Session 4: Superconducting Qubits

Sam Pallister
Quantum Engineering Centre for Doctoral Training,
University of Bristol
sam.pallister@bristol.ac.uk

July 26, 2016

Worksheet Details

This worksheet is for the fourth session of the Quantum Computing day in the 2016 "Quantum in the Summer" school. This session is focused on *superconducting qubits*, i.e. building a quantum computer where the qubits are based on loops of superconducting wire, written onto a chip. The aim is to take the code written in the previous sessions and implement it on a real superconducting device, via IBM's *Quantum Experience*.

1 Running code using IBM's Quantum Experience

IBM's Quantum Experience is a quantum computing simulator, much like QCEngine, but with a twist. It is directly connected to a real, working 5-qubit chip sitting in an IBM facility in the US, and any circuit that can be written in the simulator can instead be submitted as a job and actually run on the real device. The rest of this script will outline how to export code from QCEngine into the Quantum Experience, and how to run it on IBM's device.

The first point to note is that access to the Quantum Experience requires setting up an account with a valid email address. Also, the features necessary to export code from QCEngine require an account with "Expert" privileges. To submit a request for these privileges once logged in, go to the "Account" tab, and then click on the button asking to "Request an upgrade of user status". The upgrade request requires submitting a description of what you plan on using the Quantum Experience for, but the IBM team are super keen to get young students involved in their project and so stating that should be more than enough to get access. At any rate, being granted access typically takes a couple of hours (but can take longer depending on demand).

The main interface for inputting circuits can be found by following the "Quantum Experience Preview" tab, followed by the "Composer" tab. If you are following these instructions for the first time, you should see an interface to set up a new experiment. For now, name the experiment something sensible and click on "Ideal Quantum Processor" to load up the interface. The composer itself is a drag-and-drop tool for building circuits - you have access to a selection of basic gates, and are free to drag them onto the circuit above. Once finished, the "Simulate" button will run the circuit for you, but only via a simulator rather than sending instructions to the real chip. Any circuits saved in this tab can be found under the "My Scores" tab.

To set-up an experiment to run on the real device, start a new experiment by clicking on the "New" button in the composer tab, but then click on the "Real Quantum Processor" button. You are then given two options - to "Simulate" the circuit, or to "Run" it. Simulate itself is broken down into two options: simulate with ideal conditions, which will simulate the circuit perfectly (as in QCEngine), and to simulate the circuit in noisy conditions. This option aims to simulate the imperfect output of the real device, by modelling the noise it experiences; as such, this can be a rough and ready alternative to actually sending the circuit to be run on the real hardware. The "Run" instruction will run the circuit on the real hardware (but each user has a limited number of credits that are used up when this option is chosen - so only make use of it sparingly!). A summary of the possible choices for running circuits is shown in Table 1.

Compilation option	Outcome
Simulate with ideal conditions	Simulate without noise, just like in QCEngine.
Simulate with real conditions	Simulate with the addition of a realistic noise model, to mimic the effect of running it on a real device.
Run	Run the circuit on the real, 5-qubit superconducting chip.

Table 1: The possible compilation options for circuits in the Quantum Experience.

IBM has extensive tutorials for the Quantum Experience, under the "User Guide" tab; go there to get more familiarity with the simulator.

2 Exporting from QCEngine

We'll make use of a short example in this section, that should highlight the steps and key issues in exporting from QCEngine to the Quantum Experience. The example code is as follows:

```
qc.reset(4);
qc.write(3);
qc.hadamard(1);
qc.cnot(4, 1|2);
qc.read();
```

The code initialises four qubits in a particular state, performs a Hadamard on a single qubit and then carries out an entangling gate (a CCNOT, or "Toffoli" gate - a not gate controlled on the state of two qubits, rather than one as in the CNOT case) before measuring. Running this code in the QCEngine workbench:

http://machinelevel.com/qc/doc/qcengine_workbench.html

will produce a circuit output along with the usual graphical representations of the state of the qubits in the circuit. On the right, you should see a button labelled "Generate QASM for IBM 5-Qubit". Clicking this button will generate the following output:

```
// Automatically generated QASM, from http://qcengine.com
x q[0];
x q[1];
h q[0];
// ERROR (instr 2 cnot): too many conditions.
measure q[0];
measure q[1];
measure q[2];
measure q[3];
```

The output contains an error. This is because QCEngine has hard-coded functions for more operations than the Quantum Experience does; in particular, the error complains that the CCNOT gate in the circuit is not hard-coded in the Quantum Experience. The solution is to click the box next to "Expand Toffoli gates", which will rewrite the circuit by replacing every Toffoli gate with gates that are native to the Quantum Experience. Reclicking "Generate QASM..." again will remove this error:

```
// Automatically generated QASM, from http://qcengine.com
x q[0];
x q[1];
h q[0];
h q[2];
cx q[1], q[2];
tdg q[2];
cx q[0], q[2];
t q[2];
cx q[1], q[2];
tdg q[2];
cx q[0], q[2];
t q[2];
h q[2];
tdg q[1];
// ERROR (instr 13 cnot): CNOT target must be qubit 2.
tdg q[1];
// ERROR (instr 15 cnot): CNOT target must be qubit 2.
s q[1];
t q[0];
measure q[0];
measure q[1];
measure q[2];
measure q[3];
```

However, two more have popped up. This is for a different reason, related to the IBM device's specific hardware. It turns out that the chip IBM have designed cannot handle arbitrary choices of target qubit (the qubit that has its state modified, based on the state of another qubit). The 5-qubit device has laid out its qubits in a cross shape, and only the central qubit can be the target of a CNOT gate. Thankfully, there is another button to fix this: clicking the button labelled "Move CNOT to center" and then running again will produce the following output, with no errors:

```
// Automatically generated QASM, from http://qcengine.com
x q[0];
x q[1];
h q[0];
h q[2];
cx q[1], q[2];
tdg q[2];
cx q[0], q[2];
t q[2];
cx q[1], q[2];
tdg q[2];
cx q[0], q[2];
t q[2];
h q[2];
tdg q[1];
cx q[1], q[2];
h q[1];
h q[2];
cx q[1], q[2];
h q[1];
h q[2];
cx q[1], q[2];
cx q[0], q[2];
tdg q[2];
cx q[0], q[2];
s q[2];
t q[0];
measure q[0];
measure q[1];
measure q[2];
measure q[3];
```

Then, take the code without errors and paste it into the text editor on this page (N.B.: access to this page is the step that requires "Expert" level priveleges on the Quantum Experience site):

https://quantumexperience.ng.bluemix.net/qstage/#/editor/qasm

A graphical circuit should be automatically generated, which can either be simulated or run in the usual way.

Best of luck with your own circuits!