1 组合门性质与验证

1.1 Z_j 矩阵乘可交换性

由题目猜测可得：

$$Z_0 = I \otimes I \otimes Z$$
$$Z_1 = I \otimes Z \otimes I$$
$$Z_2 = Z \otimes I \otimes I$$

首先可知，若：

$$Z_0 Z_1 = Z_1 Z_0$$

则这两矩阵乘可交换，于是编写如下 matlab 代码，验证三个 Z 阵，两两乘可交换：

```
%% 验证合并验证算子的乘可交换性
Z=[1 0;0 -1];
Z0=kron(kron(eye(2),eye(2)),Z);
Z1=kron(kron(eye(2),Z),eye(2));
Z2=kron(kron(Z,eye(2)),eye(2));

Z0Z1=Z0*Z1;
Z1Z2=Z1*Z2;
Z0Z2=Z0*Z2;

Z0Z1 == Z1*Z0
```

```
ans = 8×8 logical array
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
```

```
Z0Z2 == Z2*Z0
```

```
ans = 8×8 logical array
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
```

```
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
```

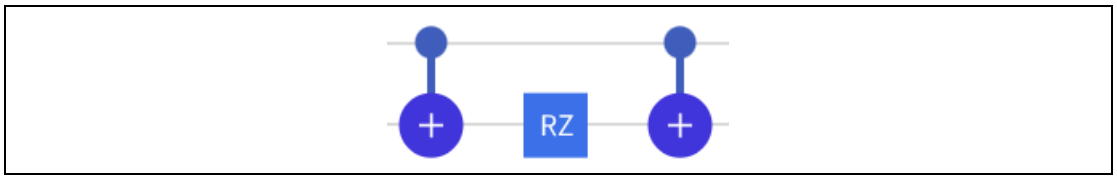
```
Z1Z2 == Z2*Z1
```

```
ans = 8×8 logical array
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
```

于是得证它们的乘可交换性，并猜测$Z_j = I_{n-1} \otimes ... I_{j+1} \otimes Z \otimes I_{j-1} \otimes I_0$也具有乘可交换性。

1.2double-qubits 单元线路的位可交换性

在提示中，我们有如下线路：



我们猜想，如下线路的等效算子和上面的相同：

于是编写 matlab 代码验证可得：

```
%% 验证 2qubits 单元线路的位可交换性
Z0_2=kron(eye(2),Z);
Z1_2=kron(Z,eye(2));
CNOT1=[eye(2) zeros(2);zeros(2) flip(eye(2),2)];% q1 为控制位
CNOT0=[1 0 0 0;
       0 0 0 1;
       0 0 1 0;
       0 1 0 0];% q0 为控制位

unit=CNOT0*expm(-1j*pi/8*Z1_2)*CNOT0;
unit==CNOT1*expm(-1j*pi/8*Z0_2)*CNOT1
```

```
ans = 4×4 logical array
   1   1   1   1
   1   1   1   1
```

1 1 1 1
1 1 1 1

于是可得这两矩阵相等，也就是我们可以有两种单元线路，用它们来构建算子$u_{j,k}=e^{-i\frac{\pi}{8}Z_jZ_k}$。

1.3单元线路之间的可交换性

现在我们已知

$$u_{j,k}=u_{k,j}$$

不禁猜测，是否有

$$u_{j,k}u_{m,n}=u_{m,n}u_{j,k}$$

也就是单元线路之间是否是可交换次序的，于是我们依然使用 matlab 编程，可得：

```
e0=expm(-1j*pi/8*Z0Z1);
e1=expm(-1j*pi/8*Z1Z2);
e2=expm(-1j*pi/8*Z0Z2);
e0e1=e0*e1;
e0e2=e0*e2;
e1e2=e1*e2;
e0e1==e1*e0
```

```
ans = 8×8 logical array
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
```

```
e0e2==e2*e0
```

```
ans = 8×8 logical array
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
```

```
e1e2==e1*e2
```

```
ans = 8×8 logical array
```

```
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
1   1   1   1   1   1   1   1
```

也就是说，单位线路的添加顺序也可交换。

由于我们已知

$$CNOT_j^i \ is\ a\ CNOT\ gate\ where\ q[i]\ controls\ q[j]$$

$$then,\ we\ have\ e^{-i\frac{\theta}{2}Z_iZ_j} = CNOT_j^i RZ(q[j],\theta)CNOT_j^i$$

如果我们假设有一个式子为：

$$CNOT_j^i RZ(q[j],\theta)CNOT_j^i CNOT_p^j RZ(q[p],\theta)CNOT_p^j$$

那么我们不禁猜测是否有办法处理中间间隔的$CNOT$门，一种想法是交换之，而另一种想法则是约并之，我们不妨来看看三个 qubit 情形下的这样两个门：

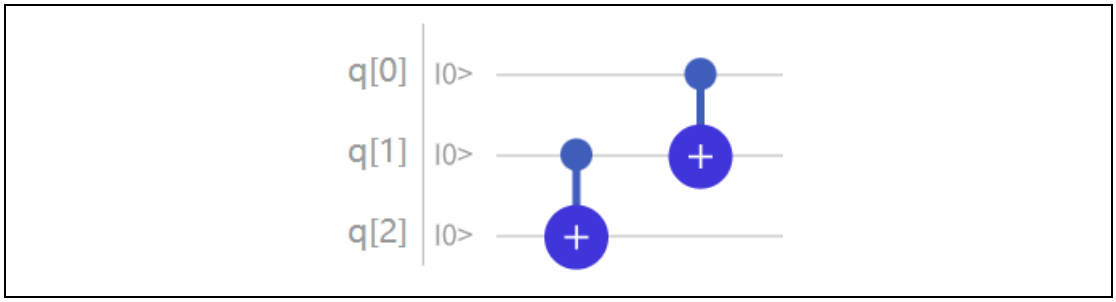
$$CNOT_1^0 = I \otimes [CNOT_1^0]$$

$$CNOT_2^1 = [CNOT_2^1] \otimes I$$

其中方括号内的表示在二维希尔伯特空间下的经典受控非门，现在我们对它作积：

$$(I \otimes [CNOT_1^0])\,([CNOT_2^1] \otimes I)$$

线路图为：



如果约化，显然它的结果会是一个三维算子，在 matlab 里尝试这件事情可得

```
kron(CNOT0,eye(2))*kron(eye(2),CNOT0)
```

```
ans = 8×8
 1     0     0     0     0     0     0     0
 0     0     0     1     0     0     0     0
 0     0     0     0     0     0     1     0
 0     0     0     0     0     1     0     0
```

```
    0     0     0     0     1     0     0     0
    0     0     0     0     0     0     0     1
    0     0     1     0     0     0     0     0
    0     1     0     0     0     0     0     0
```

这显然不利于我们解决问题，于是考虑第一种思路，即交换性，这种思路下我们想把旋转门换到一起，我们可以让第三位量子比特与第零位经过同样的制备，这样可以实现间邻的受控非门，下面是尝试过程：

```
CNOT12=kron(CNOT0,eye(2));
CNOT01=kron(eye(2),CNOT0);
CNOT21=kron(CNOT1,eye(2));
CNOT10=kron(eye(2),CNOT1);
RZ1=expm(-1j*pi/8*Z1);
RZ2=expm(-1j*pi/8*Z2);
Z3_3=kron(Z,kron(eye(2),kron(eye(2),eye(2))));
CNOT12_3=kron(eye(2),CNOT12);
CNOT01_3=kron(eye(2),CNOT01);
CNOT23_3=kron(CNOT0,kron(eye(2),eye(2)));
RZ1_3=kron(eye(2),RZ1);
RZ2_3=kron(eye(2),RZ2);
RZ3_3=expm(-1j*pi/8*Z3_3);


RCCRCCRC=RZ1_3*CNOT01_3*CNOT12_3*RZ2_3*CNOT12_3*CNOT23_3*RZ3_3*CNOT23_3
;
prodR=RZ1_3*RZ2_3*RZ3_3;
K=inv(prodR)*RCCRCCRC
```

```
K = 16×16 complex
Rows 7:16 | Columns 6:15
   0.0000 + 0.0000i   1.0000 - 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i
   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i  -0.0000 - 1.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
-0.0000 - 1.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 - 0.7071i
0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
RC1=RZ1*CNOT01
```

```
RC1 = 8×8 complex
   0.9239 - 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 - 0.3827i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 + 0.3827i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.9239 + 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239
- 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 - 0.3827i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.9239 + 0.3827i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.9239 + 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
RC1==CNOT01*RZ1
```

```
ans = 8×8 logical array
   1   1   1   1   1   1   1   1
   1   1   1   0   1   1   1   1
   1   1   1   1   1   1   1   1
   1   0   1   1   1   1   1   1
   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   0
   1   1   1   1   1   1   1   1
   1   1   1   1   1   0   1   1
```

```
RC1*inv(RZ1)
```

```
ans = 8×8 complex
```

```
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 - 0.7071i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 - 0.7071i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

RC1*inv(RZ1)*inv(CNOT01)

```
ans = 8×8 complex
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i
```

RC1*CNOT12*inv(RZ1)*inv(CNOT01)

```
ans = 8×8 complex
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i
```

```
CR2=CNOT12*RZ2
```

```
CR2 = 8×8 complex
   0.9239 - 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.9239 - 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.9239 + 0.3827i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 + 0.3827i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239
+ 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.9239 + 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 - 0.3827i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 - 0.3827i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
RC2=RZ2*CNOT12
```

```
RC2 = 8×8 complex
   0.9239 - 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.9239 - 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.9239 - 0.3827i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 - 0.3827i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239
+ 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.9239 + 0.3827i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 + 0.3827i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.9239 + 0.3827i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
RC1*inv(RZ1) == RC2*inv(RZ2)
```

```
ans = 8×8 logical array
   1   1   1   1   1   1   1   1
   1   0   1   0   1   1   1   1
   1   1   0   1   1   1   0   1
   1   0   1   1   1   1   1   0
   1   1   1   1   1   1   1   1
   1   1   1   1   1   0   1   0
   1   1   0   1   1   1   0   1
   1   1   1   0   1   0   1   1
```

```
RC1*CNOT12*inv(RC1)
```

```
ans = 8×8
     1     0     0     0     0     0     0     0
     0     0     0     0     0     1     0     0
     0     0     0     0     0     0     1     0
     0     0     0     1     0     0     0     0
     0     0     0     0     1     0     0     0
     0     1     0     0     0     0     0     0
     0     0     1     0     0     0     0     0
     0     0     0     0     0     0     0     1
```

其中，

$$K=\left(\prod_{i=1}^{3}RZ(q[i],\theta)\right)^{-1}CNOT_1^0\left(\prod_{i=1}^{3}CNOT_i^{i-1}RZ(q[i],\theta)CNOT_i^{i-1}\right)$$

这里是因为$CNOT\cdot CNOT=I$，所以省略求逆符号不写，K是一个很有趣的矩阵，它的元素组成是：

$$k(1,1)=1,k(4,2)=1,k(7,7)=1,k(6,8)=1,k(9,9)=1,k(12,10)=1$$

$$k(3,3)=e^{i\frac{\pi}{4}},\ k(2,4)=e^{i\frac{\pi}{4}},k(5,5)=e^{i\frac{\pi}{4}},k(8,6)=e^{i\frac{\pi}{4}},k(11,11)=e^{i\frac{\pi}{4}},k(10,12)=e^{i\frac{\pi}{4}}$$

$$k(13,13)=e^{-i\frac{\pi}{4}},k(16,14)=e^{-i\frac{\pi}{4}}$$

$$k(15,15)=-i,k(14,16)=-i$$

它总在对称位上有元素，并且有如下关系：

$i+j$	k
2	1
6	e+,e+,1
10	e+
14	1,1,e+
18	1
22	e+,1,e+

26	e-
30	-i,-i,e-

即，行列和为奇数次二倍奇数位上总有 1 个元素，反之则有 3 个元素，且分布在紧邻着对角线的位置，这一种特殊的矩阵，如果能将它分解成若干基础阵在这情形下的 kron 积或者矩阵积，或者两者的混合，将会使得 CNOT 门的数量减少至多一半，我们将该矩阵分块可得：

$$\begin{bmatrix} K_1 & & & \\ & K_2 & & \\ & & K_3 & \\ & & & K_4 \end{bmatrix}$$

其中，

$$K_1 = \begin{pmatrix} 1 & & & \\ & 0 & & e^{i\frac{\pi}{4}} \\ & & e^{i\frac{\pi}{4}} & \\ & 1 & & 0 \end{pmatrix}$$

$$K_2 = \begin{pmatrix} e^{i\frac{\pi}{4}} & & & \\ & 0 & & 1 \\ & & 1 & \\ & e^{i\frac{\pi}{4}} & & 0 \end{pmatrix}$$

$$K_3 = \begin{pmatrix} 1 & & & \\ & 0 & & e^{i\frac{\pi}{4}} \\ & & e^{i\frac{\pi}{4}} & \\ & 1 & & 0 \end{pmatrix}$$

$$K_4 = \begin{pmatrix} e^{-i\frac{\pi}{4}} & & & \\ & 0 & & -i \\ & & -i & \\ & e^{-i\frac{\pi}{4}} & & 0 \end{pmatrix}$$

可以看出这些阵具有良好的性质，然而第四个子阵的存在是我们难以将其写进递推里，对前三个，我们有：

$$K_{i+1} = K_i^{\dagger} e^{i\frac{\pi}{4}}$$

而，对第四个：

$$K_4 = K_3^{\dagger} e^{-i\frac{\pi}{4}}$$

但是好在，利用这些规律我们可以将 K 表出

$$K=\begin{pmatrix}1&&&\\&0&&\\&&1&\\&&&0\end{pmatrix}\otimes K_1+\begin{pmatrix}0&&&\\&e^{i\frac{\pi}{4}}&&\\&&0&\\&&&e^{-i\frac{\pi}{4}}\end{pmatrix}\otimes K_1^{\dagger}$$

下面探究K_1和$CNOT$的关系：

```
K1=[1 0 0 0;
    0 0 0 exp(1i*pi/4);
    0 0 exp(1i*pi/4) 0;
    0 1 0 0];
CNOT0*K1
```

```
ans = 4×4 complex
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i
```

```
K1*CNOT0
```

```
ans = 4×4 complex
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000 + 0.0000i
```

```
CNOT1*K1
```

```
ans = 4×4 complex
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i
   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i
```

```
K1*CNOT1
```

```
ans = 4×4 complex
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i
   0.0000 + 0.0000i   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

可得

$$K_1=\begin{pmatrix}1&&&\\&e^{i\frac{\pi}{4}}&&\\&&e^{i\frac{\pi}{4}}&\\&&&1\end{pmatrix}CNOT_1^0=CNOT_1^0\begin{pmatrix}1&&&\\&1&&\\&&e^{i\frac{\pi}{4}}&\\&&&e^{i\frac{\pi}{4}}\end{pmatrix}=CNOT_1^0\left(U_1\left(\frac{\pi}{4}\right)\otimes I\right)$$

同理可得

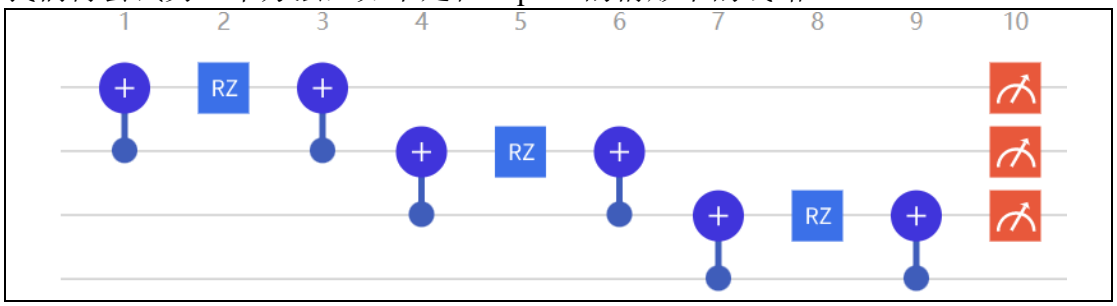
$$K_1^\dagger = \left(U_1\left(-\frac{\pi}{4}\right)\otimes I\right)CNOT_1^0$$

这样，我们可以将式子写成

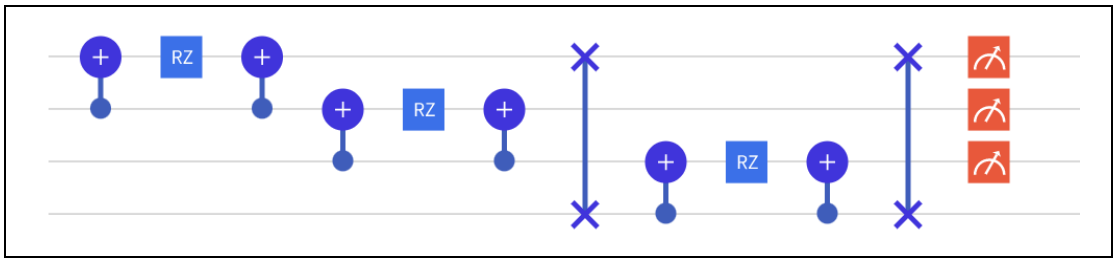
$$K = I\otimes\frac{1}{2}(I+Z)\otimes K_1 + I\otimes\left(T-\frac{1}{2}(I+Z)\right)\otimes K_1^\dagger$$

这样，如果我们可以使用 8 个量子比特，前四个构建第一个对应的量子线路，后四个构建后一项的量子线路，分别经过单元组合门后再叠加起来，理论上是可以只使用 5 个 CNOT 门的。

我们再尝试另一个方法，如下是在 4qubit 的情形下的线路：



值得指出的是，即使这里指定辅助位不能使用同样方法制备，而是要用 SWAP，效果不会改变，只是所有此类线路的门数要增加一个相同值而已。
如下图所示：



但是如果可以使用相同制备方法，那么增加辅助量子比特显然可以帮助我们减少 CNOT 门的数量，这也是为什么前面 K 矩阵的式子里，逆在左边，因为这样该算子的等效量子门可以放在最左边，从而避免放最右边时还要在辅助量子比特里加入新的 CNOT 门这种尴尬的情形。

类似于上面的方法，我们可以来玩一点新花样，如下所示，我们设

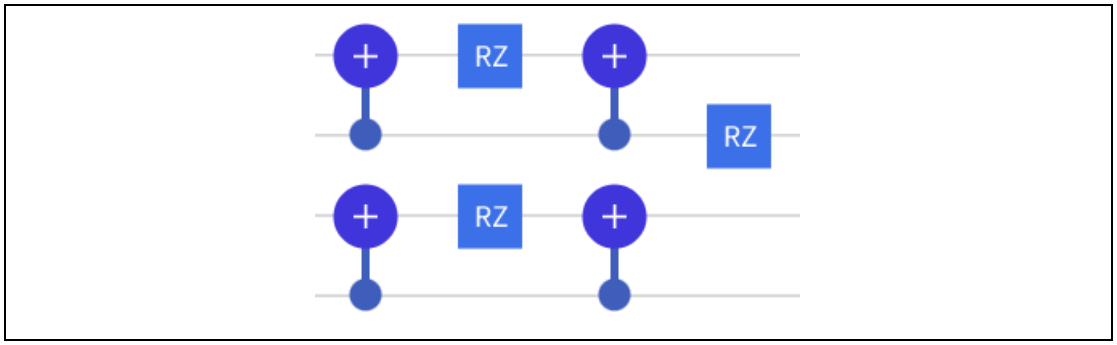
$$U_f = U\left(\frac{\pi}{8}\right)$$

$$u_j^i = CNOT_j^i RZ\left(q[j],\frac{\pi}{8}\right)CNOT_j^i$$

构建线路算子

$$M = RZ\left(q[1],\frac{\pi}{8}\right)(u_2^3\otimes u_0^1)$$

即如下线路



我们设

$$MK_f = U_f$$

编程求解出 $K_f = inv(M)U_f$：

```
RZ1_3;
RZ2_3;
RZ3_3;
Z0_3=kron(eye(2),Z0);
Z1_3=kron(eye(2),Z1);
Z2_3=kron(eye(2),Z2);
Z3_3;
prodZ_3=Z0_3*Z1_3+Z1_3*Z2_3+Z0_3*Z2_3;
Ufinal=expm(-1j*pi/8*prodZ_3);
J=kron(CNOT1,CNOT1)*kron(kron(eye(2),expm(-
1j*pi/8*Z)),kron(eye(2),expm(-1j*pi/8*Z)))*kron(CNOT1,CNOT1);
M1=RZ2_3*J;
M2=J*RZ2_3;
Kf1=(M1)\Ufinal
```

```
Kf1 = 16×16 complex
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 + 1.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
```

```
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 - 1.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 - 0.7071i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
1.0000 - 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
Kf2=M2\Ufinal
```

```
Kf2 = 16×16 complex
   1.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 + 0.7071i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i  -0.0000 + 1.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   1.0000
```

```
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.7071 - 0.7071i   0.0000 + 0.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 - 1.0000i   0.0000 + 0.0000i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.7071 - 0.7071i
0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000
+ 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
1.0000 - 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 +
0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
```

```
Kf2==Kf1
```

```
ans = 16×16 logical array
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

我们使用相同的方法分解K_f得到：

$$K_f = \begin{pmatrix} K_{f1} & & & \\ & K_{f2} & & \\ & & K_{f3} & \\ & & & K_{f4} \end{pmatrix}$$

$$K_{f1} = \mathrm{diag}\left(1, e^{i\frac{\pi}{4}}, e^{i\frac{\pi}{4}}, i\right), K_{f2} = \mathrm{diag}\left(1, e^{-i\frac{\pi}{4}}, e^{-i\frac{\pi}{4}}, -i\right),$$

$$K_{f3} = \mathrm{diag}\left(e^{-i\frac{\pi}{4}}, 1, 1, e^{i\frac{\pi}{4}}\right), K_{f4} = \mathrm{diag}\left(e^{i\frac{\pi}{4}}, 1, 1, e^{-i\frac{\pi}{4}}\right)$$

把$e^{-i\frac{\pi}{4}}, e^{i\frac{\pi}{4}}$简写为$e-, e+$，分解成克劳里克积的形式可得：

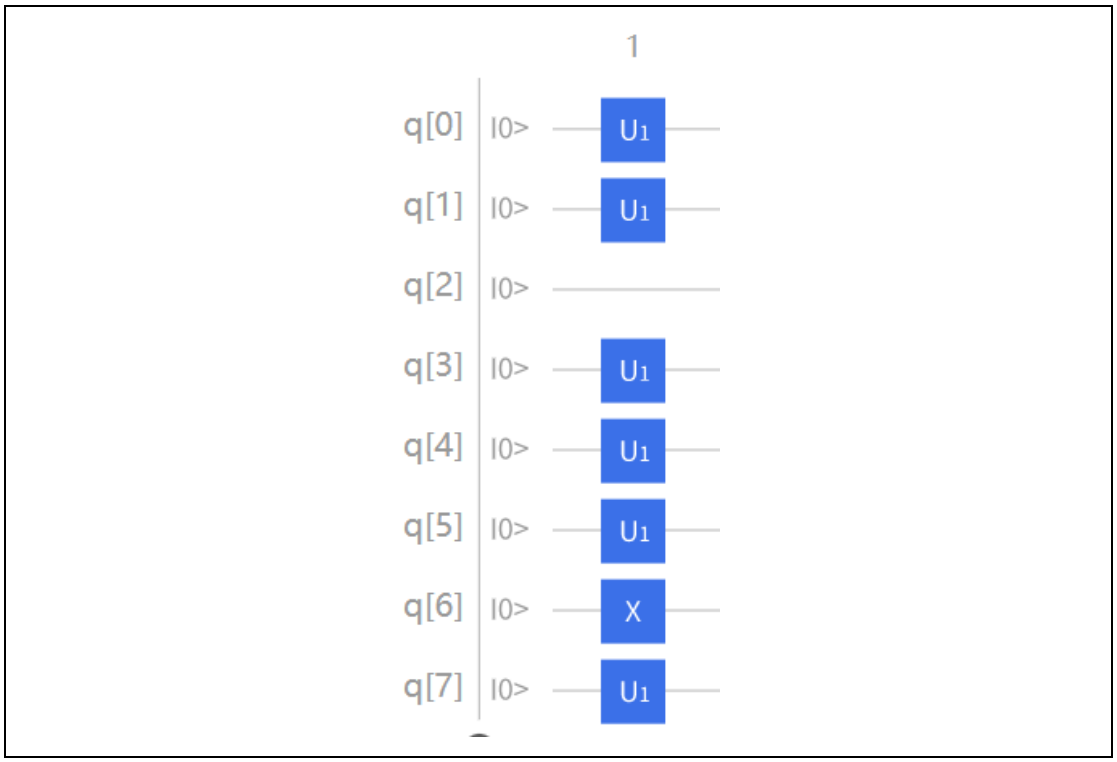
$$K_{f2} = {K_{f1}}^{\dagger}, K_{f3} = (e-)K_{f1}, K_{f4} = (e+){K_{f1}}^{\dagger}$$

$$\therefore\ K_f = \begin{pmatrix} 1 & \\ & e- \end{pmatrix} \otimes \begin{pmatrix} 1 & \\ & 0 \end{pmatrix} \otimes K_{f1} + \begin{pmatrix} 1 & \\ & e+ \end{pmatrix} \otimes \begin{pmatrix} 0 & \\ & 1 \end{pmatrix} \otimes {K_{f1}}^{\dagger}$$

$$K_{f1} = \begin{pmatrix} 1 & \\ & e+ \end{pmatrix} \otimes \begin{pmatrix} 1 & \\ & e+ \end{pmatrix}$$

$$K_{f2} = \begin{pmatrix} 1 & \\ & e- \end{pmatrix} \otimes \begin{pmatrix} 1 & \\ & e- \end{pmatrix}$$

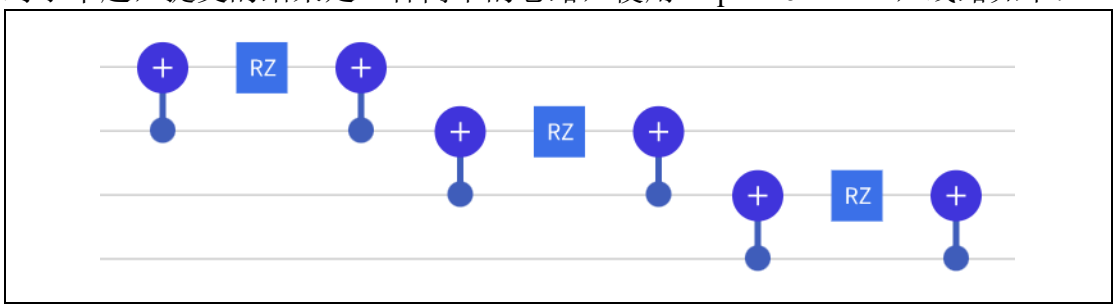
它显然具有十分优美的形式，虽然不符合题意，但是我们把它实现如下



请注意这里我们是如何实现$\begin{pmatrix} 1 & \\ & 0 \end{pmatrix}, \begin{pmatrix} 0 & \\ & 1 \end{pmatrix}$这两个门的，这里因为初始态为$|0\rangle$，所以要忽略 q[2]上的所有制备门操作，同理，对 q[6]进行 X 操作后把它变换为

$|1\rangle$，同时也忽略所有对它的制备门操作，这样就可以得到两簇通过K_f分量的量子系统，再让它们分别通过M，测量分布，并叠加表示，即可得到与原运算相同的结果，也就是在使用4个CNOT的基础上完成了题目要求，其实我们猜测CNOT门的个数可以更少，最少应该2个就足够，只不过没能验证，这种方法主要的困难在于描述产生湮灭门，以及多簇量子如何分别经过同一量子门。

对于本题，提交的结果是一种简单的思路，使用 4 qubit 6 CNOT，线路如下：



需要做一些小的转换，因为最下方协助比特还处在制备态，所以可以从测得的4比特状态中还原出前三比特：

$$\because\ |\varphi_3\rangle\otimes(|\varphi_2\rangle\otimes|\varphi_1\rangle\otimes|\varphi_0\rangle)=\begin{pmatrix}a\\b\end{pmatrix}|\psi\rangle=\begin{pmatrix}a|\psi\rangle\\b|\psi\rangle\end{pmatrix}=|Qstate\rangle$$

$$\therefore\ |\psi\rangle[i]=\frac{|Qstate\rangle[i]+|Qstate\rangle[i+8]}{a+b}$$

由此式编写 python 代码，得到输出：

```
ocuments/Python Scripts/quantum/combinegate.py"
[[(0.23096992868389576-0.23096985320411412j), (0.32664075227285083+0j), (0.23096992868389576+0.23096985320411412j), (0.32664075227285083+0j),
(0.32664075227285083+0j), (0.23096992868389576+0.23096985320411412j), (0.32664075227285083+0j), (0.23096992868389576-0.23096985320411412j)], [
(0.13529910509494067-0.3266407080578112j), (-0.32664075227285083+0.135298998350417j), (-0.3266407522728509-0.135298998350417j), (0.32664075227
285083-0.135298998350417j), (-0.32664075227285083-0.135298998350417j), (0.13529910509494067+0.3266407080578112j), (0.32664075227285083+0.13529
8998350417j), (-0.3266407522728509+0.135298998350417j)], [(0.37722449835131117-0.4714130851775202j), (0.34645483641601194-0.038452386162843688j
), (0.2721705427996167+0.217790576724118288j), (0.2000257930668322-0.022200495502101233j), (0.34645483641601194+0.038452386162843688j), (0.12574
149945043706+0.15713769505917333j), (0.2000257930668322+0.022200495502101233j), (0.09072351426653888-0.07259685890803948j)]]
```