モジュール間依存性のビジュアル表示を活用した組込みシステムのソフトウェア・ハードウェア再利用開発方式

永松 博子†　神戸 英利††　三井 浩康†　小泉 寿男†
東京電機大学 理工学部 情報システム工学科†　三菱電機株式会社††

1．はじめに

現在、組込みシステムは LSI などのハードウェア(以下 H/W)、ソフトウェア(以下 S/W)両者について開発規模が拡大している反面、開発期間の短期化が求められており、開発資産を再利用して製品開発をしなければ難しい状況にある。また、大規模な組込みシステムは開発者であっても構成の把握が困難である。また、修正モジュールのモジュール間関係を把握していないと、他のモジュールに影響を及ぼしかねないが、モジュール間関係を全て把握するのは非常に時間がかかる。これらの作業で時間をかけると、再利用の効果を低下させてしまうが、バージョン管理ソフトなどではシステムの構成把握やモジュール間関係を把握することができないという問題がある。

本研究では組込みシステムにおける H/W、S/W 両者の再利用を支援する構成管理ツールを提案・開発し、ツールを利用した再利用開発方式を提案する。

2．ビジュアル表示による S/W・H/W 再利用方式

2.1 モジュール関連表示

本研究では再利用開発を効率的に行うために構成管理ツールを提案する。構成管理ツールは関連解析部と関連表示部の２つのアプリケーションから成る。

関連解析部はソースファイルとプロジェクトファイルを読み込み、システムの構成やモジュール間の関係を解析し、関連解析データ(xml 形式)を生成する。プロジェクトファイルは、生成する実行ファイル名とそのビルド時に使用するソースファイル名が定義されたファイルである。

関連表示部は、関連解析データを読み込み、ソフトウェアブロック図を表示し、モジュール間の関数呼び出し関係を矢印で表示することができる。図 1 にシステムの階層構造とブロック図表示の関係を示す。

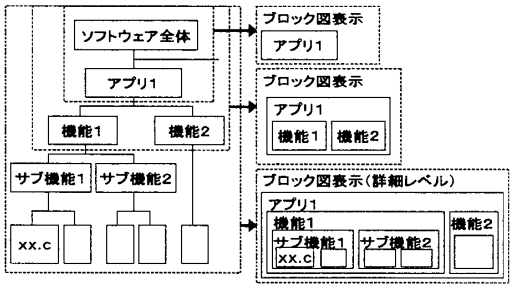

図 1 システム階層構造とブロック図

Software hardware recycling development method of embedded system that uses visual display of the dependency between modules
†Tokyo Denki University　††Mitsubishi Electric Corporation

図 2 に、H/W・S/W モジュールの記憶・保存方法を示す。表示されるブロックは、SystemC で S/W か H/W で決定済みのもの、SystemC で未決定のもの、C++などの S/W のソースファイルの 3 種類が存在する。S/W ブロックの最小単位であるソースファイルのブロックからソースファイルを開くことができ、ソースファイルの記憶場所へリンクしており、開発効率を向上させる。

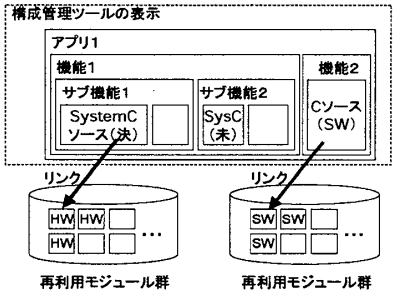

図 2 モジュールの記憶方法

2.2 ハードウェア・ソフトウェア協調設計

H/W・S/W 協調設計はシステムの機能の、どの部分を S/W で実現し、どの部分を H/W で実現するかを最適化するというアプローチである。本方式で用いる SystemC は S/W と H/W を区別せずに同一言語で記述し、後に H/W と S/W の区別を具体化できるという特徴がある。

図 3 に再利用開発の流れと協調設計の関係を示す。まず、再利用資産に含まれるモジュールを H/W、S/W、どちらか未決定のものに分ける。製作段階で修正ファイルが未決定のモジュールだった場合に H/W と S/W に振り分けるために協調設計を行う。

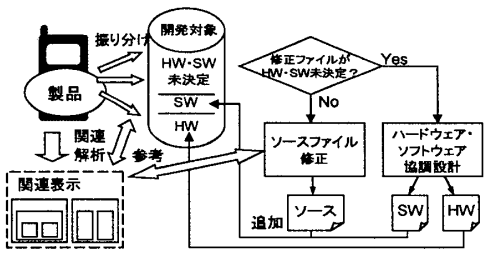

図 3 再利用開発の流れと協調設計の関係

2.3 ハードウェアを含んだ再利用開発方式

図 4 に再利用開発方式の流れを示す。

(1)製品仕様の決定：実現する機能や性能を決定する。

(2)関連解析：再利用する製品を構成管理ツールで解析し、ブロック図を作成する。さらに、関連表示と(1)で決定した製品仕様をもとに再利用部品を、H/W、S/W、H/W・S/W のどちらで実現するか未定の 3 種類に振り分ける。

振り分けた結果は、関連表示画面でブロック図に印をつけていく。

(3)調査：関連表示を参考に、新規追加・流用・修正して再利用するモジュールをそれぞれ決定する。

(4)設計製作：新規作成・修正するモジュールの製作をおこなう。ここで、修正するモジュールが H/W、S/W、H/W・S/W のどちらで実現するか未決定の 3 種類で手順が異なる。

①S/W の場合：修正すべきソースコードを、関連表示の呼び出し関係を参照しながら修正を行う。

②H/W の場合：VHDL のソースコードや、CAD を用いて修正を行う。

③未決定の場合：修正する機能モジュールを SystemC の設計手順の通りに、抽象度を下げていき、コスト・性能などを考慮してトレードオフを行い、H/W と S/W に振り分け、それぞれの記述をする。

(5)ビルド：(1)～(5)で製作されたプロダクトをビルドする。

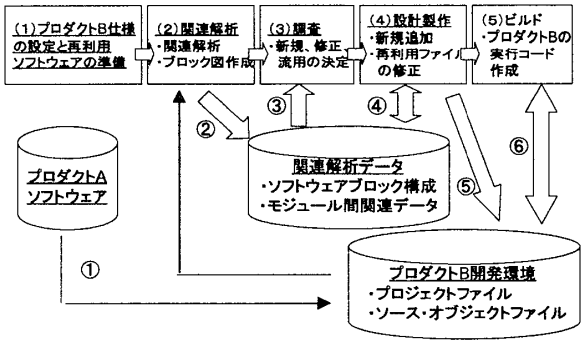

図 4 再利用開発の流れ

3．評価

本方式の有効性を確認するため、ソースファイル数 約 800 ファイル、H/W・S/W・いずれか未決定のモジュールがそれぞれ存在するようなシステムを用意した。このシステムを 3.3 節で提案した流れに沿って、モジュールの修正をおこない、本方式を評価する。

①再利用対象のシステムを構成管理ツールで解析する。

②表示結果を参考に、システムの構成を把握する。システム内のモジュールを H/W、S/W、未決定の 3 種類に振り分けた。図 5 に解析結果を示す。左の未決定のモジュールに印をつけた。このことによって未決定のモジュールがひと目でわかるようになった。

③S/W モジュールの修正を行う。修正する S/W は図 5 の右のブロックで、修正するファイルは点線で囲んだファイルである。このファイルに影響関係にあるファイルは色つきのブロックで表現され、さらに呼び出し先、呼び出し元の関数の表示ができるため、ソースの修正はツールなしに比べ、容易に行うことができた。

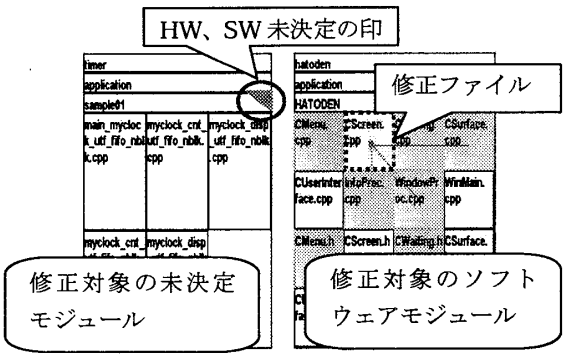

図 5 再利用システムの解析結果

④H/W と S/W で未決定のモジュールの修正を行う。図 5 の左のモジュールが対象になる。現時点では抽象度の高い UTF モデルで記述されている。このファイルを修正した。そして、時間的概念を加えて TF モデルに変換する。TF モデルでのシミュレーションの結果を考慮して、機能モジュールを H/W と S/W に振り分けた。H/W 部分は BCA モデルに変換し、S/W 部分はインタフェースを合わせた。BCA モデルは動作合成ツールで抽象度の低い RTL モデル、RTL モデルは論理合成ツールで VHDL に自動変換できるので今回は作業を省いた。この手順について、H/W と S/W のモジュールの一元管理、抽象度の高い階層での修正にブロック図を役立てることはできたものの、階層が下がると構成管理ツールなしで開発を進めなければならないという問題点がわかった。

4．まとめと今後の課題

ハードウェアとソフトウェアを一括管理し、開発の補助をする構成管理ツールを提案・製作した。また、ツールを用いてハードウェアとソフトウェアを再利用して、組込みシステムを開発する流れを示し、評価を行なった。評価を行なった結果、ソフトウェアや SystemC でもソフトウェアに近い抽象度の高い階層での修正には非常に有効であるという有効点が見つかった。しかし、抽象度の低い階層では構成管理ツールを役立てた開発ができなかったという問題点が発見された。

参考文献

[1] 遠藤祐，小泉寿男：再利用モジュールのオンライン評価を取り入れたハードウェア・ソフトウェア協調設計方式とその検証，電気学会論文誌 C, Vol.124, No.11 , pp2249－2259(2004)

[2] 神戸英利，永松博子，三井浩康，小泉寿男：組込みソフトウェアの再利用を支援するモジュール間相互関連表示法，情報処理学会組込みシステム・ソフトウェア工学 合同研究発表会 (2007)

[3] 高田広章：「組込みシステム開発技術の現状と展望」，情報処理学会論文誌，Vol.42，No.4 pp930-938 (2001)