

TOPPERS 活用アイデア・アプリケーション開発 コンテスト

部門 : アプリケーション開発部門

作品のタイトル : TOPPERS/ATK2 カーネル向け実機レス環境(athrill2)

作成者 : 森崇((株)永和システムマネジメント)

共同作業者 :

対象者 : TOPPERS/ATK2 カーネル利用者

使用する開発成果物 : TOPPERS/ATK2 カーネル

目的・狙い

■目的

TOPPERS/ATK2 カーネルを実機レス開発環境下で利用できるようにする。

■狙い

AUTOSAR 普及が進む中、手軽に AUTOSAR スタックを結合⇒動作確認できる OSS・実機レス環境のニーズは高まると予想される。そのため、TOPPERS/ATK2 カーネル向け実機レス環境を TOPPERS/OSS として公開することは、TOPPERS/AUTOSAR スタック普及の一助になると考える。

アイデア/アプリケーションの概要

TOPPERS/ATK2 カーネル向け実機レス環境を開発する。今回は、以下の AUTOSAR/OS スケーラビリティクラスに対応可能な実機レス環境を提供する。

- ・ SC1(保護なし, シングルコア)
- ・ SC1-MC(保護なし, マルチコア)
- ・ SC3(メモリ保護あり, シングルコア)

また、上記の動作確認用に、サンプルプログラムをあわせて提供する※。

※SC1/SC1-MC は OS 移植し、SC3 はサンプル確認プログラムを作成する。

1. AUTOSAR 開発で実機レス環境を利用する際の課題

AUTOSAR 普及が進む中、車載向け開発においても、机上で全体結合・動作確認できる実機レス環境は、問題の早期検出を可能にするだけでなく、ソフトウェア保守の視点(リグレッション確認)からもそのニーズは高いと考える。

一方で、実機レス環境の課題としては、以下が挙げられる。

■既製の実機レス環境を利用する場合

開発資金・期間が潤沢にあれば、既製品を使用することで実機レス環境を使用する効果は十分に享受できるが、一般的に実機レス環境は高額になりえるため、少なくとも以下に示す課題があると考えられる。

No.	分類	課題
1	導入コスト	実機レス環境は高額(数百万～)。
2	スケーラビリティ	既製の実機レス環境を利用する場合は、ライセンス制約により開発をスケールさせることができない可能性が高い。
3	サポート	環境面で未対応機能等がある場合は個別サポートが必要となる。

■OSS の実機レス環境を利用する場合

代わって、OSS 環境の場合はライセンス制約もなく、自社固有の機能改修等が可能となるが、以下に示す課題が残る。

No.	分類	課題
1	導入コスト	OSS 環境の場合は、マニュアル等が整備されていない場合もあり、ソース/ビルド/使用方法の理解に時間がかかる可能性がある。
2	車載向け CPU アーキ	既存の OSS・実機レス環境として QEMU/Skyeye 等があるが車載向け CPU である V850/RH850 は未対応である。
3	AUTOSAR/OS スケーラビリティクラス	AUTOSAR/OS スケーラビリティクラスは SC1～SC4 と幅広くあり、すべてのクラスに対応した実機レス環境は存在していない(athrill は SC1 までしか対応しきれていない)。

2. 提案内容

2017年に提案させて頂いた車載向け OSS・実機レス環境である athrill は、AUTOSAR 向けの環境として不十分である。よって、TOPPERS/ATK2 カーネル向け実機レス環境として、先述の課題に対して athrill をバージョンアップした athrill2 を新たに提案する。

No.	分類	対応内容												
1	導入コスト	athrill 機能マニュアルの整備(以下のサイトで公開中) https://qiita.com/kanetugu2018/items/cf3dea16710a3f0737e8												
2	車載向け CPU アーキ	athrill CPU のバージョンアップ(v850e1 ⇒ v850e2v3). 本バージョンアップによる athrill2 の CPU 命令実装率は以下の通りである(未サポート命令は Figure 1 を参照). <table border="1" data-bbox="544 840 1350 987"> <thead> <tr> <th>CPU アーキ</th> <th>命令数</th> <th>実装数</th> <th>実装率</th> </tr> </thead> <tbody> <tr> <td>V850(v850e2v3)</td> <td>132</td> <td>127</td> <td>96.2%</td> </tr> <tr> <td>RH850(v850e3v5)</td> <td>146</td> <td>127</td> <td>87.0%</td> </tr> </tbody> </table> RH850(v850e3v5)は上位互換を保証しているため、v850e2v3 の命令実装率を上げることで、RH850 の命令実装率も必然的に上がり、athrill2 の CPU 命令は RH850 に極めて近くなる。	CPU アーキ	命令数	実装数	実装率	V850(v850e2v3)	132	127	96.2%	RH850(v850e3v5)	146	127	87.0%
CPU アーキ	命令数	実装数	実装率											
V850(v850e2v3)	132	127	96.2%											
RH850(v850e3v5)	146	127	87.0%											
3	AUTOSAR/OS スケーラビリティクラス	v850e2/mn4 のハードウェア仕様をベースにして、athrill のコア数拡張(2 コア)、メモリ保護機能を追加する。本対応でサポート可能となるスケーラビリティクラスは以下の通り(詳細は 3 章を参照). <ul style="list-style-type: none"> ・ SC1(保護なし, シングルコア) ・ SC1-MC(保護なし, マルチコア(2 コア)) ・ SC3(メモリ保護あり, シングルコア) 												

※0内のバージョン表記は、gcc のターゲット・プロセッサオプション(-m)で指定するものである

athrill2 の未サポート命令		
命令フォーマット	V850(v850e2v3)	RH850(v850e3v5)
1	SYNCE, SYNCM, SYNCP	SYNCL, DBTRAP
2	CALLT	—
7	—	LOOP, Bcond, ROTL, LDL.W, STC.W
9	—	BINS
10	CTRET	SNOOZE
11	—	PUSHSP, POPSP, JARL
14	—	LD.DW, ST.DW

Figure 1. athrill2 の未サポート命令

3. athrill2 の AUTOSAR/OS スケーラビリティクラス適応

先述の通り、v850e2/mn4 のハードウェア仕様をベースにして、athrill のコア数拡張(2 コア)、メモリ保護機能を追加する。本対応により、athrill2 が以下の AUTOSAR/OS スケーラビリティクラスに適応できるようにすることが今回の目標である。

- ・ SC1(保護なし, シングルコア)
- ・ SC1-MC(保護なし, マルチコア(2 コア))
- ・ SC3(メモリ保護あり, シングルコア)

上記目標に対して、今回提案する athrill2 がどの程度 AUTOSAR/OS スケーラビリティクラスに適応できたかを客観的に確認できるようにするため、TOPPERS/ATK2 の公式サイトで公開されている「次世代車載システム向け RTOS ハードウェア要求仕様書 Ver.3.0.1」を参照し、athrill2 のハードウェア要求仕様準拠率を調査することとした¹。

本調査の結果概要は Figure 2 のとおりであり、最小要求(MR)はほぼ満たしていることがわかった²。

一方、SR(標準要求)/AR(性能改善要求)についての網羅度は低い。これらの要因については Figure 3 にそれぞれ見解を示すが、今回目標とする機能範囲(SC1, SC1-MC(2 コア), SC3) の実機レス環境を提供するという点では問題ではないと判断している。

スケーラビリティクラス	MR(最小要求)	SR(標準要求)	AR(性能改善要求)
SC2	100%(2/2)	50%(1/2)	0%(0/1)
SC3	100%(10/10)	50%(1/2)	0%(0/2)
SC1-MC	100%(5/5)	67%(6/9)	17%(1/6)
SC2-MC	80%(4/5)	9%(1/11)	14%(1/7)
SC3-MC	100%(6/6)	64%(7/11)	13%(1/8)

※0内の値は、「要求準拠数/要求数」である。

Figure 2. TOPPERS/ATK2 ハードウェア要求仕様の準拠割合

¹ 調査結果詳細は本サイト(<https://qiita.com/kanetugu2018/items/521bc56b39b82c1c8734>)を参照

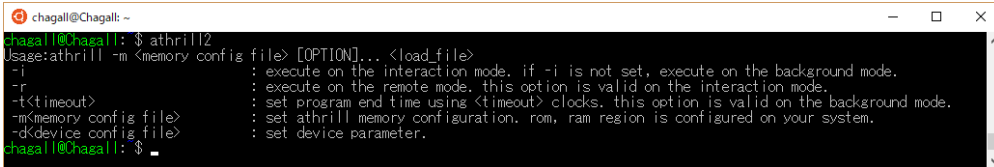
² athrill2 のハードウェア仕様は存在していないため、本評価では、athrill2 の設計・実装担当者がハードウェア要求仕様を満たしているかどうか設計/ソースレベルで確認した。

HW要素	要件レベル	要件番号	対応 SC	サポート対象	見解
タイマ	MR	HW023	SC2-MC	×	SC2 は今回未サポート. athrill2 のタイマ数拡張の技術的難易度は低い.
コア間 排他制 御機構	SR AR	HW005 HW008	SCx-MC	×	コア数が 3 以上の場合に必要となる要件. athrill2 のコア数拡張の技術的難易度は低い.
メモリ	SR AR	HW012 HW013 HW014	SCx-MC	○	athrill2 はローカルメモリ/プライベートメモリを 仮想化対象としない方針としている. 理由は, 本 機能をサポートしなくても性能的なデメリットは 発生しないためである.
キャッ シュ	SR	HW015	SCx-MC	○	athrill2 はキャッシュメモリを仮想化対象としない 方針としている. 理由は, 本機能をサポートし なくても性能的なデメリットは発生しないため である.
タイマ	SR	HW024 HW026 HW027 HW028	SC2 SC2-MC	×	ノンマスカブル割り込みでのタイマ割り込み発生 機構は SC2 対応時に検討する予定.
MPU	SR AR	HW039 HW040	SC3 SC3-MC	×	athrill2 のメモリ保護機構をマルチコア向けに拡 張する際に, 保護領域設定機能拡張を実施する予 定.
コア ID	AR	HW010	SCx-MC	○	マルチコア OS 実装の性能改善要件であり, 実機 レス環境機能実現する上で MUST 要求ではない と判断.
割り込 み	AR	HW019	SCx-MC	○	↑
MPU	AR	HW041	SC3 SC3-MC	○	MPU の OS 実装に対する性能改善要件であり, 実 機レス環境機能実現する上で MUST 要求ではな いと判断.

Figure 3. athrill2 のハードウェア要件未対応に対する見解

4. athrill2 の動作環境およびインストール方法

athrill2 の動作環境およびインストール方法は athrill と同じである。

項目	参照サイト
動作環境	https://qiita.com/kanetugu2018/items/1f2ef93c9e1fa7a29f97#%E5%8B%95%E4%BD%9C%E7%92%B0%E5%A2%83
事前準備	https://qiita.com/kanetugu2018/items/1f2ef93c9e1fa7a29f97#%E4%BA%8B%E5%89%8D%E6%BA%96%E5%82%99
インストール方法	https://qiita.com/kanetugu2018/items/1f2ef93c9e1fa7a29f97#%E3%82%A4%E3%83%B3%E3%82%B9%E3%83%88%E3%83%BC%E3%83%AB%E6%96%B9%E6%B3%95
インストール確認方法	athrill2 を空打ちして以下のメッセージが出力されることを確認する 

5. 動作確認

本節では、athrill2 がサポート対象とするスケラビリティクラスの動作確認を行う。

スケラビリティクラス	動作確認内容
SC1	TOPPERS/ATK2-SC1 を athrill2 向けに移植し、サンプルプログラムの動作確認を行う。
SC1-MC	TOPPERS/ATK2-SC1-MC を athrill2 向けに移植し、サンプルプログラムの動作確認を行う。
SC3	SC3 のハードウェア要件を満たしていることを確認できるサンプルプログラムを作成し、動作確認を行う※。

※TOPPERS/ATK2-SC3 の移植は時間的な余力がなかったため、別の機会で行う予定。

5.1. SC1

SC1 の動作確認内容を以下に示す。

項目	内容	確認結果
プログラム配置場所(github)	https://github.com/tmori/athrill/tree/master/sample/os/atk2-sc1_1.4.2/OBJ	—
athrill2 CPU 設定	100MHz	—
OS ハードウェアカウンタ設定	10ms/clock	—
動作確認[1]	atk2-sc1 のカーネル起動	OK
動作確認[2]	周期アラームコールバック起動確認(500ms 周期)	OK
動作確認[4]	アラーム実行をキャンセル	OK
動作確認[3]	タスク起動確認(Task3)	OK

上記の動作確認結果詳細については、以下のサイトを参照のこと。

<https://qiita.com/kanetugu2018/items/d07dac56c4f57ce10f65>

5.2. SC1-MC

SC1-MC の動作確認内容を以下に示す.

項目	内容	確認結果
プログラム配置場所(github)	https://github.com/tmori/athrill/tree/master/sample/os/atk2-sc1-mc_1.4.2/OBJ	—
athrill2 CPU 設定	100MHz	—
OS ハードウェアカウンタ設定	10ms/clock	—
動作確認[1]	atk2-sc1-mc のカーネル起動	OK
動作確認[2]	周期アラームコールバック 起動確認(500ms 周期)	OK
動作確認[4]	アラーム実行をキャンセル	OK
動作確認[3]	タスク起動確認 (core0 の Task1 から core1 の Task9 を起動)	OK

上記の動作確認結果詳細については、以下のサイトを参照のこと.

<https://qiita.com/kanetugu2018/items/b43c5715124f42a0393d#%E3%82%B5%E3%83%B3%E3%83%97%E3%83%AB%E3%83%90%E3%82%A4%E3%83%8A%E3%83%AA%E3%82%92%E5%AE%9F%E8%A1%8C%E3%81%99%E3%82%8B>

5.3. SC3

SC3 の動作確認内容を以下に示す.

項目	内容	確認結果
プログラム配置場所 (github)	https://github.com/tmori/athrill/tree/master/sample/athrill2/mptest2	—
athrill2 CPU 設定	100MHz	—
メモリ保護構成	カーネル領域 : 1 個(コード/データ) ユーザ領域 : 2 個(コード/データ) ユーザ共有領域 : 1 個(データ) (領域割り当ておよびアクセス権設定詳細は, Figure 4 を参照)	—
動作確認[1]	非特権モード(USER1)で, 自領域およびユーザ共有領域のみ読み書きおよび命令実行可能であることを確認する. カーネル領域および別ユーザ領域(USER2)の領域については, 読み書き/命令実行を行うと, CPU 例外が発生することを確認する.	OK
動作確認[2]	非特権モード(USER2)で, 自領域およびユーザ共有領域のみ読み書きおよび命令実行可能であることを確認する. カーネル領域および別ユーザ領域(USER1)の領域については, 読み書き/命令実行を行うと, CPU 例外が発生することを確認する.	OK
動作確認[3]	特権モードで, 全領域の読み書きおよび命令実行可能であることを確認する.	OK

上記の動作確認結果詳細については, 以下のサイトを参照のこと.

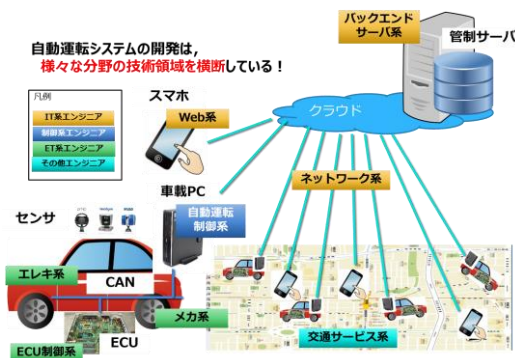
<https://qiita.com/kanetugu2018/items/ea7069ac2b9530261f16>

アドレス	種類	領域割り当て内容	アクセス権設定		
			実行	読み込み	書き込み
0x00000000	ROM	KERNEL コード領域	禁止	禁止	禁止
0x00080000	—	(未割当領域)	禁止	禁止	禁止
0x00100000	ROM	USER1 コード領域	許可(USER1)	許可(USER1)	禁止
0x00180000	—	(未割当領域)	禁止	禁止	禁止
0x00200000	ROM	USER2 コード領域	許可(USER2)	許可(USER2)	禁止
0x00280000	—	(未割当領域)	禁止	禁止	禁止
:					
0x06000000	RAM	KERNEL データ領域	禁止	禁止	禁止
0x06080000	—	(未割当領域)	禁止	禁止	禁止
0x07000000	RAM	USER1 データ領域	禁止	許可(USER1)	許可(USER1)
0x07080000	—	(未割当領域)	禁止	禁止	禁止
0x08000000	RAM	USER2 データ領域	禁止	許可(USER2)	許可(USER2)
0x08080000	—	(未割当領域)	禁止	禁止	禁止
0x09000000	RAM	USER 共有データ領域	禁止	許可(全 USER)	許可(全 USER)
0x09080000	—	(未割当領域)	禁止	禁止	禁止
:					

Figure 4. 保護領域の割り当てとアクセス権設定内容

6. 今後の展望

今回提案させて頂いた **athrill2** は車載向け ECU の実機レス環境である。一方、世の中の動きに目を向けると、自動車業界はすでに自動運転に向けて大きく舵を切っており、東京オリンピックを 1 つの目標としてその開発を加速させている。自動運転システムは、下図に示すように、ECU だけでなく、様々な分野の技術を統合して成り立つシステムであり、その開発難易度は現状よりもはるかに高いものといえる(ET と IT の統合システム)。



このような背景を鑑みた時、**athrill2** は ECU から外の世界とつながり、より大きなシミュレーション環境を構成する一要素技術として進化していく必要がある。

ここで話を転じ、この進化の方向性を考えるうえで、日本の江戸期に流行った「箱庭」という考え方が役に立つと思われる。箱庭は、庭園や名勝等の構成要素をミニチュア要素として箱の中で組み上げ、模擬的に造る景観を楽しむものであった。その日本的な発想をベースにして、自動運転システムの構成要素を箱庭(実機レス環境)の上で手軽に開発できる「箱庭 SDK」というものがあれば、次世代実機レス環境として大変役立つものになるのではないかと思う。

箱庭 SDK があることで享受できる効果を下図に示すが、**athrill2** は ET 系の技術要素と IT 系の技術要素をつなげる重要な役割を演じると考えている。この箱庭構想のもとに **athrill2** を進化させるとともに、箱庭そのもののベースを作り、その成果を TOPPERS/OSS 活動を通して情報共有していきたいと思う。

箱庭SDKの導入効果

- 全体結合しないと見えない問題を早期検出できる
- 各エンジニアは自社から手軽に遠隔結合確認できる

