RISC-V测试说明

1. 文件说明

riscv.s： 汇编测试文件

riscv.lst： 对应的反汇编文件

riscv.hex: 对应的hex文件

reg\_all.dat： 部分寄存器值

其中tebench需要读入的文件为riscv.hex，运行完成后将最终寄存器结果和reg\_all.dat中的数据进行比对以确定rtl功能是否正确。riscv.s和riscv.lst为便于大家阅读和程序定位的汇编及反汇编文件。

1. 对测试用例的说明
2. 0x0000-0x00f4为寄存器初始化程序，根据手册建议，一般32位数据的初始化由一条lui和一条addi组成，注意立即数是符号位扩展的。

为防止流水线冲突，前31条指令分别写x1-x31寄存器的高位，后31条指令分别写x1-x31寄存器的低位，注意x0恒0。

**PC至0x00f4执行结束后各个寄存器值详见reg\_all.dat第一组。**

1. 0x010c-0x0174为一般alu类指令，注意add/sub指令忽略上下溢，slt/sltu/slti/sltiu四种指令均设置两条，其中第一条不满足条件，第二条满足条件。

**PC至0x0174执行结束后各个寄存器值详见reg\_all.dat第二组**。

1. 0x01a4、0x01bc、0x01d4、0x01ec为store类指令，注意sh指令会部分覆盖sw写入memory的数据，sb指令会部分覆盖sh写入memory的数据，指令写入后memory中的数据参照riscv.s中对应的sw/sh/sb指令注释。
2. 0x0204-0x0230为相对跳转类指令，每种跳转指令均设置两条，其中第一条不满足跳转条件，第二条满足跳转条件，跳转至对应的标号位置。为便于测试，程序跳出后执行一条addi指令再跳回原PC的下一条。
3. 0x024c、0x027c、0x02ac、0x02dc、0x030c为load类指令，其load的地址为前面store指令写入的相关地址，load到的数据见riscv.s中对应的指令注释。
4. 该用例未设置结束条件，0x0328为有效指令的最后一条，可参考以PC 0x032c-0x033c为仿真结束条件。

**PC至0x0328执行结束后各个寄存器值详见reg\_all.dat第三组**。

1. 不计入初始化指令，除x29、x30、x31外各寄存器均只被写入一次，如仿真检测到寄存器值不一致，可debug写对应寄存器的指令