We only translate an if statement with contracts because otherwise Alloy can not verify the program.

1. The way an if statement is translated is that we create *sig*, function, *assert* and *check*. Where the sig represents the states of the variables used in the if statement and assert tries to assert that whenever the precondition is satisfied executing instructions in the if/else statement will satisfy the postcondition.
2. Multiplication is translated as follows: x\*y => x.mul[y]
3. Division is translated as follows: x/y => x.div[y]
4. Modulo is translated as follows: x%y => x.mod[y]
5. Addition is translated as follows: x+y => x.add[y]
6. Subtraction is translated as follows: x-y => x.sub[y]
7. Relational operations have the same syntax as their input syntax
8. Negation is translated as follows: !x => x
9. Conjunction is translated as follows: x&&y => x and y
10. Disjunction is translated as follows: x||y => x or y
11. Implication is translated as follows: x=>y => x => y
12. Equivalence is translated as follows: x <=>y => x = y

We don’t specifically support assertion commands because we decided that it is not very useful to convert the assertion to Alloy as we need to include every instruction in the language up to the assertion point and translate it to Alloy and this won’t be the same as the assert statement in programming languages such as Java for which the expression is checked during the runtime. However, an assertion can be implemented by creating an if statement where the precondition is true as the following:

If\_require(true)

If(true){

// body for which an assertion must be made

}

If\_ensure(expression) // replacing the assert(expression)