lib/main/integer-integral-domain.ath

```
load "integral-domain.ath"
load "integer-times.ath"

open Z

define Integer-Ring :=
  (renaming [Commutative-Ring.+ +  Commutative-Ring.* *
             Commutative-Ring.<0> zero  Commutative-Ring.U- negate
             Commutative-Ring.- -])

(print-instance-check Integer-Ring Commutative-Ring.Theory)

define Integer-Ring-1 :=
  (renaming [Commutative-Ring.+ +  Commutative-Ring.* *
             x`Commutative-Ring.<0> zero  Commutative-Ring.U- negate
             Commutative-Ring.- -
             Commutative-Ring-With-Identity.<1> one])

(print-instance-check Integer-Ring-1 Commutative-Ring-With-Identity.Theory)

assert (theory-axioms Commutative-Ring-With-Identity.Theory)

(!property Group.Double-Negation no-renaming Group.Theory)

(!by-instance-check Group.Double-Negation Integer-Ring-1 Commutative-Ring-With-Identity.Theory)

(!by-instance-check Group.Left-Inverse Integer-Ring-1 Commutative-Ring-With-Identity.Theory)

(!property Group.Left-Inverse no-renaming Commutative-Ring-With-Identity.Theory)

(!property Group.Left-Inverse Integer-Ring-1 Commutative-Ring-With-Identity.Theory)

(!property Group.Unique-Negation no-renaming Group.Theory)

(!by-instance-check Group.Unique-Negation Integer-Ring-1 Commutative-Ring-With-Identity.Theory)

##

define ZID :=
  (renaming [Commutative-Ring.+ +  Commutative-Ring.* *
             Commutative-Ring.<0> zero  Commutative-Ring.U- negate
             Commutative-Ring.- -  Commutative-Ring-With-Identity.<1> one])

(print-instance-check ZID Integral-Domain.Theory)
```