情報処理学会研究報告
IPSJ SIG Technical Report

Vol.2010-SE-168 No.1
Vol.2010-EMB-17 No.1
2010/6/1

組み込みシステム向けの高精度な周期実行制御法の評価

古川友樹[†1]　山内利宏[†1]　谷口秀夫[†1]

ART-Linux の周期実行制御法の問題点を明らかにし，この問題を解決するためにタイマ割り込みの発生時刻毎に起動する実時間プロセスをまとめて管理する制御法を提案した．ここでは，提案制御法を評価し，評価結果から提案制御法の有効性を示す．具体的には，実時間プロセスの起動と待機にかかる制御オーバヘッド，および同時に起動する実時間プロセスの数のバラツキについて評価した．評価の結果，起動と待機にかかる処理時間の変動を小さくできること，および待機の処理時間が短く，一定であることを示した．

Evaluation for Sophisticated Periodic Execution Control in Embedded Systems

Yuuki Furukawa,[†1] Toshihiro Yamauchi[†1]
and Hideo Taniguchi[†1]

We clarified several problems of periodic execution control in ART-Linux, and proposed the sophisticated periodic execution control that manages the real-time process scheduled to be released with respect to the time of each timer interrupt. In this paper, we evaluate the proposed control and describe effectiveness of it. In particular, we evaluate the control overhead of the process release and the wait, and the dispersion for the number of real-time processes. As a result of evaluation, the dispersion of the processing time for the process release and the wait becomes small, and the processing time of the wait is short and this value is fixed.

†1 岡山大学大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University

1. はじめに

組み込みシステムは複雑化し，その数は増加している[1)]．これに伴い，組み込み機器の資源を制御するソフトウェアの高性能化が求められている．組み込みシステムでは，実行する処理の種類が限られており，同じ処理を繰り返し実行する場合が多い．例えば，ロボットのモータを制御するため，5ms 間隔で処理を実行する必要がある．このように周期的に実行される処理を抱えるシステムでは，これを制御する周期実行制御について，制御オーバヘッドは小さく，処理時間の変動が小さいことが望まれる．

ロボットで利用されている既存のリアルタイムオペレーティングシステムとして，ART-Linux[2)3)] がある[4)]．ART-Linux には，Linux のアプリケーションやデバイスドライバを利用可能であり，小さなジッタで短い周期の実時間プロセスを周期実行制御できるという特徴がある．また，ART-Linux を利用した高性能化の研究[5)] もなされている．

文献 6) で，ART-Linux の周期実行制御法の問題点を明らかにし，それらの問題点を解決する高精度な周期実行制御法を提案した．提案制御法の特徴は，タイマ割り込みの発生時刻毎に，起動する実時間プロセスをまとめて管理することである．これにより，キュー操作の回数が削減できるため，実時間プロセスの起動と待機における制御オーバヘッドは小さくなり，処理時間の変動も小さくできる．また，同時に起動する実時間プロセスの数を調整し，タイマ割り込みから実時間プロセス実行までの時間の変動を小さくできる．

ここでは，文献 6) で述べた ART-Linux の周期実行制御法の問題点と提案制御法の期待される効果について説明する．次に，提案制御法の有効性を明らかにするために行った評価内容とその結果について述べる．具体的には，実時間プロセスの起動と待機における処理時間，および同時に起動する実時間プロセスの数のバラツキについて評価した．

2. ART-Linux の問題点と高精度な周期実行制御法の期待される効果

文献 6) で述べた ART-Linux の問題点と高精度な周期実行制御法の期待される効果について，**図 1** にまとめ，以下に説明する．

ART-Linux は，1 つのキューにより待機状態の実時間プロセスを管理する．ART-Linux では，実時間プロセスの起動処理において，WAIT キューの先頭に存在する実時間プロセスの起動待ち時間を計算し，起動を判定する．このとき，タイマ割り込みが発生した時刻に起動する全ての実時間プロセスを WAIT キューから削除し，READY キューへ接続する．このため， 実時間プロセスの起動において，制御オーバヘッドは同時に起動する実時間プ

Vol.2010-SE-168 No.1
Vol.2010-EMB-17 No.1
2010/6/1

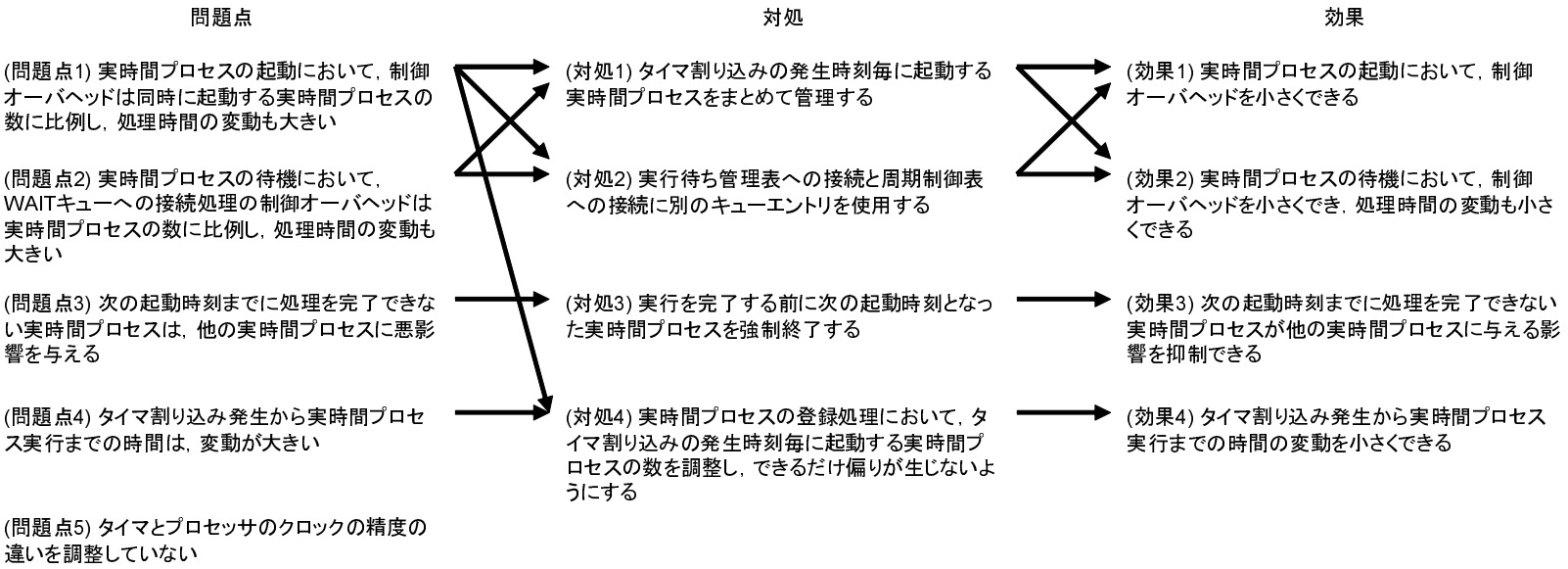

図 1 ART-Linux の問題点と高精度な周期実行制御法の期待される効果

ロセスの数に比例し，処理時間の変動も大きい（問題点 1）．したがって，タイマ割り込み発生時刻において，起動する実時間プロセスの数が多いと実時間プロセスの起動における処理時間は長い．

また，待機処理において，WAIT キューを先頭から探索し，起動待ち時間を計算し，待機を要求した実時間プロセスを接続する位置を決定する．このため，WAIT キューの長さに比例した処理が必要になり，実時間プロセスの待機における制御オーバヘッドは，WAIT キューに存在する実時間プロセスの数と各実時間プロセスの起動待ち時間により変動する．したがって，実時間プロセスの待機における WAIT キューへの接続処理の制御オーバヘッドは実時間プロセスの数に比例し，処理時間の変動も大きい（問題点 2）．例えば，待機を要求した実時間プロセスより先に起動する実時間プロセスの数が多いと，実時間プロセスの待機における処理時間は長い．

このように，ART-Linux の周期実行制御法において，実時間プロセスの起動と待機における制御オーバヘッドは，待機状態の実時間プロセスの数や各実時間プロセスの周期，または起動待ち時間の影響を受け，処理時間の変動も大きい．これは，実時間プロセスに起動待ち時間を保持させ，1 つのキューにより待機状態の実時間プロセスを管理するためである．

これに対して，提案した高精度な周期実行制御法では，タイマ割り込みの発生時刻毎に起動する実時間プロセスをまとめて管理する（対処 1）．また，起動要素にキューエントリを 2 つ持たせ，実行待ち管理表への接続と周期制御表への接続に別のキューエントリを使用する（対処 2）．このため，提案制御法では，実時間プロセスの起動において，起動待ち時間の計算と WAIT キューからの削除処理がなくなり，制御オーバヘッドを小さくできる．さらに，同時に起動する実時間プロセスの数が影響する処理が削減され，起動における処理時間の変動も小さくなる．

また，提案制御法では，実時間プロセスの待機処理において，WAIT キューへの接続処理が不要である．これは，上記で述べたように，実時間プロセスの起動処理において，WAIT

Vol.2010-SE-168 No.1
Vol.2010-EMB-17 No.1
2010/6/1

キューからの削除処理がないためである．つまり，提案制御法における実時間プロセスの待機処理は，実行待ち管理表から起動要素を削除するのみとなる．このため，実時間プロセスの待機における処理時間は，WAIT キューに存在する実時間プロセスの数や各実時間プロセス起動待ち時間に影響を受けず，一定となる．したがって，実時間プロセスの待機において，制御オーバヘッドを小さくでき，処理時間の変動も小さくできる．

（問題点 3）として，次の起動時刻までに処理を完了できない実時間プロセスは，他の実時間プロセスに悪影響を与えることがある．周期実行制御において，次の起動時刻までに処理を完了できない実時間プロセスが存在する場合が考えられる．このような実時間プロセスが存在すると，他の実時間プロセスの実行間隔が変化し，周期実行の精度は低下する．ART-Linux では，実行待ち，または実行中の実時間プロセスが次の起動時刻までに処理を完了できない場合でも，この実時間プロセスは継続して周期実行制御される．一方，提案制御法では，実時間プロセスの起動処理において，実行を完了する前に次の起動時刻となった実時間プロセスを強制終了する（対処 3）．具体的には，起動する実時間プロセスの起動要素が既に実行待ち管理表に存在する場合，実行待ち管理表と周期制御表からこの実時間プロセスの起動要素を削除し，プロセスを強制終了する．つまり，次の起動時刻までに処理を完了できない実時間プロセスを強制終了する．このため，次の起動時刻までに処理を完了できない実時間プロセスが他の実時間プロセスに与える影響を抑制できる．

文献 6) では，ART-Linux の性能測定の結果より，(問題点 4）と（問題点 5）を明らかにした．

（問題点 4）として，タイマ割り込み発生から実時間プロセス実行までの時間の変動が大きいことがある．このような時間の変動が大きいと，周期実行の精度は低下する．この要因の 1 つとして，タイマ割り込み毎に起動する実時間プロセスの数に偏りがあると，実時間プロセスの起動における処理時間の変動は大きいことがある．そこで，提案制御法では，実時間プロセスの登録処理において，タイマ割り込みの発生時刻毎に起動する実時間プロセスの数を調整し，できるだけ偏りが生じないようにする（対処 4）．これにより，実時間プロセスの起動における処理時間の変動を抑制でき，タイマ割り込み発生から実時間プロセス実行までの時間の変動を小さくできる．

（問題点 5）として，タイマとプロセッサのクロックの精度の違いを調整していないことがある．タイマクロックの精度とプロセッサクロックの精度に違いがあるため，周期に比例して周期実行の間隔の誤差が増加する．周期と周期実行の間隔の差が大きいと，周期実行の精度は低下する．文献 6) では，この問題点について対処していない．

3. 評　　価

3.1 評価項目と目的

提案制御法を ***AnT*** オペレーティングシステム（以降，***AnT***）[7] に実現し，評価した．

ART-Linux の周期実行制御法において，実時間プロセスの起動と待機における制御オーバヘッドは，WAIT キューに存在する実時間プロセスの数と各実時間プロセスの起動待ち時間に影響を受け，処理時間の変動も大きい．一方，提案制御法では，2 章で述べたように，上記の制御オーバヘッドを小さくでき，処理時間の変動も小さくできる．そこで，実時間プロセスの数が多い場合の実時間プロセスの起動と待機にかかる処理時間を測定し，ART-Linux と ***AnT*** で比較することにより，提案制御法の期待される効果を明確にする．

また，タイマ割り込み周期より長い周期を持つ複数の実時間プロセスが存在する場合において，同時に起動する実時間プロセスの数のバラツキを比較する．これにより，同時に起動する実時間プロセスの数について，できるだけ偏りが生じないようにすることで，提案制御法が実時間プロセスの起動にかかる処理時間の変動を小さくできることを示す．

評価項目を以下にまとめる．

(1)　同時に起動する実時間プロセスの数を増加させた場合の起動処理時間
同時に起動する実時間プロセスの数が多い場合において，実時間プロセスの起動にかかる処理時間を測定する．これにより，(効果 1）を明確にする．

(2)　待機状態の実時間プロセスの数を増加させた場合の待機処理時間
待機を要求した実時間プロセスより先に起動する実時間プロセスの数が多い場合において，実時間プロセスの待機にかかる処理時間を測定する．これにより，(効果 2）を明確にする．

(3)　起動する実時間プロセス数のバラツキ
タイマ割り込み周期より長い周期を持つ複数の実時間プロセスが存在する場合において，タイマ割り込みの発生時刻毎に起動する実時間プロセス数を測定する．また，このときの起動処理時間の変動を確認する．これにより，(効果 4）を明確にする．

周期実行制御において，次の起動時刻までに処理を完了できない実時間プロセスは，他の実時間プロセスに悪影響を与える．ART-Linux では，実時間プロセスが次の起動時刻までに処理を完了できない場合でも，この実時間プロセスは継続して周期実行制御される．一方，提案制御法では，次の起動時刻までに処理を完了できない実時間プロセスを全て強制終了する．このため，以降の周期実行制御において，次の起動時刻までに処理を完了できない実時間プロセスが他プロセスに与える影響はないと考える．したがって，ここでは，(効果

Vol.2010-SE-168 No.1
Vol.2010-EMB-17 No.1
2010/6/1

表 1　測定環境

CPU	Intel Pentium II 400MHz
メモリ	96MB
タイマ割り込みの周期	1ms

3）について，定量的な評価は行わない．

測定は**表 1** の環境で行い，時刻の取得には RDTSC 命令を用いた．また，処理時間の測定の際，キャッシュの影響を排除するため，1MB のデータを読み込む非実時間プロセスを動作させた．なお，***AnT*** と ART-Linux 上で動作する実時間プロセスの数は同じである．

3.2　同時に起動する実時間プロセスの数を増加させた場合の起動処理時間

実時間プロセスの起動における制御オーバヘッドを比較するため，N 個の実時間プロセスの起動にかかる処理時間を測定した．これにより，(効果 1）を評価する．具体的には，周期 1ms の実時間プロセスを N 個登録し，タイマ割り込み毎の実時間プロセスの起動処理にかかる時間を測定した．つまり，タイマ割り込み発生時において，実時間プロセスの起動処理の開始から N 個の実時間プロセスを全て実行待ちにするまでの時間を測定した．これは，同時に起動する実時間プロセスの数が N 個の場合の起動処理時間である．登録する実時間プロセスの数を 1 個から 100 個の範囲で変化させ，それぞれの場合で処理時間を求めた．結果を**図 2** に示す．

図 2 より，***AnT*** の起動処理時間は ART-Linux と同等であることがわかる．ART-Linux では，実時間プロセスの起動において，起動する全ての実時間プロセスに対し，起動待ち時間の計算，READY キューからの削除，および WAIT キューへの接続を行う．一方，提案制御法では，カレントエントリを更新し，カレントエントリに存在する全ての起動要素を実行待ち管理表へ接続する．このため，起動待ち時間の計算とキューからの削除処理がなくなり，実時間プロセスの起動における処理時間は短くなると推察した．また，同時に起動する実時間プロセスの数が影響する処理が削減されるため，実時間プロセスの数の増加に対する起動処理時間の増加量も少なくなり，起動処理時間の変動も小さくできると推察した．しかし，図 2 では，***AnT*** と ART-Linux の起動処理時間の間に差がなく，実時間プロセスの数の増加に対する起動処理時間の増加量も同等である．

この要因の 1 つとして，ART-Linux と比較して提案制御法 (***AnT***) のデータの空間的局所性が低いことがある．ART-Linux では，1 個の実時間プロセスに対し 1 個の構造体を用いる．これに対して，提案制御法では，1 個の実時間プロセスに対し複数の起動要素を用いる場合がある．この対処として，文献 6) では，周期や優先度といった実時間プロセスの情報は起動要素ではなく，別の管理表（実時間情報管理表）が保持することにした．これは，起動要素の保持するデータ量が増えると，キャッシュミスを発生する確率が高くなる可能性があるためである．

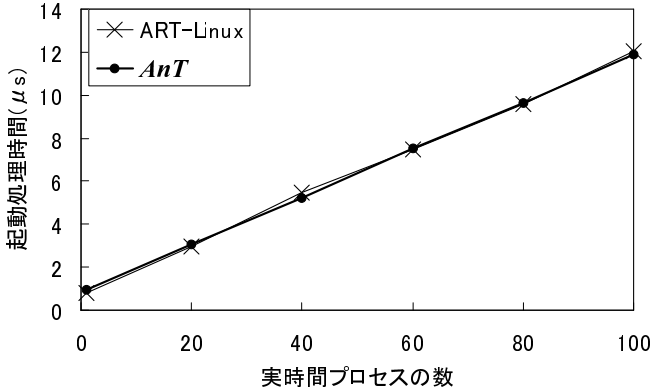

図 2　同時に起動する実時間プロセスの数を増加させた場合の起動処理時間

実時間プロセスの数が 1 である場合，***AnT*** の起動処理時間は，ART-Linux よりも長い．また，実時間プロセスの数が増加すると，***AnT*** と ART-Linux の起動処理時間の間に差はなくなり，わずかな差ではあるが，***AnT*** の起動処理時間が ART-Linux より短くなる場合も存在する．このことから，***AnT*** のデータの空間的局所性が低いため，提案制御法の効果が相対的に小さくなり，***AnT*** の起動処理時間は ART-Linux と同等になったと考える．

3.3　待機状態の実時間プロセスの数を増加させた場合の待機処理時間

実時間プロセスの待機における制御オーバヘッドを比較するため，1 個の実時間プロセスの待機にかかる処理時間を測定した．これにより，(効果 2）を評価する．具体的には，周期 1ms の実時間プロセスを N 個登録し，タイマ割り込みから N 番目に実行された実時間プロセスが待機を要求してから，待機状態となるまでの時間を測定した．この測定において，提案制御法では，周期制御表の全てのエントリに N 個の起動要素が存在する状態となる．また，ART-Linux では，待機処理時において，同じ周期を持つ待機状態の実時間プロセスが $N-1$ 個存在する状態であり，WAIT キューの実時間プロセスを $N-1$ 個探索する必要

Vol.2010-SE-168 No.1
Vol.2010-EMB-17 No.1
2010/6/1

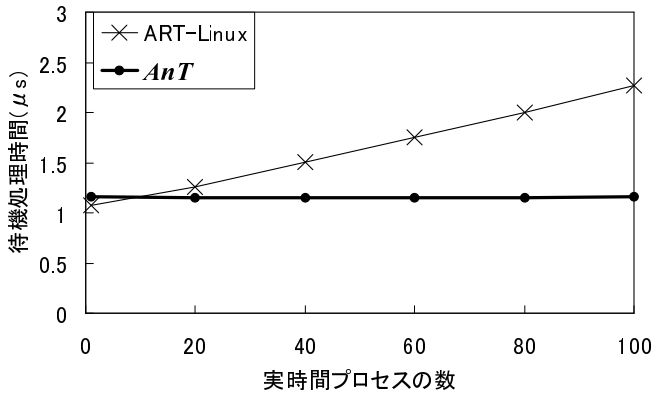

図 3 待機状態の実時間プロセスの数を増加させた場合の待機処理時間

がある．つまり，測定結果は，待機を要求した実時間プロセスより先に起動する実時間プロセスが $N-1$ 個存在する場合の待機処理時間である．登録する実時間プロセスの数を 1 個から 100 個の範囲で変化させ，それぞれの場合で処理時間を求めた．結果を**図 3** に示す．

図 3 より，***AnT*** の待機処理時間は短く，一定であることがわかる．一方，ART-Linux は，実時間プロセスの数の増加に比例して，待機処理時間も増加することがわかる．ART-Linux では，待機処理において，WAIT キューの探索と起動待ち時間の計算，READY キューからの削除，および WAIT キューへの接続を行う．このため，ART-Linux の待機処理時間は，待機状態の実時間プロセスの数と各実時間プロセスの起動待ち時間に影響を受け，待機を要求した実時間プロセスより先に起動する実時間プロセスの数に比例し，増加する．一方，提案制御法では，実行待ち管理表から起動要素を削除するのみである．このため，***AnT*** では，待機状態の実時間プロセスの数に影響を受けず，待機処理時間は一定である．

また，ART-Linux では，周期の異なる複数の実時間プロセスが存在する場合，待機処理時間の変動が大きいと推察される．これに対して，提案制御法では，待機状態の実時間プロセスの数に影響を受けないため，待機処理時間は変動しない．したがって，実時間プロセスの数が多い場合，または周期の異なる複数の実時間プロセスが存在する場合，提案制御法の効果は大きいといえる．

実時間プロセスの数が少ない場合，***AnT*** の待機処理時間は，ART-Linux よりも長い．この要因として，3.2 節で述べたように，ART-Linux と比較して提案制御法 (***AnT***) のデータの空間的局所性が低いことがある．ART-Linux では，周期や優先度といった情報を実時間プロセスに保持させる．このため，待機処理において，提案制御法における実時間情報管理表のような管理表を参照する必要がない．一方，提案制御法では，実行待ち管理表から起動要素を削除し，実時間プロセスを待機状態にするため，実時間情報管理表を参照する．しかし，ART-Linux の待機処理時間が ***AnT*** より短い場合でも，待機処理時間の差は 0.1μs 以下と小さく，***AnT*** の待機処理時間は一定である．このため，実時間プロセスの数が少ない場合でも，提案制御法は有効だと考えられる．

また，周期実行制御において，タイマ割り込みから N 番目の実時間プロセスが実行されるまでに $N-1$ 個の実時間プロセスが待機を要求する．したがって，N 番目の実時間プロセスの実行までにかかる待機の処理時間は，1 番目から $N-1$ 番目までの実時間プロセスの待機処理時間の合計となる．このため，後に実行される実時間プロセスほど，待機処理時間による影響を受ける．***AnT*** と ART-Linux の待機処理時間の合計値の差分より，***AnT*** では，100 番目に実行される実時間プロセスの実行までの制御にかかる処理時間を約 50μs 短縮できることがわかる．これは，周期 1ms に対し，約 5%の時間である．

3.4 起動する実時間プロセス数のバラツキ

周期 100ms の実時間プロセスを 100 個登録し，各タイマ割り込みの発生時刻において，起動する実時間プロセスの数と起動処理時間を測定した．また，プロセスの生成から実時間プロセスへの登録までの間にプロセスを $x$ms 停止させた．x は 0 から 99 までの範囲の乱数であり，乱数の生成には標準 C ライブラリの rand() 関数を使用した．これにより，実時間プロセスの登録が要求されるタイミングが不規則になるようにした．この測定における周期制御表のエントリ数は 100 である．これにより，(効果 4) を評価する．各タイマ割り込みの発生時刻において，***AnT*** の起動する実時間プロセス数を**図 4** に示し，ART-Linux の起動する実時間プロセス数を**図 5** に示す．また，このときの起動処理時間の最大値，平均値，および最小値を**表 2** にまとめる．なお，上記の測定を 10 回行い，測定結果の傾向に変化がないことを確認している．

図 4 より，***AnT*** における起動する実時間プロセス数は 3 か 4 であることがわかる．これは，提案制御法において，周期制御表に起動要素を接続する際，管理する起動要素の少ないエントリを探索し，発見したエントリに起動要素を接続するためである．これにより，周期制御表の各エントリに接続する起動要素の数について，偏りが生じることが少なくなる．つまり，***AnT*** における起動する実時間プロセス数の偏りは小さい．また，周期制御表を探

Vol.2010-SE-168 No.1
Vol.2010-EMB-17 No.1
2010/6/1

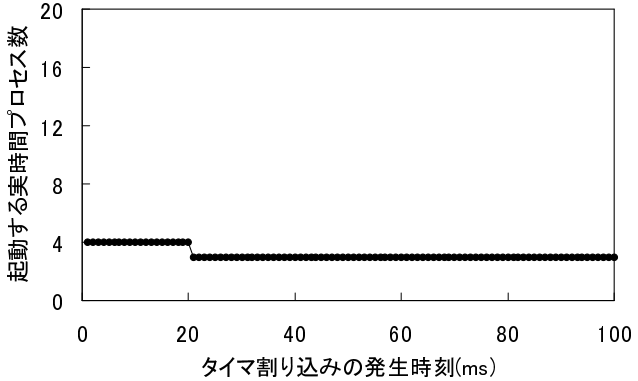

図 4 タイマ割り込みの発生時刻毎の起動する実時間プロセス数（***AnT***）

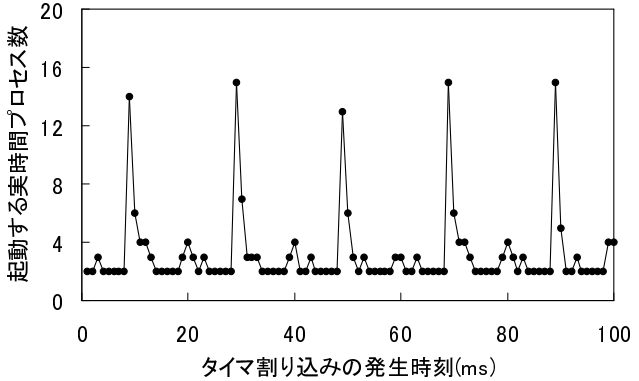

図 5 タイマ割り込みの発生時刻毎の起動する実時間プロセス数（ART-Linux）

索し，同時に起動する実時間プロセスの数を調整しているため，***AnT*** の結果は，同じ測定を繰り返し行った場合でも変化することはない．

これに対し，図 5 より，ART-Linux における起動する実時間プロセス数の偏りは大きいことがわかる．これは，ART-Linux において，同時に起動する実時間プロセスの数を調整する機能がないためである．ART-Linux では，実時間プロセスが登録されてから最初に待機を要求したとき，この実時間プロセスは設定された周期と同じ時間待機する．この実時間プロセスの次の起動において，同時に起動する実時間プロセスの数は，待機処理時に WAIT キューに存在する実時間プロセスの数や各実時間プロセスの起動待ち時間により決まる．つまり，タイマ割り込みの発生時刻毎に，起動する実時間プロセスの数が偏ることがある．したがって，ART-Linux の結果は，同じ測定を繰り返し行った場合，必ず変化する．つまり，タイマ割り込み周期よりも長い周期を持つ実時間プロセスを不規則なタイミングで複数登録した場合，ART-Linux において，起動する実時間プロセス数の偏りは大きい．

表 2 より，ART-Linux は最大値と平均値の間に約 1.8μs の差が存在するのに対し，***AnT*** では約 0.4μs の差であることがわかる．つまり，ART-Linux に比べ，***AnT*** の起動処理時間の変動は小さい．したがって，提案制御法は，タイマ割り込み発生から実時間プロセス実行までの時間の変動を小さくできる．

また，実際の実時間プロセスへの登録要求は，不規則に存在するのではなく，特定の時刻

表 2 周期 100ms の実時間プロセスを 100 個登録した場合の起動処理時間

	AnT	ART-Linux
最大値（μs）	1.50	2.42
平均値（μs）	1.15	0.66
最小値（μs）	0.95	0.43

にまとまって存在すると考えられる．つまり，タイマ割り込み毎の起動する実時間プロセス数の偏りはこの測定結果よりも大きく，提案制御法の効果は大きいと推察される．

4. おわりに

ART-Linux の問題点と提案した高精度な周期実行制御法の期待される効果について説明し，提案制御法の評価内容と結果について述べた．評価として，同時に起動する実時間プロセスの数を増加させた場合の起動処理時間，待機状態の実時間プロセスの数を増加させた場合の待機処理時間，および起動する実時間プロセス数のバラツキを測定した．

評価結果から，提案制御法について，以下のことを明らかにした．起動処理時間は ART-Linux と同等であり，実時間プロセスの数が 100 である場合，約 12μs である．また，待機処理時間は約 1.2μs と短く，一定である．さらに，起動する実時間プロセス数の偏りは小さ

Vol.2010-SE-168 No.1
Vol.2010-EMB-17 No.1
2010/6/1

く，起動処理時間の変動も小さい．

提案制御法は，実時間プロセスの数が多い場合や周期の異なる複数の実時間プロセスが存在する場合，またはタイマ割り込み周期より長い周期を持つ複数の実時間プロセスが存在する場合に効果が大きい．

残された課題として，タイマとプロセッサのクロックの精度の違いを調整していないことについての検討がある．

謝辞 本研究の一部は，科学研究費補助金（課題番号 21500055）による．

参 考 文 献

1) Buttazzo, G.: Research trends in real-time computing for embedded systems, ACM SIGBED Review, Vol.3, Issue 3 pp.1–10, (2006).
2) 石綿陽一: SMP カーネルに基づく ART-Linux の安定化と実時間処理性能の測定, 第 3 回計測自動制御学会システムインテグレーション部門講演会 論文集, (2002).
3) 石綿陽一: SH-4 プロセッサ上の ART-Linux の開発とその品質管理への応用, 第 3 回計測自動制御学会システムインテグレーション部門講演会 論文集, (2002).
4) Yokoi, K., Kanehiro, F., Kaneko, K., Kajita, S., Fujiwara, K. and Hirukawa, H.: Experimental Study of Humanoid Robot HRP-1S, Intl. J. Robotics Research, Vol.23, No.4-5, pp.351–362, (2004).
5) 堀洋平, 中島俊夫, 片下敏宏, 関山守, 戸田賢二: 専用ハードウェアによる ART-Linux の高性能化に向けて, 情報処理学会研究報告, 2004-SLDM-119, Vol.2005, No.27, pp. 109–114, (2005).
6) 古川友樹, 田端利宏, 谷口秀夫: 組み込みシステム向けの高精度な周期実行制御法の設計, 電子情報通信学会技術研究報告, Vol.109, pp.523–528, (2010).
7) 谷口秀夫, 乃村能成, 田端利宏, 安達俊光, 野村裕佑, 梅本昌典, 仁科匡人: 適応性と堅牢性をあわせ持つ ***AnT*** オペレーティングシステム, 情報処理学会研究報告, 2006-OS-103, Vol.2006, No.86, pp.71–78, (2006).