Compilers

Lab 8

Piotr Błaszyński

20th April 2022

Tasks (explained later in the document):

- add support for floating point numbers,
  - generate computation code,
  - conversion or error reporting,

The handling of floating-point numbers in MIPS is implemented using a different register and instruction set than their integer counterparts. In addition, the implementation of references to floating-point constants should be simplified by treating them as variables. The floating-point registers equivalent to $\$t0 - \$t7$ are $\$f0 - \$f31$ - a single register is used to store float numbers, a pair of registers is used to store double numbers. The requirement in the compiler applies only to numbers of type float, so only instructions concerning them will be described further. To load the value of a variable into a register, use the instruction: $l.s\ register, variable$. To store the value from a register into a variable, use the instruction: $s.s\ register, variable$. To perform arithmetic operations operations:

- $add.s\ register\_result,\ register\_arg1,\ register\_arg2$ - adding,
- $sub.s\ register\_result,\ register\_arg1,\ register\_arg2$ - subtraction,
- $mul.s\ register\_result,\ register\_arg1,\ register\_arg2$ - multiplication,
- $div.s\ register\_result,\ register\_arg1,\ register\_arg2$ - dividing,

For example code that adds two floating point numbers:

```
z=3.14+6.28;
y=3.14+5.12;
```

It should be generated (a constant may occur once or more than once in the data section, if it occurs only once then all its occurrences are replaced with the same identifier):

```
.data
  z: .float 0
  float_var1: .float 3.14
  float_var2: .float 6.28
  y: .float 0
  float_var3: .float 5.12
  tmp1: .float 0
  tmp2: .float 0
.text
  l.s  $f0, float_var1
  l.s  $f1, float_var2
  add.s $f0, $f0, $f1
  s.s $f0, tmp1

  l.s $f0, tmp1
  s.s $f0, z

  l.s  $f0, float_var1
  l.s  $f1, float_var3
  add.s $f0, $f0, $f1
  s.s $f0, tmp2

  l.s $f0, tmp2
  s.s $f0, y
```

In case the language assumes the ability to convert between variables of floating point and integer types, it is necessary to load the value into a register and then call the conversion - conversions in both directions are performed on the floating point register.
Conversion from integer to floating point value:

```
.text
  li $t0, 10
  mtc1 $t0, $f0
  cvt.s.w $f1, $f0
```

Conversion from floating point to integer value:

```
.data
 float_var1: .float 3.14
.text
  li $t0, 10
  mtc1 $t0, $f0
  cvt.s.w $f1, $f0
  l.s $f2, float_var1
  add.s $f1, $f1, $f2
  cvt.w.s $f0, $f1
  mfc1 $t0, $f0
```