LAB: Quantum Adder Circuit Design Using Qiskit

Komsan Kanjanasit

Abstract

Quantum adder circuits play a foundational role in quantum computing, serving as basic building blocks for more complex algorithms and quantum computational tasks. This manuscript explores the construction and operation of quantum half-adder and full-adder circuits using Qiskit. We detail the design and execution of these circuits, utilizing quantum gates to perform binary addition on quantum bits (qubits) and observing the results through measurement. This work demonstrates how fundamental arithmetic operations can be realized in the quantum realm.

1 Introduction

In classical computing, adders are crucial components for performing arithmetic operations. In quantum computing, we can construct similar circuits to perform addition using quantum bits (qubits). Unlike classical bits, qubits can exist in a superposition of states, allowing for parallel computation, which is integral to the efficiency of quantum algorithms. Quantum adders are used in algorithms requiring arithmetic operations, such as Shor's algorithm and quantum Fourier transform-based algorithms.

This manuscript presents the design of two types of quantum adders using Qiskit:

- **Quantum Half-Adder:** A basic circuit that adds two single qubits and outputs a sum and a carry.
- **Quantum Full-Adder:** A more complex circuit that adds two qubits and an input carry, producing a sum and a carry output.

2 Theory

2.1 Quantum Gates

Quantum gates are essential for manipulating qubits in a quantum circuit, performing operations analogous to classical logic gates but with additional quantum features. Key quantum gates used in this study are:

- **CNOT Gate (Controlled-NOT):** A two-qubit gate that flips the target qubit if the control qubit is in the state $|1\rangle$. This gate functions as a quantum XOR gate.
- **CCNOT Gate (Toffoli):** A three-qubit gate that flips the target qubit only if both control qubits are in the $|1\rangle$ state. This gate functions as a quantum AND gate.

2.2 Quantum Adder Logic

Half-Adder: A quantum half-adder takes two qubits as input and produces a sum and carry output. The sum is computed using the CNOT gate, and the carry is generated using the CCNOT gate.

Full-Adder: A quantum full-adder takes two input qubits and a carry-in qubit. It uses a combination of CNOT and CCNOT gates to compute both the sum and carry-out, allowing it to handle an additional carry-in input.

3 Methods

3.1 Quantum Half-Adder Circuit

The half-adder circuit adds two single qubits (A and B) and outputs a sum and carry qubit. The logic for the half-adder can be summarized in the following truth table:

A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Qiskit Code for Quantum Half-Adder

```python
from qiskit import QuantumCircuit, transpile
from qiskit_aer import QasmSimulator
from qiskit.visualization import plot_histogram

# Create a quantum circuit with 2 qubits and 2 classical
    bits
qc = QuantumCircuit(2, 2)

# Apply an X gate to qubit 0 to set it to 1 (input A = 1,
    input B = 0)
qc.x(0)  # Set A to 1; you can also set qubit 1 to 1 by
    using qc.x(1) to change input B

# Apply CNOT gate with qubit 0 as control and qubit 1 as
    target for SUM
```

```
qc.cx(0, 1)

# Apply CCNOT gate to generate CARRY (acting as an AND gate)
qc.ccx(0, 1, 1)

# Measure both qubits
qc.measure([0, 1], [0, 1])

# Execute the circuit on a simulator
simulator = QasmSimulator()
qc = transpile(qc, simulator)
job = simulator.run(qc)
result = job.result()
counts = result.get_counts(qc)

# Display the circuit and measurement results
qc.draw('text')
plot_histogram(counts)
```

3.2 Quantum Full-Adder Circuit

A full-adder adds two qubits (A and B) and an input carry (Cin). The circuit outputs a sum and a carry-out (Cout). The logic of the full-adder can be described by the following truth table:

A	B	Cin	Sum	Cout
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1

Qiskit Code for Quantum Full-Adder

```
# Create a quantum circuit with 4 qubits (3 inputs and 1
    auxiliary for intermediate results) and 2 classical bits
qc = QuantumCircuit(4, 2)

# Initialize input states: set qubits to represent A = 1, B
    = 1, Cin = 1
qc.x(0)  # A = 1
qc.x(1)  # B = 1
qc.x(2)  # Cin = 1

# Apply CNOT gates for SUM calculation
```

```
qc.cx(0, 3)  # A XOR B -> SUM
qc.cx(1, 3)  # (A XOR B) XOR Cin -> SUM

# Apply CCNOT for CARRY calculation
qc.ccx(0, 1, 3)  # A AND B -> Carry
qc.ccx(3, 2, 3)  # (A AND B) OR Cin AND (A XOR B)

# Measure the output: Sum in qubit 3, Carry in qubit 2
qc.measure(3, 0)  # Sum
qc.measure(2, 1)  # Carry

# Execute the circuit on a simulator
simulator = QasmSimulator()
qc = transpile(qc, simulator)
job = simulator.run(qc)
result = job.result()
counts = result.get_counts(qc)

# Display the circuit and histogram of results
qc.draw('mpl')
plot_histogram(counts)
```

4 Results

Quantum Half-Adder: The half-adder circuit outputs the expected sum and carry for each combination of input qubits. The histogram shows that, for example, when both inputs are $|1\rangle$, the output is Sum $= 0$ and Carry $= 1$.

Quantum Full-Adder: The full-adder circuit outputs the expected sum and carry-out for each combination of input qubits and carry-in. For example, when all inputs are $|1\rangle$, the output is Sum $= 1$ and Carry $= 1$.

5 Conclusion

This manuscript demonstrates the design and simulation of quantum half-adder and full-adder circuits using Qiskit. By applying CNOT and CCNOT gates, we constructed circuits that replicate classical binary addition using qubits. These quantum adder circuits provide a foundation for developing more complex quantum algorithms that require arithmetic operations, such as modular exponentiation in Shor's algorithm. Qiskit offers a robust platform for designing and simulating these circuits, highlighting the potential of quantum computation for arithmetic-based quantum algorithms.

References

- Qiskit Documentation: `https://qiskit.org/documentation/`

- IBM Quantum Experience: `https://quantum-computing.ibm.com/`