Google もNVIDIA も注目している ARM 時代の「弱い者の味方」

ご購入はこちら

オープンソースCPU「RISC-V」の研究

第9回 FPGA上のRISC-VでMyプログラムを動かす

@msyksphinz

Rocket ChipがFPGAで動作するようになったので，あらかじめ提供されているプログラムだけでなく，自分で作成したプログラムを動作させます（**図1**）．このためには，自分でプログラムを作成し，SDカードに書き込んで，Rocket Chipに書き込んで動作させる必要があります．

● SDカードに書き込むMyプログラムを作る

Rocket Chipで動作するプログラムは，まずは自由に書けばよいでしょう．ここでは，**リスト1**に示すようなプログラムを作成しました．GCD（最大公約数）を求めるプログラムです．

このプログラムをコンパイルして，Rocket Chipで実行するためのバイナリを作成するに当たり，riscv-tests/benchmarksに入っているベンチマーク・プログラムのコンパイル方法を参考にしました．このベンチマーク・プログラムは，ライブラリに相当するような部分や，外部とのインターフェースを行う部分をあらかじめsyscalls.cやcrt.Sにまとめてあります．printf()の動作や，システム・コールなどの動作をこの関数群で肩代わりするという仕組みです．詳細は前回でも解説しました．前回は，riscv-tests/benchmarksディレクトリに移動して新たにgcdディレクトリを作成してビルドを行いました．その際に生成したgcd.riscvをSDカードにコピーして動作させましょう．

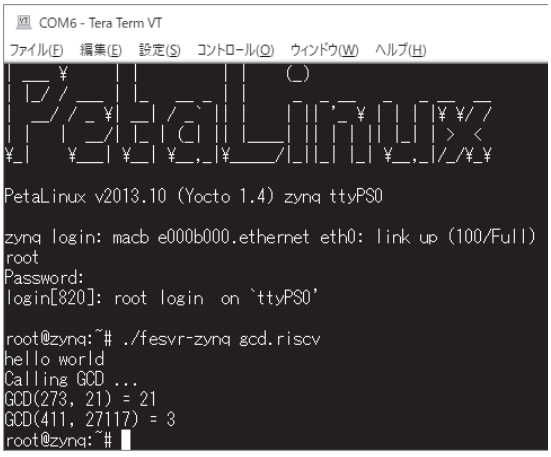

図1　実験成功…ZedBoard搭載FPGA上のRISC-V（Rocket Chip）でMyプログラム（GCD）を動かせた

● MyプログラムをSDカードに書き込む

このプログラムを先ほど作成したSDカードに書き込む必要があります．RISC-V動作用のSDカードに新しいファイルを追加するためには，fpga-zynq/zedboardに移り，make ramdisk-openやmake ramdisk-closeを使ってファイル・システムの中身を編集します．

```
make ramdisk-open↵
mkdir ramdisk↵
dd  if=fpga-images-zedboard/
uramdisk.image.gz  bs=64 skip=1 | ¥↵
gunzip -c | sudo sh -c 'cd ramdisk/
&& cpio -i'↵
[sudo] msyksphinz のパスワード：
127899+1 レコード入力
127899+1 レコード出力
8185554 bytes （8.2 MB, 7.8 MiB）
copied, 8.89832 s, 920 kB/s
35892 ブロック
```

ramdiskディレクトリを参照すると，fpga-images-zedboard/uramdisk.image.gzの中身が展開され，編集できるようになっています．

```
$ cd ramdisk/↵
$ ls -lt↵
合計 52
drwxr-xr-x   2 root root 4096   9月
```

注：本稿の内容は執筆時点のもので，随時更新されていく可能性があります．

編集注：本誌2018年2月号特集2「新時代プロセッサ システム作り」ではバークレイ以外のさまざまなコアも紹介しています．