



Published in Qiskit



Hassi Norlen

Follow

Feb 2, 2019 · 7 min read · Listen



Save



Sign in to Medium with Google



Steve Justin

stevejustin1963@gmail.com



steve Nationalgranite

steve@nationalgranite.com.au

# Qrasp — Quantum on a Raspberry PI

IBM Q System One... the world's first integrated universal approximate quantum computing system! Oooh, I want one! How hard could it be to build my own?





Sign in to Medium with Google



Steve Justin

stevejustin1963@gmail.com



steve Nationalgranite

steve@nationalgranite.com.au

IBM Q System One

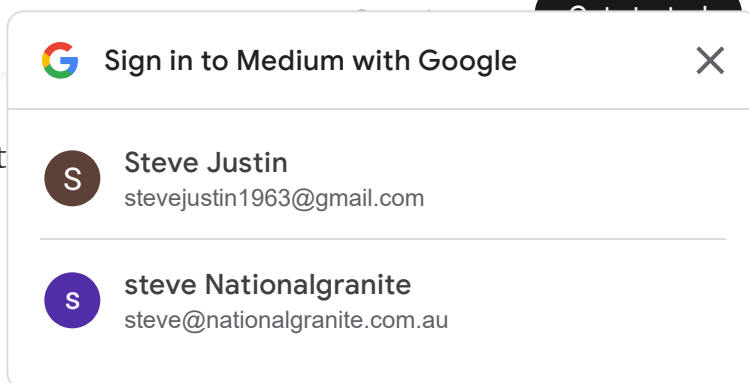
**Quick update:** For those of us who have run into issues installing Qiskit on our Raspberry Pis, there is now an excellent article that goes through the whole process. Go over to [RasQberry: Quantum Computing is the Coolest Project for Raspberry Pi](#) by Jan Lahmann and take a look!





This will not be easy. IBM Q System One offering, with the main hardware, cryostat IBM labs.

Let's see now, what do I need?

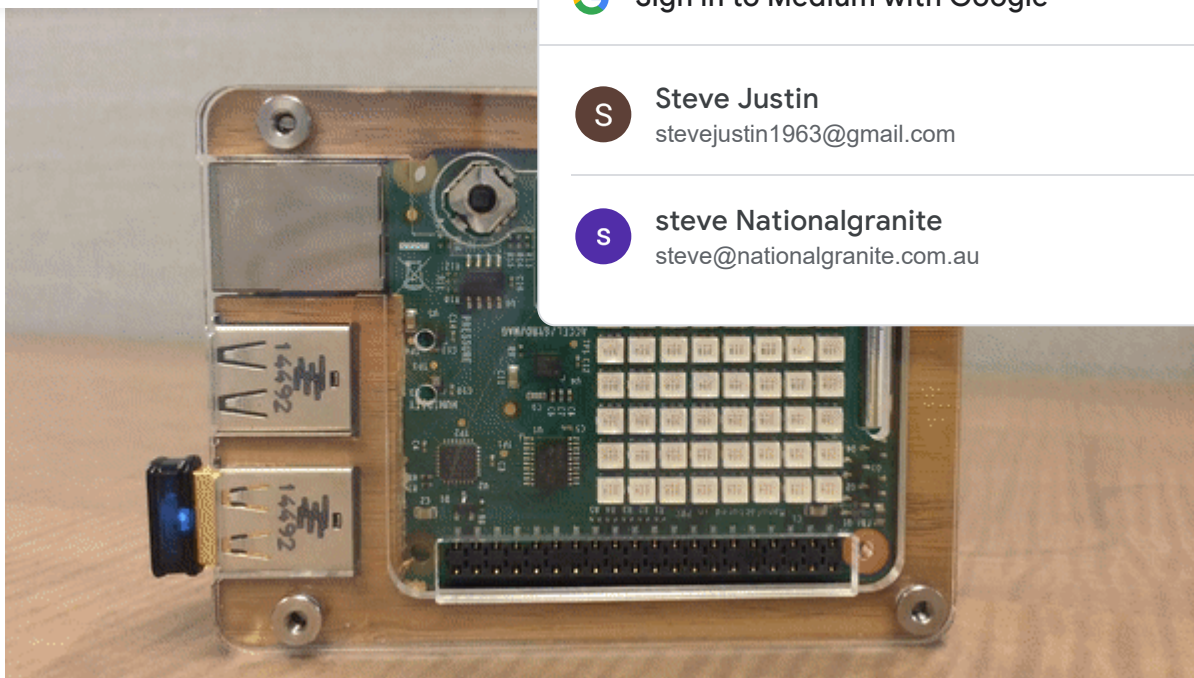


- Awesome state-of-the-art quantum chips, mounted on a steel and gold “chandelier” that is cryostat cooled to a mere 15 millikelvin, and...
  - Hey, I have a Raspberry PI lying around with nothing to do.
- Millions of lines of code to control my experiments, tweak the sensitive qubits, and return the results of my experiments.
  - Ah, it turns out that [Qiskit](#), an open source framework for programming quantum computers, runs on Python, native to the Raspberry PI.
- Gorgeous nine feet cube of half inch thick borosilicate glass by [Goppion, Milan](#).
  - Well, perhaps a nice plastic/bamboo Zebra Case by C4Labs will do instead?
- Amazing visualization tools to present the results of my quantum programs in stunning 3D animations...
  - An 8×8 pixel LED display on my [SenseHat](#) will do!

Let's call this contraption Qrasp (Quantum RASPberry PI)!

You can hack your way through building your own Qrasp without any further knowledge of quantum computing, qubits or Qiskit (I've made my [Qrasp](#) source available on GitHub), but the real feeling of wow sets in when you understand what you have in your hands. Read until the end for a couple of quick starting points on quantum computing and Qiskit.





Qrasp — A Qiskit interface built from a Raspberry PI

After playing with Qiskit for a while, figuring out how to build my own quantum programs and create snazzy visualizations of my quantum results on my laptop, I was itching for something more “tangible”. I wanted a quantum computer that is self-contained, battery powered, flashes and dazzles, that could rest on a desk and that requires no logging in and input beyond some push buttons to start and stop things?

## Quantum Programs

These are the quantum programs I want to run on my Raspberry PI:

- It should be able to show simple superposition (two and three qubits) and entanglement (Bell and GHZ states), but also be flexible enough to swap in and out various quantum programs as “plugins”.
- It should be able to run quantum programs on a simulator back end as well as on a real IBM Quantum device.

## Hardware



## Backend

The Qrasp software now lets you select to run on a quantum simulator or on actual IBM Q hardware. When you press the joystick to display the name of the backend, if Qrasp is not connected to the internet it will show the simulator.



Sign in to Medium with Google



Steve Justin

stevejustin1963@gmail.com



steve Nationalgranite

steve@nationalgranite.com.au

**Important:** To run against the IBM Q Experience, you must first configure your Qrasp environment with an API key. For info, see the Qrasp [README](#) on GitHub.



164



5

## Software

The Qrasp project came together as a set of Python scripts:

### 1. main\_controller.py

This script loads the required libraries and quantum program scripts, and uses the SenseHat joystick to select one of the Quantum programs.

### 2. qc\_sensehat\_func.py

The display script uses the SenseHat 8×8 LED display to show a bar graph of 2 or 3 qubit Qiskit results dictionaries.

### 3. Qconfig\_IBMQ\_experience.py

A configuration file for storing your IBM Q Experience API key.

### 4. Quantum program scripts

These Qiskit scripts creates and executes quantum programs that run on a local quantum simulator, and then calls the display script with the results.

Now, in your implementation, there is nothing stopping you from jumping in and modifying the Python code for these functions as you please. To get familiar with coding in Qiskit, why not go through some Qiskit Terra [tutorials](#), or why not start exploring with some [guided exercises for developers](#).

*q2\_calling\_sense\_func.py*

Python function that creates a simple, two qubit quantum circuit and sets up and measures each qubit in a superposition state.





*bell\_calling\_sense\_func.py*

Python function that creates a simple  
measures a Bell, or entangled, state.

*GHZ\_calling\_sense\_func.py*

Python function that creates a simple  
measures an entangled GHZ state.



Sign in to Medium with Google



Steve Justin

stevejustin1963@gmail.com



steve Nationalgranite

steve@nationalgranite.com.au

The source is available on GitHub, here: [Qrasp](#)

## Input

To be able to run Qrasp with no additional input, I have set the *main\_controller.py* script to run on startup, and use the built-in SenseHat joystick to select among my four basic quantum programs:

- Left: Two qubit superposition

In this program we initialize two independent qubits into the ground state  $|0\rangle$ , and then apply a Hadamard gate to each of them, which effectively puts each qubit into a superposition of  $|0\rangle$  and  $|1\rangle$ . We then measure the qubits and Qrasp creates a four column bar diagram that shows that the four possible outcomes have roughly equal probability ( $\sim 25\%$ ).

- Right: Three qubit superposition

In this program we expand the quantum circuit by adding a third qubit in its ground state, and then applying the Hadamard gate. With three qubits, the generated bar diagram shows that the eight possible outcomes still have roughly the same probability ( $\sim 12.5\%$ ).

- Up: Two qubit Bell state (entanglement)

Now things get interesting. Again we initialize two qubits into their ground state, but the gates applied differ. In this case, we apply the Hadamard gate to only one of the qubits, directly followed by a so called controlled-NOT gate that “maps” the state of the superpositioned qubit onto the other qubit in a maneuver that is called







counter intuitive outcome is called a  
computing, and was recently the sub

- Down: Three qubit GHZ state (entan  
To make a long story short, we are no  
result of, again, two outcomes. Either  
With this GHZ state  $|000\rangle$  and  $|111\rangle$  display with roughly equal ( $\sim 50\%$ ) probability.
- Push: Set the backend: AER 'qasm\_simulator' or IBMQ ('ibmqx2', 'ibmqx4' or auto-select 'least busy'. Actual hardware configurable in *main\_controller.py*).



Sign in to Medium with Google



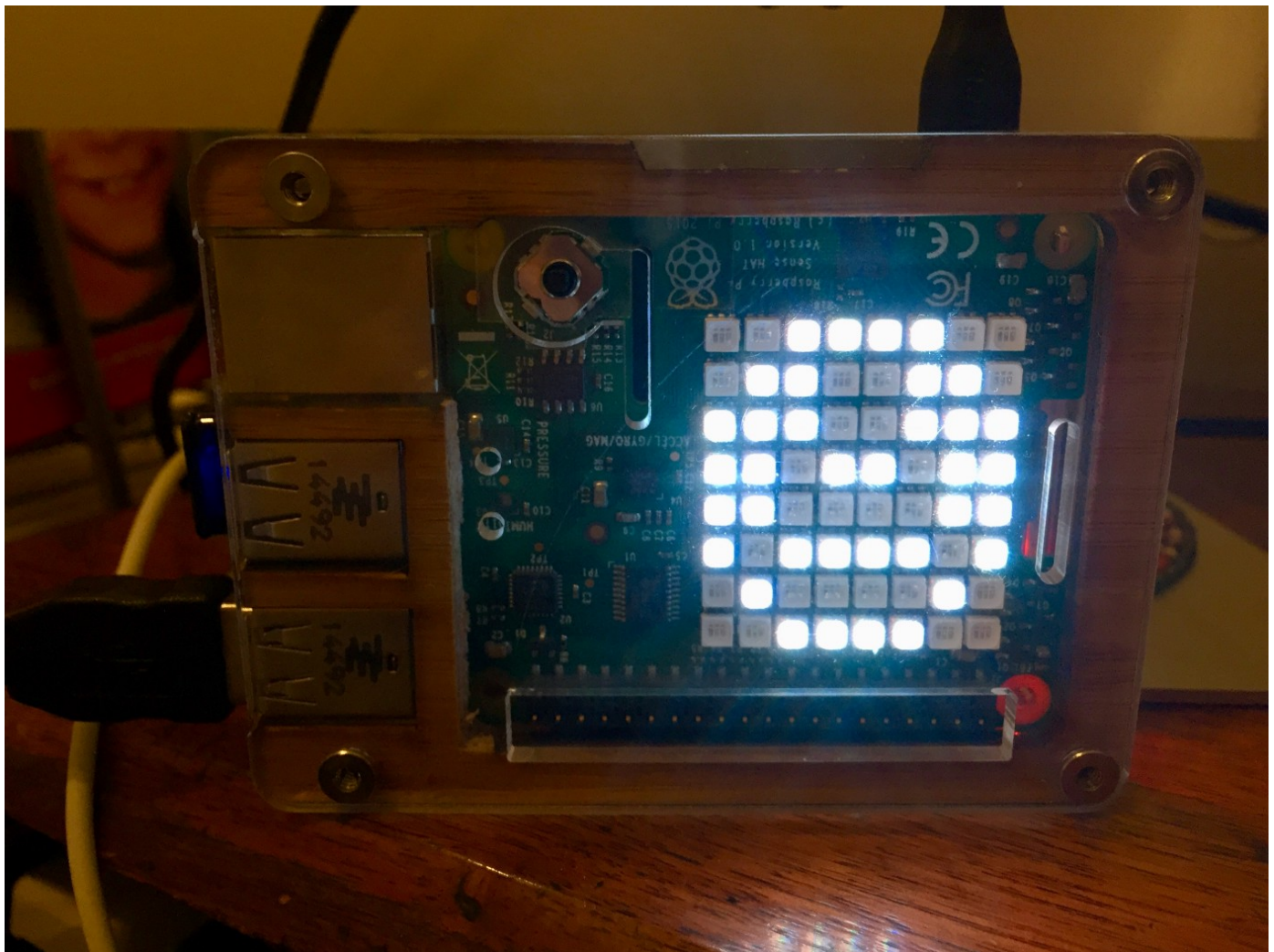
Steve Justin

stevejustin1963@gmail.com



steve Nationalgranite

steve@nationalgranite.com.au



Qrasp in Aer simulator mode.





and graphics, the only viable option would be to limit to the number of qubits in my quantum circuit of eight bars for the probability plot.

Why eight? It turns out that the possible states scale exponentially, as  $2^n$ . For a three qubit circuit,

as a Python glossary that includes the number of shots that resulted in a specific state, something like this:

```
{'000': 552, '111': 472}
```

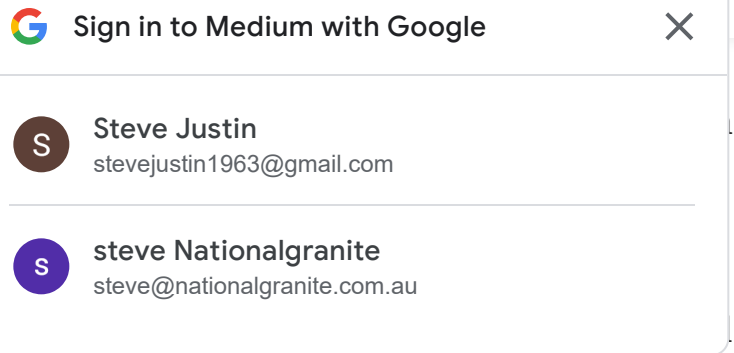
And the possible number of states, or combinations of 1 and 0 for the returned results equal  $2^3 = 8$ :

```
|000>, |001>, |010>, |011>, |100>, |101>, |110>, |111>.
```

Thus I need exactly eight columns to display all possible states of a three qubit quantum circuit. The two qubit results are displayed in just 4 columns ( $2^2$ ), so uses just half the screen.

## All together now

With all the pieces together, this is how Qrasp runs.







Sign in to Medium with Google



Steve Justin

stevejustin1963@gmail.com



steve Nationalgranite

steve@nationalgranite.com.au

After the initial, very colorful SenseHat boot sequence, the Qrasp main controller launches automatically and displays the messages “Qiskit ” and then the super-positioned 0/1 icon signaling that it is ready to go.

Moving the joystick up displays the “2Q” message, indicating a superposition of two qubits and then launches the q2\_calling\_sense\_func.py script which performs the necessary quantum calculations and then calls the display script to display the four column bar graph results of a two qubit superposition.


Now move the joystick in the other directions, and see the other quantum programs execute! Then press the joystick button to take the actual IBM Q hardware for a spin.


## Some promised starting points

- What IS quantum computing?






- Take a look at the IBM Q Experience
  - RasQberry: Quantum Computing is t  
Lahmann, IBM Quantum Ambassado  
Technology.
  - Getting started with Qiskit by Doug McElreath, Qiskit at IBM.
-  Sign in to Medium with Google ×



Steve Justin

stevejustin1963@gmail.com





steve Nationalgranite


steve@nationalgranite.com.au


  - Start exploring Qiskit coding with some guided exercises for developers by James Weaver (JavaFXpert).
- 
- 
- 
- <https://medium.com/qiskit/qrasp-a-wee-quantum-computer-74ef7f927b1e>
- 10/11


Get the Medium app

Download on the  
App Store

GET IT ON  
Google Play

Sign in to Medium with Google

Steve Justin  
stevejustin1963@gmail.com

steve Nationalgranite  
steve@nationalgranite.com.au