QC Architecture and Electronics (CESE4080) Homework 1

Tongyun Yang (5651794, tongyunyang@tudelft.nl)

2023.02.27

1 Exercise 1 Bell Pair

1.1 Question

What is the final state (before measurement)?

Answer The final state before measurement is calculated as follows:

$H(|0\rangle) = |+\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$

$CNOT(\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|01\rangle) = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$

Hence, the final state is $\frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$. Indicating 50% chance would be $|00\rangle$ and 50% chance would be $|11\rangle$.

1.2 Question

What is the state of the system after measuring q_0? Why?

Answer In the run that I simulated, the final state is $|11\rangle$, however, it might be $|00\rangle$ as well. The reason is due to the collapse postulate in [1]. It states that the act of measurement on a quantum system would causes the wave function of the system to collapse to one of the eigenstates of the observable being measured. In the case that I measured, the system collapsed to the state of $|11\rangle$.

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]

.entangle
    H q[0]
    CNOT q[0], q[1]

.measurement
    measure q[0]
    display
```

2 Exercise 2 Teleportation

2.1 Question

Repeat the experiment by changing the initialisation to an arbitrary rotation about the Y-axis by $\frac{\pi}{2}$. Display the state received by Bob after teleportation and decoding.

Answer The state received by Bob with a change on the initialisation to an arbitrary rotation about the Y-axis by $\frac{\pi}{2}$ is shown in figure 1. It shows 50% chance Bob receives $|0\rangle$ and the other 50% chance it receives $|1\rangle$. However, the possibilities of the final states are not limited to $|000\rangle$ and $|100\rangle$. As shown in the equations below, there are multiple possibilities:

$\frac{1}{2\sqrt{2}}(|000\rangle + |001\rangle + |110\rangle + |111\rangle + |010\rangle - |011\rangle + |100\rangle - |101\rangle)$

The expression illustrates the final state before applying measurement on q[0] and q[1]. Hence, whatever q[0] and q[1] measures, the binary controlled single-qubit X gate and Z gate ensures that the final state received by Bob is expressed by: $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$.

```
Version 1.0

qubits 3

.prepare
    prep_z q[0:2]

.init
    Ry q[0], pi/2

.epr
```

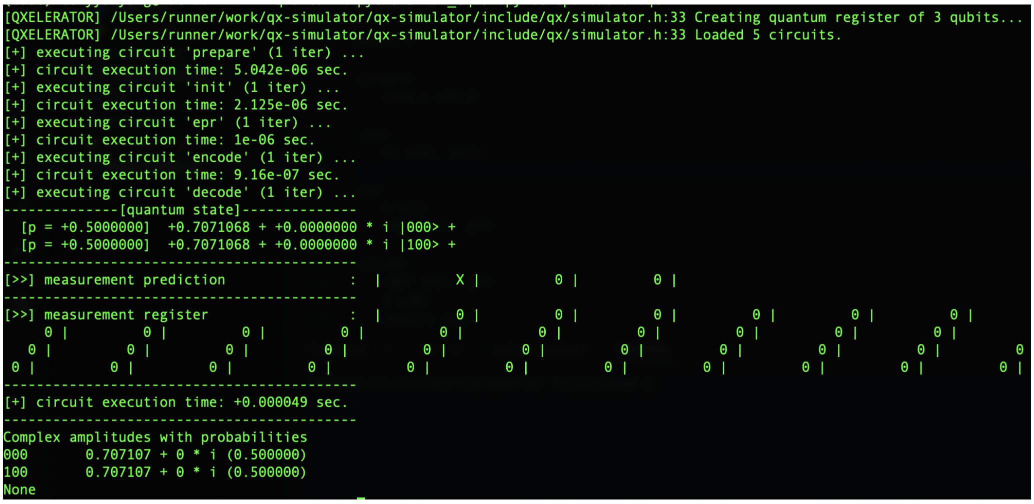


Figure 1: State received by Bob with adjustment on the initialisation

```
    H q[1]
    CNOT q[1], q[2]

.encode
    CNOT q[0], q[1]
    H q[0]
    measure q[0]
    measure q[1]

.decode
    c-x b[1], q[2]
    c-z b[0], q[2]

display
```

2.2 Question

Bob measures the qubit. What is the value he gets? What would happen if Alice initialised to an arbitrary rotation about the X-axis by $\frac{\pi}{2}$ instead?

Answer If Bob measures the state, the system collapse into one of the eigenstates. In this case, Bob measures either $|1\rangle$ or $|0\rangle$.

```
Version 1.0

qubits 3

.prepare
    prep_z q[0:2]

.init
    Ry q[0], pi/2

.epr
    H q[1]
    CNOT q[1], q[2]

.encode
    CNOT q[0], q[1]
    H q[0]
    measure q[0]
    measure q[1]

.decode
    c-x b[1], q[2]
    c-z b[0], q[2]

.measurement
    measure q[2]

display
```

If Alice initialised to an arbitrary rotation about the X-axis by $\frac{\pi}{2}$. Bob measures either $|1\rangle$ or $|0\rangle$, each with 50% chance, however, the amplitude of $|1\rangle$ is $-\frac{1}{\sqrt{2}}i$. The result is shown in figure 2.

```
Version 1.0

qubits 3

.prepare
    prep_z q[0:2]

.init
    Rx q[0], pi/2

```

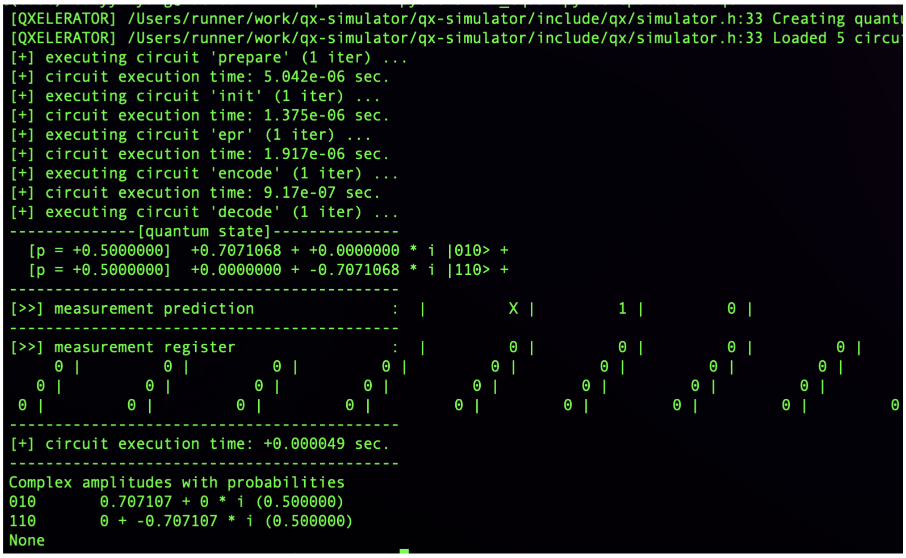


Figure 2: State received by Bob with another adjustment on the initialisation

```
.epr
    H q[1]
    CNOT q[1], q[2]

.encode
    CNOT q[0], q[1]
    H q[0]
    measure q[0]
    measure q[1]

.decode
    c-x b[1], q[2]
    c-z b[0], q[2]

.measurement
    measure q[2]

display
```

2.3 Question

What do we need to do to distinguish between these two cases?

Answer In order to distinguish between these two cases, another rotation around the x-axis should be applied, and the mathematical explanation is followed:

Let's call the final state of the previous situation Pre, and the final state of the current situation Cur.

Previous situation: Alice initialised to an arbitrary rotation about the Y-axis by $\frac{\pi}{2}$.

Current situation: Alice initialised to an arbitrary rotation about the X-axis by $\frac{\pi}{2}$.

The state of Pre could be written as $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$, and the state of Cur could be written as $\begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}}i \end{bmatrix}$. In this case, we apply a $R_x(\frac{\pi}{2})$ which could be written as $\begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}}i \\ -\frac{1}{\sqrt{2}}i & \frac{1}{\sqrt{2}} \end{bmatrix}$. Applying the rotation to both Pre and Cur state, they end up being $Pre = \begin{bmatrix} \frac{1}{2} - \frac{1}{2}i \\ -\frac{1}{2}i + \frac{1}{2} \end{bmatrix}$, $Cur = \begin{bmatrix} 0 \\ -i \end{bmatrix}$, meaning that for the current Cur state, the state of the qubit is always $|1\rangle$. If repeating the experiment multiple times is allowed, then the qubit that always has a fixed $|1\rangle$ final state is the current situation. The qubit has different final states is the previous situation.

3 Exercise 3 Quantum Plain Adder

3.1 The sum circuit

Use CNOT gates to implement the Sum block which sums two bits A0 and B0 (XOR addition) and stores the result in a third bit S0. Write the circuit and save it in a file named sum.qc. The sum circuit should include two sub-circuits: .init and .sum. We note that the .init circuit initializes A0 to

$|0\rangle$ and B0 to $|1\rangle$, the .sum circuit performs the operation and stores the result in S0.

Answer The implementation is as shown below in the code snippet:

```
Version 1.0

qubits 3

.prepare
    prep_z q[0:2]

.init
    X q[1]

.sum
    CNOT q[0], q[2]
    CNOT q[1], q[2]

.measurement
    measure q[2]

display
```

q[0] refers to A0, q[1] refers to B0, and q[2] refers to S0. Please note that the other two sub-circuits, .prepare and .measurement is added in order to enable myself to form a good habit in programming, it shows clearler the purpose of each stage.

3.2 The carry circuit

3.2.1 A. Question Use CNOT and Toffoli gates to implement the carry block. Write the circuit in a file named carry.qc. The circuit is composed of two sub-circuits .init and .carry, each sub-circuit ends with a display command. We note that the initialization circuit initializes the qubits C0, A0, B0 and C1 to the respective values $|0\rangle$, $|1\rangle$, $|1\rangle$ and $|0\rangle$.

Answer The implementation is as shown below in the code snippet:

```
Version 1.0

qubits 4

.prepare
    prep_z q[0:3]

.init
    {X q[1] | X q[2]}

.carry
    Toffoli q[1], q[2], q[3]
    CNOT q[1], q[2]
    Toffoli q[0], q[2], q[3]

.measurement
    measure q[3]

display
```

3.2.2 B. Question Copy the previous circuit into a new file named rcarry.qc. Modify the new circuit as following: Change the .init sub-circuit to initialize the qubits C0, A0, B0, C1 to the states $|0\rangle$, $|1\rangle$, $|0\rangle$ and $|1\rangle$. Rename the sub-circuit .carry to .rcarry and reverse the order of the gates of the .rcarry gates.

Answer The implementation is as shown in the code snippet below:

```
Version 1.0

qubits 4

.prepare
    prep_z q[0:3]

.init
    {X q[1] | X q[3]}

.rcarry
    Toffoli q[0], q[2], q[3]
    CNOT q[1], q[2]
    Toffoli q[1], q[2], q[3]

display
```

3.2.3 C. Question Execute the circuit rcarry.qc and observe its output state. What does the rcarry.qc circuit implement?

Answer The result is as shown in figure 3. It is the reverse of the previous carry circuit. In this case, given the result (output) of the previous carry circuit, the value of each parameter (input) could be retraced.

Figure 3: Final state after applying the opposite order of gates in problem 3.B

3.3 Plain 2-bits Adder Circuit

Now we will implement a 2-qubit adder.

Answer The implementation is shown in the code snippet below:

```
Version 1.0

qubits 7

.prepare
    prep_z q[0:6]

.init
    {X q[2] | X q[4]}

display

.carry_1
    Toffoli q[1], q[2], q[3]
    CNOT q[1], q[2]
    Toffoli q[0], q[2], q[3]

.carry_2
    Toffoli q[4], q[5], q[6]
    CNOT q[4], q[5]
    Toffoli q[3], q[5], q[6]

.sum_1
    CNOT q[4], q[5]

    CNOT q[3], q[5]
    CNOT q[4], q[5]

.sum_2
    Toffoli q[0], q[2], q[3]
    CNOT q[1], q[2]
    Toffoli q[1], q[2], q[3]

    CNOT q[0], q[2]
    CNOT q[1], q[2]

display
```

The result generated in the terminal is as shown in figure 4. In the result, it is confirmed that the implemented is correct, because the result of the addition is on q[5] and q[2], in our case, it is the second and the fifth bit (from left to right, both are 1). $0b10 + 0b01 = 0b11$, hence the result is correct.

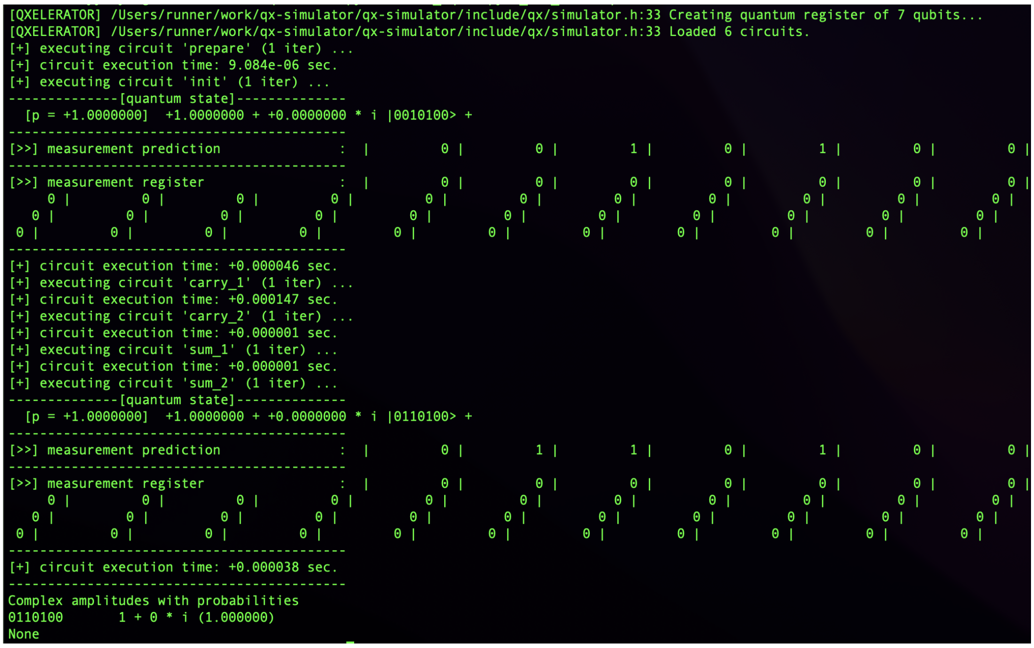


Figure 4: Outcome from the simulator of the plain 2-bit adder circuit

4 Exercise 4

4.1 The U_f Functions

4.1.1 A. Question
Write four circuits which implement the four U_f functions.

Answer The implementation of the four functions are shown below respectively, each occupies a code snippet. The reason of the designs below are the generated truth table, they will also be put below.

1. Uf1

Truth table

$\|y\rangle$	$\|x\rangle$	$\|y \oplus f(x)\rangle$	$\|x\rangle$
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	1

Hence, use CNOT gate.

Code

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]
    map q[0], qx
    map q[1], qy

.init
    X qy

display

.uf
    CNOT qx, qy

display
```

2. Uf2

Truth table

$\|y\rangle$	$\|x\rangle$	$\|y \oplus f(x)\rangle$	$\|x\rangle$
0	0	1	0
0	1	0	1
1	0	0	0
1	1	1	1

Hence, use CNOT and X gate.

Code

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]
    map q[0], qx
    map q[1], qy

.init
    X qy

display

.uf
    CNOT qx, qy
    X qy

display
```

3. Uf3

Truth table

$\|y\rangle$	$\|x\rangle$	$\|y \oplus f(x)\rangle$	$\|x\rangle$
0	0	0	0
0	1	0	1
1	0	1	0
1	1	1	1

Hence, use I gate.

Code

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]
    map q[0], qx
    map q[1], qy

.init
    X qy

display

.uf
    I qx
    I qy

display
```

4. Uf4

Truth table

$\|y\rangle$	$\|x\rangle$	$\|y \oplus f(x)\rangle$	$\|x\rangle$
0	0	1	0
0	1	1	1
1	0	0	0
1	1	0	1

Hence, use X gate.

Code

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]
    map q[0], qx
    map q[1], qy

.init
    X qy

display

.uf
    X qy

display
```

4.1.2 B. Question

4.1.2 B. Question Simulate the execution of the circuits using the QX simulator to check if your circuit implements the function correctly.

Answer It has been proven that the implementations are correct. Every single combination has been tested, and the results are the same as they are displayed in the truth tables.

4.2 The Deutsch's Functions

4.2.1 A. Question Write four circuits which implement the four U_f functions.

Answer The implementation of the four functions are shown below respectively, each occupies a code snippet.

1. **Uf1**

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]
    map q[0], qx
    map q[1], qy

.init
    X qy

display

.superposition
    {H qx |H qy}

display

.uf
    CNOT qx, qy

display

.measurement
    H qx
    measure qx

display
```

2. **Uf2**

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]
    map q[0], qx
    map q[1], qy

.init
    X qy

display

.superposition
    {H qx |H qy}

display

.uf
    CNOT qx, qy
    X qy

display

.measurement
    H qx
    measure qx

display
```

3. **Uf3**

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]
    map q[0], qx
    map q[1], qy

.init
    X qy

display

.superposition
    {H qx |H qy}

display

.uf
    I qx
    I qy

display

.measurement
    H qx
    measure qx

display
```

4. Uf4

```
Version 1.0

qubits 2

.prepare
    prep_z q[0:1]
    map q[0], qx
    map q[1], qy

.init
    X qy

display

.superposition
    {H qx |H qy}

display

.uf
    X qy

display

.measurement
    H qx
    measure qx

display
```

4.2.2 B. Question Simulate the execution of the circuits using the QX simulator to check if your circuit implements the function correctly.

Answer The four results of the simulation is as shown below in the four figures, figure 5, figure 6, figure 7 and figure 8.

From the results, it is shown clearly that when deutsch_uf1 or deutsch_uf2 (namely balanced functions) are applied, the qubit measurement is always 1. When deutsch_uf3 or deutsch_uf3 (namely constant functions) are applied, the qubit measurement is always 0. This proves that the implementations are correct.

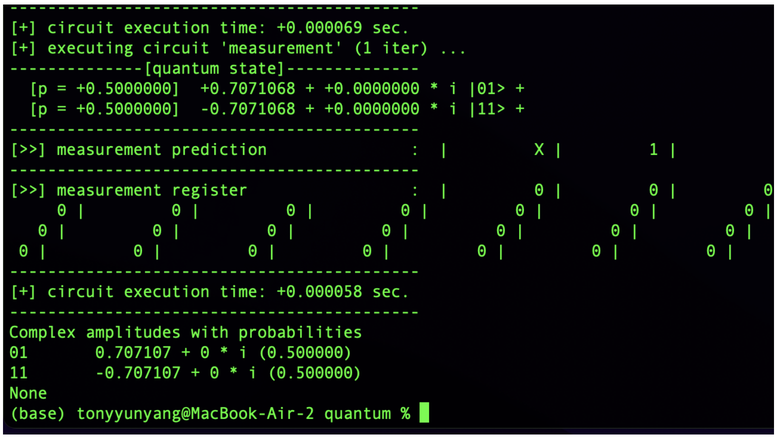

```
------------------------------------------
[+] circuit execution time: +0.000069 sec.
[+] executing circuit 'measurement' (1 iter) ...
--------------[quantum state]--------------
  [p = +0.5000000]  +0.7071068 + +0.0000000 * i |01> +
  [p = +0.5000000]  -0.7071068 + +0.0000000 * i |11> +
------------------------------------------
[>>] measurement prediction             :  |         X |         1 |
------------------------------------------
[>>] measurement register               :  |         0 |         0 |         0
     0 |         0 |         0 |         0 |         0 |         0 |         0 |
   0 |         0 |         0 |         0 |         0 |         0 |         0 |
 0 |         0 |         0 |         0 |         0 |         0 |         0 |
------------------------------------------
[+] circuit execution time: +0.000058 sec.
------------------------------------------
Complex amplitudes with probabilities
01       0.707107 + 0 * i (0.500000)
11       -0.707107 + 0 * i (0.500000)
None
(base) tonyyunyang@MacBook-Air-2 quantum %
```

Figure 5: Simulation result of deutsch_uf1

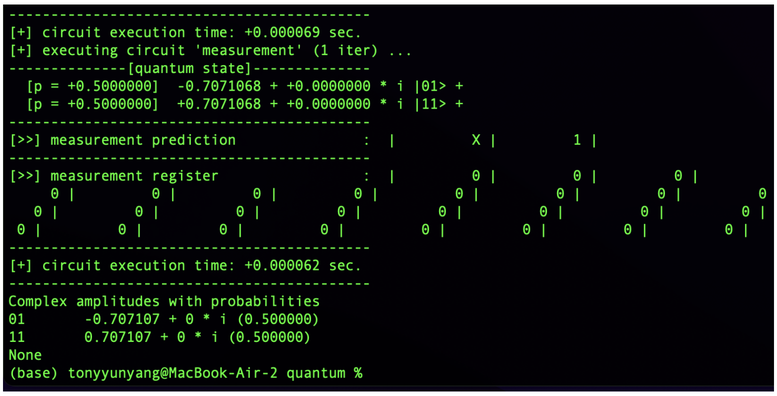

```
------------------------------------------
[+] circuit execution time: +0.000069 sec.
[+] executing circuit 'measurement' (1 iter) ...
--------------[quantum state]--------------
  [p = +0.5000000]  -0.7071068 + +0.0000000 * i |01> +
  [p = +0.5000000]  +0.7071068 + +0.0000000 * i |11> +
------------------------------------------
[>>] measurement prediction             :  |         X |         1 |
------------------------------------------
[>>] measurement register               :  |         0 |         0 |         0 |
     0 |         0 |         0 |         0 |         0 |         0 |         0 |         0
   0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |
 0 |         0 |         0 |         0 |         0 |         0 |         0 |         0 |
------------------------------------------
[+] circuit execution time: +0.000062 sec.
------------------------------------------
Complex amplitudes with probabilities
01       -0.707107 + 0 * i (0.500000)
11       0.707107 + 0 * i (0.500000)
None
(base) tonyyunyang@MacBook-Air-2 quantum %
```

Figure 6: Simulation result of deutsch_uf2

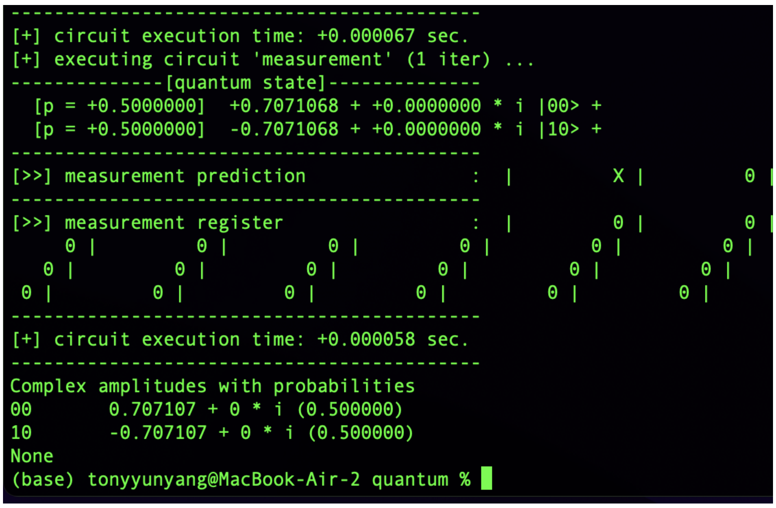
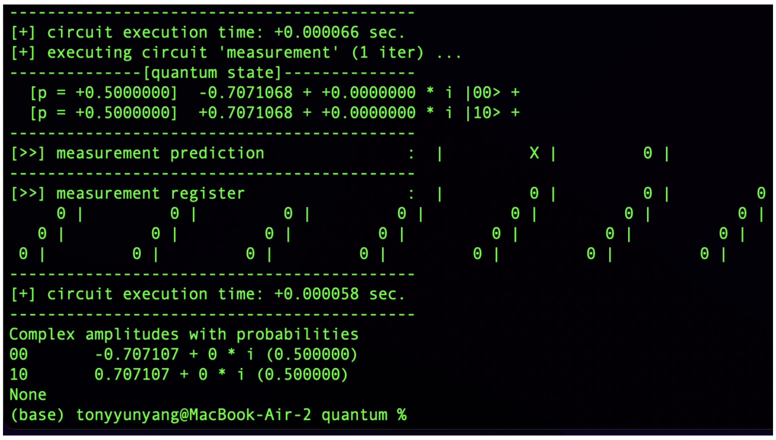

```
------------------------------------------
[+] circuit execution time: +0.000067 sec.
[+] executing circuit 'measurement' (1 iter) ...
--------------[quantum state]--------------
  [p = +0.5000000]  +0.7071068 + +0.0000000 * i |00> +
  [p = +0.5000000]  -0.7071068 + +0.0000000 * i |10> +
------------------------------------------
[>>] measurement prediction             :  |         X |         0
------------------------------------------
[>>] measurement register               :  |         0 |         0
     0 |         0 |         0 |         0 |         0 |         0 |
   0 |         0 |         0 |         0 |         0 |         0 |
 0 |         0 |         0 |         0 |         0 |         0 |
------------------------------------------
[+] circuit execution time: +0.000058 sec.
------------------------------------------
Complex amplitudes with probabilities
00       0.707107 + 0 * i (0.500000)
10       -0.707107 + 0 * i (0.500000)
None
(base) tonyyunyang@MacBook-Air-2 quantum %
```

Figure 7: Simulation result of deutsch_uf3



```
-------------------------------------------
[+] circuit execution time: +0.000066 sec.
[+] executing circuit 'measurement' (1 iter) ...
--------------[quantum state]--------------
  [p = +0.5000000]  -0.7071068 + +0.0000000 * i |00> +
  [p = +0.5000000]  +0.7071068 + +0.0000000 * i |10> +
-------------------------------------------
[>>] measurement prediction              :  |         X |         0 |
-------------------------------------------
[>>] measurement register                :  |         0 |         0 |         0
     0 |         0 |         0 |         0 |         0 |         0 |         0 |
   0 |         0 |         0 |         0 |         0 |         0 |         0 |
 0 |         0 |         0 |         0 |         0 |         0 |         0 |
-------------------------------------------
[+] circuit execution time: +0.000058 sec.
-------------------------------------------
Complex amplitudes with probabilities
00       -0.707107 + 0 * i (0.500000)
10       0.707107 + 0 * i (0.500000)
None
(base) tonyyunyang@MacBook-Air-2 quantum %
```

Figure 8: Simulation result of deutsch_uf4

References

[1] "Wave function collapse," (Date last accessed 23-02-2023). [Online]. Available: https://en.wikipedia.org/wiki/Wave_function_collapse