Практическое задание 1

Задача: По алгоритму решения задачи Дойча построить квантовый гейт U для сбалансированных функций (f(0)=0, f(1)=1 и f(0)=1, f(1)=0). Продемонстрировать результаты работы квантовой схемы, используя Maple или другое ПО. Оформить в виде документа с формулами и схемой.

Решение

Ссылка на GitHub: [nAglTI/QuantumComputations: Python scripts for Quantum Computing Assignments. (github.com)](https://github.com/nAglTI/QuantumComputations)

Алгоритм:

* Инициализация схемы (2 кубита, 1 бит)
* Применение оператора Адамара к первому кубиту (индекс 0)
* Применение NOT к двум кубитам
* Применение U-оператора
* Применение NOT к двум кубитам
* Применение оператора Адамара к первому кубиту (индекс 0)
* Измерение первого кубита и запись результата в бит
* Запуск схемы
* Получение результата, запись в переменную и вывод в консоль
* Визуализация схемы

Код решения:

def task1():  
 # Инициализация схемы (2 кубита, 1 бит)  
 qc = QuantumCircuit(2, 1)  
 # Применение оператора Адамара к первому кубиту (индекс 0)  
 qc.h(0)  
 # Применение NOT к двум кубитам  
 qc.cx(0, 1)  
 # Применение U-оператора  
 qc.unitary([[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 0, 1], [0, 0, 1, 0]], [0, 1], label='U')  
 # Применение NOT к двум кубитам  
 qc.cx(0, 1)  
 # Применение оператора Адамара к первому кубиту (индекс 0)  
 qc.h(0)  
 # Измерение первого кубита и запись результата в бит  
 qc.measure(0, 0)  
 # Запуск схемы  
 simulator = Aer.get\_backend('qasm\_simulator')  
 # Получение результата, запись в переменную и вывод в консоль  
 result = execute(qc, simulator, shots=1024).result()  
 counts = result.get\_counts(qc)  
 print("Task 1 result: " + counts)  
 # Визуализация схемы  
 qc.draw(output='mpl', filename='task1.png')

Результат и итоговая схема:



