本家お墨付きのCPUコア！

Linuxも起動できる本物MIPSコアの無償評価版が自由にFPGAに実装できる!

FPGA実装向けコア“MIPSfpga SoC”でLinuxを起動！（ハードウェア設計編）

阿部 道夫 Michio Abe

本誌前号（No.18）では，MIPSfpgaシステムのRTLコードを用いた開発の手順について説明しました．例題の周辺回路としては，LED表示やスイッチ入力という簡単なものでしたが，MIPSfpgaを使用した一連の設計手順を知ることができたと思います．今回は，Linuxのために大容量メモリとしてDDRメモリをつなぎ，さらに周辺機能としてUARTやEthernetコントローラを実装したSoCを実現してみます．

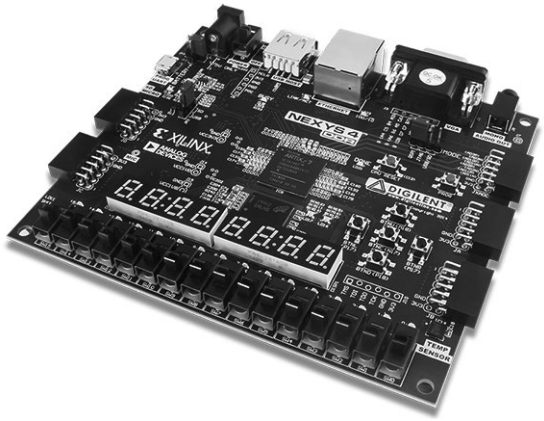

写真1　Artix-7搭載Nexys4 DDRボード（Digilent社）に**MIPSfpga SoCを実装！**

http://store.digilentinc.com/nexys-4-ddr-artix-7-fpga-trainer-board-recommended-for-ece-curriculum/

1 MIPSfpga SoCとは

今回はMIPSfpgaのCPUコアを使ったSoCの設計手順について説明します．SoCはSystem on a Chipの略称で，シリコン・チップ上にCPUコアを中心として，必要な機能や周辺回路を実装し，1つのシステムを実現したものになります．

今回の説明で使用する設計の手法は，前回とは異なり，RTLにより直接回路を記述するのではなく，Vivadoのブロック設計の機能を使って，各種周辺回路の機能ブロックを配置し，接続してSoCを実現していきます．

このMIPSfpga SoC設計では，まず，MIPS CPUコアをAHB Lite to AXIバス・ブリッジを経由してAXIバスに接続します．さらに，このAXIバスにVivadoで提供しているメモリ・ブロックやGPIOを接続し，それにLEDを接続したシステムを設計します．そして回路をコンパイルし，実際にFPGAに書き込んで動作させます．この一連の手順により，ブロック設計の方法を知ることができると思います．

次にUARTや割り込みコントローラ，DDRメモリ/Ethernetコントローラを接続し，SoCとして最小限の機能を持ったシステムを構成して，実際にFPGAで動作させていきます．

そして次回になりますが，今回設計したSoCのシステム上で動作するLinuxカーネルとファイル・システムのビルドとインストールの方法について説明し，実際にMIPSコアを実装したFPGA上でLinuxを動作させる手順について説明します．

MIPSfpgaで提供しているmicroAptiv MIPS CPUコアは，命令/データ・キャッシュやメモリ・マネジメント・ユニット（MMU）が装備されているUPタイプと呼ばれるコアになるので，フル機能のLinuxを動作させることが可能です．

ターゲットとなるFPGAボードは前回同様に，Xilinx社製FPGAであるArtix-7を搭載したNexys4 DDRボード（Digilent社）です（**写真1**）．

2 MIPSfpga SoC設計の準備

今回設計を進めるにあたって，前回（本誌No.18）で説明したように，開発用PCの準備と関連の開発ツールのインストールが完了していることを前提としています．まだ完了していない方は，前回の開発環境のインストールを参照して準備をしてください．前回の設計の手順をまだ試していない方は，説明に従い実際に設計を行うことをお勧めします．

設計を開始する前に，Vivadoは2014.4バージョンがインストールされていることを確認してください．これより新しいバージョンでは説明の通りにならないだけでなく，エラーが発生してしまいます．

また，このSoCの動作の確認とLinuxのコンソールとして使用するため，Tera Termのようなターミナル・ソフトウェアがPC上に必要になるので，インストールされていない場合は事前にインストールしておいてください．