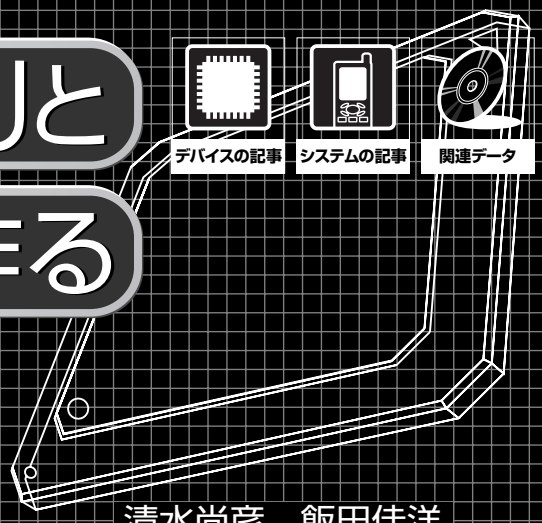
FPGA搭載用CPUとソフト開発環境を作る



第1回

コンパクトな16ビットCPUを設計する

清水尚彦，飯田佳洋

本連載では，FPGA搭載用16ビットCPUソフト・マクロの開発事例を紹介します．開発に際しては，小規模の組み込み機器で利用しやすいコンパクトなCPUを目ざしました．アーキテクチャとしては歴史的なプロセッサである「PDP-11」を選択していますが，開発自体はフルスクラッチで一から設計しています．またCPUソフト・マクロだけでなく，アセンブラやCコンパイラ，リアルタイムOSなどのソフトウェア開発環境も整備しました．自分でCPUやソフトウェア開発ツールを作りたい技術者の方々にも，FPGA用の自由度の高いCPUソフト・マクロを利用したい技術者の方々にも役に立つ内容としていきたいと思います．（筆者）

今回のCPU開発プロジェクトを，筆者らは「POP-11（PDP-11 on programmable chip）」と名づけ，次の二つのCPUを開発しました．

1）小規模な組み込みCPUソフト・マクロ：POP-11/10

POP-11/10は，小規模な組み込み機器向けの16ビットCPUです．メモリはSRAMを前提として設計しました（FPGAの内部メモリだけで動作させることを想定している）．後述のPDP-11/40からオプション（乗除算コプロセッサ，メモリ管理ユニット，DMAコントローラなど）をすべて取り払った構成を基本としています．ただしXOR演算などの簡単な拡張機能や割り込みコントローラ，RS-232-Cコントローラ，タイマなど，組み込み用途に必須と思われる機能は取り込んでいます．

2）UNIX搭載用のCPUソフト・マクロ：POP-11/40

1）のCPUソフト・マクロに，UNIX OSを動作させるために必要な拡張機能を採り入れたものがPDP-11/40互換のCPUソフト・マクロ「POP-11/40」です．メモリ管理機構を導入するとともに，IDEインターフェースやSDRAMインターフェースなどを持たせています．本連載の最終回では，このCPUに米国Caldera社のUNIX V6[注1]を移植します．

アーキテクチャのベースであるPDP-11は，米国DEC（Digital Equipment Corp.）が開発した16ビットのミニコン（小型コンピュータ）です．1970年に発表されたモデル20（10,800ドルという当時としては画期的な低価格で発売された）を皮切りに，1990年のモデル94まで20年にわたって市場に投入され，米国AT&T社 Bell Laboratoriesで開発されたUNIXを搭載したことによって特に有名になりました（**写真1**）．

〔写真1〕PDP-11のハンドブック
インターネット上の古本屋で運良く入手することができた．

●古いアーキテクチャは「使えない」？

汎用のコンピュータ・アーキテクチャの世界は1990年代を境として，パイプラインとコンパイラ指向のアーキテクチャへと大きく変わってきました．その流れに乗れないアーキテクチャやメモリ空間の壁に阻まれたアーキテクチャは次々と市場から姿を消しています．PDP-11も16ビット・アドレスの限界と，比較的複雑でパイプライン化が難しい

注1：Caldera社は，2002年に16ビットのUNIX OS（V1からV7まで）をオープン・ソースとした．いわゆるBSD形式のライセンス形態を採っており，Caldera社の著作権表示を削除せず，ライセンスの内容に注意を払いさえすれば，改変したものも含めてソース・コードとバイナリの再配布が許されている．

アドレス指定方式を持っていたため，マイクロプロセッサ化（LSI化）しても生き残ることが難しいものでした．

それなのに，いまさらなぜ古いアーキテクチャを持ち出してくるのか？と思われる方も多いかと思います．この開発の動機はいくつかありますが，大きな動機は「組み込み分野では汎用のCPUと異なるアプローチが取れるのではないか」という期待と，設計技術教育という観点です．

1）コンパクトさが強みとなる「組み込み」

組み込み分野には高集積・高速の世界から低価格・中低速の世界まで，幅広い応用があります．FPGAを論理回路部分に利用する組み込みシステムで，それほど高性能をねらわない分野であれば，FPGA上に安価に利用できるCPUがあるとたいへん助かります．

例えば，本プロジェクトで開発したPOP-11/10は，約1,700論理セルで基本部分を構成できます（米国Altera社のFPGAに実装した場合）．そしてFPGAの内部メモリにOSとアプリケーション・ソフトウェアを収めれば，1チップのリアルタイム・システムが完成することになります[注2]．

2）システム的な視野を養う

電子回路などの分野では，大学の教育は微分方程式や線形代数の基礎から応用分野まで，演習付きで時間をかけて教育するカリキュラムを作り上げています．一方，組み込み応用を代表とするディジタル技術分野では，論理回路とソフトウェア，アーキテクチャなどをバラバラに教育しているだけで，システムとしての動作を，演習を踏まえて教育するようなしくみはあまりないように感じています．

中堅以上の組み込み技術者の方は，その開発経歴の中でハードウェアもソフトウェアもOSもすべて理解できる程度のプロジェクトを扱った経験があるかと思います．しかし，最近の複雑化したプロジェクトではどうしても専門分野を細分化する必要が生じがちです．しごとのペースも速く，残業に追われて担当分野以外を勉強する余裕はなくなってきています．その一方で，専門分野が細分化されればされるほど，幅広い領域をカバーするシステム的視野を持つことが重要になってきます（**図1**）．

本稿で扱うアーキテクチャや論理回路，コンパイラ，OSは，どれも若い技術者や学生が短期間に十分理解できる程度の小さな規模です．それでいて，簡単な組み込み応用に利用できる程度の実用性を備えています．この連載を徹底的に理解するまで読みながら，かつ自分の手で動かしてみていただければ，システム的視野のとっかかり程度は養えるのではないかと考えています．

連載の第1回目にあたる今回は，CPUをどのように設計したかを解説します．なお，本誌付属のCD-ROMに，本稿で開発したCPUに簡易モニタ・プログラムを付けたコンフィグレーション・ファイル（CQ出版から販売されている「FLEX10KE評価キット」用）を収録しています．お手元に環境のある方は，実際に手を動かしながら本稿を読んでいただければと思います．

どのようにCPUを設計するか？

CPU設計にはさまざまな方向性があります．処理能力を向上させる，実装面積を減らす，消費電力を下げるなど，目的によって要求が変わってきます．今回のようにFPGAで動作させるときは，利用可能ゲート数がもっとも大きな制約になります．できるだけ小さく収まるような設計を心がけました．

また，なるべく変更を加えずにUNIXを動作させるため，PDP-11との互換性も失わないように配慮しました．

●回路を小さくするには？

回路を小さくするには，大きなビット幅を持ったラッチ回路，すなわちレジスタ・ファイルを減らせば，必要な論理ゲートを大きく減らすことができます．また，機能の切り分けやデータパス記述の最適化も回路の縮小と動作周波

〔図1〕細分化された専門分野だけでなく，幅広い領域をカバーする視野を持つことが重要になってくる

注2：さらにロー・エンド向けとして，筆者らの研究室では400論理セルで構成できる独自の16ビット・プロセッサを公開している（http://shimizu-lab.dt.u-tokai.ac.jp/cpu.htmlを参照）．ただし，ツール類のことを考えると一般的な実用には向いていない．