組み込みシステムのための アスペクト指向に基づくコンフィギュレーション

松岡 幸典† 横山 孝典† 志田 晃一郎† 兪 明連†
武蔵工業大学大学院 工学研究科†

1 はじめに

生活家電や工業製品などをコンピュータ制御する組み込みシステムの重要性が高まっている．

近年では，組み込みソフトウェアの規模が増大し，開発にはオペレーティングシステム(OS)が必須のものとなった．組み込みOSは，様々なハードウェアに搭載されることを想定し，設定に応じて動作や構成を変更できることが望ましい．そのために，設定変更が可能なOSの効率的な開発法が求められている．

一方，プログラムの複数の側面を分離して開発する手法として，アスペクト指向プログラミング[1]が注目を集めている．これまでに，アスペクト指向を設定変更や機能拡張の容易な組み込みOSの開発に適用する研究が行われている．Afonsoらは，ソースコードの可読性や拡張性の向上を狙いに，C++で記述された組み込みOSを対象に，排他制御機構，システムログ機能をアスペクトとしてモジュール化した[2]．CiAOプロジェクトでは，メモリ保護や割り込みの同期機構に関する記述をアスペクトに定義した，新たな組み込み向けOSファミリを試作している[3]．

これらの従来研究では，ソフトウェアアーキテクチャの構成や，制御フローの変更にアスペクトを用いている．残されている組み込みOS開発の課題のひとつとして，複数のハードウェアアーキテクチャへの対応がある．この問題を対象にアスペクト指向プログラミングを適用できる可能性がある．

そこで本研究では，ハードウェアに依存するコードをアスペクトとして記述するOS開発手法を提案し，その利点を分析する．

2 アスペクトによるハードウェア依存部の分離

アスペクト指向では，複数のモジュールに関わる処理（横断的関心事）を，アスペクトと呼ぶ独立したモジュールとして定義する．その概要を図1に示す．アスペクトは，横断的関心事である処理本体をあらわすアドバイスと，処理の挿入位置（ジョインポイント）を指定するポイントカットで構成される．アスペクト指向言語処理系は，ポイントカットの指定に応じて，各モジュールにアドバイスの処理を織り込む．

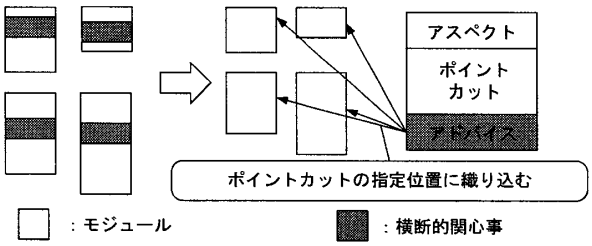

図1．アスペクト指向プログラミングの概要

本研究ではOSのハードウェア依存部を共通部から分離してアスペクトとして記述する．そして，対象ハードウェアに応じてそのハードウェア向けのアスペクトを選択し，織り込むことで，ハードウェアに特化したOSを構成（コンフィギュレーション）する．この方法は，従来よく用いられてきたコンパイルスイッチによってマクロを切り替えて対象ハードウェアを選択する手法に比べて，ソースコードの可読性の向上が期待できる．また，専用のコンフィギュレータを用いる場合と異なり，標準的な言語処理系の機能のみで実現できるため，ハードウェア対応のための専用言語を必要としない．

3 組み込みOSのハードウェア依存部抽出

TOPPERSプロジェクト[4]で公開されているオープンソースのOSである，OSEKカーネル，JSPカーネル，ASPカーネルのコードから，どのようなコードがハードウェアに依存しているのかを分析した．ハードウェア依存部には，大きく分けてマイクロプロセッサ依存部とシステム依存部の2つがある．以下それぞれについて述べる．

(1) マイクロプロセッサ(MPU)依存部

組み込みOSは，複数の異なるアーキテクチャを持つMPUに搭載されるのが普通である．MPUによって，周辺機器との通信方法や，割り込みを段

Aspect-Oriented Configuration for Embedded Systems
†MATSUOKA,Yukinori, YOKOYAMA,Takanori,
SHIDA,Koichiro and YOO,Myungryun
Research Division in Engineering,
Graduate School of Musashi Institute of Technology

階的にマスクできるかどうかなどが異なる．したがって，通信用ポートの初期化処理，割り込みをマスクするための処理などがMPUに依存する．

また，言語処理系によって，同じ処理内容でもインラインアセンブラの構文が違い，異なる記述を要することがある．

(2) システム依存部

対象システムによって，組み込みコンピュータ上に搭載される周辺機器が異なる．したがって，それぞれの周辺機器に特化した初期化処理，終了処理が必要になる．又，搭載機器によって割り込みベクターテーブルの内容も変化する．

デバイスドライバをOSに含む場合は，それもシステム依存部に含まれる．例として通信用のデバイスを考えると，通信用のポートへの書き込みと読み出し処理，割り込み処理などを定義する必要がある．

4 ハードウェア依存部のアスペクト記述法

アスペクトとして依存部を定義することで，OSの共通部から依存処理の呼び出しを除去できる．これにより，ポイントカットの書き換えや，まとめて記述されていた依存処理の分割を行うことが可能になり，任意のタイミングで依存処理を実行できる．従来は関数呼び出しを一か所にまとめるためにハードウェア依存部としてひとくくりにしていた，割り込み機能のための処理やタスク管理機能のための処理などを，アスペクト指向を用いてそれぞれ別のモジュールとして記述できる．

C言語向けのアスペクト指向言語処理系の多くは，アスペクトを織り込んだ結果としてC言語のソースコードを出力する．アドバイスとして記述された処理は，ポイントカット指定した位置からの関数呼び出しに置き換えられる．図2に示すように，ハードウェア依存部とOS共通部を，動作するひとつのモジュールとして統合するために織り込みを用いれば，依存部のコードをマクロ定義する方法と比べOS共通部の処理フローが明確になり可読性が大きく向上する．また，ハードウェア依存関係解決のためのツールや言語を新たに用意する必要もない．

5 記述実験

アスペクトによるハードウェア対応コードのコンフィギュレーションが可能であることを確認するため，TOPPERS/OSEKカーネルのCPU依存部およびシステム依存部の初期化・終了処理をアスペクト記述して再実装した．対象ハードウェアとしてH8S/2638Fが搭載された評価ボードと，M16C/26

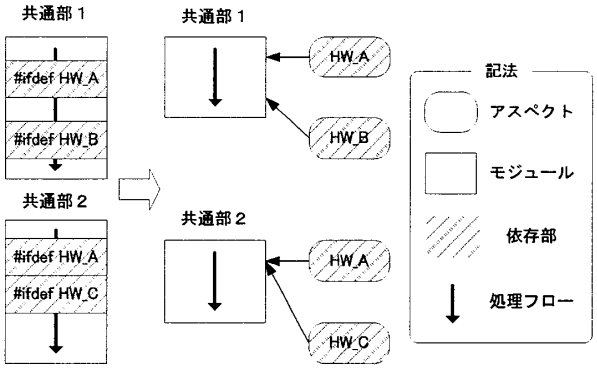

図2．マクロ記述との比較

が搭載された評価ボードを用いた．アスペクト指向言語処理系としてAspeCt C Compiler 0.7[5]を用いてそれら2つのハードウェア向けのコンフィギュレーションを行い，それらのOSを使用したプログラムを実行させ，本手法の有効性を評価した．

実験の結果，記述・実験を行った多くのハードウェア依存部については，アスペクトによる分離が可能であることを確認した．ただし，現在使用している処理系ではインライン関数としてアスペクトを織り込むことができず，関数呼び出しが発生すると問題が起こるような低レベルのコードへの対応は難しいこともわかった．これについては，今後インライン化に対応した処理系を使用することで解決できる可能性が高い．

6 おわりに

OSに含まれるハードウェアに依存する処理の一部をアスペクト記述するとともに，織り込むことで対象ハードウェア向けのOSを構成する手法を提案した．

今後は，デバイス制御などのより多くの対象についてアスペクト記述し，多様な機器向けの実装検証を進める予定である．

参考文献

[1] G. Kiczale et al., Aspect-Oriented Programming, *Proc. of European Conference on Object-Oriented Programming (ECOOP), 1997.*

[2] F. Alfonso, et al., Applying Aspects to a Real-Time Embedded Operating System, *Proc. of 6th workshop on Aspects, components, and patterns for infrastructure software*, 2007.

[3] D. Lohman, et al., PURE Embedded Operating Systems — CiAO, *IEEE Intel Workshop on Operating System Platforms for Embedded Real-Time Applications*, 2006.

[4] TOPPERS Project, http://www.toppers.jp/

[5] The AspeCt-oriented C compiler, http://www.aspectc.net