組込みマイコン用ソフトウェア開発向けシステムシミュレータ

System Simulator for Embedded Microcomputer Software Development

富永　芳章　Y. Tominaga
三浦　貴　T. Miura
五十嵐　真悟　M. Igarashi

組込みマイコン用のソフトウェアの品質と開発効率を向上させる目的で，システムシミュレータを開発した。システムシミュレータは，CPUだけでなく周辺回路や制御対象などのシステム全体を模擬するもので，テスト用ハードウェアやテスト専用機器などを使用することなく，ソフトウェアのテストができるようになる。

つまり，ハードウェアが存在しない開発の初期段階から，ソフトウェアの検証ができるため，組込みマイコン用ソフトウェアの開発でもっとも困難であったテスト工程を大幅に効率化し，ソフトウェアの品質を向上させることができる。

We have developed a system simulator as a tool for testing embedded microcomputer software. This tool simulates not only a CPU but the entire embedded system, including peripheral devices and the external environment such as the objects being controlled. With this simulator, input-output procedures can be tested without using actual test hardware or emulators.

This simulator offers an efficient testing process, which is most difficult when developing embedded microcomputer software in the conventional way. It also improves the quality of the software itself.

1 まえがき

組込み用マイコンは，家電製品，産業用機器やプラント制御システムまで広く利用されており，これを制御するソフトウェアも多様化，複雑化している。

これらの組込みマイコンを利用したシステム（以下，マイコン組込みシステムと称呼）は，通常それ自体はソフトウェア開発環境をもたない。このため，ホストマシンと呼ばれる別のコンピュータシステムを利用して，ソフトウェアを開発する方法（このような開発方法はクロス開発と呼ばれる）が一般的であるが，ソフトウェアのテストでは，テスト用に試作されるハードウェアとマイコン用のエミュレータ（対象の組込みマイコンの代わりにソフトウェアの実行を可能にするための機器）などの専用機器が必要となるため，次のような問題があった。

(1) 開発スケジュールの問題　一般にマイコン組込みシステムは，ソフトウェアとハードウェアが並行に開発されるため，ハードウェアの開発が遅れると，実製品上でのソフトウェアのテストが行えない。つまり，ソフトウェアの開発スケジュールがハードウェアの開発スケジュールに拘束される。

(2) エミュレータの数量の問題　エミュレータは，組込みマイコン用ソフトウェアのテストを行ううえで強力なツールである。しかし，高価であるために，テストにかかわる技術者に行き渡るだけの数量を確保することが難しく，テストの効率向上のボトルネックになっている。

ここで紹介する組込みマイコン用ソフトウェア開発向けのシステムシミュレータは，これらの問題を解決するものであり，テスト用のハードウェアやエミュレータなどの専用機器を利用しないでテストすることができる。

なお，当社は組込みマイコン用のソフトウェア資産の蓄積と応用を促進する目的で，ソフトウェア開発ツールのコマンド，ユーザインタフェースなどを統一したUDE（Unified Development Environment）開発システムを整備している。このシミュレータもUDE開発システムの一環として開発した。

2 システムシミュレータに求められる要件

マイコン組込みシステムは，図1のような構成が一般的である。従来のシミュレータのほとんどはCPUシミュレータであって，ホストマシン上で図1のCPUコア（命令を解釈実行するCPUの中核）を模擬するツールであった。しかし，組込みマイコン用ソフトウェアのバグの多くは，ハードウェアとの入出力処理に潜んでおり，CPUシミュレータを利用しただけでは，バグの発見や品質の確保が難しい。

ハードウェアの開発スケジュールに拘束されずに，確実に

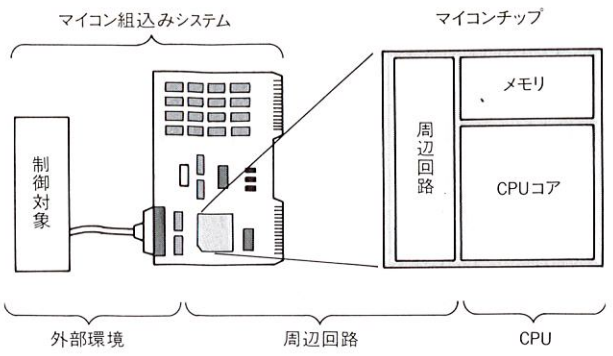

図1．マイコン組込みシステムの構成　マイコン組込みシステムは，CPU コア，周辺回路および外部環境の三つから構成される。
Configuration of embedded microcomputer system

入出力処理のテストを行うには，CPU だけでなく周辺回路や制御対象となる機器を含む外部環境までのシステム全体のシミュレーションが必要である。

さらに，CPU コア，周辺回路および外部環境は物理的には並列に動作するため，これらの動作タイミングなども考慮したデバッグやテストには，各部分が並列に動作するシミュレータが不可欠である。

また，システムシミュレータはさまざまなマイコンのソフトウェア開発に利用されるため，複数の CPU コアへの対応や，CPU デリバティブ（同一の CPU コアを利用し，チップ上の周辺回路の構成などが異なる派生製品）への対応が容易であることが望まれる。さらには，制御対象の機器を含む外部環境は製品ごとにまったく異なるため，これらの外部環境を含んだシミュレータを短期間で容易に提供できる実現方式が望まれる。

3 シミュレーション方式

3.1 四つのシミュレータ

前述したシステムシミュレータに対する要件は次のとおりである。

(1) マイコン組込みシステム全体をシミュレートする。
(2) CPU コア，周辺回路，外部環境を並列に動作させる。
(3) CPU デリバティブへの対応や，ユーザごとに異なる外部環境への対応が容易である。

これらの要件を満たすには，システムシミュレータは CPU コア部，周辺回路および外部環境をそれぞれソフトウェア的に独立かつ並列に動作する形で実現しなければならない。

このシステムシミュレータでは，全体を CPU コア部のシミュレータである“CSIM”（Core SIMulator），周辺回路部のシミュレータである“PSIM”（Peripheral SIMulator）および外部環境シミュレータである“ESIM”（Environment SIMulator）から構成することで，前述の要件を実現している。また，各シミュレータ間の同期と CSIM と PSIM のデータの受渡しの効率化を図るために，マイコン内部のバス（CPU がメモリや周辺回路とデータの受渡しを行うための信号線）をシミュレートする“BSIM”（Bus SIMulator）を組み込んだ（**図2**）。

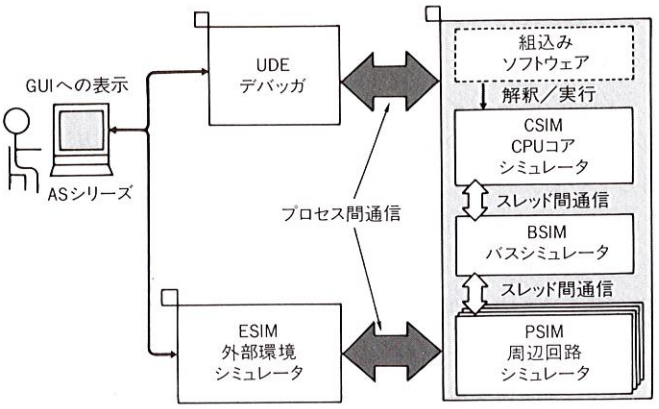

図2．システムシミュレータの全体構成　CSIM，BSIM および PSIM はスレッド，ESIM および UDE デバッガは別プロセスとして構成する。
Configuration of system simulator

3.2 シミュレータの実装方式

前述のシミュレータ群をそれぞれソフトウェア的に独立かつ並列に動作させる方法として，それらをプロセスと呼ばれる形で実現する方法がある。プロセスは，オペレーティングシステム（OS）がプログラムを管理するための単位であり，UNIX(注1)などの OS は複数のプロセスを同時かつ並列に動作させる機能をもっている。したがって，各シミュレータをプロセスとして実装することにより，それぞれを並列に動作させることができる。

しかし，プロセスは異なるプロセスのデータを直接参照することができないため，シミュレータ間で入出力要求やデータのやり取りなどが発生した場合，プロセス間通信でこれを実現しなければならない。このプロセス間通信は，通常のデータの参照に比べてはるかに低速であり，シミュレーション速度を大幅に低下させる恐れがある。

各シミュレータを並列に動作させるためのもう一つの実装方法として，それらをスレッドという単位で実現する方法が考えられる。スレッドとは，一つのプロセス内部をさらに細かく分割したプログラムの流れであり，Solaris(注2)，Windows NT(注3)などの OS で利用することができる。スレッドは，プロセスの内部で定義されているデータを共有しながら並列に動作することができるため，同じプロセスの内部で動作する

(注1) UNIX は，X/Open カンパニーリミテッドがライセンスしている米国ならびに他の国における登録商標。
(注2) Solaris は，Sun Microsystems 社の商標。
(注3) Windows NT は，Microsoft 社の商標。

スレッド間でデータの受渡しを行っても，単なるデータの参照となり，プロセスのように通信を行う必要はない。

したがって，高速にデータの受渡しを行うことができるが，スレッドを利用すると，プログラムの独立性が低下する可能性がある。例えば，あるスレッドに変更を加えた場合，その変更が他のスレッドにまったく影響を与えないものであっても，変更されたスレッドが含まれるプロセス全体を再構築する必要がある。

このため，このシミュレータではシミュレータ間の独立性の程度と，データの受渡しの頻度を考慮して二つの方法を併用している。

CPU コアと周辺回路は，アクセスが頻繁であり，データの受渡しを高速で行う必要がある。このため，CSIM と PSIM は同一プロセス内の別スレッドとして実装し，データの受渡しの高速化を図った。

また，外部環境は CPU チップのユーザごとにどのような応用製品に用いられるかがまったく異なる。さらに，外部環境は，各製品のユーザインタフェースなどのように，周辺回路に比較して，遅い応答でも十分な場合がほとんどである。このため，ESIM はプログラムの独立性を高く保つため，別プロセスとして実装した。

3.3 シミュレータ間の同期

組込みマイコン用ソフトウェアでは，外部の機器をリアルタイムに制御するために，入出力のタイミングや応答時間が厳密に規定される場合があり，ソフトウェアがこの仕様を満たしているか否かをテストする必要がある。このため，CSIM と PSIM に共通の時計を設定し，一定時間ごと（これを基準時間と呼ぶ）に，CSIM と PSIM の動作を同期させる機構を設けている。

この基準時間を細かくとるほど，精度の高いシミュレーションが可能となるが，その分計算量が増加するため，シミュレーション速度は低下する。つまり，シミュレーションの精度と実行速度はトレードオフの関係にあるため，基準時間をどの程度の分解能とするかは重要な問題となる。

このシステムシミュレータでは，マイコン内部のバスが発生するタイミングを基準時間として採用し，BSIM がこの時間を CSIM と PSIM に供給する構成とした。

また，入出力のタイミングや応答時間を厳密に考える必要のない，論理的なテストだけを高速に行いたい場合は，基準時間による同期の機能を停止させることで，高速なシミュレーションが可能となる。

4 システムシミュレータの構成

図 2 に示したシステムシミュレータ内の各シミュレータは，以下のように動作する。

CSIM は，ターゲットの CPU 用のプログラムをホストマシン上で仮想的に実行する。高速なシミュレーションを行うために，CSIM は対象のプログラムをあらかじめ仮想命令に変換しておき，これをインタプリタによって処理している。

BSIM は，CSIM および PSIM の同期に必要な基準時間を生成し，これを通知する。さらに BSIM は，CSIM に接続されている PSIM に関する情報を管理する。PSIM はシステムの初期化時に BSIM に登録され，CSIM から PSIM へのアクセスは BSIM が保持する PSIM の接続情報を利用して行われる。これにより，PSIM の接続に関する詳細を CSIM から隠蔽(ぺい)することができ，さまざまな CPU デリバティブに対応して，容易に PSIM を接続できるようになる。

シミュレーションの実行制御には，UDE デバッガを利用する。UDE デバッガを利用することによって，エミュレータなどの他のテストツールと共通のコマンド，GUI (Graphical User Interface) が利用できる。

ESIM はマイコンチップの外部の制御機器などを模擬し，PSIM からの出力を GUI に表示する，ESIM 上での GUI の操作を PSIM への入力として送信するといった処理を行っている。ESIM によっては，**図 3** のように製品の外観を GUI として使用したシミュレータが開発可能である。このような GUI を利用すれば，実製品を操作するのと同等の感覚で対話的な入出力処理のテストが可能となり，効率よくテストが行える。一方，このような GUI は製品ごとに開発しなければならないため，1〜3 人・月程度の期間と工数が必要と思われる。より早期に ESIM を利用したテストを行う必要がある場合は，**図 4** のグラフ，発光ダイオード (LED) のような製品に依存しない GUI を利用した汎(はん)用の ESIM の利用が可能である。

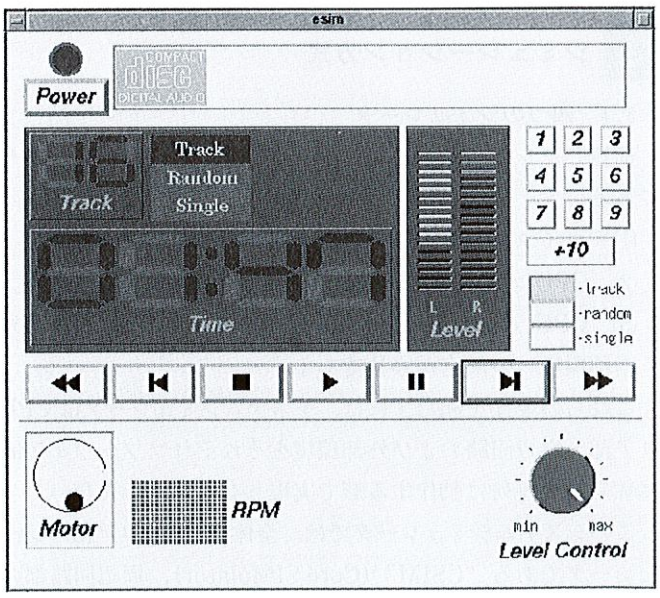

図 3．ESIM の開発事例　画面は試作した CD プレーヤの ESIM。ソフトウェアのテストが対話的な操作でできる。

Example of ESIM display

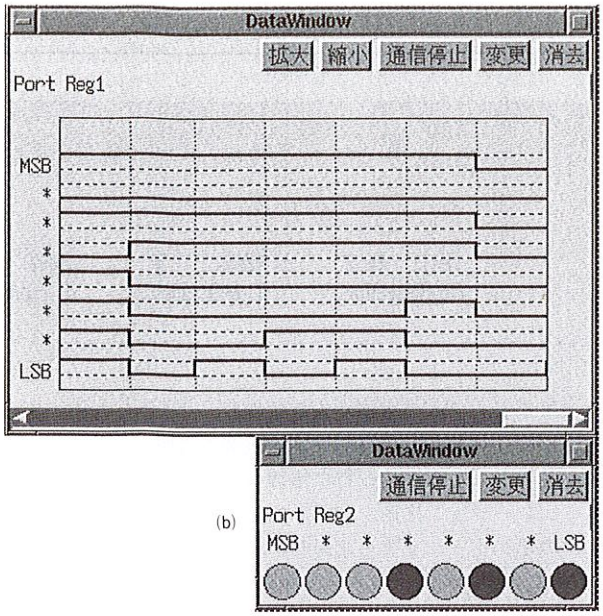

図4．汎用ESIM　ポートの出力やレジスタの値などをグラフ(a)やLED(b)など製品に依存しないGUIで表示する。

Example of universal ESIM display

現在，システムシミュレータは，当社のマイコンTLCS$_{TM}$-900Hシリーズを対象として開発している。また，このシミュレータの動作環境は，次のとおりである。

(1) プラットホーム：ASシリーズ

(2) OS：Solaris 2.3以上

(3) ウィンドウシステム：X Window System，Version 11 Release 5以上

CSIMのシミュレーション速度は，プラットホームをAS4085とした場合，実際のTLCS$_{TM}$-900Hの約1/100となっている。

5 システムシミュレータ利用の評価

このシステムシミュレータを，機能を絞ったCDプレーヤ用のソフトウェア開発に対し試行・評価した。この結果，以下のような効果が確認された。

(1) 入出力処理のバグの発見が容易　例えば，ソフトウェア内部で変数の値と，LEDの表示が対応していないといった，実製品上でのテストでしかわからなかったような入出力処理のバグを容易に発見することができた。実製品開発に適用した場合，これらのバグ全体の70％程度は，システムシミュレータで発見できる見込みである。

(2) 操作仕様の評価が可能　実製品を利用せずに操作性を評価できた。これにより，操作仕様の不備や矛盾に伴うバグを早期に発見することができた。

(3) テストの効率化　エミュレータなどの専用機器を用いることなく，同時に多人数でのテストが可能になった。実製品開発においては，テストの期間を20％～50％程度短縮できると見積もられる。

つまり，システムシミュレータを利用することによって，ハードウェアが完成する以前に，ソフトウェア全体をテストすることが可能となった。これにより，実製品開発においてコンカレントエンジニアリングが実現可能となり，開発期間の短縮など生産性の向上が期待できる。

さらに，早期かつ容易にバグを発見することができ，品質の作込みが容易になった。

6 あとがき

組込みマイコン用ソフトウェア開発ツールとしてのシステムシミュレータの特長と効果について述べた。今後は，シミュレーションの対象の拡大を図っていく。また，製品ごとに異なるESIMを早期かつ安価に提供するための開発支援環境を作る予定である。

ここで紹介した例では，時間による同期機能を停止してシミュレーションを行ったため，実用に耐える実行速度が得られたが，実製品の開発ではより精密なシミュレーションが必要になる場合も考えられる。このため，CPUコアのシミュレーション速度の高速化を図る。

文　献

(1) 山田勝康，他：マイクロコントローラの開発環境，東芝レビュー，**49**，11，pp.815-818 (1994)

(2) 岡島正明，他：組込みソフト開発支援のためのシステムシミュレーション環境，第51回情報処理全国大会予稿集，**5**-259 (1995)

富永　芳章　Yoshiaki Tominaga

1981年入社。プログラム開発支援ツールの設計に従事。現在，半導体システム技術センター　マイコンシステムソフトウェア技術部主務。

Semiconductor System Engineering Center

三浦　貴　Takashi Miura

1991年入社。マイコンの開発環境設計に従事。現在，半導体デバイス技術研究所システムLSI技術開発部。

Semiconductor Device Engineering Lab.

五十嵐　真悟　Masato Igarashi

1989年入社。組込みソフトウェア開発支援ツールの研究開発に従事。現在，研究開発センター　システム・ソフトウェア生産技術研究所。

Systems & Software Engineering Lab.