Compiling OpenQASM on a Neutral Atom Quantum Computer

Thesis Proposal

Author: Xavier Spronken (i6225376)

x.spronken@student.maastrichtuniversity.nl

Supervisor: Claire Blackman

Word Count:

Academic Year 2022-23

Contents

1 Introduction

1.1 Quantum Computers

While quantum computers can be used for analog computation or simulating quantum systems, it is also possible to make them perform quantum information processing (QIP). The carriers of information in QIP are qubits. These differ from classical qubits in that they make use of quantum superposition. Thus a qubit's state (either $|1\rangle$ or $|0\rangle$) can only be known precisely upon measurement. Before this, the qubit is said to be in a superposition of both states, where the probability of measuring either one of those states is P($|1\rangle$) + P($|0\rangle$) = 1. Mathematicaly a qubit is represented as $|a\rangle = \alpha|1\rangle + \beta|0\rangle = \begin{bmatrix}\alpha \\ \beta\end{bmatrix}$, where $\alpha^2 + \beta^2 = P(|1\rangle) + P(|0\rangle) = 1$

An intuitive method to view the state of a qubit before measuring is through the Bloch sphere. In this representation, the $|1\rangle$ state and $|0\rangle$ state are respectively set to the "North Pole" and "South Pole" of the sphere. In addition to the probability of getting either state upon measurement, the Bloch sphere shows the phase of the qubit thanks to the latitudinal direction of the arrow in the sphere.

Before qubit measurement, it is possible to change the state of a qubit through so-called quantum logic gates, these are the quantum computing equivalents of logic gates that make up classical electronic circuits. When applied to a qubit, a quantum gate will set it to a specific superposition of states. This corresponds to a specific rotation of the arrow on the Bloch sphere. Since a gate corresponds to a rotation a does not place the qubit in a specific state, the resulting superposition depends both on the gate and the state of the qubit before the gate was applied. Some well-known quantum gates are the Pauli X Y and Z gates, each corresponding to a rotation of 180 degrees around the x, y and z axes of the Bloch sphere. Gates are mathematically represented as matrices that can be applied to the qubit state vector. The Pauli X gate for example would be $\begin{bmatrix}0 & 1 \\ 1 & 0\end{bmatrix}$

Quantum gates can be applied to single or multiple qubits at the same time, in general, these are control gates, which will only apply a rotation to qubit 1 if qubit 2 is already in a certain state. While there are multiple different types of gates, it is possible to create a universal set. This is a set of logic gates that can be combined to create any other gate. The set of arbitrary rotation gates $R_x(\theta), R_y(\theta), R_z(\theta)$, combined with a phase shift gate $P(\varphi)$ and a control-X gate would be such a set.

1.2 QASM

1.3 Pasqal's Neutral Atom Quantum Computer

1.4 QASM on NAQC

2 Project Objective

The objective of this thesis is to create a QASM compiler for Pasqal's neutral atom quantum computer. This is achievable in four steps, the first is to parse the QASM code to retrieve the number of qubits used and which gates are applied to them. Once the code has been parsed, a register of qubits has to be created that satisfies the constraints imposed by the multi-qubit gates. Following the creation of the register the gates can be translated into adequate optical pulses. Finally, the compiled quantum circuit can be tested through simulation, or on a real quantum device if available.

3 Methodology

3.1 Tools

The compiler will be written using the Python language. This choice was made due to the popularity of the language in the scientific community. Thanks to this there are many useful and well-documented toolbox libraries available to help simplify the project, three of these tools will be of particular importance. Firstly OpenQASM3 provides a Python library with parsing methods which will be used to retrieve all the required information from the user-input code. The NetworkX library will provide the optimization tools necessary to generate an adequate array of qubits to fill the register, based on the constraints imposed by the different multi-qubit gates in the input code. And finally, the Pulser Library, maintained by Pasqal, will allow for the simulation of their Quantum device. This library will be used to simulate the quantum computer the compiler will be written for. It allows the creation of different qubit registers and optical pulses and simulates the outcome of a quantum circuit on Pasqal's computer. If the simulations are successful, there may also be a possibility to test the compiler on Pasqal's device rather than simply using the simulations.

Some restrictions need to be taken into account regarding circuit simulation, notably that the hardware used to run the code will be a personal laptop. This will limit the complexity of quantum circuits that can be tested to about 5-10 qubits as anything greater would take large amounts of time to simulate completion of this project, it can be quite slow when it comes to computation

3.2 Approach

To maximize the chances of producing a functional compiler by the end of the project, a specific approach has to be used. Instead of trying to make each step (Parsing, Optimizing, Translating, Simulating), a full compilation pipeline will be created for increasingly complex QASM algorithms. This means that the project will start by creating a simple parser, optimizer, translator and simulator which will work with one or two qubits and a small number of select single-qubit gates. As the project progresses, the pipeline will be modified to handle more complex algorithms, which involve more qubits and multi-qubit gates. Once the compiler can handle multi-qubit gates, the work will shift to translating as many gates as possible into their corresponding optical pulses and making sure the compiler can still work with these new gates.

3.3 Ethical considerations

The software tools used to produce this project are all free and published under open-source licenses, and the different resources used will be cited adequately. The code for the compiler will be stored in a GitHub repository and made public under an open-source license once it has acquired enough substance to be usable or useful to other researchers. Thus no further ethical considerations are thought to be required regarding the methodological approaches of this project.

3.4 Time planning

As mentioned above in the approach, time planning will be structured towards producing a limited but functional piece of software if the project is held up or problems occur. Thus the project will be divided into three steps. First, the basic infrastructure will be created for simple algorithms that only involve single-qubit gates. Since these gates do not impose any constraints on how qubits should be stored in the registry, optimization will be handled in the next step. This second step involves adapting the compiler to be able to handle multi-qubit gates and the constraints that accompany them. Work on parsing should not be necessary if it was correctly written in step 1. Once the multi-qubit compiler is achieved, focus can be brought to translating all the remaining OpenQASM gates to optical pulses. At the end of each of these steps, the compiler will be tested using a known quantum algorithm and the results will be formatted and integrated into the final thesis paper. The timeline of this project is summarized in the Gantt chart below.

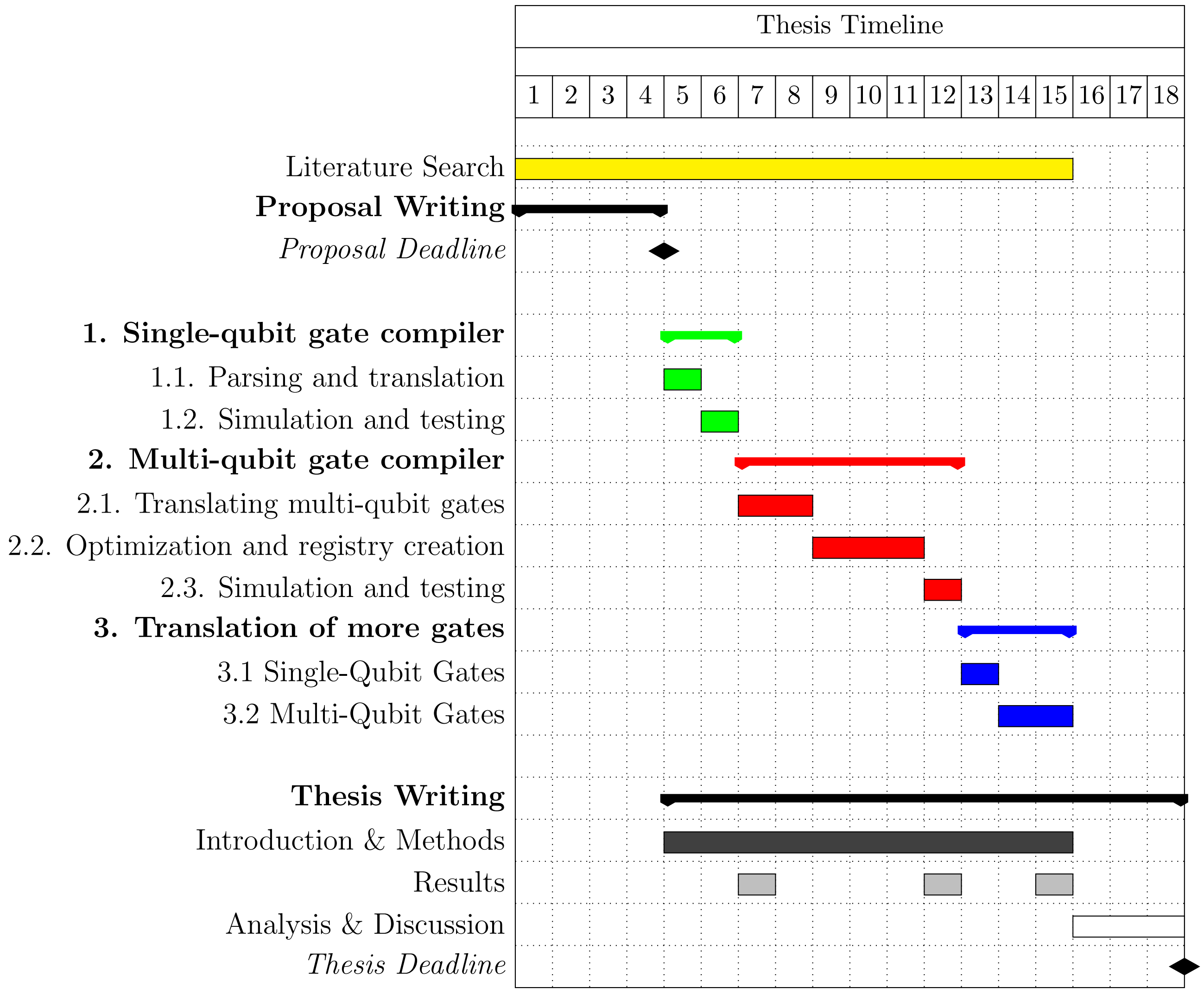

Figure 1: Gantt chart of the thesis timeline

4 Project Impact

4.1 Impact

Considering all the different approaches to implementing quantum computing, OpenQASM's position as a hardware-agnostic language may set it up to be a universal layer that all higher-level quantum software languages could translate to and work on any quantum computer. This would allow quantum algorithms, such as Grover's algorithm, to work on any quantum computer, as long as it was written in OpenQASM. However, for this to be the case OpenQASM compilers need to exist for all the different types of quantum computers. This isn't the case yet for neutral atom quantum computers, or at least there is no open-source version available to the public. Thus this project aims to be the first open-source OpenQASM compiler for neutral atom quantum computers, more specifically Pasqal's device.

4.2 Ethical considerations

The main ethical concern for this project is that, while the project would produce an open-source solution to compiling OpenQASM, the main beneficiary of this code would be Pasqal, a for-profit start-up. However, they seem to be committed to open-sourcing most of their code as is the case with Pulser. Thus a compiler such as this one would help develop Pasqal's quantum technology further, while also keeping a majority of the knowledge open to the public and competition.

References