In [1]:
```
import numpy as np

# Import standard Qiskit libraries
from qiskit import QuantumCircuit, transpile, Aer, IBMQ
from qiskit.tools.jupyter import *
from qiskit.visualization import *
from ibm_quantum_widgets import *
from qiskit.providers.aer import QasmSimulator
```

```
<frozen importlib._bootstrap>:219: RuntimeWarning: scipy._lib.messagestrea
m.MessageStream size changed, may indicate binary incompatibility. Expected
56 from C header, got 64 from PyObject
```

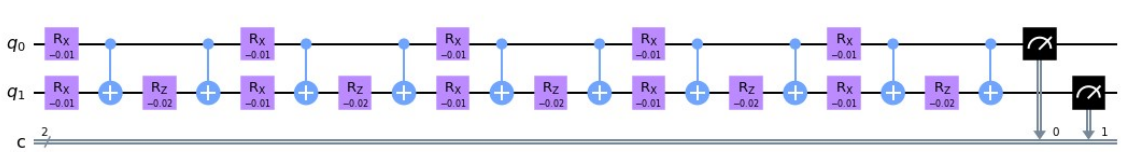
In [2]:
```
# Load your IBM Quantum account
provider = IBMQ.load_account()
```

In [3]:
```
### Physical parameters (atomic units) ###
J = 1.0    # Exchange coupling
B = 0.5    # Transverse magnetic field
dt = 0.01  # Time-discretization unit
ttot = 0.05
num_steps = int(ttot/dt)
```

In [4]:
```
### Build a circuit ###
```

In [5]:
```
circ = QuantumCircuit(2, 2)  # 2 quantum & 2 classical registers
for i in range(num_steps):
    circ.rx(-2*dt*B, 0)  # Transverse-field propagation of spin 0
    circ.rx(-2*dt*B, 1)  # Transverse-field propagation of spin 0
    circ.cx(0, 1)        # Exchange-coupling time propagation (1)
    circ.rz(-2*dt*J, 1)  #                                    (2)
    circ.cx(0, 1)        #                                    (3)
circ.measure(range(2), range(2))  # Measure both spins
circ.draw('mpl')
```

Out[5]:



In [6]:
```
### Simulate on OpenQASM backend ###
```

In [7]:
```
# Use Aer's Qasm simulator
from qiskit.providers.aer import QasmSimulator
backend = QasmSimulator()
```

In [8]:
```
# Transpile the quantum circuit to low-level QASM instructions
from qiskit import transpile
circ_compiled = transpile(circ, backend)
```

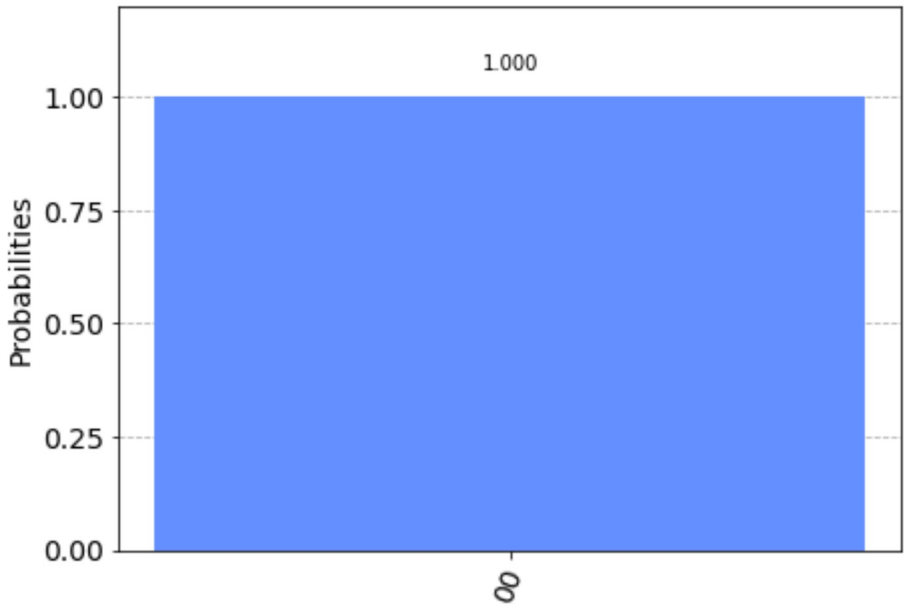
In [9]:
```
# Execute the circuit on the Qasm simulator, repeating 1024 times
job_sim = backend.run(circ_compiled, shots=1024)
```

In [10]:
```
# Grab the results from the job
result_sim = job_sim.result()
```

In [11]:
```
# Get the result
counts = result_sim.get_counts(circ_compiled)
```

In [12]:
```
# Plot histogram
from qiskit.visualization import plot_histogram
plot_histogram(counts)
```

Out[12]:



In [ ]:
```

```