ISS-P-138

RTOS の割り込み応答性能を向上させる Dynamic-tick 処理の提案

濱野　恵輔†　　南角　茂樹†
† 大阪電気通信大学大学院総合情報学研究科

1. はじめに

組込みシステムにはシステムの信頼性や開発者の生産性を向上させる目的で RTOS が多く利用されている. 近年ではシステムの高性能化および多機能化に伴い, プログラムの保守性の観点からRTOSの必要性がさらに高まっている. 一方で RTOS の使用にはオーバーヘッドの問題が存在し, 割り込み応答性能を低下させる要因となっている. 例えば機械制御など高い割り込み応答性能を要求するシステムで RTOS が利用できない場合がある. 本稿では, RTOS によるオーバーヘッドの原因の一つとなるシステム時刻の tick 処理[1]に着目し, 問題を解決する Dynamic-tick 処理を提案する.

2. 従来の問題点

組込みシステムにおいてシステム時刻とはシステムが起動した時点からの経過時間を示す. 多くの RTOS で時間管理機能が提供されており, システム時刻はその機能の一つである. このシステム時刻を実現する処理を tick 処理と呼び, 周期的なタイマー割り込みによって実装される. システムの時間という排他的なデータを扱うため, tick 処理には割り込み禁止区間がある. また, 割り込み処理の前後のコンテキストの保存/復元処理も割り込み禁止で実行される. 割り込み禁止区間を含む処理がタイマー割り込みによって周期的に行われるために, 割り込み応答性能が損なわれる.

3. Dynamic-tick 処理の提案

2 章で述べた問題を解決するため, tick 処理の周期を動的に変化させることで, 時間管理機能を保ちつつオーバーヘッドを抑える Dynamic-tick 処理を提案する. システムが通常動作しているとき, dly_tsk などの時間管理機能の APIを使わない場合は tick 処理によるオーバーヘッドを減らすために tick 処理の周期を長くする. 時間管理機能の APIを使う場合は tick 処理の周期を必要に応じて変更し, 時間管理機能を正しく動作させる.

図 1 に Dynamic-tick 処理の動作例を示す. 図において縦の点線は 1ms 間隔を示す. Dynamic-tick 処理では時間管理機能の API を使わない場合, タイマー割り込みの周期を長くして 2ms 周期で tick 処理を行う. Task の処理時間は dly_tsk を含めて 2ms とする. Task が実行された後, dly_tsk が起動する. 時間管理機能の API である dly_tsk が起動すると, Task の実行を指定時間遅らせる. 今回は 3ms 遅らせる. dly_tsk が起動した時点では tick 処理の周期は 2ms である. このままでは正しい指定時間を処理することができないので, tick 処理の周期を変更する処理を行う. これにより, dly_tsk の指定時間を正しく処理する. dly_tsk の処理を終えた後, 周期を元に戻してから Task の処理を再開する. 以上により tick 処理による割り込み禁止区間の処理が削減され, 割り込み応答性能を向上させることが出来る.

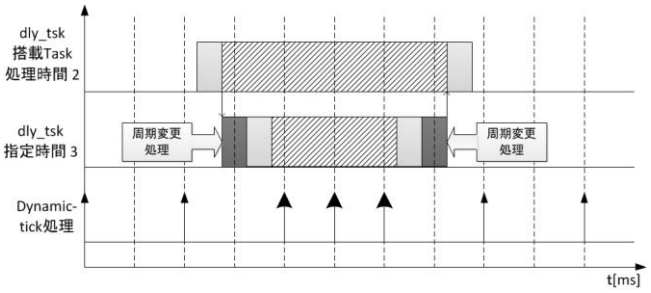

図1. Dynamic-tick 処理の動作例

4. 実現方式

提案方式の実現手法として以下の二つを検討している.

方式 1　tick 処理の周期変更をする API による方式

tick 処理の周期を変更する API を実装し, 時間管理機能の API を用いる直前と直後で, 開発者が周期変更 API を呼び出して周期を変更する. この場合, 周期の変更が開発者に委ねられるため, 設計ミスが発生する可能性が考えられる.

方式 2　時間管理機能の API 内で周期変更を行う方式

時間管理機能の API の内部に tick 処理の周期を変更する処理を追加する. これにより設計ミスの問題は少なくなると考えられる.

5. 今後の予定

本稿で示した Dynamic-tick 処理を適用することで, tick 処理によるオーバーヘッドを削減できると考える. 今後は提案手法を実装し有効性を検証する.

参考文献

[1] 丸山 修考・石原 亨・高田 広章・安浦 寛人,「超高速応答を実現するハードウェア割り込み処理機構」, 電子情報通信学会技術研究報告, DC, ディペンダブルコンピューティング 111(325), 31-36, 2011-11-21