Appendix 3　FPGA開発ツールやRISC-V用コンパイラのインストール

ハードウェア&ソフトウェア開発環境の構築

石原 ひでみ Hidemi Ishihara

1. 開発用PCの準備

● RISC-V用コンパイラがLinux環境用なので…

当初，開発用PCにはWindows環境でも対応できるよう考えていたのですが，RISC-V用コンパイラをWindows環境でコンパイルしたところ，うまくビルドできませんでした．時間もなかったため，今回は開発用PCにはLinux環境を用意することにしました．筆者はUbuntu 16.04.2 LTSをインストールして使用しています．

● 仮想マシン上ではうまく動かない場合も？

また筆者は，Windows上にVirtualBoxやVMwareなどの仮想マシンを使用してUbuntuをインストールしたPCが1台あるのですが，このPCではUSB接続のFPGAダウンロード・ケーブルの認識が正常に行えない場合がありました．可能であれば，仮想マシンを使わずにPCに直接インストールしたUbuntuを用意するのが安全のようです．

2. ハードウェア開発ツールのインストール

● MAX 10なら Quartus Prime

ターゲットFPGAとしてMAX 10を使う場合は，FPGA開発ツールとしてQuartus Prime（Intel社，旧Altera社）をインストールしてください．無償で利用できるライト・エディションで構いません．筆者はバージョン17.0を使用しました．なお開発用PCの環境がUbuntuなので，Linux版をインストールしてください．

● Artix-7なら Vivado

ターゲットFPGAとしてArtix-7を使う場合は，FPGA開発ツールとしてVivado（Xilinx社）をインストールしてください．無償で利用できるWebPACKエディションで構いません．筆者はバージョン2017.2を使用しました．なお開発用PCの環境がUbuntuなので，Linux版をインストールしてください．

3. ソフトウェア開発ツールのインストール

● RISC-V用クロス・コンパイラが必要

今回実装したRISC-VはRV32IMでI（整数命令などの最小限の命令セット）とM（乗除算命令）を実装しています．RISC-VはRV32I，RV64Iの他にF（浮動小数点命令），A（アトミック命令）などを必要に応じて実装していきます．コンパイル環境も，実装したRISC-Vの命令セットに合わせて構築しないと，実行できない命令コードをコンパイラが出力してしまう状況が発生します．ここでは，実装したRISC-Vに合わせたクロス・コンパイル環境を構築します．

また，RISC-Vのテスト・パターンはバイナリ形式で配布されているわけではなく，アセンブラのソースコードで記述されているため，テスト・パターンを実行する場合にコンパイル環境が必要になってきます．そこでクロス開発環境を整備してから，RISC-Vのテスト・パターンをコンパイルしてシミュレーション環境で使用できるように変換してテストを実施していきます．

● 標準構成ツールチェーンを使う場合

ツールチェーンのバージョンにこだわらない場合は，**図1**に示すウェブ・サイトから標準構成ツールチェーン（本誌では標準構成と呼ぶことにする）をダウンロードしてビルドします．

標準構成では，クロス・コンパイラを含めたソフトウェア・ツールは，次のコンポーネントから構成されます．

- riscv-gnu-toolchain：RISC-Vクロス・コンパイラ
- riscv-fesvr：Host-Target InterFace（HTIF）上のホスト・プロセッサとターゲット・プロセッサ間でコールを処理するフロントエンド・サーバ
- riscv-isa-sim：ISAシミュレータと実行環境
- riscv-opcodes：シミュレータによって実行可能な全てのOPコードを列挙
- riscv-pk：RISC-V Newlibポートでビルドおよびリンクされたコードによって生成されたシステム・コールを処理するプロキシ・カーネル
- riscv-tests：組み合わせテストとベンチマークのセット