

RTOS 利用システムのフルハードウェア化における タスク通知の実装

Design of Task Notification Module in Full Hardware Implementation of RTOS-Based Systems

三上 啓¹
Kei Mikami

石浦 菜岐佐²
Nagisa Ishiura

関西学院大学 理工学部¹

関西学院大学 工学部²

School of Science and Technology, Kwansei Gakuin Univ.

School of Engineering, Kwansei Gakuin Univ.

1 はじめに

RTOS を用いたシステムの応答性能を向上させる手法として、RTOS が提供する機能とタスク/ハンドラの全てをハードウェア化する手法が提案されている [1] [2]. 本稿では、文献 [2] のハードウェア構成においてタスク間の軽量な同期・通信機能であるタスク通知の実装を行う。

2 タスク通知と前提とするハードウェア構成

タスク通知は各タスクが一つずつ保有し、基本的な操作として notify, wait, give, take を提供する。以下、タスク T のタスク通知を N_T とする。タスク T_1 が notify(T_2, val) を実行すると、 N_{T_2} は「受信状態」になり val を記憶する。 T_2 が wait() を実行すると、 N_{T_2} は受信状態であれば記憶している値を T_2 に返して値をクリアする。そうでなければ T_2 は N_{T_2} が受信状態になるまで「待ち状態」になる。 T_1 が give(T_2) を実行すると N_{T_2} の値は 1 増え、 T_2 が take() を実行すると値は 1 減る。 N_{T_2} の値が 0 のときに T_2 が take() を実行すると、 T_2 は give(T_2) が実行されるまで待ち状態になる。

文献 [2] のハードウェア構成を図 1 に示す。 T_i はタスクを、 S_j は RTOS のサービスを実行するモジュールである。サービスは一度に一つだけ処理され、RA はその調停を行う。 T_i はレジスタ TF_i と TA_i に所望のサービス番号と引数を書き込みサービスを要求する。RA が XT に実行を許可するタスクの番号、 XF に TF_i 、 XA に TA_i を書き込むと、サービスモジュールが処理を行って XA に戻り値を書き込み、それが TA_i に返される。status は各タスクの状態 STATE、タイマー値 TIMER 等を、wait[S_j][T_i] は T_i がサービス S_j を待っていることを記憶するレジスタである。

3 タスク通知の実装

本稿ではタスク通知の主な機能をサービスモジュールの一つ (図 1 の S3) として実装する。全てのタスク通知を 1 モジュールで管理し、各タスク通知の値と状態をレジスタ value と state に記憶する。

タスク通知モジュールの notify/wait 実行の流れを図 2 に示す。notify(T, val) の T と val はそれぞれ $XA[0]$ 、 $XA[1]$ に与えられる。タスク通知が待ち状態であれば wait[N_t][t] を 0 にして待ち解除する。state を受信状態にし、value に $XA[1]$ を、戻り値を $X[0]$ にセットしてマネージャに完了を通知する。タスク XT が wait() を実行した時に state が受信状態であれば、 $XA[0]$ に value を返すと同時に value の値をクリアし、 XT を実行可能状態にする。そうでなければ XT を待ち状態にし、タイマー値 ($XA[2]$) をセットする。TIMER と STATE はともに status レジスタ内にあるため更新には 2 サイクル

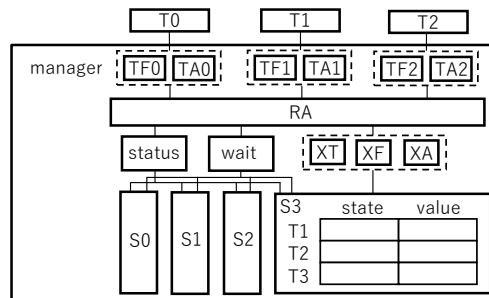


図 1 文献 [2] のハードウェア構成

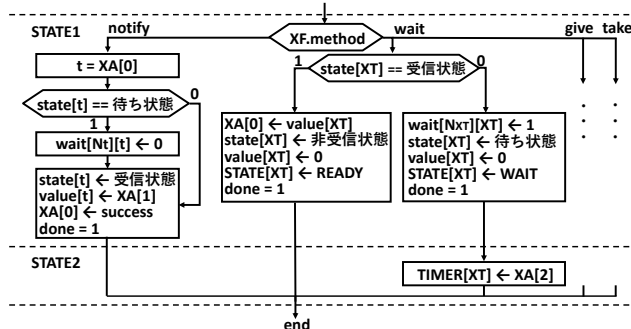


図 2 タスク通知の処理の流れ

を要するが、制御は 1 サイクルでマネージャに戻せる。take/give の処理も wait/notify と同様である。

4 実装結果

提案手法に基づくタスク通知モジュールを Verilog HDL で設計し、Xilinx Vivado (2016.4) で Artix-7 をターゲットに論理合成した。回路規模は LUT 数が 430、FF 数が 138、遅延が 6.405ns で、いずれも十分小さな値となった。また notify/wait と give/take いずれもタスクが XF 、 XA に要求を書き込んでから戻り値が XA にセットされるまで 4 サイクルで実行できた。

5 おわりに

本稿では、RTOS 利用システムのフルハードウェア化においてタスク通知を実装した。タスク数に応じたサービスモジュールの自動生成が今後の課題である。

謝辞 本研究は一部 JSPS 科研費 19H04081 の助成による。

参考文献

- [1] Y. Oosako, N. Ishiura, H. Tomiyama, and H. Kanbara: "Synthesis of full hardware implementation of RTOS-based systems," in *Proc. RSP 2018*, pp. 1-7 (Oct. 2018).
- [2] 六車, 石浦, 安堂, 富山, 神原: "RTOS 利用システムのフルハードウェア化におけるサービス処理機能の集約," 信学技報, VLD2020-75 (Mar. 2021).