

## リアルタイムシステムにおけるイベントビューワの提案

3D-6

北島 正久, 植木 克彦, 平山 雅之†  
株式会社東芝 研究開発センター S&S 研究所‡

## 1.はじめに

マイコン組み込みシステムの開発において、システムの複雑化、要求されるリアルタイム制約を守る必要性等から、マルチタスク・リアルタイムOS（以下RTOSと表記する）を採用することが一般的となってきた。

しかし、RTOS上で動作するアプリケーションは、タスク相互関係の理解の困難さといった従来のシステムに比べて複雑な問題を抱えており、RTOS対応のデバッグ機能を備えたデバッグ環境を必要としている。そこで我々は、実行時に記録したタスク、セマフォ状態等の変化の履歴を表示する機能等を持つイベントビューワを開発した。

本報告では、イベントビューワの機能や表示内容、また、それによる効率的なデバッグ・テスト方法について述べる。

## 2.デバッグ時に必要とされる情報

RTOSを使ったシステムをデバッグする時に、ユーザが興味を持つ情報は以下の3点と考えられる。

1. プログラム（タスク）が期待通りの順番で動いているかどうか。
2. プログラム（タスク）が期待通りの状態になり、処理を行っているかどうか。
3. プログラム（タスク）が期待通りの時間内に処理を終えているかどうか。

しかし、RTOSを用いたシステムの特徴として以下の問題点がある。

- a. 動作の再現が困難な場合がある。
- b. 目的の状態プログラムを停止させるのが難しい。
- c. RTOSが持っているシステムの内部状態を調査しにくい。
- d. システムコールやタスク切り替えの履歴の理解調査が難しい。

これらの問題点を解決するために、RTOSに対応したデバッグ機能が必要である。イベントビューワでは、わかりづらいRTOSの状態をGUIで分かりやすく表示することで、これらの問題点に対応する。

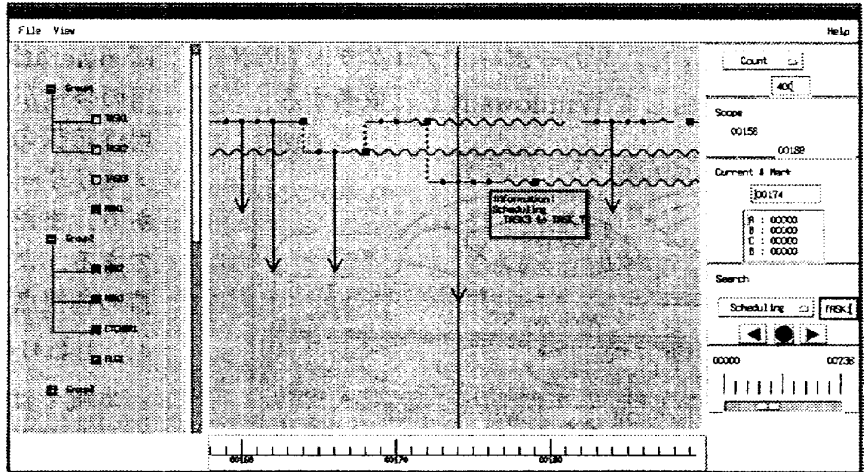


図1 イベントビューワの画面構成

## 3.実装環境

このイベントビューワは、ターゲットシステムと対象OSから出力されたログファイルを読み込む外部方式で実現した。UNIX (Solaris2.x) 上で実装し、対象OSはμITRONとした。

## 4.イベントビューワの機能

以上を踏まえてイベントビューワを実装した。画面構成を図1に、その主な機能を以下に示す。

○イベント状態表示機能（“2.デバッグ時に必要とされる情報”のユーザが興味を持つ情報1.、2.に対応）

- ・時間やイベント数を基準にした系列で表示。
- ・タスクの状態に合わせて状態線を表示する。
- ・システムコール発行・終了、タスク・ハンドラの切り替え等を表示。

○イベント内容表示機能（“2.デバッグ時に必要とされる情報”の問題点c.、d.に対応）

- ・システムコールに関する情報（図2参照）、タスク/ハンドラ切り替えに関する情報を表示。
- その他、検索機能やグルーピング機能等も持つ。

## 5.デバッグ工程での使用方法と効果について

RTOSに限らずデバッグ時に必要な機能として、プログラムの実行や停止、メモリやレジスタの内容の確認等の機能が挙げられる。しかし、RTOSを使ったシステムは、動作の非再現性の影響を強く受けることや、タスクの相互関係を理解することが難しいという問題点があり、上記の確認が容易でない。そこで、ここでは動作の再現ではなく動作の履歴を調べることでこの点を解消し、デバッグ効率の向上に結びつける。

必要な機能として考えられるものを以下に示す。

\*A Study of the Event-Viewer on Real-Time Embedded System

†Tadahisa KITAJIMA, Katsuhiko UEKI, Masayuki HIRAYAMA

‡ Systems & Software Laboratory, Research & Development Center, TOSHIBA Corporation

- ・システムコールの発行、終了を表示する機能
  - ・オブジェクト（タスクやその他の資源）の状態を表示する機能
  - ・タスク、ハンドラの切り替えを表示する機能
- イベントビューワは、これらの機能をサポートし、実行時のタスク、セマフォ状態が可視化されることにより、システムの挙動を把握しやすくなり、予定外の動きや不正な動きを発見しやすくなる。その結果、効率的なデバッグが行える。

6. 効率的なテストデバッグ手法の提案

様々なテスト工程で、記録されたログをグラフィカルに表示して、システムとしてのテスト時の動作確認を効率的に行う手法を提案する。

- ・サブシステム間でメッセージの交換を行うシステム等で、設計したプロトコル通りに動いているかどうか検査する。
- ・テスト時に動作を記録してプロトコルの検査用ツールとして使う。

この時に必要となる情報は、メッセージの流れ、内容、意味である。本イベントビューワでは、メッセージの解析のために、例えば図2のようなイベント内容表示機能を用意しており、検査に有効である。

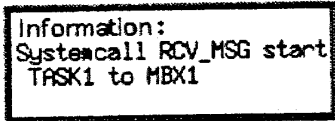


図2 イベント内容表示機能

テストへの適用の例として、PHS 開発時の子機間通信に関するテスト環境を考える。子機（ここでは PHS1 と PHS2 とする）同士の通信の接続テストの手順として、以下を想定する（図3 参照）。

1. 両方の PHS（PHS1 と PHS2）の通信の状況をそれぞれでログにとる。
2. 両者の情報を一つの時系列にまとめたログを作成する。
3. イベントビューワでまとめたログを読み込み、表示をする。

これにより、両方のソフトウェアのタスクなどを同期が取れた状態で観察することができ、テスト時の確認が容易になる。

加えて、イベントビューワのグルーピング機能を用い、実際の通信プロトコルの各層に対応づけ、ログを階層化して表示することを考える。グルーピング機能とは、表示したオブジェクトをグループ化/解除できる機能である。

図3では最上位層の Layer3 だけを表示していたが、例えば図4のように、Layer3 を展開し、その下位層である Layer2 も表示させることで、各層でのイベントの流れや、それぞれの子機において対応している層の間でのイベントのやり取り等を分かりやすく見ることができる。このように、見た

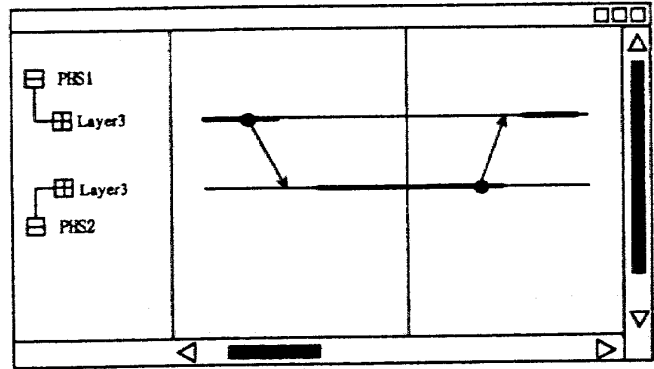


図3 システムテストにおけるイベントビューワの使用例

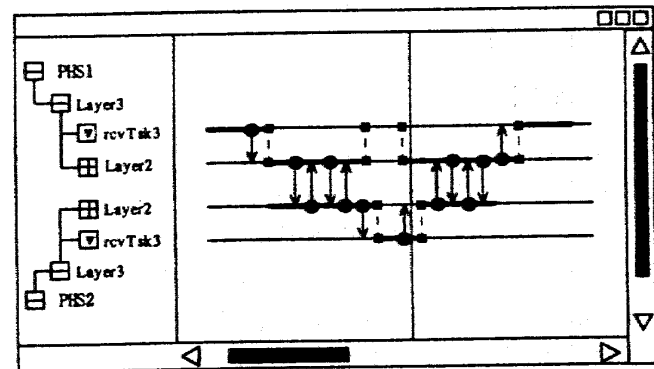


図4 図3の Layer3 を展開したもの

い階層だけを表示させ、対応している層ごとの整理された情報を元にテストを行えば、効率化に役立つと思われる。

以上のように、通信系やC/S系のシステムなど、それぞれが単独のアプリケーションを持ち、通信等を介して相互作用しているシステムの統合テストにも適用することが出来る。

7. まとめ

本報告では、RTOS を用いた組込みシステム開発支援環境として、イベントビューワの開発とそれを用いた効率的なテスト・デバッグ手法について述べた。実行時のタスク・セマフォ状態の変化を表示し、システム挙動の把握、予想外の動きや不正な動きの発見を容易にすることで、より正確なデバッグが可能となる。また、デバッグ時だけでなくテスト工程へ適用していくことで、検査効率を向上させることが出来る。

今後は、今回実装した機能以外の有効な RTOS デバッグ機能や、RTOS 環境下でのユーザプログラムのテスト方式などの検討を行う。また、今回は UNIX 上で実装を行ったが、PC への移行についても検討し、実装していきたい。

参考文献

[1] 宿口雅弘, "組込みシステムのデバッグ手法", 情報処理, Vol.38, No.10, PP. 886-891 (1997)  
 [2] Charles E. McDowell and David P. Helmbold : Debugging Concurrent Programs, in ACM Computing Surveys, Vol.21, No.4 (1989)