The IBM Q Initiative as a Resource for HEP Quantum Computing

Next Steps in Quantum Science for HEP
Fermilab
Sept 12-14, 2018

Patrick Dreher
Research Professor, Computer Science
Associate Faculty, Department of Physics
NC State University

Resource Perspective for HEP Quantum Computing Seen from an IBM Q Hub

OUTLINE

- IBM Q Initiative
- The IBM Q Network
- The NC State Hub
  - Educational Component
  - Research Component
- IBM Q as a Resource for High Energy Physics
  - Simulators
  - Quantum computing hardware
- Practical Next Steps and Summary

What is the IBM Q Initiative

- IBM Q is an industry-first initiative to build commercially available universal quantum computing systems for business and science.

IBM Q Network Summary Description

- IBM is establishing the IBM Q Network as a worldwide organization of companies, academic institutions and national research lab pursuing practical applications for quantum computing

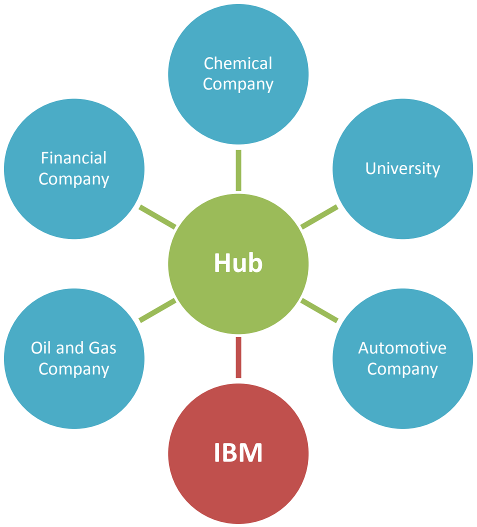
- Goal is to significantly increase the computational capability of future quantum systems and demonstrate computational capabilities beyond today's classical computing systems

The IBM Q Network

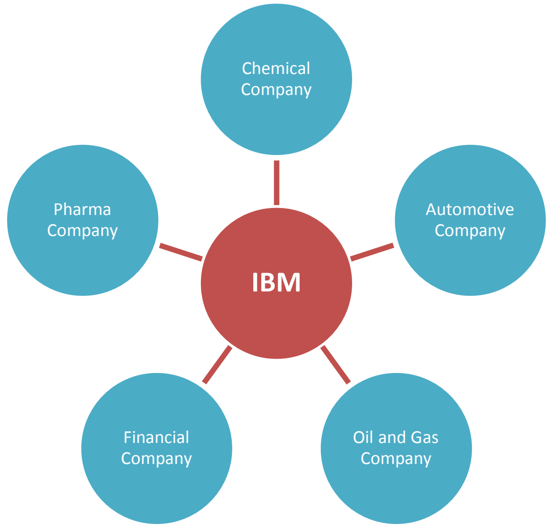
(Information Provided by IBM – September 2018)

Hubs	Hubs	Partners	Education Partner	Members	Startups
Keio University Tokyo, Japan **Oxford University** Oxford, UK **Oak Ridge National Laboratory** Oak Ridge, TN, USA **University of Melbourne** Parkville VIC, Australia	**North Carolina State University** Raleigh, NC, USA **Universität der Bundeswehr** München, Germany	**JPMC** New York, NY, USA **Accenture** Dublin, Ireland **Samsung** Seoul, South Korea **Daimler** Stuttgart, Germany **JSR** Minato-ku, Tokyo, Japan	**MIT** Cambridge, MA, MA, USA **Academic Partner** **Universidade do Minho** Portugal	**Honda** Minato, Tokyo, Japan **Barclays** London, UK **Hitachi Metals** Minato, Tokyo, Japan **Nagase** Chuo-ku, Tokyo, Tokyo, Japan	1QBit Cambridge Quantum Computing Q-CTRL QC Ware Quantum Benchmark QxBranch Strangeworks Zapata Computing

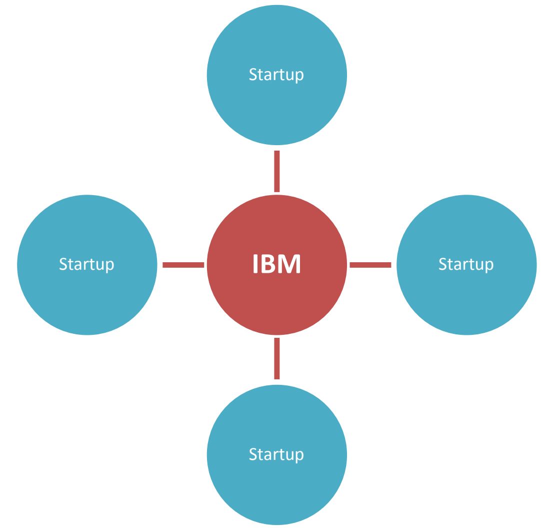
Types of Engagement within the IBM Q Network*



Hubs
Regional centers for training, education, technical support workshops, events and opportunities for joint work



Partners
Focused on quantum computing in a specific industry or academic field



Members
Startups and industry

* Figures courtesy of IBM

IBM Q Network: North Carolina State Hub

Announced May 10, 2018

Tentative Operational Date: October 1, 2018

IBM Announces Collaboration with North Carolina State University to Accelerate Quantum Computing

NC State to join the IBM Q Network as first university-based IBM Q Hub in North America

NEWS PROVIDED BY
IBM →
May 10, 2018, 09:00 ET

SHARE THIS ARTICLE

YORKTOWN HEIGHTS, N.Y., May 10, 2018 /PRNewswire/ -- IBM (NYSE: IBM) today announced that North Carolina State University (NC State) will join the IBM Q Network™ as the first university-based IBM Q Hub in North America. The university will work directly with IBM to advance quantum computing and industry collaborations, as part of the IBM Q Network's growing quantum computing ecosystem.

NC State IBM Q Hub

- The NC State Hub is advancing 3 programs
  1. Educational role
     a) Rapid expansion of course offerings internally within the University
     b) External quantum computing curriculum development
        i. Support for hub partner training
        ii. Short courses and seminars
        iii. Public educational outreach in quantum computing
  2. Potential research opportunities
     a) Partnerships with other universities
     b) Research collaborations
  3. Building research and project Industrial Partners – this programmatic aspect of the hub not strongly tied to HEP

NCSU Hub Educational Role

- External
  - Series of short introductory and intermediate QC courses open to the larger community
    - Modelled after NCSU existing Data Science Initiative
    - Aimed at students and professionals in business, research, and government
  - Specialized QC courses for NCSU hub partners – customized per request
  - Quantum computing campus speaker series
  - Additional course and training being developed in coordination with ORNL -- Quantum Computing Institute (ORNL IBM Hub)

NCSU Hub Educational Role

- Internal
  - Upper level undergrad and grad courses being developed in several academic departments that will have resource ties to Hub
  - Working with beta version of portal software developed by IBM Yorktown Heights for use in classes working with the IBM Q Composer tool for 5 qubit QC Circuits

NCSU Hub Research Role

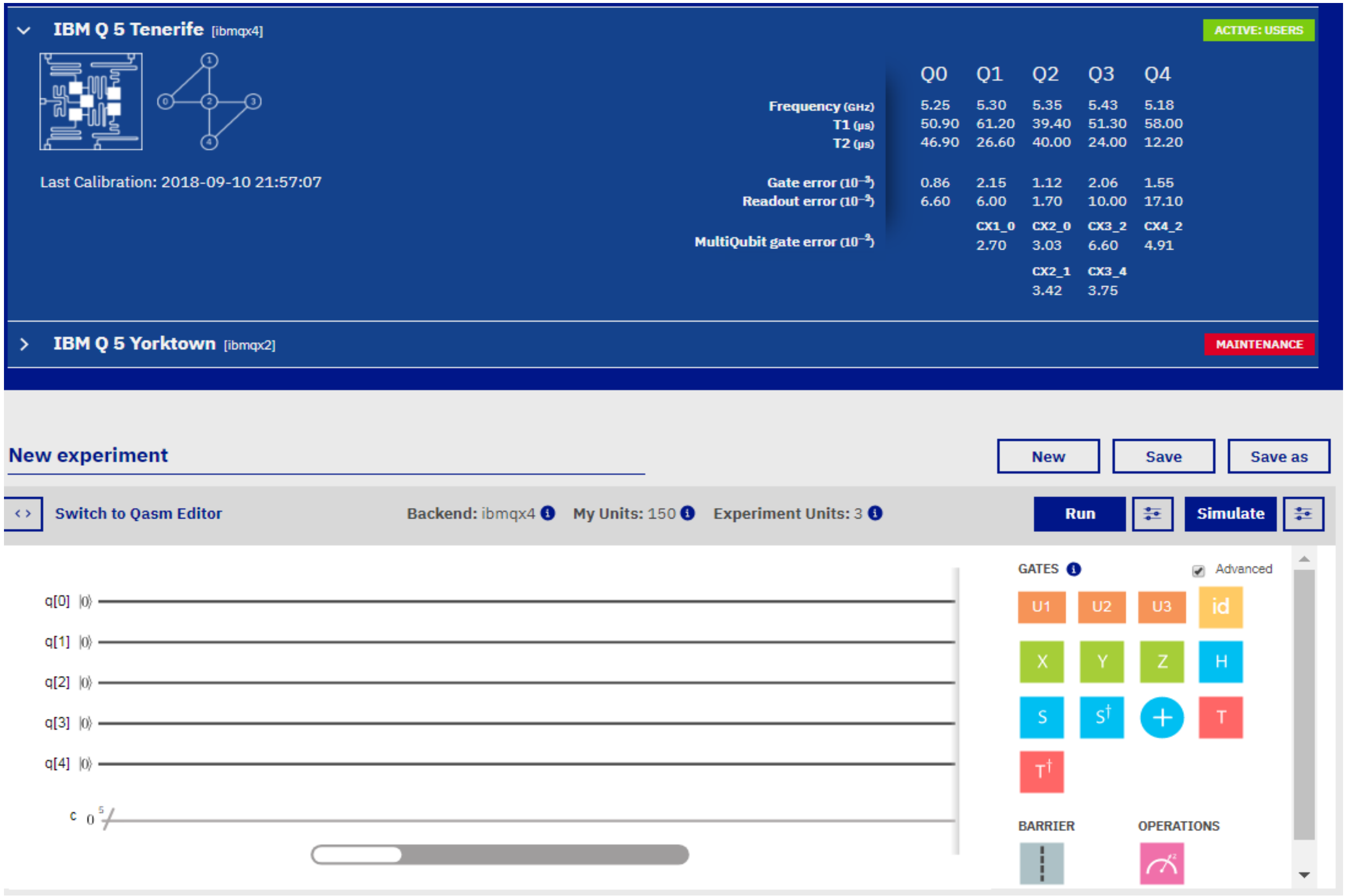
- Research – several aspects
  - Infrastructure
    - Develop procedures and software for customized desktop installations for IBM Q access (Jupyter notebooks, container based images for IBM Q access on cloud computing, etc.)
  - Operational
    - Develop and test the connectivity and sessions to the simulators and various IBM QC hardware devices
  - Internal
    - Various NCSU faculty have begun QC research projects using the IBM Q simulators and QC hardware
  - External
    - University is in discussions with several universities regarding partner type participation with NCSU in the IBM Q

Working with the IBM Q in High Energy Physics

- For HEP - focus is on the educational and research aspects of a hub
- General Resources
  - Composer (Graphical Gate Interface - 5 qubit only)
    - Beta version of portal for educational courses (NCSU-IBM)
  - Simulators
  - Software environments
  - Multiple quantum computer hardware platforms
    - 20 qubit machine (today) and 50 qubit machine (anticipated early 2019)
- Research possibilities

Resources - IBM Q Composer

Interactive Graphical Interface to experiment constructing gate operations for 5 qubit machine



Gate Construct Example with Composer

Composer simulation also generates QASM code

Experiment #20180911010213

Device: Simulator

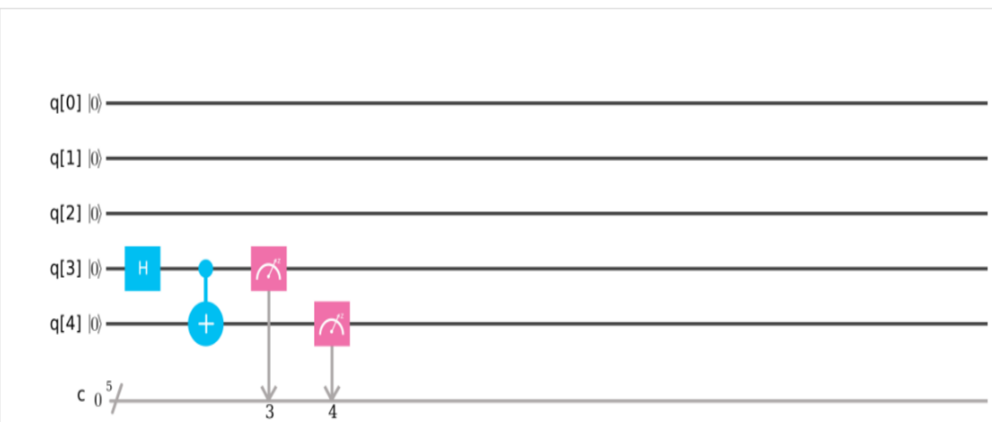
Quantum State: Computation Basis

Download CSV

```
include "qelib1.inc";
qreg q[5];
creg c[5];

h q[3];
cx q[3],q[4];
measure q[3] -> c[3];
measure q[4] -> c[4];
```

Import QASM   Download QASM

Quantum Circuit



OPENQASM 2.0

```
include "qelib1.inc";

qreg q[5];
creg c[5];

h q[3];
cx q[3],q[4];
measure q[3] -> c[3];
measure q[4] -> c[4];
```

Open in Composer

Edit in QASM Editor

Resources - Simulators

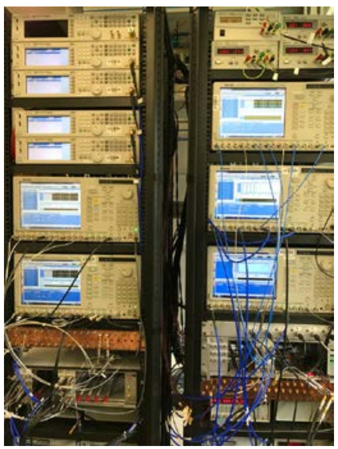
- Composer only works on 5 qubit machine
- Working with larger QC hardware will require writing code
- There is a 32 qubit QASM simulator
- Open Quantum Assembly Language (QASM)
  - Interface language for the Quantum Experience that enables experiments with small depth quantum circuits to be coded and run on QC platforms

Resources - Software Environments

- Jupyter notebooks / desktop environments for developing QC code
- Ongoing software development work to build language hierarchy for QC program execution
  - Programming languages
  - Convert to program binary
  - Convert to QC executable
  - Construct low level gate operations

Resources - QC Hardware Platforms*

Microwave electronics



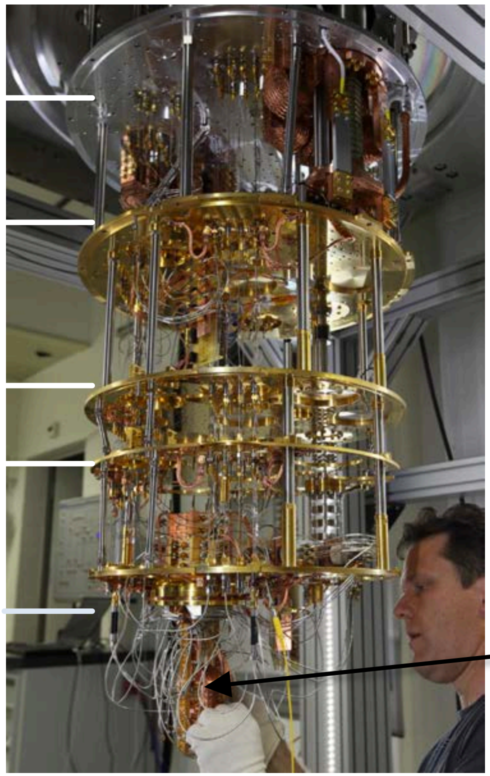
40K

3K

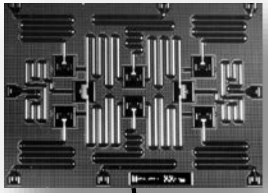
0.9K

0.1K

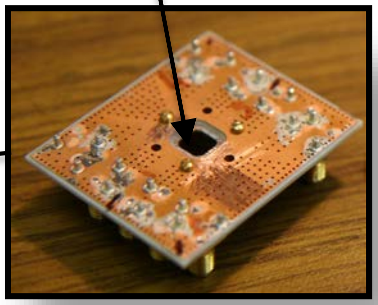
0.015K



Refrigerator to cool qubits to 10 - 15 mK with a mixture of ^{3}He and ^{4}He



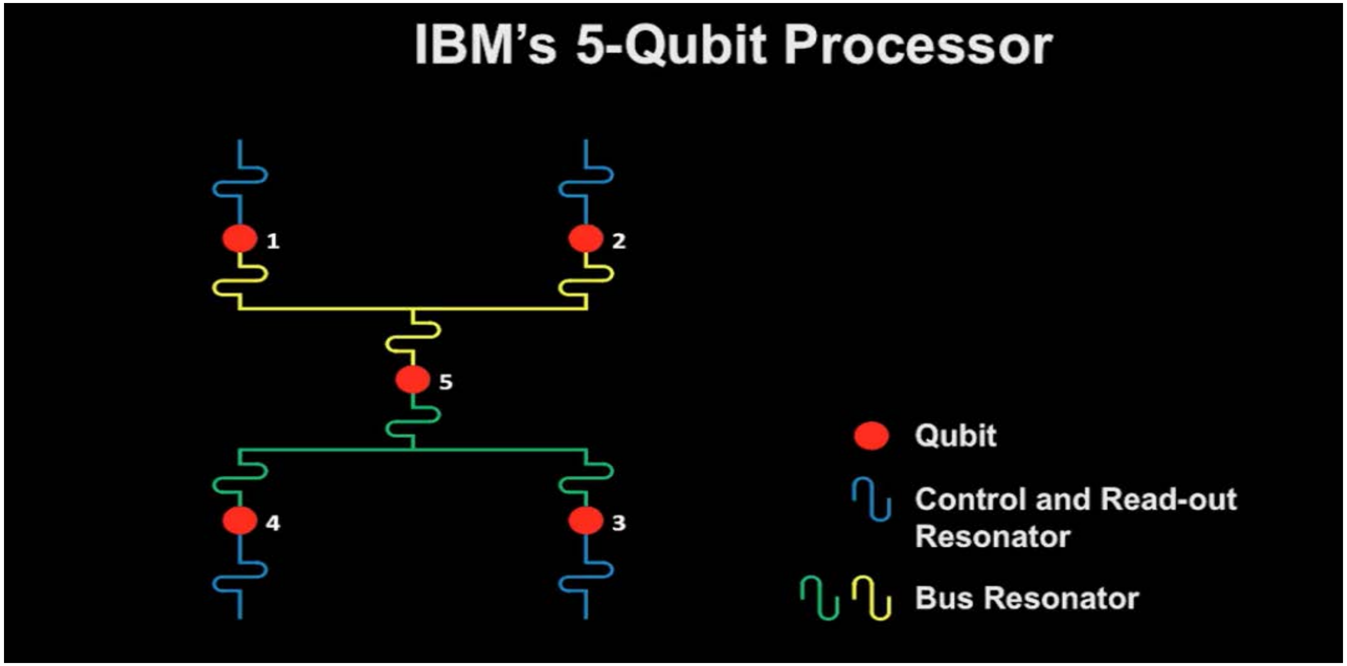
Chip with superconducting qubits and resonators



PCB with the qubit chip at 15 mK
Protected from the environment by multiple shields

* Slide courtesy of IBM

5 Qubit Tenerife (ibmqx4)



Availability & status

For public use

- Online

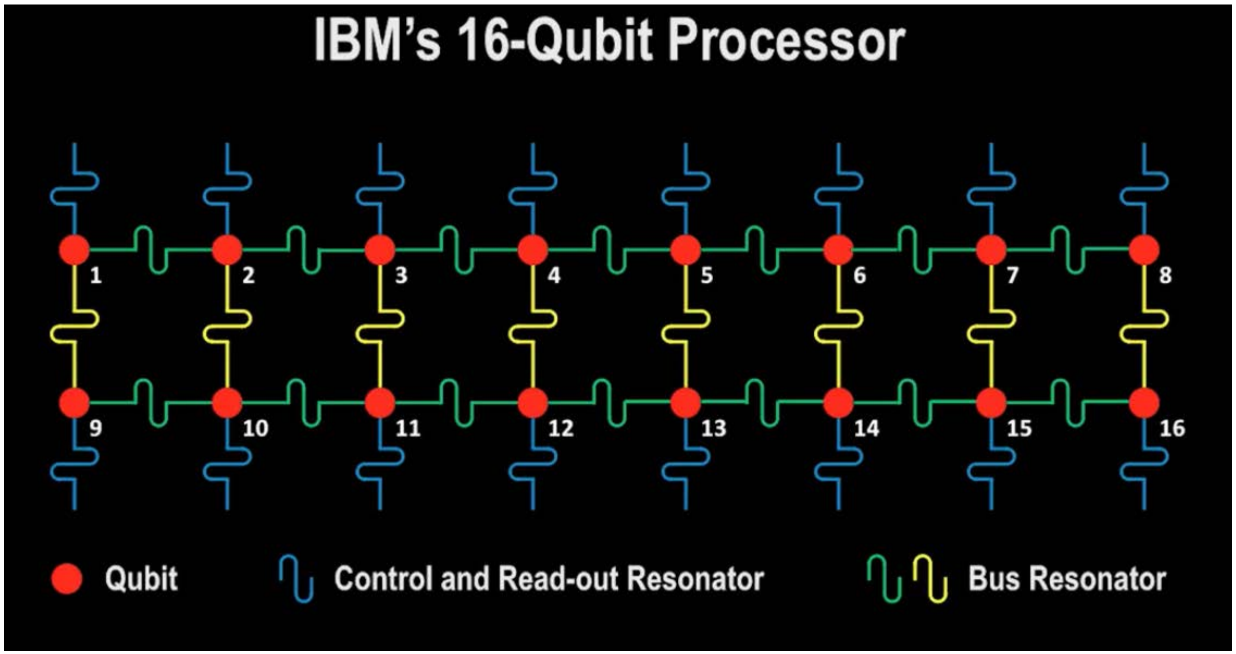
Last calibration occured

2018-09-10 9:57:07 pm

Average measurements

Frequency (GHz)	5.25
T1 (μs)	50.90
T2 (μs)	46.90
Gate error (10^{-3})	0.86
Readout error (10^{-2})	6.60

16 Qubit Rüschlikon (ibmqx5)



Availability & status

For public use

- Online

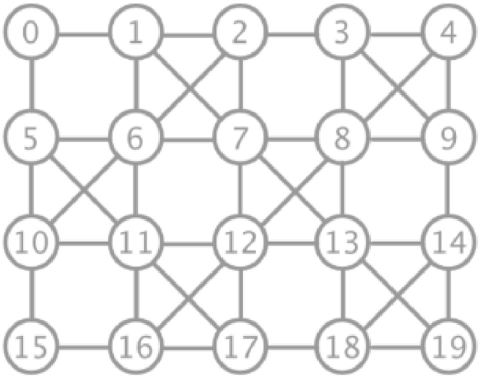
Last calibration occured

2018-09-10 10:46:18 pm

Average measurements

Frequency (GHz)	5.26
T1 (μs)	43.90
T2 (μs)	28.00
Gate error (10^{-3})	2.66
Readout error (10^{-2})	5.15

20 Qubit Toyko (ibmq_20_Toyko)



Availability & status

For IBM Q clients

- Online

Last calibration occured

2018-09-10 8:15:18 pm

Average measurements

Frequency (GHz)	4.97
T1 (μs)	87.13
T2 (μs)	58.87
Gate error (10^{-3})	1.79
Readout error (10^{-2})	8.91

What Can Be Expected for HEP Algorithms Implemented on a QC

Hamiltonians of Interest in HEP (this conference/workshop)

- High Energy physics problems with potential interest for implementation on a quantum computer
  - Spin systems with analogous strong or weak interaction properties
  - Condensed matter
  - Dynamic evolution of systems
  - Finite density systems – quark gluon plasma
  - Abelian gauge theories
  - Non-abelian gauge theories
    - Quenched and unquenched QCD
  - Alternative formulations - Quantum Link Models
  - Quantum gravity
  - Other talks this conference-workshop

Potential Approaches for QC Simulations (this conference/workshop)

- Potential approaches to developing QC codes using present and near-term future HW and SW
  - Quantum classical approach
    - Transform 1st quantized to 2nd quantized Hamiltonian (Jordan-Wigner, Bravyi-Kitaev, etc)
    - Phase estimation algorithm
    - Variational Quantum Eigensolver
    - other
  - Directly construct product state Hamiltonian that can be mapped to a quantum computer
    - Transform 1st quantized to 2nd quantized Hamiltonian (Jordan-Wigner, Bravyi-Kitaev, etc)
    - Quantum Fourier transform – transforms creation annihilation Hamiltonian formulation to a momentum space Hamiltonian
    - Bogoliubov transformation – decouple modes with opposite momentum

Practical Limitations Present and Near-Term Future QC Simulations

- Number of qubits available is limited
- Available qubits are characterized by NISQ (Noisy Intermediate-Scale Quantum)
- Machine dependent constraints with direct connectivity between qubits
  - Complexity of qubit Hamiltonian
  - Qubit coherence and gate errors
  - Circuit depth
  - Issues of Scalability

Example of Practical QC Limitations: Bernstein Vazirani Algorithm

- The Bernstein Vazirani Algorithm is a mathematical – computer science algorithm that poses the problem of how to determine the value of a hidden integer $s \in \{0,1\}^n$ from an oracle f_s that returns a bit mod 2 from

$$s \cdot x \equiv \sum_i s_i x_i$$

upon receiving an input $x \in \{0,1\}^n$

Classical Determination of the Value “s”

- Classically the hidden integer can be discovered by querying the oracle with

  $$x = 1, 2, \ldots, 2^{n-1}$$

- Each query provides the i^{th} bit of information that can be discovered
- The classical approach requires n probes of the oracle to determine the value of “s”

Quantum Mechanical Construction of Bernstein Vazirani Problem

- Querying the quantum oracle one notices that

$$(-1)^{s\cdot x} = (-1)^{s_1\cdot x_1}\dots(-1)^{s_n\cdot x_n} = \prod_{i:s_i=1}(-1)^{x_i}$$

- This equation can be re-expressed by a unitary transformation that is decomposable as single-qubit unitaries

where $\mathbb{Q}_{f_s} = \mathbb{Q}^1 \otimes \mathbb{Q}^2 \otimes \dots . \mathbb{Q}^i \otimes \dots \mathbb{Q}^n$

$\mathbb{Q}^i = (1 - a_i)I - s_i Z$

Z is the Pauli Z matrix

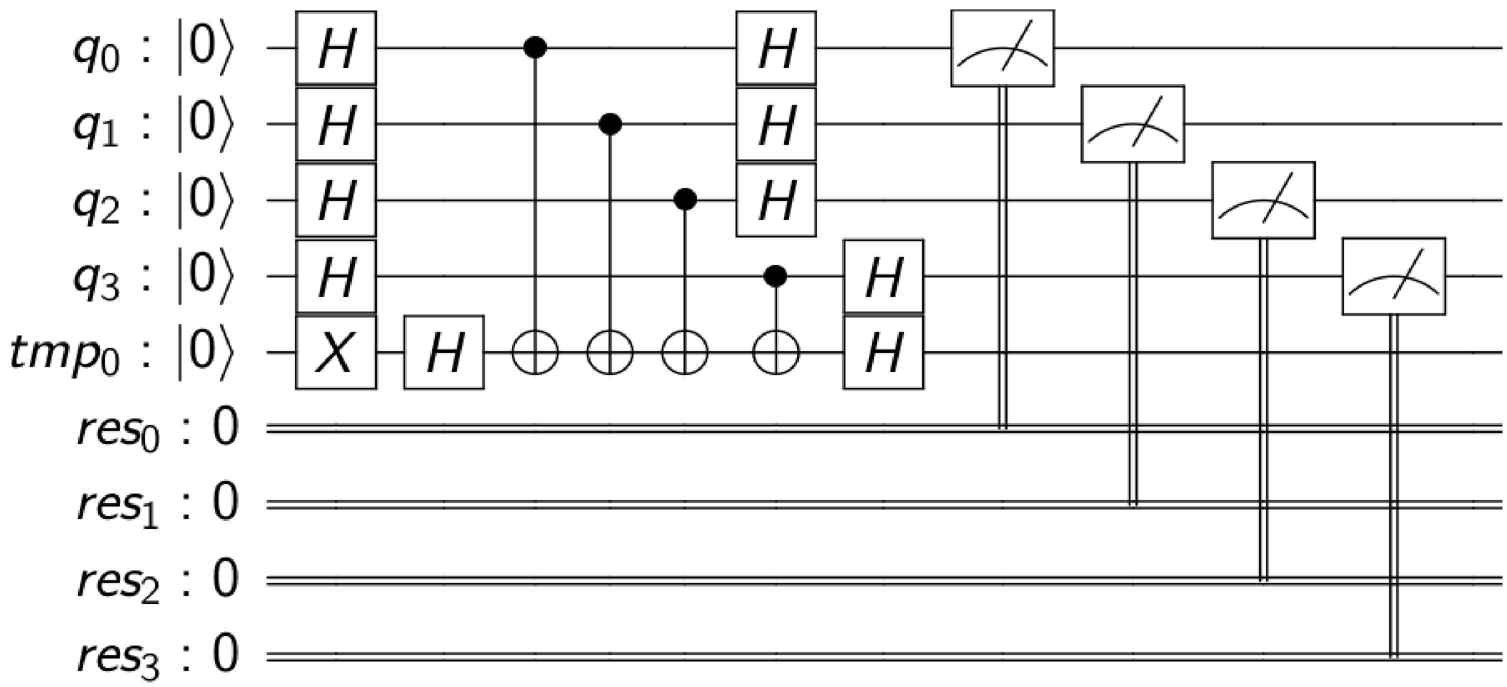
I is the identity matrix

$s_i \in \{0,1\}$

Bernstein Vazirani Procedure

- Procedure
  - Start with a |0> state
  - Apply series of transformations that can be described by unitaries acting on a single qubit
  - Apply Hadamard gates to each qubit
  - Call the decomposable quantum oracle
  - Apply another set of Hadamard gates
  - Measure the result

Graphical Representation of 4 Qubit Example of Bernstein Vazirani Problem



Simulators and QC hardware

- Pick s=9 and run the BV algorithm on both the IBM Q simulator and quantum hardware

```
In [73]: for backend in backends:
             pprint(backend.status)
```

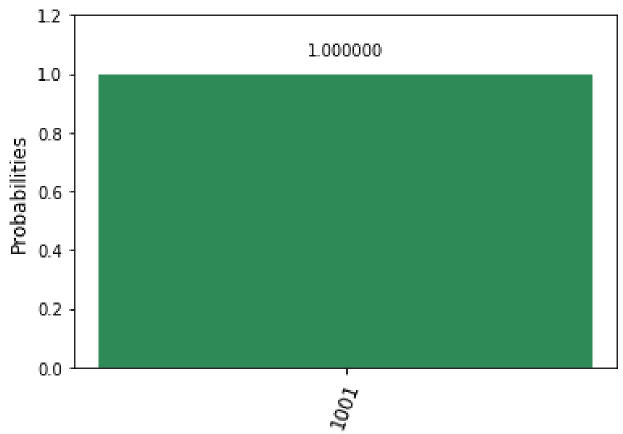
```
{'available': True, 'name': 'ibmqx5', 'pending_jobs': 18}
{'available': True, 'name': 'ibmq_qasm_simulator', 'pending_jobs': 0}
{'available': True, 'name': 'local_unitary_simulator_py'}
{'available': True, 'name': 'ibmqx4', 'pending_jobs': 0}
{'available': True, 'name': 'local_qasm_simulator_py'}
{'available': False, 'name': 'ibmqx2', 'pending_jobs': 0}
{'available': True, 'name': 'local_statevector_simulator_py'}
```

```
In [89]: s = 9 # 1001
         oracle = QuantumCircuit(q, tmp, res)
         for i in range(len(q)):
             if (s & (1 << i)):
                 oracle.cx(q[i],tmp[0])
```

```
In [99]: result = qp.execute(('bv'), backend='ibmq_qasm_simulator', timeout=300, shots=1024)
         #
         #  IBM simulators
         #       ibmq_qasm_simulator
         #
         # local simulators
         #      local_qasm_simulator_py
         #      local_unitary_simulator_py
         #      local_statevector_simulator_py
         # QC
         # ibmqx2  ibmqx4  ibmqx5
```

```
In [100]: # from qiskit.tools.visualization import plot_histogram
          counts = result.get_counts('bv')
          print(counts)
          plot_histogram(counts)
```

```
{'1001': 1024}
```

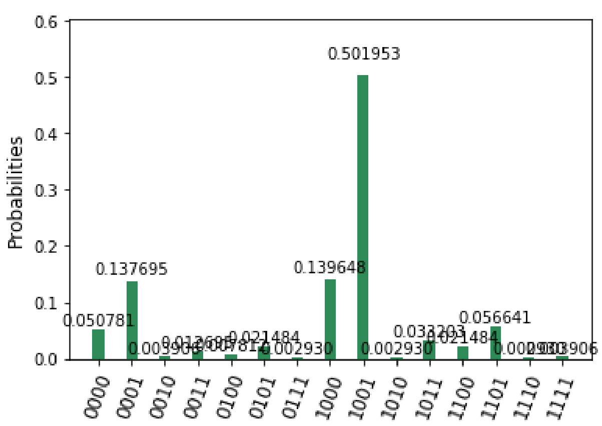


In [77]:
```python
s = 9 # 1001
oracle = QuantumCircuit(q, tmp, res)
for i in range(len(q)):
    if (s & (1 << i)):
        oracle.cx(q[i],tmp[0])
```

In [87]:
```python
result = qp.execute(('bv'), backend='ibmqx4', timeout=300, shots=1024)
#
#  IBM simulators
#       ibmq_qasm_simulator
#
# local simulators
#       local_qasm_simulator_py
#       local_unitary_simulator_py
#       local_statevector_simulator_py
# QC
# ibmqx2  ibmqx4  ibmqx5
```

In [75]:
```python
# from qiskit.tools.visualization import plot_histogram
counts = result.get_counts('bv')
print(counts)
plot_histogram(counts)
```

```
{'1111': 4, '1010': 3, '1011': 34, '1001': 514, '0001': 141, '0000': 52, '1101': 58, '1110': 3, '0010': 4, '1000': 143, '
0100': 8, '0011': 13, '1100': 22, '0111': 3, '0101': 22}
```

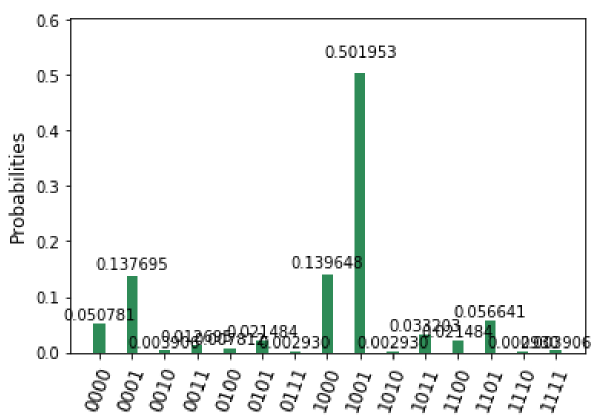


```
In [64]: s = 9 # 1001
         oracle = QuantumCircuit(q, tmp, res)
         for i in range(len(q)):
             if (s & (1 << i)):
                 oracle.cx(q[i],tmp[0])
```

```
In [74]: result = qp.execute(['bv'], backend='ibmqx5', timeout=300, shots=1024)
         #
         #  IBM simulators
         #       ibmq_qasm_simulator
         #
         # local simulators
         #      local_qasm_simulator_py
         #      local_unitary_simulator_py
         #      local_statevector_simulator_py
         # QC
         # ibmqx2  ibmqx4  ibmqx5
```

```
In [75]: # from qiskit.tools.visualization import plot_histogram
         counts = result.get_counts('bv')
         print(counts)
         plot_histogram(counts)
```

```
{'1111': 4, '1010': 3, '1011': 34, '1001': 514, '0001': 141, '0000': 52, '1101': 58, '1110': 3, '0010': 4, '1000': 143, '
0100': 8, '0011': 13, '1100': 22, '0111': 3, '0101': 22}
```



20 Qubit Machine

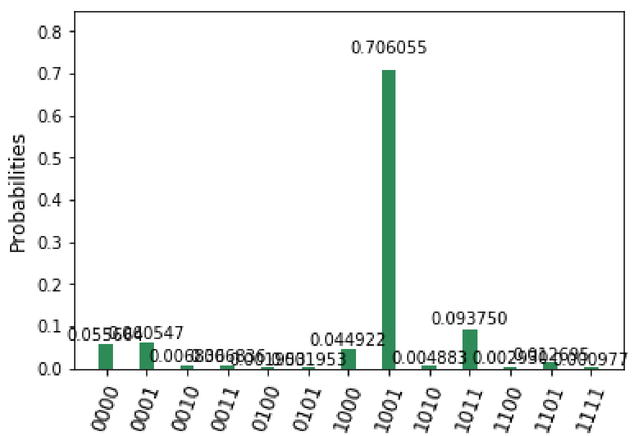
```
In [11]: for backend in backends:
             pprint(backend.status)

{'available': True, 'name': 'ibmq_qasm_simulator', 'pending_jobs': 0}
{'available': True, 'name': 'local_qasm_simulator_py'}
{'available': True, 'name': 'ibmq_20_tokyo', 'pending_jobs': -1}
{'available': True, 'name': 'local_statevector_simulator_py'}
{'available': True, 'name': 'local_unitary_simulator_py'}
```

```
In [2]: s = 9 # 1001
        oracle = QuantumCircuit(q, tmp, res)
        for i in range(len(q)):
            if (s & (1 << i)):
                oracle.cx(q[i],tmp[0])
```

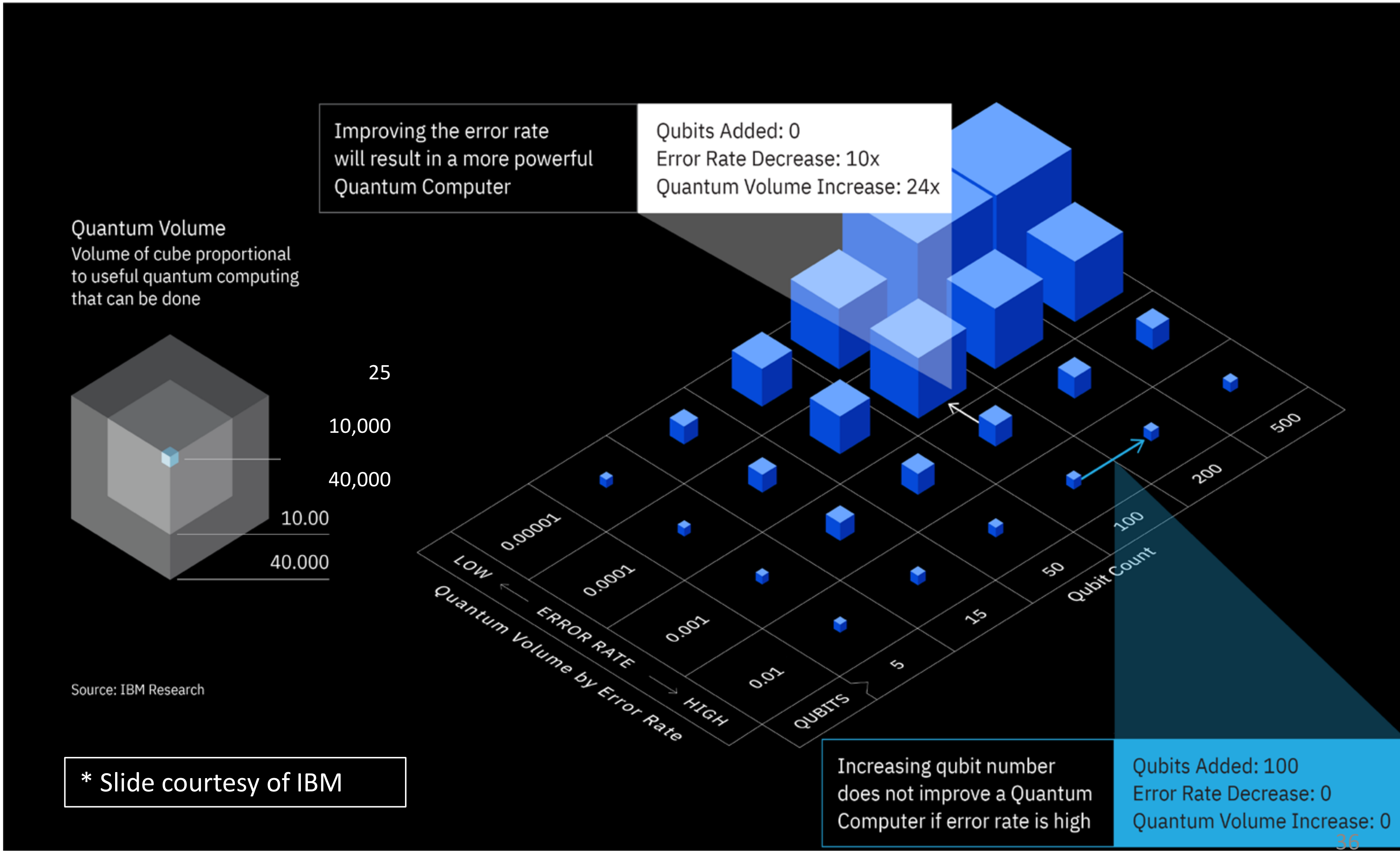
```
In [12]: result = qp.execute(('bv'), backend='ibmq_20_tokyo', timeout=300, shots=1024)
```

```
In [13]: # from qiskit.tools.visualization import plot_histogram
         counts = result.get_counts('bv')
         print(counts)
         plot_histogram(counts)
```

```
{'0101': 2, '1001': 723, '1101': 13, '1100': 3, '0000': 57, '0100': 2, '1010': 5, '1011': 96, '0011': 7, '1000': 46, '000
1': 62, '0010': 7, '1111': 1}
```



Number of Qubits versus Error Rate*



* Slide courtesy of IBM

Summary from a Hub Perspective Reflections on Practical Next Steps for HEP

- Need realistic near term perspective for transforming HEP problems/algorithms to QC ready formulations
  - QC spans multiple intellectual disciplines → strong need for educational outreach at all levels & across disciplines
  - Gain practical experience using QC simulators and hardware resources
  - QC HW facilities are scarce – capitalize on opportunities to build multidisciplinary research partnerships with cross pollination of skill sets
  - Work toward hybrid NISQ solutions in intermediate term (FTQEC not practical in near/intermediate future)

Thank You For Your Attention

Questions