2017年度最終レポート

天野英晴

1. pocoにminus flagとCMP命令を導入

- POCOの大小比較はbmi, bplを用いるため、SUB命令で引き算を行う必要がある。このため、比較対象のレジスタの片方が破壊される。
  - これを防ぐためminus flagを設け、CMP（Compare命令）により設定する。
  - bmi, bpl命令はminus flagを見て分岐するかを決める
- 新規命令
  - CMP rd, rs　Compare: rd-rsの結果が負になればminus flagをセット
  - R型　　00000dddsss11001
- 改造する命令
  - bmi, bpl: minus flagの結果により分岐、指定するレジスタの値は無視
- 実行
  - tar xvf poco_flag.tar
  - pocoのディレクトリの下で、
  - make でiverilog可能
  - ./shapa max.asm –o imem.dat
  - ./a.out | moreでシミュレーション実行
- 確認：max.asmでr3が9になればOK。
- 提出物：poco.v

2.同じ命令をパイプラインプロセッサpocop.vに実装せよ

- pocopのディレクトリの下で
- make
- ./shapa max.asm –o imem.dat
- ./a.out | moreで実行
- 同じくr3が9になればOK
- 提出物：pocop.v

3.iseで合成せよ

- １，２の設計をiseで合成し、動作周波数、LUT数を評価せよ。
- 公平な比較を行うために、メモリを含めた合成をする
  - １はfpga.vを、2はfpgap.vをトップ階層としてaddし、その下にメモリもaddしてから合成する。
- 動作周波数は適当に設定すること
- 提出物：両者のデータ

４．考察

- 今回の改造の利点と欠点を簡単に説明しなさい
- 3の結果と共に4の考察はメールの本文に示すこと。