組み込みシステムのプラットフォーム標準化による ソフトウェア資産の再利用性向上の評価

佐藤 浩一‡ 豊山 祐一‡ 田中 誠‡ 越塚 登‡‡ 坂村 健‡‡‡
YRP ユビキタス・ネットワーキング研究所‡
東京大学情報基盤センター‡‡ 東京大学大学院情報学環‡‡‡

1. はじめに

近年、携帯電話やデジタル家電をはじめとして組込みシステムのソフトウェア規模が増大し、その開発効率向上が急務になっている。組込みシステムでは、機器制御やユーザインタフェースのリアルタイム性を維持しつつ、通信や画像・音声処理、セキュリティなどの機能実現が求められる。従ってリアルタイム OS（RTOS）上におけるソフトウェア資産の再利用への要望が高い[1]が、現状それは十分に実現されていない。

T-Engine[2]は我々が中心となり国内外 300 社以上の企業と共同して研究開発している、組込みリアルタイムシステムの標準プラットフォームである。T-Engine は、組込みシステムにおけるソフトウェア資産の再利用性向上を最大の目標としている。そして本研究の目的は、T-Engine アーキテクチャにおける組込みシステムソフトウェアの再利用性向上を検証・評価することである。各ハードウェアプラットフォーム間で共通に動作するソフトウェアのソースコード解析、テストスーツプログラムや実用ソフトウェアの移植実験などを通して、異なる CPU を搭載したシステム間でもソフトウェアの高い再利用が可能となったことがわかった。

2. 従来の RTOS 上におけるソフトウェア再利用

図１に組込みシステムの一般的なシステム構成モデルを示す。本研究にて再利用性の評価対象となるソフトウェアは図中で“ミドルウェア”と記した部分である。

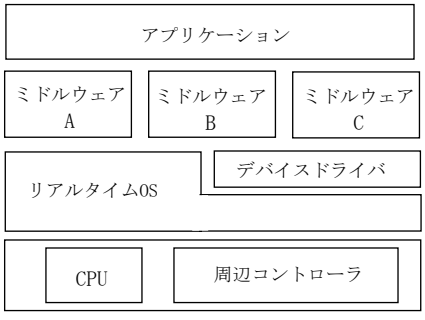

図１　組込みシステムの構成

組込みシステムでは、要求される動作性能、消費電力、アーキテクチャが対象システムにより異なるため、CPU や周辺機器などのハードウェアが多様になるといった特徴がある。そのため、組込みシステムのプラットフォームでは、ベンダ製品であってもオープンなものであっても、ハードウェアの特性に応じて機能のサブセット化や実装依存となる部分が多くあり、標準化の度合いが弱かった。例えば、一般的な RTOS が標準的に提供する機能は、スケジューリング管理や割り込み管理、タスク間の同期通信機能、メモリ管理機能であり、標準化される API が扱うハードウェア資源は CPU とメモリに限定されている。これ以外のハードウェア資源は、システム独自の仕様や実装をもった、デバイスドライバによって抽象化 API が提供される。また RTOS が直接管理する CPU やメモリ部分であっても、機能のサブセット化や最適化などが施され、上位ソフトウェアの再利用が可能なほどに厳密な標準化はなされていない[3]。従って、RTOS 上のミドルウェアを他のプラットフォーム上で再利用するためには、相当量のソースコードの書き換えが必要とされた。特に、調査や試作的に開発する場合であっても多大な費用がかかり、これが組込みソフトウェア開発手法の硬直化を招く結果となっていた。

3. T-Engine アーキテクチャのアプローチ

前章で述べた問題を踏まえ、ソフトウェア開発のコスト低減と期間短縮を実現するため、T-Engine アーキテクチャでは次の目標を設定している。

(1) CPU が異なるシステム間でもミドルウェアのソースコードを修正することなくコンパイルさえすれば再利用が可能なこと。但し、性能を確保するため、仮想マシンやスクリプト言語は使用せず、あくまでもネイティブコードを動作させるものとする。
(2) CPU が同一のシステム間ではミドルウェアがバイナリコードレベルで再利用可能なこと。
(3) システム部分をアプリケーション部分から保護する機構をサポートできること。組込みシステムでも今後セキュリティ対応が必須になると考えている。
(4) 各ミドルウェアのシステムへの着脱が容易なこと。

そして以下に説明するプラットフォームを開発することで上記目標を達成した。図 2 にその構成を示す。

(ア) RTOS をシングルソースコード化した。タスクディスパッチャや割り込みハンドラエントリなどのハードウェア依存部分を除いた大部分を共通のソースコード(C 言語記述)とした。これにより実装依存部分を極力なくすことが可能になる。
(イ) プラットフォームとしてボードレベルで搭載するデバイスインタフェース機能を標準化した。更に、主なデバイスに関して標準ドライバ API 仕様を定めた。ミドルウェアを標準ドライバ API を利用して開発すればそのソースコードを修正することなく各ボードで動作させることが可能になる。表 1 に API を標準化したデバイス例を記す。

"Evaluation of software reusability by standardization of platform for embedded systems", Koichi Sato‡, Yuichi Toyoyama‡, Makoto Tanaka‡, Noboru Koshizuka‡‡ and Ken Sakamura‡‡‡, ‡YRP Ubiquitous Networking Laboratory, ‡‡Information Technology Center, The University of Tokyo, ‡‡‡Graduate School of Interdisciplinary Information Studies, The University of Tokyo.

(ウ) タスク、セマフォをはじめ全てのオブジェクトについて ID 番号をオブジェクト生成時に自動的に RTOS 側で割り当てる仕様とした。逆に ID 番号を指定してオブジェクトを生成する仕様は一切排除した。これにより ID 番号の割り当て調整のためのプログラム修正が不要になる。

(エ) 各タスクに実行リングレベルが指定できるようにし、またメモリ管理機能においてメモリ確保時の保護リングレベルを設定できる RTOS 仕様とした。これによりシステムの保護機構が実現できる。

(オ) ミドルウェアの管理機能を RTOS でサポートした。各ミドルウェアは本機能を使って登録や削除ができる。更に本機能はシステム全体として実行ブレーク処理や低消費電力モード移行に対応する機構をもサポートしており、登録されたミドルウェアがシステム全体と連動して動作できるようになる。

(カ) 複数のミドルウェアを組み合わせた際にグローバルシンボルが二重定義にならないようシンボル名の命名規約(T-Format)を定めた。

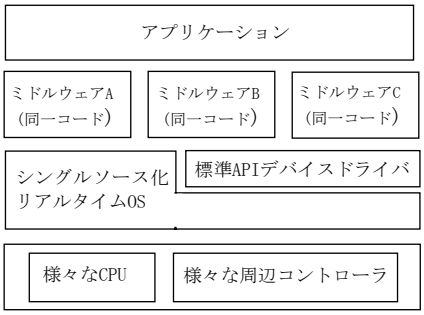

図2 T-Engineアーキテクチャ

表1 API仕様を標準化した主なデバイス

シリアルデバイス
ディスクデバイス
キーボード/ポインティングデバイス
ディスプレイデバイス
コンソールデバイス
LANデバイス
USBデバイス(マネージャ)
PCMCIAデバイス(マネージャ)
オーディオデバイス

4. ミドルウェア再利用性の評価

下記評価によって T-Engine アーキテクチャ上でミドルウェアの再利用が実現できることを確認した。

(1)本 RTOS のコード解析

表 2 に本 RTOS を実装したハードウェアプラットフォーム種類と RTOS における共通ソースコード部分の割合を示す。共通部分の割合はバイナリコードにして 90%程度と高比率であった。RTOS の制御アルゴリズムやデータ構造はほぼ全て共通部分に含まれており、上位ソフトウェアに同一動作機能を提供できる構成になった。表 3 に示した部分以外は共通のソースコードになっている。

(2) テストスーツプログラムによる評価

本 RTOS がサポートしている約 110 個のシステムコールに対し、様々にパラメータを振って各システムコールを発行する動作確認プログラム(約 3000 回のシステムコールを発行)を動作させた。このプログラムは、タスク例外ハンドラのエントリ部分(この部分はアセンブラ記述する必要がある)以外は全てC言語で記述されている。そして表 2 の全てのプラットフォームにおいて、同一ソースの動作確認プログラムが、CPU 毎にコンパイルするだけで、それぞれ同一の動作をすることが確認できた。

(3) 実際のミドルウェアによる評価

以下のミドルウェアについて同一ソースコードをコンパイルするだけで表 2 の全プラットフォーム(一部ハードウェア制限で動作しない場合を除く)で動作することを確認した。また必要なミドルウェア単位でシステムに追加、削除できることも確認した。

- TCP/IP プロトコルスタック
- ファイルシステム
- ウインドウシステム

なお、TCP/IP スタックでは LAN ドライバ、ファイルシステムではディスクドライバ、ウインドウシステムではディスプレイドライバ及びポインティングデバイスドライバについて、それぞれ標準 API 仕様に従ったデバイスドライバを各プラットフォーム上で動作させている。

表2 開発したプラットフォーム

搭載 CPU	リアルタイム OS の共通コード割合
SH3, SH4	90.4%
ARM7, ARM9	90.6%
MIPS 系	87.4%
M32R	91.5%

表3 本RTOSの非共通化部分

CPU初期化、タスクディスパッチャ キャッシュ制御、割り込みエントリ 高級言語対応エントリ、レジスタ設定/参照 SVCエントリ、システムタイマ初期化

5. まとめ

組込みシステムにおけるソフトウェア資産の再利用を実現する T-Engine アーキテクチャの有効性検証・評価を行った。我々は既に ARM、MIPS、SH、M32R などの CPU についてプラットフォームを開発しており、それらの間でソフトウェア部品の再利用性が確認できた。T-Engine アーキテクチャにより、今後組込みシステムの開発効率向上に大きく貢献できると考えている。

謝辞

本研究の一部は通信・放送機構からの委託研究に基づき行われたものである。

参考文献

[1] 社団法人日本システムハウス協会、社団法人トロン協会「2002 年度組込みシステムにおけるリアルタイム OS 利用動向に関するアンケート調査報告書」

[2] Ken Sakamura and Noboru Koshizuka: “T-Engine: The Open Realtime Embedded Systems Platform”, IEEE MICRO, Vol. 22, No. 6, December, 2002.

[3] K. Sakamura, Ed. “μITRON 3.0: An Open and Portable Real-time Operating System for Embedded Systems: Concept and Specification”, IEEE CS Press, 1998.