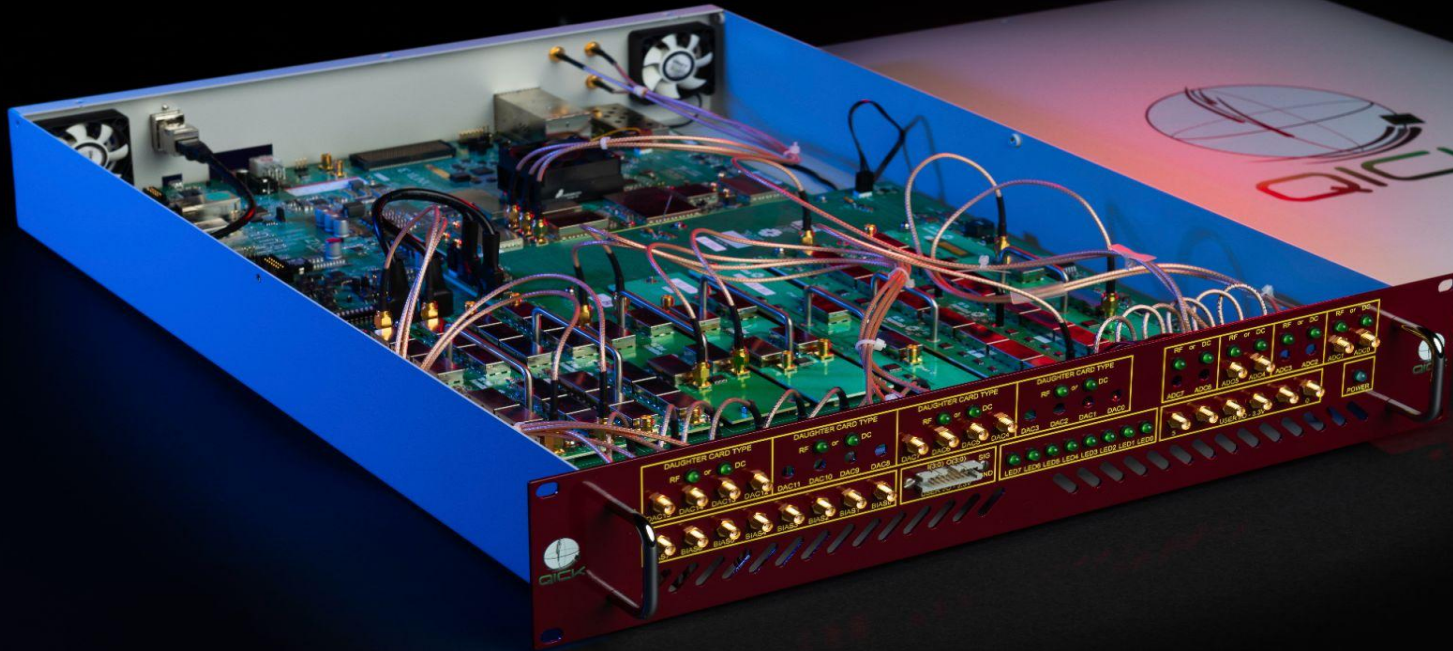


Quantum Computing and Superconducting Qubits

Sho Uemura, Fermilab



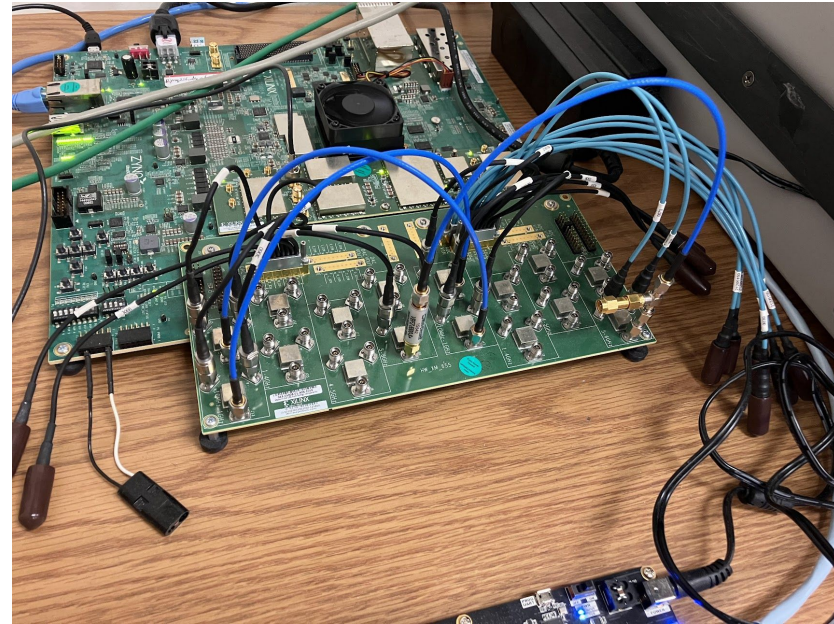
A tutorial on quantum computing

- Quantum computing: why people care, the current state of the art, how it works from an RF perspective, what it has to do with feedback
- With heavy personal biases:
 - My qubit controller: Fermilab's QICK (Quantum Instrumentation Control Kit)
 - But see also LBL's QubiC (Qubit Control)
 - QICK's preferred qubit type: superconducting qubits
 - I will be loose with details because I'm describing a big field of which my actual role is a very small part



Why this tutorial?

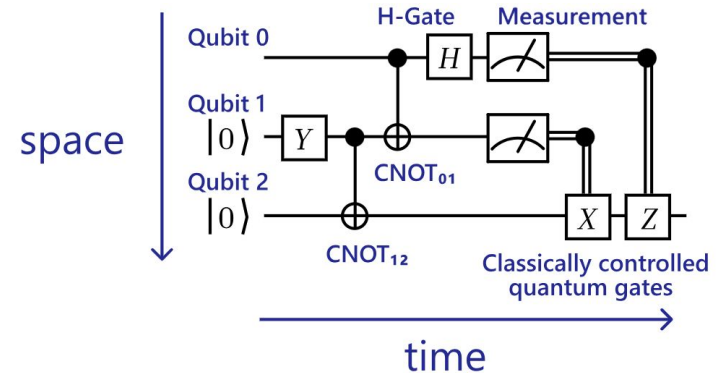
- RFSoc hardware platform: useful for both LLRF and quantum
- Heritage: the QICK and QubiC teams are both originally LLRF groups
- Feedback??
 - Error correction: “digital” not continuous, but comparable latency requirements
 - Optimal control: open loop, but uses system modeling and control theory



What do we need for a quantum computer?

- Qubits: a quantum system with two (or more) addressable energy levels, scalable to large qubit counts
- Initialization: a way to initialize qubit states
- Coherence: qubit states must live long enough for computation
- Gates: a “universal” set of gates that can be used to build arbitrary multi-qubit algorithms
 - Arbitrary single-qubit rotations, plus a two-qubit entangling operation
- Measurement: a way to measure the qubit state

“DiVincenzo
criteria”



Why is QC useful?

- Heuristic arguments:
 - Exponentially larger state space: N qubits are described by 2^N complex numbers
 - “Trying all paths at once”: run a classical computation on a superposition of inputs, get a superposition of outputs
 - Simulating a quantum system should be easier with a quantum computer
- Specific algorithms have big speedups over classical computing
 - Shor’s algorithm for prime factorization (breaking RSA encryption): exponential speedup
 - Grover’s algorithm for search: quadratic speedup

◆ AI Overview

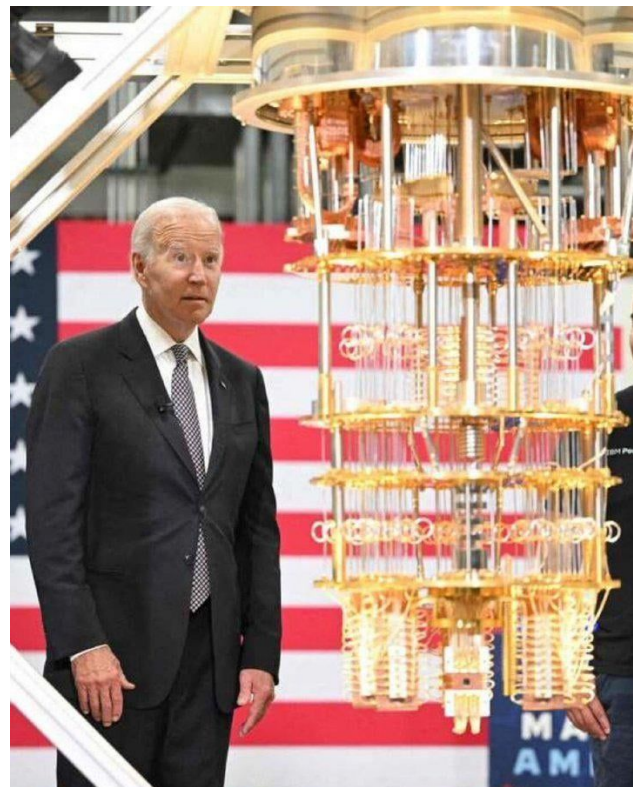
SHOR’S ALGORITHM ICONS IN VARIOUS STYLES V02



Licensed by Google

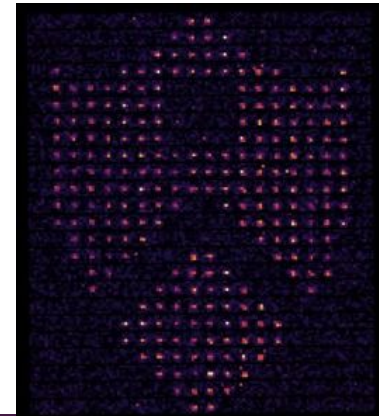
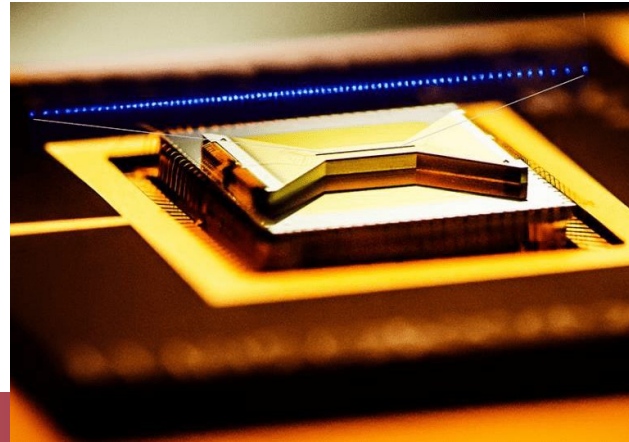
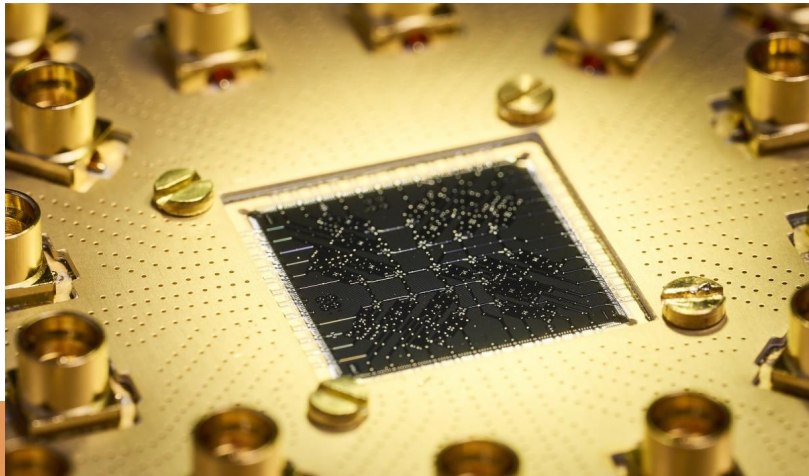
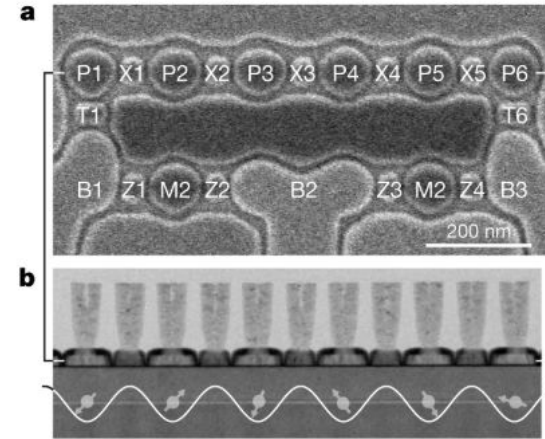
Quantum computing: why?

- Different applications
 - Quantum simulation
 - Optimization/minimization/search
 - Cryptography
 - Sensing
- Interest in different communities for different reasons
 - Government: cryptography? prestige/control?
 - Industry: finance? pharmaceuticals? hype?
 - Academia: HEP? sensing? fun?



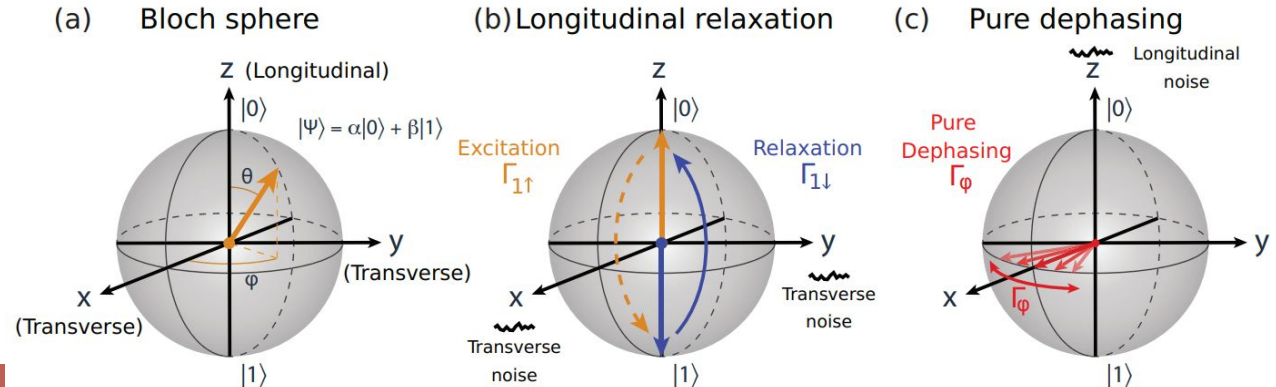
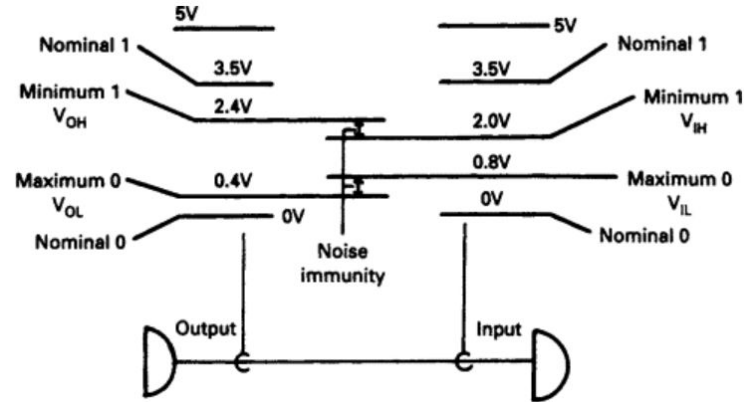
Qubit types

- Not an exhaustive list of modalities:
 - Ions in electromagnetic traps
 - Neutral atoms in optical traps
 - Spins (electrons) in semiconductors
 - Superconducting circuits
- Differences in coherence, speed, scalability, complexity of control . . .



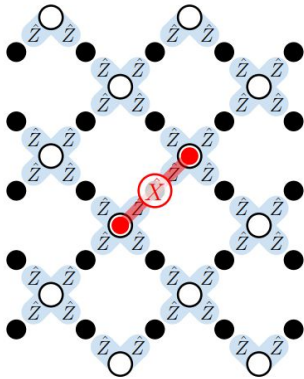
Why is QC hard? Errors!

- A classical digital bit relies on constantly measuring+discretizing an analog signal
- A qubit has more information than you can measure: measurement projects each qubit onto a digital value and destroys entanglement
- You have no noise immunity, so errors accumulate during computation

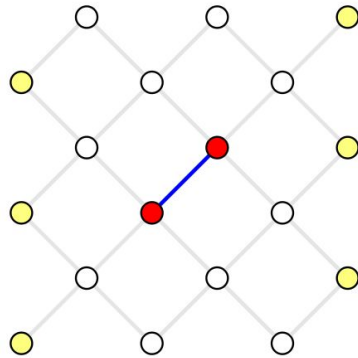


Quantum error correction

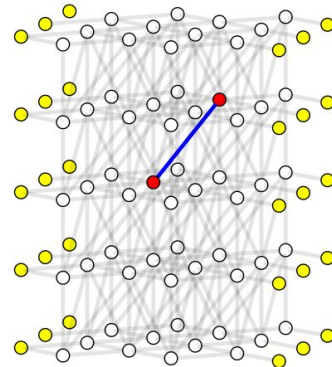
- Use many physical qubits to build a “logical qubit” with lower error rate
- Example here of a “surface code”: one logical qubit is a $d \times d$ array of data qubits
 - Ancilla qubits (white) check the data qubits (black): repeat d times
 - Decode the syndrome to identify the flipped data qubit(s), apply the fix
- Overhead on qubits: for $d=5$, one logical qubit = 49 physical qubits (25 data qubits and 2×12 ancillas)
- Overhead on computation: a two-qubit gate on two logical qubits needs to involve all of the physical qubits
- Decoding is hard: more complicated syndromes lead to ambiguity
- Everything (ancilla measurements, decoding, correction) must happen faster than decoherence



(a) Surface Code $d = 5$.



(b) Decoding Graph.



(c) 3D Decoding Graph.

Milestones

- Working QEC: a logical qubit less noisy than its underlying physical qubits, or a logical qubit that gets less noisy with size
 - We are here
- Fault tolerance: QEC becomes worth doing
 - Not yet, you'd rather have 100 noisy physical qubits than 1 somewhat-less-noisy logical qubit - the “noisy intermediate-scale quantum computing (NISQ)” era
 - We need better qubits, and more of them
- Quantum advantage: doing something with a quantum computer that you can't do with a classical computer
 - It doesn't have to be something useful . . . are we here yet? There have been claims
- Doing something useful? Nah

The screenshot shows the top of the Quantum Advantage Tracker website. At the top, there's a navigation bar with the title "Quantum Advantage Tracker", a dropdown menu for "Advantage trackers", a "Participate" button, and a logo. Below the navigation bar, the main heading "Benchmarking quantum advantage" is displayed in a large, bold font. Underneath the heading, a paragraph states: "As claims of quantum advantage emerge, this project provides a platform-agnostic framework to collect, validate, and compare results." Further down, it mentions "Contributors include researchers from 15+ organizations". At the bottom, there are logos for "UChicago" and "University of Chicago" separated by a green dot.

Quantum Advantage Tracker

Advantage trackers ▾ Participate

Benchmarking quantum advantage

As claims of quantum advantage emerge, this project provides a platform-agnostic framework to collect, validate, and compare results.

Contributors include researchers from 15+ organizations

UChicago • University of Chicago

Performance metrics

- Logical qubit count, error rate/number of reliable operations, speed
- Ability to run useful algorithms
 - Some algorithms care more about some metrics than others (e.g. some algorithms are “wide” and others are “deep,” some are more tolerant of error)
- One reference: US DOE “Q Competition” goals
 - 10 qubits, 1e3 operations, and a “scientifically relevant subroutine” within a year
 - 100 qubits, 1e5 ops, and a “scientific workflow demonstration” by September 2028

SRQC Generation	# Logical Qubits	Hard Operations	Scientific relevance
Prototype	10	10^3	Subroutine
First-generation	100	10^5	Workflow
Next-generation	1000	10^7	Production

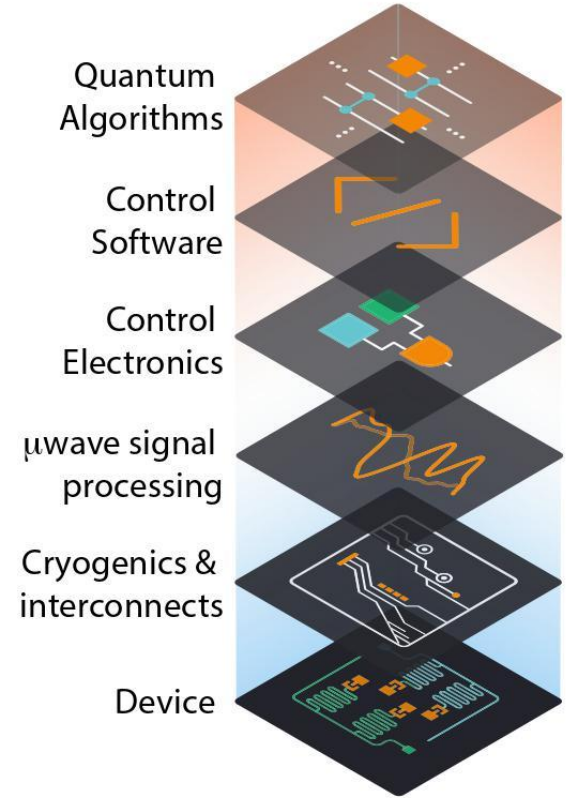
Are quantum computers useful yet?

- We are in the “NISQ era”
 - Several modalities have reached ~1000 physical qubits
 - Error correction works, but is not yet useful
- We are probably still far from useful applications (not everyone agrees!)
 - This is about algorithms as much as engineering
 - The technology that wins the race to the next milestone may not be the one that wins in the end - fundamental R&D is important



The business

- Full-stack: everything in-house, buy nothing
 - Big Tech: IBM, Google, AWS . . .
 - Pure-play: IQM, QuEra, Quantinuum . . .
- Academia and early-stage startups: focus on the device, buy the rest
- Vendors: qubits, control electronics, algorithms/compilers, QEC decoders
- Open source?

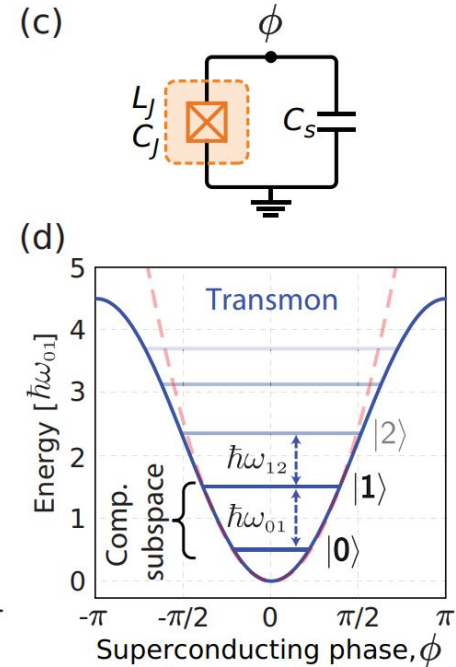
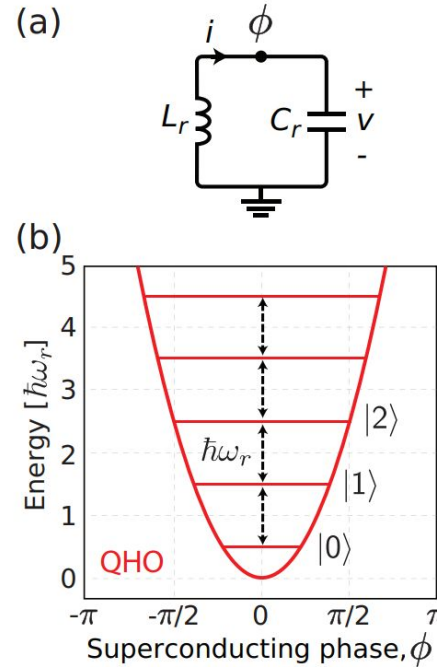


Superconducting qubits

- Overviews:
 - [\[1904.06560\] A Quantum Engineer's Guide to Superconducting Qubits](#)
 - [\[2106.06173\] A practical guide for building superconducting quantum devices](#)
 - [\[1908.02656\] A Review on Quantum Computing: Qubits, Cryogenic Electronics and Cryogenic MOSFET Physics](#)
- More details on qubits:
 - [\[1304.2322\] Coherent Josephson qubit suitable for scalable quantum integrated circuits](#)
 - [\[cond-mat/0703002\] Charge insensitive qubit design derived from the Cooper pair box](#)
- I'll talk about the most generic qubit, the “transmon” - but there is plenty of R&D going on: fluxonium, cat qubits, dual-rail qubits . . .

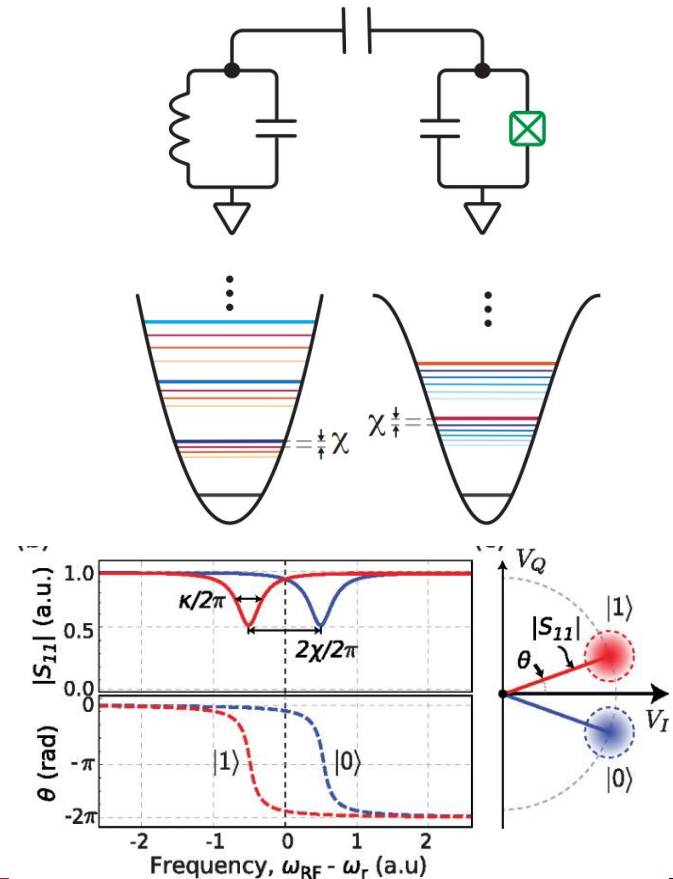
Superconducting qubits

- A transmon qubit is a capacitor and a Josephson junction: a nonlinear LC resonator
- The nonuniformity of energy levels makes them addressable: apply RF power at f_{01} to rotate between $|0\rangle$ and $|1\rangle$
 - Compare to a harmonic oscillator, where f_r hits all energy levels



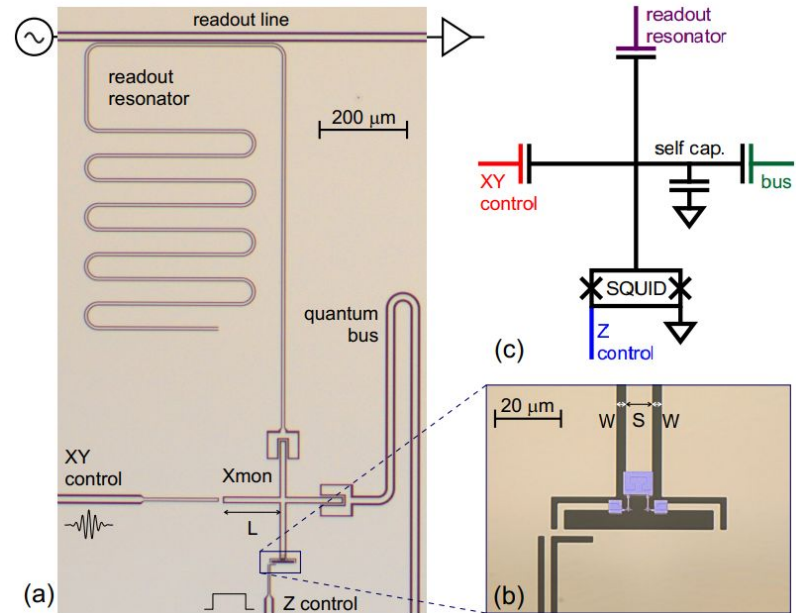
Reading out a qubit

- If you couple a resonator to a qubit, the field in one shifts the energy levels of the other
- In other words: measuring the resonator frequency tells you the energy eigenstate of the qubit
 - This is a projective measurement: whatever state the qubit is in, the readout collapses it onto $|0\rangle$ or $|1\rangle$



A typical superconducting qubit

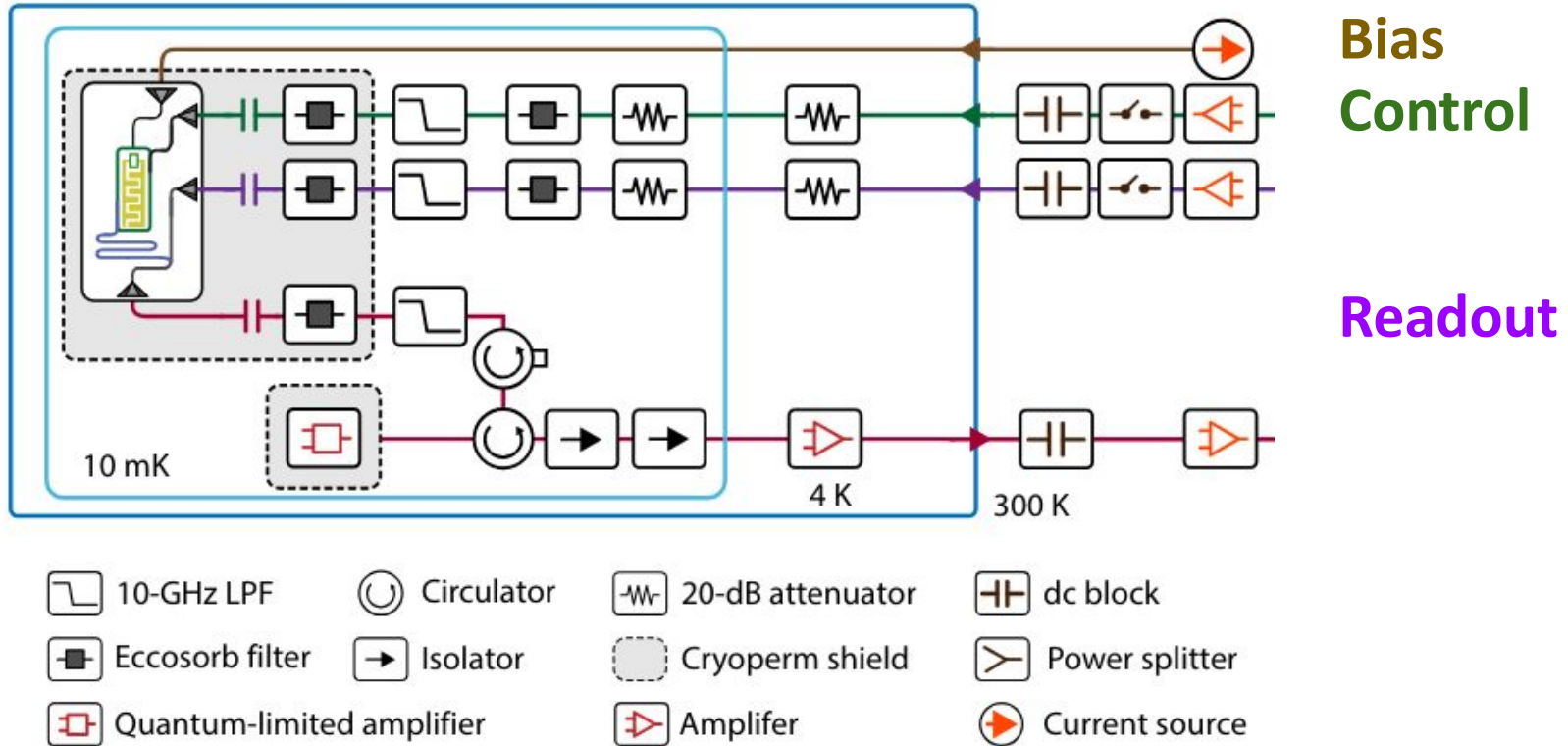
- Al on sapphire, everything planar except the junctions
- “Xmon” layout
 - Cross-shaped capacitor with SQUID loop (pair of Josephson junctions) on one leg for inductance tunable by a magnetic field
 - Capacitive coupling to a control line, a readout resonator (which couples to a shared readout line), and a qubit coupler “bus”



Typical numbers (rough!!)

- Frequencies:
 - Qubit 4-6 GHz
 - Resonator 6-8 GHz
- Timescales:
 - Coherence ~ 100 μs
 - Readout ~ 1 μs
 - Gates 10 ns single-qubit, 100 ns two-qubit

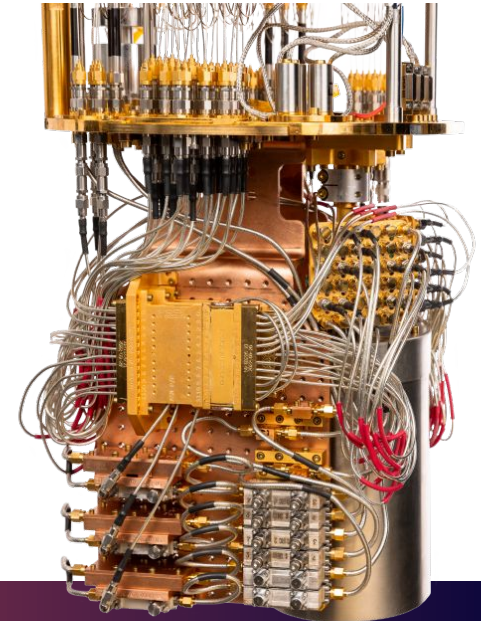
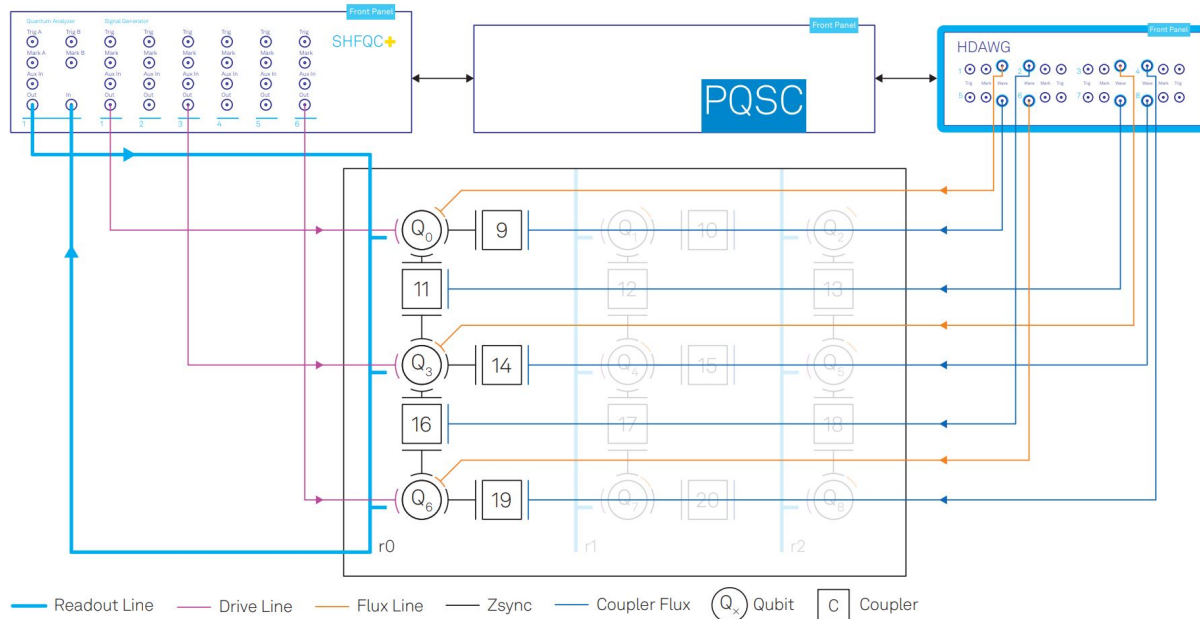
I have a qubit . . . where do I put it?



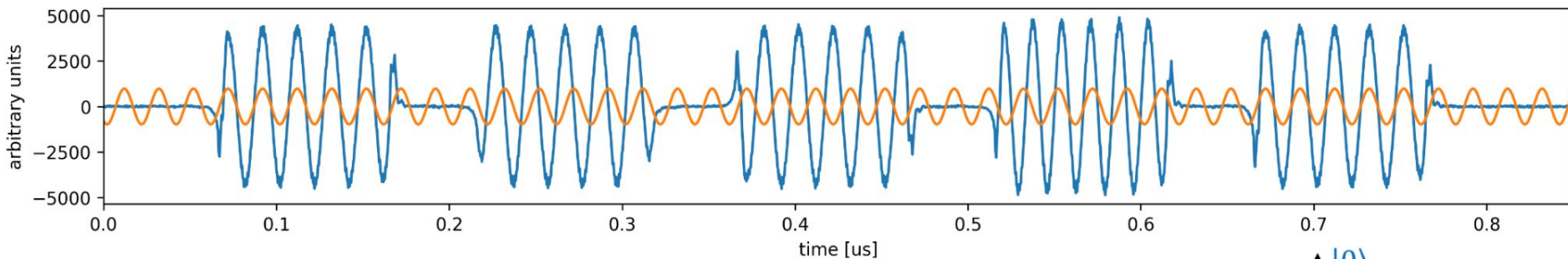
Lots of cables . . .

- Superconducting qubits need a lot of RF lines, maybe:
 - Drive: One RF output per qubit
 - Readout: One freq-multiplexed RF out+in per 3-8 qubits
 - Coupling/tuning: 1-3 DC-coupled outputs per qubit, some fast

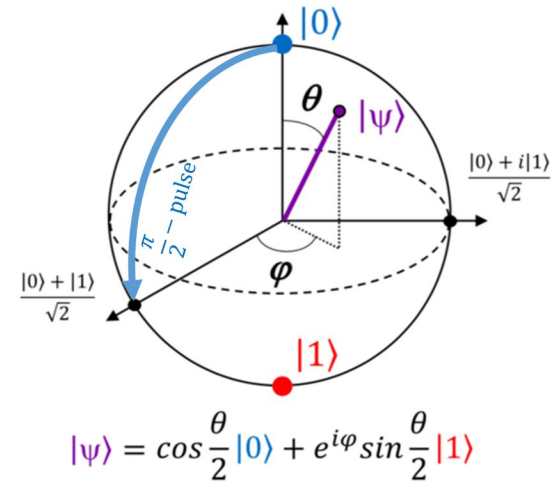
Rigetti “Novara”
9-qubit QPU



How do you drive a qubit?

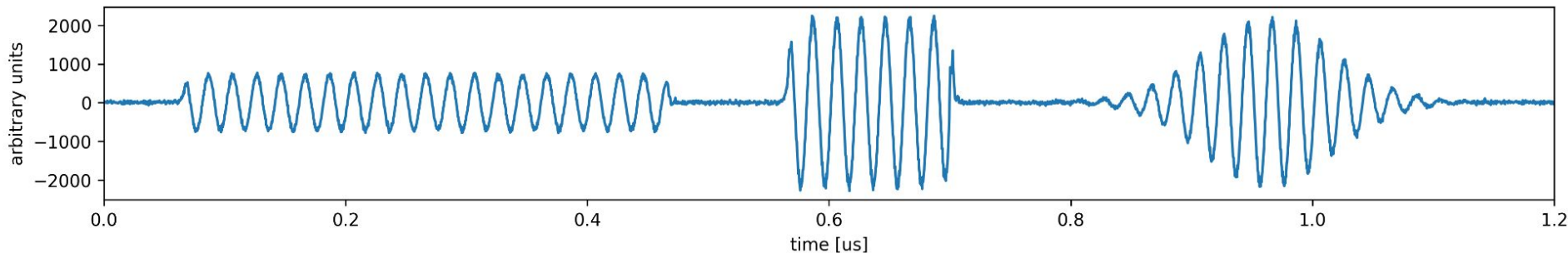


- A qubit is a resonant device: each state transition has some resonant frequency and responds to pulses at that frequency
 - The phase sets the axis of rotation, the amplitude sets the angular velocity
 - If you have multiple X-gates, they all need to match the qubit's reference phase (phase coherence); you need to track the reference phase to define your X and Y axes
 - How do you do a Z-gate? It's just a change of basis
- Imagine the orange sinusoid represents the qubit's reference frame. Which blue pulses are driving it on resonance? Which pulses have the same phase?



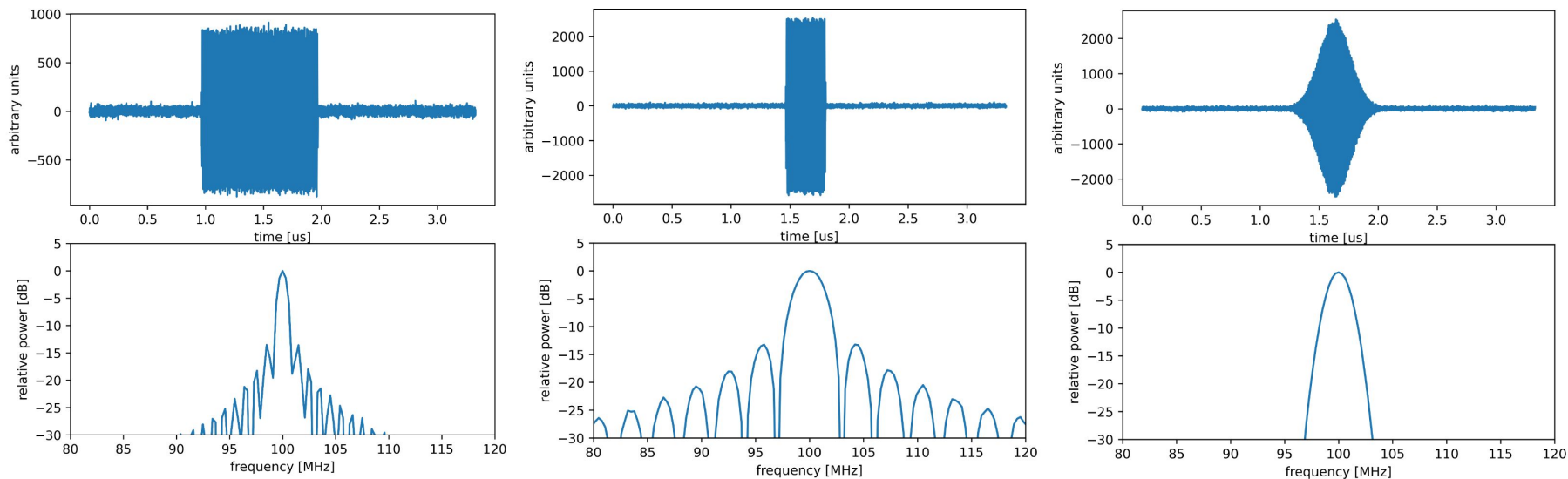
Typical SC qubit frequencies are 4-8 GHz; don't take the plot scales seriously

How do you drive a qubit?



- You want to apply your gates quickly, before your qubit decoheres
- What matters is the time integral of the amplitude, so higher amplitude lets you use a shorter pulse: these example pulses all apply the same amount of rotation
- So you can make your pulses arbitrarily short by cranking up the amplitude, right?

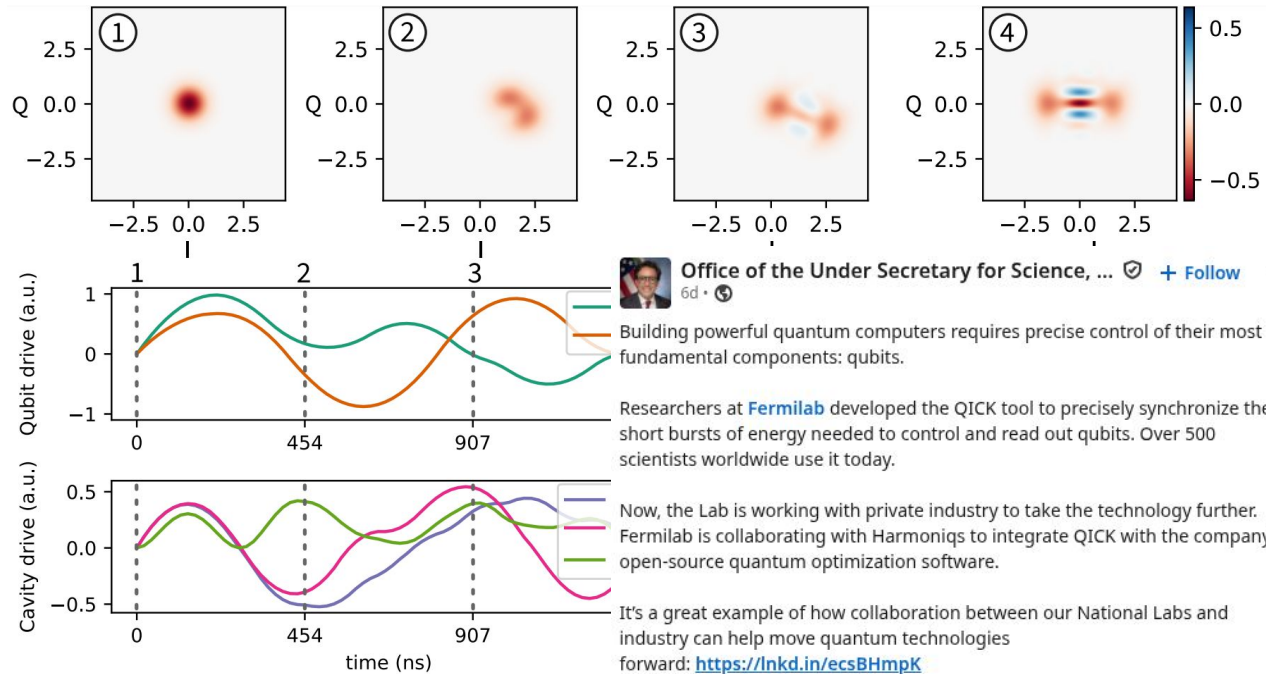
How do you drive a qubit?



- Consider the Fourier transform of your pulse
 - Narrow in time $\leftrightarrow$ wide in frequency
 - If your frequency spectrum is too wide, you will excite other qubit transitions: a 10 ns square pulse won't play nicely with $|f_{21} - f_{01}| = 200\text{-}300\text{ MHz}$
 - Gaussian pulses are compact in both time- and frequency-space
 - Fancier techniques to notch out f_{21} or otherwise suppress side effects

Optimal control

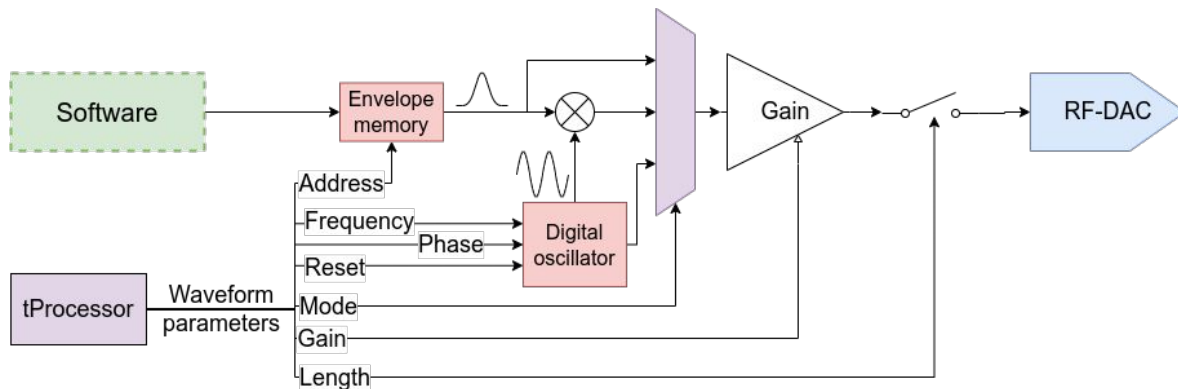
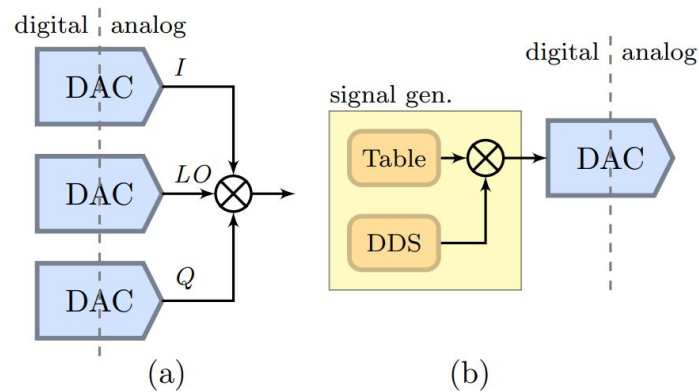
- Piccolo.jl by Harmoniqs - optimal control with awareness of qubit dynamics and hardware behavior
- Partnering with QICK: soon users of each tool can easily use the other
- Dario Gil likes it!



Fermilab and Harmoniqs integrate open-source tools to advance qubit control optimization
news.fnal.gov

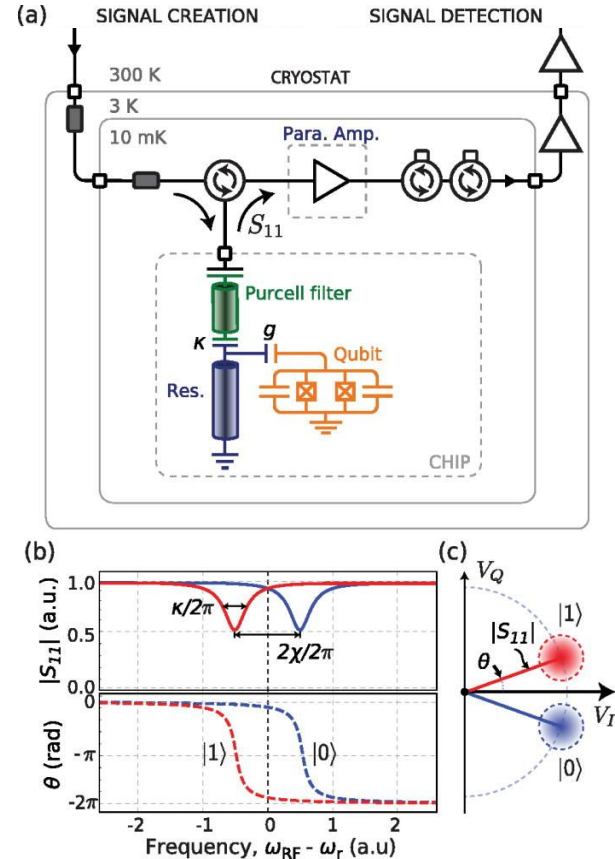
Generating a drive pulse

- Traditional technique: mix slow AWGs (~ 100 MHz) with an LO using an IQ mixer
 - You need the IQ mixer because a spur or LO leakage might excite f_{12}
- Direct digital synthesis: generate a “digital LO,” store IQ envelope in a table, multiply to get the full RF waveform, play with a multi-Gsps DAC
 - Nothing to calibrate
 - Only spurs are Nyquist images - easy to filter
 - Easy to parametrize pulses: use the same envelope with different amplitude, phase, frequency



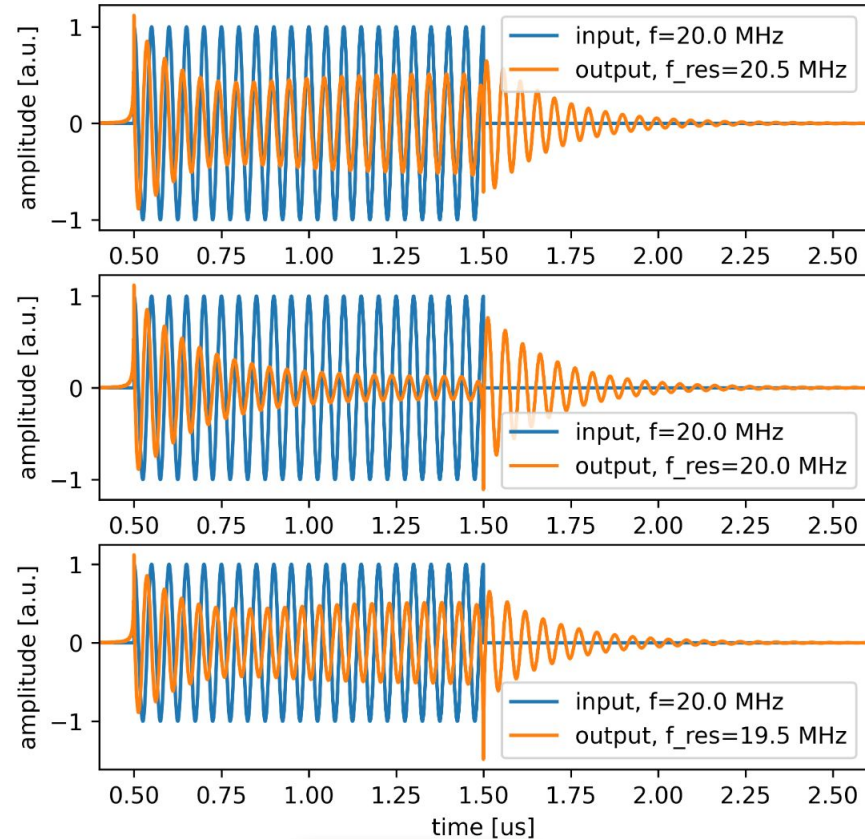
How do you read out a qubit?

- Hit the resonator with a probe pulse and measure what comes back, decide which frequency the resonator is at
- Might read out at the ground or excited freq and look at the transmission amplitude, or read out at the midpoint and look at the phase



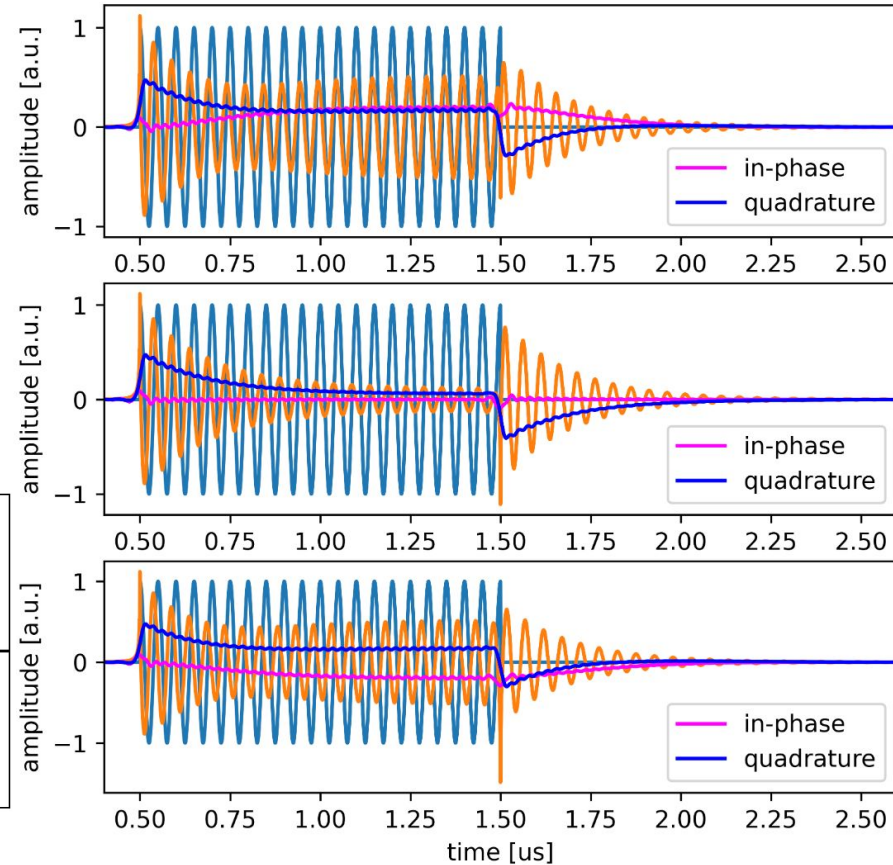
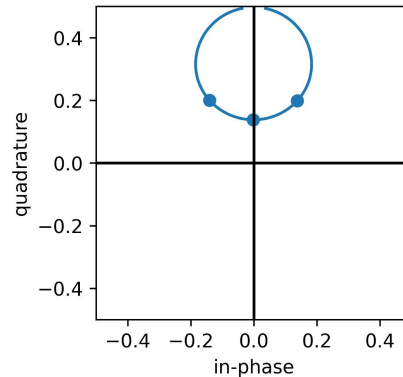
How do you read out a qubit?

- Let's see what we get when the resonator frequency is higher than our probe, equal to our probe, and lower than our probe
- What do you notice about the amplitude of the returning signal?
- What do you notice about the phase . . . can you tell?
 - We don't normally look at these waveforms directly - let's do some processing



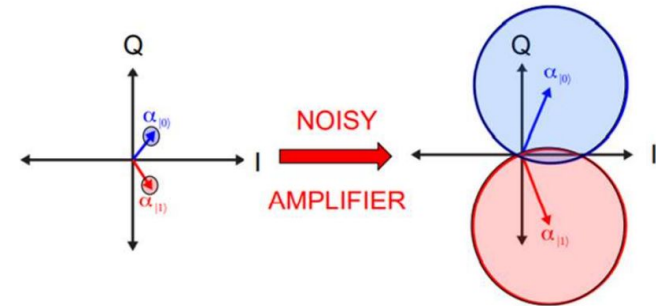
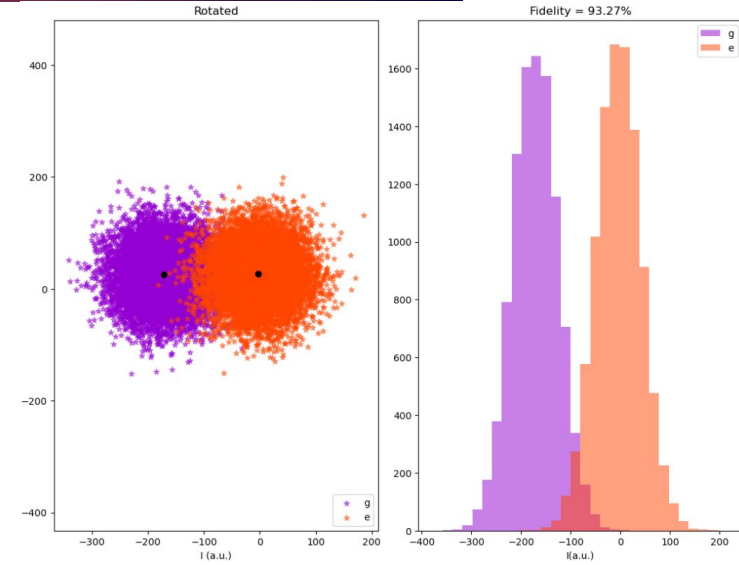
How do you read out a qubit?

- Downconvert and decompose into in-phase (I) and quadrature (Q)
- Now it's clearer that the phase response varies between the three cases
- We can integrate over a window to get a single complex number that represents the result of our readout



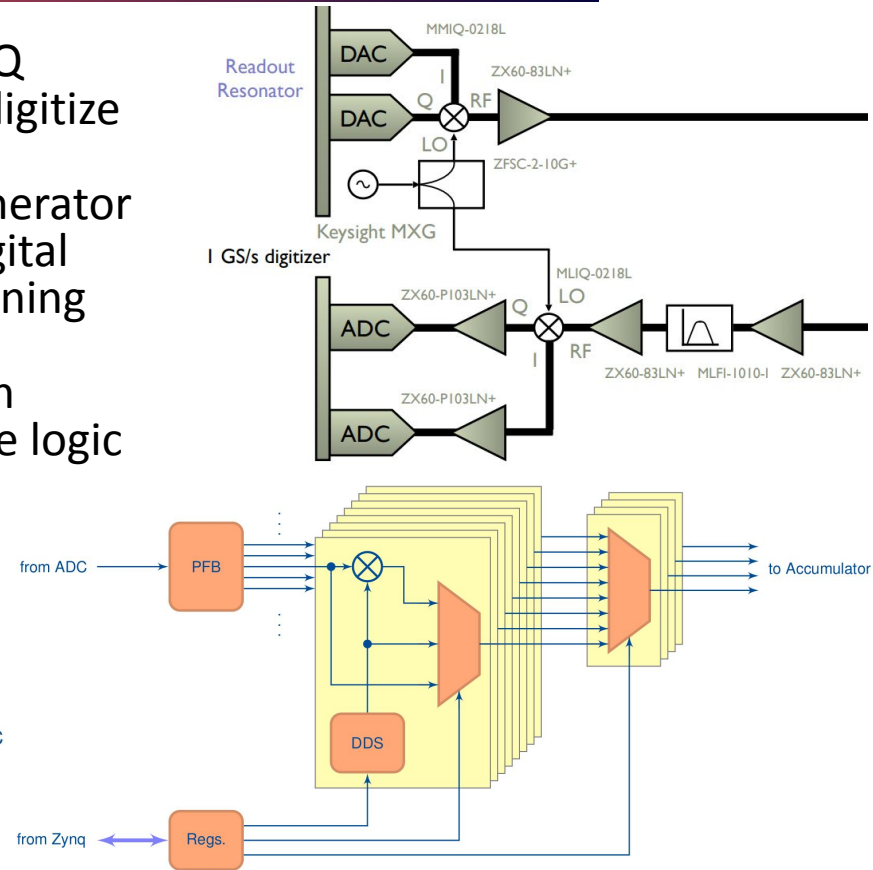
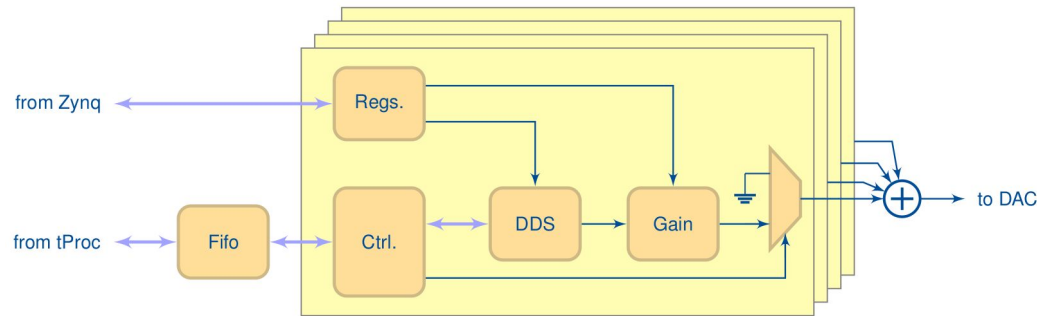
How do you read out a qubit?

- Ultimately the two qubit states appear as blobs in IQ space; the better the separation, the better the readout fidelity
- You want to get good fidelity with minimum time, but too much readout power will mess up your qubit, so signal-to-noise is limited
 - Use parametric amplifiers for quantum-limited SNR



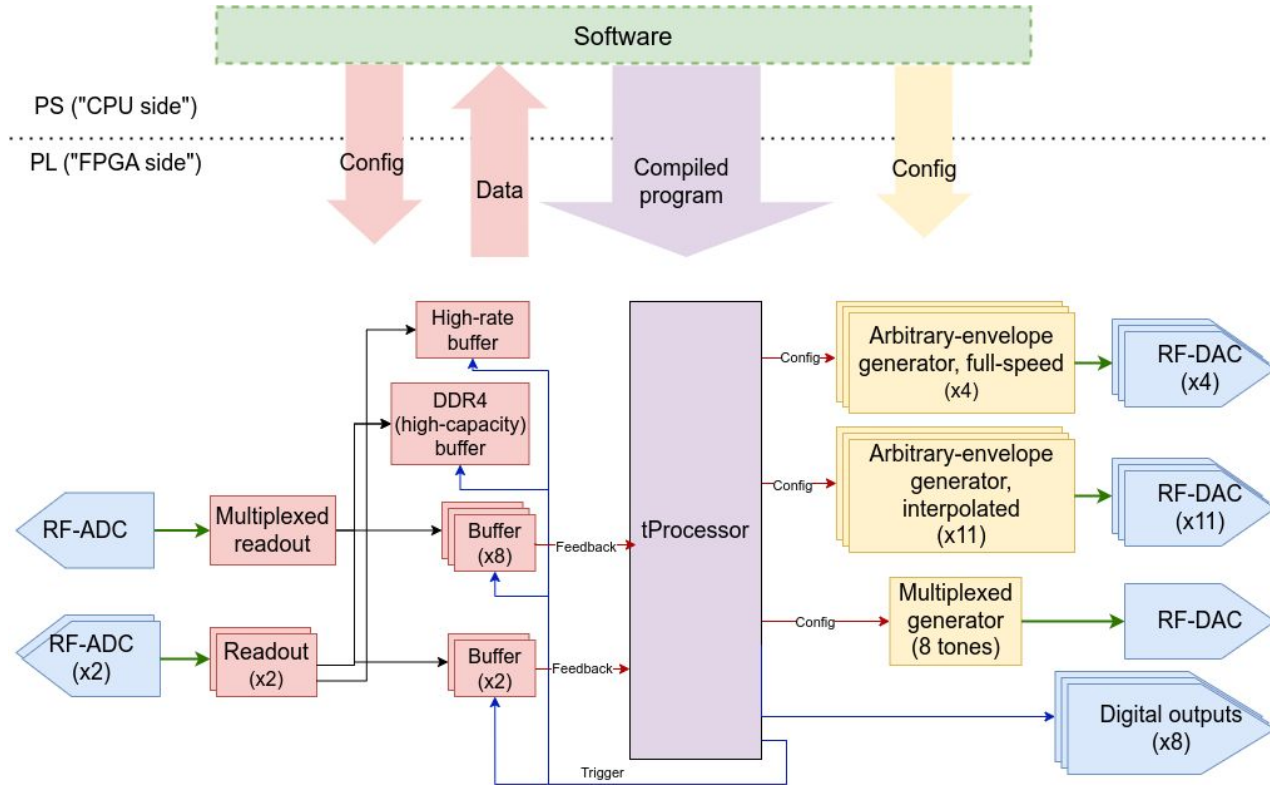
Readout electronics

- Traditionally: mix up, mix down, digitize I+Q
- Direct digitization (with direct synthesis): digitize the RF, do downconversion digitally
 - Phase- and frequency- locking the generator and readout is trivial when it's just digital data streams (and a DAC and ADC running on the same reference)
 - Multiplexing is just more logic . . . with polyphase filter banks, not much more logic



Putting it all together

- Traditionally: lots of benchtop equipment wired together with triggers and gates, or maybe a PXIe system
 - Often no concept of a pulse - one AWG waveform for the whole pulse sequence
- Need a custom execution core:
 - Sends parametrized pulse commands to generators
 - Sends acquisition triggers to readouts
 - Runs in real time - everything scheduled precisely
 - Can branch on readout results



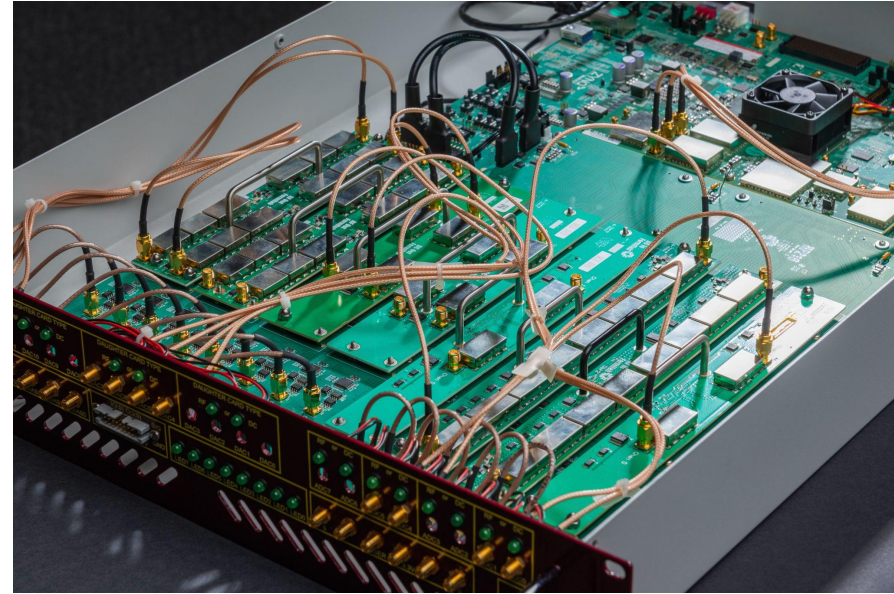
What is QICK?

- An RF control and readout system used for qubits, quantum networks, and quantum sensors
- Open source firmware, software, and hardware
- A collaborative effort between the US DOE's QIS centers, university research groups, and industry
- A growing community of ~500 users worldwide



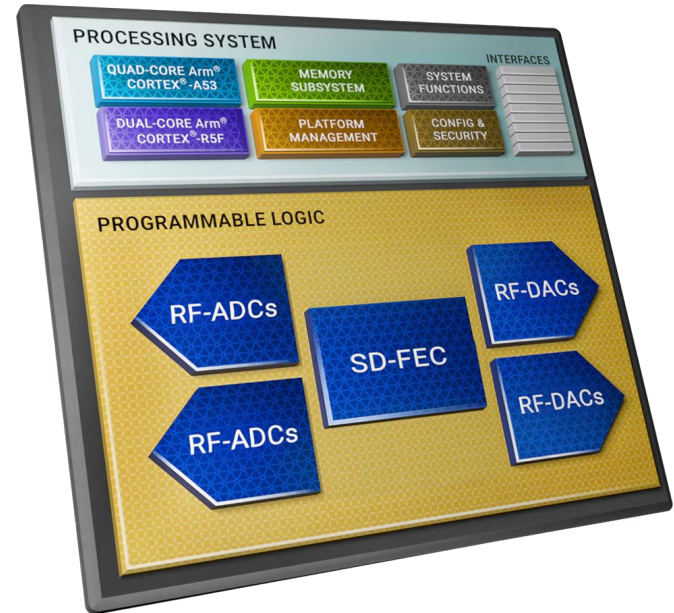
Goals

- Provide state-of-the-art, low cost, and flexible control for the global quantum community
 - Develop the capabilities and interfaces needed for leading-edge experiments
 - Leverage industrial partnerships to provide hardware at minimal cost
 - Follow an open-source and modular “kit” mentality that allows for tinkering and adaptation
 - Flatten the learning curve and grow the community through documentation, training materials, and open support



QICK is powerful

- Closely-coupled AMD RFSoc integrates CPU+FPGA+RF on a single chip
 - Easy development, fewer bottlenecks, low latency (~200 ns end-to-end)
- Direct digital synthesis and readout
 - No analog mixers at typical frequencies
 - Unlimited frequency-hopping
 - Complete phase+frequency stability relative to a single reference
- Pulse sequences, parameter sweeps, and feedback are completely under FPGA control
 - No CPU latencies

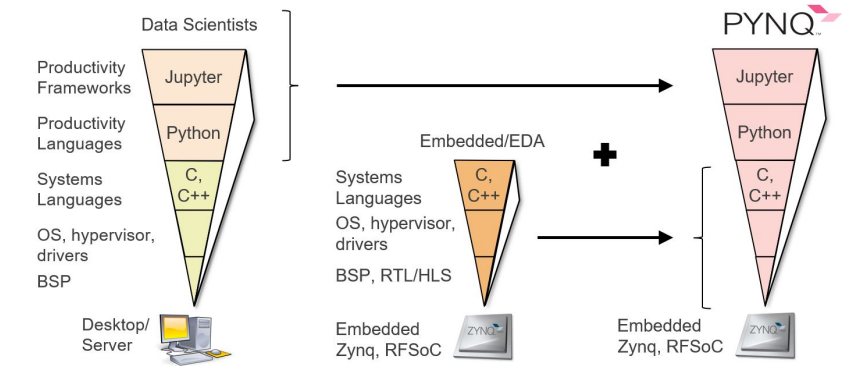


AMD
together we advance_

QICK is easy to work with

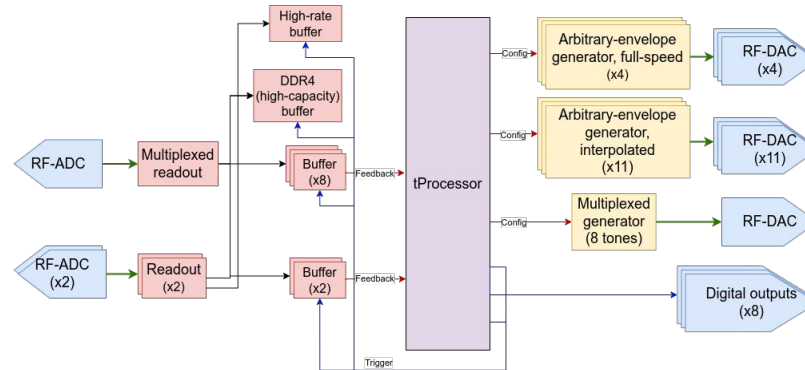
- AMD's open-source RFSoc-PYNQ framework provides a ready-built Linux OS image with direct Python access to firmware blocks and the RFSoc hardware
- Write and run QICK code in Python - on the board's built-in Jupyter server or on your PC

PYNQ Unifies Data Science & Embedded Stacks



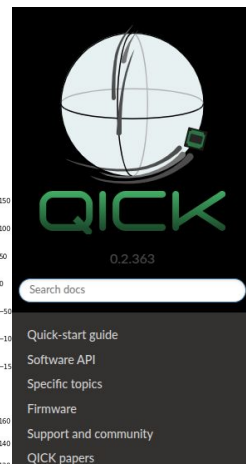
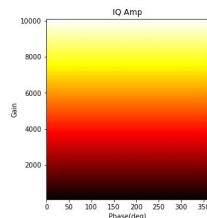
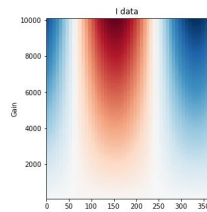
QICK is flexible

- Modular firmware architecture allows QICK firmware blocks to be chosen and configured as needed for a given application
 - All blocks are compatible with all boards, software automatically detects firmware configuration
- Experiments are defined as programs executed by a custom “timed processor” core - anything from simple pulse sequences to complex sweeps, branching, and math
- QICK is a “kit” - extensions to the basic firmware and software add functions for different qubit platforms



QICK is user-friendly

- Documentation guides new users from setting up a board to running standard qubit measurements
- Public support channel and weekly office hours



Example 2: when the parameter we sweep over needs to jump between a constant and a sweep value

A practical example can be transmon pulse spec, in which the first qubit pulse of each experiment will be a frequency that prepares the qubit to e state, and the second pulse's frequency needs to sweep around the e frequency. Here, for simplicity, we make the gain jump between a constant value and a sweep value to demonstrate the

```
In [7]: class ND SweepProgram_2(ND AVeragerProgram):
def initialize(self):
    cfg = self.cfg
    res_ch = cfg["res_ch"]

    self.declare_gen(ch=cfg["res_ch"], nqz=1, ro_ch=cfg["ro_chs"][0])
    for ch in cfg["ro_chs"]:
```

Welcome to the QICK documentation!

[View page source](#)

Welcome to the QICK documentation!



Event Info 26 Interested

Fri Sep 20th · 3:00 PM

QICK Community Call

Unitary Fund

community-call

26 people are interested

Created by [frances.p.f.](#)

QICK is an open-source platform, integrating hardware, firmware, and software for the RF readout and control of quantum systems.

Join our weekly office hours! Meet other QICK users, and get answers to any questions you might have about QICK.

You can also chat with us any time in [Unitary Fund](#) · [# qick](#)

Events in series

Fri Sep 20th · 3:00 PM ...

Fri Sep 27th · 3:00 PM ...

Fri Oct 4th · 3:00 PM ...

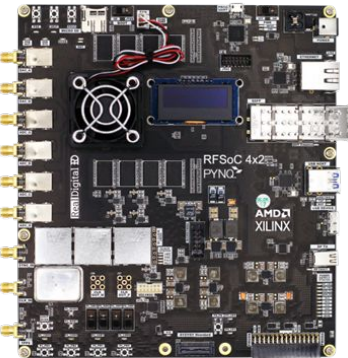
Fri Oct 11th · 3:00 PM ...

[View future events](#)

QICK is educational

QICK runs on AMD University Program's RFSoc4x2 kit, good for small setups and teaching

- QICK-based class on readout, using a firmware-simulated resonator: 2024 US QIS summer school at ORNL, adapted by university lab courses



QICK Summer School 2024

Labels for the QICK class at US QIS Summer School 2024

Secure slides and [links](#) to intro slides are [here](#)

We will be using the resonator and qubit emulation functionality of the QICK for this class.

Resonator and Qubit Emulator

The diagram shows a signal flow from an ADC block to a PFB (Pulse Frequency Beam) block, which is connected to a DAC block. The DAC block is connected to a resonator circuit, which is represented by a circle with a vertical line and a horizontal line. The resonator circuit is connected to a DAC block, which is connected to a PFB block. The PFB block is connected to an ADC block. The diagram also shows a feedback loop from the DAC block to the ADC block.

Single shot readout, no T1 decay

The qubit being in a certain state (eg. the ground state) causes a shift in the frequency of the qubit's resonant frequency. This change in frequency results in a change in the complex L_{21} of a resonant mode transmitted through the bridge.

When you take a single shot of the measurement, by measuring the resonant frequency you can find out the state of the qubit without waiting for the state to relax or waiting for the state to relax. In this experiment, in QIS, you will read out the resonator at one location (eg. in ground state vs. qubit in excited state).

In a real experiment, you might read out at the ground or excited state frequency and look at the transmission amplitude, or read out at the resonant and look at the phase.

We measure the "resonant frequency" of our qubit system by preparing the qubit in the excited state and in the ground state and looking at the distribution of the resonant frequency in both cases.

The three plots show the resonant frequency distribution for ground and excited states. The first plot shows the resonant frequency distribution for the ground state, with a peak at approximately 100 MHz. The second plot shows the resonant frequency distribution for the excited state, with a peak at approximately 110 MHz. The third plot shows the difference between the two distributions, with a peak at approximately 10 MHz.

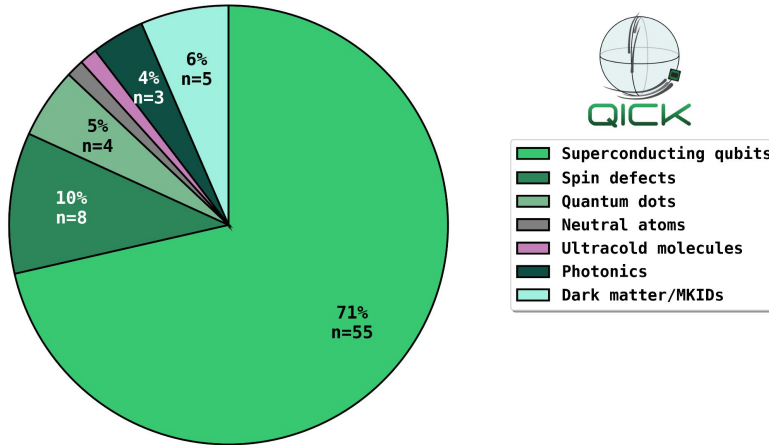
US Quantum Information Science Summer School 2024



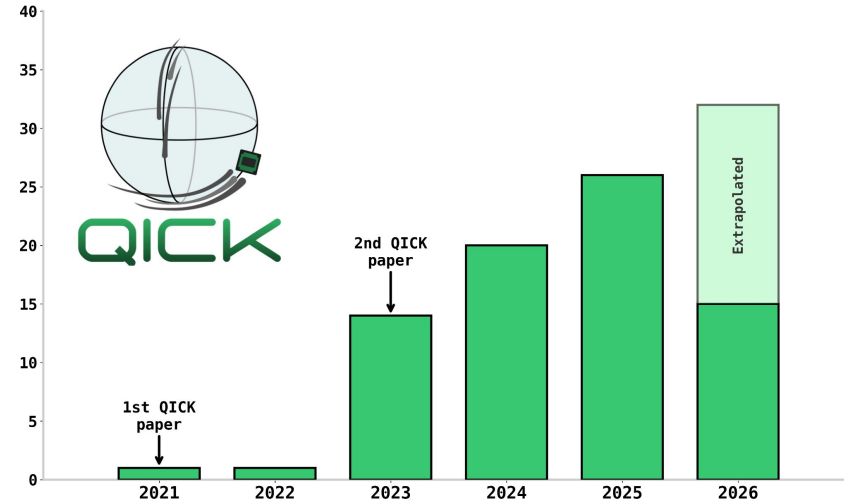
How are we doing?

- Scientific output using QICK maintains a steady rise
- Superconducting qubits are still our core, but other communities are finding QICK useful

Scientific papers using QICK by qubit type

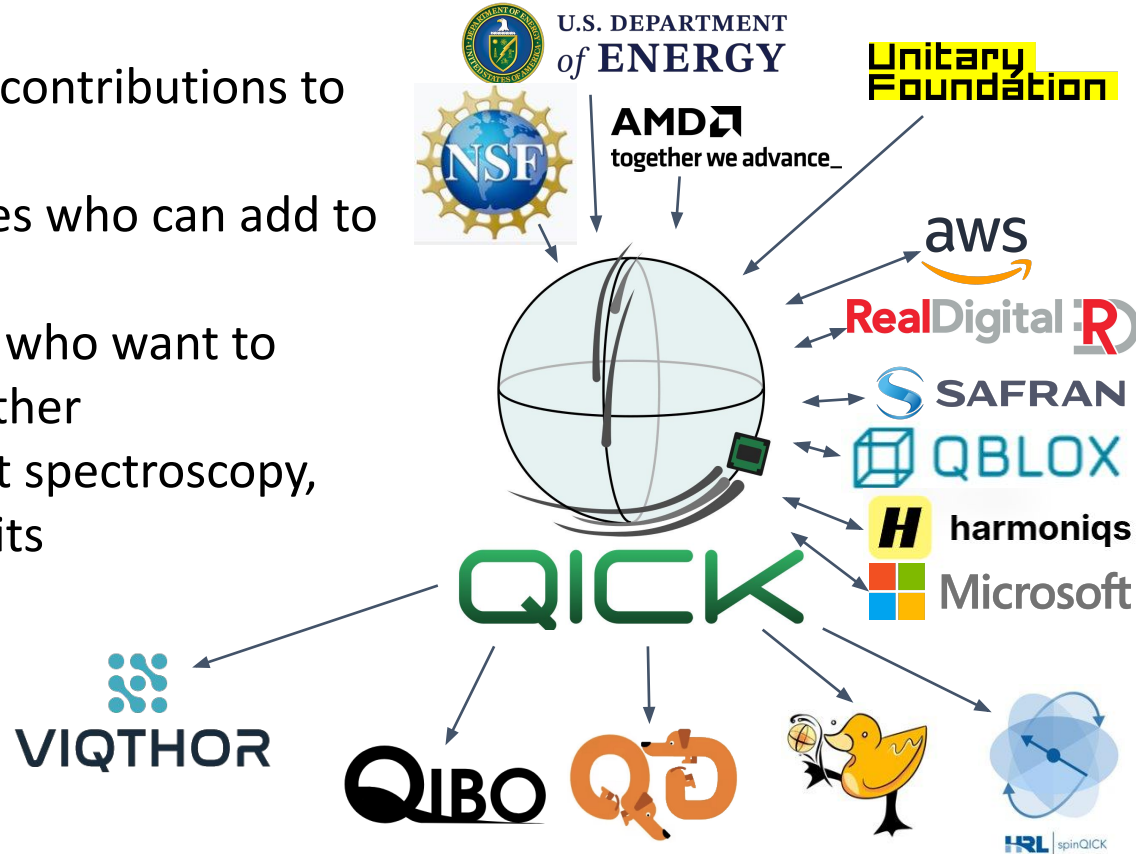


Scientific papers using QICK published per year



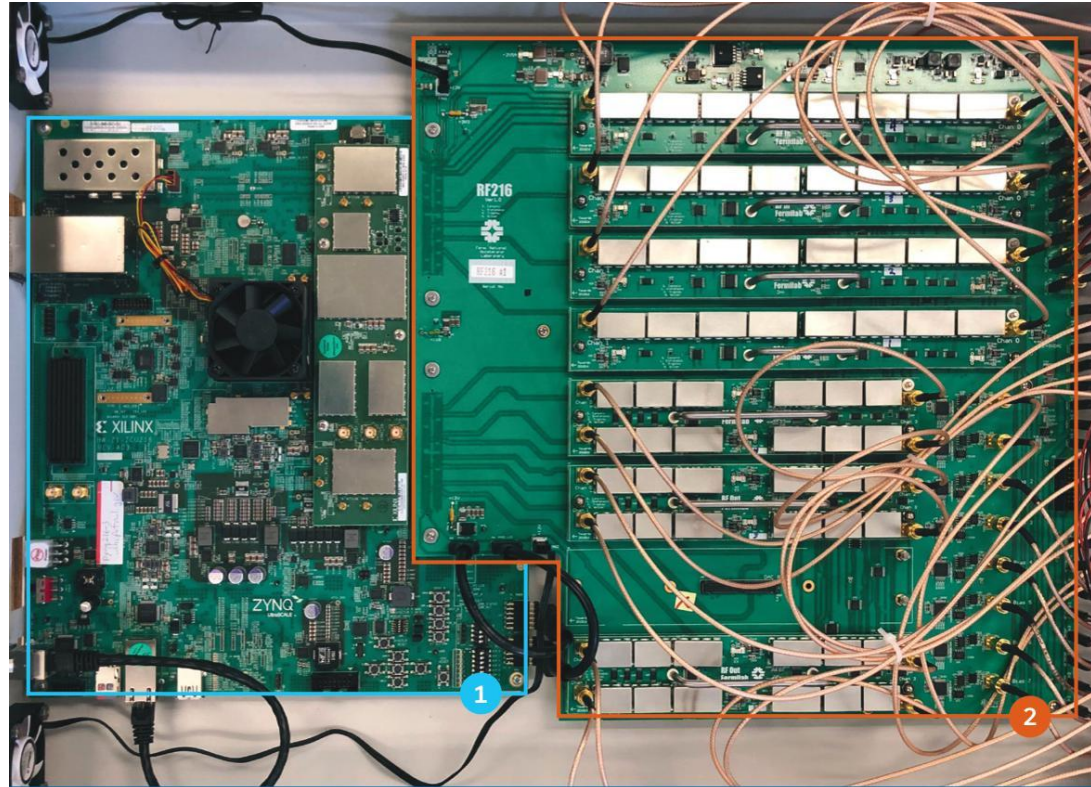
Growth through collaboration

- We get funding and other contributions to support QICK
- We partner with companies who can add to the QICK ecosystem
- We collaborate with users who want to extend QICK or push it further
 - QICK-DAWG for defect spectroscopy, spinQICK for spin qubits



The QICK box

- Designed by the Fermilab group as a “lab in a box”
- Manufactured and sold by industry partners (RealDigital, Safran, soon Qblox)

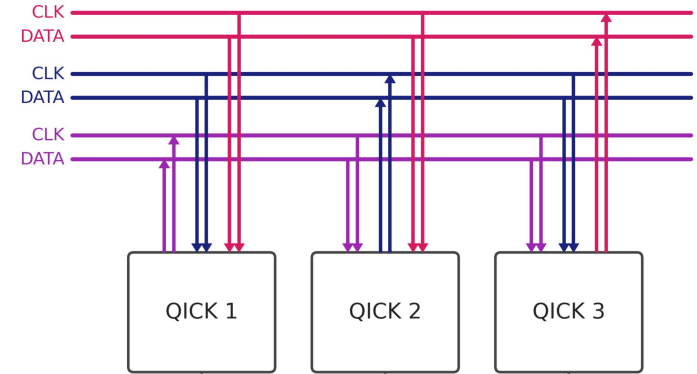


1 AMD/Xilinx FPGA board ZCU216

2 Fermilab custom RF, DC, bias, I/O board for the ZCU216

Next: scaling up

- “XCOM” interface: synchronize and network up to 15 boards
 - All-to-all topology, optimized for deterministic latency (<100 ns)
 - Enables large QICK systems and QEC



Future: getting smarter?

- ML/AI:
 - State discrimination
 - Calibration
- QEC decoding:
 - On the controller FPGA
 - Offloaded to a GPU through a low-latency link
 - NVIDIA: NVQLink (4 us latency), Ising Decoding (1 us runtime)
- Versal RF is coming
 - Faster DACs and ADCs (wider Nyquist zones, 18 GHz analog bandwidth)
 - More built-in DSP logic, “AI Engines” for high-throughput data processing
 - We do not expect everybody will need Versal; we’ll still support the RFSoc

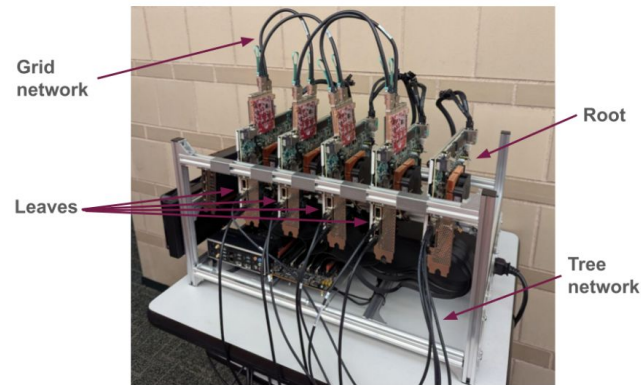
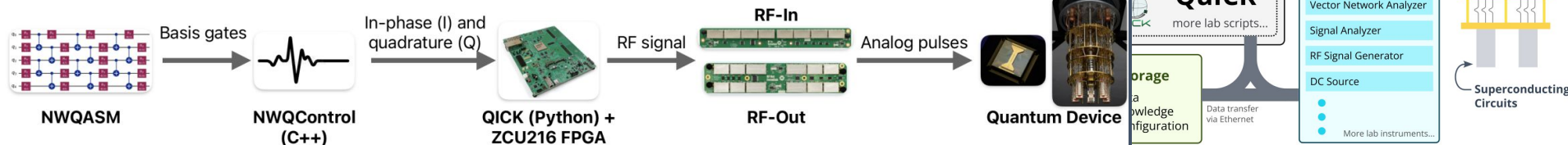
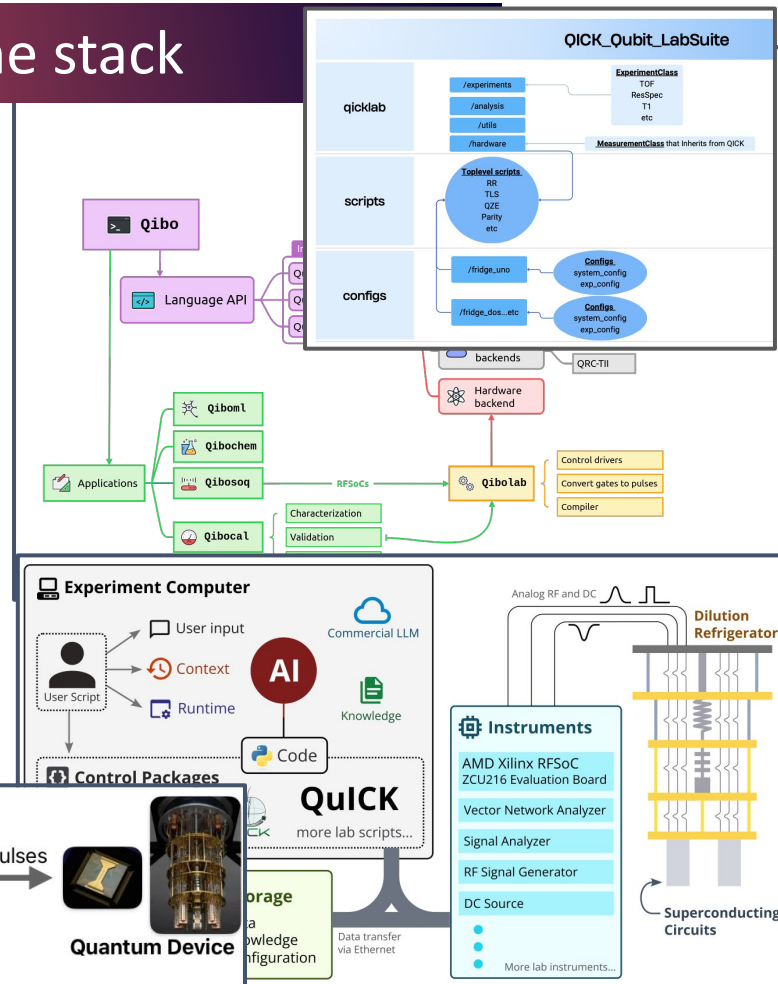


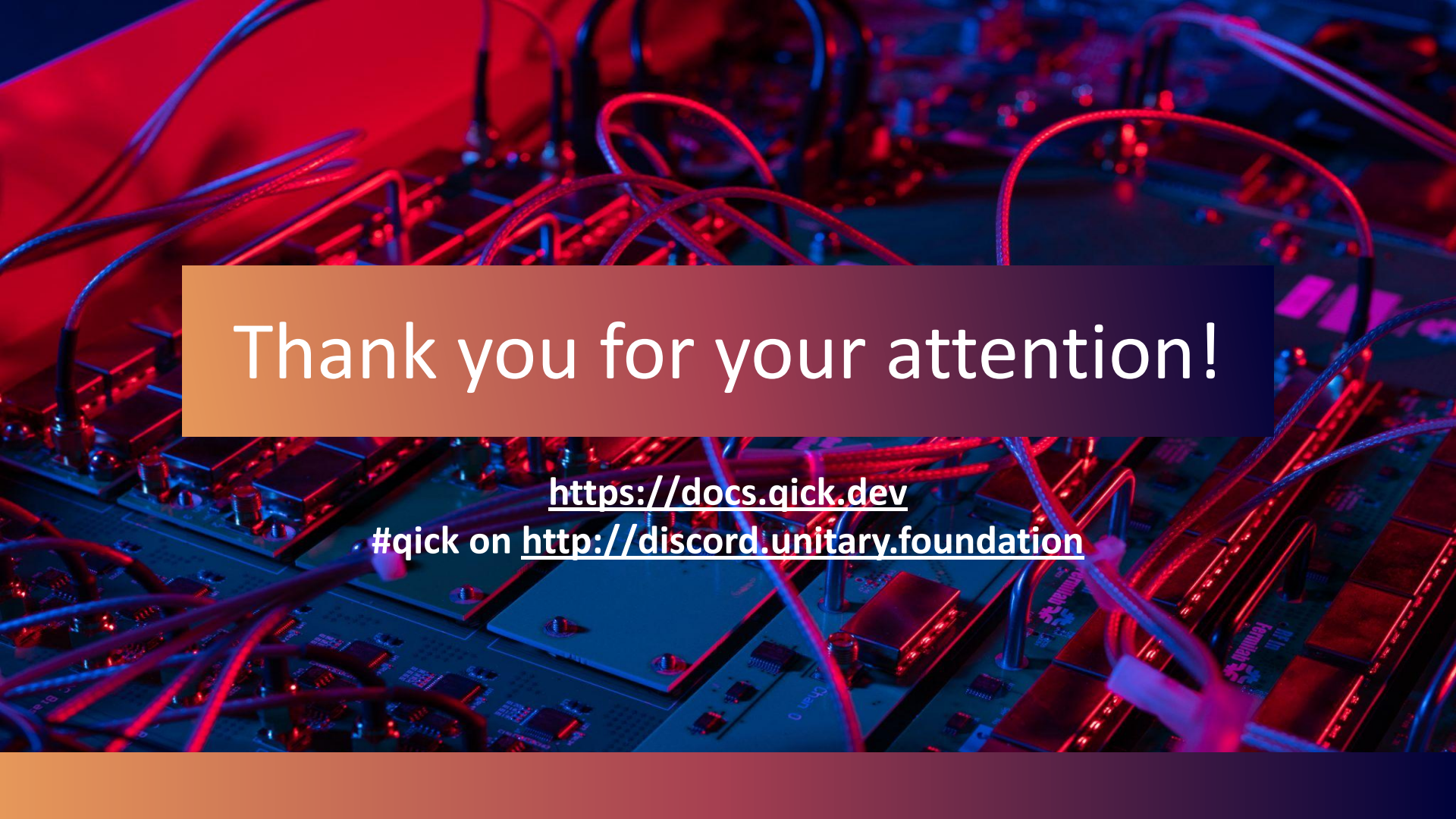
Fig. 6: Implementation of DECONET/HELIOS using five FPGAs. The right-most FPGA is the root of the tree, and the other four FPGAs are leaf nodes.



Future: moving up the stack

- We think of QICK as a kit or instrument - we give you a firm foundation and a toolbox, you integrate it with your lab's codebase (UI, data management, calibration management, compiler, etc.)
- Some groups are working on integrating QICK into compilers or end-to-end workflows
 - We are watching with interest





Thank you for your attention!

<https://docs.qick.dev>

#qick on <http://discord.unitary.foundation>