4U-05

RT-Messengerを用いた時刻同期機構の実装と評価

†鞠谷　達士　†上田　和樹　†矢向　高弘
†慶應義塾大学

1　はじめに

近年，産業分野におけるファクトリーオートメーションは大規模化や複雑化が進んでいる．多くのアクチュエータやセンサがそれぞれのプロセッサによって分散制御されているようなシステムにおいて，それぞれのノード間で制御信号など実時間性を持つデータを通信し合うような動作を行わせる場合，システムを正確なタイミングで動作させるためにはノード間で高精度な時刻同期を行うことが必要不可欠である．

現在，ネットワークによる時刻同期でデファクトスタンダードになっているものはNTP[1]である．しかしNTPではミリ秒程度の精度しか得ることが出来ず，リアルタイムシステムの要求に応えることが出来ない．そこでNTPより高精度なネットワーク時刻同期手法としてIEEE1588[2]が策定された．IEEE1588はハードウェアタイムスタンプやネットワーク構成によりナノ秒単位の時刻同期精度が実現可能である．このように低コストで高精度な時刻同期手法が望まれている．

また，大学・研究機関では安価かつ汎用性のある環境での研究開発が望まれていることから，実時間制御システムの研究分野ではフリーかつオープンソースな実時間OSであるRTAI[3]が用いられている．RTAIを用いたノード間で時刻同期を行うためには，RTAIのスケジューラが参照するクロックを同期する必要がある．しかし，RTAIのクロックは内部カウンタユニットによってソフトウェア実装されているため，IEEE1588のようなハードウェアタイムスタンプを用いることが出来ない．ソフトウェアタイムスタンプを用いた場合，ネットワークスタック内の処理時間に変動があるため高精度な時刻同期が出来ない．

本研究室では，ネットワークを介して実時間制御システムを用いるための実時間通信機構であるRT-Messenger[4]をRTAI上で提案・実装してきた．RT-Messengerはネットワークスタック内の処理時間を最小化することが出来る機構である．そこで本論文では，RT-Messengerを用いてソフトウェアタイムスタンプによるIEEE1588時刻同期機構を実装し評価した．

Design, Implementation, and Evaluation of Time Synchronization Mechanism using RT-Messenger
†Tatsushi KIKUTANI　†Kazuki UEDA　†Takahiro YAKOH
†Keio University

2　関連研究

本研究で用いているRTAIとRT-Messengerについて説明する．

2.1　RTAI

RTAIはLinuxカーネルに実時間性を持たせるためのインタフェース群である．RTAIのスケジューラでは全てのリアルタイムタスクとLinuxカーネルは同格であり，Linuxカーネルは最も優先度の低いタスクとして扱われる．そのため，Linuxカーネルがいかなる処理をおこなっていても，リアルタイムタスクが実行可能状態になれば，即座にタスクを切り替えて優先度の高いリアルタイムタスクを実行することが可能である．

2.2　RT-Messenger

RT-MessengerはリアルタイムOSであるRTAI上で実装が行われているため，リアルタイムタスク上で送受信処理が実行可能状態になると，その他のプロセスに影響されずに即座に処理を実行することが出来る．また，送受信処理においてリアルタイムタスクから直接，NICのデバイスドライバにパケットの受け渡しを行っているためキューイングによる予測不能な遅延を回避することができ，通信遅延のジッタを最小化することができる．

3　提案システム

本論文における提案システムの構成は図1のようになっている．

提案システムは三つのファンクションに分かれている．各ファンクションについて説明する．

3.1　Message exchange function

このファンクションではRT-Messengerを用いて時刻情報のやりとりを行う．まずマスターノードからスレーブノードにSyncメッセージを送る．Syncメッセージの送信時刻を t_1，受信時刻を t_2 とする．（送信時刻 t_1 はSyncメッセージに格納してスレーブノードに送られる．）その後，スレーブノードからマスターノードにDelay_Reqメッセージを送る．Delay_Reqメッセージの送信時刻を t_3，受信時刻を t_4 とする．マスターノードはDelay_Reqを受信した後，受信時刻 t_4 をDelay_Respメッセージに格納してスレーブノードに送る．以上により，スレーブノードは t_1 から t_4 の四つの時刻情報を得ることができる．

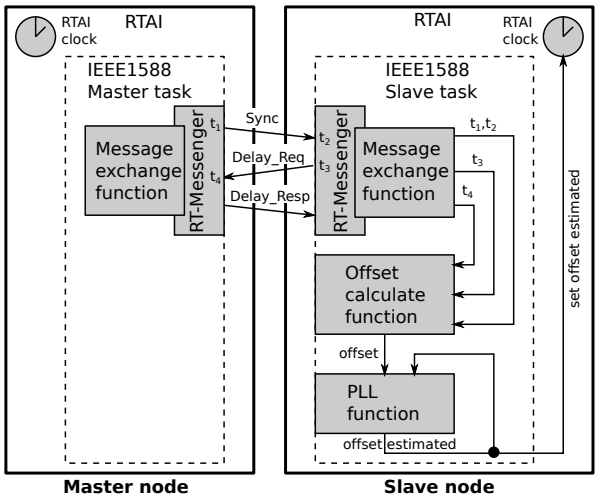

図 1: システム構成

3.2 Offset calculate function

スレーブノードは得られた４つの時刻情報に基いてオフセットを計算する．マスタースレーブ間のクロックのずれを t_{offset} とし，メッセージの通信遅延を往路と復路が等しいと仮定してその値を t_{delay} とすると，それらは以下の式で表される．

$$t_{offset} = \frac{(t_2 - t_1) + (t_4 - t_3)}{2} \quad (1)$$

$$t_{delay} = \frac{(t_2 - t_1) - (t_4 - t_3)}{2} \quad (2)$$

3.3 PLL (phase locked loop) function

RTAI を駆動するタイマは 16.6 MHz の水晶発振子で駆動されているが，周波数は温度や印加電圧の変動にともなって変動するため，時間変化によって微小なぶれが生じている．そのため，クロックの値は一定に増加しておらず，そのぶれはオフセットの値にも影響を与える．これを低減するためにオフセットの値に PLL を適用した．PLL は通常，複素数平面上での浮動小数点演算を行うが，本提案システムはカーネル空間で実装を行ったため，整数型の四則演算のみを用いて PLL を実装した．PLL 適用後のオフセットの値を算出し，その値を RTAI のクロックに足し合わせることで RTAI のクロックの時刻同期を行っている．

4 性能評価

本論文の目的は RTAI のクロックの高精度な時刻同期である．そのため，提案システムを用いて時刻同期を行い，マスタースレーブ間のオフセットの測定を行った．1 s ごとに時刻同期を行ったところ，オフセットは図 2 のようになった．

オフセットの値は全体の 99.6%が 50 µs 以下となった．これは，同じソフトウェアタイムスタンプによる実装

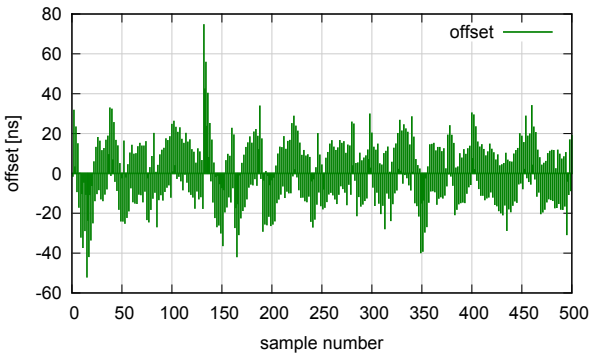

図 2: マスタースレーブ間のオフセット

であり現在デファクトスタンダードになっている NTP より格段に良い時刻同期精度である．また，分散制御分野で求められているマイクロ秒精度の時刻同期の要求をみたすことができた．

5 おわりに

RTAI のスケジューラが参照するクロックの時刻同期システムを提案した．RTAI のクロックはソフトウェア実装されているため，時刻同期精度の悪いソフトウェアタイムスタンプによる実装しかすることができない．そこで，ネットワークスタック内の処理時間を最小化する RT-Messenger を用いて時刻情報のやりとりを行うことで精度の向上を図った．結果，NTP より格段に良い 50 µs 精度の時刻同期を行うことができた．

謝辞

本研究は JSPS 科研費 24300086 の助成を受けたものです．

参考文献

[1] D.L. Mills. Internet time synchronization: The network time protocol. *IEEE Trans. Comm.*, Vol. 39, pp. 1482–1493, Oct. 1992.

[2] IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std 1588-2008*, Vol. 1, pp. 1–269, Jul. 2008.

[3] P. Mantegazza, E. L. Dozio, and S. Papacharalambous. RTAI: Real Time Application Interface. *Linux Journal*, Vol. 2000, , April 2000.

[4] H. Sato and T. Yakoh. A real-time communication mechanism for RTLinux. *26th Annual Conference of the IEEE Industrial Electronics Society, IECON 2000*, Vol. 4, pp. 2437–2442, 2000.