情報処理学会論文誌 Vol.57 No.8 1765–1774 (Aug. 2016)

GNSSを用いた時刻同期機能を有するリアルタイムOS

松原 彩音[1,a)] 俞 明連[1,b)] 横山 孝典[1,c)]

受付日 2015年11月19日, 採録日 2016年5月17日

概要：本論文では，無線ネットワークや広域ネットワークにより接続された疎結合分散型のサイバーフィジカルシステム（CPS）のための，GNSS（Global Navigation Satellite Systems，全地球航法衛星システム）を用いた時刻同期機能を有するリアルタイム OS（RTOS）を提案する．分散型組み込み制御システムのようなハードリアルタイムな CPS において，同一物理時間に従う実世界の制御対象を複数のコンピュータにより正確に制御するには，それらコンピュータが同一のシステム時刻に基づいて同期して処理を行うことが望ましい．そこで我々は，GNSS により提供される協定世界時（UTC）に同期したシステム時刻に従ってアプリケーションのタスクをスケジューリングし，実行できる RTOS を開発した．本 RTOS は，GNSS モジュールの出力信号を利用したティック周期補正機構，ティック位相補正機構，システム時刻補正機構を有する．疎結合分散システムを構成する各コンピュータに本 RTOS を搭載することで，それらコンピュータ上のタスクを高精度で同期したシステム時刻に基づいてスケジューリングでき，実世界に存在する対象物を同期して制御することが可能になる．

キーワード：サイバーフィジカルシステム，リアルタイム OS，時刻同期，分散制御システム，GNSS

A Real-time Operating System with GNSS-based Tick Synchronization

Ayane Matsubara[1,a)] Myungryun Yoo[1,b)] Takanori Yokoyama[1,c)]

Received: November 19, 2015, Accepted: May 17, 2016

Abstract: The paper presents a real-time operating system (RTOS) with GNSS (Global Navigation Satellite Systems)-based tick synchronization for loosely coupled distributed cyber-physical systems (CPSs) such as distributed control systems connected with wireless networks or wide-area networks. In CPS, computations directly interact with the physical world with the universal physical time. A precisely synchronized system time is required for high performance control of the physical world. We have developed a RTOS that manages application tasks based on a system time, which is precisely synchronized with UTC (Coordinated Universal Time) provided by GNSS. The RTOS has mechanisms for tick rate compensation, tick phase compensation and system time compensation, which utilize GNSS receiver signals. Embedded computers with the RTOS can synchronously execute application tasks according to the precisely synchronized system time. The precise synchronous execution improves the performance of the control of the physical world.

Keywords: cyber-physical systems, real-time operating systems, system time synchronization, distributed control systems, GNSS

1. はじめに

サイバーフィジカルシステム（CPS, Cyber-Physical Systems）は物理時間に従う実世界と直接相互作用するため，時間を抽象化して扱うことができず，実時間に基づいた処理が要求される [1]．自動車制御システム（automotive control systems），マルチビークルシステム（multi-vehicle control systems），アビオニクス装置（avionics systems）のような組み込み制御システムはハードリアルタイムな CPS で，制御処理における遅延やジッタは制御性能低下の原因となる [2]．分散型の組み込み制御システムでは，実世界に存在する単一の制御対象あるいは強い依存関係がある複数

[1] 東京都市大学
Tokyo City University, Setagaya, Tokyo 158–8557, Japan
[a)] g1581523@tcu.ac.jp
[b)] myoo@tcu.ac.jp
[c)] tyoko@tcu.ac.jp

© 2016 Information Processing Society of Japan

の制御対象を複数のコンピュータにより制御することが多い．したがって，高性能な制御を実現するには，複数のコンピュータが高精度で同期したシステム時刻に基づいて処理を行うことが望ましい．組み込み制御システムでは，リアルタイム OS（RTOS）のシステム時刻に従ってタスクのスケジューリングを行うが，各コンピュータ上のタスクを高精度に同期して実行するには，システム時刻の値のみでなく，ティックの周期や位相についても高い精度で一致させる必要がある．

世の中の多くの分散システムでは，各コンピュータのシステム時刻は同期していないか，疎結合な形の同期を行っているのが普通である．疎結合な形での同期機構としては，NTP（Network Time Protocol）[3] が広く用いられている．しかし，システム時刻の値のみでなく，ティックの周期や位相をも高精度で一致させるには十分ではなく，本論文が対象としているハードリアルタイムな CPS にそのまま適用することは困難である．

組み込み制御システム向けのクロック同期機能を持つものとして，時間駆動アーキテクチャ（time-triggered architecture）が提案されている [4]．また OSEK/VDX は，OSEKtime と呼ばれる時間駆動 OS の仕様を策定している [5]．知場らは，時間駆動ネットワークの 1 つである FlexRay [7] のクロック同期機能を用いて，同期したシステム時刻を持つ分散 RTOS を提案している [6]．しかしこれらのシステムは有線ネットワークを用いてクロック同期を行っており，今後増えると予測される，無線ネットワークや広域ネットワークにより接続された疎結合分散型の CPS には適用できない．このため，有線ネットワークを必要とせずにシステム時刻を高精度に同期する手法が求められている．

正確な時刻の伝達方法として GPS（Global Positioning System）や GLONASS（Global Navigation Satellite System）のような GNSS（Global Navigation Satellite Systems，全地航法衛星システム）が有用であるといわれており [8], NTP サーバにも用いられている．各計算機に GNSS モジュールを搭載し，GNSS が提供する協定世界時（UTC, Coordinated Universal Time）にシステム時刻を同期させれば，有線ネットワークを使用せずに高精度の時刻同期を実現できる可能性がある．

Kim らは，広域分散処理システムを対象とした TMOSM（TMO（Time-triggered Message-triggered Object）Support Middleware）と呼ばれるミドルウェアを開発している．そして，そのミドルウェア向けに GPS を用いた時刻同期モジュールを提案している [9]．しかし，TMOSM は OS 上で動作するミドルウェアであり，リアルタイムタスクのスケジューリングに直接利用することはできない．また Quesada らは，GPS モジュールが出力する 1 秒周期の PPS（Pulse Per Second）信号を用いてクロック同期を実現する手法を提案し，評価を行っている [10]．しかしこれまでのところ，ティックの周期や位相を含むシステム時刻の同期に GNSS を利用した RTOS の研究は見当たらない．

そこで本研究の目的は，無線ネットワークや広域ネットワークにより接続された疎結合分散型 CPS のための，GNSS を用いた時刻同期機能を有する RTOS を提供することである．本論文では，GNSS により提供される協定世界時に同期したシステム時刻に従ってタスクのスケジューリングが可能な RTOS を提案する．本 RTOS の時刻同期機能は，ティック周期，ティック位相，およびシステム時刻の値を補正することで実現する．疎結合分散型 CPS を構成する各コンピュータに本 RTOS を搭載することで，それらコンピュータ上のタスクを同一のシステム時刻に基づいてスケジューリングでき，実世界に存在する対象物を同期して制御することが可能になる．

我々は，TOPPERS プロジェクト [11] により開発された TOPPERS/ATK1 と呼ばれる RTOS を拡張して，時刻同期機能を有する RTOS を実装する．TOPPERS/ATK1 は，自動車制御分野の標準である OSEK OS 仕様 [12] に基づいた，オープンソースの RTOS である．具体的には，GNSS モジュールの出力信号を用いたティック周期補正機構，ティック位相補正機構，システム時刻補正機構を TOPPERS/ATK1 に追加することで，時刻同期機能を実現する．本 RTOS を用いることで，システム全体で同期してタスクをスケジューリングできる．またこれにより，Phase-Modification Protocol [13] のような，グローバル時刻ベースのスケジューリングアルゴリズムを採用することも可能になる．

本論文の構成は以下のとおりである．まず 2 章で開発方針を述べる．次に 3 章で GNSS を用いた時刻同期方法について説明し，4 章で提案した時刻同期機構を持つ RTOS の機能と実装について述べる．そして 5 章で開発した RTOS の評価を行い，6 章でまとめを述べる．

2. 開発方針

図 1 は，本論文で提案する RTOS を搭載した複数の組み込みコンピュータ（Embedded Computer）が，GNSS により同期したシステム時刻（GNSS-Based Synchronized System Time）に従って，実世界（Physical World）の制御対象（Physical Object）を制御するようすを描いている．各組み込みコンピュータ上の RTOS のシステム時刻は，GNSS モジュールの出力信号に同期している．**図 2** は，CPS を構成する各組み込みコンピュータにおけるシステム時刻（System Time）が同期しているようすを示したものである．横軸は物理時間（Physical Time）である．各組み込みコンピュータ上の RTOS は，同期したシステム時刻に基づいてタスクをスケジューリングし，実行する．

制御処理における遅延やジッタは制御性能低下の原因と

© 2016 Information Processing Society of Japan

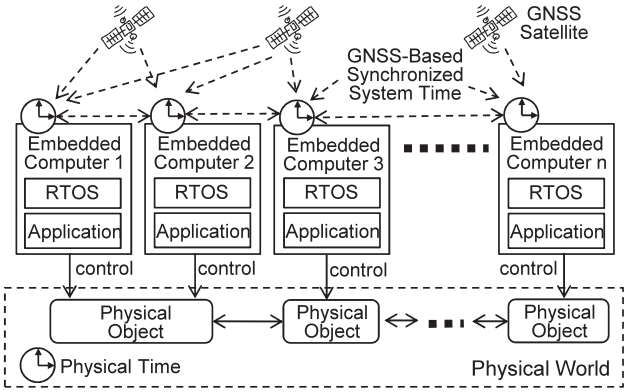

図 1 GNSS を用いたシステム時刻同期機能を利用したサイバーフィジカルシステム

Fig. 1 Cyber-Physical System with GNSS-Based Synchronized System Time.

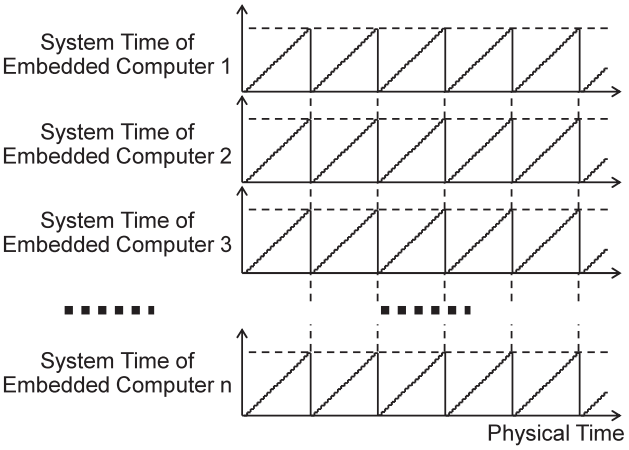

図 2 サイバーフィジカルシステムにおける同期したシステム時刻

Fig. 2 Synchronized System Time in Cyber-Physical System.

なるため，各組み込みコンピュータで実行する入力，出力，計算等の処理は同一時刻に基づいて実行すべきである．また，システム全体としては，タスク以外に，割込み処理時間，入出力ハードウェアの遅延時間，通信時間等も考慮する必要がある．しかし本論文では，タスクを対象とし，すべての組み込みコンピュータが同一時刻に基づいてタスクスケジューリングが可能な RTOS の実現を目的とする．具体的には，世界協定時（UTC）に同期した GNSS モジュールの出力信号を利用して，RTOS のシステム時刻を同期させる．

代表的な組み込み制御システムとして自動車制御システムがある．これまでは自動車単体の制御が中心であったが，短い車間距離で列車のように走行するプラトーン走行等車車間通信を用いた複数自動車の制御や，路車間通信を利用した自動車を含む高度道路交通システムが重要になると考えられ，それらを主な対象として時刻同期機能を持つ RTOS を開発する．しかしそれらシステムの多くはまだ研究段階で，要求される時刻同期性能は明らかになっていない．そこで本研究では，それらの中心となるのは自動車の走行制御であることから，自動車パワートレイン制御と同等の時刻同期性能が要求されるシステムを想定し，自動車制御分野の標準的な OS である OSEK OS をベースに開発を進める．

なお自動車制御分野では，よりジッタが少ないタスク動作が要求されるアプリケーション向けの時間駆動 OS として OSEKtime 仕様が提案されている．OSEK OS 仕様では，論理的なスケジューリング動作（logical level for scheduling acitivities）よりも割込み処理の方が優先度が高いため，割込み処理がタスクの動作に影響を与えることがある．これに対し，OSEKtime のタスク（時間駆動タスク）は，OSEK OS の割込み処理よりも優先度を高くする規定になっており，割込み処理の影響を受けないタスク動作が要求されるアプリケーションの場合は OSEKtime が有効である．しかし，現在のところ OSEKtime 仕様に基づく RTOS が広く使用されるには至っていないため，本研究ではまず OSEK OS ベースの RTOS を開発する．

システム時刻はタスクのスケジューリングに用いられるため，スケジューリング性能に影響を及ぼさない程度の精度で同期する必要がある．そこで本研究では，タスクスケジューリングや周期タスクを実現する RTOS の機構のオーバヘッドと同程度あるいはそれ以下の誤差でシステム時刻の同期を実現することを目標とする．

GNSS モジュールは十分な GNSS 衛星からの電波を受信できる環境でのみ PPS 信号を出力し，そうでなくなると出力は停止する．このため本 RTOS では，PPS 信号が出力されている間のみ時刻同期処理を行うこととする．出力が停止している間は時刻同期処理を行わず，出力が再開されると同期処理も再開する．なお本 RTOS は，通常は十分な GNSS 衛星からの電波を受信でき，電波を受信できないのは一時的であるような環境で動作するシステムを対象とする．

ただし GNSS モジュールからの信号が停止した場合に，RTOS 自体の動作も停止することは許されないのが普通である．たとえばプラトーン走行において，時刻同期がなされず高精度の制御が困難になった場合でも，車間距離を広くとるように制御を変更する等して，安全に走行を続ける必要がある．そこで，RTOS のシステム時刻が同期状態か非同期状態かをアプリケーションが知るための API や，同期状態から非同期状態に遷移したことをアプリケーションに知らせる API を提供する．

実装にあたっては，組み込み制御システムのコスト要求が厳しいことを考慮し，時刻同期のための特殊なハードウェアを追加実装することなく，汎用のプロセッサおよび GNSS モジュールのみで構成することを前提とする．また，なるべく広範囲の RTOS に適用可能な方式とするため，RTOS のソフトウェア構成や仕様をできるだけ変更しないで実装することを目指す．

© 2016 Information Processing Society of Japan

3. GNSS を用いた時刻同期

3.1 システム時刻

一般に，RTOS のシステム時刻はソフトウェアタイマとして実装される．**図 3** は一般的な RTOS におけるティック（Tick）とシステム時刻（System Time）を表したものである．ティックは RTOS における時間の単位で，ティックごとに発生する割込み処理によりシステム時刻の値が更新される．本論文では，この割込み処理をティック割込み（Tick Interrupt）と呼ぶことにする．Tmax はシステム時刻の最大値で，システム時刻が Tmax になると，次のティック割込みで 0 にリセットされる．

ティック割込みはハードウェアタイマからの割込み信号で起動される．**図 4** は，ハードウェアタイマとシステム時刻の関係を表したものである．ティック割込みを発生させるハードウェアタイマにはコンペアマッチタイマが用いられるのが普通である．ハードウェアタイマはクロック信号（Clock Signal for Hardware Timer）をカウントし，その値がコンペアマッチ値（Compare Match Value）に一致すると，割込みを発生させるとともに，値をリセットする．設定するコンペアマッチ値を変えることで，ティックタイムを変更することができる．

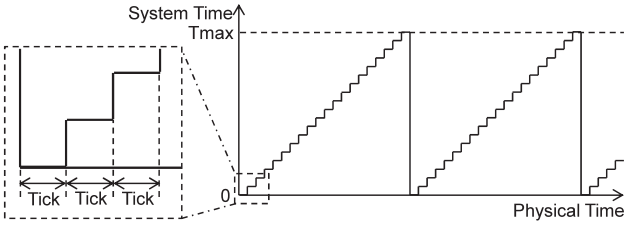

図 3 RTOS におけるティックとシステム時刻
Fig. 3 Tick and system time of RTOS.

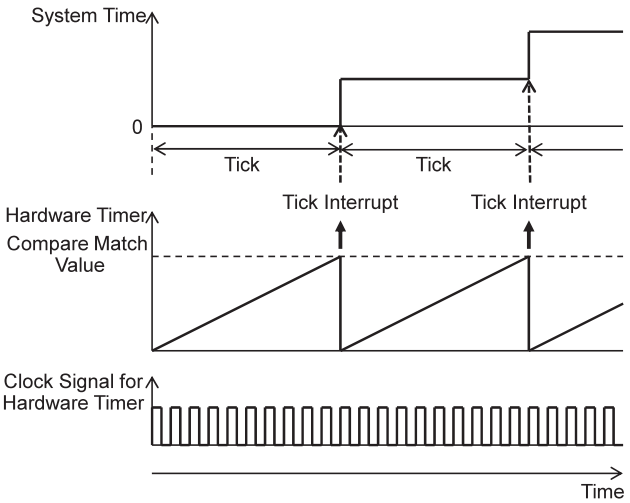

図 4 ティック割込み
Fig. 4 Tick interrupt.

3.2 時刻同期方式

システム時刻が同期しているとは，システム時刻が同じタイミングで同じ値に更新されることである．同じタイミングで更新するにはティック位相が一致していなければならず，ティック位相がずれないようにするにはティック周期を正確に保つ必要がある．ティック周期を正確に保つのがティック周期補正機能，ティック位相を一致させるのがティック位相補正機能である．また，システム時刻の値が違った場合に正しい値にするのがシステム時刻補正機能である．本 RTOS では，GNSS モジュールの出力信号を参照してティック周期，ティック位相，およびシステム時刻を補正する機構を実装することで，時刻同期を実現する．それらの補正を行うには，それらのずれを検出するとともに，正確な値になるように調整しなければならない．

多くの GNSS モジュールは UTC に同期した 1 秒周期の PPS 信号を出力する．PPS 信号のほか，PPS 信号とコヒーレントな高周波クロック信号を出力する GNSS モジュールもある．我々は，PPS 信号と高周波クロック信号の両者を出力する GNSS モジュールを使用し，高周波クロック信号を利用してティック周期のずれの検出するとともに，PPS 信号を利用してティック位相とシステム時刻のずれを検出する．そしてずれを検出したときは補正処理を行うが，ティック周期，ティック位相，システム時刻のいずれの補正処理もハードウェアタイマのコンペアマッチ値を調整することで行う．

図 5 は今回の実装に用いた GNSS モジュール（GNSS Receiver Module）とマイクロプロセッサを表している．前述のように，GNSS モジュールからの信号が停止しても，RTOS 自体は動作を継続する必要がある．そこで，ティック割込みの発行は，従来の RTOS と同じハードウェアタイマで行うこととする．Timer0 がティック割込み用のハードウェアタイマ（コンペマッチタイマ）で，割込みコントローラ（Interrupt Controller）を通してティック割込みを発生する．GNSS モジュールから出力される高周波クロック信号（Low Jitter Clock Signal）をカウントする高周波クロックタイマは Timer1 で，そのカウント値を参照して

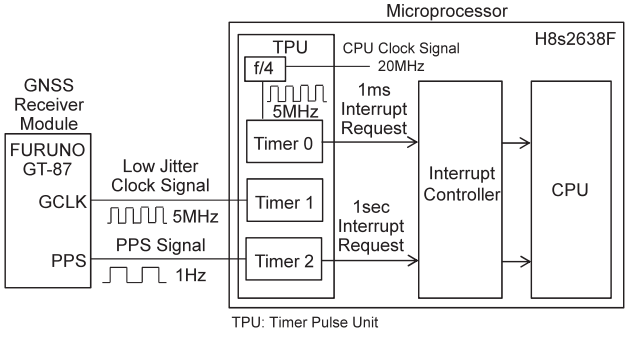

図 5 GNSS モジュールとマイクロプロセッサ
Fig. 5 GNSS receiver module and microprocessor.

© 2016 Information Processing Society of Japan

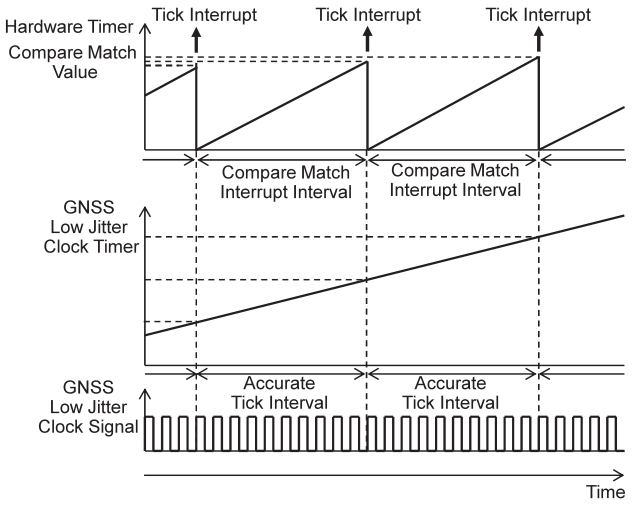

図 6 ティック周期補正
Fig. 6 Tick rate compensation.

ティック周期のずれを検出する．PPS 信号は Timer2 を通して割込みコントローラに接続され，PPS 信号の立ち上がり時に割込み処理を起動し，ティック位相やシステム時刻のずれを検出する．以下の節でそれらの処理の詳細を述べる．

3.3 ティック周期補正

GNSS モジュールが出力する高周波クロック信号を用いてティック周期のずれを検出する．**図 6** は，高周波クロック信号（GNSS Low Jitter Clock Signal），高周波クロック信号をカウントする高周波クロックタイマ（GNSS Low Jitter Clock Timer）の値，およびティック割込みを発生するためのハードウェアタイマの値の関係を表している．

高周波クロックタイマは UTC に同期した高周波クロック信号をカウントするため，その値を参照することで，ティック周期が正確かどうかを知ることができる．すなわち，RTOS のティック周期，すなわち図 6 中のティック割込み間隔（Compare Match Interrupt Interval）と，高周波クロックタイマの値によって表される正確なティック周期（Accurate Tick Interval）を比較することで，ずれを検出する．具体的には，ティック割込み時に高周波クロックタイマの値を読み出し，前回の値との差分をとることでティック割込み間隔を計算し，本来のティック周期と比較して，ずれを検出する．そして，ずれが閾値より大きかった場合，ティック周期のずれをなくすように，ずれの分だけハードウェアタイマのコンペアマッチ値を調整する．この補正処理は，ティック位相補正やシステム時刻補正のためにコンペアマッチ値の調整を行っている場合を除き，毎ティック行う．

今回の実装ではティックを 1 ms とし，クロック周波数 5 MHz のハードウェアタイマを用いたため，コンペアマッチ値は 5000（正確には 4999）となる．また，高周波クロック信号の周波数も 5 MHz としたため，正確なティック周期は 5000 カウントとなる．閾値は 5 カウント（1 μs）とした．たとえば，高周波クロックタイマの値を読み出して，ティック割込み間隔が 4990 カウントと計算された場合は，10 カウント（2 μs）がずれとなるため，コンペアマッチ値を 10 増やすようにする．

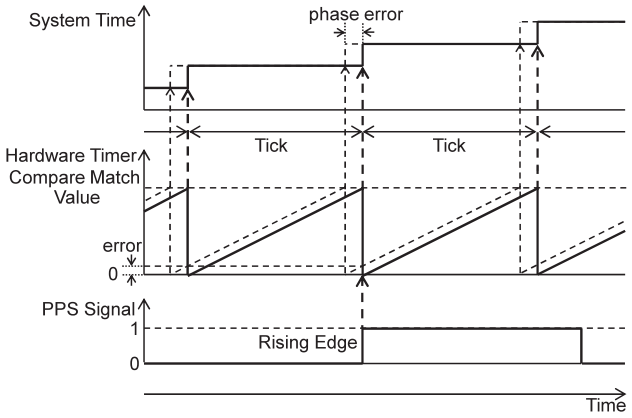

図 7 ティック位相補正
Fig. 7 Tick phase compensation.

3.4 ティック位相補正

GNSS モジュールが出力する PPS 信号を利用してティック位相のずれを検出する．**図 7** は，PPS 信号（PPS Signal），ハードウェアタイマの値，およびシステム時刻の値の関係を表している．ティック位相にずれがなければ，PPS 信号の立ち上がりエッジがシステム時刻を更新するタイミングと一致するはずである．図において，実線で表されたハードウェアタイマの値とシステム時刻の値はティック位相にずれがない場合を示している．一方，破線で表されたハードウェアタイマの値とシステム時刻の値はティック位相にずれ（phase error）がある場合を示しており，PPS 信号の立ち上がり時にハードウェアタイマの値（図の破線の場合の error）を参照し，ゼロでない場合はずれがあると検出する．

PPS 信号立ち上がり時に，ティック位相のずれが閾値以上になったと検出された場合は，そのずれを減少させるように，ハードウェアタイマのコンペアマッチ値を操作してティック割込み間隔を調整する．具体的には，図 7 の error の値がコンペアマッチ値の 1/2 以下の場合は，ティック割込み間隔を調整量 Δt 分だけ大きくし，1/2 以上の場合は Δt だけ小さくする．これを error / Δt 分のティック数の間行うことで，ずれをゼロにする．

今回使用した Δt はティック割込み間隔（1 ms）の 1%（10 μs）を基本とした．しかしこれのみでは 10 μs 以下の精度で調整できないため，ずれが 10 μs 以下になったら Δt を 1 μs とするようにした．たとえば，error が 32 μs と検出された場合は，3 ティック分 10 μs，2 ティック分 1 μs

© 2016 Information Processing Society of Japan

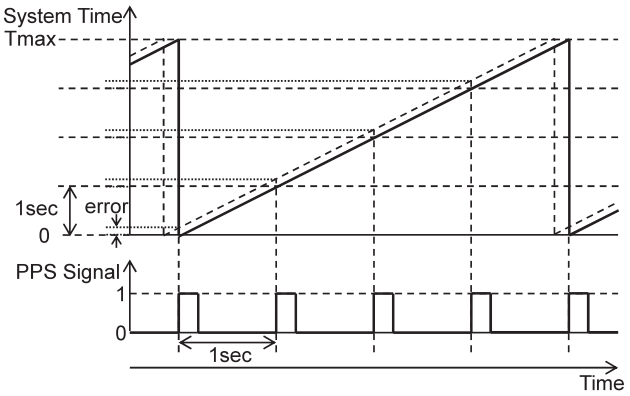

図 8 システム時刻補正 (1)
Fig. 8 System time compensation (1).

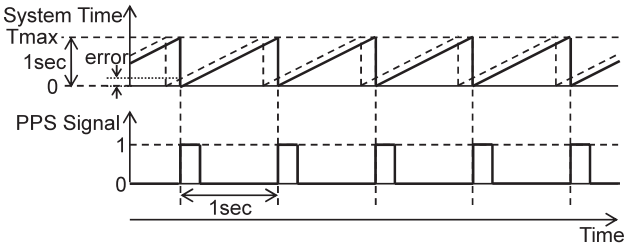

図 9 システム時刻補正 (2)
Fig. 9 System time compensation (2).

とし，計 5 ティック（5 ms）の間調整を行う．今回の実装で調整時間が最大となるのは error が 499 μs のときで，58 ティック（58 ms）となる．

なお，ティック位相の調整を行っている間はティック周期の補正処理は行わないようにする．

3.5 システム時刻補正

GNSS モジュールが PPS 信号および高周波クロック信号を出力し続け，ティック周期補正とティック位相補正が実行されている限りは，システム時刻は正しく保たれる．しかし GNSS モジュールの出力が停止すると，その間は補正が行われないため，システム時刻にずれが発生する可能性がある．そこで，GNSS モジュールの出力が再開された時点でシステム時刻が正しくないことが検出された場合は，システム時刻の調整を行う．

図 8 は PPS 信号とシステム時刻の関係を表している．PPS 信号の立ち上がりエッジのタイミングにおけるシステム時刻の値は，0 または 1 秒の整数倍に対応する値でなければならない．図において，実線で表されたシステム時刻の値はシステム時刻にずれがない場合を示している．一方，破線で表されたシステム時刻の値はシステム時刻にずれ（error）が生じている場合を示している．

図 9 は，システム時刻の最大値（Tmax）を 1 秒に対応するように設定した場合の，PPS 信号とシステム時刻の関係を表している．この場合は，PPS 信号の立ち上がりエッジのタイミングにおけるシステム時刻の値はつねに 0 に一致しなければならない．後述するように，TOPPERS/ATK1 のデフォルトのシステム時刻の最大値は 1 秒に対応するため，今回は図 9 に示した形でシステム時刻を補正する．

システム時刻の最大値が 1 秒でティックタイムが 1 ms の場合，システム時刻の値の範囲は 0〜999 となる．PPS 信号立ち上がり時のシステム時刻がゼロ（正確にはシステム時刻の値が 0 でティック位相のずれを表す図 7 の error の値がコンペアマッチ値の 1/2 以下，あるいは，システム時刻の値が 999 で error の値がコンペアマッチ値の 1/2 以上，以下同じ）でないと検出された場合は，ゼロになるまで，ハードウェアタイマのコンペアマッチ値を操作してティック割込み間隔を調整する．具体的には，図 9 の error の値がシステム時刻の最大値の 1/2 以下（499 以下）の場合は，ティック割込み間隔を調整量 ΔT 分だけ大きくし，最大値の 1/2 以上（500 以上）の場合は ΔT だけ小さくする．前者の場合はシステム時刻の更新速度が遅くなり，後者の場合は更新速度が速くなる．PPS 信号立ち上がり時のシステム時刻の値がゼロになったら，ティック割込み間隔をもとに戻す．

ティック割込み間隔はハードウェアタイマのコンペアマッチ値の設定を変更することで調整する．今回使用した ΔT はティック割込み間隔（1 ms）の 1%（10 μs）を基本とした．しかしこの場合，システム時刻の値は 1 秒間に 10 変動するため，10 未満の調整はできない．そこで，システム時刻の値が 10 未満あるいは 990 以上になった場合は，ΔT をティック割込み間隔（1 ms）の 0.1%（1 μs）とする．これにより，たとえば，PPS 信号立ち上がり時のシステム時刻が 43 と検出された場合は，ティック割込み間隔の 10 μs 削減を 4 秒間，1 μs 削減を 3 秒間行うことになり，計 7 秒でゼロにすることができる．

今回の実装で調整時間が最大となるのは，PPS 信号立ち上がり時のシステム時刻が 499 と検出された場合で，59 秒間かかる．通常はその後ティック位相補正処理が行われるので，60 秒程度かかる．GNSS モジュールが動作を開始してから，GNSS 衛星からの電波を受信し PPS 信号を出力するまでには，数十秒から 2 分程度かかる．したがって，GNSS 衛星からの電波を受信してから時刻同期が完了するまでに最大 3 分程度かかる可能性があり，アプリケーション設計時に考慮すべき事項である．

なお，システム時刻の調整を行っている間は，ティック位相の補正処理やティック周期の補正処理は行わないようにする．

4. 時刻同期機能を有するリアルタイム OS

4.1 ベースとする RTOS

本節では，本開発のベースとした OSEK OS 仕様に基づく TOPPERS/ATK1 について説明する．OSEK OS はタスク管理，アラーム，イベント制御，リソース管理，および

© 2016 Information Processing Society of Japan

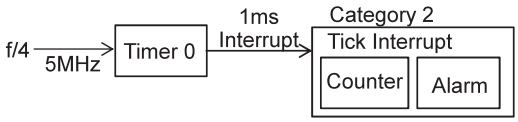

図 10 ティック割込み
Fig. 10 Tick interrupt.

割込み管理機能を有している．OSEK OS では，タスクの周期的起動など，タイマに関連した機能はアラームによって提供される．

TOPPERS/ATK1 のシステムタイマは，ハードウェアタイマにより起動されるティック割込みによって管理される．マイクロプロセッサ H8S/2638F（クロック周波数 20 MHz）を対象とした TOPPERS/ATK1 における，ハードウェアタイマとティック割込みを**図 10** に示す．20 MHz のクロック信号を 4 分周して得た 5 MHz の f/4 クロック信号をカウントするハードウェアタイマ Timer0 を用いて，ティック割込みを発生させる．デフォルトのティックタイムは 1 ms であり，Timer0 のコンペアマッチ値は 5000（正確には 4999）に設定される．デフォルトのシステム時刻の最大値は 1000 で，1 秒に対応する．

OSEK OS の周期タスクはカウンタとアラームを用いて実装される．システム時刻を刻むカウンタをシステムカウンタと呼ぶ．アラームを用いてタスクの起動やイベントの設定ができる．コールバックルーチンを呼び出すことも可能である（アラームコールバックルーチンと呼ばれる）．周期タスクはシステムカウンタに接続したアラームによって起動される．

図 10 に示すように，ティック割込みはシステムカウンタ（Counter）およびシステムカウンタに接続されたアラーム（Alarm）の処理を実行する．TOPPERS/ATK1 のティック割込みはカテゴリ 2 の ISR（Interrupt Service Routine）により実装されている．OSEK OS の ISR にはカテゴリ 1 とカテゴリ 2 があり，システムサービスの呼び出しが可能なのは後者のみである．優先度は後者より前者の方が高い．

TOPPERS/ATK1 に時刻同期機能を実装するにあたっては，必要以上に TOPPERS/ATK1 の仕様やソフトウェア構成，ハードウェアの設定値等を変更しない方針とする．また，前述のように，OSEK OS はアラーム機構によって周期タスクを実現している．したがって，今回実装するシステム時刻同期の精度は，アラーム機構のオーバヘッドと同等あるいはそれ以下の精度を実現することを目標とする．

4.2 時刻同期に関連する API

時刻同期は GNSS モジュールから PPS 信号が出力されている間のみ行われる．そこで，時刻同期がなされているかどうか（同期状態か非同期状態か）をアプリケーションが知るためのシステムサービス（システムコール）を追加する．我々は，時刻同期機能を持つ時間駆動 OS である OSEKtime 仕様 [5] を参照し，そこで定義されているシステムサービス ttGetOSSyncStatus() を参考に，下記のような API を持つシステムサービスを定義する．

```
StatusType GetOSSyncStatus(SyncRefType StatusRef)
```

引数 StatusRef は時刻同期状態の値を記憶する変数へのポインタである．時刻同期状態の値は，SYNCHRONOUS か ASYNCHRONOUS のいずれかである．SYNCHRONOUS は時刻同期機能により正しいシステム時刻が提供されている状態を表す．ASYNCHRONOUS は，PPS 信号が出力されず時刻同期が行われていないか，行われているが補正中で正しいシステム時刻に達していない状態を表す．

また，時刻同期が停止したときにユーザ定義動作を実行するため，下記のようなフックルーチンを追加した．ユーザによって AsynchronousHook が定義されている場合に PPS 信号が停止すると，フックルーチンを実行する．

```
void AsynchronousHook(void)
```

同期状態と非同期状態の判定は以下のように行う．PPS 信号出力が停止したらその時点で非同期状態とする．また，PPS 信号出力が再開された後，システム時刻補正およびティック位相補正が完了し，ティック位相のずれが閾値以下になったら同期状態とする．

4.3 実装

本 RTOS の実装には，マイクロプロセッサ H8S/2638F を搭載した評価ボードを用いた．4.1 節で述べたように，クロック周波数は 20 MHz，ティックは 1 ms，ティック割込み用のハードウェアタイマ（Timer0）のクロックは 5 MHz，コンペアマッチ値は 5000（正確には 4999）である．

使用した GNSS モジュールは古野電気社 [14] 製の GT-87 で，GPS，GLONASS，SBAS および QZSS をサポートしている．PPS 信号の精度は 15 ns(1σ)(@-130 dBm)，50 ns(1σ)(@-150 dBm) である．本 GNSS モジュールの高周波クロック信号は 4 kHz〜40 MHz に設定できる．一般には，高周波クロック信号の周波数が高い方がより精度良く誤差を検出可能であるが，検出と調整のいずれか一方のみの精度を高くしてもトータルとしての同期精度が大きく向上するわけではない．そこで今回は，ティック割込み用タイマのクロックと同じ 5 MHz に設定した．

先に示した図 5 に記載されているように，ティック割込み用の Timer0 のほか，Timer1 と Timer2 のハードウェアタイマも使用する．Timer1 には高周波クロック信号（Low Jitter Clock Signal）を接続し，3.2 節で述べた高周波クロックタイマとして使用する．Timer2 には PPS 信号（PPS Signal）を接続し，1 秒周期の割込みを発生する．

TOPPERS/ATK1 に時刻同期機能を追加して目的とする RTOS を実装する．時刻同期機能は，3 章で述べたティッ

© 2016 Information Processing Society of Japan

ク周期補正，ティック位相補正，システム時刻補正から成る．

図 11 はハードウェアタイマと本 RTOS で実装した割込み処理を示している．ティック割込みのほか，新たに 1 ms 割込み（1 ms Interrupt）と PPS 割込み（PPS Interrupt）という 2 つの ISR を設けた．1 ms 割込みと PPS 割込みは，カテゴリ 2 より優先度が高くオーバヘッドの小さいカテゴリ 1 の ISR とする．また，PPS 割込みは最高優先度，1 ms は 2 番目の優先度とする．そして，TOPPERS/ATK1 でティック割込みを起動していた Timer0 を用いて，1 ms 割込みを起動する．ティック割込みは 1 ms 割込みが発行するトラップ（ソフトウェア割込み）で起動する．また，Timer2 により PPS 割込みを起動する．

図 11 に示すように，PPS 割込みは 3.4 節で述べた方法を用いてティック位相のずれをチェックする（Tick Phase Check）．ずれが閾値（ティックの 0.5%：5 μs）より大きい場合はティック位相補正を行わせるためのフラグをセットする．ずれが閾値より小さい場合はフラグをリセットする．また，3.5 節で述べた方法を用いてシステム時刻のずれをチェックする（System Time Check）．ずれが検出された場合はシステム時刻補正を行わせるためのフラグをセットする．ずれが検出されなかった場合はフラグをリセットする．

1 ms 割込みは 3.3 節で述べた方法を用いてティック周期のずれをチェックする（Tick Rate Check）．1 ms 割込みは PPS 割込みや割込み禁止により遅延する可能性があるため，処理の最初に Timer0 の値を読み出し，その値が遅延限度値（6 μs）より小さい場合のみティック周期のずれをチェックすることとした．そして，ティック周期のずれが閾値（ティックの 0.1%：1 μs）より大きい場合は，3.3 節で述べた方法を用いて Timer0 のコンペアマッチ値調整（Compare Match Adjust）を行い，ティック周期の補正を行う．

1 ms 割込みのコンペアマッチ値調整は，ティック位相補正やシステム時刻補正のための処理も行う．ティック位相補正を行わせるためのフラグがセットされていれば，3.4 節で述べたように Timer0 のコンペアマッチ値を調整し，ティック位相を補正する．システム時刻補正を行わせるためのフラグがセットされていれば，3.5 節で述べたように Timer0 のコンペアマッチ値を調整し，システム時刻を補正する．

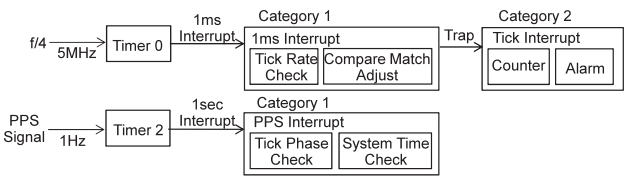

図 11 ISR の構成
Fig. 11 ISR structure.

なお，ティック位相補正やシステム時刻補正のために Timer0 のコンペアマッチ値を調整すると，ティック周期もその調整量だけ変化する．そこで，ティック周期の変化が 1%以下になるように，コンペアマッチ値の調整範囲を限定する．

5. 評価

5.1 時刻同期処理のオーバヘッド

追加した時刻同期処理のオーバヘッドを評価するため，PPS 割込み，1 ms 割込み，およびティック割込みの CPU 実行時間を測定した．**表 1** に，開発した RTOS（Extended RTOS）の各割込み処理（Interrupt）の実行時間（Execution Time）の平均値（Ave），最大値（Max），最小値（Min）を示す．PPS 割込みについては，同期状態にある場合（synchronous），システム時刻のずれはないがティック位相のずれを検出した場合（phase error），システム時刻のずれを検出した場合（system time error），PPS 信号が安定して出力されるのを確認するために出力再開後 3 回連続出力されるまで待っている場合（waiting）に分けて示している．1 ms 割込みについては，ティック周期のチェックを行う場合（with rate check），ティック周期のチェックを行わない場合（without rate check），PPS 信号が出力されていない場合（no PPS）に分けて示している．ティック割込みについては，アラームが定義されていない場合（without alarm），アラームが定義されているがタスク起動は行わない場合（with alarm），アラームが定義されていてタスク起動を行う場合（with task activation）に分けて示している．また比較のため，表 1 にはオリジナルの TOPPERS/ATK1（Original）のティック割込みの実行時間も示している．

PPS 割込みの通常の実行時間は 11.4 μs で，オリジナルの TOPPERS/ATK1 におけるアラームが定義されていな

表 1 割込み処理の実行時間
Table 1 Execution times of interrupts.

RTOS	Interrupt		Execution Time [μs] Ave	Max	Min
Extended RTOS	PPS	synchronous	11.4	11.4	11.4
		phase error	11.4	11.4	11.4
		system time error	10.7	10.7	10.6
		waiting	2.1	2.1	2.1
	1 ms	with rate check	9.2	9.2	9.2
		without rate check	3.7	3.7	3.7
		no PPS	3.6	3.6	3.6
	Tick	without alarm	10.9	10.9	10.9
		with alarm	13.9	13.9	13.9
		with task activation	46.7	46.7	46.7
Original	Tick	without alarm	11.5	11.5	11.5
		with alarm	14.5	14.5	14.5
		with task activation	47.2	47.2	47.2

© 2016 Information Processing Society of Japan

い場合のティック割込みの実行時間と同程度である．また，1 ms 割込みの実行時間は最大で 9.2 μs であり，オリジナルのティック割込みの実行時間よりも小さい値である．したがって，開発した RTOS のオーバヘッドは実用上問題ない程度であると考えている．

5.2 時刻同期の精度

ティック割込み用のハードウェアタイマ（Timer 0）の時刻同期の精度を評価するため，ハードウェアタイマがコンペアマッチ出力を行うように設定し，その時刻差を測定した．この時刻差が 1 ms 割込みの起動時刻の差になる．具体的には，GNSS モジュールを搭載した 2 台の評価ボードのコンペアマッチ出力信号をオシロスコープで観測し，それらの信号の立ち上がり時刻の差を計測した．これを，手持ちの 4 台の評価ボードと 4 つの GNSS モジュールについて組合せを変えて行った．コンペアマッチ出力の時刻差は通常は 10 μs 以下であるが，最悪の場合は 18 μs であった．したがって，ティック割込み用のハードウェアタイマの時刻同期の誤差は最大で 18 μs と考える．

表 1 に示したオリジナルの TOPPERS/ATK1 におけるアラームが定義されタスクが起動されない場合とタスクが起動される場合のティック割込みの差より，TOPPERS/ATK1 におけるタスク起動のオーバヘッドは 33 μs 程度と考えられる．したがって，上記時刻同期の誤差は最大でもタスク起動のオーバヘッド程度である．

TOPPERS/ATK1 では排他制御のために割込み禁止が用いられており，それによって割込み処理が遅延する場合がある．ティック位相やシステム時刻のずれは PPS 割込みで検出している．PPS 割込みは最高優先度に設定したが，割込み禁止の影響を受ける可能性がある．しかしながら，先に述べたハードウェアタイマの時刻同期の誤差には，割込み禁止時間時間の影響は含まれていない．TOPPERS/ATK1 の割込み禁止時間を測定したところ，最大で 16.5 μs であった．したがって，割込み禁止を考慮すると，時刻同期の誤差は最大で 35 μs 程度と，タスク起動のオーバヘッドと同程度になる可能性がある．なお，ティック周期のずれは 1 ms 割込みで行っているが，遅延時間が限度値（6 μs）より大きいときはチェックを行わないため，割込み禁止の影響は受けにくいと考える．

アプリケーションを考えた場合，自動車パワートレイン制御における典型的な周期タスクの周期は 10 ms あるいはそれ以上，典型的な非周期タスクであるクランクシャフトの回転に同期して起動されるタスクの起動間隔は回転数 6000 rpm のときで 10 ms である．したがって，時刻同期の誤差は最大でもタスクの周期あるいは起動間隔の 0.3%程度である．以上のことから，OSEK OS が使用されている自動車制御システムのようなアプリケーション分野では，本 RTOS における時刻同期の誤差は許容できる範囲と考える．

今回は本 GNSS モジュールの高周波クロック信号を，ティック割込み用のタイマのクロックと同じ 5 MHz に設定した．これは前述のように，いずれか一方のみの精度を高くしてもトータルとしての同期精度が大きく向上するわけではないとの判断からである．ティック割込み用タイマとしてより高いクロック周波数を使用する場合は，GNSS モジュールの高周波クロック信号の周波数も上げることで，より高精度な検出と調整が可能になる．ただし，GNSS モジュールの高周波クロック信号の周波数には上限があるため，ティック割込み用タイマのクロック周波数と同程度に設定できるとは限らない．しかし，今回使用したモジュールの上限周波数 40 MHz の周期は 25 ns で，ティックが 1 ms の場合はその 0.0025%であることから，40 MHz あれば十分な検出精度が得られるものと推定する．

RTOS における時刻同期の要求精度は，RTOS のティックタイムで決まるものと考える．評価実験ではクロック周波数 20 MHz の CISC プロセッサを用いたが，ティックタイムが同程度であれば，よりクロック周波数が高い高性能プロセッサにも適用可能と考えている．一般に，自動車制御分野で高性能プロセッサが要求されるのは，制御周期を短縮するためではなく処理量の増大に対応するためで，高性能プロセッサを用いた場合でも RTOS のティックは 1 ms 程度であるのが普通である．ティックタイムが同じであれば，高性能プロセッサを用いた方が同期誤差を小さくできる可能性がある．しかしながら，より短いティックタイムが必要な場合や，時刻同期の最大誤差を 1 μs 以下に抑える必要があるような場合，たとえば 100 μs といった制御周期が要求されるアプリケーション向けには，OSEK OS ではなく，OSEKTime [5] のような時間駆動 OS をベースにする必要があると考える．

6. おわりに

本論文では，無線ネットワークや広域ネットワークにより接続された疎結合分散型の CPS のための，GNSS を用いた時刻同期機能を有する RTOS を提案した．本 RTOS は OSEK OS を拡張したもので，その時刻同期機能はティック周期補正機構，ティック位相補正機構，およびシステム時刻補正機構により実現している．本 RTOS を使用することで，高精度で同期したシステム時刻に基づいてアプリケーションのタスクをスケジューリングでき，実世界に存在する対象物を同期して制御することが可能になる．また評価実験を行った結果より，OSEK OS が使用される自動車制御のようなアプリケーション分野において実用上問題ない程度のオーバヘッドや時刻同期誤差であることを確認した．

今回はシステム時刻の最大値を 1 秒としたが，システム時刻の最大値をより大きくするには，PPS 信号による同

© 2016 Information Processing Society of Japan

期のみでは不十分で，GNSS モジュールからシリアル通信（UART）により文字列として提供される UTC の情報を利用する必要がある．UTC 情報を利用したシステム時刻の拡張やリアルタイムクロックの提供は今後の課題である．また，さらに高精度の時刻同期が必要なアプリケーション分野に対応するため，より高精度な制御が可能なティック周期補正やティック位相補正のアルゴリズムを検討するとともに，GNSS を用いた時刻同期機能を有する時間駆動 OS を開発することも今後の課題である．

謝辞 本研究で使用した TOPPERS/ATK1 の開発者に感謝する．また，有益なコメントをいただいた匿名の査読者に感謝する．本研究は JSPS 科研費 15K00084 の助成を受けたものである．

参考文献

[1] Lee, E.A.: Cyber Physical Systems: Design Challenges, *Proc. 11th IEEE International Symposium on Object Oriented Real-Time Distributed Computing* (*ISORC*), pp.363–369 (2008).
[2] Cervin, A., Henriksson, D., Lincoln, B., Eker, J. and Arzen, K.: How Does Control Timing Affect Performance? Analysis and Simulation of Timing Using Jitterbug and TrueTime, *IEEE Control Systems*, Vol.23, No.3, pp.16–30 (2003).
[3] Mills, D.L.: Internet Time Synchronization: The Network Time Protocol, *IEEE Trans. Comm.*, Vol.39, No.10, pp.1482–1493 (1991).
[4] Kopetz, H.: Should Responsive Systems be Event-Triggered or Time-Triggered?, *IEICE Trans. Information & Systems*, Vol.E76-D, No.11, pp.1325–1332 (1993).
[5] OSEK/VDX: Time-Triggered Operating System, Version 1.0 (2001).
[6] 知場貴洋，齊藤政典，伊丹悠一，兪 明連，横山孝典：位置透過性のあるシステムコールを有する組み込み制御システム向け分散リアルタイム OS，情報処理学会論文誌，Vol.53, No.12, pp.2702–2714 (2012).
[7] Makowitz, R. and Temple, C.: FlexRay – A Communication Network for Automotive Control Systems, *Proc. 2006 IEEE International Workshop on Factory Communication Systems*, pp.207–212 (2006).
[8] Lewandowski, W., Azoubib, J. and Klepczynski, W.J.: GPS: Primary Tool for Time Transfer, *Proc. IEEE*, Vol.87, No.1, pp.163–172 (1999).
[9] Kim, K.H. and Jenks, S.F.: The TMO Scheme for Wide-Area Distributed Real-Time Computing and Distributed Time-Triggered Simulation, *Proc. IEEE International Parallel and Distributed Processing Symposium*, pp.1–6 (2007).
[10] Quesada, J., Uriarte Llano, J., Sebastian, R., Castro, M. and Jacob, E.: Evaluation of Clock Synchronization Methods for Measurement and Control Using Embedded Linux SBCs, *Proc. 9th International Conference on Remote Engineering and Virtual Instrumentation* (*REV*), pp.1–7 (2012).
[11] TOPPERS Project, available from ⟨http://www.toppers.jp/⟩(accessed 2015-11-18).
[12] OSEK/VDX, Operating System, Version 2.2.3 (2005).
[13] Sun, J. and Liu, J.: Synchronization Protocols in Distributed Real-Time Systems, *Proc. 16th International Conference on Distributed Computing Systems*, pp.38–45 (1996).
[14] Furuno Electric Co., Ltd, available from ⟨http://www.furuno.co.jp/⟩

松原 彩音 （学生会員）

2015 年東京都市大学知識工学部情報科学科卒業．同年同大学大学院工学研究科情報工学専攻修士課程入学．現在，同課程在学中．組み込み OS の研究に従事．

兪 明連 （正会員）

1994 年安東国立大学校工学部コンピュータ工学科卒業．1996 年浦項工科大学情報通信専攻修士課程修了．同年安東情報大学講師．2002 年嶺南大学コンピュータ工学専攻博士課程修了．2006 年早稲田大学大学院情報生産システム研究科博士後期課程修了．2007 年武蔵工業大学．現在，東京都市大学准教授．スケジューリング理論の研究に従事．博士（工学）．電子情報通信学会，IEEE 各会員．

横山 孝典 （正会員）

1981 年東北大学工学部通信工学科卒業．1983 年同大学大学院工学研究科電気及通信工学専攻修士課程修了．同年（株）日立製作所入社．1987 年から 1990 年まで（財）新世代コンピュータ技術開発機構出向．2004 年武蔵工業大学．現在，東京都市大学教授．組み込みシステム，分散システム，ソフトウェア工学等の研究に従事．博士（情報科学）．電子情報通信学会，IEEE，ACM 各会員．

© 2016 Information Processing Society of Japan