ARM 64bit プロセッサを用いた 高信頼デュアル OS 環境の開発と評価

利長 勇児[1,a)]　野澤 成樹[1]　本田 晋也[2,b)]

概要：近年の組込みシステムは高機能性を求められることが多くリアルタイム性を保ちつつ汎用 OS を利用したいという要求がある．一方，64bit プロセッサを搭載した SoC が登場し，組込みシステムにも普及すると予想される．本研究では，ARM 64bit プロセッサ上でリアルタイム OS と汎用 OS の同時実行環境を開発した．評価の結果，汎用 OS からリアルタイム OS への応答性のオーバヘッドは 0.8 μ s となることを確認した．

キーワード：組込みシステム，デュアル OS，64bit プロセッサ，信頼性

1. はじめに

近年，組込みシステムは多様化しており，車載システムや家電製品などの分野では，リアルタイム性や信頼性に加え，高機能性への要求が高まっている．こうした要求を同時に実現する方法として，リアルタイム OS と汎用 OS を同時に実行し，両者の特徴を併せ持ったシステムを構築する方法が考えられる．例えば SafeG[6] は，リアルタイム OS と汎用 OS を同時実行し，加えてセキュリティ支援ハードウェアによるリアルタイム OS の保護が可能な高信頼デュアル OS 環境である．

一方で，近年組込みシステム向け 64bit プロセッサが登場した．ARM は従来の命令セットを一新することで消費電力を改善する [2]．また，メモリ空間を拡張することで大容量メモリを搭載するサーバやスマートフォンなどに普及し，それに伴い，Linux やコンパイラ技術の進化 [3] も予想できる．これら技術を組込みシステムに流用することが考えられる．

そこで本研究では，64bit プロセッサを用い，高信頼なデュアル OS 環境を構築した．また，64bit プロセッサ上へリアルタイム OS を移植し評価を行った．

2. ARM 64bit プロセッサ

組込みシステム向けの 64bit プロセッサとして，ARMv8 アーキテクチャ [1] がある．従来の 32bit アーキテクチャとは命令セットが異なり，汎用レジスタの数とデータ幅が増えた．アドレス空間は最大 48bit，256TB のデータを扱える．従来の例外や特権に関する様々なモードは，4 段階の Exception Level(EL) に整理された．一方，従来と同様にプログラムはセキュア，またはノンセキュア状態で実行され，セキュアに設定したデータやデバイスはノンセキュア状態で実行するプログラムから保護できる．**図 1** に示すように，アプリケーション，OS，仮想化環境，セキュアファームウェアが別々のレベルとセキュア状態で動作する．セキュアとノンセキュアで 2 つの OS を実行することで，デュアル OS 環境を構築することが考えられる．

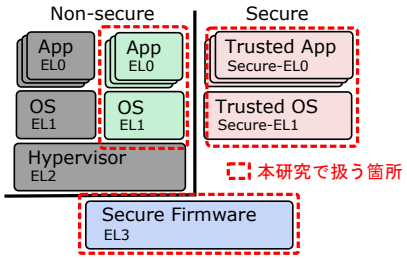

図 1　ARM 64bit アーキテクチャ上でのデュアル OS 構成

3. ARM 64bit 向けデュアル OS 環境の開発

3.1 開発方針

ARM 64bit プロセッサ上でリアルタイム OS と汎用 OS の同時実行環境を構築するには，従来の SafeG などのデュアル OS 環境を移植する方法が考えられる．しかし，Errata 対応が必要 [4] であり，また，OS の切り替え処理はプロセッサ依存な処理が含まれるのでコード変更量が多くなる．そこで本研究では，プロセッサベンダが提供する ARM Trusted Firmware(ATF)[4] を利用する．ATF は，図 1 の EL3 で動作し，EL1 のセキュアとノンセキュアでそれぞれプログラムを実行可能な環境である．セキュアでリアルタイム OS，ノンセキュアで汎用 OS を実行することで，デュ

1　コニカミノルタ株式会社
2　名古屋大学大学院 情報科学研究科
a)　yuji.toshinaga@konicaminolta.com
b)　honda@ertl.jp

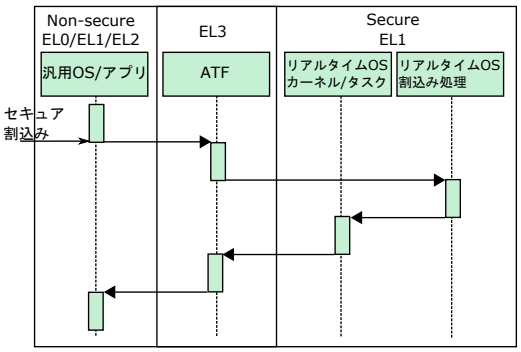

図 2 リアルタイム OS と汎用 OS 実行時のシーケンス

表 1 評価環境

評価ボード	Juno ARM Development Platform
プロセッサ	Cortex-A53 のシングルコア
プロセッサ周波数	850MHz
キャッシュ	L1 32KB, L2 1MB
リアルタイム OS	ASP カーネル (1.9.2, 64bit へ移植)
汎用 OS	Linux (3.15.0, 64bit)

アル OS 環境を構築する．

3.2 デュアル OS 環境の開発

ATF は，セキュアとノンセキュアでプログラムを同時実行し，割込みとサービスコールにより処理を切り替える．ノンセキュア時にセキュアが管理する割込みが発生すると ATF は切り替え処理を行い，セキュアへ処理を移す．セキュア側は割込み処理を行い，割込みハンドラ末尾で ATF サービスコールを呼び ATF へ処理を移す．すると，再び ATF は切り替え処理を行いノンセキュアへ処理を戻すことができる．本研究では，ATF や OS を一部変更し同様の処理を汎用 OS とリアルタイム OS 間で行う．

ATF の変更点は，割込み復帰処理の準備である．リアルタイム OS は割込み処理後に例外復帰命令で割込みハンドラからカーネルやタスクに処理を戻す．ATF の切り替え処理に，例外復帰命令のための準備処理を加え割込みハンドラからカーネルやタスクへ移行できるようにする．

リアルタイム OS の変更点は，ベクタテーブルの定義と，アイドル処理である．割込み発生時に ATF からリアルタイム OS へジャンプするために，ベクタテーブルを定義する．また、リアルタイム OS の処理が終了した後，汎用 OS へ処理を戻すためにリアルタイム OS のアイドル処理で ATF サービスコールを呼ぶ．シーケンスを**図 2** に示す．このようにして，ATF 上で汎用 OS とリアルタイム OS を同時実行する環境を構築する．

4. 評価

ATF 上で汎用 OS とリアルタイム OS が動作することを確認し，割込み応答時間の評価を行った．使用した評価環境は**表 1**，評価結果は**図 3** である．

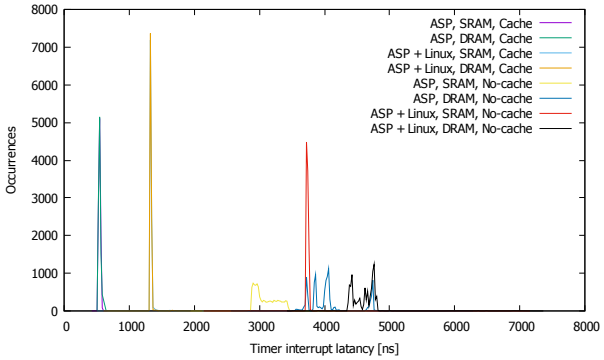

図 3 タイマ割込み応答時間

リアルタイム OS として，TOPPERS/ASP カーネル（ASP カーネル）[5] を ARM 64bit プロセッサのセキュアで動作するように移植し，使用した．

タイマ割込みを 10000 回発生させ，ASP カーネル単体で実行した場合と，Linux と ASP カーネルを同時実行した場合の割込み応答時間を計測した．ATF は SRAM 上でキャッシュを有効に，Linux は DRAM 上でキャッシュを有効にして実行した．ASP カーネルは配置場所，キャッシュ有無の条件を変更しながら実行した．ASP カーネルを SRAM 上でキャッシュを有効にした場合，ASP カーネル単体実行時は平均 0.5 μ s となり，Linux と ASP カーネル同時実行時は平均 1.3 μ s となった．オーバヘッドは 0.8 μ s であることが確認できた．

5. 今後の課題

本研究では ATF を利用したデュアル OS 環境を構築した．現状では OS 間の通信手段がない．そこで，SafeG が持つ通信機能を参考に同様の通信機能を構築する方法が考えらえる．また，ARMv8 の他種類のコアでも動作可能にし，性能差を計測することが考えらえる．

謝辞 本研究を進めるにあたり，貴重なご意見を頂いた名古屋大学大学院情報科学研究科高田・本田研究室の方々に厚く御礼申し上げます．

参考文献

[1] ARM Ltd.: *ARM Architecture Reference Manual ARMv8, for ARMv8-A architecture profile.*
[2] ARM Ltd.: WHITE PAPER Technology Preview: The ARMv8 Architecture.
[3] Beyls, K.: LLVM AArch64, *FOSDEM LLVM dev room*, ARM Ltd. (2015).
[4] Thoelke, A.: ARM Trusted Firmware for ARMv8-A, *LCU13*, ARM Ltd. (2013).
[5] TOPPERS プロジェクト：TOPPERS/ASP カーネル，`https://www.toppers.jp/asp-kernel.html`.
[6] 中嶋健一郎，本田晋也，手嶋茂晴，高田広章：セキュリティ支援ハードウェアによるハイブリッド OS システムの高信頼化，*The IEICE Transactions on Information Systems*, Vol. J93-D, No. 2, pp. 75–85 (2010).