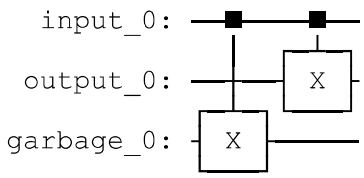
In [1]:
```
# 1) Show that the output is correctly written to the 'final output' register
  (and only to this register) when the 'output' register is initialized as |0>
```

In [2]:
```
from qiskit import Aer, execute, QuantumCircuit, QuantumRegister
from qiskit.visualization import plot_bloch_multivector, plot_histogram
```

In [3]:
```
input_bit = QuantumRegister(1, 'input')
output_bit = QuantumRegister(1, 'output')
garbage_bit = QuantumRegister(1, 'garbage')
```

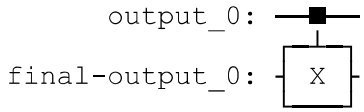
In [4]:
```
Vf = QuantumCircuit(input_bit, output_bit, garbage_bit)
Vf.cx(input_bit, garbage_bit)
Vf.cx(input_bit, output_bit)
Vf.draw()
```

Out[4]:



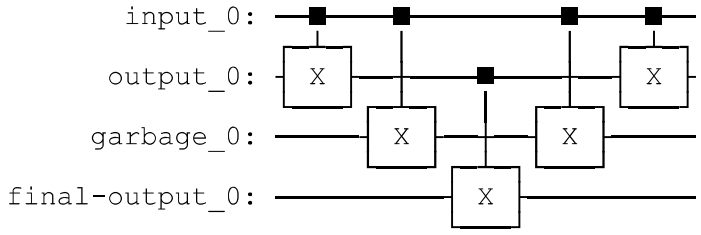
In [5]:
```
final_output_bit = QuantumRegister(1, 'final-output')

copy = QuantumCircuit(output_bit, final_output_bit)
copy.cx(output_bit, final_output_bit)
copy.draw()
```

Out[5]:



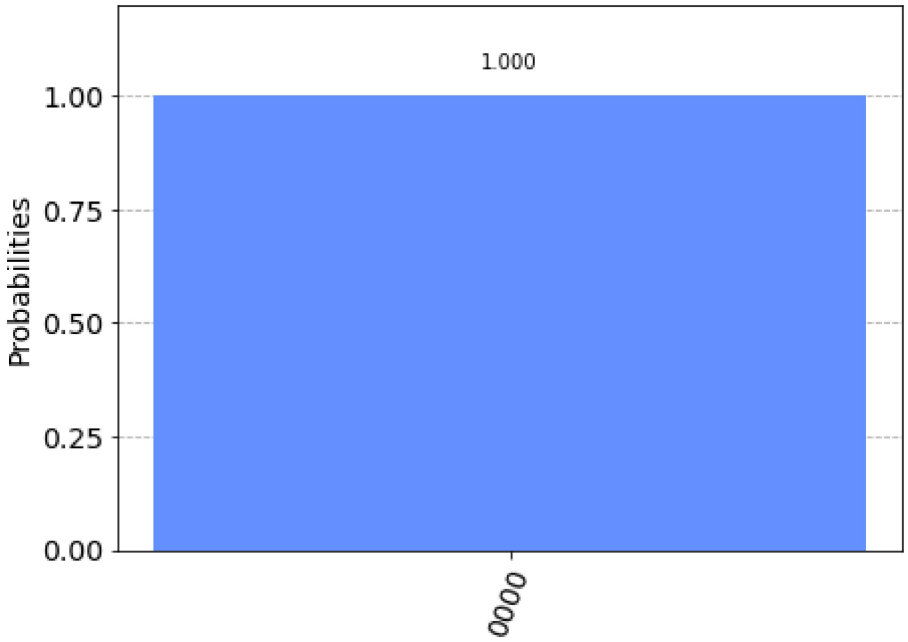
In [6]:
```
circ = (Vf.inverse() + copy + Vf)
circ.draw()
```

Out[6]:



In [7]:
```
sim = Aer.get_backend('qasm_simulator')
circ.measure_all()
result = execute(circ, sim).result()
counts = result.get_counts(circ)
plot_histogram(counts)
```

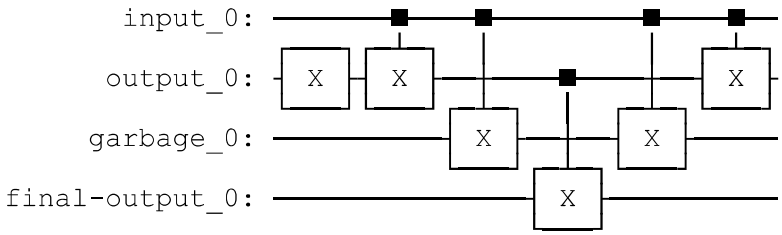
Out[7]:



In [8]:
```
#2) Determine what happens when the 'output' register is initialized as |1>
```

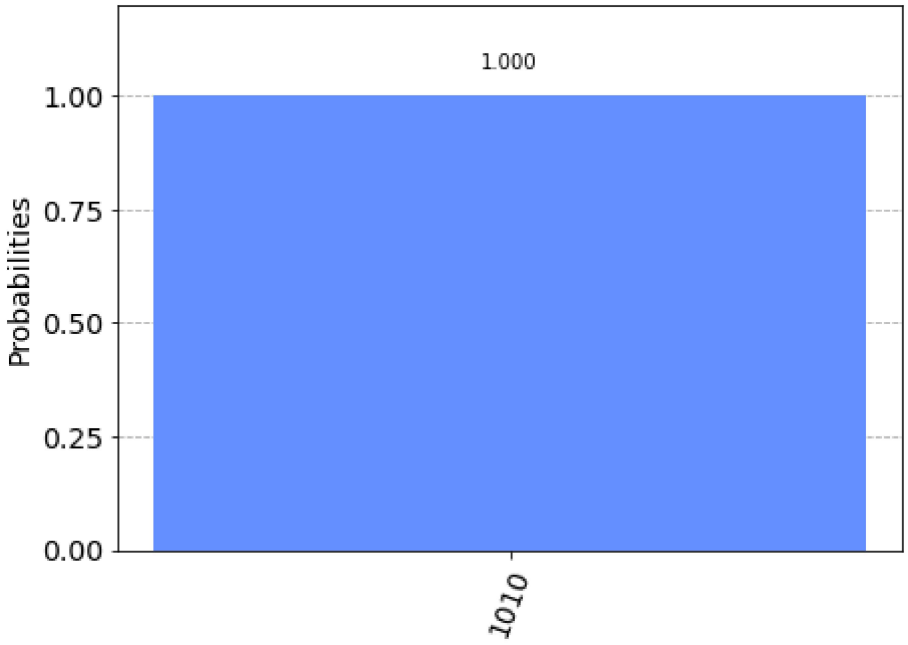
In [9]:
```
qc = QuantumCircuit(input_bit, output_bit, garbage_bit, final_output_bit)
qc.x(1)
qc = qc + Vf.inverse() + copy + Vf
qc.draw()
```

Out[9]:



In [10]:
```
qc.measure_all()
result1 = execute(qc, sim).result()
counts1 = result1.get_counts(qc)
plot_histogram(counts1)
```

Out[10]:



In [ ]:
```

```