MULT90063 Introduction to Quantum Computing

Lab Session 9

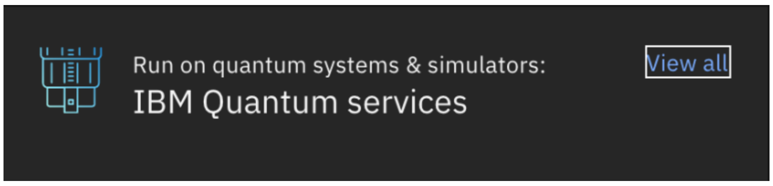
9.1 Introduction

Welcome to Lab 9 of MULT90063 Introduction to Quantum Computing. The purpose of this lab session is to program and run circuits in the IBM Quantum Experience.

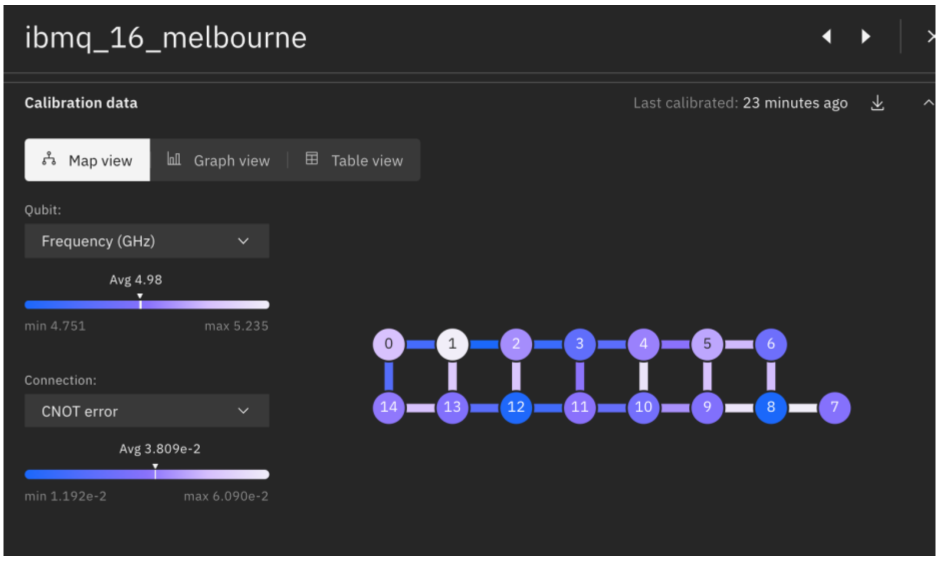
9.2 IBM Quantum Experience

The IBM Quantum Experience allows you program and run circuits on actual hardware, as described in the lectures. You may work in pairs and/or groups to pool run units and avoid long wait times. Before using your "experiment units" check your circuits by running the simulator and/or the QUI. Note the IBM simulator is not like the QUI – it runs a certain number of times to gather the statistics to give you the (approximate) probability distribution at the measurement (unlike the QUI you have to put in measurement gates to get results for both the IBM devices and the simulator). The simulator runs the code noise/error free (just as in the QUI, unless you put in error gates deliberately), but make sure to set the number of simulator runs to at least 1000 so the results aren't skewed by low statistics.

Hint: if you're programming a lot of gates switch to the QASM editor and then back to the composer to check. Before you run any experiment, be sure to save the circuit. If there is a cached version of your circuit, which you have used before, re-use it as waiting for machines may take time!


Run on quantum systems & simulators:
IBM Quantum services
View all

Exercise 9.2.1 When you access the system, if you click on "View all" IBM quantum services (as above) you will see several devices which you can use. Clicking on any one of them brings up details about that device. For example, here is a summary of the IBM Melbourne device:


ibmq_16_melbourne

Calibration data

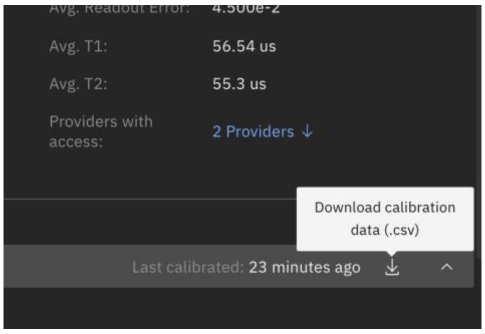
Last calibrated: 23 minutes ago

Map view | Graph view | Table view

Qubit:
Frequency (GHz)
Avg 4.98
min 4.751 max 5.235

Connection:
CNOT error
Avg 3.809e-2
min 1.192e-2 max 6.090e-2

Choose a device which you will use during this session, and examine the details of error rates on CNOT and single qubit gates. Note that it also indicates which CNOT gates are allowed on this device. Record this as a screenshot for reference.

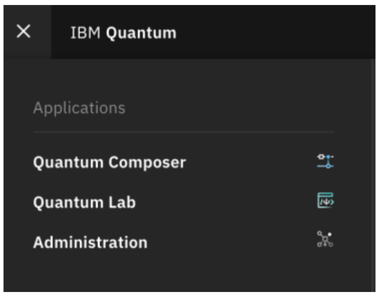


Avg. Readout Error:	4.500e-2
Avg. T1:	56.54 us
Avg. T2:	55.3 us
Providers with access:	2 Providers ↓

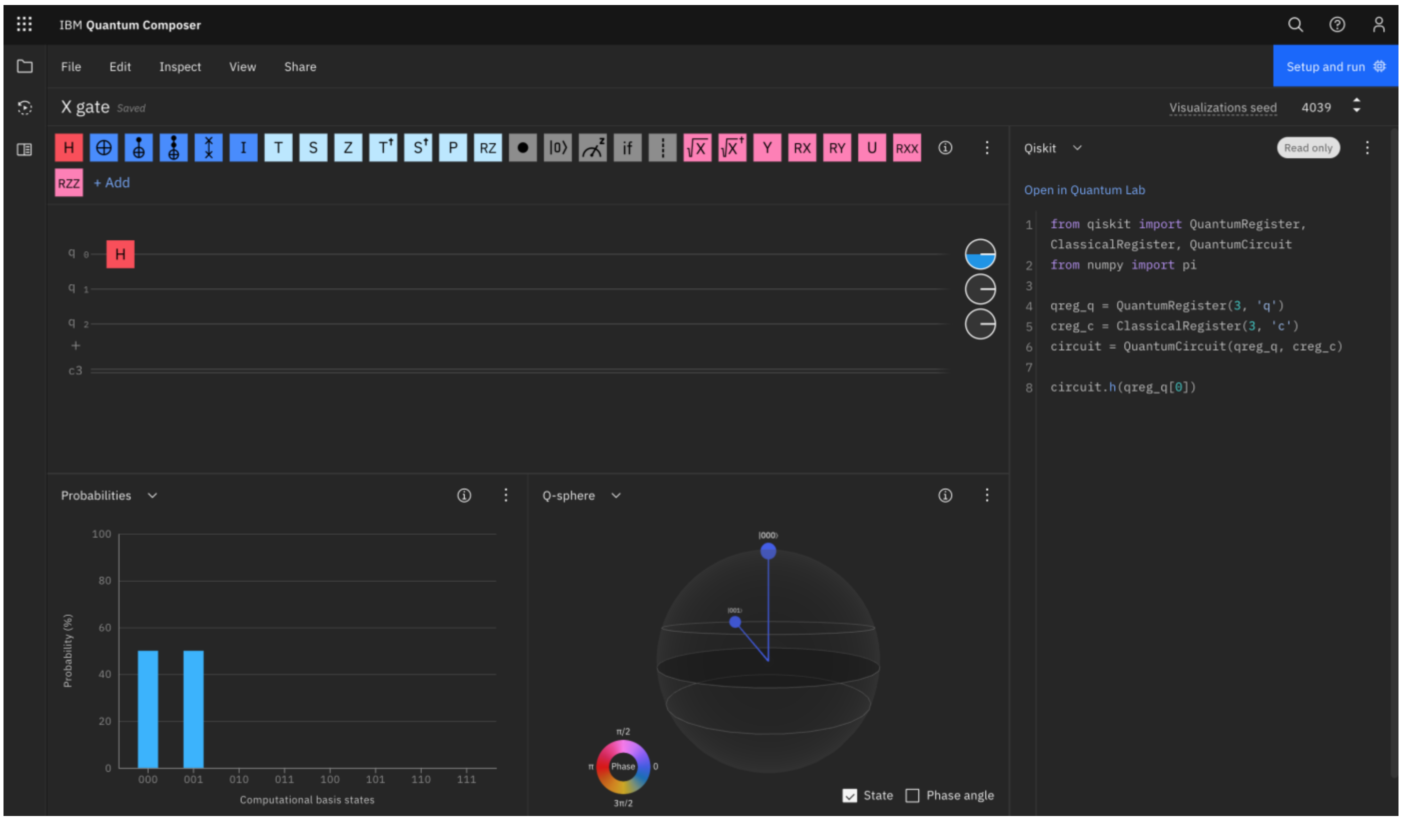
Download calibration data (.csv)
Last calibrated: 23 minutes ago

Additional details can be found by clicking on the “Download calibration data” button. Do this, and examine the file that you download. The T1 and T2 times quoted are an indication of how long qubits remain faithful to their state (i.e. stay coherent). T1 is related to how long it takes for say a $|1\rangle$ state to decay into a $|0\rangle$ state through natural relaxation processes (akin to an “X” error). T2 is related to how long it takes for the phase to change by 180º (akin to a “Z” error). We can program the computer and observe the effect of these processes (after repeating many times to build up the probability distributions).

9.3 Superposition states

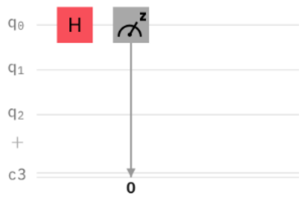


We will make use of the Circuit Composer functionality in the IBM experience website. Click on the “Circuit Composer” icon in the left sidebar.



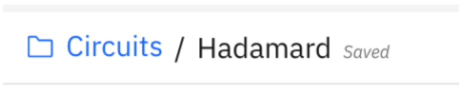
```
from qiskit import QuantumRegister,
ClassicalRegister, QuantumCircuit
from numpy import pi

qreg_q = QuantumRegister(3, 'q')
creg_c = ClassicalRegister(3, 'c')
circuit = QuantumCircuit(qreg_q, creg_c)

circuit.h(qreg_q[0])
```

Exercise 9.3.1 Program an Hadamard gate to create an equal superposition state. Examine the different animations and information screens available to you, including the "Q-Sphere", measurement probabilities, and the code (both Qiskit and QASM) for the circuit. Be sure to also add a measurement gate to measure the outcome of the circuit. Your circuit should now look like this:



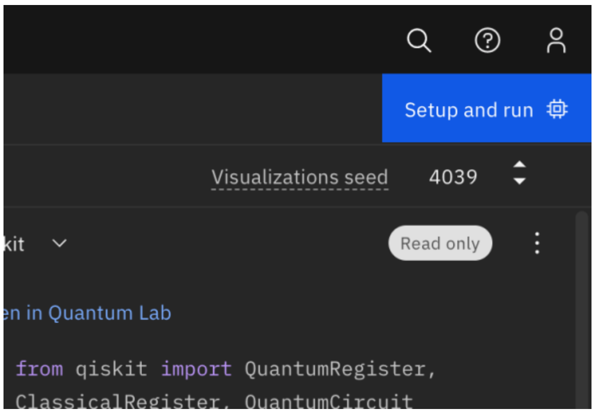
Note that you can add or remove qubits, by clicking on the qubit's name (q0, q1, q2…) or the plus sign.

Choose a suitable filename of "Hadamard" (or similar) by clicking on the filename in the top left hand corner.



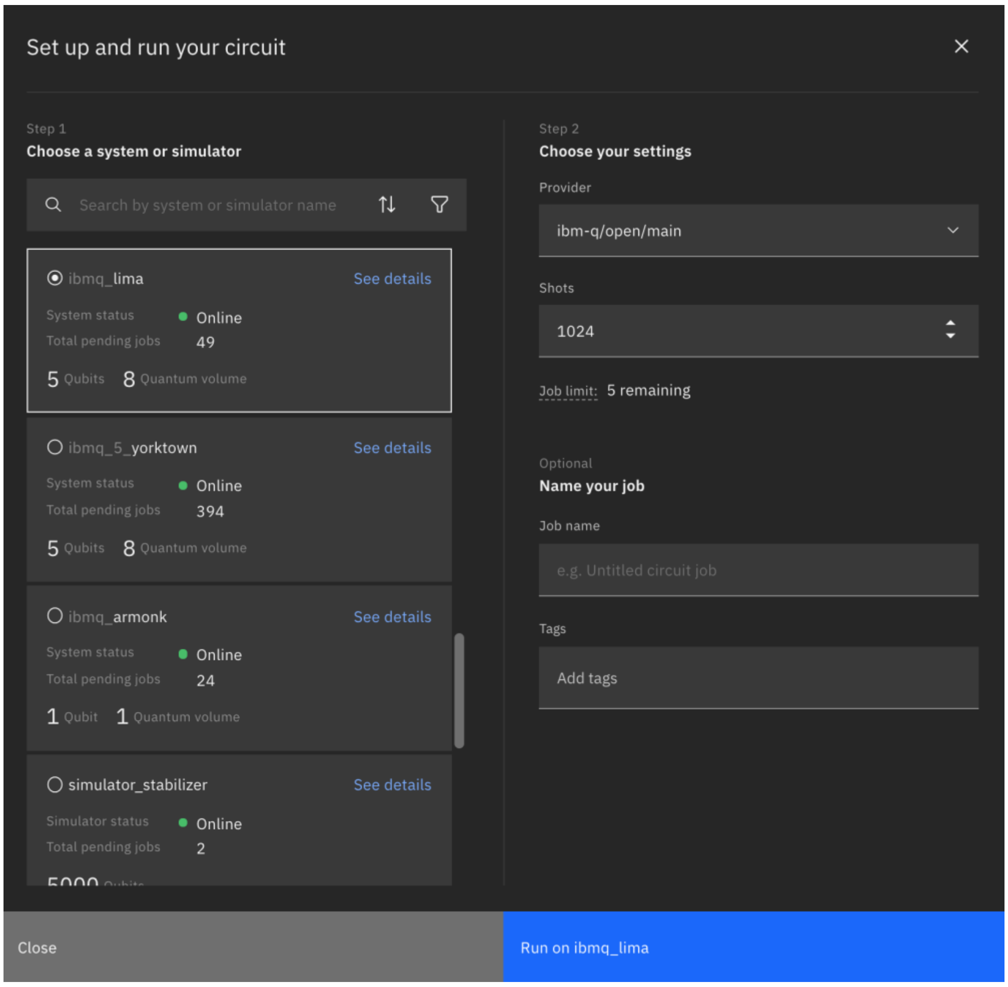
Circuits / Hadamard Saved

Now let us execute the circuit. First, select the "Setup and run" button in the top right.



```
from qiskit import QuantumRegister,
ClassicalRegister, QuantumCircuit
```

As directed, choose a device to run on, select a number of shots (1024 is fine), and optionally name your job. Once you are happy with your settings, press the blue “Run on …” button.


Set up and run your circuit

Step 1
Choose a system or simulator

Search by system or simulator name

ibmq_lima — See details
System status: Online
Total pending jobs: 49
5 Qubits, 8 Quantum volume

ibmq_5_yorktown — See details
System status: Online
Total pending jobs: 394
5 Qubits, 8 Quantum volume

ibmq_armonk — See details
System status: Online
Total pending jobs: 24
1 Qubit, 1 Quantum volume

simulator_stabilizer — See details
Simulator status: Online
Total pending jobs: 2

Step 2
Choose your settings

Provider: ibm-q/open/main

Shots: 1024

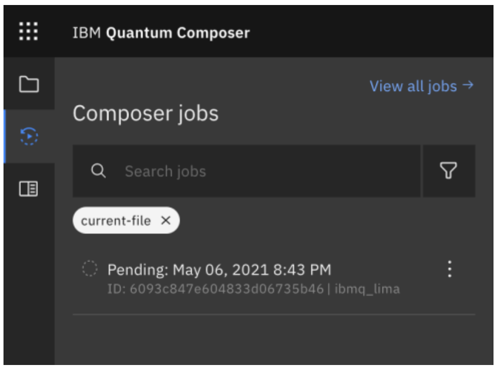
Job limit: 5 remaining

Optional
Name your job

Job name: e.g. Untitled circuit job

Tags: Add tags

By clicking on the “Composer Jobs” tab (in the left hand side-bar), you should see your job queued, when it is has been run, you can also access the results of the jobs here. The queue might be several minutes, as there are many users of the quantum devices. You can submit more than one job at once, so it is possible to use this time to continue on in the lab either alone, or pooling resources with your classmates.


IBM Quantum Composer

Composer jobs

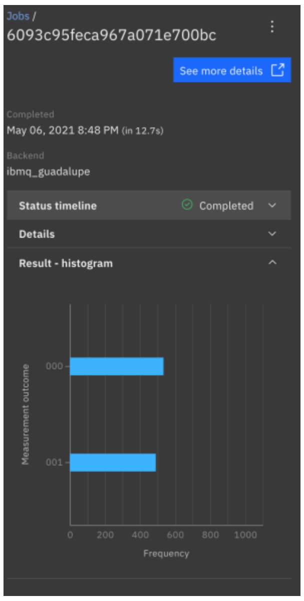
View all jobs →

Search jobs

current-file ×

Pending: May 06, 2021 8:43 PM
ID: 6093c847e604833d06735b46 | ibmq_lima

Once your job shows that is completed, you can examine the results by clicking on the job in the jobs pane. Note that you can see the probability of each measurement, and by clicking "See more details" can additionally see the original circuit, and the "transpiled" circuit which actually ran on the machine. In this case you should get approximately 50/50%.


Jobs /
6093c95feca967a071e700bc

See more details

Completed
May 06, 2021 8:48 PM (in 12.7s)

Backend
ibmq_guadalupe

Status timeline — Completed

Details

Result - histogram

Now add a time delay by adding a lot of H gates, with barrier gates in between them to ensure they are not optimised away. You need dozens of H to see an effect, so don't waste too many units going small. Run this experiment, and see if you can see the effect of noise in your device.

Exercise 9.3.2 *Single qubit superposition decay*. Reset and program a sequence HTH from the $|0\rangle$ state to put it into a known superposition (see Lab 1, or check with the QUI). Add a measurement gate (and as before run the circuit and examine the output).

Exercise 9.3.3 Make a 5-qubit superposition over all binary numbers, run the simulator (by selecting the simulator as the backend) to check you understand what you will get out of the experiment and then run the experiment on a real quantum computing device.

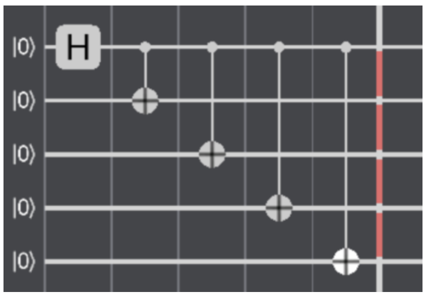
Pause for a second and appreciate what you're doing here – you are programming real qubits, getting actual results, and understanding (hopefully) what's happening.

9.4 Entangled states

Let's now start generating some entanglement.

Exercise 9.4.1 *Bell state.* Reset and program a sequence to put two qubits into a Bell state (see Lectures and/or check with the QUI).

Exercise 9.4.2 *Cat state entanglement.* See if you can entangle all 5 qubits in a GHZ state, $|\psi\rangle = (|00000\rangle + |11111\rangle)/2$. In the QUI this would be generated by a circuit given by the following:



But on the real device you can't directly connect all qubits, and not necessarily in the directions you want. Pick the obvious qubits to act as the controls. Simulate to check you've programmed it correctly and then run the experiment. How did it go?

Exercise 9.4.3 See if you can understand the results from your experiments in terms of the errors quoted for the single and two-qubit gates, by programming a 5-qubit GHZ state circuit in the QUI with X, Y and Z control error gates applied (randomly) with commensurate error levels.

9.5 Algorithms

Now start programming some quantum algorithms and see how far you can push the machine.

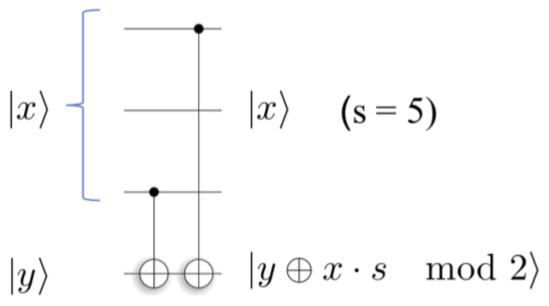
Exercise 9.5.1 Program and run a (non-trivial) instance of Deutsch-Josza. Again, you can use the QUI to investigate and understand how well the machine performs.

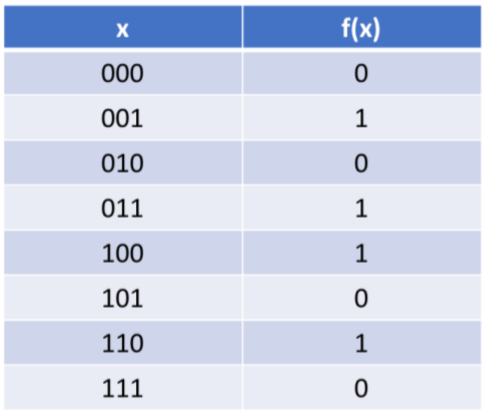
Exercise 9.5.2 Program and run a two-qubit QFT$_4$ circuit. You will need to use a U gate corresponding to the rotation R$_k$ and refer to Wednesday's lecture on how to create the controlled-U operation. Investigate how well the machine performs (and note the measurement bit ordering if you are comparing with the QUI).

Exercise 9.5.3 Can you program and successfully run a three-qubit QFT$_8$ circuit? Investigate and understand how well the machine performs.

Exercise 9.5.4 Can you program and run a Grover search over two qubits? Investigate and understand how well the machine performs, attempting to demonstrate an instance of Grover's algorithm achieving the highest fidelity possible.

Exercise 9.5.5 In the Bernstein-Vazirani problem we have a function $f = x \cdot s$ mod 2 which maps the product of two n-bit numbers to $\{0,1\}$. Program the following (oracle) function in the QUI for the case s = 5 and verify the table of results (e.g. setting the initial state into a superposition via a column of H on the x-register).





x	f(x)
000	0
001	1
010	0
011	1
100	1
101	0
110	1
111	0

Taking care to note the CNOT constraints in the physical system, can you implement BV algorithm for this function and obtain a probability distribution with sufficient precision to determine the value of *s* above? Implement this version of the BV algorithm and examine the result.