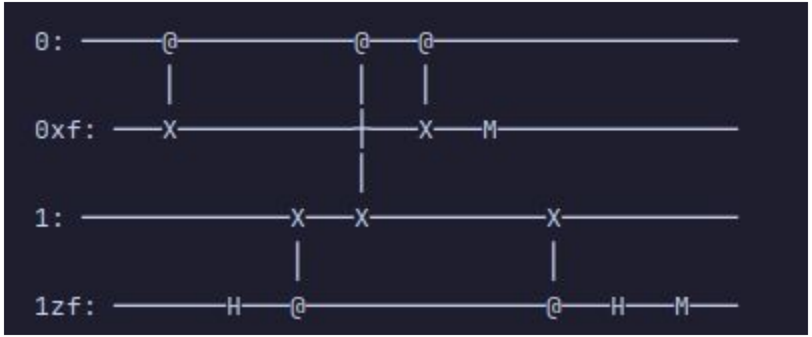
Flag Qubit Compiler

Quan Hoang

What is Flag circuit ?:

Flag circuits use extra ancilla qubits to signal when errors resulting from v faults in the circuit have weight greater than v.

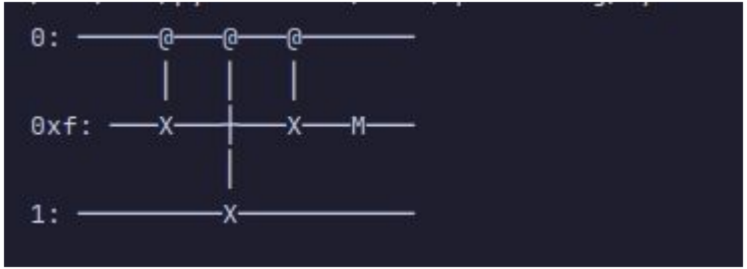
Example:



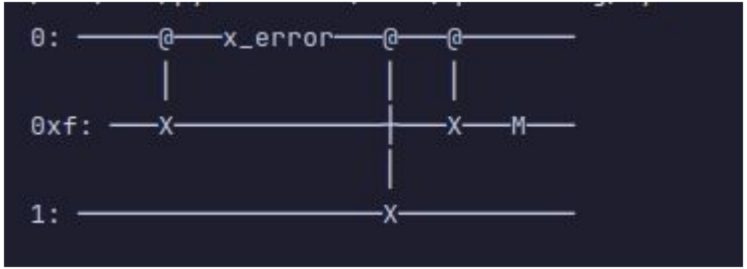
```
0: ─────@───────────@───@───────────────────
        │           │   │
0xf: ───X───────────┼───X───M───────────────
                    │
1: ─────────────X───X───────────X───────────
                │               │
1zf: ───────H───@───────────────@───H───M───
```

How does flag Circuit work ?

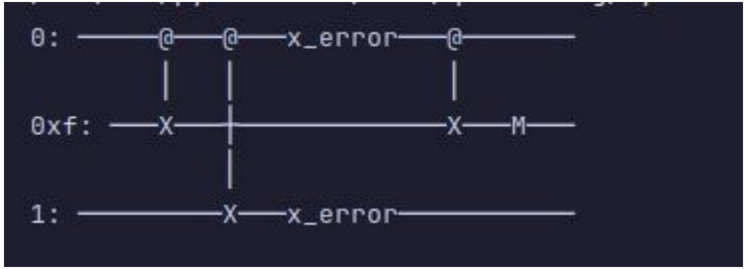
Error will propagate into the wire of flag qubits.Therefore change measurement outcome.



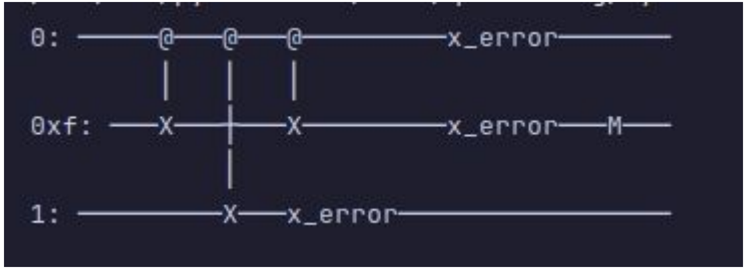
```
0: ──────@───@───@──────────
         │   │   │
0xf: ────X───┼───X───M──────
             │
1: ──────────X──────────────
```



```
0: ──────@───x_error───@───@──────────
         │             │   │
0xf: ────X─────────────┼───X───M──────
                       │
1: ────────────────────X──────────────
```



```
0: ──────@───@───x_error───@──────────
         │   │             │
0xf: ────X───┼─────────────X───M──────
             │
1: ──────────X───x_error──────────────
```



```
0: ──────@───@───@───────────x_error──────────
         │   │   │
0xf: ────X───┼───X───────────x_error───M──────
             │
1: ──────────X───x_error──────────────────────
```

Flag compiler:

- Use Jabalizer to turn a circuit into its ICM form ,which The ICM which include an array of Cnot.
- Create flag ancilla and engtangle them into the circuit.

=>The result is an equivalence circuit (doesn’t change the computation)

How Flags are added to the circuit :

In this project we will assess the effective of flags circuit created by the compiler based on three different strategy:

- Flags are entangled randomly to the circuit.
- Flags are entangled between Cnot that are applied serially
- Flags are entangled based on an error map.

Evaluation of flag circuit:

- Using Stim an high performance stabilizer simulation,we obtain the final state vector of a circuit without errors.
- Add errors of weigh v into the output and run simulation to obtain a list of state vectors

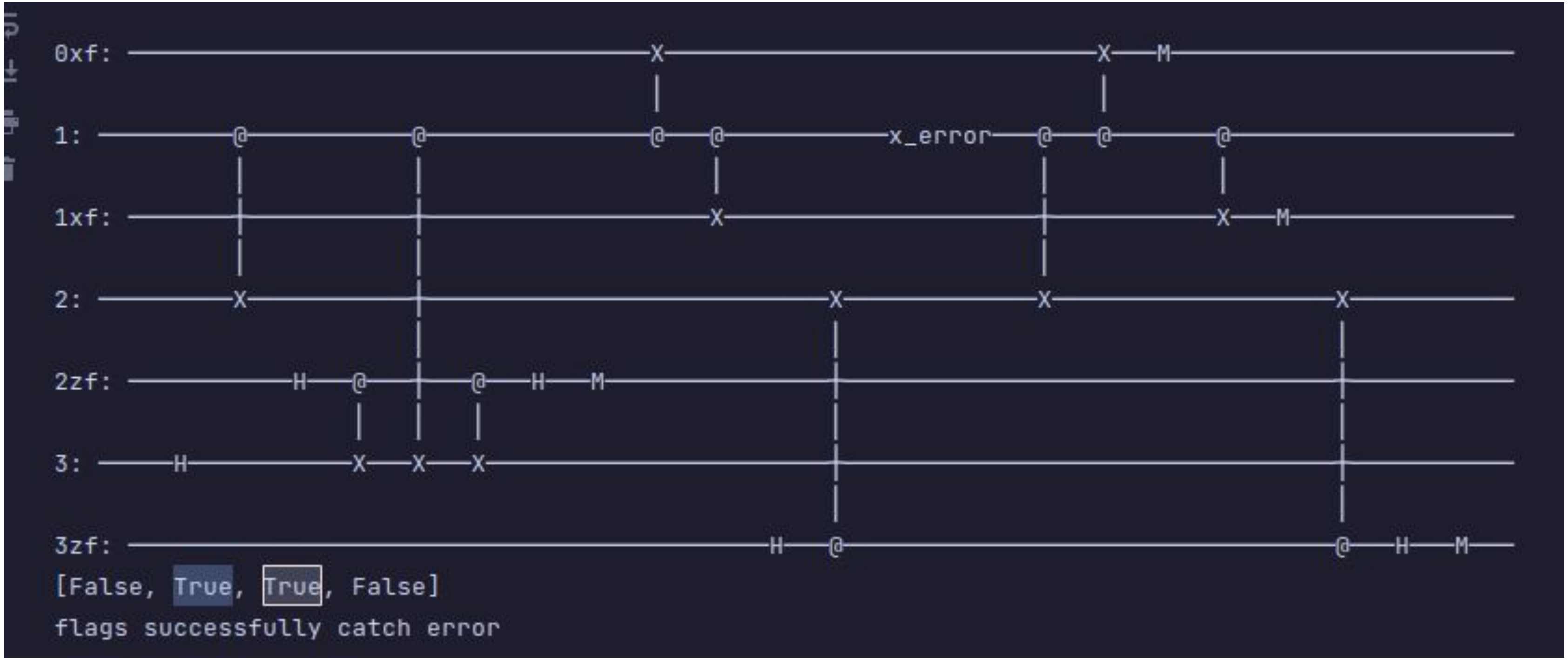
  => State vectors of circuits with errors resulting from v faults in the circuit have weight greater than v will not belong to the above list of state vectors

Evaluation of flag circuit:

The each individual errors will be characterized into four cases:

- Errors propagated into higher weighs errors and at least one flag measurement return True.
- Errors propagated into higher weighs errors and all flags measurement return False.
- Errors propagated into smaller or equal weight but flags measurement return True.(false alarm)
- Errors propagated into smaller or equal weight and flags return False.

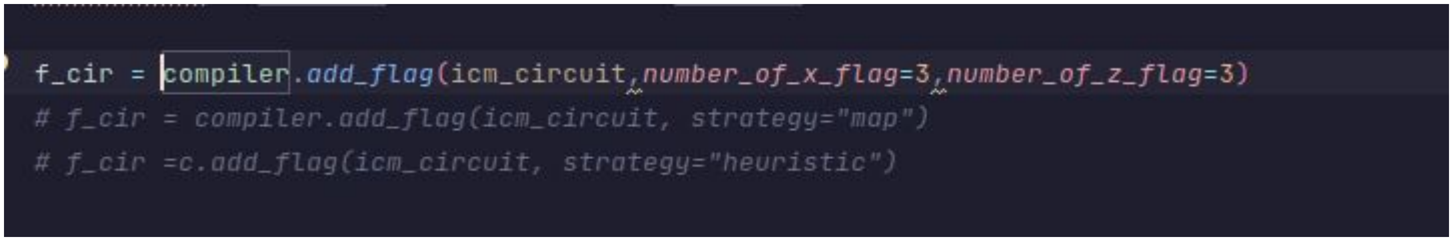
Example: (flag 1xf successfully inform us about error)



```
0xf: ───────────────────────────────────X─────────────────────────────X───M────────────────────────
                                        │                             │
1: ─────────@───────────@───────────────@───@───────────x_error───@───@───────@────────────────────
            │           │                   │                     │           │
1xf: ───────┼───────────┼───────────────────X─────────────────────┼───────────X───M────────────────
            │           │                                         │
2: ─────────X───────────┼───────────────────────────X─────────────X───────────────────X────────────
                        │                           │                                 │
2zf: ───────────H───@───┼───@───H───M───────────────┼─────────────────────────────────┼────────────
                    │   │   │                       │                                 │
3: ─────H───────────X───X───X───────────────────────┼─────────────────────────────────┼────────────
                                                    │                                 │
3zf: ───────────────────────────────────────────H───@─────────────────────────────────@───H───M────
[False, True, True, False]
flags successfully catch error
```

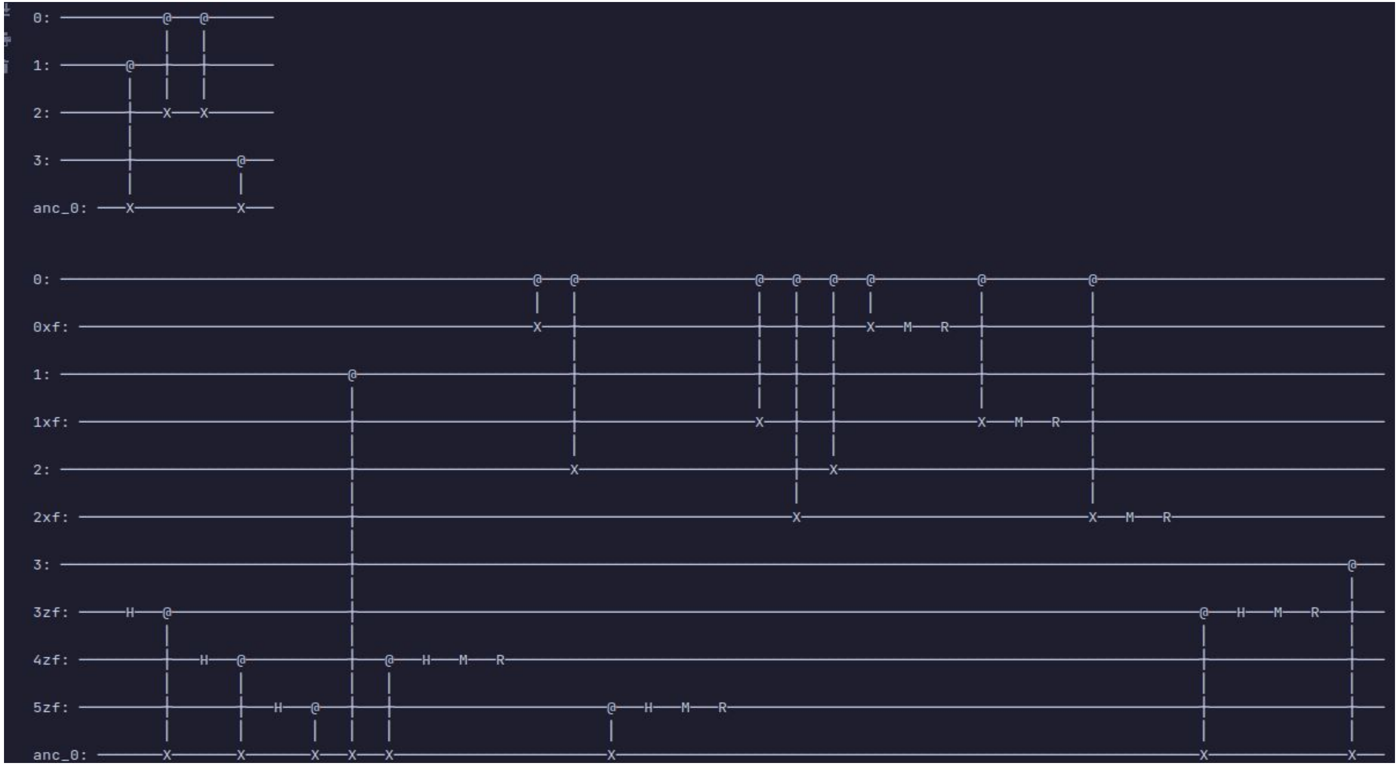
Randomized flag Circuit

- A number of x flags and z flags are entangled between Cnot.



```
f_cir = compiler.add_flag(icm_circuit,number_of_x_flag=3,number_of_z_flag=3)
# f_cir = compiler.add_flag(icm_circuit, strategy="map")
# f_cir =c.add_flag(icm_circuit, strategy="heuristic")
```

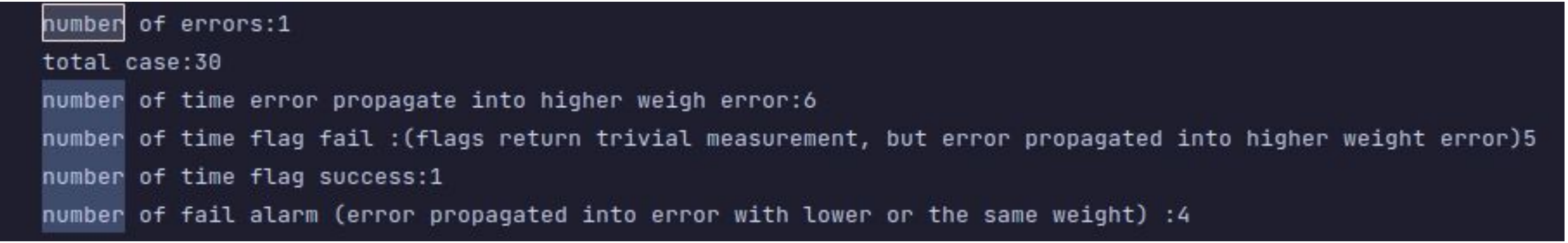
Randomized flag Circuit:



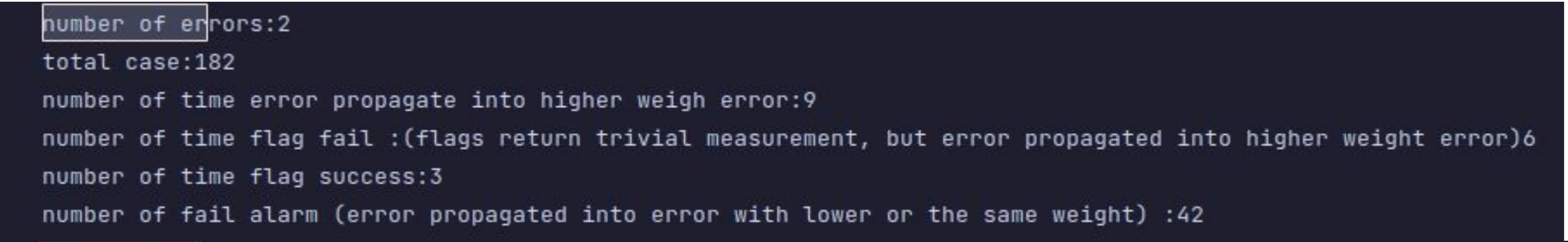
```
0: ───────────@───@───────
              │   │
1: ───────@───┼───┼───────
          │   │   │
2: ───────┼───X───X───────
          │
3: ───────┼───────────@───
          │           │
anc_0: ───X───────────X───

0: ───────────────────────────────────────────────────@───@───────────────────@───@───@───@───────────@───────────@────────────────────────────
                                                      │   │                   │   │   │   │           │           │
0xf: ─────────────────────────────────────────────────X───┼───────────────────┼───┼───┼───X───M───R───┼───────────┼────────────────────────────
                                                          │                   │   │   │               │           │
1: ───────────────────────────────@───────────────────────┼───────────────────┼───┼───┼───────────────┼───────────┼────────────────────────────
                                  │                       │                   │   │   │               │           │
1xf: ─────────────────────────────┼───────────────────────┼───────────────────X───┼───┼───────────────X───M───R───┼────────────────────────────
                                  │                       │                       │   │                           │
2: ───────────────────────────────┼───────────────────────X───────────────────────┼───X───────────────────────────┼────────────────────────────
                                  │                                               │                               │
2xf: ─────────────────────────────┼───────────────────────────────────────────────X───────────────────────────────X───M───R────────────────────
                                  │
3: ───────────────────────────────┼───────────────────────────────────────────────────────────────────────────────────────────────────────@───
                                  │                                                                                                       │
3zf: ─────H───@───────────────────┼───────────────────────────────────────────────────────────────────────────────────────@───H───M───R───┼───
              │                   │                                                                                       │               │
4zf: ─────────┼───H───@───────────┼───@───H───M───R───────────────────────────────────────────────────────────────────────┼───────────────┼───
              │       │           │   │                                                                                   │               │
5zf: ─────────┼───────┼───H───@───┼───┼───────────────────@───H───M───R───────────────────────────────────────────────────┼───────────────┼───
              │       │       │   │   │                   │                                                               │               │
anc_0: ───────X───────X───────X───X───X───────────────────X───────────────────────────────────────────────────────────────X───────────────X───
```

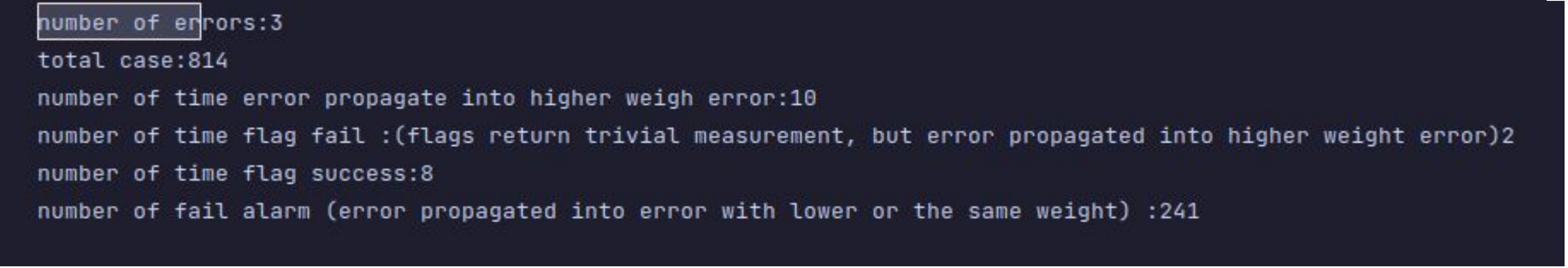
Randomized flag circuit



```
number of errors:1
total case:30
number of time error propagate into higher weigh error:6
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)5
number of time flag success:1
number of fail alarm (error propagated into error with lower or the same weight) :4
```



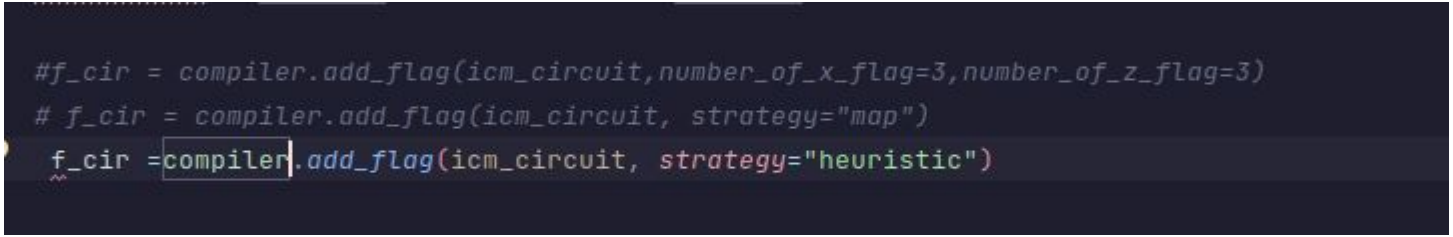
```
number of errors:2
total case:182
number of time error propagate into higher weigh error:9
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)6
number of time flag success:3
number of fail alarm (error propagated into error with lower or the same weight) :42
```



```
number of errors:3
total case:814
number of time error propagate into higher weigh error:10
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)2
number of time flag success:8
number of fail alarm (error propagated into error with lower or the same weight) :241
```

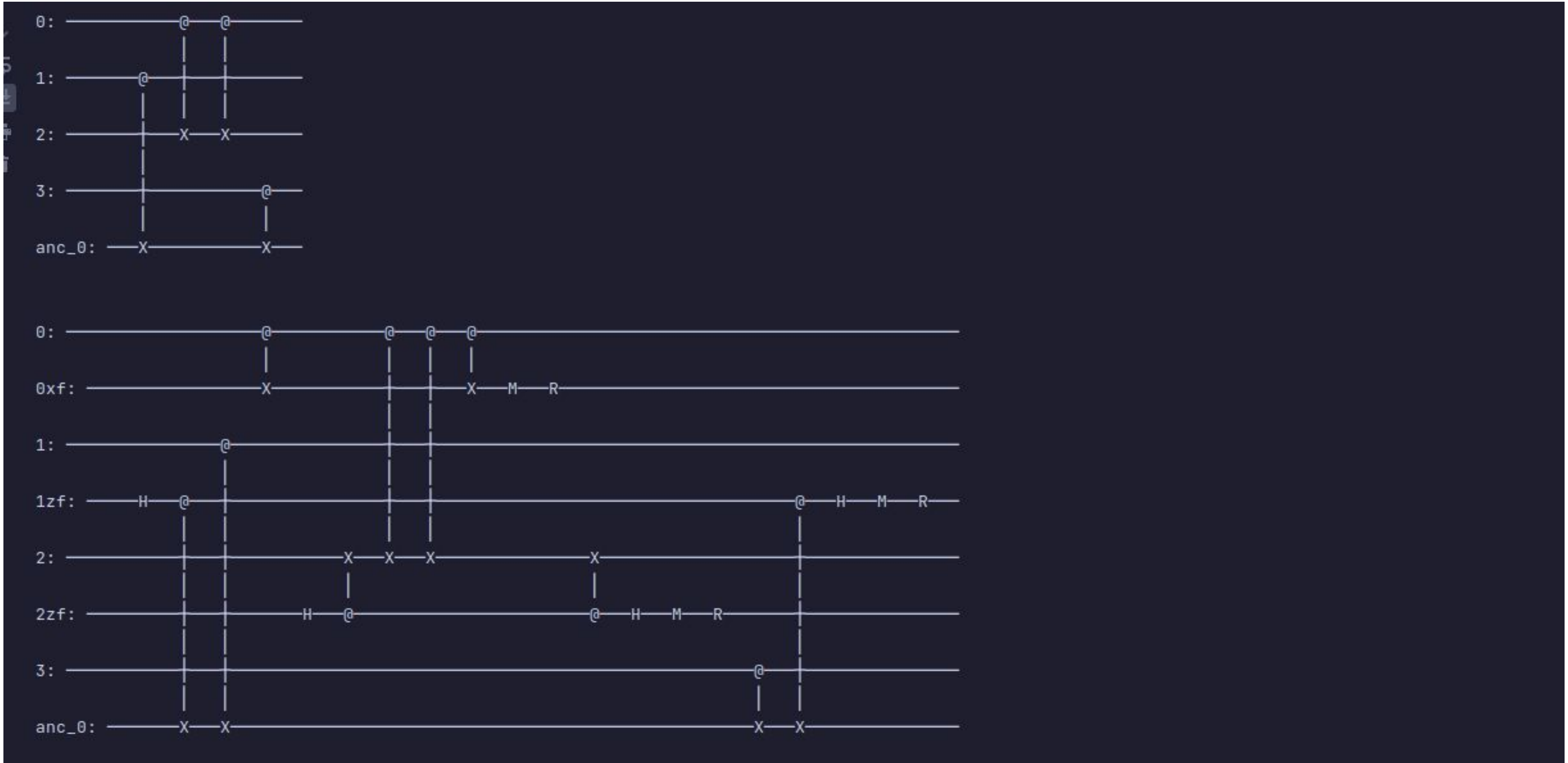
Heuristic flag circuit

- Flags are entangled between 2 or more Cnot that are applied serially



```
#f_cir = compiler.add_flag(icm_circuit,number_of_x_flag=3,number_of_z_flag=3)
# f_cir = compiler.add_flag(icm_circuit, strategy="map")
 f_cir =compiler.add_flag(icm_circuit, strategy="heuristic")
```

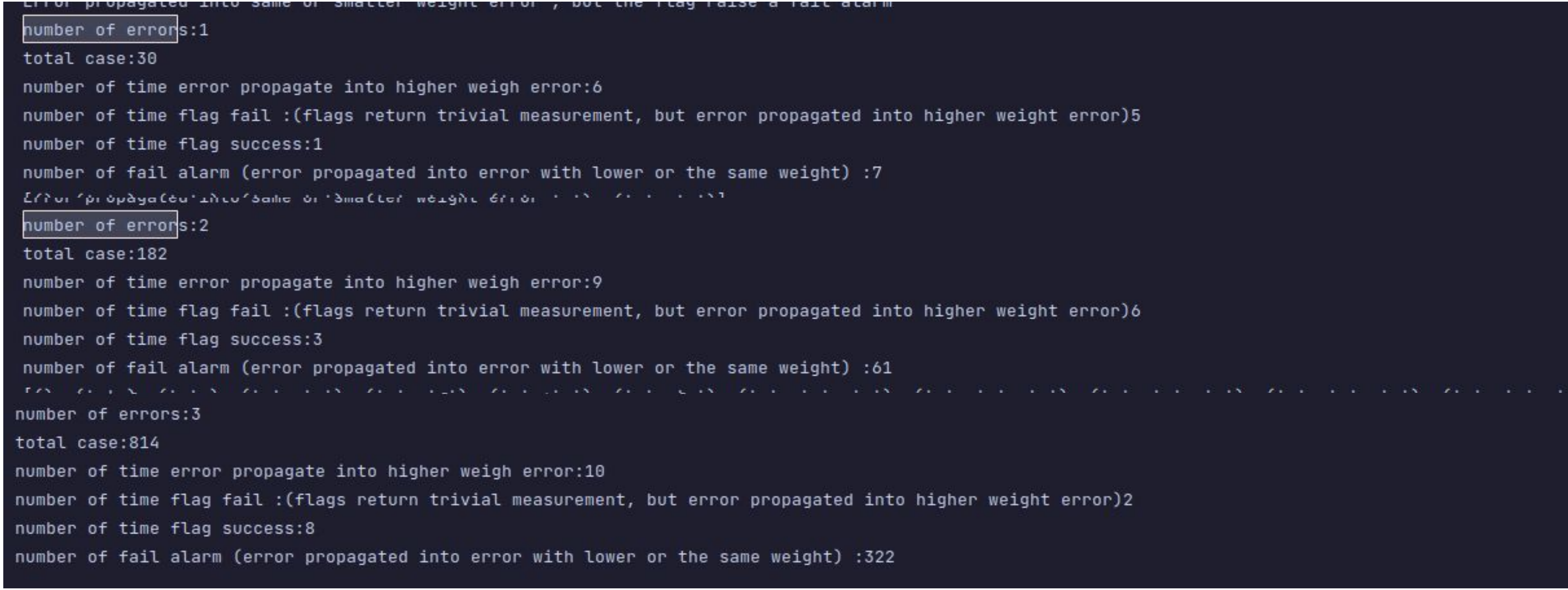
Heuristic flag circuit



```
0: ───────────@───@───────
              │   │
1: ───────@───┼───┼───────
          │   │   │
2: ───────┼───X───X───────
          │
3: ───────┼───────────@───
          │           │
anc_0: ───X───────────X───
```

```
0: ───────────────────@───────────@───@───@───────────────────────────────────────────────
                      │           │   │   │
0xf: ─────────────────X───────────┼───┼───X───M───R───────────────────────────────────────
                                  │   │
1: ───────────────@───────────────┼───┼───────────────────────────────────────────────────
                  │               │   │
1zf: ─────H───@───┼───────────────┼───┼───────────────────────────────────@───H───M───R───
              │   │               │   │                                   │
2: ───────────┼───┼───────────X───X───X───────────────X───────────────────┼───────────────
              │   │           │                       │                   │
2zf: ─────────┼───┼───────H───@───────────────────────@───H───M───R───────┼───────────────
              │   │                                                       │
3: ───────────┼───┼───────────────────────────────────────────────────@───┼───────────────
              │   │                                                   │   │
anc_0: ───────X───X───────────────────────────────────────────────────X───X───────────────
```

Heuristic flag circuit

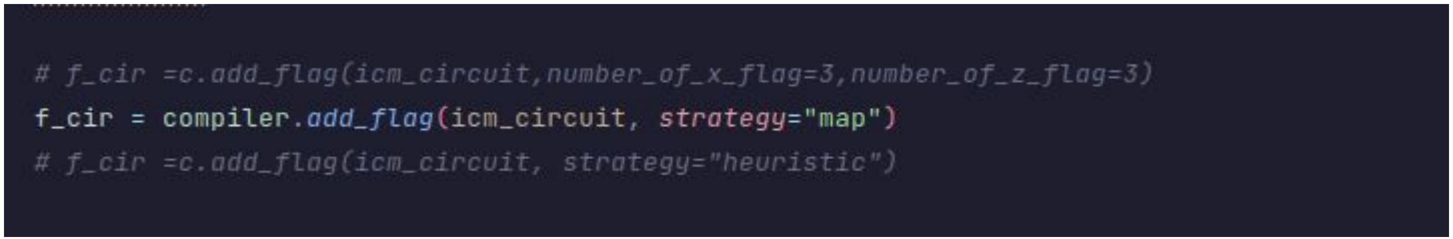


```
Error propagated into same or smaller weight error , but the flag raise a fail alarm
number of errors:1
total case:30
number of time error propagate into higher weigh error:6
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)5
number of time flag success:1
number of fail alarm (error propagated into error with lower or the same weight) :7
Error propagated into same or smaller weight error [illegible]
number of errors:2
total case:182
number of time error propagate into higher weigh error:9
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)6
number of time flag success:3
number of fail alarm (error propagated into error with lower or the same weight) :61
[illegible]
number of errors:3
total case:814
number of time error propagate into higher weigh error:10
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)2
number of time flag success:8
number of fail alarm (error propagated into error with lower or the same weight) :322
```

Flag Circuit map based on the errors map

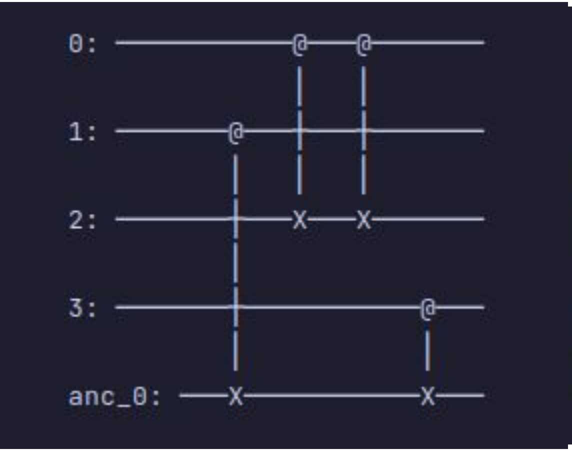
- Using memorization we built an map that represent how cnots propagate error:
- Add flag to between Cnot that can propagate a single gate error into higher weigh errors.

=> The result is a flag circuit that never fail to catch error

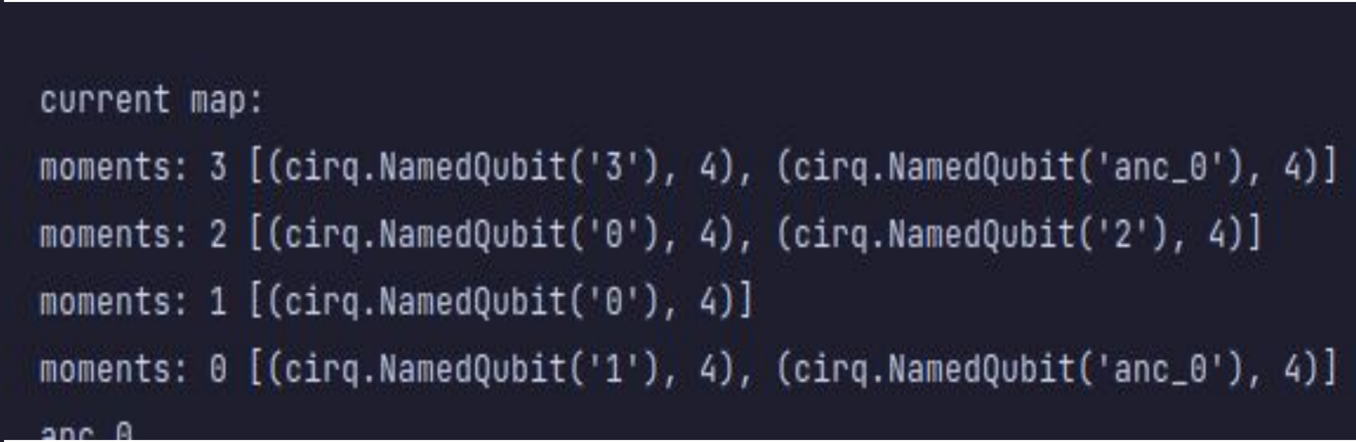


```
# f_cir =c.add_flag(icm_circuit,number_of_x_flag=3,number_of_z_flag=3)
f_cir = compiler.add_flag(icm_circuit, strategy="map")
# f_cir =c.add_flag(icm_circuit, strategy="heuristic")
```

Example of a map for X errors

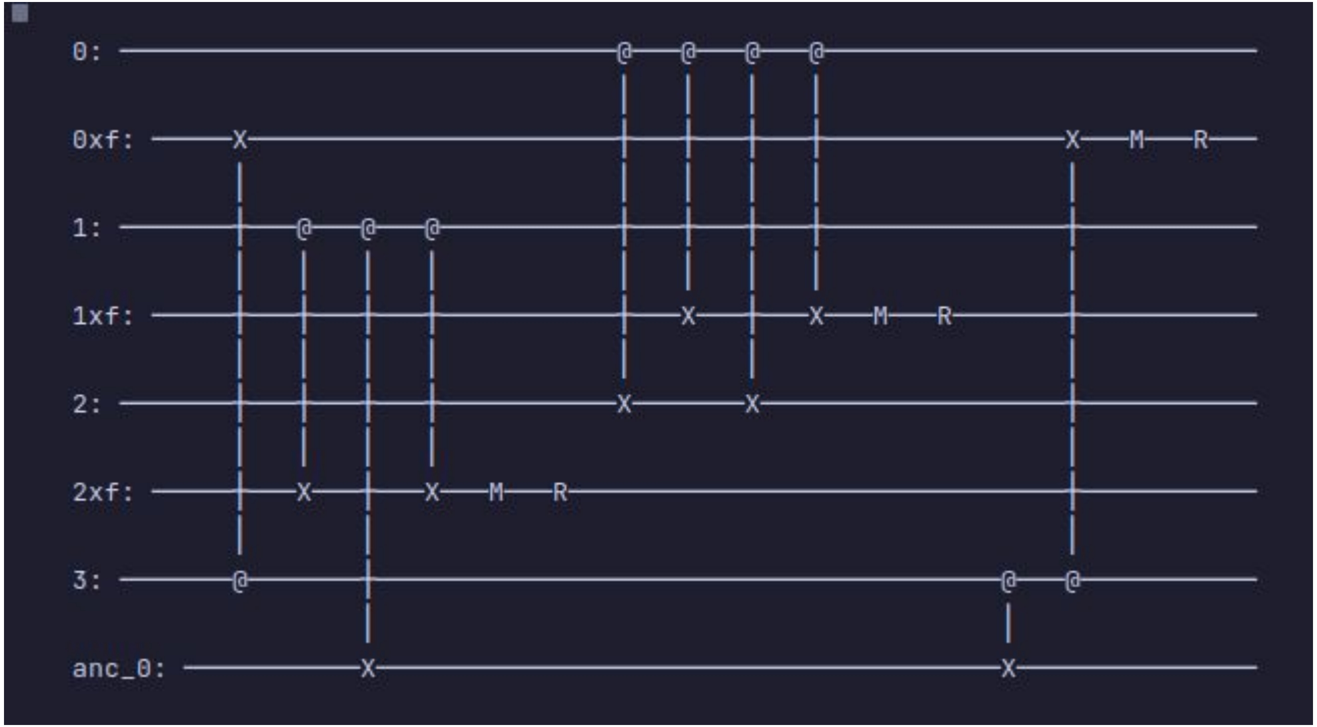


```
0: ──────────────@───@──────
                 │   │
1: ──────────@───┼───┼──────
             │   │   │
2: ──────────┼───X───X──────
             │
3: ──────────┼───────────@──
             │           │
anc_0: ──────X───────────X──
```



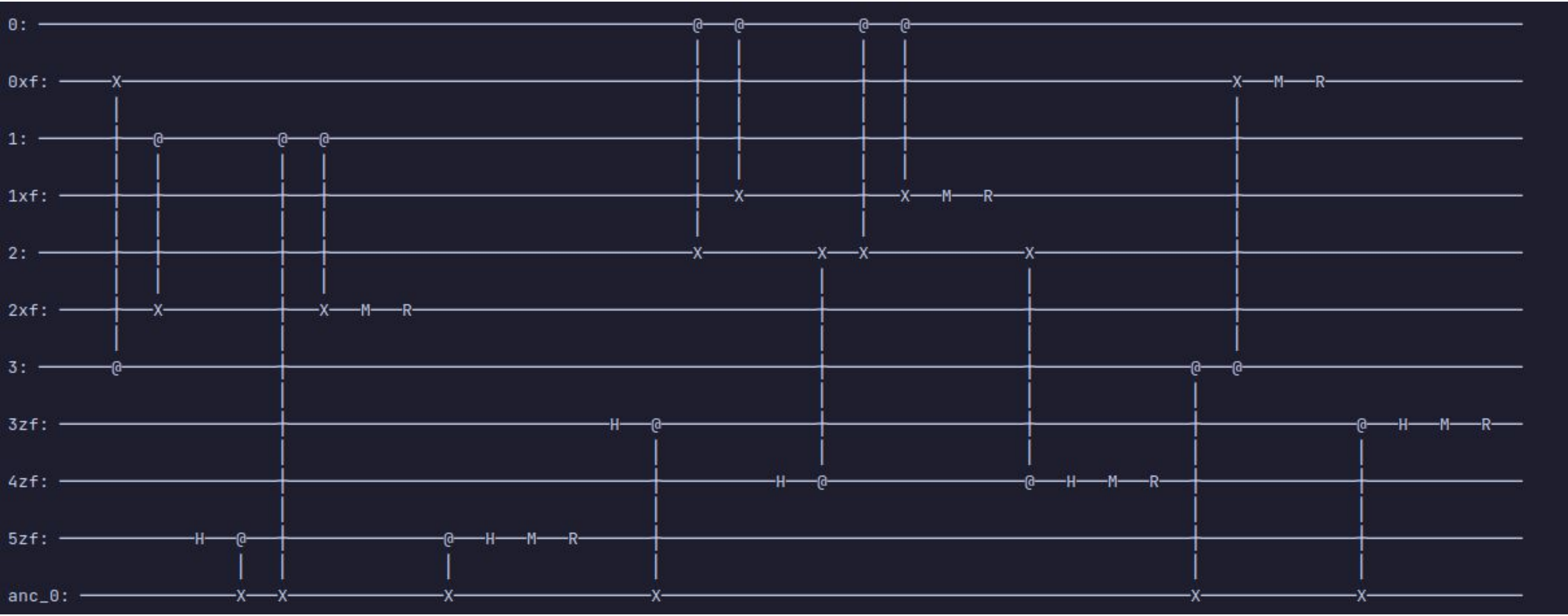
```
current map:
moments: 3 [(cirq.NamedQubit('3'), 4), (cirq.NamedQubit('anc_0'), 4)]
moments: 2 [(cirq.NamedQubit('0'), 4), (cirq.NamedQubit('2'), 4)]
moments: 1 [(cirq.NamedQubit('0'), 4)]
moments: 0 [(cirq.NamedQubit('1'), 4), (cirq.NamedQubit('anc_0'), 4)]
anc_0
```

Result for circuit that catch X errors:

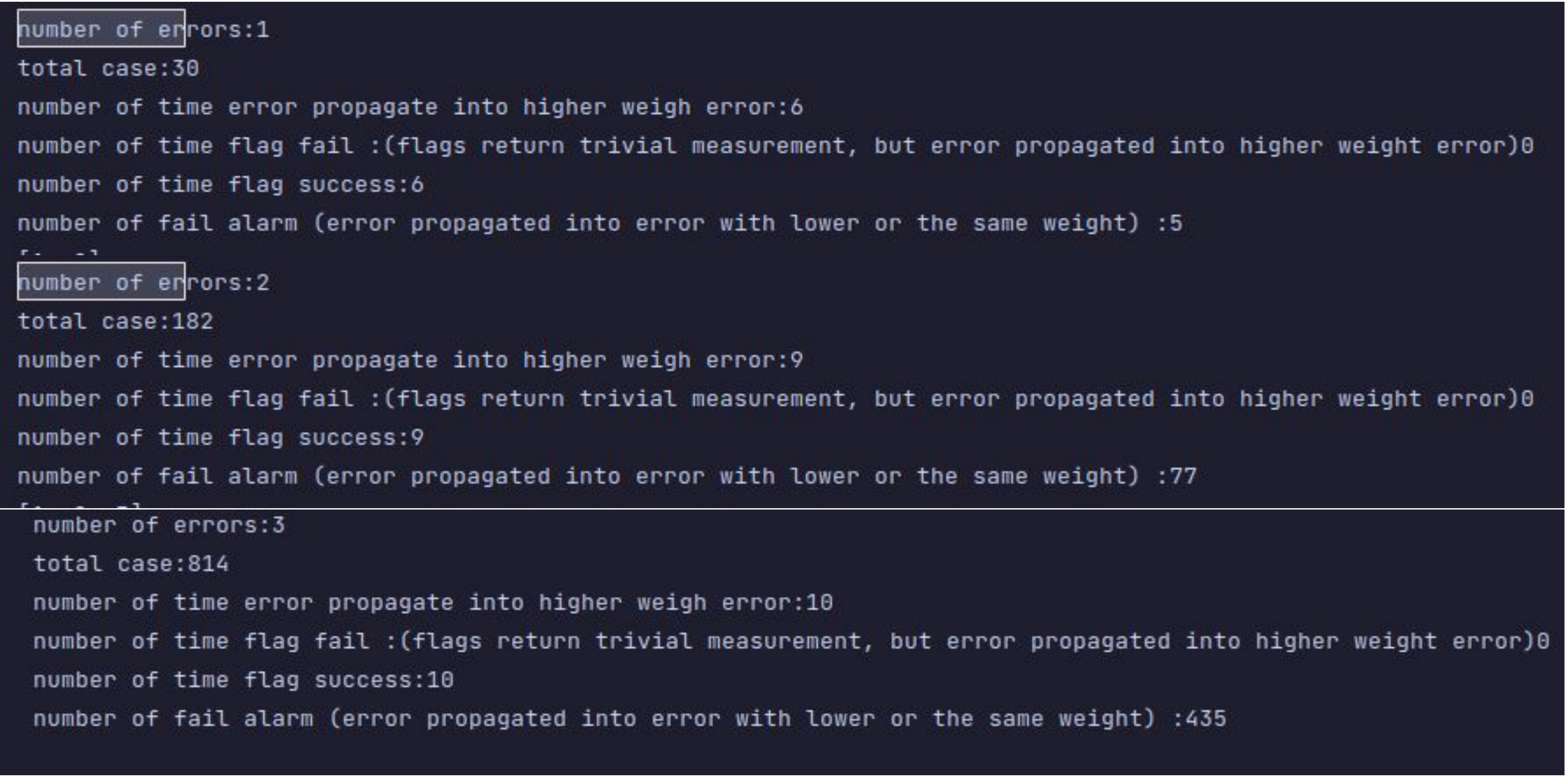


```
0: ───────────────────────────────@───@───@───@───────────────────────────
                                  │   │   │   │
0xf: ─────X───────────────────────┼───┼───┼───┼───────────────X───M───R───
          │                       │   │   │   │               │
1: ───────┼───@───@───@───────────┼───┼───┼───┼───────────────┼───────────
          │   │   │   │           │   │   │   │               │
1xf: ─────┼───┼───┼───┼───────────┼───X───┼───X───M───R───────┼───────────
          │   │   │   │           │       │                   │
2: ───────┼───┼───┼───┼───────────X───────X───────────────────┼───────────
          │   │   │   │                                       │
2xf: ─────┼───X───┼───X───M───R───────────────────────────────┼───────────
          │       │                                           │
3: ───────@───────┼───────────────────────────────────────@───@───────────
                  │                                       │
anc_0: ───────────X───────────────────────────────────────X───────────────
```

Flag qubits based on the errors map



```
0: ───────────────────────────────────────────────────────────────@───@───────────@───@───────────────────────────────────────────────────────
                                                                  │   │           │   │
0xf: ─────X───────────────────────────────────────────────────────┼───┼───────────┼───┼───────────────────────────X───M───R───────────────────
          │                                                       │   │           │   │                           │
1: ───────┼───@───────────@───@───────────────────────────────────┼───┼───────────┼───┼───────────────────────────┼───────────────────────────
          │   │           │   │                                   │   │           │   │                           │
1xf: ─────┼───┼───────────┼───┼───────────────────────────────────┼───X───────────┼───X───M───R───────────────────┼───────────────────────────
          │   │           │   │                                   │               │                               │
2: ───────┼───┼───────────┼───┼───────────────────────────────────X───────────X───X───────────X───────────────────┼───────────────────────────
          │   │           │   │                                               │                   │               │
2xf: ─────┼───X───────────┼───X───M───R───────────────────────────────────────┼───────────────────┼───────────────┼───────────────────────────
          │               │                                                   │                   │               │
3: ───────@───────────────┼───────────────────────────────────────────────────┼───────────────────┼───────────────@───@───────────────────────
                          │                                                   │                   │               │
3zf: ─────────────────────┼───────────────────────────H───@───────────────────┼───────────────────┼───────────────┼───────────────@───H───M───R───
                          │                               │                   │                   │               │               │
4zf: ─────────────────────┼───────────────────────────────┼───────────H───@───────────────────@───H───M───R───┼───────────────┼───────────
                          │                               │                                                   │               │
5zf: ─────────────H───@───┼───────────@───H───M───R───────┼───────────────────────────────────────────────────┼───────────────┼───────────
                      │   │           │                   │                                                   │               │
anc_0: ───────────────X───X───────────X───────────────────X───────────────────────────────────────────────────X───────────────X───────────
```



```
number of errors:1
total case:30
number of time error propagate into higher weigh error:6
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)0
number of time flag success:6
number of fail alarm (error propagated into error with lower or the same weight) :5
```

```
number of errors:2
total case:182
number of time error propagate into higher weigh error:9
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)0
number of time flag success:9
number of fail alarm (error propagated into error with lower or the same weight) :77
```

```
number of errors:3
total case:814
number of time error propagate into higher weigh error:10
number of time flag fail :(flags return trivial measurement, but error propagated into higher weight error)0
number of time flag success:10
number of fail alarm (error propagated into error with lower or the same weight) :435
```

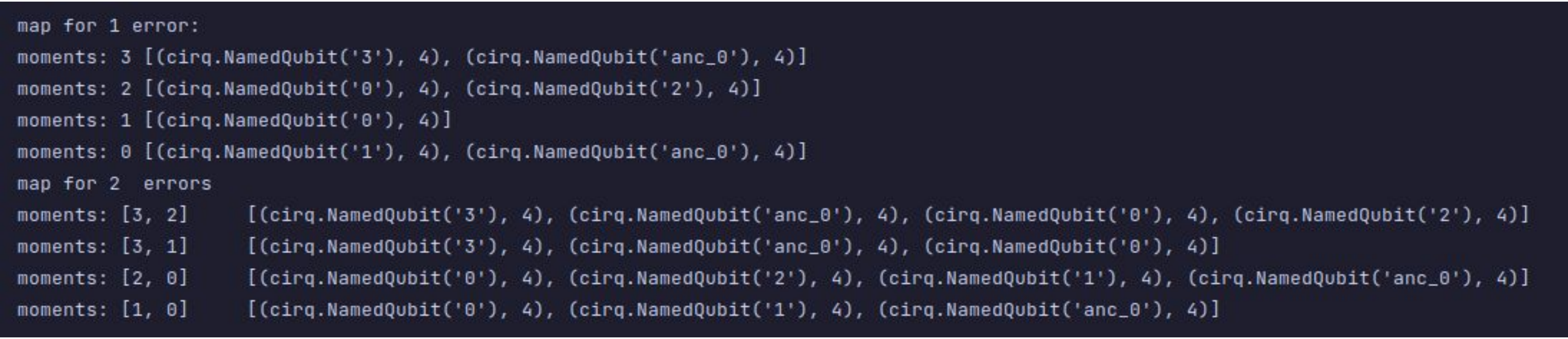
How many flags are needed to detect maximum errors???

- If an error P results in weight p error in output , error Q results in weight q error in output , the combination of P Q result in error of weight <= p + q
    - Proof: Error can be cancer out.

=> Using the errors map we can determine the minimal flags needed.

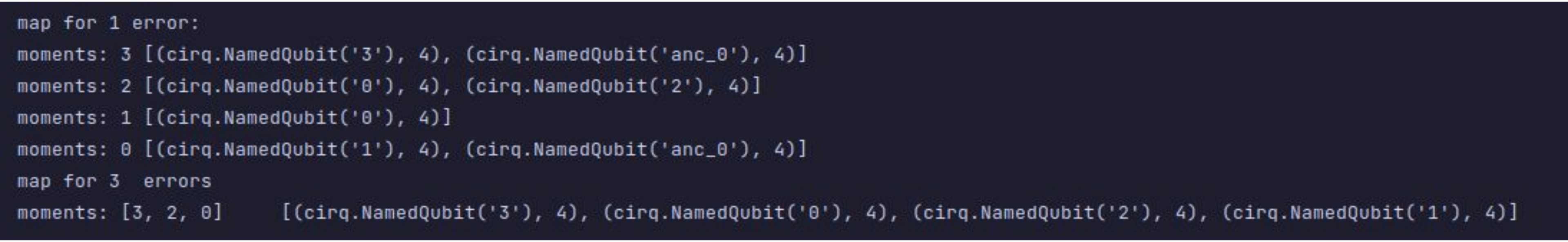
=> We need fewer flags to detect a high-weight error than to detect a lower-weight error.

How many flags are needed to detect propagation errors???



```
map for 1 error:
moments: 3 [(cirq.NamedQubit('3'), 4), (cirq.NamedQubit('anc_0'), 4)]
moments: 2 [(cirq.NamedQubit('0'), 4), (cirq.NamedQubit('2'), 4)]
moments: 1 [(cirq.NamedQubit('0'), 4)]
moments: 0 [(cirq.NamedQubit('1'), 4), (cirq.NamedQubit('anc_0'), 4)]
map for 2  errors
moments: [3, 2]     [(cirq.NamedQubit('3'), 4), (cirq.NamedQubit('anc_0'), 4), (cirq.NamedQubit('0'), 4), (cirq.NamedQubit('2'), 4)]
moments: [3, 1]     [(cirq.NamedQubit('3'), 4), (cirq.NamedQubit('anc_0'), 4), (cirq.NamedQubit('0'), 4)]
moments: [2, 0]     [(cirq.NamedQubit('0'), 4), (cirq.NamedQubit('2'), 4), (cirq.NamedQubit('1'), 4), (cirq.NamedQubit('anc_0'), 4)]
moments: [1, 0]     [(cirq.NamedQubit('0'), 4), (cirq.NamedQubit('1'), 4), (cirq.NamedQubit('anc_0'), 4)]
```

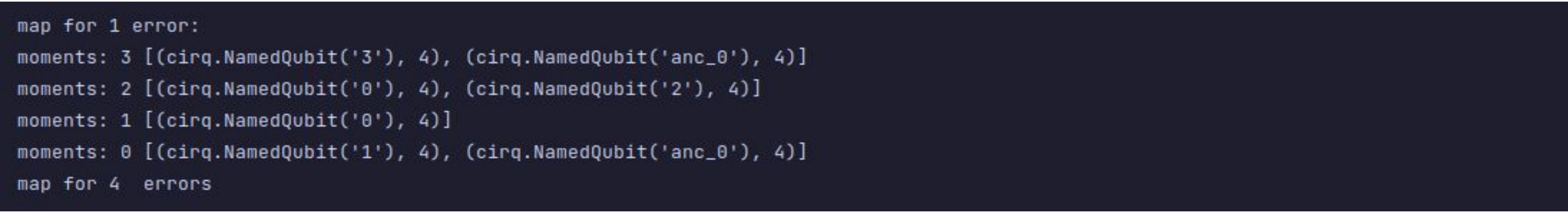
How many flags are needed to detect propagation errors???



```
map for 1 error:
moments: 3 [(cirq.NamedQubit('3'), 4), (cirq.NamedQubit('anc_0'), 4)]
moments: 2 [(cirq.NamedQubit('0'), 4), (cirq.NamedQubit('2'), 4)]
moments: 1 [(cirq.NamedQubit('0'), 4)]
moments: 0 [(cirq.NamedQubit('1'), 4), (cirq.NamedQubit('anc_0'), 4)]
map for 3  errors
moments: [3, 2, 0]     [(cirq.NamedQubit('3'), 4), (cirq.NamedQubit('0'), 4), (cirq.NamedQubit('2'), 4), (cirq.NamedQubit('1'), 4)]
```

How many flags are needed to detect Maximum errors???

For this particular circuit , there is no combination of 4 errors that result in error greater than weigh 4



```
map for 1 error:
moments: 3 [(cirq.NamedQubit('3'), 4), (cirq.NamedQubit('anc_0'), 4)]
moments: 2 [(cirq.NamedQubit('0'), 4), (cirq.NamedQubit('2'), 4)]
moments: 1 [(cirq.NamedQubit('0'), 4)]
moments: 0 [(cirq.NamedQubit('1'), 4), (cirq.NamedQubit('anc_0'), 4)]
map for 4  errors
```

What have been achieved ?

- High level circuit can be decomposed into ICM circuit
- A compiler that can automatically add flag to ICM circuit
- A strategy for adding flags that don’t fail at catching errors
- An error map that can be used to analyze error propagation of a circuit

What can be improve ?

1. The evaluation process is slow which make it not possible to work with complicated circuit with big number qubits.This problem can be resolved by using stabilizer
2. The installation step of ICM form should also be taken into consider because Hadamard gate can negate the effect of errors.

2. The Number of false alarm can significantly reduced based on map error.(many flag raise at the same time is a good sign of false alarm)

3. Parallel computing can be used to speed up the evaluation and the creation of error maps