Vol.2017-ARC-225 No.12
Vol.2017-SLDM-179 No.12
Vol.2017-EMB-44 No.12
2017/3/9

車載システム向け軽量パーティショニング機構

鈴木 豪仁[1]　本田 晋也[1]

概要：
保護 RTOS を利用したパーティショニング機構は実現コストが高く，ローコストが求められるシステムへには適していない．本論文では，車載システムの中でも特にローコストが要求されるシステムに適したパーティショニング機構である DefensiveZone の見直しと拡張を実施した．DefensiveZone は MPU や保護 RTOS を利用せず，追加の保護ハードウェアモジュールと簡易的な仮想マシンによって保護を実現した．性能評価の結果，保護 RTOS に比べて OS 実行や信頼関数呼び出しのオーバヘッドを小さくできることを確認した．

Lightweight partitioning architecture for automotive systems

Suzuki Takehito[1]　Honda Shinya[1]

Abstract:
Partitioning using protection RTOS has high cost. Therefore, it is not suitable for systems requiring low cost. This paper introduce DefensiveZone, that is partitioning architecture suitable for low cost automotive systems. DefensiveZone realized protection with additional protection hardware and simple virtual machine without using MPU and protection RTOS. As a result of performance evaluation, it could be confirmed that DefensiveZone has less overhead for OS execution and calling trusted function than protection RTOS.

1. はじめに

現在，ECU（Electronic Control System）開発におけるパーティショニング機構の主流はメモリ保護機能を持ったRTOS（保護 RTOS）を利用したものである．ただし，保護 RTOS を利用することで OS 実行オーバヘッドが増大し，実行性能は低下する．この性能低下を補うためには高性能なプロセッサが必要となるため，プロセッサコストの増大につながる．保護が必要とされるのは，パワートレイン系のように高機能な ECU に限らない．ボディ系のように低コストな実現が求められる ECU でも，パーティショニング機構への対応が求められる機会は今後増加すると予想される．しかし，保護 RTOS を用いたパーティショニングは前述のように実現コストが大きい．車載システム向けのパーティショニング機構は他にも研究 [1] が進められているが，保護 RTOS と同様に MPU（Memory Protection Unit）が必要となるため，コスト問題の解決には至っていないと考えられる．そのため，低コストで保護を実現できるパーティショニング機構が必要とされている．

我々は，ISO26262 で定められているパーティショニングを低コストで実現することを目的としたパーティショニング機構である DefensiveZone を研究開発している．DefensiveZone は，保護ハードウェアモジュールの追加によってパーティショニングを実現する．システムの実行状態の管理を追加ハードウェアのひとつが担うため，プロセッサの特権モードや MPU を搭載していないプロセッサと保護機能を持たない RTOS でも DefensiveZone の利用が可能であり，実現コストを抑えることが可能である．しかし，DefensiveZone にはいくつかの課題が存在していた．具体的には，パーティション切り替えがソフトウェアとハードウェアの間で非同期であるために，切り替え時に保護が無効となるタイミングが存在してしまうことや，パーティション間の通信機能が乏しい上に実行オーバヘッドが大きいことが挙げられる．

[1] 名古屋大学 大学院情報科学研究科
Graduate School of Information Science Nagoya University

Vol.2017-ARC-225 No.12
Vol.2017-SLDM-179 No.12
Vol.2017-EMB-44 No.12
2017/3/9

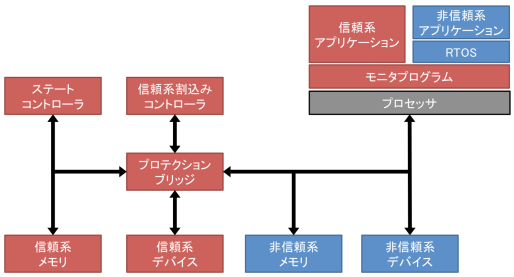

図 1 DefensiveZone

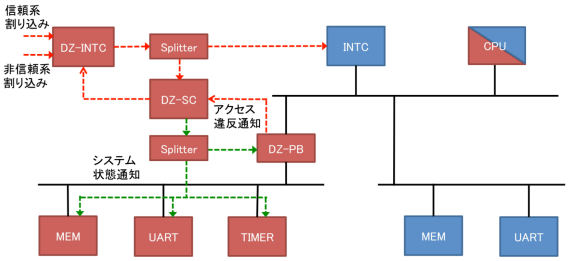

図 2 DefensiveZone のハードウェアアーキテクチャ

我々はこれらの課題に対する解決策を検討し実装した．パーティション切り替えが非同期である問題に対しては，同期的なパーティション切り替えを実現するゲートウェイ機構を実装した．パーティション間通信に関しては，低オーバヘッドで実行可能な信頼関数呼出し機構や，非信頼系が信頼系に依頼していた処理を非信頼系自ら実行するために信頼系の割込みを一時的に禁止する機構を実装した．また，対策を実装し拡張した DefensiveZone の性能評価を行い，保護 RTOS との比較を実施した．

2. DefensiveZone

本節では，本研究の研究対象である DefensiveZone を解説する．特に，DefensiveZone のソフトウェアやハードウェアなど過去の研究 [2] で実現された部分について述べる．

2.1 概要

DefensiveZone は，低コストな車載システム向けのパーティショニング機構である．実現コストを低減するため，以下の要件に従って設計されている．

- **プロセッサ**
  - 内部仕様が不明な既存製品に変更を加えず再利用可能
  - 特権モードが利用できないプロセッサを利用可能
  - MPU を搭載していないプロセッサを利用可能
- **RTOS**
  - 既存品を再利用可能
  - メモリ保護機能を利用しない

これらの要件に沿って，本研究ではプロセッサに Altera 社製 FPGA 用ソフトコアプロセッサである Nios Ⅱ[3] を採用した．Nios Ⅱ にはレジスタバンクが搭載されている．DefensiveZone ではこの機能を利用して非信頼系はバンク 0，信頼系はバンク 1 と使い分けている．また，非信頼系の RTOS には TOPPERS/ATK2-SC1[4] を採用した．

DefensiveZone は，図 1 のように追加の保護ハードウェアモジュールと簡易的な仮想マシンによって保護を実現している．DefensiveZone がサポートするのは，信頼系と非信頼系という 2 種類のパーティションのみであり，非信頼系にのみ OS が搭載されることを想定している．

2.2 ハードウェア構成

DefensiveZone のハードウェア構成を図 2 に示す．DefensiveZone では，メモリや各種デバイスは信頼系と非信頼系にそれぞれ割り当てられる．図 2 中で赤色のデバイスは信頼系に属しており，非信頼系実行中にアクセスが禁止されている．一方で青色の非信頼系のデバイスは信頼系実行中でも自由にアクセスが可能である．DefensiveZone の実現に必要となる保護ハードウェアモジュールは以下の 3 種類である．

- **信頼系割込みコントローラ（DZ-INTC）**
  入力された信頼系割込みの中で最も優先度が高いものを非信頼系の割込みコントローラおよびステートコントローラへと送信する．このとき出力される割込みは NMI として扱われる．
- **ステートコントローラ（DZ-SC）**
  DefensiveZone ではプロセッサの特権モードを利用していないため，システムが実行している系の管理が別途必要となる．ステートコントローラはこの管理を担当するモジュールであり，システムを構成する他のモジュールへとシステムの実行状態の通知を行う役割も持つ．内部に以下のレジスタを持つ．
  - **システム状態レジスタ（REG_WORLD レジスタ）**
    システムの実行状態が保存されるレジスタである．信頼系実行中は 1，非信頼系実行中は 0 が設定される．非信頼系実行中に信頼系割込みコントローラから信頼系割込み発生の通知を受けて 1 が設定されるほか，信頼系からは自由にアクセスが可能である．
  - **アクセス違反レジスタ（ILLEGAL_ACCESS レジスタ）**
    他の信頼系デバイスからのアクセス違反通知を保存するレジスタである．アクセス違反が発生するとそれぞれのデバイスに対応したビットに 1 が設定される．いずれかのビットに 1 が設定された場合に，信頼系割込みを発生させる．
- **プロテクションブリッジ（DZ-PB）**
  信頼系デバイスへのアクセスを監視し制御する役割を持つモジュールである．信頼系デバイスへアクセスするためにはプロテクションブリッジを経由する必要がある．信頼系実行状態であればブリッジの先へアクセ

Vol.2017-ARC-225 No.12
Vol.2017-SLDM-179 No.12
Vol.2017-EMB-44 No.12
2017/3/9

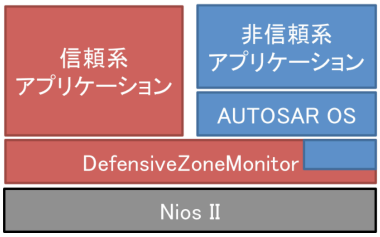

図 3 DefensiveZone のソフトウェア構成

図 4 非信頼系から信頼系への切り替え

スを許可する．非信頼系実行状態であればアクセスを禁止し遮断すると同時に，アクセス違反割込みを発生させる．

2.3 ソフトウェア構成

DefensiveZone のソフトウェアは，図 3 のように非信頼系の RTOS とアプリケーション，信頼系のアプリケーションに加えて DefensiveZoneMonitor（DZ-Monitor）と名付けたモニタプログラムから構成される．

DZ-Monitor はパーティションの切り替えを担うソフトウェアである．また，パーティションの切り替え時にはそれぞれの系のコンテキストの退避及び復帰を行う．

非信頼系から信頼系へ切り替わるトリガーとなるのが信頼系割込みであり，信頼系割込みによって DZ-Monitor 内に配置されたベクタが実行される．ベクタからハンドラ入り口へとジャンプし，コンテキストの保存や復帰を行ってから信頼系の割込みハンドラが実行される．このようなパーティション切り替えの処理フローを図 4 に示す．

2.4 信頼系へのパーティション切り替えタイミングに関する問題

前述したように，DefensiveZone ではステートコントローラがシステムの実行状態管理を担っている．信頼系へと切り替わるトリガーは信頼系割込みであり，その発生通知は信頼系割込みコントローラからプロセッサとステートコントローラの両方へ送信される．この通知の送信経路は図 5 のように異なっており，到達タイミングに数サイクルの差

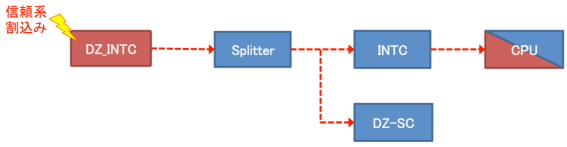

図 5 割込み通知送信経路

が生じる．具体的には，プロセッサへの伝達には通常の割込みコントローラを経由する必要があるため，入力された非信頼系割込みの数に応じた遅延が生じ，ステートコントローラへの伝達に比べ 2〜5 サイクルの遅れが生じる．

この差は小さなものではあるが，システムの信頼性に致命的な欠陥を引き起こす原因ともなりうる．短時間ではあるものの，各モジュールは信頼系実行状態に切り替わっているにも関わらずプロセッサが非信頼系のコードを実行できてしまうタイミングが存在してしまうためである．

2.5 パーティション間通信に関する課題

DefensiveZone には，信頼系と非信頼系で通信を行う手段が信頼系割込みのみしか用意されておらず，非信頼系が特定の処理を信頼系に依頼するにも割込みを用いるしかない．しかし，信頼系割込みによるパーティションの切り替えはオーバヘッドが大きいため高速な通信が必要とされる場面には不向きである．

この課題を解決するためには，より軽量なパーティション間通信機構が必要となる．具体的な方法としては，ATK2-SC3 にも搭載されている信頼関数呼出し機構が考えられる．また，信頼系に依頼していた処理を非信頼系自身で実行できればさらなる高速化が可能である．ただし，処理内容によっては実行中に割込まれた結果取り扱っていたデータの整合性が損なわれてしまう恐れもあるため，通常は禁止できない信頼系の割込みも含めて禁止した上で実行する必要がある．

3. パーティション切り替え同期機構

前節で述べたように，DefensiveZone には信頼系パーティションへの切り替え時に保護が機能しなくなる瞬間が存在する．本節では，この課題への対策であるゲートウェイ機構について解説する．

ゲートウェイ機構では，非信頼系からは読み出しのみ可能なメモリ領域（ゲートウェイメモリ）を用意し，ベクタを配置する．そこで特定の処理をシーケンシャルに実行することで一時的に信頼できる状態を作り出した上で，ベクタからシステム実行状態レジスタに 1 をセットし，信頼系実行状態に切り替わるのを待ってから信頼系のコードに移動する．ゲートウェイ機構では，一連の処理を正しく実行したかどうか判別するために，鍵を用いた認証を取り入れた．ゲートウェイ機構を利用したシステム実行状態切り替えの流れを解説する．

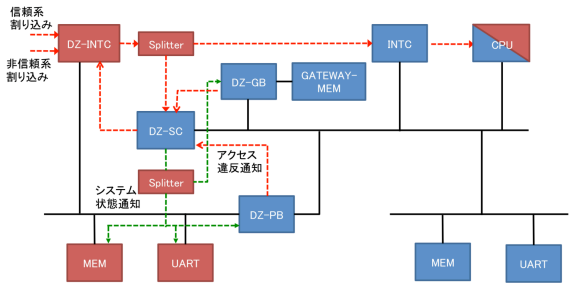

図 6 DefensiveZone のハードウェアアーキテクチャ（ゲートウェイ機構対応）

(1) 信頼系割込みでゲートウェイメモリ内の割込みベクタがフェッチされたとき，ゲートウェイブリッジが鍵を生成する．
(2) 割込みベクタを順番にフェッチしていく．
(3) 即値ロード命令のフェッチ時に，ゲートウェイブリッジからステートコントローラおよびプロセッサへ鍵が送信される．
(4) 鍵を受け取ったプロセッサが，その鍵をステートコントローラのシステム状態レジスタへ書き込み，認証に成功するとシステム状態が信頼系へ切り替わる．
(5) システム状態レジスタの値を読み出し，確実に信頼系に切り替わるのを待つ．
(6) 信頼系に切り替わったことを確認したら，DZ-Monitor 内のハンドラへジャンプする．

ゲートウェイ機構を実現するために，DefensiveZone に以下のような変更を加えた．

3.1 ハードウェア変更点

図 6 のようにハードウェアアーキテクチャを変更した．以下に詳細な変更点を解説する．

- **ゲートウェイブリッジおよびゲートウェイメモリ追加**
  ゲートウェイメモリは切り替え処理を配置する専用のメモリ領域である．そして，ゲートウェイメモリへのアクセスを制御するのがゲートウェイブリッジである．ゲートウェイメモリの実行シーケンスをチェックするために，ゲートウェイブリッジは内部に状態マシンを持つ．また，プロセッサへを鍵を送信するために，フェッチ内容が即値ロード命令である場合に命令の即値部分を鍵で上書きする機構を導入した．
- **ステートコントローラ拡張**
  信頼系モジュールから非信頼系のモジュールへと配置を変更した上で，非信頼系からは自由にシステム実行状態を変更できないように拡張した．切り替え処理を正しく実行した場合にのみ書き込みを受け付けるために，状態マシンを作成し，正規の手順で鍵が発行された場合以外は認証を行えないようにした．

3.2 ソフトウェア変更点

信頼系割込みベクタを DZ-Monitor からゲートウェイメモリへと移動した．割込みベクタをシーケンシャルに実行していき，即値ロード命令を実行するとゲートウェイブリッジにより命令が書き換えられる．書き換えられた命令を実行するとレジスタに鍵が格納され，その鍵を使ってシステム実行状態を変更する．

4. DefensiveZone の同期通信機能

4.1 信頼関数呼出し機構

本研究ではゲートウェイ機構のアーキテクチャを利用し，次のような方法で信頼関数呼出しを実現した．

(1) 非信頼系から関数呼出しでゲートウェイメモリの特定アドレス（関数エントリ）を実行
(2) 信頼系割込みの場合と同様に，シーケンシャルに命令を実行し鍵を発行する
(3) システム実行状態を信頼系へ変更する
(4) 信頼関数を呼出し，実行する
(5) 信頼関数の実行が終了したら再びゲートウェイメモリに戻る
(6) システム実行状態を非信頼系へ変更する
(7) 非信頼系処理へリターンし，処理を継続する

この機構はゲートウェイ機構を利用したものであり，以下のような拡張を加えることで実現した．

4.1.1 ソフトウェア変更点

信頼関数呼出しでは，スタックポインタの切り替えやリターンアドレスの退避および復帰を自力で行わなければならない．それに加え，ゲートウェイメモリを実行中に割込みによって処理が中断されてしまう恐れがあるため，処理の先頭と終端に割込みの禁止及び解除命令を追加した．

4.1.2 ハードウェア変更点

ゲートウェイブリッジの状態マシンを拡張し，関数エントリをフェッチしてから割込みを禁止するまでの状態を追加した．

4.2 信頼系割込み一時禁止機構

非信頼系が信頼系に依頼していた処理を非信頼系自身が実行する際，途中で割込みが発生しないようにしなければならない．しかし，NMI である信頼系割込みは非信頼系割込みと同様の方法では禁止することができない．そこで本研究では割込みコントローラを拡張することで信頼系割込みの禁止設定を可能にした．ただし，信頼割込みが永続的に禁止されると保護が正常にはたらかなくなるため，割込み禁止設定は一定時間で自動的に解除されるようにし，連続して禁止の設定ができないように制限を設けた．DefensiveZone の割込みコントローラは信頼系のものと非信頼系のものが接続されているが，非信頼系の割込みコントローラを拡張することで，信頼系割り込みの禁止設定に

Vol.2017-ARC-225 No.12
Vol.2017-SLDM-179 No.12
Vol.2017-EMB-44 No.12
2017/3/9

表 1 評価環境

FPGA 開発ツール	Altera Quartus II 16.1
コンパイラ	nios2-elf-gcc:gcc version 5.3.0
評価ボード	Altera DE2-115
プロセッサ	Nios II/f
プロセッサ動作周波数	50[MHz]
DefensiveZone に搭載する非信頼系 RTOS	TOPPERS/ATK2-SC1 1.4.0
保護 RTOS	TOPPERS/ATK2-SC3 1.4.0
ベンチマークプログラム	TOPPERS/A-OSBENCH 1.1[5]

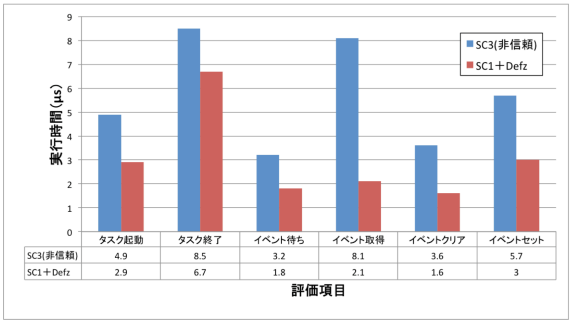

図 7 システムサービス実行時間の評価結果

対応した．これは，信頼系割り込みは非信頼系の割込みコントローラに割込み要因の一つとして入力されており，この信号を操作することで信頼系割込みを禁止することが可能なためである．割込みコントローラに T_INT_DISABLE というレジスタを作成し，レジスタの最下位ビットが 1 のときには非信頼系割込みコントローラに入力される信頼系割込みの信号を 0 で上書きすることで，どのような信頼系割込みが発生してもプロセッサに届く前に無効化することができる．レジスタ操作は，割込みコントローラ内に作成した状態マシンによって制御されている．

5. 性能評価

本節では DefensiveZone の性能およびハードウェア面積の評価を実施し，その結果から得られた考察を述べる．

5.1 評価内容

5.1.1 保護 RTOS との比較

DefensiveZone は保護 RTOS よりもローエンドなシステムに適したパーティショニングシステムを実現するという目的のもとで設計されているため，以下の評価項目について，保護 RTOS で同様の機能を実行した場合との比較を行った．

- 非信頼系からのシステムサービス実行時間
- 信頼関数応答時間

また，DefensiveZone の搭載前後におけるハードウェア面積の変化についても比較した．

5.1.2 パーティション間通信機能の比較

本研究では DefensiveZone のパーティション間通信機構を新たに 2 種類実装した．これらの機構と信頼系割込みによる通信について，リングバッファを用いた通信を題材に性能比較を実施した．

評価対象の操作はバッファへの送信とし，10 個の unsigned int 型データをバッファに送信する処理を完了するまでに要する時間を比較した．

5.2 評価環境

本研究における性能評価は，表 1 の環境で実施した．

5.3 評価結果および考察

各評価の結果と，そこから導き出される考察を述べる．本研究では 10,000 回連続して評価対象の処理を実行した中の最頻値で比較を行った．

5.3.1 保護 RTOS との比較

非信頼系からのシステムサービス実行時間

DefensiveZone および ATK2-SC3 で A-OSBENCH を実行した結果は図 7 のとおりである．

グラフから読み取れるように，ATK2-SC3 と比較すると DefensiveZone の方がおおよそ 2μ 秒オーバヘッドが小さく抑えられた．このような結果になった理由は，両者のシステムサービス呼出し方法の違いにある．ATK2-SC3 の場合は，メモリ保護によって非信頼系から直接システムサービスを実行することができないため，代わりにソフトウェア割込みを経由して呼出している．一方で DefensiveZone に搭載している ATK2-SC1 はメモリ保護機能を持っておらず，OS そのものが非信頼系に所属するソフトウェアであるため，通常の関数呼出しでシステムサービスを実行することができる．ソフトウェア割込みは重く時間のかかる処理であるため，ATK2-SC3 の方がオーバヘッドが大きくなったのである．また，DefensiveZone の評価結果は DefensiveZone 搭載前の結果とほぼ同等の結果となっていることも確認できた．

信頼関数応答時間

DefensiveZone および ATK2-SC3 の信頼系関数応答時間を計測した結果は図 8 のとおりである．DefensiveZone の最頻値は 0.9μ 秒（10,000 回中 5063 回），ATK2-SC3 の最頻値は 2.8μ 秒（10000 回中 7356 回）となった．

グラフから読み取れるように，DefensiveZone の信頼関数呼出しは ATK2-SC3 の同機能と比較して高速であると言える．ATK2-SC3 が OS から信頼関数を呼出しているのに対し，DefensiveZone が信頼関数呼出しのために実行しているのはゲートウェイメモリ内の信頼関数呼出し処理のみである．DefensiveZone で信頼関数を呼出すにはアセンブリコードを 20 行程度実行するだけでよいため，DefensiveZone の信頼関数呼出しは高速に実行することができる．

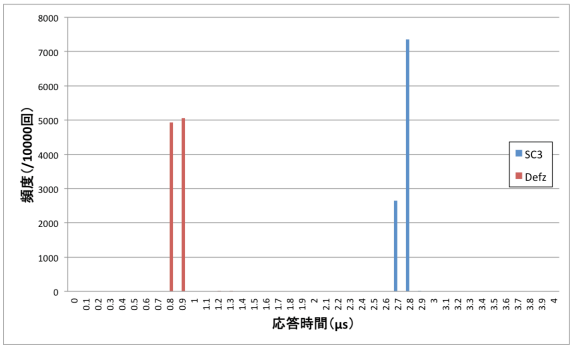

図 8 信頼関数応答時間の評価結果

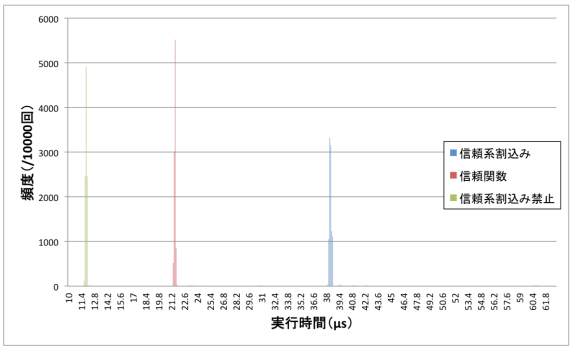

図 9 パーティション間通信の実行時間比較

5.3.2 リングバッファによるパーティション間通信

それぞれの通信方法について評価を実施し，その結果をまとめたものが図 9 のグラフである．

それぞれの通信方法における実行時間の最頻値は，

- 信頼系割込み：38.4μ 秒（10,000 回中 3,315 回）
- 信頼関数：21.6μ 秒（10,000 回中 5,517 回）
- 信頼系割込み一時禁止：11.9μ 秒（10,000 回中 4,920 回）

となった．この結果から，本研究で実装したパーティション間通信の手法はいずれも信頼系割込みよりも高速な実行が可能であることを確認できた．特に，依頼していた処理が非信頼系でも実行可能な処理であれば，信頼系割込み一時禁止機構を用いて非信頼系自身が実行することで実行時間を 3 分の 1 以下に抑える事が可能であり，有効な手段であることを確認できた．

5.3.3 ハードウェア面積評価

DefensiveZone の実現には専用の保護ハードウェアモジュールが必要であり，DefensiveZone を利用しない場合と比較してハードウェアの面積が増加する．ここでは，このハードウェア面積の増加がどれほどなのかを解説する．

比較対象は ATK2-SC3 の性能評価に用いたハードウェアデザインとし，評価項目は LE 数，レジスタ数，メモリ容量の 3 項目とした．比較した結果を表 2 に示す．

LE 数に関しては，ボード上に配置可能な最大数を基準にして 2%しか増加しておらず，また実際に配置されている LE 数も約 12%の増加に抑えることができている．レジスタ数も約 16%の増加に抑えられている．唯一メモリ容量は DefensiveZone 搭載によって大幅に増加したが，原因としてはゲートウェイメモリの追加が考えられる．ゲートウェイメモリは現在研究のために余裕を持って大きな領域を確保しており，実際に使用しているメモリ領域は表 2 よりも大幅に小さい．そのため，今後の研究を経て最適化することで大幅なカットが可能であると考えられる．また，DefensiveZone ではレジスタバンクを利用しているために使用するメモリが増加していることも要因のひとつであると考えられる．

表 2 ハードウェア面積比較

	DefensiveZone 非搭載	DefensiveZone 搭載	増加率
LE 数	25,813 /11,4480 (23%)	29,026 /11,4480 (25%)	約 12%
レジスタ数	16122	18699	約 16%
メモリ容量	1,125,376 /3,981,312 (28%)	2,704312 /3,981,312 (68%)	約 140%

6. おわりに

本研究では，低コストなパーティショニング機構である DefensiveZone の見直し及び拡張を実施した．拡張後の DefensiveZone の性能評価を行った結果，DefensiveZone は保護 RTOS よりもローエンドな車載システムに適したパーティショニング機構であると結論付けることができた．

今後の展望としては，デバイスのパーティショニング [6] と組合わせてより効果的な保護を実現することが挙げられる．

謝辞

本研究の一部は，（株）半導体理工学研究センターとの共同研究による．

参考文献

[1] Dominik Reinhart, Gary Morgan ”An Embedded Hypervisor for Safety-Relevant Automotive E/E-Systems” Industrial Embedded Systems (SIES), 2014 9th IEEE International Symposium

[2] 土本幸司, 川島裕崇, 本田晋也, 高田広章 ”車載制御システム向けパーティショニング機構” 車載システム，組込み技術とネットワークに関するワークショップ ETNET2013

[3] Nios II プロセッサ (online) available from ⟨https://www.altera.co.jp/products/processors/overview.html⟩ (accessed 2017-1-24).

[4] TOPPERS プロジェクト/ATK2 (online) available from ⟨https://www.toppers.jp/atk2.html⟩ (accessed 2017-1-24).

[5] TOPPERS/A-OSBENCH とは (online) available from ⟨https://www.toppers.jp/a-osbench.html⟩ (accessed 2017-1-24).

[6] 加藤祐輔，本田晋也，高田広章 ”ミックスドクリティカルシステム向けペリフェラル共有機構” Symposium on Cryptography and Information Security (SCIS) 2016.