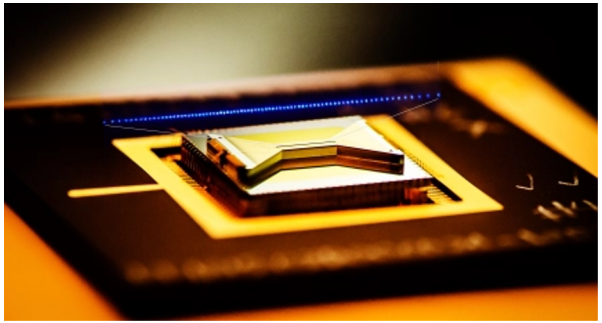
Quantum Tic Tac Toe

Alfa, Amrit, Chakhriya, Rohit, Josep

Instructions



The game has 9 tiles, each corresponds to a real IonQ qubit!

The game has 2 players, player 1 and player 2.

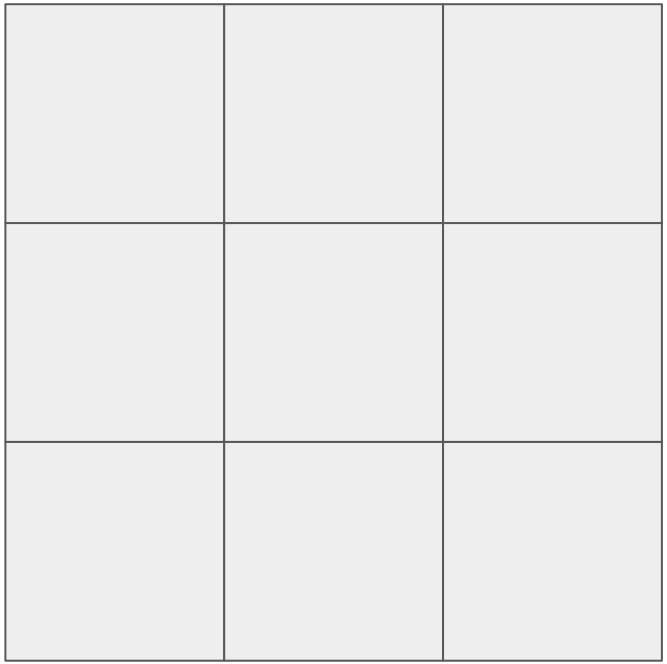
They take turns as in classical tic tac toe marking one tile at a time.

Player 1 makes a tile with a |0> and player 2 makes a tile with a |1>.

Out quantum tic tac toe has a twist. During a player turn, a player can decide to entangle two tiles instead of marking a tile. If that is the case, then the entangled tiles can not be marked for any other player during the game.

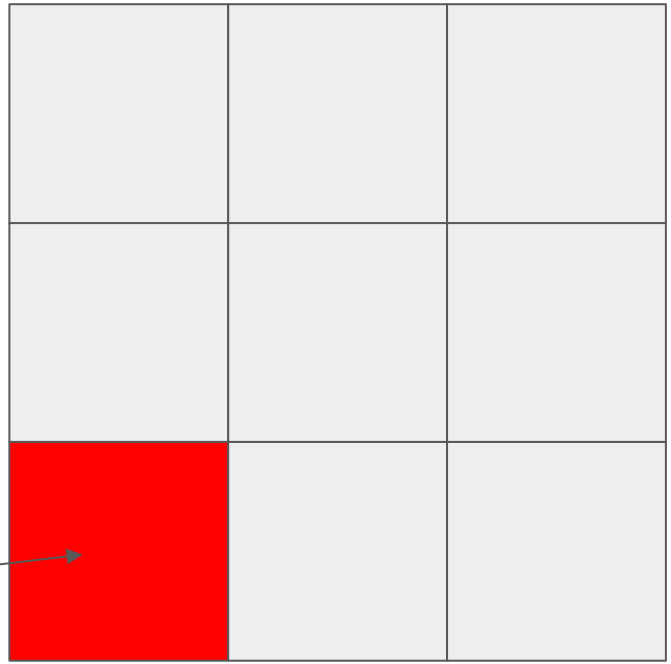
Once all the tiles are marked or entangled the quantum circuit of the game is initialized and the qubits are measured revealing the final state of the game. An error correcting algorithm makes sure that there was no errors. If there are errors the code is run again until there are no errors. Who ever has 3 in a line wins. If both will then the code is run again until one wins.

Game example



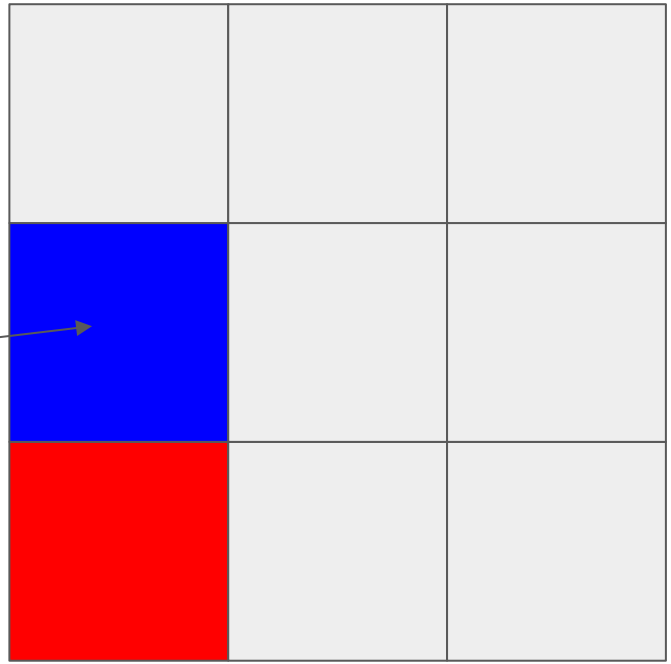
Game example: turn 1, plays player 1

Player 1 starts with a classical move



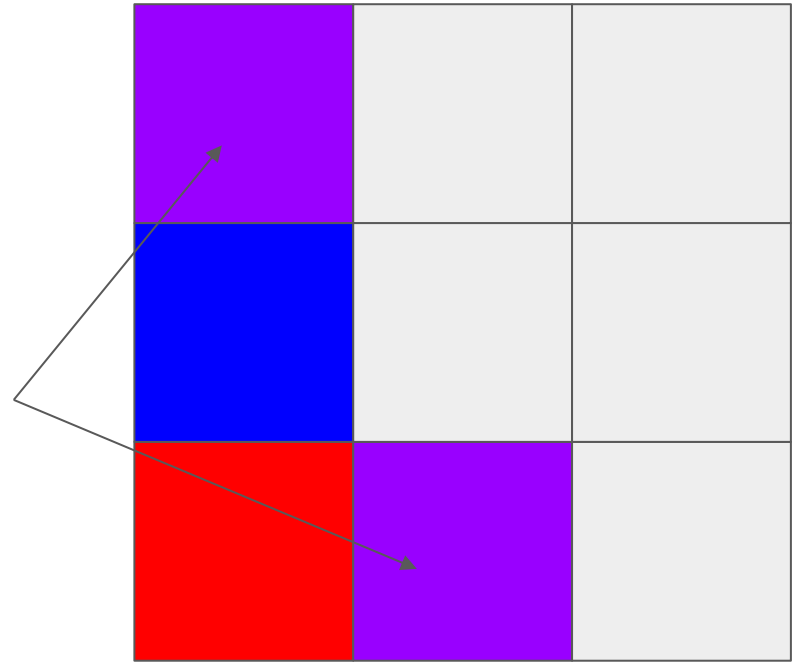
Game example: turn 2, plays player 2

Player 2 starts with a classical move



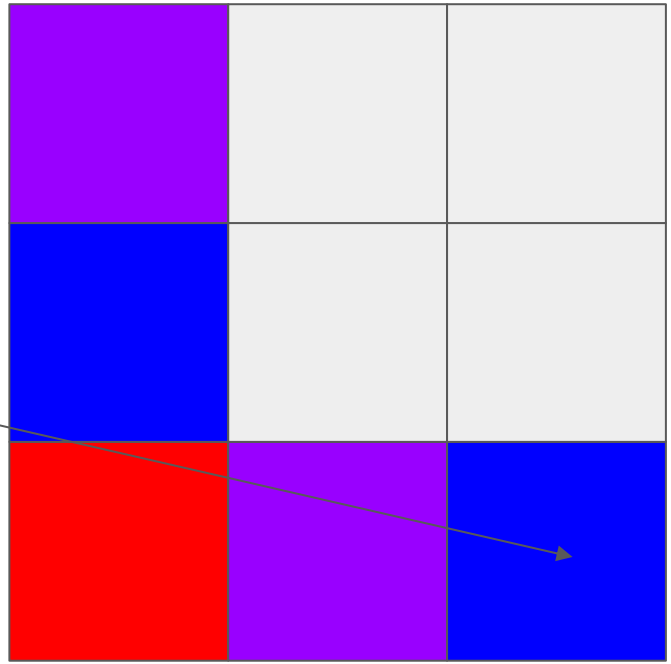
Game example: turn 3, plays player 1

Player 1 entangles two tiles



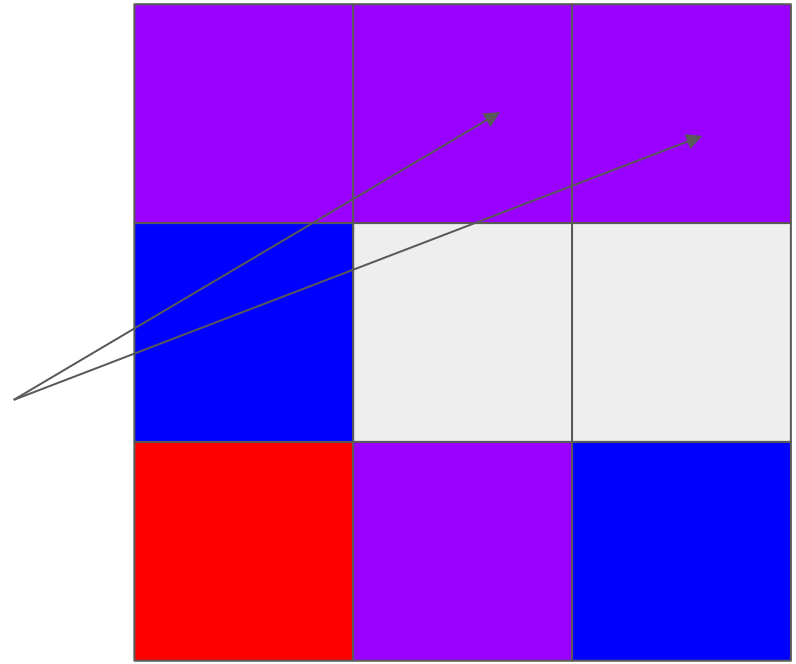
Game example: turn 4, plays player 2

Player 2 plays a classical move



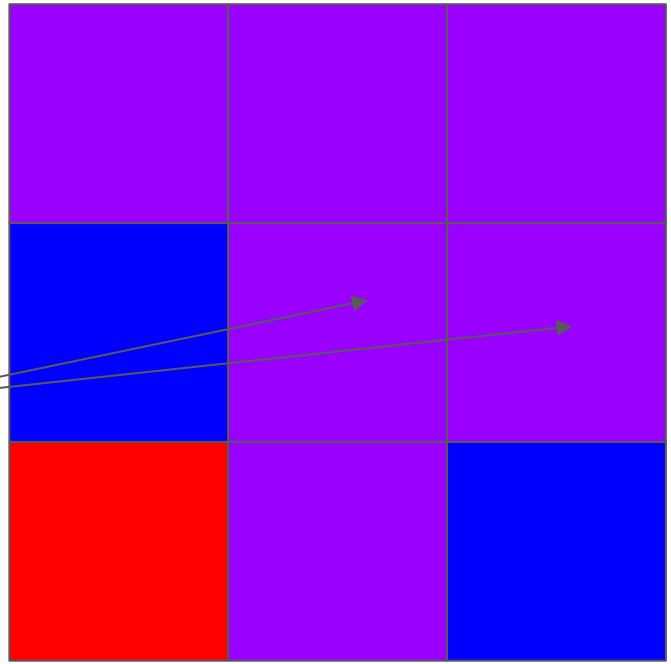
Game example: turn 5, plays player 1

Payer 1 entangles two tiles



Game example: turn 6, plays player 2

Payer 2 entangles two tiles

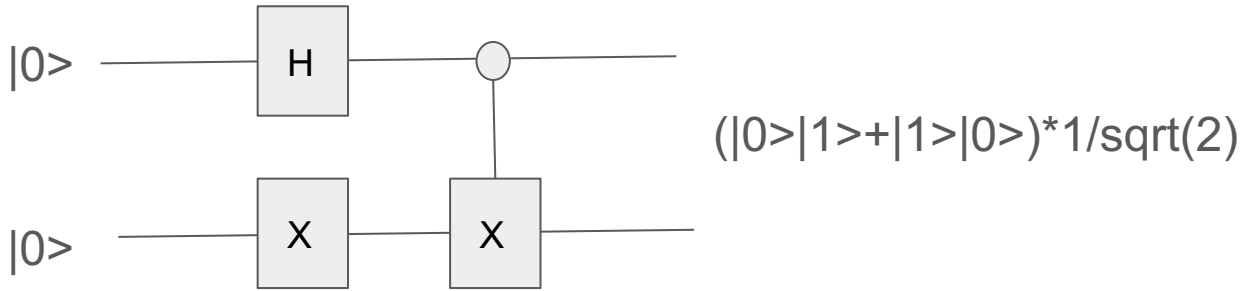


The state of the board is generated by our code

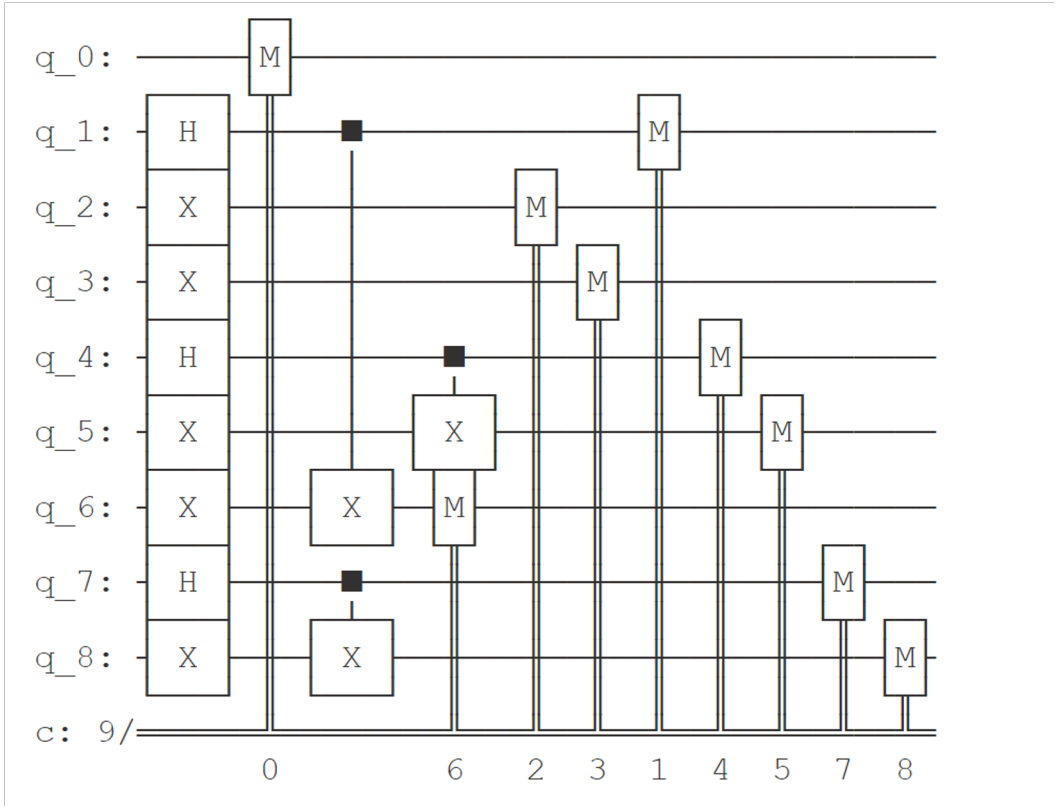
-The tiles that player 1 choses are set to |0>

-The tiles that player 2 choses are set to |1> by applying a X operation to the initial qubit.

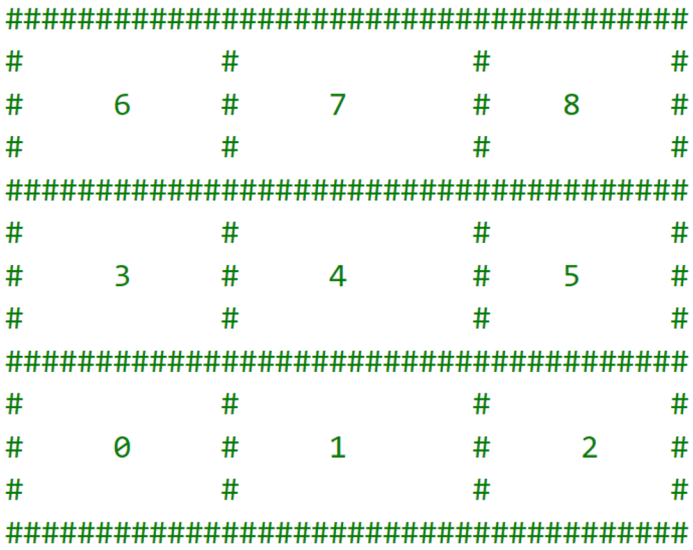
-The tile pairs that are entangled are entangled to |0>|1>+|1>|0> using the following circuit



The circuit that represents the system state for our example



Following this tile number



The circuit is sent to be executed by ION-Q QPU

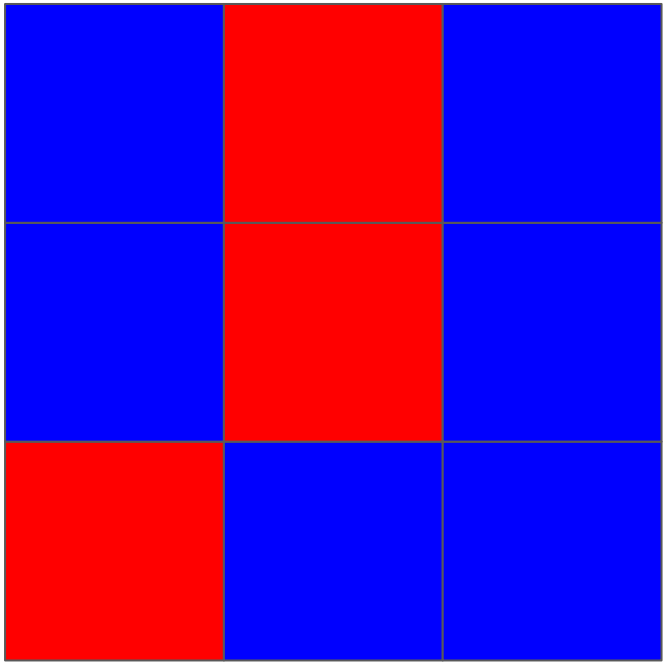
The quantum computer collapses the state

```
#simulator_backend = provider.get_backend("ionq.simulator")
simulator_backend = provider.get_backend("ionq.qpu")
job = simulator_backend.run(circuit, shots=1)
job_id = job.id()
print("quantum tic", job_id)
job_monitor(job)
result = job.result()
print(result.get_counts())
```

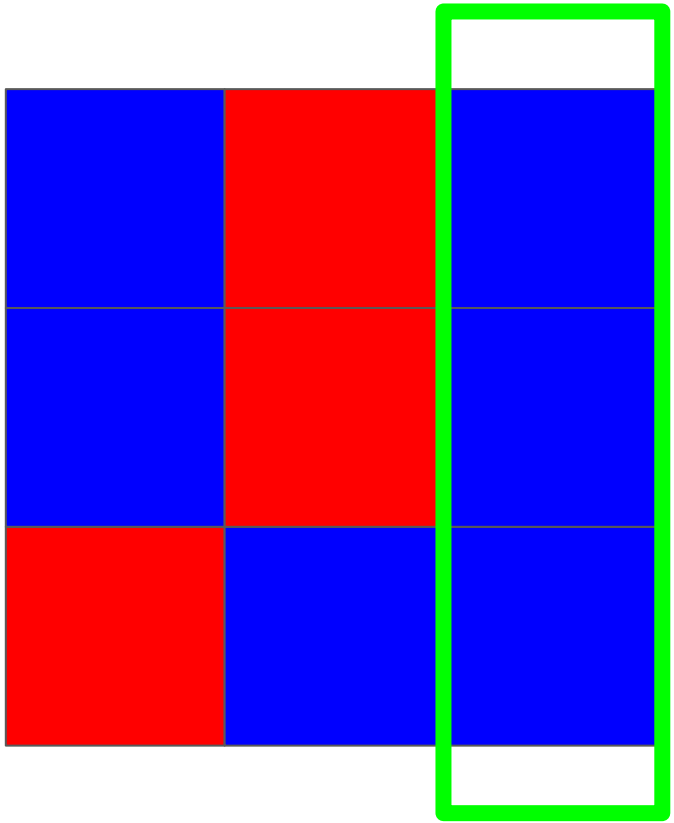
[39] ✓ 3 h 0 min

```
quantum tic 060b816e-8171-11ec-a940-00155d0077a0
Job Status: job has successfully run
{'100101110': 1}
```

Qubit 8 Qubit 0



Player 2 wins!



- If both players win, the circuit runs again until one player wins.
- If no players win, the circuit runs again
- If errors in the collapsed have been detected then the circuit runs again

Some times an error can occur and the state |0>|1>+|1>|0> can collapse to |11> or |00>. This will be considered an error