iQuHACK 2026 Challenge

Superquantum

January 2026

1 Challenge overview

In fault-tolerant quantum computing, non-Clifford gates are very expensive. Real quantum algorithms are built from large amounts of reversible arithmetic and T-heavy subroutines. Your goal in this challenge is to to minimize fault-tolerant cost by compiling few-qubit quantum circuits into a sequences of Clifford+T gates, i.e., the gates generated by $\{H, T, CNOT\}$ operations. Your circuits must only contain $\{H, T, T^\dagger, CNOT\}$ gates. You can find the specifics on these circuits in the next section. Each subsequent compilation task is meant to be a little more involved than a previous ones. Also, many consequential pairs of circuits are related, so try to really understand the structure of your solutions.

The topic of Clifford+T compilation is rich and mature, so there are a lot of useful references you can find online that will help you with the challenge. You might find the following paper and corresponding software handy: Refs. [1, 2]. You are also welcome to use Superquantum's very own `rmsynth` compiler [3] which is based on the research by Amy & Mosca [4, 5]. Note, however, that you are *not* required to use any particular software, but we ask you to submit your quantum circuits as `qasm` files. More on that in Section 3.

2 Unitaries to compile & scoring

As mentioned you will be asked to compile unitaries into Clifford+T sequences. There are ten 2-qubit ones and a single 4-qubit one:

1. Controlled-Y Gate: $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -i \\ 0 & 0 & i & 0 \end{bmatrix}$.

   This is just a sanity check to make sure you understand the task and that the submission works properly.

2. Controlled-Ry($\pi/7$): $\approx \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.97 & -0.22 \\ 0 & 0 & 0.22 & 0.97 \end{bmatrix}$.

Here, you will really need to apply your compilers!

3. Exponential of a Pauli string: $\exp(i\frac{\pi}{7}Z \otimes Z)$.

   This is a crucial ingredient in quantum simulation algorithms.

4. Exponential of a Hamiltonian $H_1 = XX + YY$: $\exp(i\frac{\pi}{7}H_1)$.

   Think about the structure of the summands in H_1.

5. Exponential of a Hamiltonian $H_2 = XX + YY + ZZ$: $\exp(i\frac{\pi}{2}H_2)$.

   Think about the structure of the terms as well as the matrix form. Does it resemble anything? The compiled circuit is easier than you might think.

6. Exponential of a Hamiltonian: $H_3 = XX + ZI + IZ$: $\exp(i\frac{\pi}{7}H_3)$.

   This is a time evolution under a 2-qubit transverse field Ising model.

7. State preparation. Design $U \in \mathbb{C}^{4\times 4}$ that maps

$$\begin{aligned} |00\rangle \mapsto &(0.1061479384 - 0.679641467i)|00\rangle \\ &+(-0.3622775887 - 0.453613136i)|01\rangle \\ &+(0.2614190429 + 0.0445330969i)|10\rangle \\ &+(0.3276449279 - 0.1101628411i)|11\rangle. \end{aligned}$$

   Preparation of arbitrary quantum states is an important and a notoriously hard task in quantum computing. Note that there are many unitaries that can perform this mapping. We ask you to submit the one for which you obtain the best metrics.

   In case relevant, this state is generated via
   `qiskit.quantum_info.random_statevector(4, seed=42)`.

8. Structured unitary 1:

$$\frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix}.$$

   Figure out what the structure is! You should be able to efficiently compile it using some of the previous results.

9. Structured unitary 2:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & -\frac{1}{2}+\frac{i}{2} & \frac{1}{2}+\frac{i}{2} \\ 1 & i & 0 & 0 \\ 1 & i & -1+i & 0 \end{bmatrix}.$$

   Same as above – figure out the structure of this gate and it will be easy from there.

10. Random unitary:

$$\begin{bmatrix} 0.1448081895 + 0.1752383997\,i & -0.5189281551 - 0.5242425896\,i & -0.1495585824 + 0.312754999\,i & 0.1691348143 - 0.5053863118\,i \\ -0.9271743926 - 0.0878506193\,i & -0.1126033063 - 0.1818584963\,i & 0.1225587186 + 0.0964028611\,i & -0.2449850904 - 0.0504584131\,i \\ -0.0079842758 - 0.2035507051\,i & -0.3893205530 - 0.0518092515\,i & 0.2605170566 + 0.3286402481\,i & 0.4451730754 + 0.6558933250\,i \\ 0.0313792249 + 0.1961395216\,i & 0.4980474972 + 0.0884604926\,i & 0.3407886532 + 0.7506609982\,i & 0.0146480652 - 0.1575584270\,i \end{bmatrix}.$$

Random quantum circuits are widely used in "quantum supremacy" experiments [6] and are even believed to have connections to black hole physics [7].

In case relevant, this matrix is generated via `qiskit.quantum_info.random_unitary(4, seed=42)`.

11. 4-qubit diagonal unitary. Consider U acting on 4 qubits such that $U|x\rangle = e^{i\varphi(x)}|x\rangle$ and $x \in \{0,1\}^4$. The phase $\varphi(x)$ is the following:

$$\begin{aligned} &\varphi(0000) = 0, \\ &\varphi(0001) = \pi, \quad \varphi(0010) = \frac{5}{4}\pi, \quad \varphi(0011) = \frac{7}{4}\pi, \\ &\varphi(0100) = \frac{5}{4}\pi, \quad \varphi(0101) = \frac{7}{4}\pi, \quad \varphi(0110) = \frac{3}{2}\pi, \quad \varphi(0111) = \frac{3}{2}\pi, \\ &\varphi(1000) = \frac{5}{4}\pi, \quad \varphi(1001) = \frac{7}{4}\pi, \quad \varphi(1010) = \frac{3}{2}\pi, \quad \varphi(1011) = \frac{3}{2}\pi, \\ &\varphi(1100) = \frac{3}{2}\pi, \quad \varphi(1101) = \frac{3}{2}\pi, \quad \varphi(1110) = \frac{7}{4}\pi, \quad \varphi(1111) = \frac{5}{4}\pi. \end{aligned}$$

Compile this circuit with as few T gates and CNOT gates as possible. (Who paid attention at the workshop?)

3 Submission and scoring

Throughout the duration of the challenge, the compilation results should be submitted through iquhack.superquantum.io. Upon opening the website, you will be prompted to enter your team's API key which will be provided to you shortly after the start of the hackathon. After entering the key, you will be able to see the submissions from all the teams and the corresponding submission metrics. You will also be able to make your own submission for any of the challenges. The submissions should be made as plaintext OpenQASM circuit specifications, with both OpenQASM 2 and OpenQASM 3 being supported. After each successful submission, your team will be issued a 3 minute cooldown before you can make another submission for any of the challenges. Even after making a submission, please keep the corresponding OpenQASM file locally in case of any technical issues down the line.

There is no single scoring function used to determine the best submission for a challenge. Instead, the judges will take all of the present submission metrics into account to evaluate each team. The final team placement will also depend heavily on the final write-up and presentation detailing the specific compilation approaches used and any supporting information that must be present in the team's final GitHub submission repository.

The two metrics that that you will be able to see for all teams for all challenges throughout the hacking period are

1. Operator norm distance:

$$d(U, \tilde{U}) = \min_{\phi \in [0, 2\pi)} \|U - e^{i\phi}\tilde{U}\|_{\mathrm{op}},$$

   where U is the exact unitary we want you to approximate and $\tilde{U}$ is your Clifford+T sequence. We take care of global phase calculation.

2. T-count. This is simply an integer representing the number of T and $T^\dagger$ gates in your submission.

Note that your circuits must only contain $\{H, T, T^\dagger, CNOT\}$ gates. Our grader will not accept the circuit otherwise. To convert some standard gates into H and T, please consult the next section.

4 Useful identities

Here are some formulas you might find useful:

- As you all remember from the workshop, Clifford gates preserve the Pauli group. Here are few examples of this:

$$\begin{aligned} HXH &= Z, \\ HZH &= X, \\ SXS^\dagger &= Y, \\ SYS^\dagger &= -X. \end{aligned}$$

- Let Hermitian M satisfy $M^2 = I$. Then,

$$e^{i\theta M} = \cos(\theta)I + i\sin(\theta)M$$

- Key to a lot of quantum simulation problems is the Trotter product formula:

$$e^{A+B} = \lim_{k\to\infty} (e^{A/k}e^{B/k})^k$$

  for arbitrary complex square matrices A and B.

References

[1] P. Selinger and N. J. Ross, *Exact and approximate synthesis of quantum circuits (newsynth package)*, https://www.mathstat.dal.ca/~selinger/newsynth/, Software package for quantum circuit synthesis, 2018.

[2]N. J. Ross and P. Selinger, "Optimal ancilla-free Clifford+T approximation of z-rotations", arXiv preprint arXiv:1403.2975 (2014).

[3]Superquantum, *rmsynth: high-performance Clifford+T circuit optimizer using phase-polynomial methods and punctured reed–muller decoding*, https://github.com/super-quantum/rmsynth, GitHub repository, Apache-2.0 license, 2026.

[4]M. Amy and M. Mosca, "T-count optimization and reed-muller codes", IEEE Transactions on Information Theory **65**, 4771–4784 (2019).

[5]M. Amy, D. Maslov, and M. Mosca, "Polynomial-time t-depth optimization of clifford+t circuits via matroid partitioning", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **33**, 1476–1489 (2014).

[6]F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. Brandao, D. A. Buell, et al., "Quantum supremacy using a programmable superconducting processor", Nature **574**, 505–510 (2019).

[7]L. Susskind, *Complexity and Gravity*, https://www.youtube.com/watch?v=6OXdhV5BOcY, Prospects in Theoretical Physics 2018: From Qubits to Spacetime (IAS lecture), July 2018.