えるスケジューラを記述した切替え条件ファイルを生成する．生成する切替え条件ファイルで監視する対象を次に記す.

ⅰ．タスク
ⅱ．I/O
ⅲ．セマフォ
ⅳ．システムコール
ⅴ．優先度
ⅵ．起床間隔

(2) 切替え機構

スケジューラ切替え機構はRTOS内に組み込んだ切替え機構本体と切替え条件生成スクリプトで生成された切替え条件からなる.

切替え機構の構成を図2に記す.

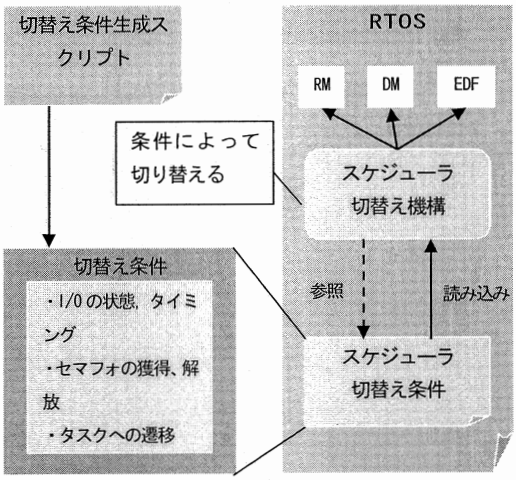

図2　スケジューラ切替え機構の構成

スケジューラ切替え機構本体は，読み込んだ切替え条件を参照し，条件の一致により実装済みの他のスケジューラへの切替えを行う．スケジューラ切替え条件は，切替え機構に対してスケジューラ切替えの条件を受け渡す役目を担う.

5. 設計

5.1. 実験動作ログ取得解析機構

実験動作ログ取得解析機構は，個々に実装した数種類のスケジューラでRTOSを動かし，実行中に実時間保証のできなくなる原因を検出する動作ログを記録する．記録した動作ログは，PCで動作ログ解析用スクリプトを用いて，実時間保証不可の原因の発生時の状態を取り出したログを生成する．動作ログ解析スクリプトの役割を図3に示す.

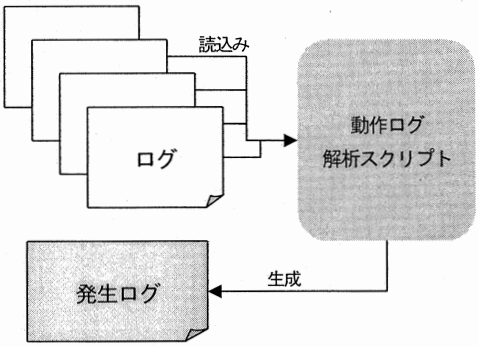

図3　動作ログの解析

優先度逆転問題の解析を行う際は，タスク遷移時ログの遷移前後の優先度を比較し，遷移後の優先度が低いとき，タスク終了時ログの遷移時刻と同じ時刻でタスクが終了しているか確認して，終了していなければ優先度逆転の発生として発生ログに記録する．記録する発生ログの内容を次に記す.

ⅰ．発生時刻：整数値（μ秒）
ⅱ．高優先とタスクID：整数値（1～）
ⅲ．高優先度タスク優先度：1～16
ⅳ．実行タスクID：整数値（1～）
ⅴ．実行タスク優先度：1～16
ⅵ．高優先度タスク起動時刻：整数値（μ秒）

例えば，システム時刻 4500000μ秒に起動時刻 4300000μ秒で優先度10のタスク1が待ち状態にあるにもかかわらず，優先度5のタスク2のほうが実行状態にあるとき，発生ログは図4のようになる.

```
4500000,1,10,2,5,4300000
  ⋮
```

図4　スケジューリング不能発生ログのサンプル

デッドラインミスの解析は，周期タスクと非周期タスクとで分けて解析する．解析する際は，周期タスクならば実行時間とデッドラインを，非周期タスクならば終了時刻とデッドラインを比較する．どちらの場合も，デッドラインを超過していた場合，デッドラインミスの発生として発生ログに記録する．記録する発生ログの内容を次に記す.

ⅰ．発生時刻：整数値（μ秒）
ⅱ．発生タスクID：整数値（1～）
ⅲ．発生タスク起動時刻：整数値（μ秒）

デッドロックの解析の際は，遷移時タスクと終了時タスクログより，終了していないタスクのIDを探し，I/Oデバ

イスタスクログと同期プリミティブログの中で I/O やセマフォを獲得したまま返却をしていないタスク ID と，終了していないタスク ID を比較する．獲得したままで終了していないタスクがあった場合，デッドロックの発生として発生ログに記録する．記録する発生ログの内容を次に記す．

ⅰ．発生時刻
ⅱ．発生タスク ID
ⅲ．発生タスク起動時刻
ⅳ．クリティカルセクション（I/O アドレス，セマフォ ID）

ⅱ～ⅳはデッドロックを起こしているタスクの数だけ記録する．

上記の，三つの実時間保証ができなくなる原因の発生ログを生成し，スケジューラ切替え条件生成用スクリプトに読み込ませる．

5.2. スケジューラ切替え機構

5.2.1. スケジューラ切替え条件生成スクリプト

スケジューラの切替え条件の生成部は，動作ログから導き出された実時間保証不可の原因の発生ログを読み込んで，保証原因の発生時のタスクや I/O の状態から，スケジューラ切替えの条件の生成を行う．

(1) 発生ログの読込み

発生ログの読込みは，ログの形式が図 4 のようになっていることから 1 行ずつ行単位で読込んでいく．ログファイルは配列上に位置関係が対応するように読み込む．一つのログは，一つの配列上に記録される．また，切替え条件に I/O やセマフォなどの共有資源等の状態も用いるため，実験動作ログ取得解析機構の動作ログ解析スクリプトに読込んだ動作ログの配列も読込んでおく．

(2) 実時間保証不可状態の抽出

動作ログと発生ログから，実時間を保証することができなくなる時に，原因が発生した時と発生したタスクの起動時の待ち状態タスク，I/O，セマフォの状態を抽出する．そのほかのスケジューラの動作ログから，この状態と一致する地点を探し，発生していないかを確認する．

(3) スケジューラ切替え条件の生成

実時間保証不可原因の発生があったのであれば，この状態になった時に発生のないスケジューラに切り替えるように，条件ファイルを生成する．発生していれば，検証の対象を別のスケジューラに移し，同一条件で発生しないかを検証する．この作業を繰り返し行い，動作の基本とするスケジューラが実時間保証不可原因の発生点をすべて回避して，切り替えるように条件ファイルを生成する．

このとき，原因が発生していないスケジューラが複数存在した場合，スケジューラにつけた優先度の高いスケジューラを使用する．スケジューラの優先度の設定は，実験時に切替えの条件になる状態からスケジューリング不能になるまでの時間が長いスケジューラの優先度を高く，短いスケジューラの優先度を低く設定する．

また切替え条件をそのままテキストファイルに記述しても RTOS とともに 1 ファイルに ROM 化できないので，切替え条件は C 言語のヘッダファイルにマクロとして記述される．切り替える条件として記述される内容は，実行状態のタスク，使用している I/O やセマフォ，システムコール，実行しているスケジューラ，切り替えるスケジューラである．

生成する切替え条件は，切替えを行うために実装したスケジューラ n 個に対して，$n(n-1)$ 個生成する．切替え条件ファイルはスケジューラ切替え機構にヘッダファイルとしてインクルードされ，RTOS とともにコンパイルされ ROM 化する．

5.2.2. 切替え機構

スケジューラの切替えは，切替え条件のヘッダファイルの条件式マクロを読込んで，条件にタスクや I/O，セマフォ，使用するシステムコールに一致したときスケジューラの切替えを行う．

(1) 切替え条件に該当する状態の検出

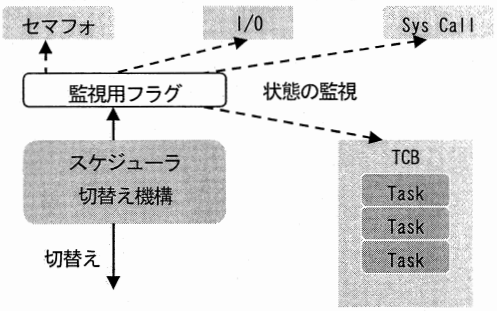

図 5　状態検出

切替えの条件に該当する状態を検出するには，1 ビット（BOOL 型）の変数をフラグとして埋込み，使用したタイミングで切替え機構に通知し，切替えを行う（図 5）．ただし，タスクはコンパイル前に静的 API を用いてコンフィギュレーションファイルに記述するという μITRON の仕様上，Task Control Block 構造体のメンバとして追加記述する．フラグの値は初期状態の時および使用していない状態のときは 0 で，使用状態の時に 1 となる．

(2) スケジューラの切替え

スケジューラの切替えは，状態変化フラグの変化を見てスケジューラの切替えを行う．

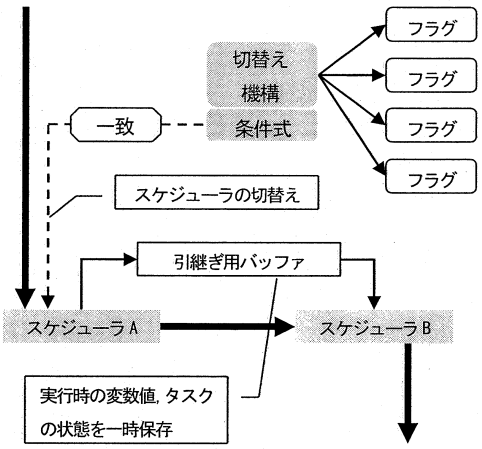

図6　スケジューラの切替え

切替えの時，実行状態および待ち状態にあるタスクの内容と変数値を，切替え時の引継ぎ用バッファに保存し，実行していたスケジューラを終了する．スケジューラを終了したら，切替え先のスケジューラに引継ぎ用バッファに保存しておいた内容を渡して実行を再開する．引継ぎ用バッファの内容は，切替えが終わるとメモリ上を圧迫するだけなので解放され，再び切替え時に再利用される．スケジューラの切替えは，条件式に当てはまるだけ繰り返される．

6. 実現

6.1. 実現環境

表1　実現環境

CPU	SH7709S (SH3) 117MHz
SDRAM	32MB (メインメモリ)
FLASH	8MB (ブートおよびdisk領域)
SRAM	512MB (バッテリーバックアップdisk領域)
RTOS	TOPPERS/JSP 1.4.2
コンパイラ	gcc-2.95.3

本研究では，実現環境として次の表1の組込みシステム[6]およびRTOS[7]を使用した．

6.2. 対象スケジューラ

本研究で比較実験を行い，切替え機構を用いて切替えを行うスケジューラを次に記す．

ⅰ．Rate Monotonic Scheduler (RM)
ⅱ・Deadline Monotonic Scheduler (DM)
ⅲ．Earliest Deadline First Scheduler (EDF)
ⅳ．Least Laxity First Scheduler (LLF)

この4つのリアルタイムスケジューラを用いた理由として，いずれも組込みシステムへの数多くの実装例があるという点と，スケジューラが必要とするパラメータが異なるという点からである．

6.3. 評価実験

(1) 検出対象スケジューリング不能要因

本研究でのリアルタイムスケジューラ評価実験で，観測対象とする実時間の保証ができなくなる原因は4．1．で挙げた次の三つである．

ⅰ．優先度逆転問題
ⅱ．デッドラインミス
ⅲ．デッドロック

これらの実時間保証のできなくなる原因を観測対象にした理由を，次に記す．

ⅰ．優先度逆転問題およびデッドロックは，システムの遅延や実行不能になる事例として，一般的に数多く取り上げられていることから，今回観測対象として選択した．
ⅱ．デッドラインミスは，組込みシステムの前提条件として，守られているか否かを観測する必要があると考え，観測対象として選択した．

(2) リアルタイムスケジューラ評価実験

スケジューラの評価実験は個々のスケジューラ単独での評価を取り，ログを生成する第一段階の実験と，第一段階で取ったログを基にスケジューラを切替えて動作を行う第二段階の二つに分かれている．

一段階目は，6．2．で挙げた四つのスケジューラ個々の動作ログを取得する．事前に四つのスケジューラを単独で実装し，RTOSをクロスコンパイラでコンパイルした四つのROM化したRTOSを準備する．このとき，実験を行う環境[2]は動作ログを取得する組込み機器と外部からイベントを与える組込み機器に分かれているため，動作ログを取得する組込み機器，外部からイベントを与える組込み機器両方に出力を行うタスクを設定する．例として，動作ログを記録する組込み機器に設定するタスクを表2に外部からイベントを与えるタスクを表3に記す．

表2　動作ログ記録機器側タスク

タスク対象	タスク動作内容
LED	1s点灯，1s消灯を繰り返す
LED	1s点灯，3s消灯を繰り返す
PIO	3s出力，2s停止を繰り返す

表3　外部イベント用機器側タスク

タスク対象	タスク動作内容
PIO	1s 出力，1s 停止を繰り返す
PIO	2s 出力，5s 停止を繰り返す
PIO	レジスタ初期値 0，10s で 1023 まで増加，10s で 0 まで減少を繰り返す

これらのタスクを設定した RTOS を二台の組込み機器に実装し，評価実験を行う．このとき二台の組込み機器の同期を取り連動するよう動かす．

取得した動作ログは，計算機に転送しテキスト形式のログファイルに書き込まれる．計算機上では，書き込まれたログファイルを動作ログ解析スクリプトに読込ませ，実時間の保証ができなくなる原因の発生時にログを生成する．実時間保証不可原因の発生ログと動作ログはスケジューラ切替え条件ファイル生成スクリプトに読込まれ，切替え条件を生成する．

(3) スケジューラ切替え

実装する RTOS は，生成した切替え条件と四つのスケジューラをまとめてコンパイルして ROM 化する．

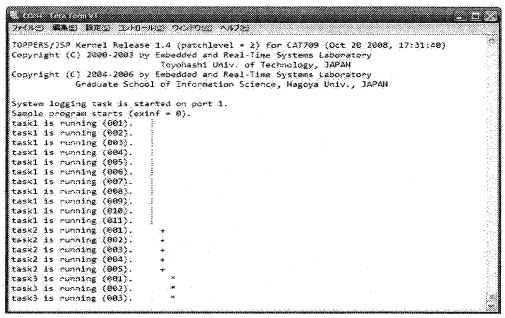

図7　起動画面

ROM 化を行った RTOS は計算機からシリアルポート経由で組込み機器に転送され，起動する（図7）．実行するスケジューラは，実験を個々に行った際，スケジューリング不能になった所で切替え条件により，RTOS が続行可能なスケジューラに切り替わり回避する．

本研究により，動作記録を用いて複数のスケジューラを切り替えるという手法で，実時間を保証したスケジューリングをすることができるようになった．

この手法は，過去の結果からスケジューリングを行うという点ではアダプティブ・スケジューリングと同様だが，タスクの実行時間の見積もりなど，スケジューリングのパラメータを RTOS 実行中に再計算をせずに事前に計測しておくので，その分の処理時間だけ優位性が見込まれる．また，アダプティブ・スケジューリングでは，新しいタスクの起動要求やタスクが終了するたびに動的な優先度を再計算する必要があるが，スケジューラの切替えは実行中のスケジューラの停止、実行中のタスクの状態の保存，スケジューラの切替え，保存したタスクの状態の受け渡し，スケジューラの再開の五つの処理の発生頻度も，実時間保証ができなくなる原因の発生時だけであり少ない．

7. おわりに

本稿では，一般的に用いられるリアルタイムスケジューラが RTOS 実行時にどのようなときに実時間の保証を妨げる原因となる事象を起こすかを検出する実験基盤，ならびにそのシステムを用いて複数のリアルタイムスケジューラを切替えて RTOS を実行する機構の開発について述べた．本研究により，既存のスケジューリングアルゴリズムを用いて，組込みシステムの時間保護を図ることが可能となる．

今後の課題として，切替えを行ったときのリアルタイム性低下原因の情報をフィードバックし，その情報を基に更なる実時間保証性能の向上を図る機構を実装があげられる．

参考文献

[1] 黒井崇史，早川栄一: “リアルタイムスケジューラ実験基盤を用いた OS の実時間性の向上”，第 70 回情報処理学会全国大会講演論文集, Vol.1, pp.117-118 (2008).

[2] The Cheddar project
http:// beru.univ-brest.fr/~singhoff/cheddar/index.html

[3] B. Ahn, J. Kim, D. Lee, and S. Lee: “A Real Time Scheduling Method for Embedded Multimedia Application”, *Proc. of The 2006 International Conference on Pervasive Systems & Computing* (*ICPSC'06*), pp.104-107 (2006).

[4] C. Lu, J. Stankovic, G. Tao, and S. Son: “Feedback Control Real-Time Scheduling: Framework, Modeling, and Algorithms”, Journal of Real-Time Systems, Special Issue on Control-Theoretical Approaches to Real-Time Computing, Vol.23, No.1-2, pp.85-126 (2002).

[5] 栗谷一路，田中清史: “リアルタイム OS における適応型スケジューリング方式”，情報処理学会研究報告，2002-SLDM-107, pp.127-132 (2002).

[6] シリコンリナックス株式会社 CAT709
http:// www.si-linux.co.jp/product/cat709/index.html

[7] TOPPERS/JSP カーネル
http://www.toppers.jp/jsp-kernel.html