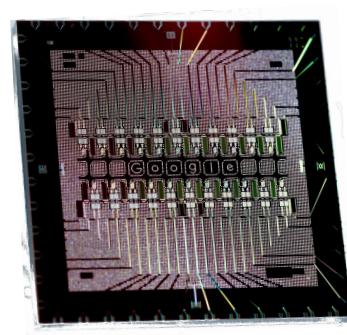


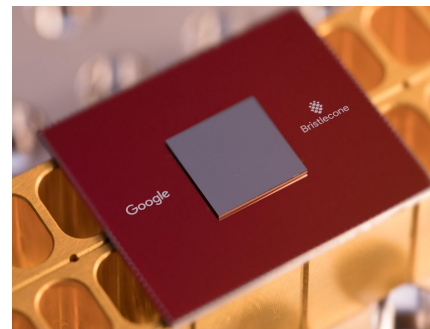
An Update on Google's Quantum Computing Initiative

November 6, 2018
kkissell@google.com

Google Cloud



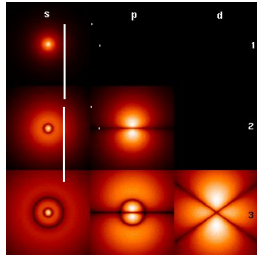
Cirq



Control of single quantum systems, to quantum computers

1 nm

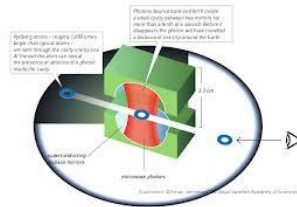
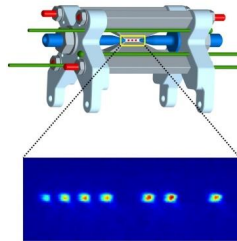
**H atom
wavefunctions:**



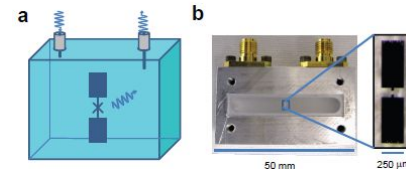
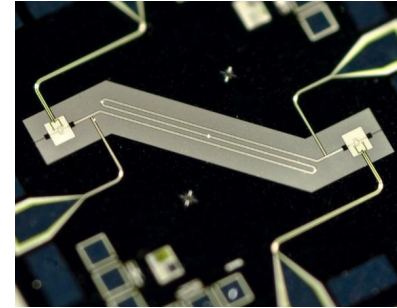
Problem:
Light is 1000x larger

Google Cloud

1 μm



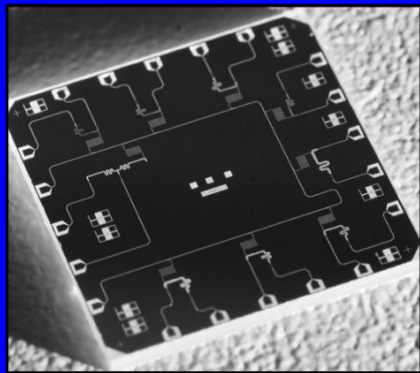
1 mm



Large “atom” has room for complex control

Confidential & Proprietary

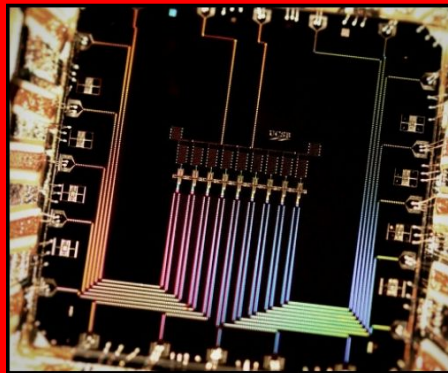
Quantum Chips at Google



Fluxmon

Flux qubit with tunable coupling

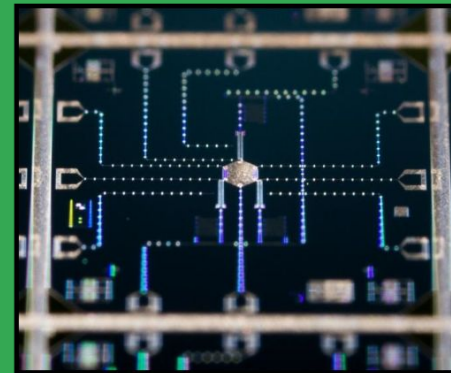
Good for optimization problems



Xmon

X-shaped transmon qubit

Good for building a digital, gate-based quantum computer (requires error correction)



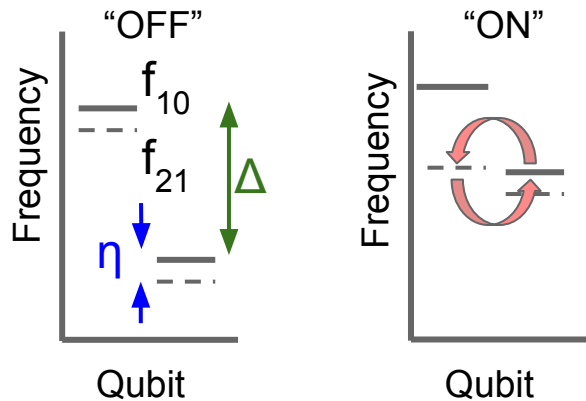
Gmon

Transmon qubit with tunable nearest-neighbor coupling

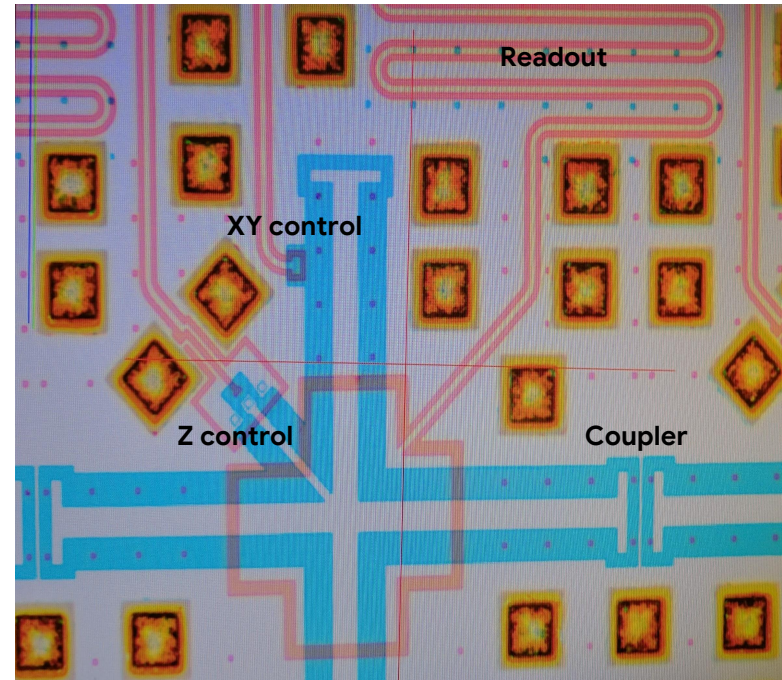
Good for simulation problems

Xmon: Direct coupling + Tunable Transmons

- Direct qubit-qubit capacitive coupling
- Turn interaction on and off with frequency control



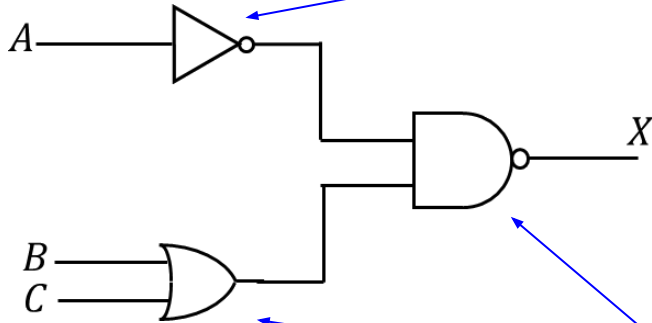
$$\text{Coupling rate } \Omega_{zz} \approx 4\eta g^2 / \Delta^2$$



Logic Built from Universal Gates

Classical circuit:

1 bit NOT
2 bit AND
Wiring fan-out



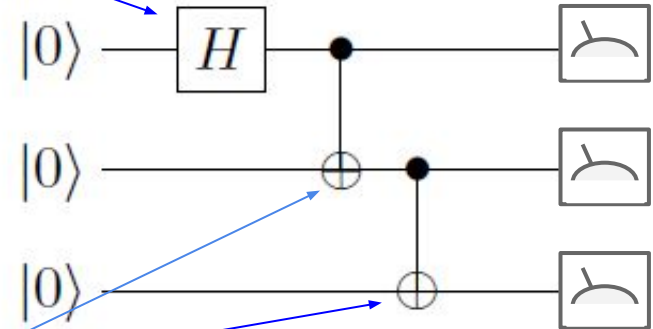
1 Input Gates

2 Input Gates

(space+time)

Quantum circuit:

1 qubit rotation
2 qubit CNOT
No copy



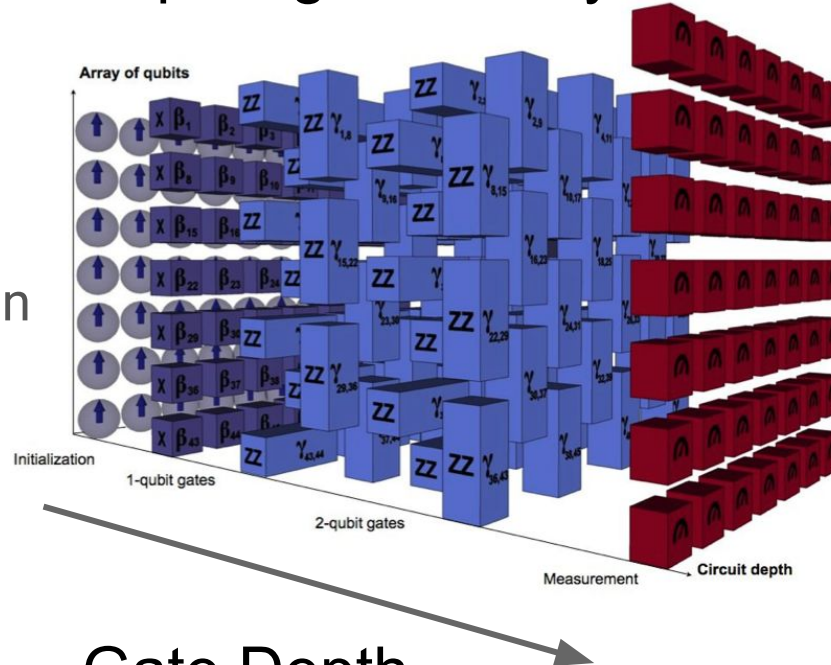
time

Space-Time Volume of a Quantum Gate Computation

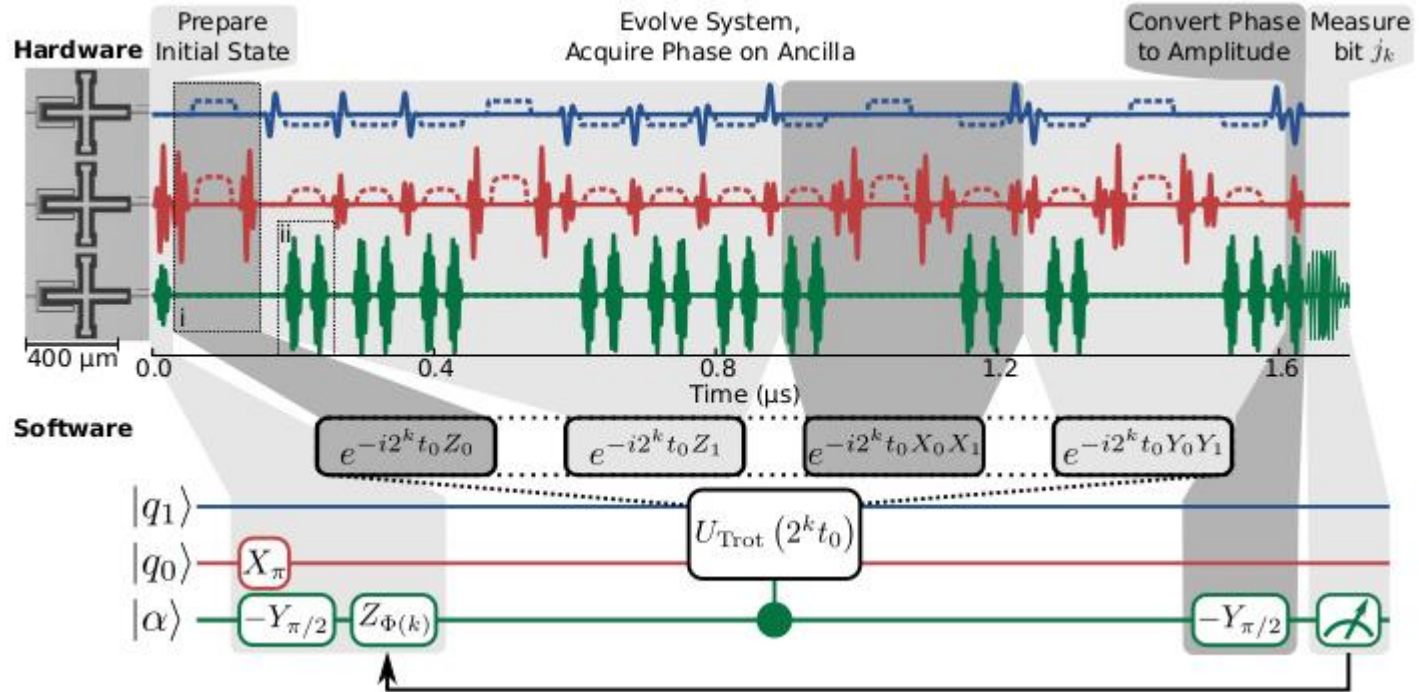
Uncorrected Gate
“Circuits” Limited by
Fidelity of Operations
and Decoherence Times

Fidelity is the Third Dimension

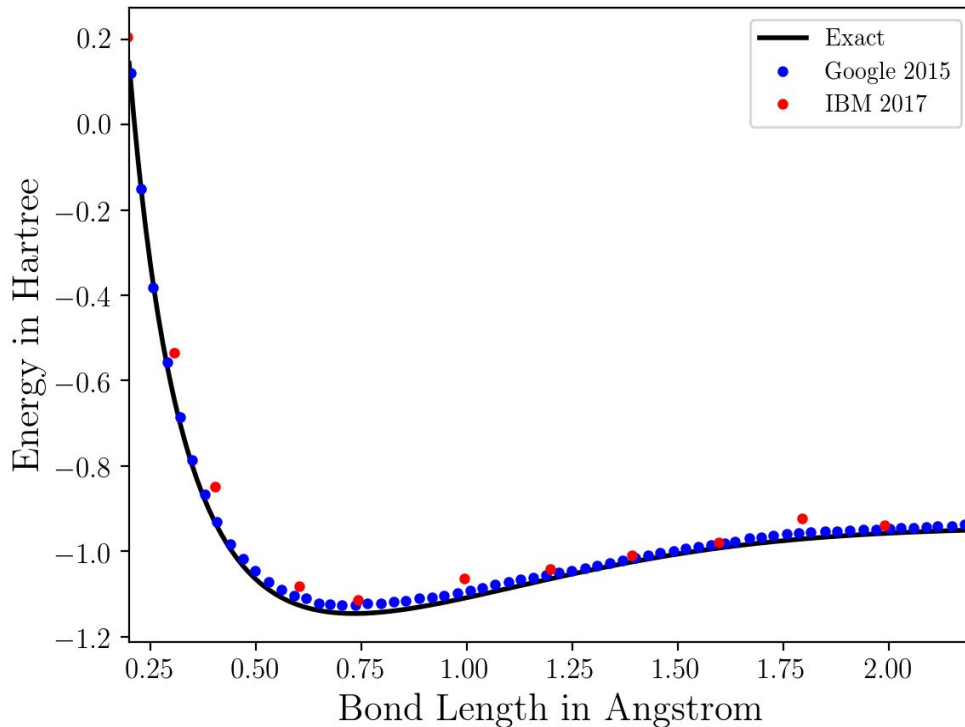
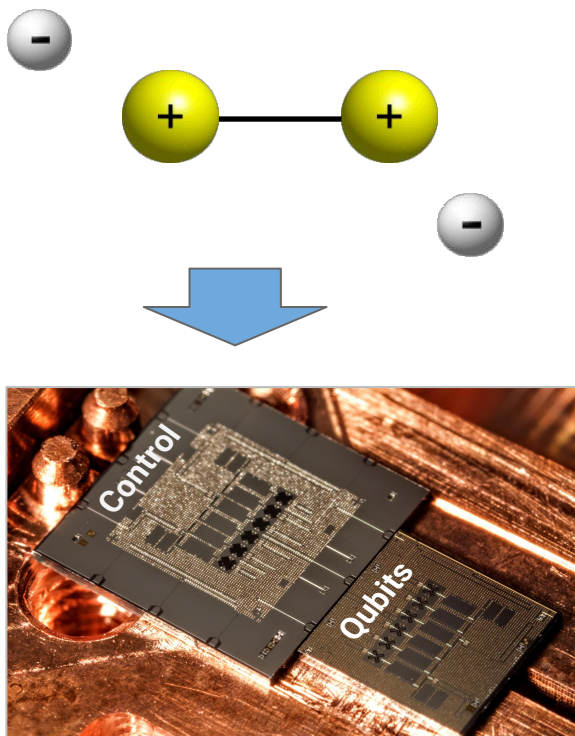
2 qubit gate fidelity = 99.5%



Execution of a Quantum Simulation



Quantum Simulation Results, H₂ Molecule



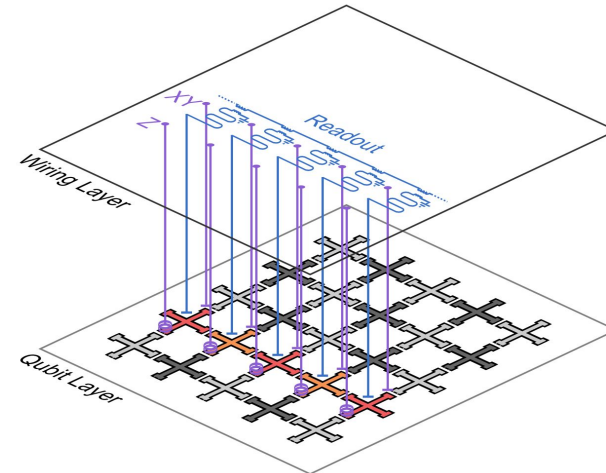
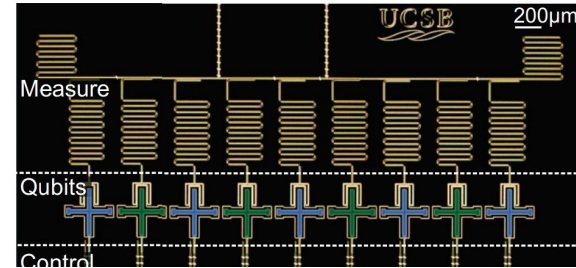
9 Qubit: Good performance, Limited Scaling

9 qubit device has good performance

- Err_{CZ} down to 0.6%
- $\text{Err}_{\text{SQ}} < 0.1\%$
- $\text{Err}_{\text{RO}} = 1\%$

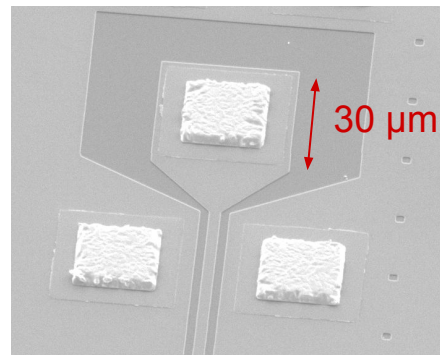
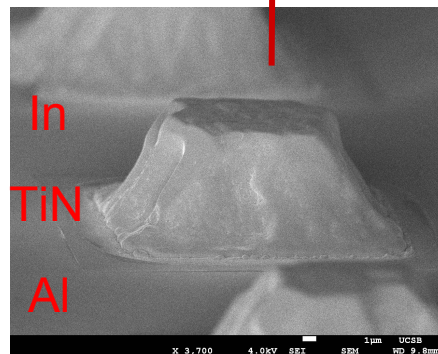
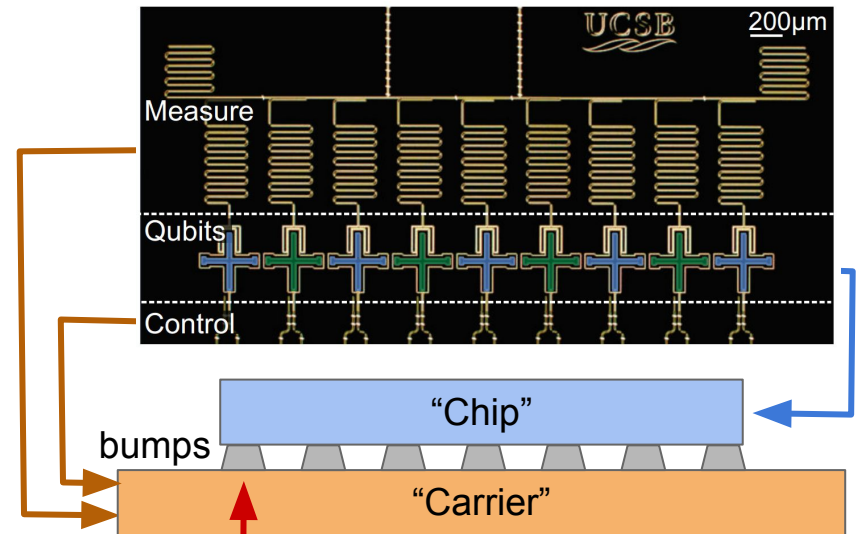
Limited to 1D connectivity (planar geometry)

Scale-up strategy: move qubits, control to different planes



Bump-Bond Architecture

- Bond together two separate chips
 - Qubits → “Chip”
 - Control → “Carrier”
- Superconducting interconnect
- Use lossless vacuum as dielectric



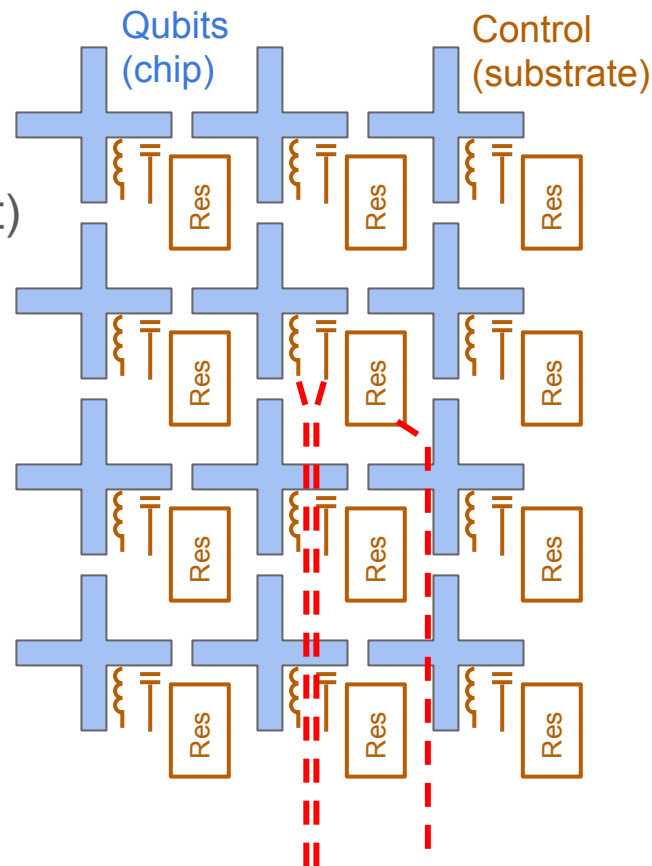
Scaling to 2D

Design must be “tileable” (control fits in qubit footprint)

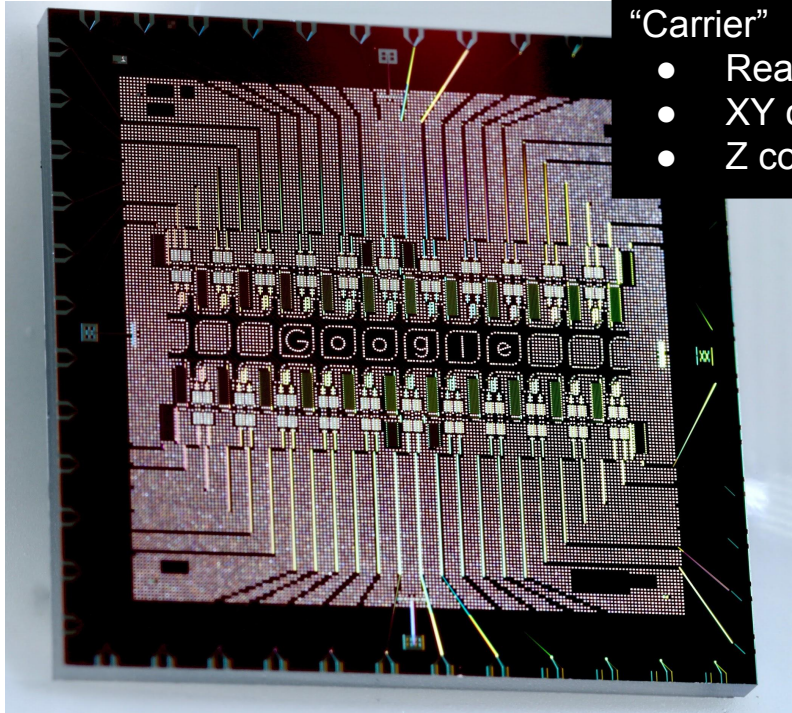
- Readout resonator
- XY coupler
- SQUID coupler

Need to shield qubits from interior wire routing

- Small coupling to 50Ω line will decohere qubit

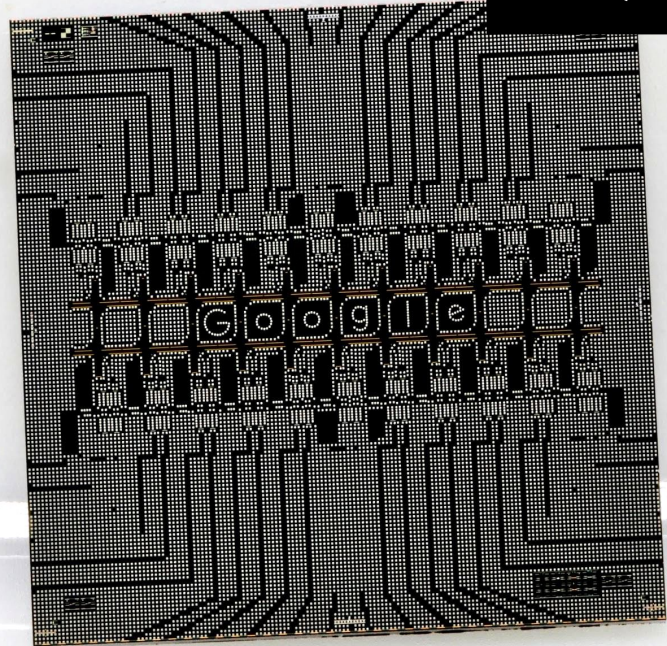


“Foxtail” 22 Qubit Device



“Carrier”

- Readout
- XY control
- Z control



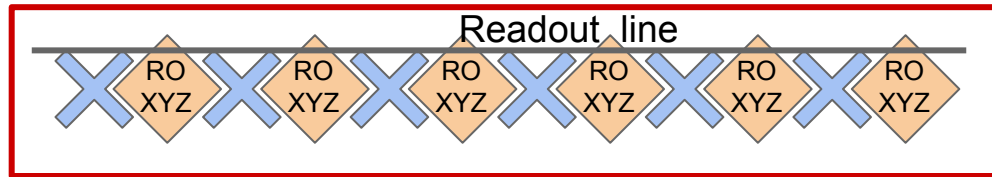
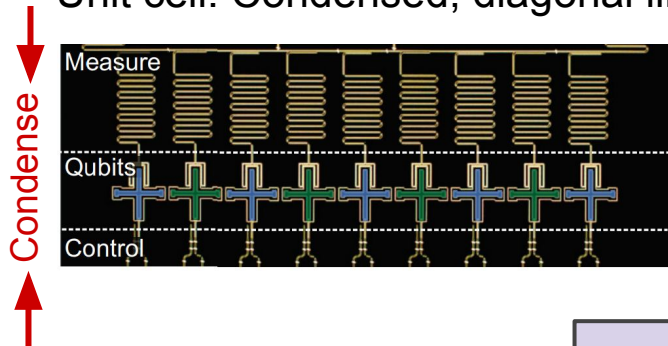
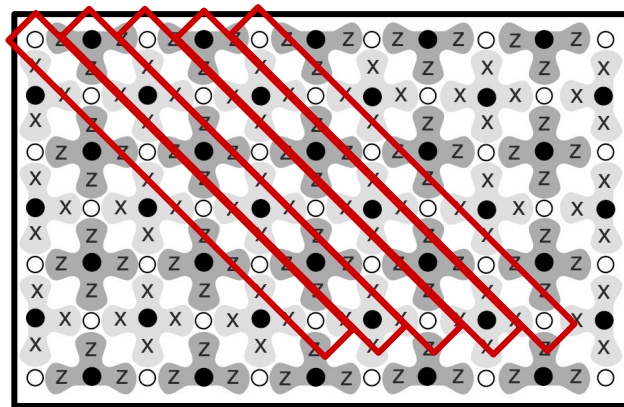
“Chip”

- Qubits

2D Unit Cell

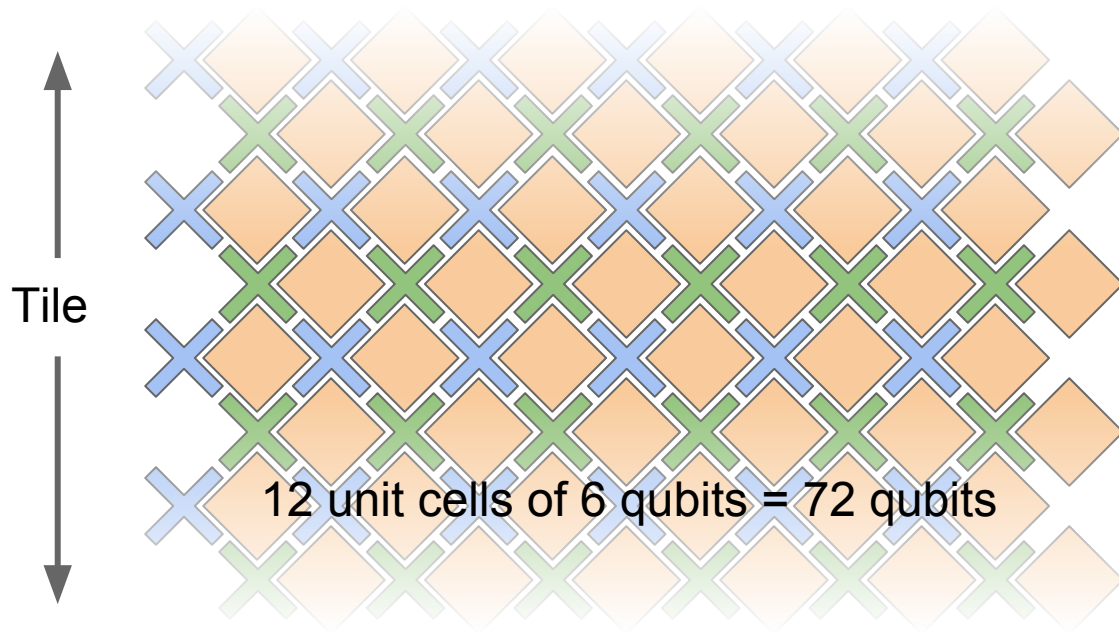
- Diagonal for surface code:
all “measure” qubits on same line
- Condense footprint across 2 chips
- Introduce shielded wiring between qubits
- Tile unit cell for 2D array

Unit cell: Condensed, diagonal linear chain



Unit cell designed for surface code

“Bristlecone” Architecture



Bristlecone





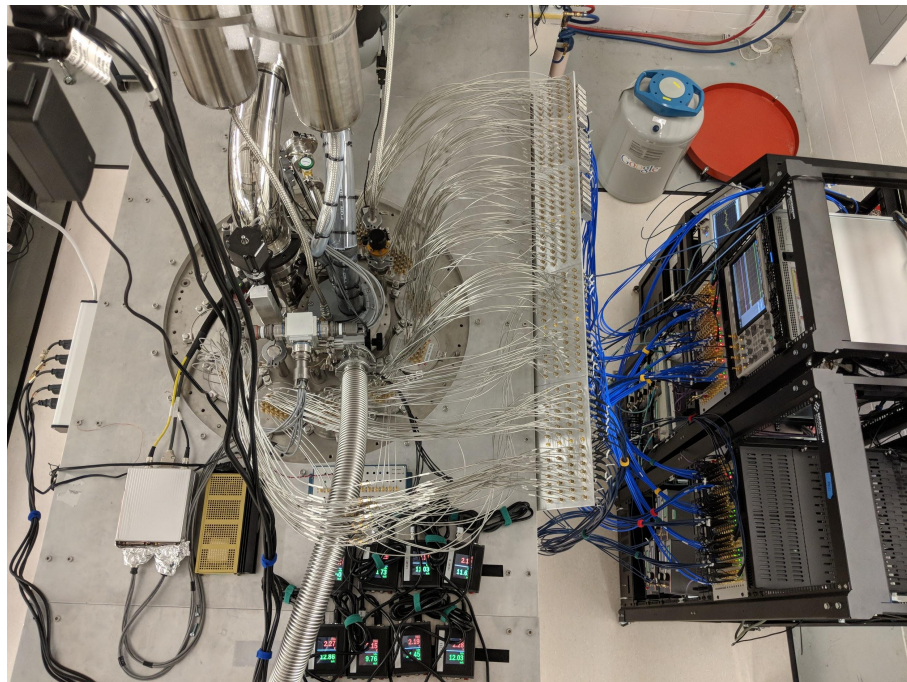
Google


Bristlecone

Photo: Erik Lucero

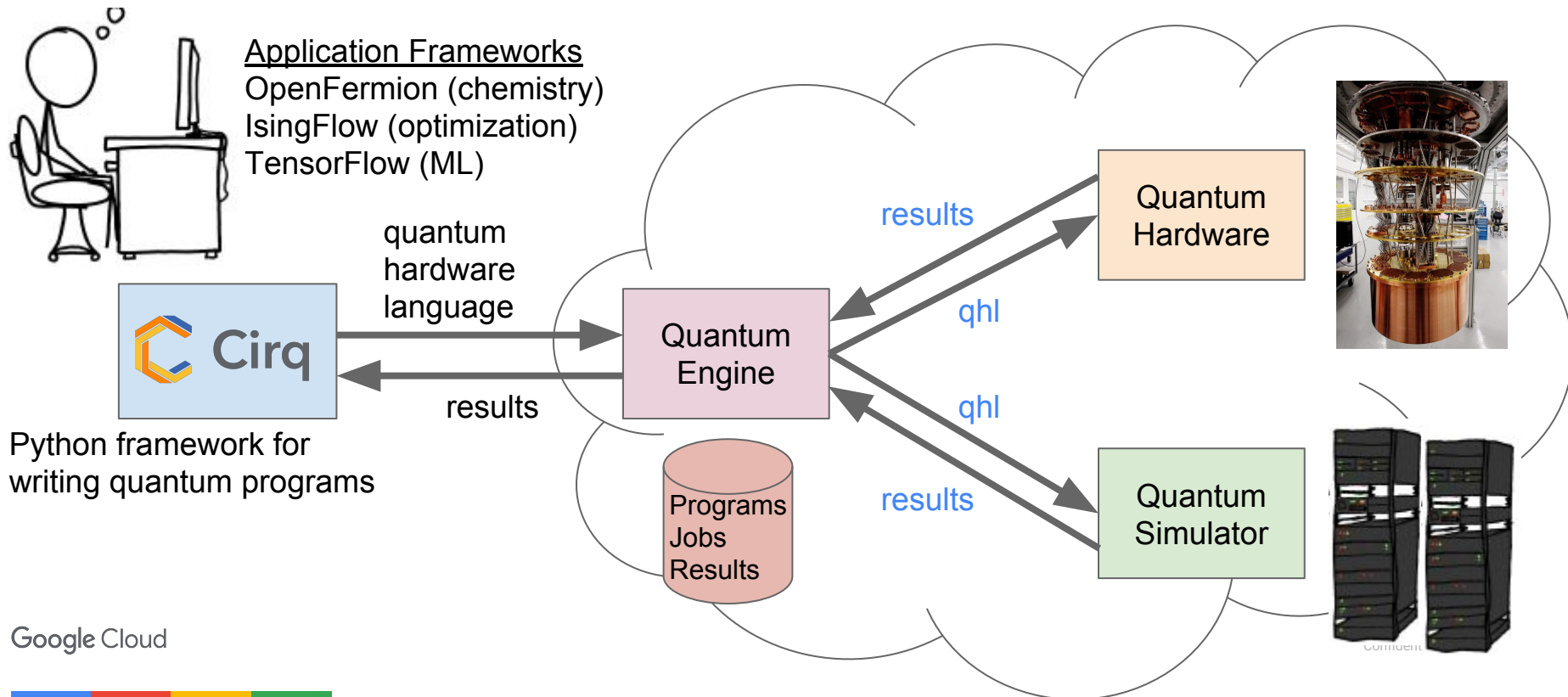
Google Cloud

“72 qubits cold in fridge”



C'est quoi ce Cirq?

Cloud Quantum Computing Workflow

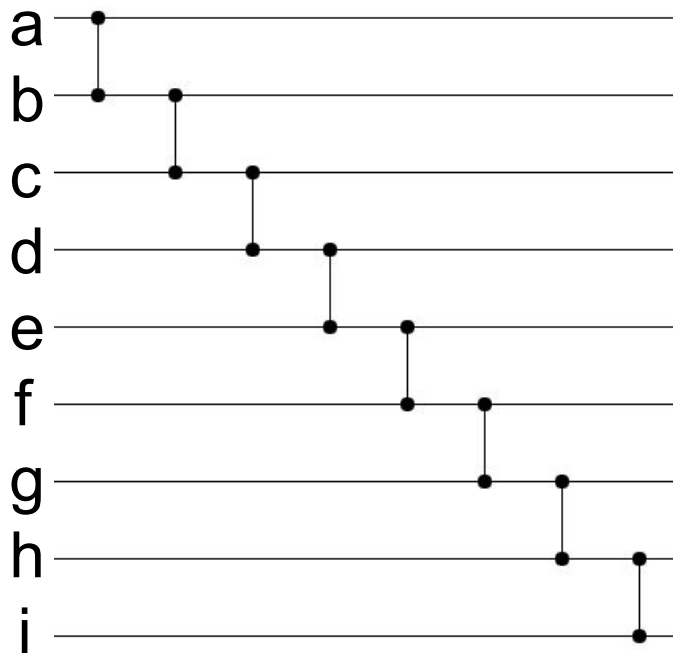


Is this a good quantum circuit?

CZ	a	b
CZ	b	c
CZ	c	d
CZ	d	e
CZ	e	f
CZ	f	g
CZ	g	h
CZ	h	i

Is this a good quantum circuit?

CZ a b
CZ b c
CZ c d
CZ d e
CZ e f
CZ f g
CZ g h
CZ h i



No: large depth

Is this a good quantum circuit? Attempt #2

CZ a b

CZ c d

CZ e f

CZ g h

CZ b c

CZ d e

CZ f g

CZ h i

Is this a good quantum circuit? Attempt #2

CZ a b

CZ c d

CZ e f

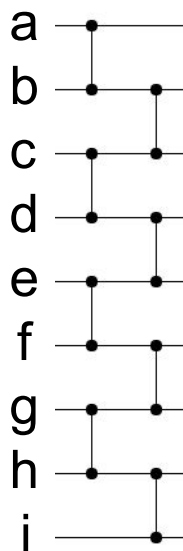
CZ g h

CZ b c

CZ d e

CZ f g

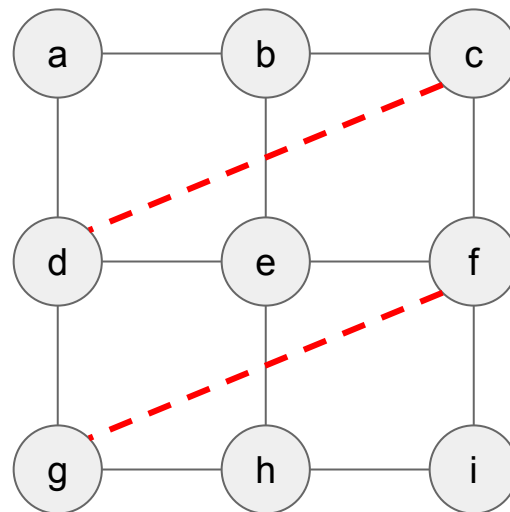
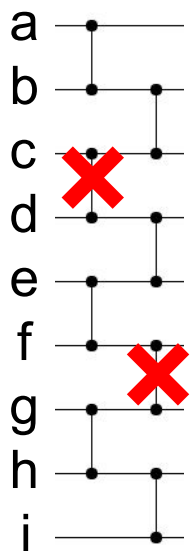
CZ h i



Is this a good quantum circuit? Attempt #2

CZ a b
CZ ~~c d~~
CZ e f
CZ g h

CZ b c
CZ d e
CZ ~~f g~~
CZ h i



No: Does Not Map to
Physical Topology

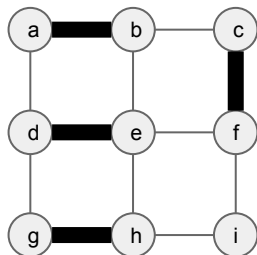
Is this a good quantum circuit? Attempt #3

CZ a b
CZ c f
CZ e d
CZ g h

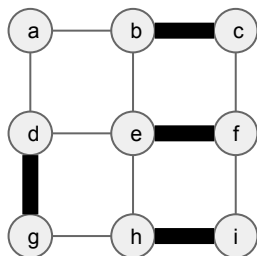
CZ b c
CZ f e
CZ d g
CZ h i

Is this a good quantum circuit? Attempt #3

CZ a b
CZ c f
CZ e d
CZ g h

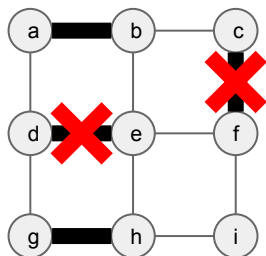


CZ b c
CZ f e
CZ d g
CZ h i

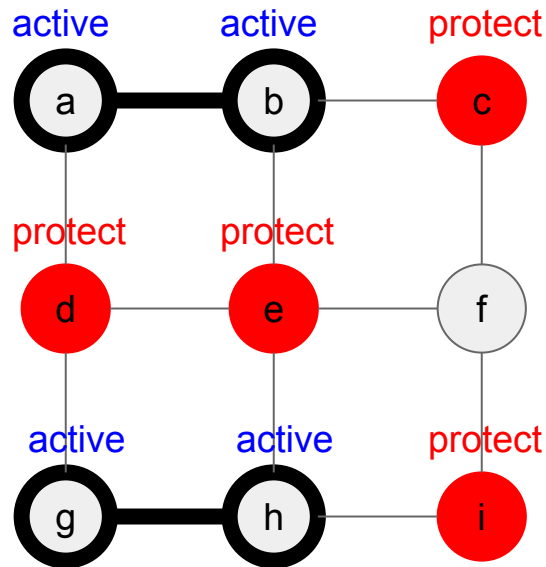
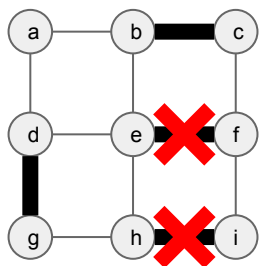


Is this a good quantum circuit? Attempt #3

CZ a b
CZ c f
CZ e d
CZ g h

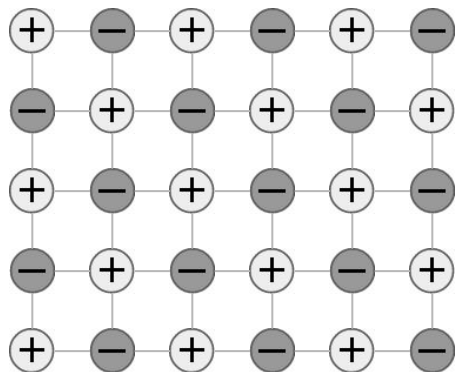


CZ b c
CZ f e
CZ d g
CZ h i



Maybe. But in some devices
2-qubit gates can't be adjacent.

CZs on an Xmon Device



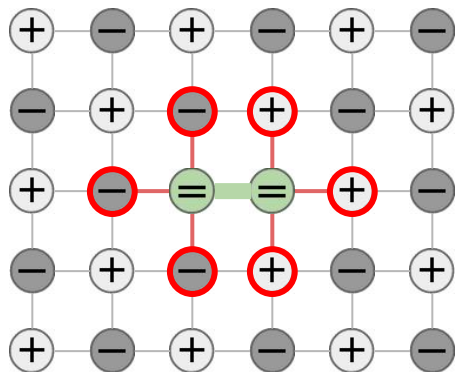
Neighboring qubits are always interacting

Interaction is much stronger when frequencies closer

Keep neighboring qubits' frequencies far apart when idling

Thus checkerboard pattern of high/low frequencies

CZs on an Xmon Device



Meet frequencies in the middle for awhile to perform a CZ

Move nearby qubit freqs even further away to reduce error

Can't do two CZs next to each other; causes extra interactions

Takes 8 layers to perform a CZ along every edge

Hardware-Agnostic Languages for NISQ?

Hardware control

Mix of industry tools
and proprietary

Assembly languages

OpenQASM
Quil
aQasm

Frameworks

PyQuil
QISKit
ProjectQ

Languages

Q#



higher levels of abstraction

Cirq is built in the belief that NISQ programming tools need to be hardware aware, not hardware agnostic.



- An open source Python framework for writing, optimizing, and running quantum programs on near term hardware.

```
import cirq

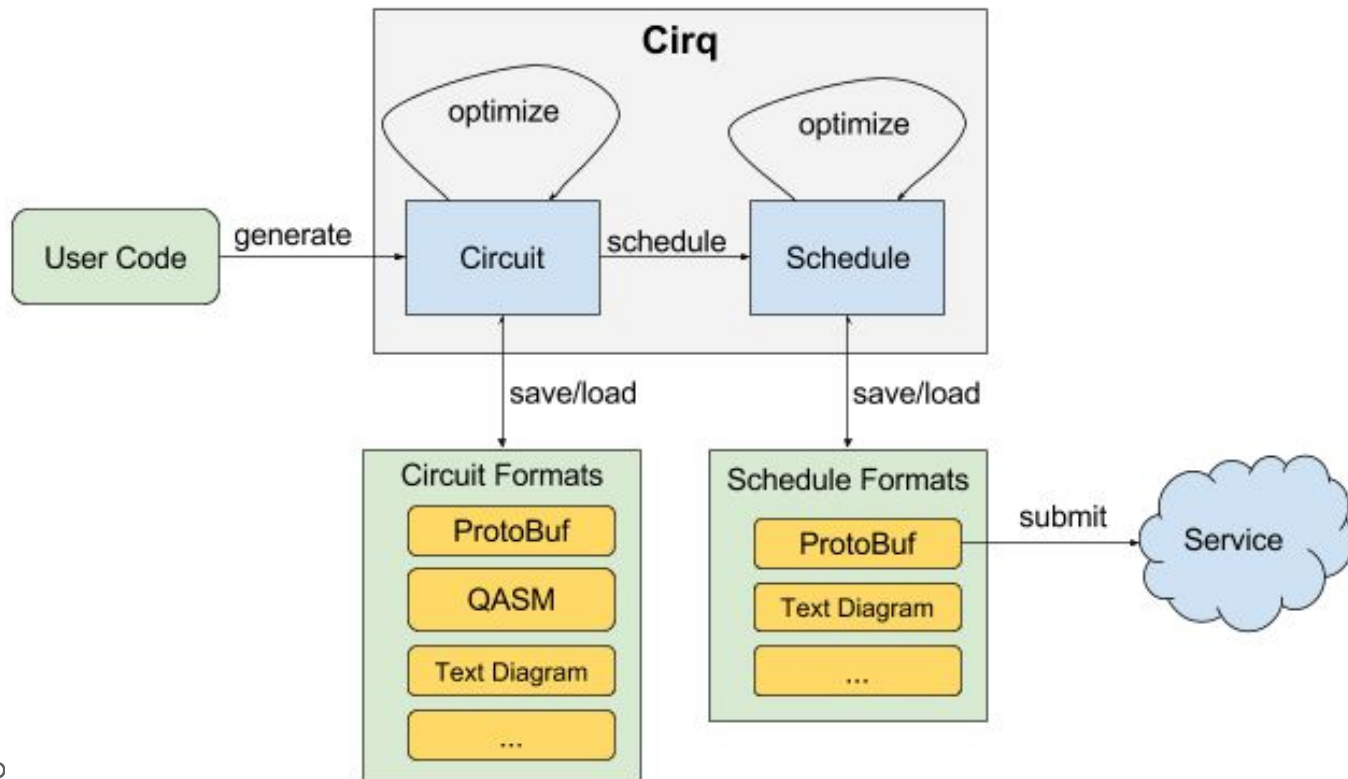
# Define a qubit.
qubit = cirq.GridQubit(0, 0)

# Create a circuit (qubits start in the |0> state).
circuit = cirq.Circuit.from_ops(
    cirq.X(qubit)**0.5, # Square root of NOT.
    cirq.MeasurementGate('result').on(qubit) # Measurement.
)

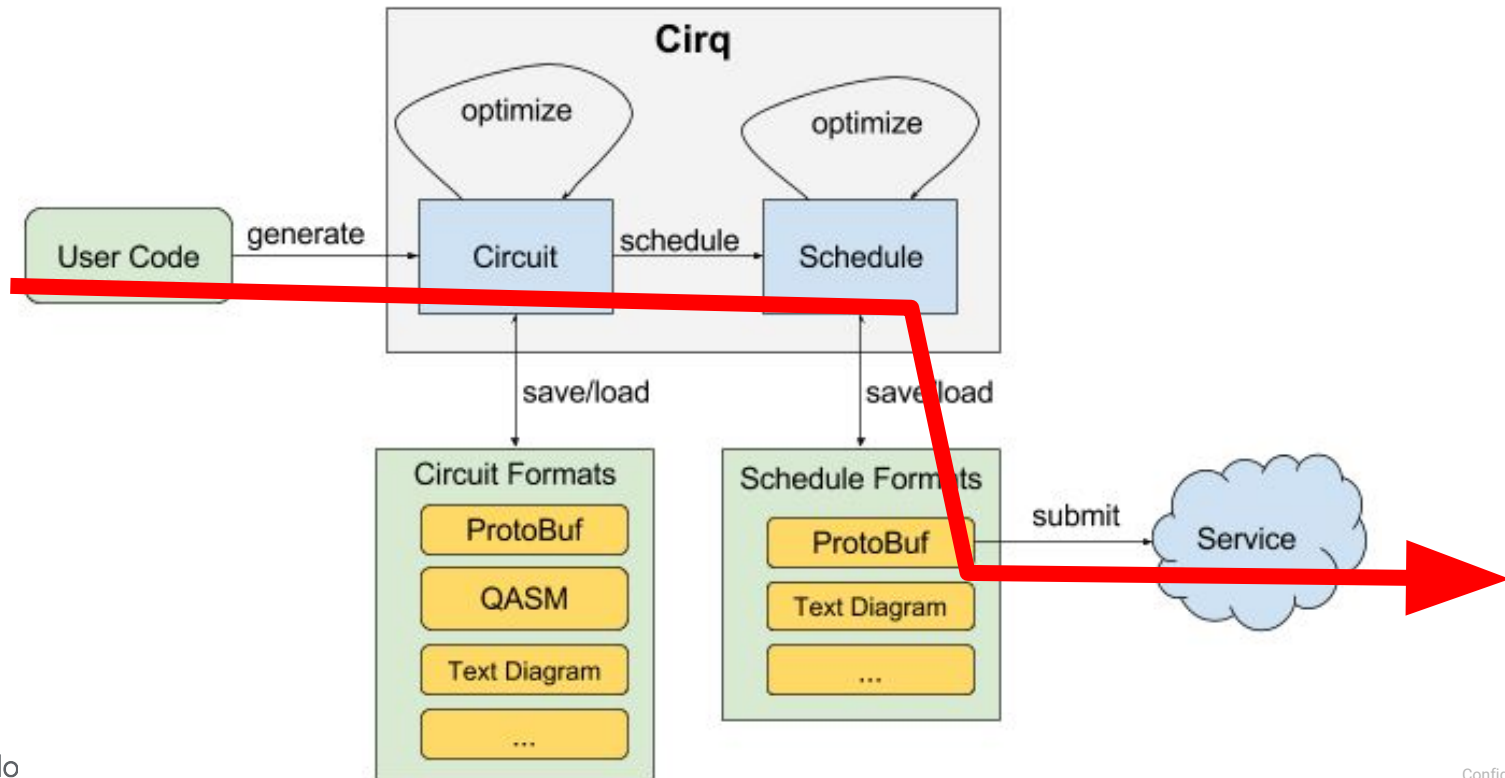
print(circuit)
```

⌕ (0, 0): —X^{0.5}—M—

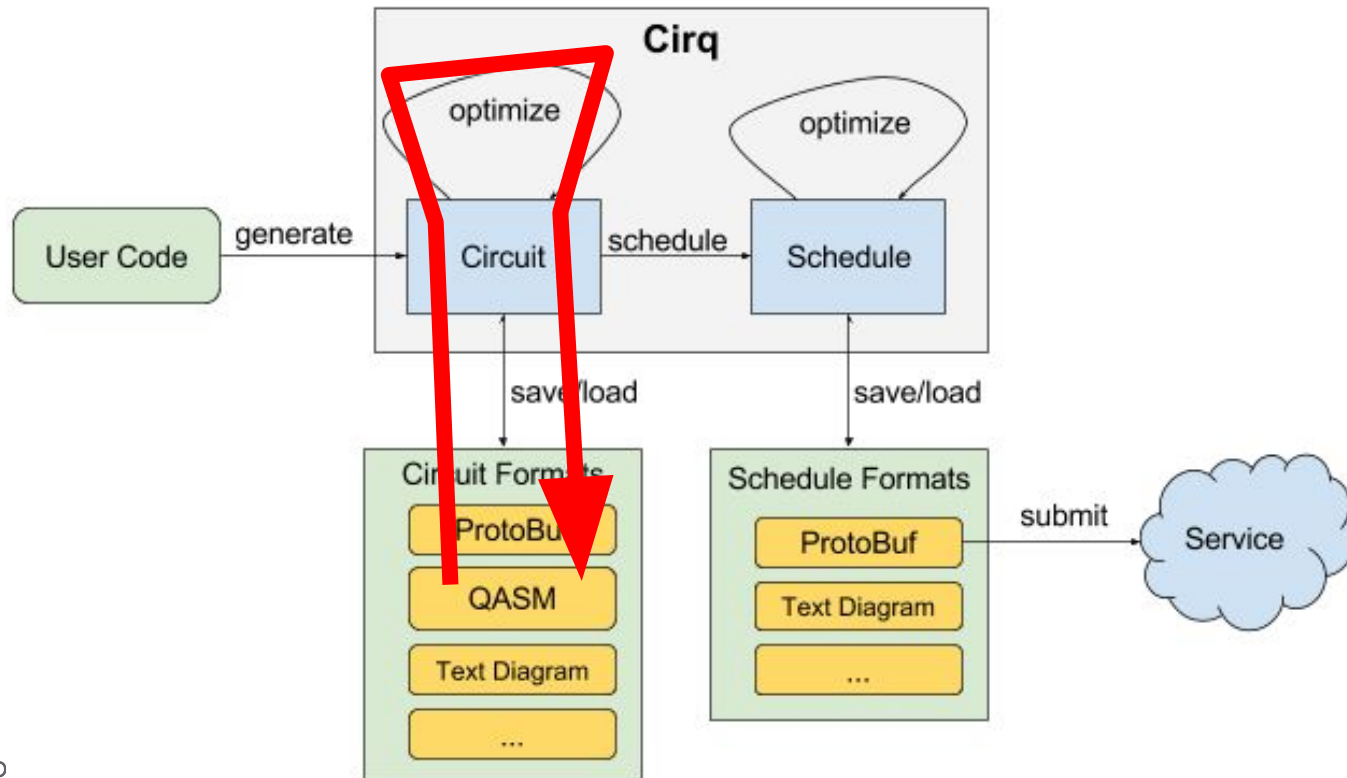
Cirq Structure



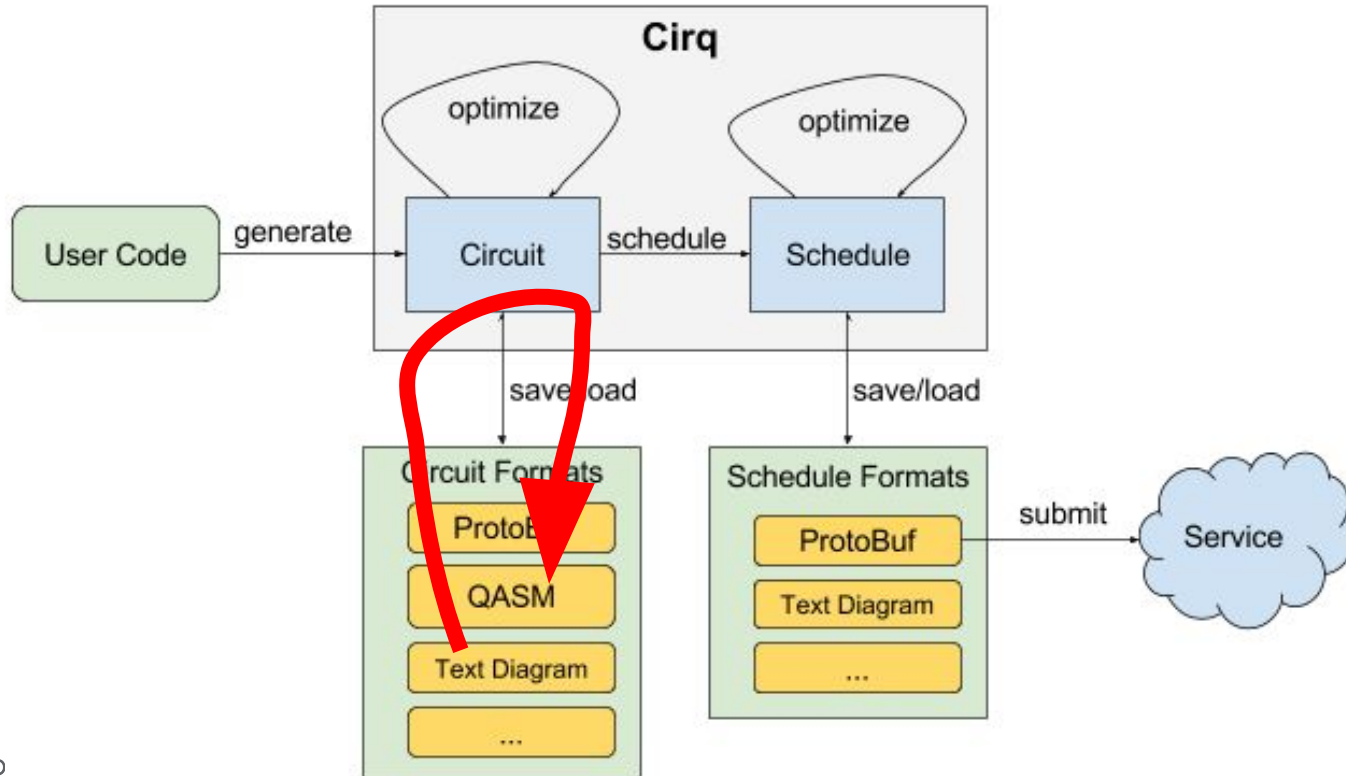
Use Case: Quantum Program Writer



Use Case: Optimizer

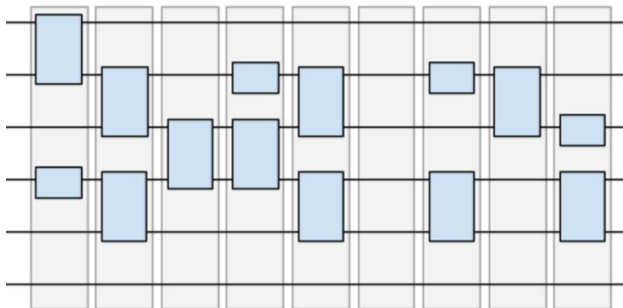


Use Case: Transcoder



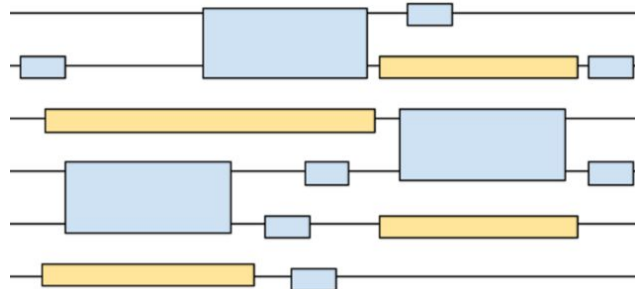
Workhorses: Circuit & Schedule

Circuit



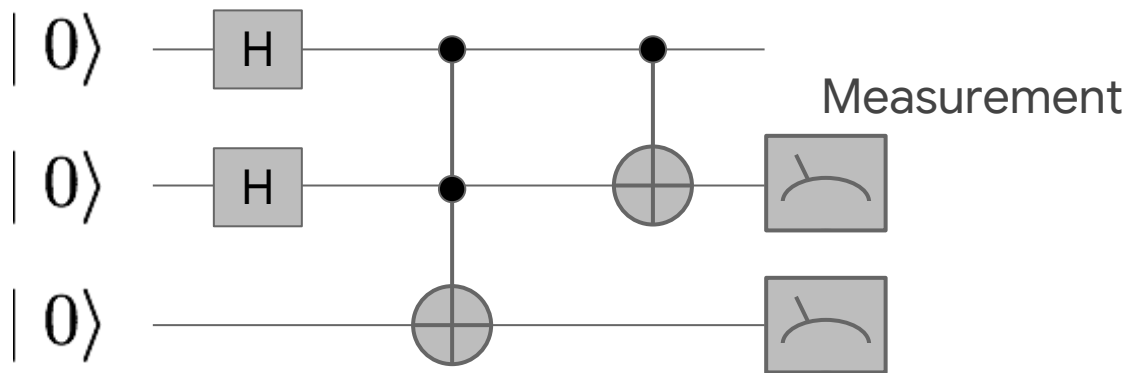
- Discretized
- A **Circuit** is a list of **Moments**
- **Moments** are made up of **Operations**
- An **Operation** is a **Gate** plus **Qubits** it acts upon
- Ignores timing and durations
- Can enforce some **Device** constraints (WIP)

Schedule



- Continuous
- A **Schedule** is made up of **ScheduledOperations** plus a **Device**
- A **ScheduledOperation** is an **Operation** plus a start time and duration
- Timing explicit
- **Device** contains all of the constraints for hardware

1-bit Calculator



$$0 + 0 = ?$$

$$0 + 1 = ?$$

$$1 + 0 = ?$$

$$1 + 1 = ?$$

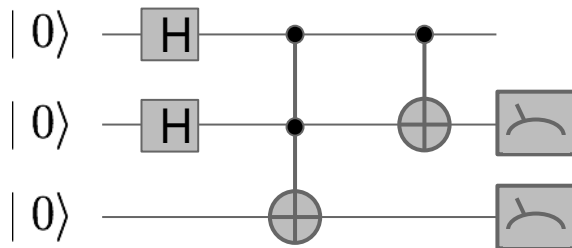
This circuit executes four calculations simultaneously

The 1-bit Calculator in Cirq - Build a Circuit

```
import matplotlib.pyplot as plot
from pandas import DataFrame
import cirq
from cirq.ops import CNOT, TOFFOLI
runs = 1000
# Create 3 qubits in a line
q1 = cirq.GridQubit(0,0)
q2 = cirq.GridQubit(0,1)
q3 = cirq.GridQubit(0,2)
# Create a circuit for the qubits
circuit = cirq.Circuit.from_ops(
    cirq.H(q1), cirq.H(q2), # Start with H gates on q1 and q2
    TOFFOLI(q1,q2,q3), CNOT(q1,q2),
    cirq.measure(q2, key='m1'), cirq.measure(q3, key='m2'))
print("Circuit:")
print(circuit)
```

Circuit:

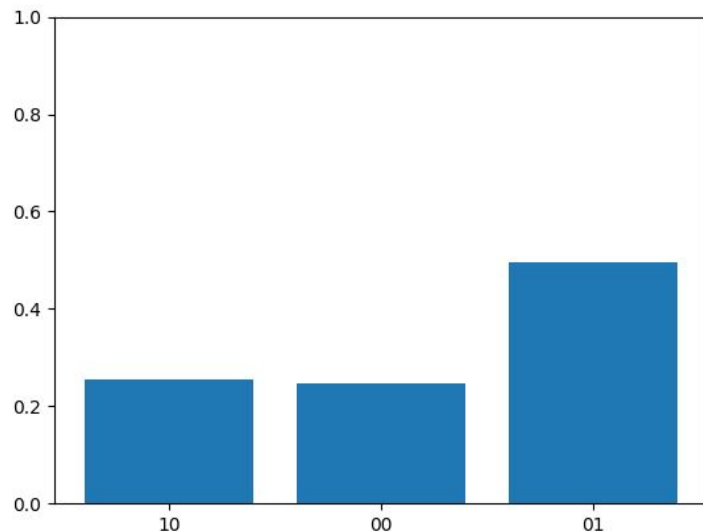
```
(0, 0): —H—@—@—————
          |   |
(0, 1): —H—@—X—M('m1')—
          |
(0, 2): —X—M('m2')—
```



The 1-bit Calculator in Cirq - Simulate and Sample

```
# Instantiate a simulator and run the circuit
simulator = cirq.google.XmonSimulator()
result = simulator.run(circuit, repetitions=runs)
summary = {'00':0, '01':0, '10':0}
for m1, m2 in zip(result.measurements['m1'], result.measurements['m2']):
    if m1[0] and not m2[0]:
        summary['01'] += 1.0 / runs
    elif not m1[0] and m2[0]:
        summary['10'] += 1.0 / runs
    else:
        summary['00'] += 1.0 / runs

print()
print('Result:')
fig = plot.figure()
subplot = fig.add_subplot(111)
subplot.set_xticks(range(3))
subplot.set_ylim([0, 1.0])
subplot.bar(range(3), summary.values())
_ = subplot.set_xticklabels(summary.keys())
plot.show()
```

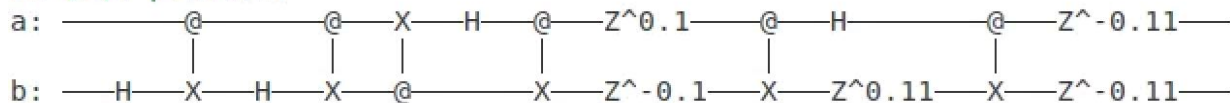


Circuit Optimization in Cirq

```
In [1]: from optimization_demo import *
```

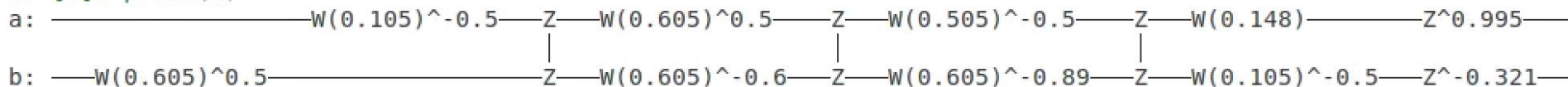
```
In [2]: c = make_inefficient_circuit()
```

```
In [3]: print(c)
```



```
In [4]: optimize(c)
```

```
In [5]: print(c)
```



6 → 3 Two-input Ops

Thanks for Your Attention!

Resources:

 **Cirq** <https://github.com/quantumlib/Cirq>

Open \rangle \langle
Fermion

<https://github.com/quantumlib/OpenFermion>

Google Cloud

