Quantum Adder Circuits in Quantum Computing

Introduction

A quantum adder circuit performs the addition of binary numbers on quantum states. The simplest form of quantum addition is the half-adder, which only takes two input bits and produces a sum and a carry output. More complex versions include full adders and ripple-carry adders that can handle multiple bits and can add two n-bit numbers. These circuits are useful in quantum algorithms that involve arithmetic operations, such as quantum phase estimation and Shor's algorithm.

Types of Quantum Adders

- **Half-Adder:** Adds two bits, producing a sum and a carry bit.
- **Full Adder:** Adds three bits (two inputs plus a carry-in bit), producing a sum and a carry-out bit.
- **Ripple-Carry Adder:** Chains multiple full adders to add multi-bit numbers.

In quantum computing, addition is performed using quantum gates such as the CNOT and Toffoli (CCX) gates, which allow conditional flipping of qubits based on the values of other qubits.

Quantum Half-Adder Circuit

A quantum half-adder takes two qubits (representing two binary inputs, A and B) and produces a sum and a carry qubit. The half-adder circuit is relatively simple and can be implemented as follows:

- **Sum:** Achieved using a CNOT gate, which performs the XOR operation.
- **Carry:** Achieved using a Toffoli (CCX) gate, which performs the AND operation.

Circuit for Half-Adder

- **CNOT gate:** Applied between the input qubits, flipping one of the qubits conditionally, producing the XOR output (sum).
- **Toffoli gate:** Applied with the two input qubits as controls and an additional qubit as the target to store the carry output.

```
from qiskit import QuantumCircuit
qc = QuantumCircuit(3, 2)  # 3 qubits (2 inputs, 1 carry)
    and 2 classical bits for measurement

# Apply CNOT for sum (XOR)
qc.cx(0, 1)

# Apply Toffoli for carry (AND)
qc.ccx(0, 1, 2)

# Measure the sum and carry
qc.measure(1, 0)  # Sum
qc.measure(2, 1)  # Carry
qc.draw('mpl')
```

Quantum Full Adder Circuit

A full adder extends the half-adder by taking a third input, the carry-in bit, and producing two outputs: sum and carry-out. The full adder uses two half-adders and an OR operation for the carry.

Steps for Full Adder

- **First Half-Adder:**
  - Inputs: A and B.
  - Outputs: Intermediate sum S_1 and carry C_1.
- **Second Half-Adder:**
  - Inputs: Intermediate sum S_1 and carry-in bit C_{in}.
  - Outputs: Final sum S and intermediate carry C_2.
- **Final Carry-Out:** OR operation between C_1 and C_2, implemented with additional gates.

```python
from qiskit import QuantumCircuit
qc = QuantumCircuit(4, 2)  # 4 qubits (3 inputs, 1 carry-out) and 2 classical bits for measurement

# First half-adder (A and B)
qc.cx(0, 2)      # Sum S1 (A XOR B)
qc.ccx(0, 1, 3) # Carry C1 (A AND B)

# Second half-adder (S1 and Cin)
qc.cx(1, 2)      # Sum S = S1 XOR Cin
qc.ccx(1, 2, 3) # Carry C2 (S1 AND Cin)

# Measure final sum and carry-out
qc.measure(2, 0)  # Sum
qc.measure(3, 1)  # Carry-out
qc.draw('mpl')
```

Quantum Ripple-Carry Adder

The ripple-carry adder extends the full adder to multiple bits. It consists of cascading multiple full adders so that the carry-out of one stage becomes the carry-in of the next. Each full adder in the chain is controlled by its corresponding bit positions in the two input numbers and the carry from the previous stage.

For example, to add two 3-bit numbers $A = a_2a_1a_0$ and $B = b_2b_1b_0$:

- Start by adding the least significant bits a_0 and b_0 with no initial carry.
- Pass the carry from the addition of a_0 and b_0 to the addition of a_1 and b_1.
- Continue this process until the most significant bits are added, producing the final carry-out.

Circuit for Ripple-Carry Adder

Here's a generalized approach to construct a ripple-carry adder:

```python
def ripple_carry_adder(n):
    qc = QuantumCircuit(2 * n + 1, n + 1)  # (n qubits for A, n for B, 1 for carry-out)
    for i in range(n):
        qc.cx(i, n + i)        # Sum (A XOR B)
        qc.ccx(i, n + i, 2 * n)  # Carry (A AND B)
    return qc

# Example: Ripple-Carry Adder for 3-bit numbers
qc = ripple_carry_adder(3)
qc.draw('mpl')
```

Summary

- **Half-Adder:** Adds two qubits, producing a sum and carry.
- **Full Adder:** Extends the half-adder by adding a carry-in bit, producing a sum and carry-out.
- **Ripple-Carry Adder:** Chains multiple full adders to add multi-bit numbers.

These quantum adder circuits form the foundation of more complex arithmetic circuits in quantum algorithms. They leverage entanglement and superposition, allowing quantum computers to perform arithmetic operations that can be integrated into larger algorithms like Shor's for factoring large numbers.