IonQ & QCenter Quantum Challenge

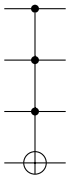
September 24, 2021 – October 26, 2021

1 Circuit Compilation and Optimization

In this challenge (Challenge 1), you will explore compilation techniques that are used to optimize quantum circuits, a sequence of quantum computational instructions executed on a quantum computer. Just as in conventional computing, optimizing compilation plays a vital role in harnessing the most power out of a quantum computer. Especially in near term machines, where each instruction cannot be implemented with perfect fidelity, inability to reduce the circuit footprints can become the bottleneck in running a quantum program.

1.1 Task 1 (Level 1)

Multi-control Toffoli gates are commonly used in quantum computing. The specific one we work with in this task has three control qubits and one target qubit. This is also known as the Controlled-Controlled-Controlled-NOT gate, or CCCNOT gate. It is commonly illustrated as below in circuit diagrams:



Black, filled circles denote the control qubits. $\oplus$ denotes the target qubit. Its action can be described as applying a inverting, NOT operation on the last (target) qubit if and only if the first three (control) qubits are in the state of $|1\rangle$.

The matrix representation of a CCCNOT gate is shown in Fig. 1. It is the same as an identity matrix, except for the block in the red square. We use a binary number system with the target being the least significant bit. This means the CCCNOT gate of our interest induces a transition between $|1110\rangle$ and $|1111\rangle$.

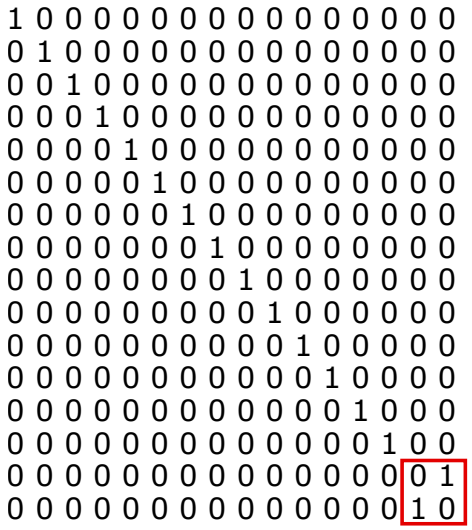


Figure 1: Matrix Representation of a CCCNOT gate. Columns and rows are ordered using big endianness, from $|0000\rangle$ to $|1111\rangle$ from left to right and from top to bottom, respectively.

Compose a circuit that implements the CCCNOT gate specified above using the universal gate set

$$\left\{ CNOT := \begin{pmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&0&1 \\ 0&0&1&0 \end{pmatrix}, R_x(\theta) := \begin{pmatrix} \cos(\theta/2) & -i\sin(\theta/2) \\ -i\sin(\theta/2) & \cos(\theta/2) \end{pmatrix}, \right.$$
$$\left. R_y(\theta) := \begin{pmatrix} \cos(\theta/2) & -\sin(\theta/2) \\ \sin(\theta/2) & \cos(\theta/2) \end{pmatrix}, R_z(\theta) := \begin{pmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{pmatrix} \right\}. \quad (1)$$

This universal gate set is generally supported on most quantum computing platforms. Your submission may be a piece of code that contains quantum circuits, generated with Qiskit, Cirq, or other tools of your choice. An ASCII file that contains a netlist representation of the quantum circuit would also suffice, as long as the notations used are unambiguously defined.

Submit, in addition, the quantum computational results performed on IonQ hardware as a part of the solution. The hardware runs should be performed as a good practice, just like in conventional computing, as a part of the test that aims to confirm the correctness of the computational programs. In this case, your quantum program is intended to implement a CCCNOT gate. Discuss what suite of tests, of your choice, are run and justify the choice, in the spirit of testing. Further, explain the reason why the observed outcome from the quantum computer may potentially differ from your expectations. Solutions will be fully credited as long as the circuit included generates the correct unitary,

up to a global phase, using the specified gate set and the test results reported therein suitably support the correctness with adequate justification.

1.2 Task 2 (Level 1)

Compose a circuit that implements the CCCNOT gate, specified above, using a universal gate set, this time made of $R_{xx}(\pi/2)$, $R_x(\theta)$, $R_y(\theta)$, $R_z(\theta)$. $R_{xx}(\pi/2)$ gate, also called an Ising gate, is the two qubit gate natively supported by the ion-trap quantum computers at IonQ. The matrix representation of $R_{xx}(\pi/2)$ gate is given below:

$$R_{xx}(\pi/2) := \begin{pmatrix} \frac{\sqrt{2}}{2} & 0 & 0 & -i\frac{\sqrt{2}}{2} \\ 0 & \frac{\sqrt{2}}{2} & -i\frac{\sqrt{2}}{2} & 0 \\ 0 & -i\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ -i\frac{\sqrt{2}}{2} & 0 & 0 & \frac{\sqrt{2}}{2} \end{pmatrix}. \tag{2}$$

The physics phenomena we exploit to implement a gate between two qubits on a trapped-ion quantum computer result in the unitary of the form shown above. For forward-looking and practical reasons, say quantum error correction or high-accuracy calibration, we fix the argument θ of $R_{xx}(\theta)$ gate to be $\pi/2$. Note, it is often times more straightforward to consider hardware-specific optimization, if quantum circuits are synthesized directly with native gates.

Please submit your solution in the same format as in **Task 1**. Again, it may be a piece of code that contains quantum circuits, generated with Qiskit, Cirq, or other tools of your choice. An ASCII file that contains a netlist representation of the quantum circuit would also suffice, as long as the notations used are unambiguously defined. Please further submit the hardware test runs, as in **Task 1** and confirm the testing strategy's validity by comparing the test results with those of **Task 1**. Solutions will be fully credited as long as the circuit included generates the correct unitary, up to a global phase, using the specified gate set and the test results reported therein suitably support the correctness of the circuit.

1.3 Task 3 (Level 2)

What is worth more costs more. Entanglement is the key to the magic of quantum computing. Two-qubit gates, such as R_{xx} gates, as our tools to generate entanglement, cause more errors than single-qubit gates, such as R_z gates, which do not generate entanglement between qubits. So, an important goal for circuit optimization is then to minimize the two-qubit gate count, otherwise known as circuit size. Optimize the circuits you composed in **Tasks 1** and **2** to minimize the number of two-qubit gates used, while sticking with the same gate sets used, respectively. With the amount of two-qubit gates kept at minimum, you should also minimize the number of single-qubit rotation, except $R_z(\theta)$. $R_z(\theta)$ gates are treated differently than other single-qubit rotation gates, because they, on IonQ's machines, are implemented by advancing/retarding the

phase of electronic control signals. This is in contrast to R_x or R_y gates which involve physically driving atomic transition with laser beams. Thus, R_z gates are implemented with higher fidelity than R_x or R_y gates.

Submitted solutions, again in the same format as before, will be scored according to the number N_2 of two-qubit gates and the number N_1 of single-qubit gates (except $R_z(\theta)$). Specifically, the following formula will be used as the circuit cost: $N_2 + 0.1 \cdot N_1$. The point awarded is computed as the cost obtained minus 15. The submitted, optimized circuits are subject to correctness verification. Explicit hardware test results, if submitted, will be granted a bonus point.

1.4 Task 4 (Level 2)

In conventional computing, parallel operations are heavily used to reduce the run time of a program. Though not supported in all of today's quantum computers, parallel quantum operations are a critical component for efficient quantum computing. Optimize the circuit that implements a CCCCNOT gate – note this is *one more control* than a CCCNOT gate – using the gate sets used in **Tasks 1** and **2**, this time minimizing the circuit depth. The depth of the circuit here is defined as the number of two-qubit gate layers needed to complete the program. For this exercise, ignore, for the purpose of counting the layers, single-qubit gates.

Please submit your solutions in formats that clearly delineate the different two-qubit gate layers. All single-qubit gates should appear either at the very beginning or the very end of the circuit or in between two adjacent two-qubit gate layers. A two-qubit gate layer may contain as many two-qubit gates from the gate sets used in **Tasks 1** and **2**, respectively, so long as all of the two-qubit gates that appear in the given layer can be implemented simultaneously, in parallel. This can be checked via commutation between two-qubit gates, i.e., all two-qubit gates used in the given layer commute. Solutions will be scored according to the number of the two-qubit gate layers. Incorrect solutions will be given zero score.

1.5 Task 5 (Level 3)

Use a discrete quantum gate set $\{R_{xx}(\pi/2), R_x(\pi/2), R_z(\pi), R_z(\pm\pi/2), R_z(\pi/4)\}$ to generate an unitary V that implements the following unitary U:

$$U = \frac{1}{\sqrt{8}} \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & e^{i\pi/4} & i & e^{i3\pi/4} & -1 & e^{i5\pi/4} & -i & e^{i7\pi/4} \\ 1 & i & -1 & -i & 1 & -i & -1 & -i \\ 1 & e^{i3\pi/4} & -i & e^{i\pi/4} & -1 & e^{i7\pi/4} & i & e^{i5\pi/4} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & e^{i5\pi/4} & i & e^{i7\pi/4} & -1 & e^{i\pi/4} & -i & e^{i3\pi/4} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & e^{i7\pi/4} & -i & e^{i5\pi/4} & -1 & e^{i3\pi/4} & i & e^{i\pi/4} \end{pmatrix}. \tag{3}$$

You may use measurements at your discretion. Your goal is to use a minimum number of $R_z(\pi/4)$ gates. This choice is motivated by the fault-tolerant, error-corrected quantum computing, wherein $\{R_{xx}(\pi/2), R_x(\pi/2), R_z(\pi), R_z(\pm\pi/2)\}$ gates are inexpensive to implement, whereas $R_z(\pi/4)$ gates are expensive to implement.

Please submit your solution in formats that clearly lay out your circuit. The primary goal will be used to rank the solutions. If there is a tie, the number of qubits used (circuit width), the number of two-qubit gates used, the number of two-qubit gate layers required (meaning your format needs to clearly delineate the two-qubit gate layers, see **Task 4** above), and the total number of gates used will be used to break the tie, in that order.

2 Ion Trap Design

One prominent challenge across the field of quantum computing is how to scale existing qubit architectures. You will be asked to design and simulate basic features of a surface ion trap that can support parallel strings of ions. Most linear surface traps (read below for their brief introduction) that are currently operated have one confining well, formed by a radio-frequency (RF) potential, that runs along the linear axis of the trap. Within this well, one traps ions, forming a linear chain, also along the axis. Creating trapping potentials with more than one well is then one of many possible approaches for scaling ion-trap based quantum computers.

Background – RF Ion Traps create confinement of charged particles by rapidly switching between two electric potential configurations. The two electric potential configurations (landscapes) are created by applying suitably chosen positive and negative voltages to specially arranged electrodes. The switching can be achieved for instance by considering an RF voltage, i.e., an AC voltage with ∼1-100MHz in a typical case. More concretely, the goal here is to rapidly change between two quadrupole potentials (saddle potentials). So long as you switch between the two potentials fast enough (and account for certain electrode / trap parameter requirements that we will not focus on), particles in the center of the trap will be confined. For this problem, we will focus on linear Paul traps. There are other types of ion traps, but linear Paul traps (and surface traps that mimic these principals) are the most common traps used in trapped-ion quantum computers. Shown in Fig. 2 is a cross section diagram of a linear Paul trap. The 4 electrodes shown, labeled RF or GND, extend out very far in $\pm\ \hat{x}$ direction.

Let's consider the first potential needed to confine the ions: positive polarity of the RF voltage. In this case, a positive ion will be pushed away from the two RF electrodes and, so long as the ion is near $z = 0$, it will be pushed in $\hat{y}$ toward the center. However, the positive ion can escape in $\hat{z}$ because it is attracted to the GND electrodes! So, before it can escape too far, we switch to a negative voltage on the RF electrodes and now a positive ion will be pulled in $\hat{z}$ away from the GND electrodes back to the center. (For visualization, you may find the following link helpful: https://www.youtube.com/watch?v=XTJznUkAmIY.) In general, ions need to be close to the center of the trap and have low enough energy to be confined. There is an approximation that we will look at later in this challenge that allows us to examine the time-averaged pseudopotential, induced by the two fast-switching potentials.

We've looked at the confinement in the radial $\hat{y}\hat{z}$ plane. But, what about in the axial direction ($\hat{x}$)? Static DC endcap electrodes are used to push the ions back to the middle of the trap in $\hat{x}$. You can think of the 4 rods in the diagram creating a long tube of confinement that runs along the long axis ($\hat{x}$) of the trap. The DC endcaps essentially pinch this tube off at the ends. While this trap can be very consistent and confine large numbers of ions, precisely controlling the location of ions can be challenging.

As a helpful exercise to prove useful later, please research Paul traps and

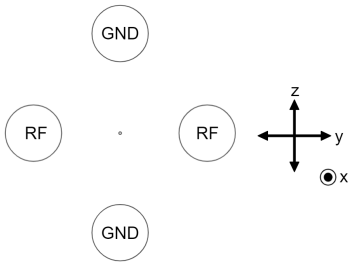


Figure 2: Cross section diagram of a linear Paul trap. 4 electrodes (often circular for ease of manufacturing) create an approximate quadrupole potential local to the trapping region. Two DC endcaps (not shown) create the confinement in the $\hat{x}$ direction. The units in this diagram are consistent throughout the entirety of this problem.

build an understanding of the basic principals and how they work. A few important concepts that may not be immediately clear may include 'optical access', 'shuttling', and 'stability parameters'.

Optical access is the angular clearance needed for either laser addressing of the ions or imaging of the ions. In the first case, lasers are used to manipulate the electronic state of the ions. This manipulation can be used to cool the ions (Doppler cooling, EIT cooling, sideband cooling), to perform logic gates, and more. Many of the significant ion-trap quantum computing processes are mediated by lasers. Sometimes the focus of a given laser can be tens or hundreds of microns wide, and other times it has to be only be a few microns wide. In order to focus a laser very tightly, a shorter focal length lens is needed and a larger cone of angular clearance is needed to get that laser light to the ion(s). The same principal applies for collecting photons from the ions (for state detection of qubits), the more unobstructed angular range available for a lens to cover, the more photons that can be collected. In the case of surface traps, lasers should not hit the trap surface. Ensuring that the ions are high enough above the surface, and that the trap is thin enough in key areas, allows enough optical access for very tight laser focuses (notice the bow-tie shape of many contemporary surface traps!).

Shuttling is the process of moving ions around in a trap. This is typically achieved by using 'DC' electrodes (N.B. Not to be confused with the end-cap electrodes). DC electrodes can be used to 'pinch' the tube of confinement that runs along the axis of the trap. Think of the electrodes as some small electrodes placed at strategic locations along the trap axis to provide a potential barrier at the locations for instance. By carefully varying the voltages on these electrodes (much slower than the RF, so we consider them 'DC') one can actually move the pinch locations around and accordingly move the ions around.

Stability parameters are basic operating guidelines for the trap. Specifically,

a and q are widely used in the literature. They are derived from equations of motion for an ion trap system (which should be easy to find by looking up quadrupole ion trap'). Not all operating conditions will be stable and confine ions. Indeed, you will be asked to work within a provided stability parameter range for the final task of this challenge problem.

2.1 Task 1 (Level 1): Surface Trap Background

Surface ion traps project 3D RF electrode geometries (oftentimes something similar to a 4-rod linear ion trap) onto a 2D plane. They can create largely the same type of confining radial potentials while leveraging microfabrication to also incorporate static DC control electrodes. These DC control electrodes are much closer to the ions and this allows for higher resolution control of ions along the linear axis (conventionally referred to as the axial direction).

A powerful and relatively simple tool for the design of these devices is the Gapless Plane Approximation [**https://arxiv.org/pdf/0808.1623.pdf**]. This reference covers the approximation as well as other electrostatic methods for ion trap design and analysis. Within the context of surface traps, this analytic approach is used to calculate the electric field imposed by an arbitrary flat region (held at voltage V) among an otherwise grounded infinite surface. The solution for each electrode of interest on a surface trap can be superimposed to relatively quickly simulate the electric field and potential landscape for a surface trap. We will focus only on simulation of the RF electrodes.

2.1.1

Using notation from Figure 6 in the above reference (please carefully read the reference), provide a diagram of the multipole electrode parametrization from cylindrical coordinates to a flat 2D surface at x=0 (in the $\hat{y}\hat{z}$ plane). Do so for a symmetric 2nd-order multipole configuration ($\Theta_0 = \pi/2$) with the following angular coverage $\Theta_w = \pi/4$ and an ion height (radius of the circular diagram) of 50um.

2.1.2

Now keep the ion height and Θ_w the same, but diagram the projection of an asymmetric configuration with $\Theta_0 = \pi/4$.

2.1.3

In a few sentences, explain how surface traps are able to create similar radial electric fields and potentials as 3D linear traps. Also elaborate on the effect of the projection and practical limitations of multipole placement in the radial configuration.

2.2 Task 2 (Level 2): RF Electrode Simulation

Utilize the gapless plane approximation and Biot-Savart law to simulate the electric field and potential in the ($x = 0$, $\hat{y}\hat{z}$) plane for the geometry from section 2.1.1. This method allows us to simulate electrostatic potentials, but when we apply an RF voltage to electrodes it is not static. This simulation is a snapshot of what the RF potential would look like at a given point in time. We will explore how to interpret these results in an electrodynamic context later.

Model the electrodes to extend ±1500um in $\hat{x}$. Place a 1V potential on your electrodes (you can scale your field and potential calculations after the fact to reflect any applied voltage). Use Biot-Savart law to calculate the field (and potential) at any given point. To create a plot, you need to select a range of points within the ($x = 0$, $\hat{y}\hat{z}$) plane. For this exercise, take: $10 < z < 80$ and $-100 < y < 100$. Choose any step size that you feel is appropriate for the region. Provide a contour plot of the static electric potential. Also, provide the calculated value of the potential at the points (0,10,50) and (0,0,50). Your result should resemble Figure 3 (please also plot axes to scale):

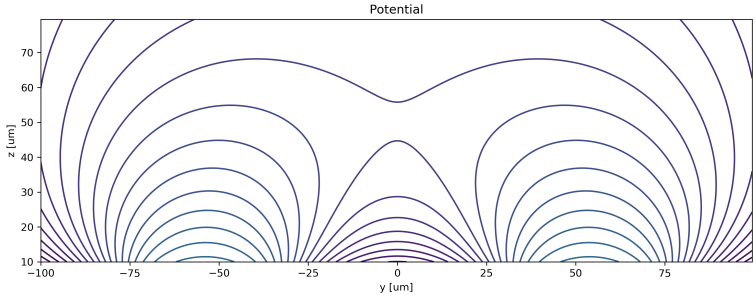


Figure 3: Example of the solution to section 2.2

Hint: You will need to calculate a finite Biot-Savart integral for this problem. The case of infinitely long electrodes (in $\hat{x}$) is a much simpler problem that yields nearly the same result. Use this as a check! We have enforced electrode symmetry in $\hat{x}$ for the finite problem, how might this help you check or debug your code? In general, symmetry is a useful tool in ion trap design!

*ALL CODE SUBMISSIONS (FOR THIS TASK AND FOLLOWING TASKS) SHOULD INCLUDE YOUR SIMULATION CODE WRITTEN IN EITHER PYTHON OR MATLAB. IT SHOULD BE STRAIGHTFORWARD FOR JUDGES TO PASTE AND RUN (USING COMMON PACKAGES OR LIBRARIES *SUCH AS NUMPY OR MATPLOTLIB*), PLEASE HAVE YOUR CODE SAVE ANY PLOTS/FIGURES IN PNG FORMAT.

2.3 Task 3 (Level 2): Ponderomotive Potential and Pseduopotential

By now, we have simulated the static potential for a 2nd-order multipole electrode configuration. As we touched on in the introductory material, the potential that results from this static simulation is not confining. At our targeted ion position of (0,0,50) the potential landscape looks like a saddle point. However, so long as the RF electric field changes significantly faster than anything else in the system, the behavior of an ion in an RF field can be approximately described by the ponderomotive potential:

$$U_p(\boldsymbol{r}) = \frac{e^2}{4M\Omega^2}|\boldsymbol{E}_{RF}(\boldsymbol{r})|^2 \tag{4}$$

where e is the charge of your ion, M is the mass of the ion, and Ω is the frequency of your RF drive.

This approximation is only particularly valid near the saddle points in the potential (also called RF null points) and the resulting, locally accurate potential is called the pseudopotential. Justify the use of this approximation and explain why it is only valid near the RF null. After that, apply this approximation to your electric field calculations to yield a pseudopotential contour plot (in units of eV). Assume that you are trapping ^{171}Yb and have an RF drive at 30MHz with an amplitude of 150V. Also provide the pseudopotential value (in eV) at the point (0,25,40) as well as the designed trapping point.

Note: this approximation is not valid throughout the entire region of your potential plot, but for this part of the problem apply it to the whole region. Often times, pseudopotential plots will simply be cut-off at a certain numerical value such as 0.1eV.

2.4 Task 4 (Level 3): Challenge Trap Design

Now that we have built the toolset to quickly evaluate electrode geometries, you will design the RF and DC/ground (GND) structure for a trap that can support two *parallel* strings of ions. That's right, so far, we have worked on a linear geometry only, and we would like to now go above and beyond to push for a multi-chain architecture, where the chains are parallel to one another. This trap must be able to support BOTH one and two strings of ions. Some designs may achieve this by shuttling chains between zones. Other approaches include using RF rails that can support different voltages. You may choose how to achieve this functionality (there are some publications that cover designs that do this, go look for them via arxiv or iopscience!). This is where creativity is valuable, judges are particularly interested in unique ways to achieve this functionality. You are free to assume that you have a reasonable ability to move (shuttle) ions around the trap using DC electrodes. You do not have to model or design these DC electrodes. Provide relevant models or drawings of your trap as well

as potential and pseudopotential simulations for your design operating in both ion-string configurations.

There are a number of constraints imposed on your design:

- Ion height above the surface of the trap must between 50um and 200um
- Numerical Aperture (in a general optics definition, not laser physics) for radial laser addressing of ions in the 2 string configuration must be at least 0.1 (*Hint: this sets an effective maximum width of your trap in the radial direction*)
- Maximum RF voltage amplitude is 250V
- Use a single RF frequency between 10 and 100MHz
- The 2 string configuration must have a well separation between 40 and 100um
- Stability: assume there is no DC potential at the ion and use an linear ion trap's stability parameter for an oscillating field, q, to restrict operation within a stability range of $0.1 < q_{y,z} < 0.3$ for your *single-chain configuration*:

  $q_{y,z} = \frac{2eV_{RF}}{mr_0^2\Omega^2}$

  where e is electron charge, V_{RF} is RF voltage amplitude, m is ion mass, r_0 is the effective trap radius (approximate this to be double your ion height, this is a value that is beyond the scope of this question but you should be able to adjust other operational parameters to get within this range), and Ω is radial RF frequency. All ion traps must operate within a stable regime to hold ions and the range given here is a common one used by surface traps.

Finally, elaborate on your design and discuss the trade-offs and considerations made when designing the trap. How do your choices affect practical operation of the trap? How might that impact choice of ion species, DC control, laser addressing, imaging of the ions, fabrication, etc? *This is designed to make you think about the problem of designing an ion trap. In practice, there are many considerations to make when designing these devices. The ability to identify and balance those is valuable!*

3 Simulating Hubbard Model with Quantum Computers

In this challenge problem, you will try to use a quantum computer to simulate the time evolution of a Hubbard model system. The Hubbard model is a very simple model system but contains a rich variety of physics. It is believed that simulating the Hubbard model could lead to a better understanding of strongly correlated systems, such as high-temperature superconductivity.

Specifically, we consider the smallest Hubbard model that contains 2 sites. The Hamiltonian of the system is

$$H = -t\left(a_{1\alpha}^{\dagger}a_{2\alpha} + a_{2\alpha}^{\dagger}a_{1\alpha} + a_{1\beta}^{\dagger}a_{2\beta} + a_{2\beta}^{\dagger}a_{1\beta}\right) + U\left(a_{1\alpha}^{\dagger}a_{1\alpha}a_{1\beta}^{\dagger}a_{1\beta} + a_{2\alpha}^{\dagger}a_{2\alpha}a_{2\beta}^{\dagger}a_{2\beta}\right) \tag{5}$$

in which $a_i^{\dagger}, a_i$ are creation and annihilation operators on site i. α and β indicate spin-up and spin-down electrons.

As you can see, the Hubbard model contains two parts. The first part contains the "hopping" terms, in which electrons are allowed to hop from site 1 to site 2, and vice versa. This models the kinetic energy of the electrons. The second part is the on-site repulsion term, in which if a site is occupied by two electrons, it will induce a energy penalty (repulsion). The scale of the kinetic energy and the repulsion energy is quantified by t and U (t and U are both non-negative). As you could imagine, as we change t and U, the physics will be very different. For example, in the limit where $t \ll U$, the electrons will be able to hop freely without any energy penalty. This is sometimes referred to as the free electron model.

For the 2-site Hubbard model with 2 electrons, there are 4 orbitals in total. Suppose we denote the first two orbitals for the alpha electrons and the second two orbitals for the beta electrons, then the order of orbitals are

$$1\alpha, 2\alpha, 1\beta, 2\beta \tag{6}$$

The state $|0101\rangle$ indicates that the first site is doubly occupied. State $|1010\rangle$ indicates that the second site is doubly occupied.

3.1 Task 1 (Level 1)

In order to simulate the Hubbard model on a quantum computer, we first need to transform the Hamiltonian into qubit space. In other words, we need to write the creation and the annihilation operators in terms of Pauli matrices. One way to do so is the Jordan-Wigner transformation (JWT). By using the JWT, write the Hamiltonian in terms of Pauli matrices. See this link for a more detailed explanation of the JWT technique.

3.2 Task 2 (Level 1)

For the following three scenarios, suitably pick t and U and diagonalize the Hamiltonian and get the ground state eigenvectors, and see if you can interpret

the physical meaning of the eigenvectors.

- $U \ll t$.
- $U = t \neq 0$.
- $U \gg t$.

3.3 Task 3 (Level 2)

In the JWT'ed Hamiltonian, you will see terms like XY and $XXXY$ (and other similar terms, the XY and $XXXY$ are just examples, and it may or may not show up in the JWT'ed Hamiltonian). Since we want to implement the time evolution operator $e^{-iH\tau}$, where τ is the evolution time, the fundamental building block is to implement $e^{-i\tau XY}$ and $e^{-i\tau XXXY}$ for arbitrary τ. Draw the quantum circuit that implements $e^{-i\tau XY}$ and $e^{-i\tau XXXY}$.

3.4 Task 4 (Level 2)

As you might have seen, the JWT'ed Hamiltonian looks like

$$H = \sum_i g_i P_i \tag{7}$$

in which P_i is a tensor product of Pauli matrices (XY, $XXXY$, YX, $ZZ \cdots$). Using the first-order Trotter approximation, the propagator becomes

$$e^{-iH\tau} = \prod_i e^{-ig_i P_i \tau}. \tag{8}$$

Now draw the circuit that implements the full propagator.

3.5 Task 5 (Level 2)

Let's now set $U = 0$ and $t = 0.1$, as you will see, this greatly reduces the number of terms in the Hamiltonian and results in a much simpler circuit.

First, prepare the initial state of the quantum computer to $|0101\rangle$. Then apply $e^{-iH\tau}$ for different evolution time τ, and then measure all four qubits. Keep in mind that it might be necessary to break τ to N time steps and apply $e^{-iH\tau/N}$ for N times in order to reduce the Trotterization error. For a single qubit, if the measurement gives 1, then the orbital is occupied (occupation number = 1), and if the measurement gives 0, then the orbital is unoccupied (occupation number = 0). Compute the average occupation number for all 4 orbitals as a function of τ. Compare simulator and quantum hardware results.

3.6 Task 6 (Level 3)

Let's now consider $U > 0$ and $t = 0.1$. Again, first prepares the initial qubit in the $|0101\rangle$ state, and then apply $e^{-iH\tau}$ for different τ, and then measure all four qubits and compute the same expectation value as of last question. Compare simulator and quantum hardware results. Locate a critical U value, based on the results obtained from the simulator and the quantum hardware, at which the behavior of the expectation value as a function of τ transitions dramatically. Describe clearly the methodology used to find the critical value, including the used algorithm to search for the value and the criteria used to determine the transition. Justify your choice of maximal value of τ used.