

```

1 THEORY Group
2 IMPORT THEORY PROJECTS
3 /SimpleDEq THEORIES /SimpleDEq/Monoid.dtf | org.eventb.theory.core.deployedTheoryRoot#Monoid
4 TYPE PARAMETERS G
5 OPERATORS
6 invertible <predicate> (op: (G × G) → G, e: G)
7 well-definedness G ≠ ∅
8 direct definition
9   ∀ x · x ∈ G ⇒ (∃ y · y ∈ G ⇒ (op(x ↦ y) = e ∧ op(y ↦ x) = e))
10 commutative <predicate> (op: (G × G) → G)
11 well-definedness G ≠ ∅
12 direct definition
13   ∀ x, y · x ∈ G ∧ y ∈ G ⇒ op(x ↦ y) = op(y ↦ x)
14 Group <predicate> (op: (G × G) → G, e: G)
15 well-definedness G ≠ ∅
16 direct definition
17   Monoid(op, e) ∧ invertible(op, e)
18 AbelianGroup <predicate> (op: (G × G) → G, e: G)
19 well-definedness G ≠ ∅
20 direct definition
21   Group(op, e) ∧ commutative(op)
22 inverses <predicate> (op: (G × G) → G, e: G, x: G, y: G)
23 well-definedness G ≠ ∅
24 direct definition
25   invertible(op, e) ⇒ (op(x ↦ y) = e ∧ op(y ↦ x) = e)
26 THEOREMS
27 inversesCommutative:
28   ∀ op, e, x, y · op ∈ ((G × G) → G) ∧ e ∈ G ∧ x ∈ G ∧ y ∈ G ∧ Group(op, e) ⇒
29     (inverses(op, e, x, y) ⇔ inverses(op, e, y, x))
30 latinSquare:
31   ∀ op, e, x, y · op ∈ ((G × G) → G) ∧ e ∈ G ∧ x ∈ G ∧ y ∈ G ∧ Group(op, e) ⇒ (
32     ∃ g · g ∈ G ⇒ (op(x ↦ g) = y ∧ (∀ g2 · g2 ∈ G ∧ op(x ↦ g2) = y ⇒ g = g2))
33   )
34 leftCancellation:
35   ∀ op, e · op ∈ ((G × G) → G) ∧ e ∈ G ∧ Group(op, e) ⇒ (
36     ∀ a, b, c · a ∈ G ∧ b ∈ G ∧ c ∈ G ⇒
37       ((op(a ↦ b) = op(a ↦ c)) ⇔ (b = c))
38   )
39 rightCancellation:
40   ∀ op, e · op ∈ ((G × G) → G) ∧ e ∈ G ∧ Group(op, e) ⇒ (
41     ∀ a, b, c · a ∈ G ∧ b ∈ G ∧ c ∈ G ⇒
42       ((op(b ↦ a) = op(c ↦ a)) ⇔ (b = c))
43   )
44 inverseEqn:
45   ∀ op, e · op ∈ ((G × G) → G) ∧ e ∈ G ∧ Group(op, e) ⇒ (
46     ∀ x, y · x ∈ G ∧ y ∈ G ⇒ (
47       op(x ↦ y) = e ⇔ inverses(op, e, x, y)
48     )
49   )
50 zeroInverse:
51   ∀ op, e · op ∈ ((G × G) → G) ∧ e ∈ G ∧ invertible(op, e) ∧ neutral(op, e) ⇒ inverses(op, e, e,
52     e)

```

**END**