これからの組込みシステム開発

株式会社東芝 研究開発センター システム技術ラボラトリー　研究主務

荒木　大

1．はじめに

携帯端末、デジタル家電などの組込みシステム製品は、設計規模の増大、設計の複雑さ、市場からの開発期間短縮の要求によって、従来の形の設計手法では限界が見えてきました。例えば、デバイス技術の進歩によって、ワンチップに1000万を超えるトランジスタの集積が可能となり、メモリ、プロセッサ、ハードウェアロジックをワンチップに複数搭載したシステムLSIも可能になってきましたが、このような大規模な設計を行うための手法と設計ツールが十分には整備されていません。

従来は、組込みシステム製品を構成するLSIやソフトウェアは別々の設計部門で独立に行なわれており、概ねハードウェア設計がソフトウェア設計よりも先行して行なわれています。ハードウェア設計はその費用と期間の観点から、製品の機能仕様が十分には確定していない段階で設計をスタートさせます。また、ソフトウェア設計者にとっては、ハードウェアが出来上がってこないと実質的な開発を開始し難いという事情もあります。さらに、ハードウェアは、ソフトウェアのように設計のやり直しが行い難いので、後段のソフトウェア設計の側で不都合を吸収するしか方法がありません。多機能化／高性能化が進み、システム全体での最適化が重要になっていますが、現状の設計フローはシステム全体での最適化の実現を困難にしています。

こういった観点から、コ・デザインやシステムレベル設計が重要視されています。組込みシステム製品を構成するLSIやソフトウェアを、共通言語を使って設計を行い、シミュレーションや性能解析が行える環境を整備することで、ハードウェア設計とソフトウェア設計を並行して行う手法です。この手法によって、設計工数の削減が図れると共に、詳細設計に入る前の段階でシステム全体の機能とアーキテクチャの最適化を行うことが可能になります。また、設計資産（IP）の共通化と再利用の促進も行えます。

コ・デザインやシステムレベル設計のためには、LSI 設計とソフトウェア設計の両方に利用できる設計言語と機能仕様から実装設計に変換・合成するための設計手法が必要です。本稿では、システムレベル設計言語として注目を浴びているSpecC言語を活用した設計手法と、SpecC言語を利用するための設計ツール VisualSpecを紹介します。

2．SpecC 言語

2.1．SpecC 言語とは

SpecC 言語[1, 2]は 1997 年に UCI（カリフォルニア大学アーバイン校）の D.Gajski 教授等が開発したシステムレベル設計言語で、ハードウェア／ソフトウェア混在システムの設計仕様を記述することができます。

●SpecC 言語は、ANSI-C 言語をベースにて、並列性、割込み処理、状態遷移、通信といった組込みシステムの動作仕様を記述するための拡張構文を用意して文法的な拡張を行っています。これにより、コンパクトで可読性の高い記述が可能です。

●SpecC 言語で記述される設計仕様は、システムレベルの仕様記述として位置づけられるために、記法上はハードウェアとソフトウェアの文法の差異がありません。すなわち、実装の詳細をニュートラルな状態のままで、各々の機能をハードウェアあるいはソフトウェアとして実現した場合のトレードオフを容易に分析・検討することができます。

●システムレベル設計において必要なさまざまな抽象レベルのモデル、たとえば、システムの機能仕様記述、システムのアーキテクチャ記述、さらにはプロセッサやバスなどのシステム構築部品までを単一の言語で統一的に記述できます。また、各設計段階で常にシミュレーションが可能です。

●SpecC 言語を利用することによって、仕様から設計へとシステム開発を進める際に、連続的に開発の基幹データを共有することができます。また、ビヘイビア（behavior）とチャネル（channel)の二つのクラス構造を使って仕様記述を行う点でオブジェクト指向設計の手法を取り入れており、IP あるいは設計データベースの構築にも向いています。これにより設計再利用の実現が容易になります。

●モデルの機能と接続（通信）が完全に分離でき、コンポーネントを階層的に分解して定義することができます。動作の階層化ではシステムを逐次実行と並列実行の部分動作ごとに分解して定義できます。構造の階層化によって、コンポーネント間の接続構造を表現できます。さらに仕様記述のモジュール化によって設計の再利用も容易に行えるよう考慮されています。

2.2．SpecC 言語の文法

SpecC 言語の持つ計算モデルは Program-State Machine（PSM）[1]と呼ばれます。PSM は、階層型並列ステートマシンと手続き型プログラミング言語を組み合わせたモデルです。

図 5.3-1 に SpecC プログラムの構造を簡単に示します。システムの全体は、階層的に積み上げた「ビヘイビア」と呼ぶブロック（図 5.3-1 の X,Y,Z）によって表現します。それぞれのビヘイビア間で行う通信は、各ビヘイビアに定義するポ

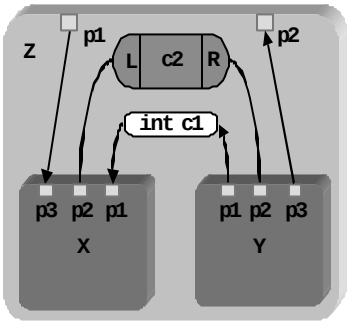

図2. SpecC プログラムの基本構造

ートとポート間の接続関係によって確立する。通信プロトコルは「チャネル」と呼ぶブロック（図 5.3-1 の c1, c2）の中で手続きを定義することによって表現します。

図 5.3-1 のモデルを SpecC 言語で記述したコードが図 5.3-2 です。SpecC プログラムは、ビヘイビア（behavior）、チャネル（channel）、インタフェース（interface）の三つの要素からなります。ビヘ

```
interface L {
        void write(int x);
};
interface R {
        int read(void);
};
channel C implements L, R        // チャネルCはLとRのインタフェースを持つ
{
  void write(int x) {
          ...
        }
int read(void) {
          ...
        }
};
behavior X(in int p1, L p2, in int p3)      // ビヘイビアXの定義
{
  void main(void) {
          p2.write(p1);
        }
};
behavior Y(out int p1, R p2, out int p3)  // ビヘイビアYの定義
{
  void main (void) {
          p3 = p2.read();
        }
};
behavior Z(in int p1, out int p2)         // ビヘイビアZの定義
{
int      c1;              // 整数変数
C        c2;              // チャネル変数
X x(p1, c2, c1);          // 子ビヘイビアxの宣言とポート接続
Y y(c1, c2, p2);          // 子ビヘイビアyの宣言とポート接続
void main(void)
  {
        par{ x.main( ); y.main( ); }     // xとyは並列
  }
};
```

図3. SpecC プログラムの例

イビアあるいはチャネルの記述は、オブジェクト指向的なプログラミングによって行えます。すなわち、各ビヘイビアとチャネルの仕様をクラスとして定義して、インスタンスの展開によってシステムの構造的な階層構造が決定されます。

この例ではビヘイビア Z が二つのサブ・ビヘイビアxとyを持っています。xはビヘイビアXのインスタンスであり、yはビヘイビアYのインスタンスです。par文の中で、xとyが並列に実行されると定義されています。xとyの間では整数c1とチャネルc2を経由して通信を行います。

2.3. SpecC 言語による設計方法論

SpecC 言語による設計手法を図 5.3-9 に示します。図に示されるように設計プロセスは大きく「機能設計」、「アーキテクチャ設計」、「実装設計」からなります。「機能設計」では、ステートマシンとアルゴリズム記述によってシステムの機能的な振る舞いを記述します。この機能仕様には実装の詳細は含まれていませんが、通信と計算のそれぞれの機能が一通り記

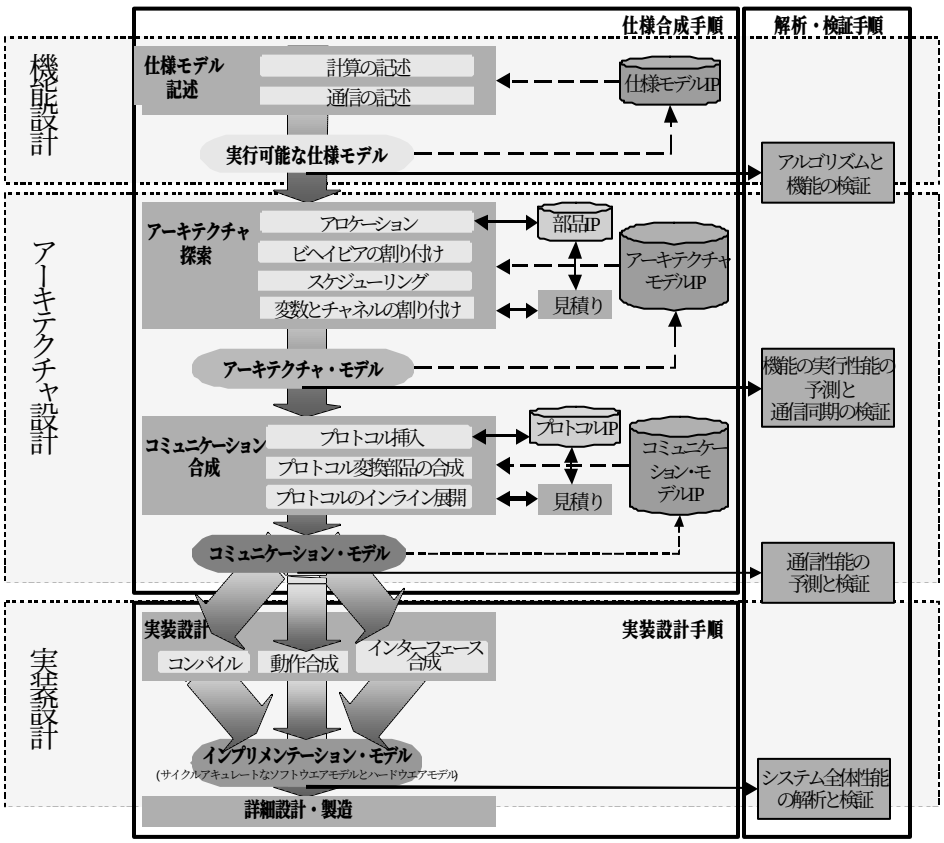

図4. SpecC 言語による設計フロー（文献[1]より）

述されるので、シミュレータによる実行が可能となります。このステップは、製品設計の初期段階におけるラピッド・プロトタイプを開発する工程に相当します。つまり、製品企画と基本機能を検討している段階で、機器の動作、製品の見た目、使い勝手などをユーザと同じ視点で検証するためのモックアップを開発することで、仕様の誤認や解釈の違いによる下流での設計変更・修正の削減が図れるなどの効果が期待できます。

「アーキテクチャ設計」は大きく「アーキテクチャ探索」と「コミュニケーション合成」に分かれます。「アーキテクチャ探索」では、まず、プロセッサ、メモリ、バス、カスタム・ハードウェアといったシステムを構成するコンポーネントの割り付け（アロケーション）を行います。次に、各ビヘイビアを各ハードウェアあるいはソフトウェア・コンポーネンする機能間での割り当ての変更と実行順序の依存関係の解析による並び替え（スケジューリング）を行います。これとともに、変数およびチャネルで表現されたビヘイビア間の通信をバス・コンポーネントに割り付けます。

「コミュニケーション合成」では、まず、バス・プロトコルの選択を行って通信部のモデル記述の具体化を行います。これは標準的なプロトコル IP を蓄積したデータベースから検索する形になります。次に、インタフェース回路とプロトコル変換部の合成を行います。最後に、プロトコル仕様の部分をチャネルからビヘイビアにインライン展開を行います。

「アーキテクチャ設計」を終えて「実装設計」に引き渡される設計仕様（コミュニケーション・モデル）は、バックエンドの下流設計ツールにそのまま渡せる形式になっています。つまり、インタフェース・モデルの設計仕様に含まれるソフトウェア部分の SpecC 言語コードは、C/C++のソースコードとして取り出せます。一方のハードウェア部分はビヘイビアレベルの HDL 記述として取り出すことができます。下流設計ツールとは、既存のコンパイラ、動作合成ツール、インタフェース合成ツール等が相当します。

SpecC 言語による設計手法の特徴は次の二点にあります。

- 機能設計から実装設計に至る各設計作業が、すべて、SpecC 言語によって記述された前段階の設計モデルに対して、置き換えあるいは洗練を行う形で定義されています。
- それぞれの設計作業を一貫的にかつ連続的に実行でき、抽象的な機能仕様の段階から実装設計までがシームレスにつながります。

また、各設計ステップでは、シミュレーションによる動作検証を行うことができます。たとえば、機能仕様の段階では、記述されている機能が設計意図に合致するかをシミュレーションで検証できます。機能分割とスケジューリングが終わった後では、異なるプロセッサ部品に振り分

けられたビヘイビア間が仕様どおりに同期するかをシミュレーションで検証できます。コミュニケーション合成の後では、設計モデルを使ったシミュレーションで、演算部と通信部を含めたシステムの性能まで検証することができます。
さらに、各設計ステップでの設計資産をIPデータベースとして蓄積・再利用することができますが、これらの IP 部品も SpecC 言語を使って開発することができます。

2.4. STOC (SpecC Technology Open Consortium)

STOC は、SpecC言語をもとにした業界標準のシステム仕様記述言語と仕様データ・フォーマット、および設計手法の開発と普及を目的として、1999 年11月 10 日に設立されました。
STOC には日米の組込みソフトウェア開発ツールベンダー、EDA ベンダー、自動車、電機、通信、情報・電子機器などの応用システム製造業、および大学など、約 50 の団体が参加しています(2001年1月現在)。活動状況は、STOC のホームページ（http://www.SpecC.org）で公開されているとともに、米国での DAC(Design automation conference)、ESC（Embedded systems conference)、国内の主要展示会でコンソーシアムのブースが設置され、セミナー等も開催されています。
2001年1月には言語仕様1.0が公表されましたが、この言語仕様はUCIが開発したオリジナルの言語仕様とほとんど差異はなく、コンソーシアムとしての公式バージョンを定めた形になっています。また、2001年6月にオープン・ソースコードでライセンス・フリーなリファレンス・コンパイラの公開が予定されています。STOCでは二つのワーキンググループ活動が行なわれています。「事例 WG」では、SpecC 言語による事例の仕様記述とその過程で得られるノウハウの収集を行い、SpecC 言語を用いる設計のガイドラインや方法論などの整備が行なわれています。
「言語仕様WG」は、SpecC言語の言語仕様の策定を目的とした WG です。現行の SpecC 言語仕様の評価とその拡張を検討し、より強力で、効率良く仕様記述が可能な言語仕様を策定します。

3. VisualSpec

CATS 社から販売されている仕様オーサリングツール VisualSpec[2]は、SpecC 言語を利用した仕様記述の作成、および機能仕様をアーキテクチャレベルの設計仕様に合成するプロセスを支援します。
図 5 に VisualSpec の設計環境の関連図を示します。「SpecC記述」では、SpecC言語の持つ各構文に合わせて、プログラム状態間の階層関係や有限状態機械、通信、例外処理の構造と動作をビジュアルに編集することができます。「SpecC 合成ツール」には、2.3 で説明したアーキテクチャ探索やコミュニケーション合成を行うためのソフトウェア／ハードウェ

ア分割などの合成コマンドが用意されています。「SpecC コンパイラ」と「SpecC シミュレータ」では、SpecC 言語のコードを Windows ホスト上で実行するためのランタイムエンジンとデバッガが用意されています。また、これ以外のオプションツールとして、SpecC コードの IP ライブラリの開発ツール、ドキュメント作成ツールなどがオプションとして用意されています。

VisualSpec と他の設計ツールとを連携したトータル設計環境について紹介します。

●ソフトウェアの詳細設計に向けては、VisualSpec で作成した設計データを CATS 社の ZIPC の拡張状態遷移表に変換できます。また、SpecC コードをそのまま WindowsCE、iTRON などのターゲット OS 上で稼動するランタイムエンジンも用意されています。

●ハードウェアの詳細設計に対しては、SpecC コードを VHDL コードに変換することができます。この VHDL コードは、YXI 社の XE（国内ではソリトン社から販売）などの動作合成ツールで RTL 合成が行えます。

●VisualSpec でソフトウェア／ハードウェア分割した SpecC コードは、ハードウェア部分を HDL 変換したコードとソフトウェア部分の SpecC コードとの間で協調シミュレーションが行えます。Mentor Graphics 社の HDL シミュレータ ModelSim とソフトウェア設計を協調動作させるために、Programming

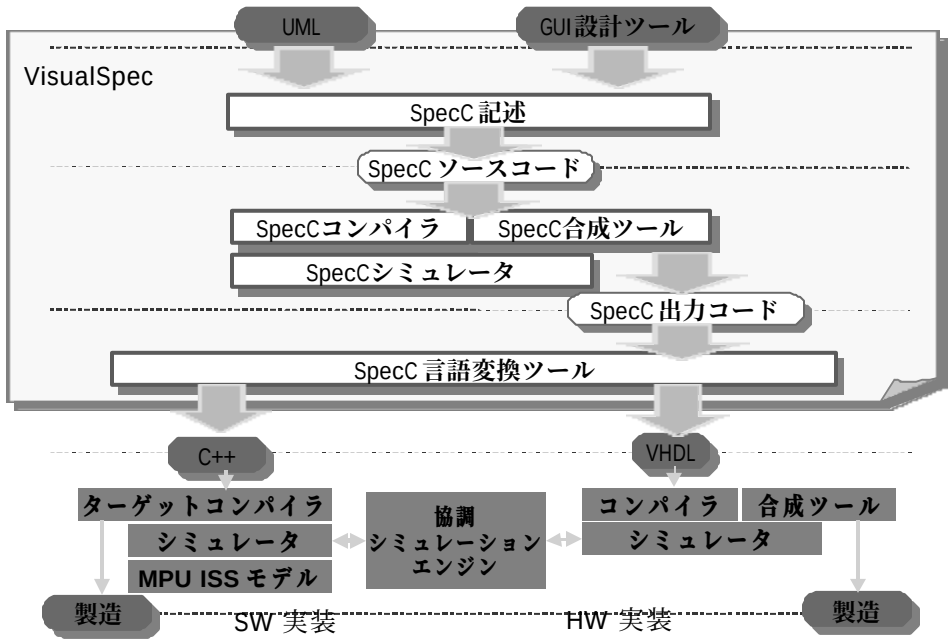

図5.　VisualSpecを利用したSpecC 言語向け設計ツールチェーン

Language Interface（PLI）を利用した通信コードが自動生成されます。

●VisualSpec と上流設計ツールとの関係では、組込みシステム向けの GUI 設計ツール（ILC 社の INTACORE など）と設計データの連携が行え、画面の設計情報から GUI の制御ロジックを生成できます。また、UML ツール（Rational Rose など）からステートチャート設計を SpecC コードに変換することができます。（2001 年夏に提供予定）

4．参考文献

[1]D.Gajski et al.，木下常雄他訳「SpecC：仕様記述言語と方法論」CQ 出版，2000 年 12 月

[2]木下常雄、石井忠俊 他，「組込みソフトと LSI を C 言語でコデザイン!」，Design Wave Magazine 2000 年 7 月号特集，CQ 出版，2000 年 7 月