1. 組込みシステム実現技術

一色 剛[†]

キーワード ■ 組込みシステム, 組込みソフトウェア, 組込みプロセッサ, システムオンチップ, ハードウェア・ソフトウェア強調設計, プラットフォームベース設計

1. ま え が き

組込みシステムは、現代の高度情報化社会を根底から支え、我々の生活のありとあらゆる場所に存在する極めて普遍的なコンポーネントである。具体的には、産業用途（産業ロボット、プラント制御）、設備機器（照明、空調、エレベータ）、運輸（自動車、鉄道、航空、船舶、交通制御）、医療機器、OA機器、PC周辺機器、通信機器（携帯電話、ネットワークルータ）、家電（冷蔵庫、炊飯器、洗濯機、エアコン）、AV機器（テレビ、ビデオ、デジカメ）、娯楽（ゲーム、音楽再生端末）等、その応用分野は年々拡大してきている。このように拡大の一途を辿っている組込みシステムの実現技術を牽引してきたのは、マイクロプロセッサやメモリーを始めとする半導体技術と、多種多様な機能を実現するソフトウェア技術であり、これら技術の目覚ましい進歩に伴い、システムの機能性、動作環境、動作仕様、設計規模、設計手法、設計環境、開発モデル等が急激に変化している。

本稿では、このように日々進化し続ける組込みシステムの実現技術について、設計論的な側面からその動向や課題について解説する。

2. 組込みシステムの構成要素

組込みシステムの定義は、その応用範囲の広さゆえに包括的な扱いは必ずしも適切ではないが、その一つとして、「専用の機能を実行するために、ハードウェアとソフトウェアが緊密に統合されたコンピュータシステム」¹⁾という記述は、組込みシステムの特徴を現すキーワード（専用の機能、ハードウェア、ソフトウェア、コンピュータ）を含んだ的確な表現である。組込みシステムの構成要素は、その機能を定義付けるソフトウェア、これを実行するマイクロプロセッサ、メモリー（SRAM、DRAM、SDRAM、EEPROM、FLASH等）、専用回路（周辺装置を制御するためのデバイスインタフェースや専用演算処理回路）および

周辺装置（スイッチ、LED、LCDパネル、スピーカ、通信ポート、制御対象機器）からなる。

初期の組込みシステムでは、これらの構成要素はそれぞれ標準半導体デバイス（マイクロプロセッサ、メモリー）および専用機能デバイス（ASIC）をボード上に組み立てる方式であり、ここでの主な開発項目は、システム機能を実現するためのデバイス選定、ASICデバイス開発、ボード開発、およびソフトウェア開発である。この時にシステムを差別化する主な要因は、ソフトウェアによる充実した機能の提供と、ソフトウェア処理では重すぎる部分のハードウェア化（ASIC化）である。その後、半導体デバイスの集積度の向上により、これらの構成要素の大部分を1チップで実現するシステムオンチップ（SoC）が出現し、システムレベルでの設計最適化（コスト削減、省サイズ化、省電力化、高性能化、高機能化）の機会は格段に増えたものの、この大きな設計自由度を実際の組込みシステム製品としての差別化へ有効に活用する設計手法や開発環境の確立は、今後の大きな課題として残されている。この課題については、以後ソフトウェアとハードウェアの両面から具体的に論ずることにする。

3. 組込みソフトウェア開発

ソフトウェアは、前述の組込みシステムの定義の通り、組込みシステムの中心的な構成要素である。「組込み」ソフトウェアが「組込みでない」システム（PC、WS、サーバ）のソフトウェアと大きく異なる点は以下の通りである。

- (1) 厳しい資源制約：強いコスト削減、省サイズ、省電力の要求から、限られたメモリー量とプロセッサ処理能力しか与えられない場合が多い。
- (2) 高い信頼性：自動車やプラント制御のように、いかなるシステム障害も重大事態を引き起こす可能性があるため、徹底したシステム検証による高い信頼性を保証する必要がある。
- (3) リアルタイム性：システム外部環境の変化（外部イベント）に対して、定められた時間内（デッドライン）に応答することを保証する「リアルタイムシステム」である場合が多い。このようなリアルタイム性を実現するために、リアルタイムOS（RTOS）を組込んだ

[†] 東京工業大学 理工学研究科 集積システム専攻
"Embedded Systems Implementation Technology" by Tsuyoshi Isshiki
(Department of Communications and Integrated Systems Graduate School of Science and Engineering, Tokyo Institute of Technology, Yokohama)

マルチタスクプログラミング開発が必要となる¹⁾²⁾。

- (4) クロスプラットフォーム開発環境：通常PCやWS上（ホストプラットフォーム）に存在する組込みソフトウェア開発環境と、ソフトウェアが起動するターゲットプラットフォーム（組込みシステム）が異なる開発を「クロスプラットフォーム開発」と呼ぶ。

3.1 RTOSとマルチタスクプログラミング

最近の組込みシステムにおける複数の入出力に対するリアルタイム応答要求やアプリケーションの大規模化に伴い、RTOSとマルチタスクプログラミングによる組込みソフトウェア開発が広く浸透している。RTOSは、組込みソフトウェアの要件である「厳しい資源制約」「高い信頼性」「リアルタイム性」を満たすために、マルチタスク処理機構、ソフトウェア・ハードウェア資源管理機構、省メモリ構成、高信頼性、アプリケーションに応じたカスタマイズの容易性等を提供する。現在の主力RTOSとしては、汎用OSから派生したWindows CE, Embedded Linuxのほか、日本市場ではITRON/T-Engineが、欧米市場ではVx Works, Symbian OSを始め数多く存在する³⁾。

マルチタスクプログラミング²⁾は、複数の入出力に対するリアルタイム応答を実現するために、アプリケーションを複数のプログラムユニット（タスク）に分解し、これらタスクがプロセッサ資源を時間的に共有しながら並行処理をする仕組みを実装する手法である。特にリアルタイム制約を満たすためには、入出力デバイスの特性（イベント発生頻度、イベントの周期性）や入出力処理の特性（処理時間、リアルタイム制約）を考慮した適切なタスク分割や、RTOS上のタスクスケジューラのための適切なタスク優先度の設定が必要であり、使用するRTOSの特性とシステム仕様の十分な理解が必要となる作業である。

3.2 ターゲットハードウェアに密着した開発

前述のクロスプラットフォーム開発環境では、組込みシステムの動作に必要な完結したソフトウェアシステムを作り上げる必要があり、ターゲットプラットフォームのハードウェアアーキテクチャに対する十分な理解が不可欠となる。例えば、システム電源投入後のシステム初期化シーケンスとして、ブート、ハードウェア初期化、メモリーシステム初期化、RTOSの初期化、RTOSの起動、そして最終的にアプリケーションの起動までを正確に実装することや、各ハードウェアデバイスを制御するためのデバイスドライバの実装も必要である。

3.3 高いデバッグ・検証コスト

組込みソフトウェアのデバッグ・検証には、膨大なコストと時間を要する。この理由としては、第一に、システムとして高信頼性が要求されるため徹底したデバッグが必要であることや、並行処理マルチタスクによるアプリケーション実装のためバグの特定が非常に困難であること、システム動作が外部環境と密に連動する場合は実環境でのシス

テム検証が必要となりデバッグ機能が大幅に限定されること、等が挙げられる。さらに、ソフトウェアの開発段階では、ターゲットプラットフォームは単なるプロトタイプであり、ハードウェア自体が不安定である場合も多く、ハードウェアの不具合（設計ミス、組立てミス）が発生し、バグの特定を一層難しくすることもある。

4. 組込みハードウェア開発

「組込みハードウェア」という言葉は、あまり一般的ではないが、ここでは、組込みソフトウェア以外の構成要素、すなわちプロセッサと専用回路（デバイスインタフェースと専用演算処理回路）を指すものとする。

4.1 組込みプロセッサ

組込みシステムは、マイクロプロセッサ技術の進歩に密接に関わっている。商業的に成功した最初のマイクロプロセッサであるインテル4004⁴⁾は、当時のビジコン社の電卓のために専用に開発されたものであり、現在の組込みシステムの原型であると位置づけることができる。その後もしばらくは、4ビット・8ビットマイコン（マイクロコントローラ）が組込みシステムの心臓部として使われ、ソフトウェア開発はもっぱらアセンブリ言語が中心であった。現在の代表的な「組込みプロセッサ」としてMIPS, 68000, PowerPC, ARM, SH等が挙げられるが⁵⁾、これらの多くは（SHを除く）、PCやWSをターゲットとした32ビット汎用プロセッサとして開発されたものがベースである。これら組込みプロセッサは、大規模なソフトウェア開発のためのC言語などの高級プログラミング言語による開発環境の提供は必須であり、その他の要件として、省電力、省プログラムサイズ、小回路規模等が重要である。組込みプロセッサは、単体の標準品として製造される場合や、SoC上の機能回路ブロックの一つ（プロセッサコア）として埋め込まれる場合がある。

これら既存の組込みプロセッサと並んで最近注目を集めている新しい技術が、Application-Specific Instruction-set Processor (ASIP)^{6)~8)}である。このASIP技術は、ターゲットアプリケーションに特化した命令セット、データパス、メモリーシステムおよび専用のコンパイラを豊富な設計ツール群によって直接合成する技術であり、組込みシステムの今後のアーキテクチャ設計手法を担う重要な技術といえよう。

4.2 専用回路設計

SoC以前の組込みシステムの世代では、プロセッサと専用回路は別々のデバイスであり、システムのパフォーマンスの向上やターゲットシステム特有のハードウェア要件は専用回路（ASIC）によって実現されていた。この間、ASIC開発はVLSI設計技術の中心的な課題であり、さまざまな設計自動化技術（自動配置配線、論理合成、高位合成）が研究・実用化されてきた。特に、自動配置配線技術と論理合

成技術は、ASIC設計のみならず、あらゆるVLSI設計において必要不可欠な設計技術であり、幅広く普及している。一方、高位合成技術(C言語およびその拡張言語により計算アルゴリズムを記述し、これを効率的に処理する専用回路を合成する技術)は、近年実用化が進んできており、実際の製品開発にも活用されている^{9) 10)}。

固定化した機能のみを実現するASICは、一般的に多品種少量生産の傾向にあり、設計開発コストおよび製造初期コスト(マスク費用等)をいかに抑えるかが重要である。ASICの設計方式としては、スタンダードセル(レイアウト済みの論理ゲートセルをライブラリとして用意し設計コストを削減する)、ゲートアレイ(トランジスタ形成までの製造工程を共通化し、設計回路をメタル配線によって実現する)とFPGA(Field Programmable Gate Array: プログラム可能な論理デバイス)がある。特にFPGAは、製造初期コストがなく、設計した論理回路がプログラムすることで実現できるため、極めて短期間に開発することができ、ハードウェアのプロトタイプ開発として広く活用されているほか、組込みシステムの最終製品の部品として搭載される場合も多い。最近のFPGAは、多数のメモリーブロックや組込みプロセッサコアを内蔵するなど豊富な機能を提供している^{11) 12)}。

5. システムレベル設計

これまで、組込みシステムにおけるソフトウェアとハードウェアについて個別に議論してきたが、当然のことながらこれらの開発は密接に関連しており、これらを含めたシステム全体の設計最適化手法の確立が、特に今後のSoC開発において大きな課題である。

5.1 ハードウェア・ソフトウェア協調設計

ハードウェア・ソフトウェア協調設計(以後HW・SW協調設計と呼ぶ)という概念は、本格的なSoC時代以前から高位合成技術に続く次の設計パラダイムとして盛んに研究されてきており¹³⁾、システムレベル設計手法の原点ともいえる。この協調設計の定義は、ハードウェアやソフトウェアがそれぞれ何を指すかに大きく依存するが、最も広義に解釈した場合、ハードウェアは組込みプロセッサと専用回路であり、ソフトウェアはアプリケーション、デバイスドライバ、OS等である。この中では、以下のような設計フローが必要となる。

- (1) HW・SW分割：アプリケーションを組込みプロセッサで処理するソフトウェア部と、専用回路で処理するハードウェア部への分割を行い、これらのインタフェース(プロセッサと専用回路を接続するバス仕様、通信プロトコル等)を定義する。
- (2) HW・SW並行開発：ハードウェア部は、高位合成ツールによるRTL(Register Transfer Level: 論理合成可能なアーキテクチャ記述)生成や手動によるRTL設

計を行う。ソフトウェア部は、採用する組込みプロセッサに向けたコードチューニングや、デバイスドライバ開発、OSポーティングを行う。

- (3) HW・SW協調検証：システムの動作検証は、すべてのハードウェア部品(プロセッサ、専用回路)をRTLモデルでシミュレーションするために膨大な時間を要するため、プロセッサ部を、より高速な命令セットシミュレータ(ISS: Instruction Set Simulator)を用いて、専用回路のRTLシミュレータと協調させたHW・SWコシミュレーション(Co-simulation)技術が有効となる。

特にHW・SW分割問題は、プロセッサアーキテクチャ、プロセッサ数、専用回路部アーキテクチャ、システムバスアーキテクチャ等、数多くの設計要素に大きく依存するため、自動化が非常に難しい設計工程である。

5.2 プラットフォームベース設計

前述のHW・SW協調設計と並んで重要視されている概念が、設計再利用(Design Reuse)とプラットフォームベース設計である。設計規模の拡大に対する設計生産性の相対的な低下は、ハードウェア・ソフトウェアに共通した深刻な問題である。ハードウェア開発に関しては、ムーアの法則(半導体集積度は1.5年で2倍に向上)に従った進歩し続けた半導体製造技術に設計技術が追いついていない「(ハードウェア)設計危機」が長年警鐘されている一方、ソフトウェア開発でも最近の組込みシステムのソフトウェア規模の膨大化(携帯電話や自動車ナビゲーションシステムは数百万行のコード規模と言われる)による「ソフトウェア危機」が叫ばれている。これら双方の設計危機の克服には、革新的な設計技術の創出による設計生産性の大幅な向上と、設計再利用による設計工数の削減が必要である。

設計再利用は、過去の設計資産や外部から提供されるIP(Intellectual Property)モジュールを、現在行っている設計に組み込むことを指す。一方、プラットフォームベース設計は、設計再利用の概念を有効活用するためのハードウェア・ソフトウェアそれぞれの設計手法であり、ここではIPモジュールインタフェースの標準化と、この標準インタフェースに基づいたIPライブラリと関連ツール群からなる開発環境がベースとなる。ハードウェアプラットフォームでは、組込みプロセッサ、標準バス、周辺回路IP群(外部メモリーインタフェース、シリアルIOインタフェース、ネットワークインタフェース等)から構成される。ソフトウェアプラットフォームとしては、ITRON²⁾に見られるようなOSカーネルや、ミドルウェアのインタフェース標準化とこれらに準拠したソフトウェア部品群から構成される。ハードウェアとソフトウェア両方を網羅するプラットフォームとしては、携帯電話の分野で普及しているOMAP(Open Multimedia Applications Platform)があり、ここでは、RTOS、ミドルウェア、アプリケーション、組込みプロセ

ッサ (ARMプロセッサとTI社のDSPからなるデュアルコア)、ハードウェアIP等、組込みシステムの構成要素をすべて網羅する完結したプラットフォームである¹⁴⁾。

IPを活用したプラットフォームベース設計は、システムの最適性のある程度犠牲する代わりに開発効率を最優先し、アプリケーションソフトウェアを中心とした製品の差別化を図るというアプローチであり、開発品種の増加や設計期間の短縮の圧力が特に顕著な応用分野で普及している。

5.3 マルチコアSoC

今後のSoCアーキテクチャの大きな方向性として、複数のプロセッサコアを搭載したマルチコアSoC (MPSoC: Multiprocessor SoCとも呼ぶ) が挙げられる¹⁵⁾。マルチコア化は、汎用プロセッサの分野で進んでいるほか、前述のOMAPプラットフォームにもあるように、組込みシステムの分野でも急速に広まりつつある。マルチコアSoCは、専用回路部で実現していいたいくつもの機能をプログラム可能なDSPやASIPで実現することにより、SoCとしての柔軟性を持たすという大きなメリットがある。この柔軟性により、一つのSoCでいくつもの異なった組込みシステムへの搭載が可能となることによる市場拡大の効果や、製造後のバグ修正・ファームウェア更新が可能となることによる製造リスク低減や製品の寿命延長の効果が期待できる。

マルチコアSoCの設計手法はまだ充分には確立しておらず、多くの課題を抱えている。

- (1) 非対称 (ヘテロ) アーキテクチャ: プロセッサコア、バス構造やメモリー構造すべてにおいて非対称なアーキテクチャを持つ場合が多く、アプリケーションに最適なアーキテクチャの導出はその設計空間の広さのために非常に困難である。
- (2) ソフトウェア開発環境: マルチコアSoC上でのソフトウェア開発は、多数のツールチェーンを駆使した非常に複雑で困難な工程であり、アプリケーションの分割、コア間の通信実装、性能チューニング、デバッグ・検証、マルチコア用RTOS開発等を十分に支援する開発環境が必要である。

6. 今後の組込みシステム開発

システムオンチップ (SoC) の出現により、組込みシステムは新たな時代に突入したと言える。そこでは、組込みシ

ステムの機能を実現するほとんどの構成要素を1チップに集積化できるために、さまざまなシステムレベルでの設計最適化が可能であるが、個々の最適化問題が困難なために、十分に効果的なシステムレベル最適化が可能な開発環境はまだ整っておらず、今後とも大きな研究課題である。逆に、現在の最大の関心事は、システムレベル最適化よりも、プラットフォームベース設計の普及に見るように、設計効率化であると言える。プラットフォームベース設計は、今後とも拡大していくことが考えられるが、共通のプラットフォーム上での製品開発は、差別化要因がソフトウェアか最悪の場合Time-To-Marketだけとなり、激しい競争と製品の著しい短命化を招く恐れがあり、新たな差別化要因を創出する仕組みを作ることが、今後の設計プラットフォームの課題と言えよう。

(2007年4月3日受付)

〔文 献〕

- 1) Q. Li and C. Yao: "リアルタイム組込みOS", 翔泳社 (2005)
- 2) 坂村 健: "μITRON4.0標準ガイドブック", パーソナルメディア (2001)
- 3) N. Laengrich: "ARMプロセッサ向けOSの理解", Information Quarterly, 3-2, pp.62-66 (2005)
- 4) Intel 4004, www.intel4004.com
- 5) J. Turley: "Embedded Processors", Extreme Tech解説記事 (www.extremetech.com) (Jan. 2002)
- 6) 雀 秀一: "コンフィギャラブル・プロセッサARC", Design Wave Magazine (Oct. 2000)
- 7) Tensilica, www.xtensa.com
- 8) ASIP Solutions, www.asip-solutions.com
- 9) 若林一敏: "システムLSIのC言語ベース設計の現状と将来", DAシンポジウム (2006)
- 10) 山田晃久ほか: "C言語からの設計システムBachを用いたMPEG-4 IPコアの開発", シャープ技報, 78 (2000)
- 11) Xilinx, www.xilinx.com
- 12) Altera, www.altera.com
- 13) G. De Micheli and R.K. Gupta: "Hardware/Software Co-design", Proc. IEEE, 85, 3, pp.349-365 (Mar. 1997)
- 14) G. Martin and F. Schirrmeister: "A Design Chain for Embedded Systems", IEEE Computer (Mar. 2002)
- 15) G. Martin: "Overview of the MPSoC Design Challenge", DAC (2006)

いっしき 剛 1990年、東工大電気・電子工学科卒業。1992年、同修士。1996年、UC Santa Cruz PhD取得 (Computer Engineering)。1996年、東工大助手、2000年、同助教授、2001年、東大VDEC助教授、2003年、東工大助教授。主な研究分野は、マルチメディアLSI設計、システムレベル設計環境、マルチコアSoC用アプリケーション分割手法、通信合成手法等。