RISC16 Simulator Debugging

January 20, 2016

1 Bugs

☑ In the sequential version, writting instructions in the program memory is not enough. It must first be saved in ROM, then imported back and only then can be executed.

→ SYou just need to press the "Assembly" button, you moron.

☑ If there are multiple comment symbols after an instruction, the whole line is ignored all together.

→ e.g. When importing `addi 1,1,0//double//comment`, the line does not appear in the program memory.

→ The problem probably comes from the comment parsing when importing the file in `MemProg::fileopen()`.

☑ When coding a label in the program memory, the said label is not recognized as an address and generates errors. The code needs to be exported and imported back again so that the label is set in the address column correctly.

→ We can only code using the ASM column, the others are read-only. However, the label needs to be in the Address column. Hence, when we input something like `loop:  beq 2,0,end`, the label `loop` stays in the same column as the opcode even though it should be considered as an address. To work around that, we can export and import back the code; the labels are then moved away from the ASM column to the Address column.

→ When using the `Assembly` function (through the dedicated button), the labels should be parsed and moved to the Address column.

→ The problem comes from `MemProg::getIns`, the function parsing the ASM instruction upon assembly. It supposes that the first token of the assembly line is always an instruction, otherwise it throws an "error : bad instruction" at the face of the poor user. We should probably first check if there is no label before the instruction.

→ Checking first for the label was a good idea. Now the label is correctly put in the address column upon assembly. However, when parsing the rest of the

code, if a label is used on a line and has not been defined before that line, it throws a warning fucking everything up ("Unknown label or format error"). Assembling twice, first to parse the label and second for the rest, does not help.

→ The label parsing needs to be a full step on its own. See the new `getLabel` loop in `MemProg`.

☐ The compiler can't handle a `movi` with a label not yet declared. It throws an error in `movi()` in `MemProg.java:442`.

2 Performance improvements

☐ When loading a ROM, the file is opened twice in a row.

→ See `Memprog::fileopen(String path)`.