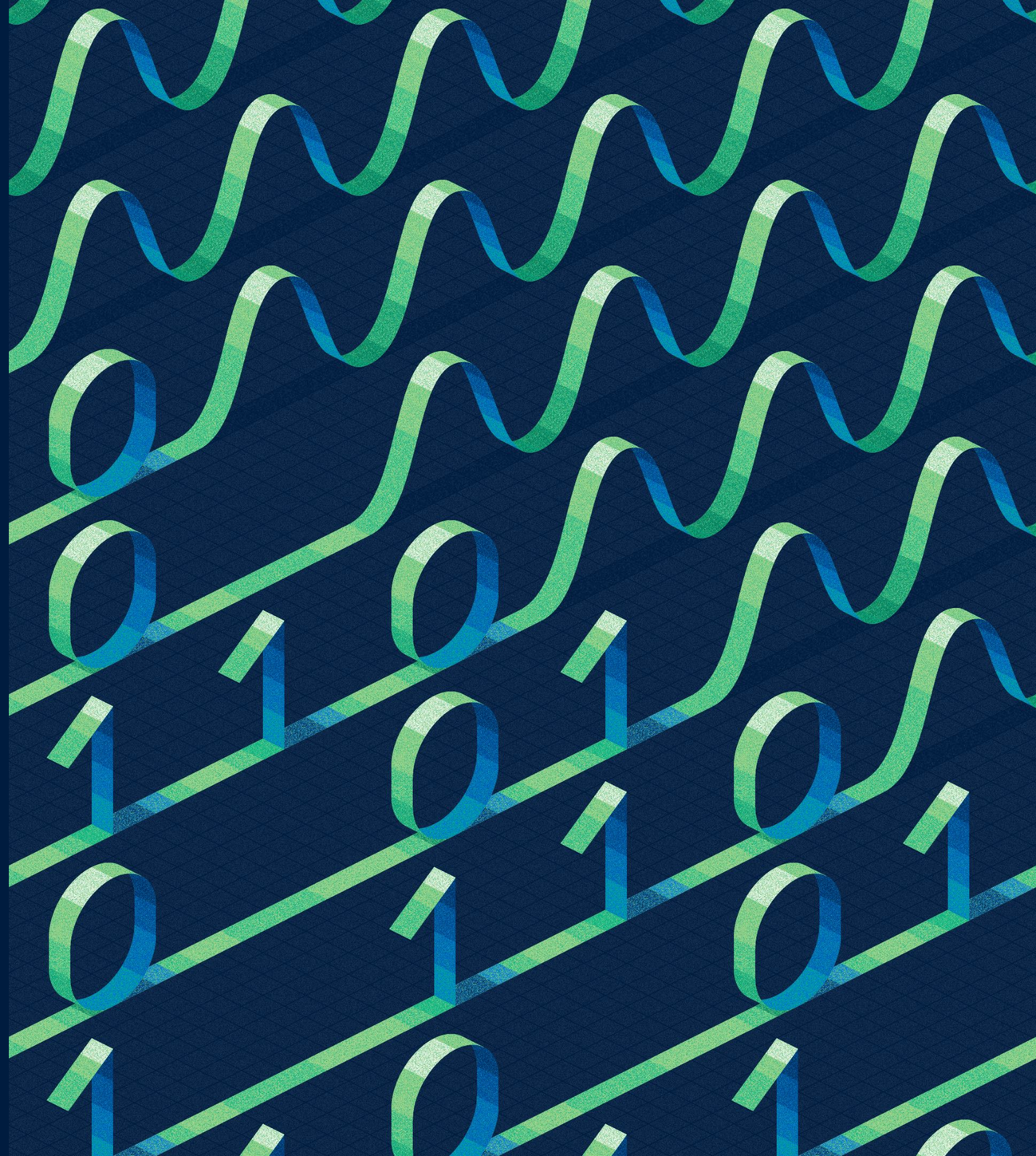
horizon quantum

Building the software infrastructure to power tomorrow's computers



CAUTIONARY NOTES

Notes and Important Disclosures

This presentation (together with oral statements made in connection herewith, the "Presentation") is for information purposes only to assist interested parties with evaluation Horizon Quantum Holdings Ltd. ("Horizon" or the "Company").

No Offer or Solicitation

This presentation does not constitute a solicitation of a proxy, consent, or authorization with respect to any securities. This presentation also does not constitute an offer to sell, or solicitation of an offer to buy any securities, nor will there be any sale of securities in any states or jurisdictions in which such offer, solicitation, or sale would be unlawful prior to registration or qualification under the securities laws of any such jurisdiction. No offering of securities will be made except by means of a prospectus meeting the requirements of Section 10 of the Securities Act of 1933, as amended, or an exemption therefrom. Neither the U.S. Securities and Exchange Commission (the "SEC") nor any securities commission of any other U.S. or non-U.S. jurisdiction has approved, disapproved, or determined that this Presentation is truthful or complete. Any representation to the contrary is a criminal offense.

Financial Information

The independent auditor of Horizon has examined or compiled the financial information and data contained the Presentation; accordingly, no such independent auditor provides any assurance with respect to any financial information included herein. Such information and data may not be included in, may be adjusted in, or may be presented differently in, any registration statement, prospectus, proxy statement or other report or document to be filed or furnished by Horizon, or any other report of document to be filed with the SEC.

Industry and Market Data

This Presentation has been prepared by Horizon and includes market data and other statistical information from third-party industry publications and sources as well as from research reports prepared for other purposes. Although Horizon believes these third-party sources are reliable as of their respective dates, Horizon and its affiliates have not independently verified the accuracy or completeness of this information and cannot assure you of (and make no representation or warranty, express or implied with respect to) the accuracy or completeness of such data or statistical information. Horizon and its affiliates, directors, officers, employees, members, partners, shareholders or agents make any representation or warranty with respect to the accuracy of such information.

Trademarks

This Presentation contains trademarks, service marks, trade names, and copyrights of Horizon and other companies, which are the property of their respective owners. The use or display of third parties' trademarks, service marks, trade name or products in this Presentation is not intended to, and does not imply, a relationship with Horizon, or any endorsement or sponsorship by or of Horizon. Solely for convenience, the trademarks, service marks and trade names referred to in this Presentation may appear with the TM or SM symbols, but such references are not intended to indicate, in any way, that Horizon will not assert, to the fullest extent permitted under applicable law, their rights or the right of the applicable licensor to these trademarks and trade names.

Forward Looking Statements

This communication includes "forward-looking statements" with respect to Horizon. The expectations, estimates, and projections of the business of Horizon may differ from its actual results and consequently, you should not rely on these forward-looking statements as predictions of future events. Words such as "expect," "estimate," "anticipate," "intend," "may," "will," "could," "should," "potential," "plan" and similar expressions are intended to identify such forward-looking statements. These forward-looking statements include, without limitation, expectations related to future financial performance, operating results, software and hardware of Horizon as well as larger trends in quantum computing. These forward-looking statements involve significant risks and uncertainties that could cause the actual results to differ materially from the expected results and are subject to, without limitation, (i) known and unknown risks, including the risks and uncertainties indicated from time to time in Horizon's filings with the SEC, including those under heading "Risk Factors" in Horizon's Registration Statement on Form F-4 filed with the SEC on January 30, 2026 (ii) uncertainties; (iii) assumptions; and (iv) other factors beyond Horizon's control that are difficult to predict because they relate to events and depend on circumstances that will occur in the future. They are neither statements of historical fact nor promises or guarantees of future performance. Therefore, actual results may differ materially and adversely from those expressed or implied in any forward-looking statements and Horizon therefore cautions against placing undue reliance on any of these forward-looking statements.

Use of Non-GAAP Financial Measures

The non-GAAP financial measures contained in this presentation (including, without limitation, Adjusted EDITDA and EBITDA) are not GAAP Measures of Horizon's financial performance or liquidity and should not be considered as alternatives to net income (loss) as measures of financial performance or cash flows from operations as measures of liquidity, or any other performance measure derived in accordance with GAAP. A reconciliation of such non-GAAP measures to their most directly comparable GAAP measures included in this Presentation. You are encouraged to evaluate each adjustment to non-GAAP financial measures and reasons management considered it appropriate for supplemental analysis. Horizon believes that the use of these non-GAAP financial measures provides additional tools for investors to use in evaluation ongoing operating results and trends. Horizon's presentation of these measures should not be constructed as an inference that our future results will be unaffected by unusual or non-recurring items. In addition, these measures may not be comparable to similarly titled measures used by other companies in our industry or across different industries. These non-GAAP financial measures have limitations as analytical tools when assessing Horizon's operating and financial performances, and you should not consider them in isolation, or as a substitute for any consolidated statement of operations data prepared in accordance with U.S. GAAP.

> Our mission is to unlock broad quantum advantage by building software infrastructure that empowers developers to use quantum computing to solve the world's toughest computational problems.

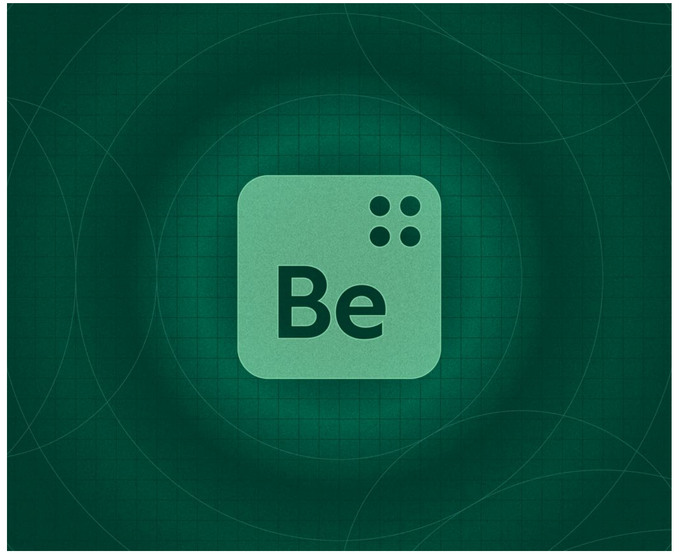
Q2 FY 26 Highlights

Cash Infusion

Cash infusion of $28.7M from the voluntary exercise of public warrants through August 3, 2026, strengthened our runway.

Object-oriented quantum programming

Beryllium became accessible to early access users



Strategic Hardware-Software Integration

Hardware collaborations

Joint Development of Embedded Calibration Technologies

QUANTUM MACHINES

Improvements and expansion of in-house testbed

Ember-1 testbed system becomes available to first users

Modular superconducting system (Rigetti, Maybell, & Quantum Machines)

Frontier quantum computing system to be located in Ireland

IonQ, 256-qubit trapped-ion system

First-mover advantage in quantum software infrastructure

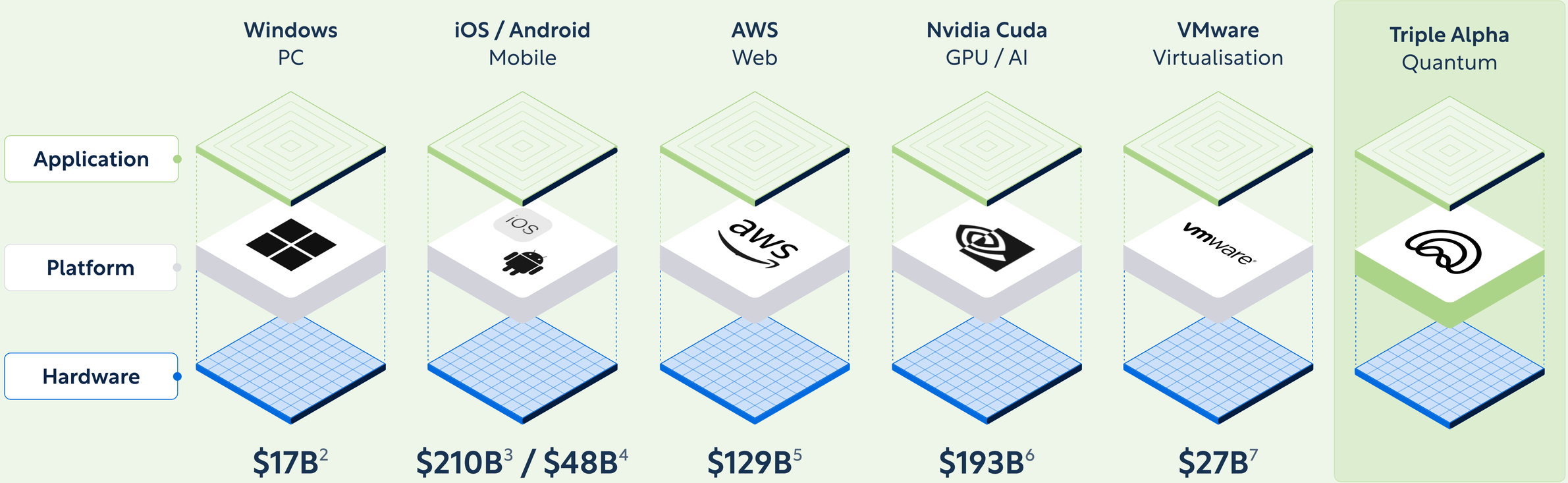
- We believe that **software will drive commercial adoption** of quantum hardware
- **Hardware-agnostic approach** anticipated to give users the flexibility to leverage the best quantum systems for *their solutions* **regardless of underlying modality or vendor**
- **The leading public software infrastructure pure-play** in quantum computing **with capital efficient model**
- **World-class, deep-science team** focused on quantum software
- Product & GTM roadmap designed to deliver **commercial demand**
- **Compelling market opportunity** to deliver the software infrastructure for quantum computers

No computer is effective without software...

... and classical code doesn't run on quantum machines.

The **platform layer** unlocks the value of every new computing technology

Horizon Quantum is building the software infrastructure to set the standard for developing, deploying and executing quantum software, regardless of which hardware prevails[1]



Windows PC	iOS / Android Mobile	AWS Web	Nvidia Cuda GPU / AI	VMware Virtualisation
$17B[2]	**$210B**[3] **/ $48B**[4]	**$129B**[5]	**$193B**[6]	**$27B**[7]
2025 revenue	2025 revenue	2025 revenue	2025 revenue	2025 revenue

[1] Revenues included for illustrative purposes to highlight the historic value of platform development. These values are not included for comparative purposes or suggestive of future revenues.

[2] As reported on Microsoft Corporation's Annual Report on Form 10-K, filed for the Fiscal Year ended June 30, 2025; revenue by product - Windows and Devices.

[3] As reported on Apple Inc.'s Annual Report on Form 10-K, filed for the Fiscal Year ended September 27, 2025; disaggregated net sales iPhone.

[4] As reported on Alphabet Inc.'s Annual Report on Form 10-K, filed for the Fiscal Year ended December 31, 2025; division Google Subscriptions, Platforms and Devices.

[5] As reported on Amazon.com, Inc.'s Annual Report on Form 10-K, filed for the Fiscal Year ended December 31, 2025; net sales by group AWS.

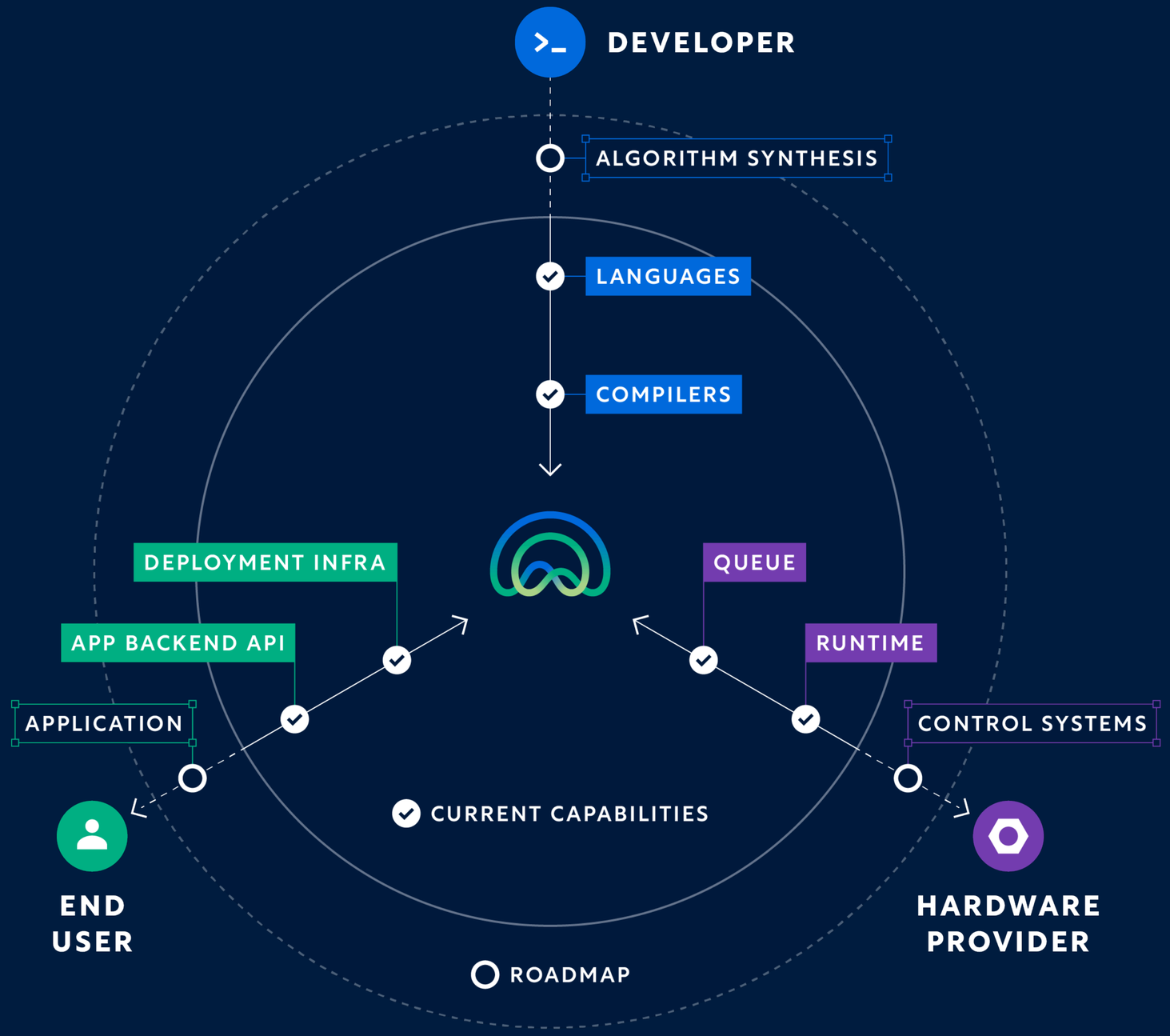
[6] As reported on Nvidia Corporation's Annual Report on Form 10-K, filed for the Fiscal Year ended January 25, 2026; Compute & Networking revenue segment.

[7] As reported on Broadcom Inc.'s Annual Report on Form 10-K for the Fiscal Year ended November 2, 2025; division Infrastructure Software.

At the heart of quantum computing

WE SEEK TO BRIDGE THE GAP BETWEEN TODAY'S HARDWARE AND TOMORROW'S APPLICATIONS

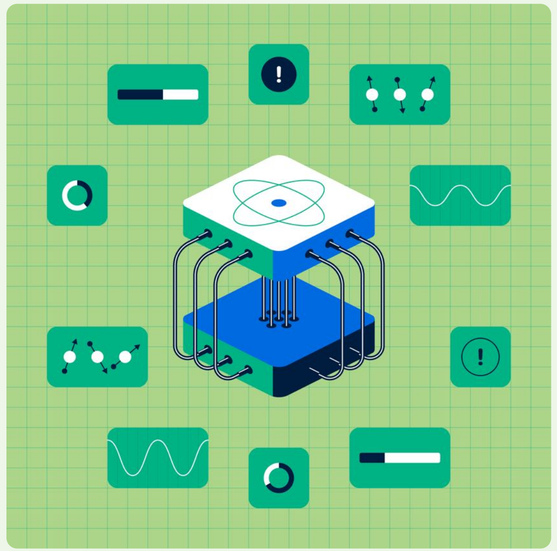
By connecting developers, end users, and hardware providers, we believe our quantum software infrastructure is positioned to unlock the potential of quantum computing to solve-real world problems.



Today, developers face major barriers to harness quantum computing **just like in the early days of classical computing**

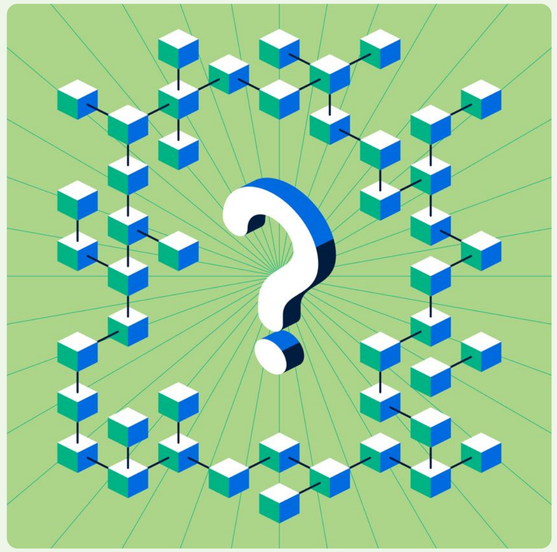
01

Creating quantum algorithms is difficult, non-intuitive and inaccessible to all but a few hundred specialists



02

Hardware is diverse and imperfect, rendering programs non-portable and unreliable

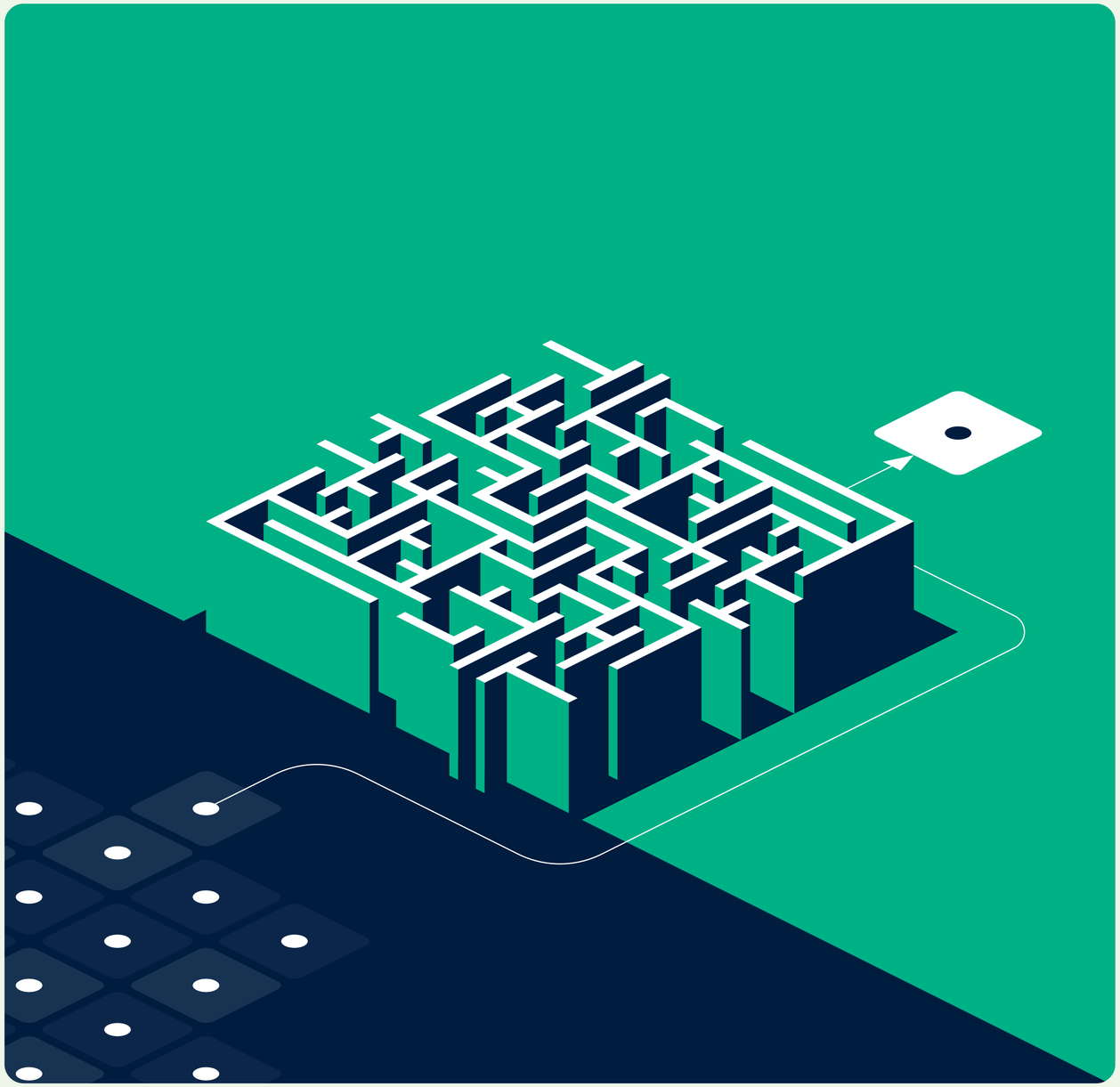


03

Programming languages lack abstraction, requiring programs to be written logic gate by logic gate

Hardware is only half the picture –

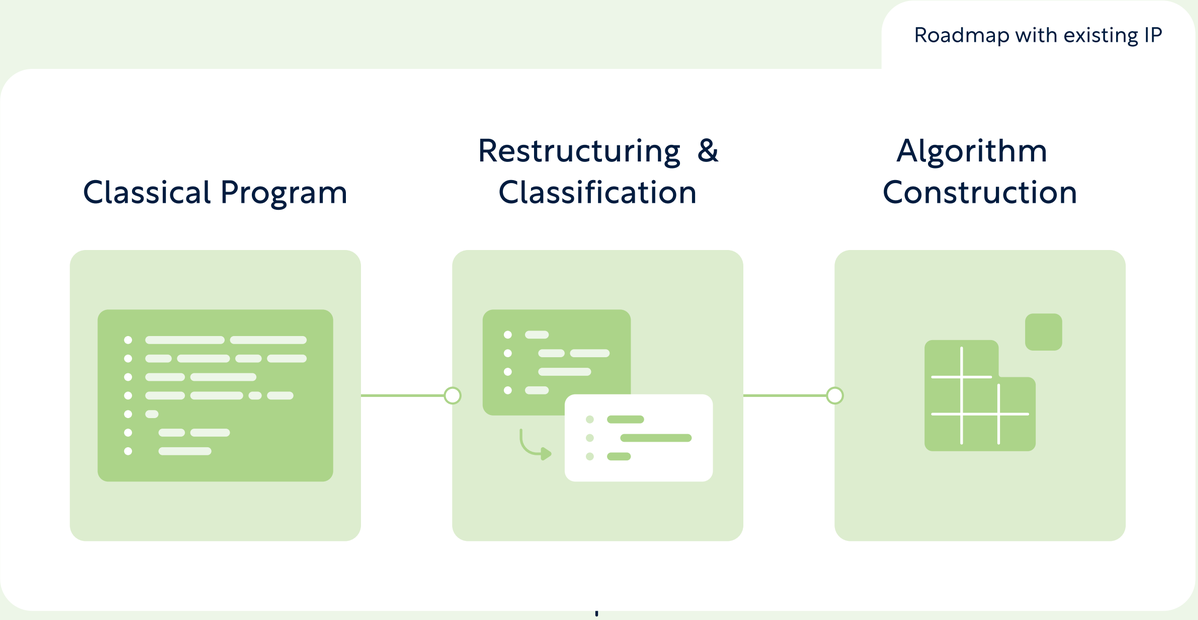
software is needed to drive quantum advantage.

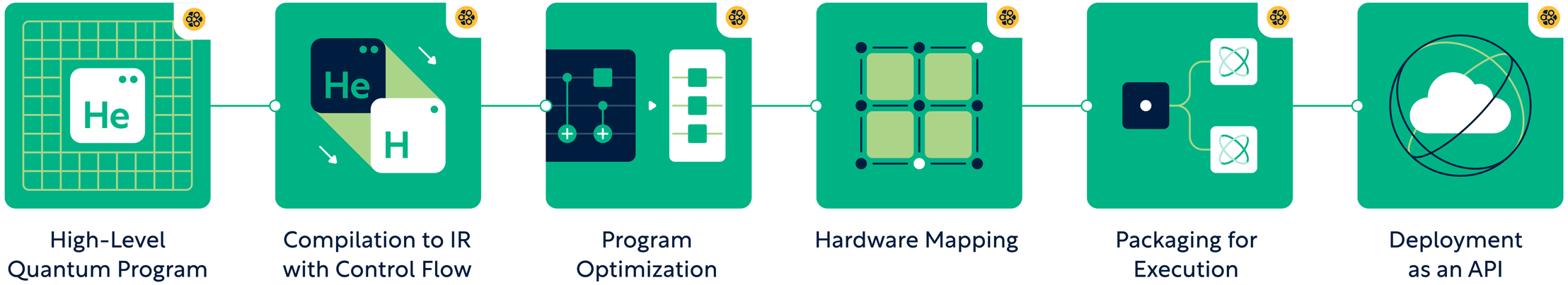


What if programming a quantum computer **was as simple as writing a Python script?**

We are developing technology to bridge from **classical to quantum**

We are executing an ambitious plan to enable software developers to access the power of quantum computing by developing tools to automatically accelerate classical software using quantum processing.



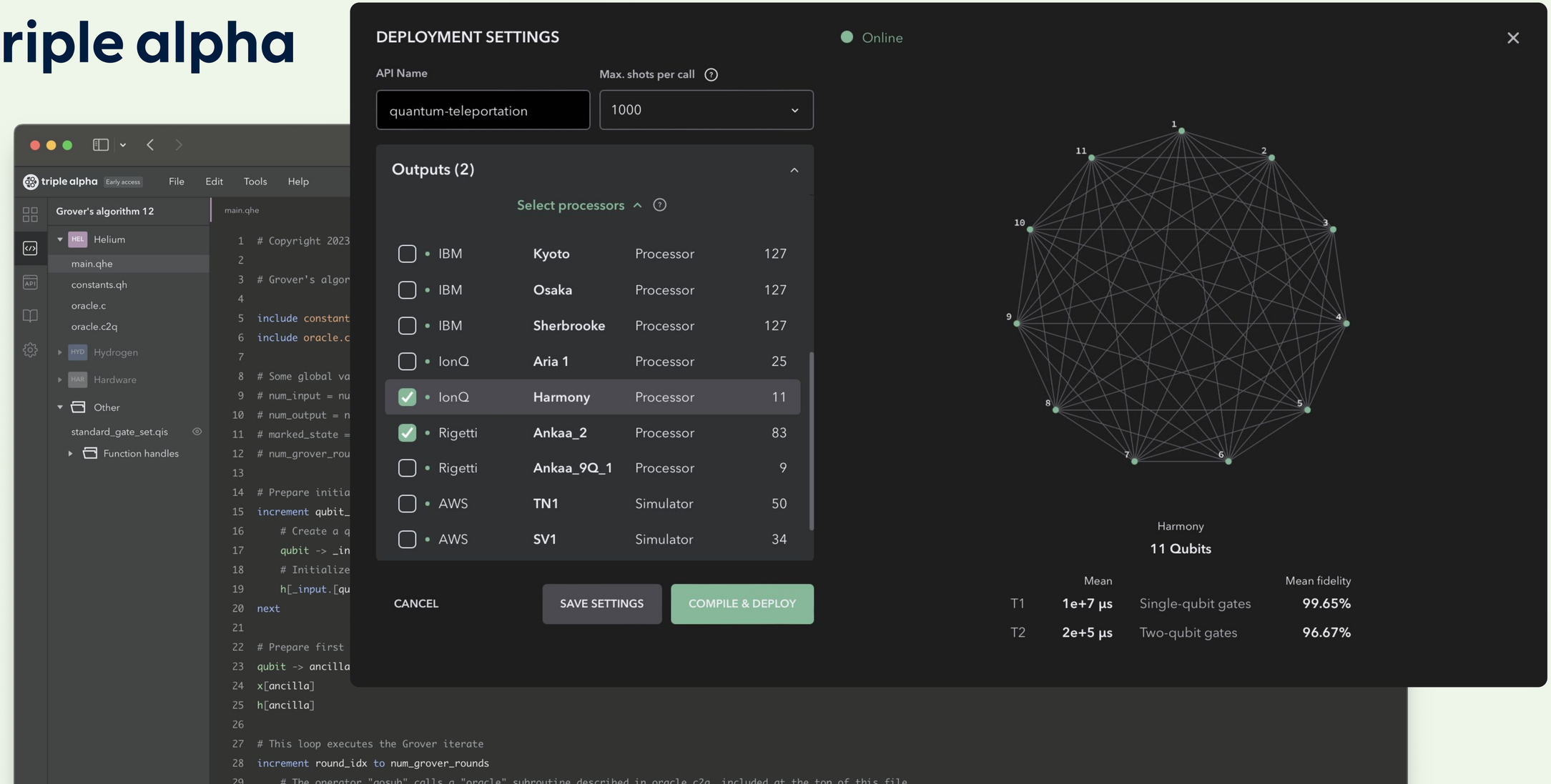


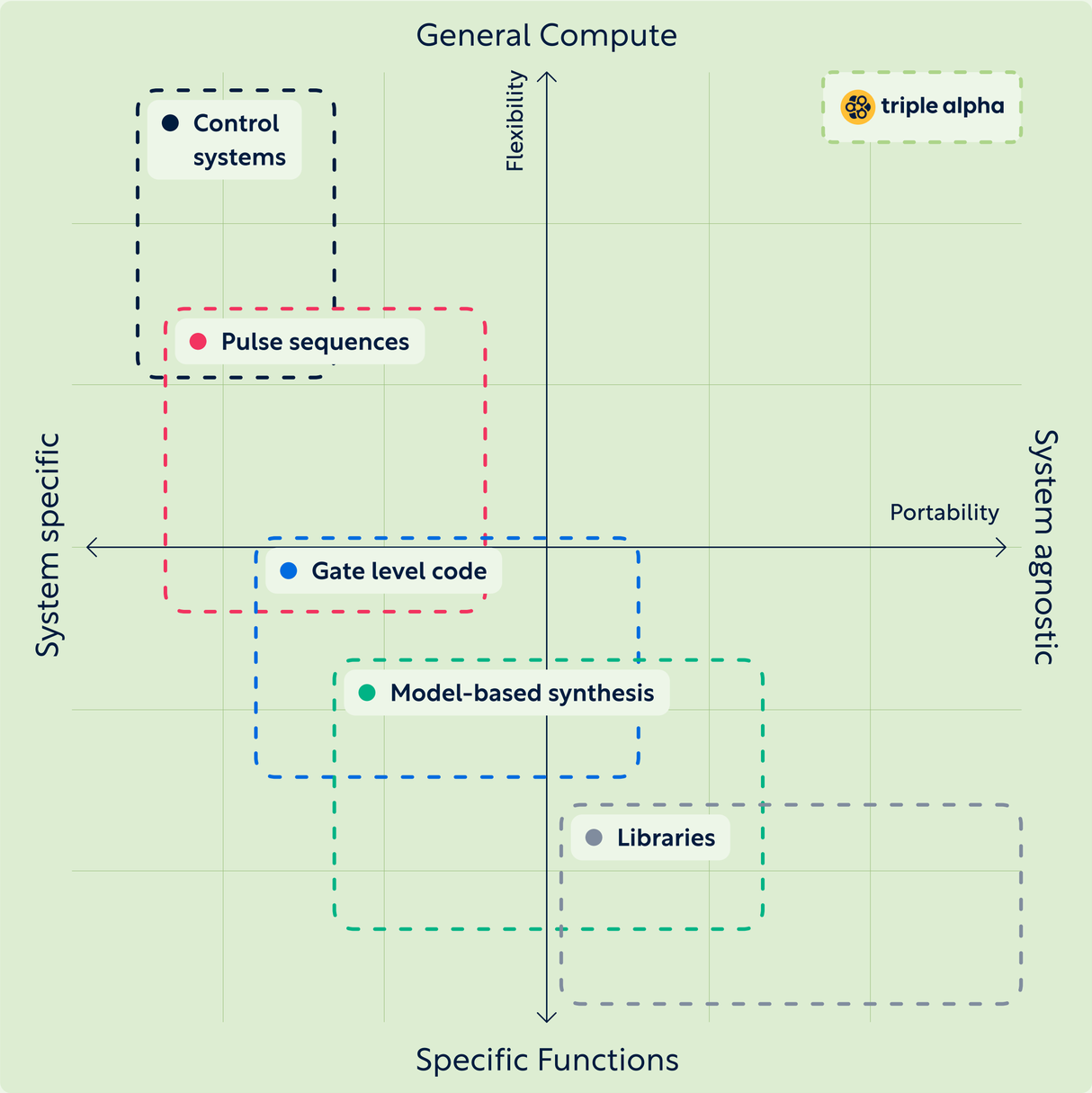
Currently available in **Triple Alpha**

We are already enabling developers to **code, compile and deploy** sophisticated applications with Triple Alpha

Triple Alpha is currently in early access with select hardware and software partners

triple alpha





Source: Functionality analysis diagram prepared by Horizon Quantum management for illustrative purposes.

Triple Alpha makes serious quantum software development possible

Existing approaches to programming quantum computers usually navigate a trade-off between flexibility and portability. Triple Alpha avoids this trade-off entirely, offering a high level of control while remaining hardware agnostic.

Control systems

QUA (Quantum Machines), **LabOne Q** (Zurich Instruments)

Pulse sequences

OpenPulse (IBM), **Quil-t** (Rigetti)

Gate level code

Qiskit (IBM), **PyQuil** (Rigetti), **Circ** (Google)

Model-based synthesis

Quantum Algorithm Design Platform (Classiq)

Libraries

OpenFermion, **TensorFlow Quantum** (Google), **PennyLane** (Xanadu)

Our runtime environment is emerging as a true quantum operating system kernel

Triple Alpha overcomes limitations of hardware backends in software, seamlessly stitching together multiple hardware calls to enable enhanced functionality, allowing developers to do much more with existing hardware systems. With capabilities like dynamic memory allocation and network I/O, we believe Triple Alpha is laying the foundation for the first true quantum OS kernel.

CURRENT CAPABILITIES	IBM	Rigetti	IonQ	IQM	OQC	with Triple Alpha
Mid-circuit measurement	✓	✓	✗	✓	✗	✓
Control flow (if/else, while loops)	⚠	✓	✗	⚠	✗	✓
Concurrent classical functions	✗	✗	✗	✗	✗	✓
Dynamic memory allocation	✗	✗	✗	✗	✗	✓
Network I/O mid computation	✗	✗	✗	✗	✗	✓

Source: Current capabilities table prepared by Horizon Quantum management for illustrative purposes.

Our tools are designed to set a new standard for **quantum software infrastructure**

Development Infrastructure

Triple Alpha allows developers to create programs using high-level programming languages and compile them to optimised system-specific code.

- High level programming
- Optimising complier
- Portable low level intermediate representation (IR)
- Instruction set translation
- Program optimisation engine
- Scheduling of pulse level operations
- Generation of system specific code
- Resource planning / estimation

Deployment Infrastructure

Our deployment infrastructure allows programs to be deployed as APIs for easy integration between quantum program and user-facing interface.

- Deployment of programs as APIs
- Easy integration with front-end technologies
- Authentication
- Session management for deployed programs
- Program execution management
- Status monitoring
- Usage monitoring

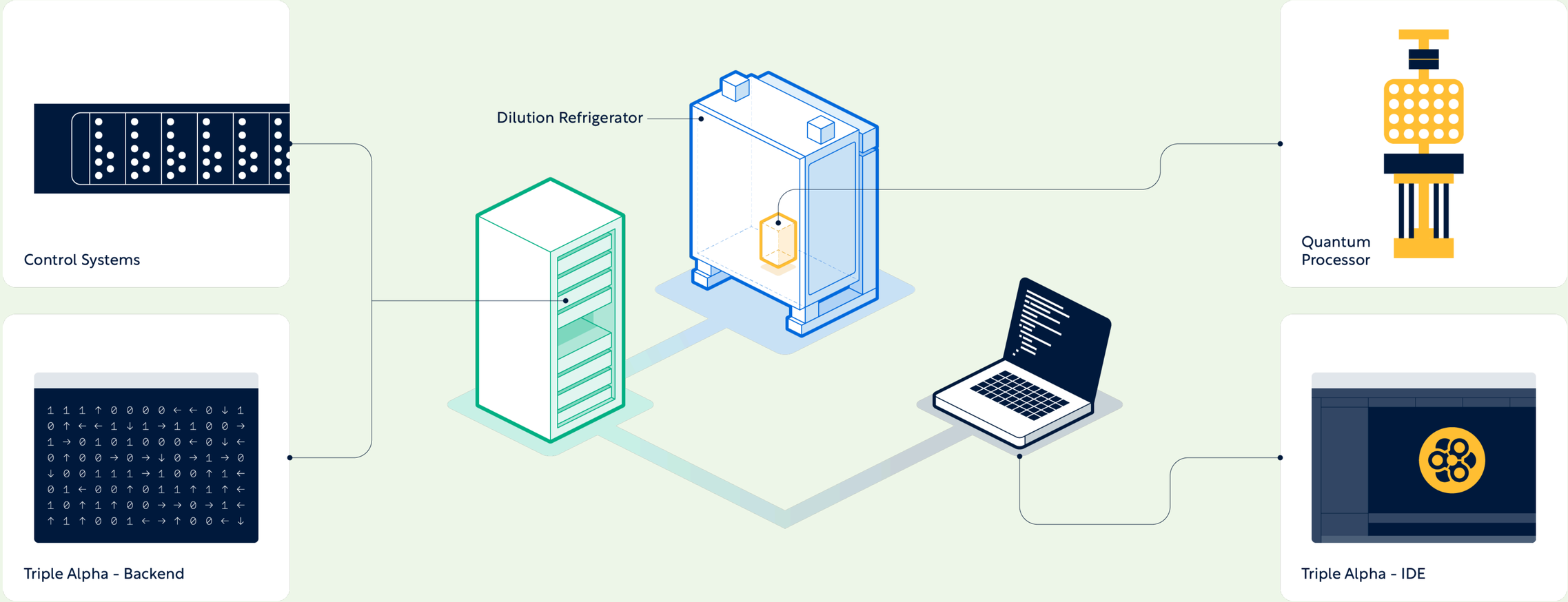
Execution Infrastructure

Our execution stack enables programs to be run on a wide variety of quantum computers and simulators, using advanced techniques to extend system capabilities.

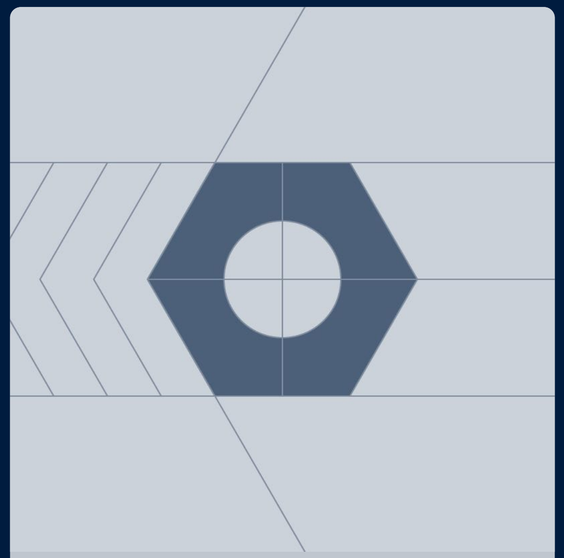
- Control systems abstraction
- Full control flow support
- Full mid-computation measurement support
- Full mid-computation classical function support
- Full mid-computation I/O support
- Control system logic
- Dynamic memory management
- Execution on hardware systems

We take hardware integration seriously **by operating our own quantum computers**

Horizon Quantum has completed the assembly and integration of a quantum computer to become the first quantum software company to own and run a quantum computer. Our on-site testbed has capacity for additional quantum computers. We view tight integration between hardware and software as critical to realizing the full potential of quantum computing.

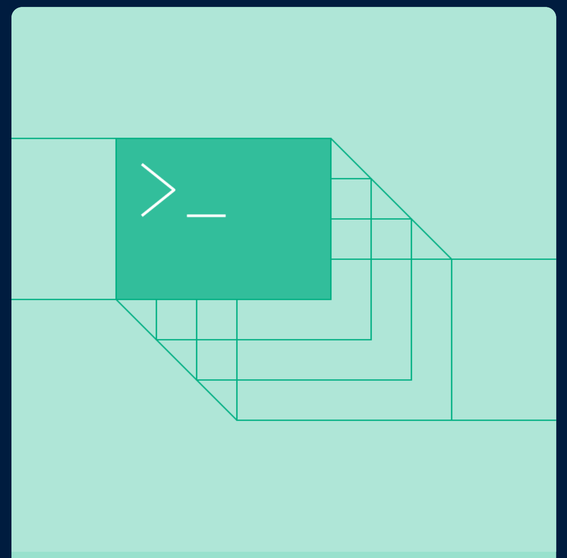


Our strategy positions us to grow as quantum computing scales



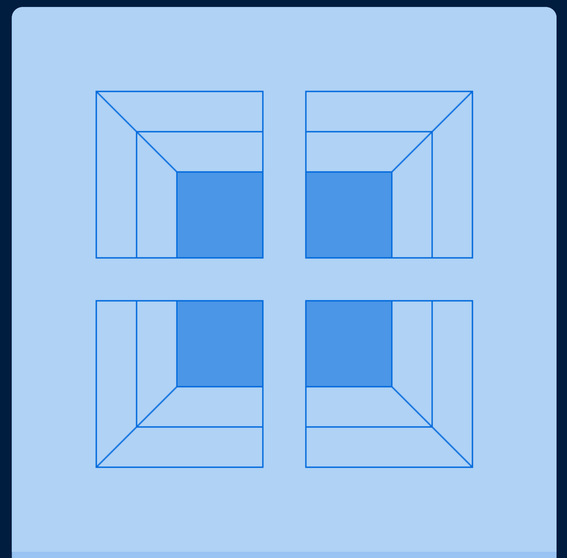
Hardware Partners

QPU
Control Systems



Software Partners

Quantum software
Developers



Academia and Alternative Partners

Research Groups
Cloud Providers

Inbound access requests received from :

40+
Major Corporations

95+ / 9
Universities / Times Higher Education Top 20

15+
National labs, government agencies and research institutes.

10+
Quantum software companies

Our approach with partners is designed to enable our tools to become **the default software infrastructure for hardware vendors**

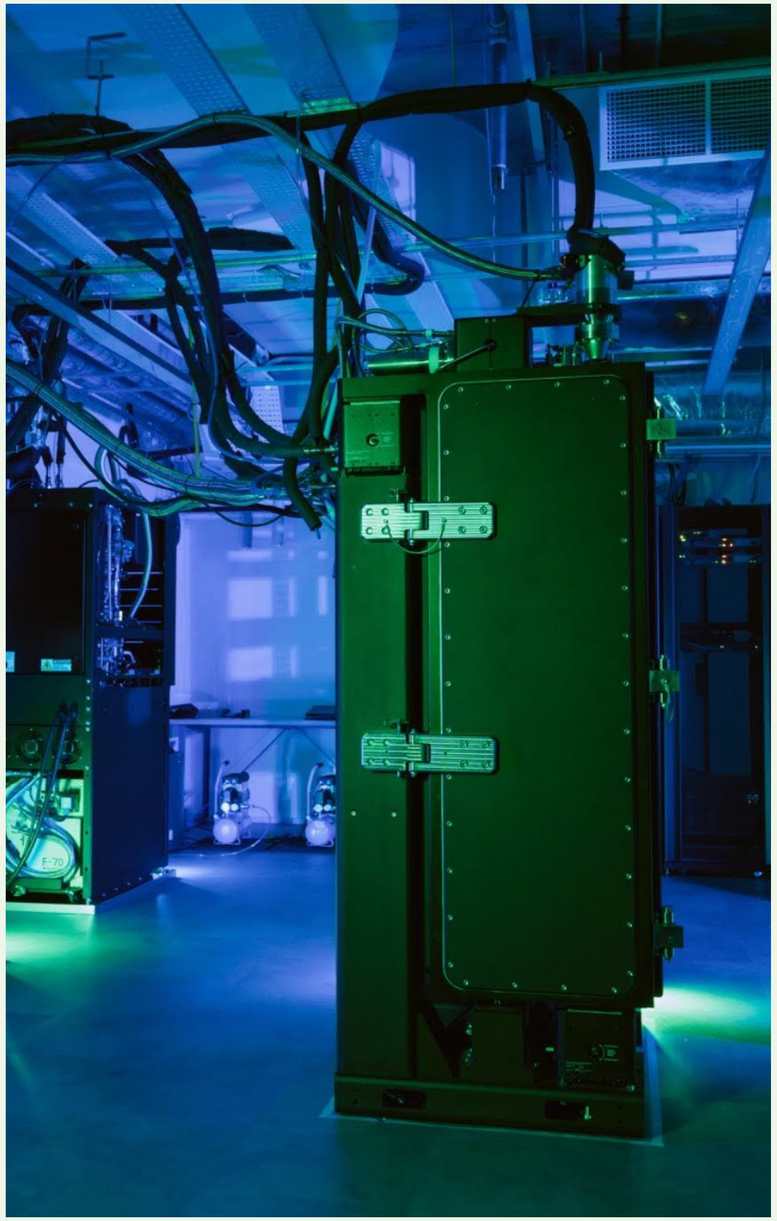
Our approach with partners provides a pathway for manufacturers to **increasingly integrate their hardware with Triple Alpha.** This enables us to offer hardware from a manufacturer to our users at an early stage of engagement and potentially provides a path for Triple Alpha to become **the default software stack for their customers**.

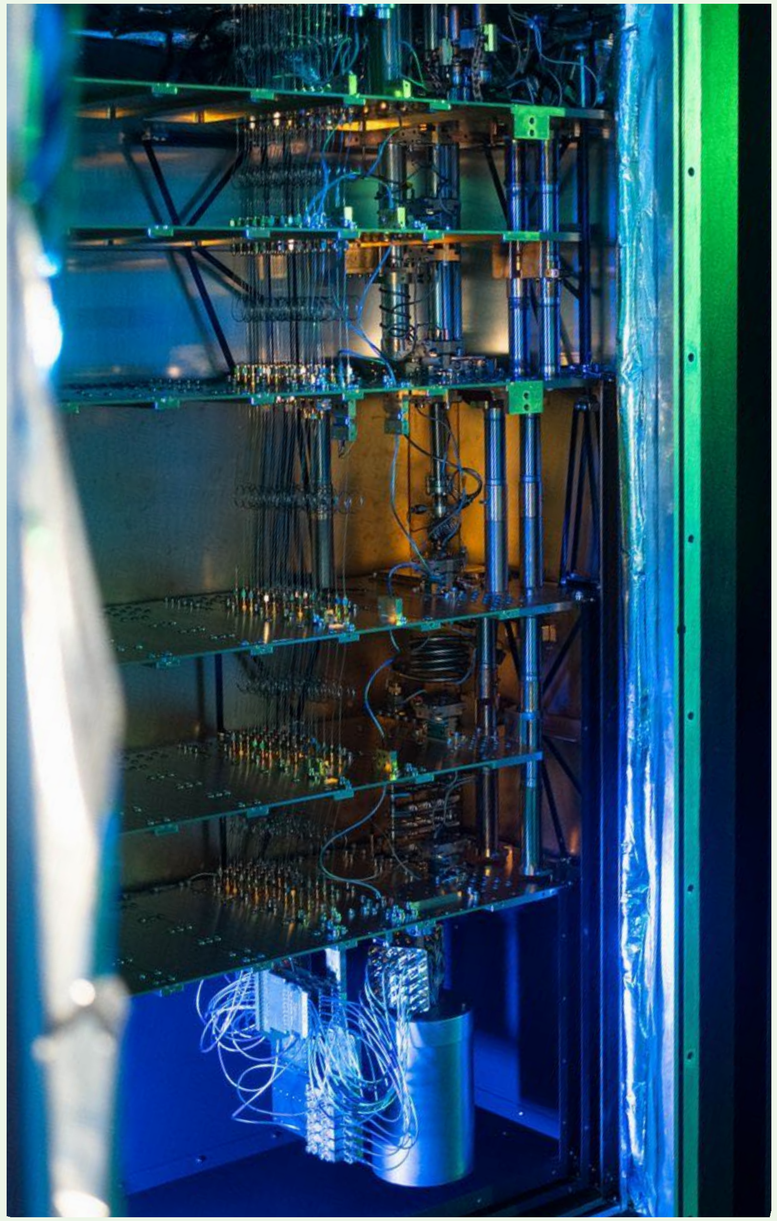
INCREASING SOFTWARE / HARDWARE INTEGRATION MODEL

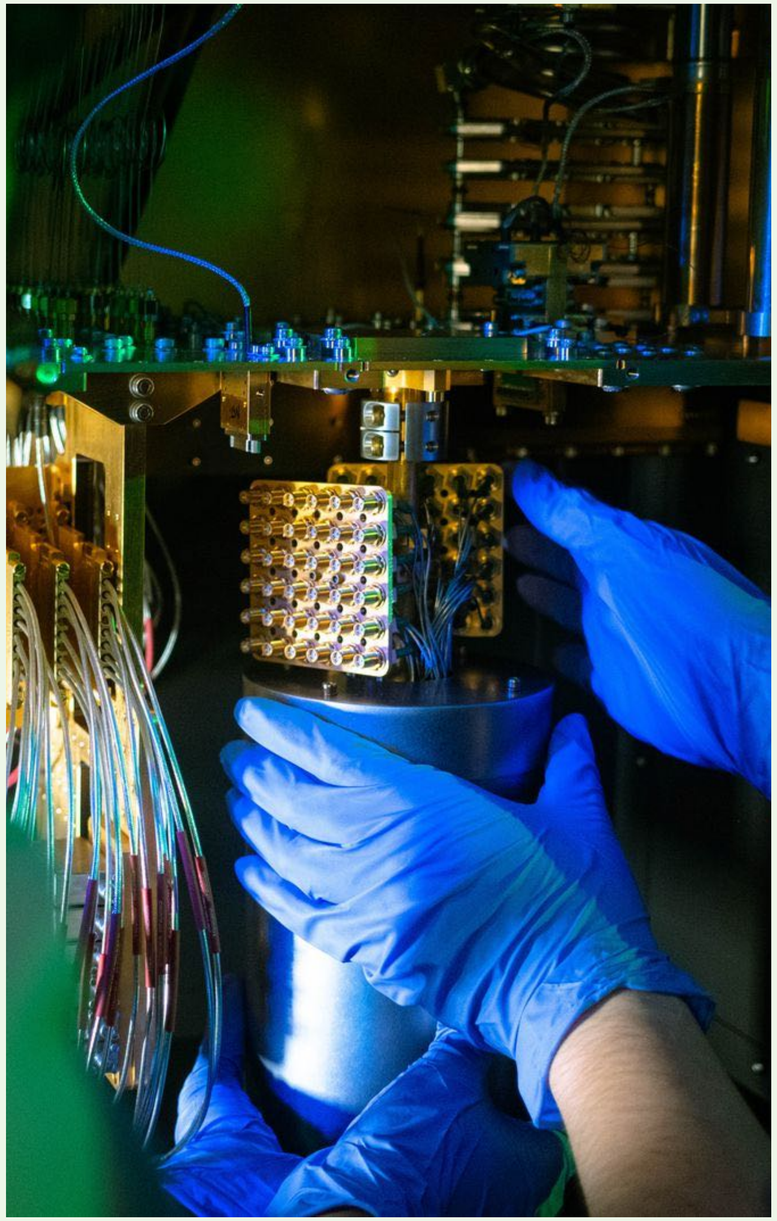
- Triple Alpha becomes the default software layer on the hardware platform
- Hardware manufacturer makes Triple Alpha available to its customers
- Horizon Quantum works with hardware vendor to solve QAI challenges the manufacturer faces
- Hardware becomes available to Horizon Quantum's external collaborators using Triple Alpha
- Hardware becomes available to Horizon Quantum for internal use within Triple Alpha

Applications run through our infrastructure.

This is expected to enable usage-based pricing that aligns with **value creation for developers**.





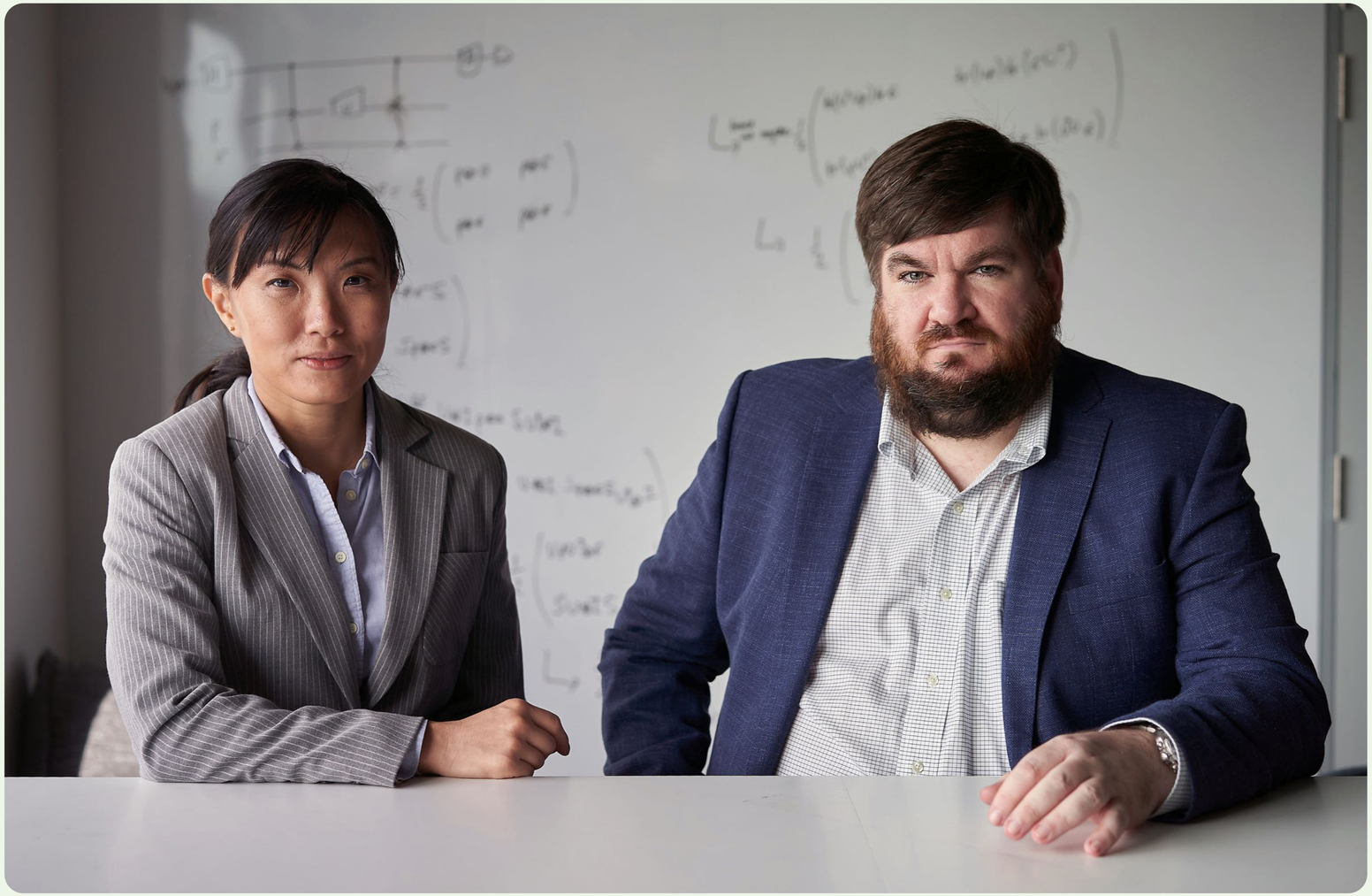


Testbed systems

At a very coarse grain level, quantum computers can be broadly split into two categories: solid-state qubits and AMO qubits. Our testbed systems cover both categories.

	Solid-state qubits	AMO qubits
Underlying physics	Engineered qubits based on solid-state physics	Natural qubits based on atomic, molecular and optical (AMO) physics
Flexibility and variability	Flexible parameters but high variability across devices	Structure limited by nature, but qubits are fundamentally identical
Operating requirements	Typically require electrical or microwave control and cryogenic environments	Typically require optical stability, high vacuum
Testbed system	Ember-1 (Rigetti, Quantum Machines and Maybell)	Upcoming IonQ 256-qubit system

We've assembled a world-class, deep-science team to **close the gap between quantum hardware and practical application**



Dr Joe Fitzsimons

Chief Executive Officer

Invented blind quantum computing

University of Oxford · Centre for Quantum Technologies · Singapore University of Technology and Design

Over 20 years experience in quantum computing, with key contributions to system architectures, computational complexity theory and applications of quantum computing.

Dr Si-Hui Tan

Chief Science Officer

Pioneered applications of quantum illumination

Caltech · Massachusetts Institute of Technology · Agency for Science, Technology and Research, Singapore · Singapore University of Technology and Design

20 years experience in quantum information science, contributing to optical quantum information processing, quantum networking and secure quantum computing.

A team with **experience across technology organisations,** both big and small

Greg Gould
Chief Financial Officer
Goldman Sachs

Catherine Fitzsimons
Chief Legal & Compliance Officer
Fidelity International

Amanda Chew
Chief Product Officer
Microsoft

Dr Raymond Lloyd
SVP of Engineering
IBM

Eoin Scanlon
VP of Operations

Philip Tan
VP of Commercial Operations
Graphcore
Lenovo

Sumanth Puttur
VP of People
Google

Katherine Bailon
VP of Investor Relations
Goldman Sachs

Dr Denis Chevallier
Director of Hardware
bleximo

Yanina Blaclard
VP of Marketing and Communications
Societe Generale

A board of directors that adds world-class experience

Dr Joe Fitzsimons

Founder & Chief Executive Officer of Horizon Quantum, Chairman of the Board

Quantum computing pioneer with over 20 years' experience, having made key novel contributions to system architectures, computational complexity theory and applications of quantum computing.

University of Oxford · Centre for Quantum Technologies · Singapore University of Technology and Design

Danielle Lambert

Former VP of Human Resources Apple Computer

As the head of Apple HR, helped build out the teams for the iPod, iPhone, iPad and Apple Retail Stores. Played a pivotal role in the founding of Nest Labs as an initial investor and advisor through the company's rapid growth and acquisition by Google. Active advisor and investor in a broad range of tech and consumer companies.

Nest

Peter Oey

Chief Financial Officer of Grab

CFO for the prominent Southeast Asia superapp Grab and a seasoned executive with 25+ years in corporate finance spanning both Fortune 100 including Activision and private companies including LegalZoom. Leads Grab's financial operations, including corporate accounting & reporting, treasury, financial planning & analysis and investor relations, among others.

Grab · Activision · LegalZoom

Jill Turner

Chief Human Resources Officer of Broadcom

Global Human Resources Executive with 20+ years of experience at leading Fortune 500 technology companies. Proven strategic advisor with experience in corporate governance, executive compensation, large-scale M&A integration, succession planning, and enhanced shareholder value. Previous senior HR leadership roles at Lumen Technologies (formerly CenturyLink) and Honeywell.

Broadcom · Honeywell · Lumen

Harry You

Chairman of dMY Technology Group and Member of Executive Committee of Broadcom

Business leader with 40 years of unmatched deep management experience across technology broadly and quantum specifically, creating significant shareholder value at Broadcom (Board Member), IonQ (former Lead Independent Director), EMC (former EVP who structured Dell's buyout of EMC), Oracle (former CFO), Accenture (former CFO), and BearingPoint (former CEO),

EMC² · IonQ · Oracle · accenture · Broadcom

Quantum computing has reached an inflection point

PRACTICAL QUANTUM ADVANTAGE IS EXPECTED IN THE COMING YEARS

Error correction is now a reality

Quantum error correction has **progressed rapidly in the last 18 months**, with full quantum error correction convincingly demonstrated for the first time in late 2024.

Quantum computers are now hard to simulate

Convincing demonstrations have shown that classical computers can no longer easily simulate quantum computers.

Overhead is tumbling

Not long ago, it was thought that 1,000+ physical qubits would be required to produce an error-free logical qubit. Recent progress in error correction codes has reduced this by orders of magnitude.

New qubit platforms are emerging

Quantum computers based on **neutral atoms** are scaling up rapidly, and new qubit architectures exploiting biased noise are showing potential shortcuts to error-free quantum computation.

In past technology cycles, **software has captured more value than hardware**.

Quantum computing may follow this trend.

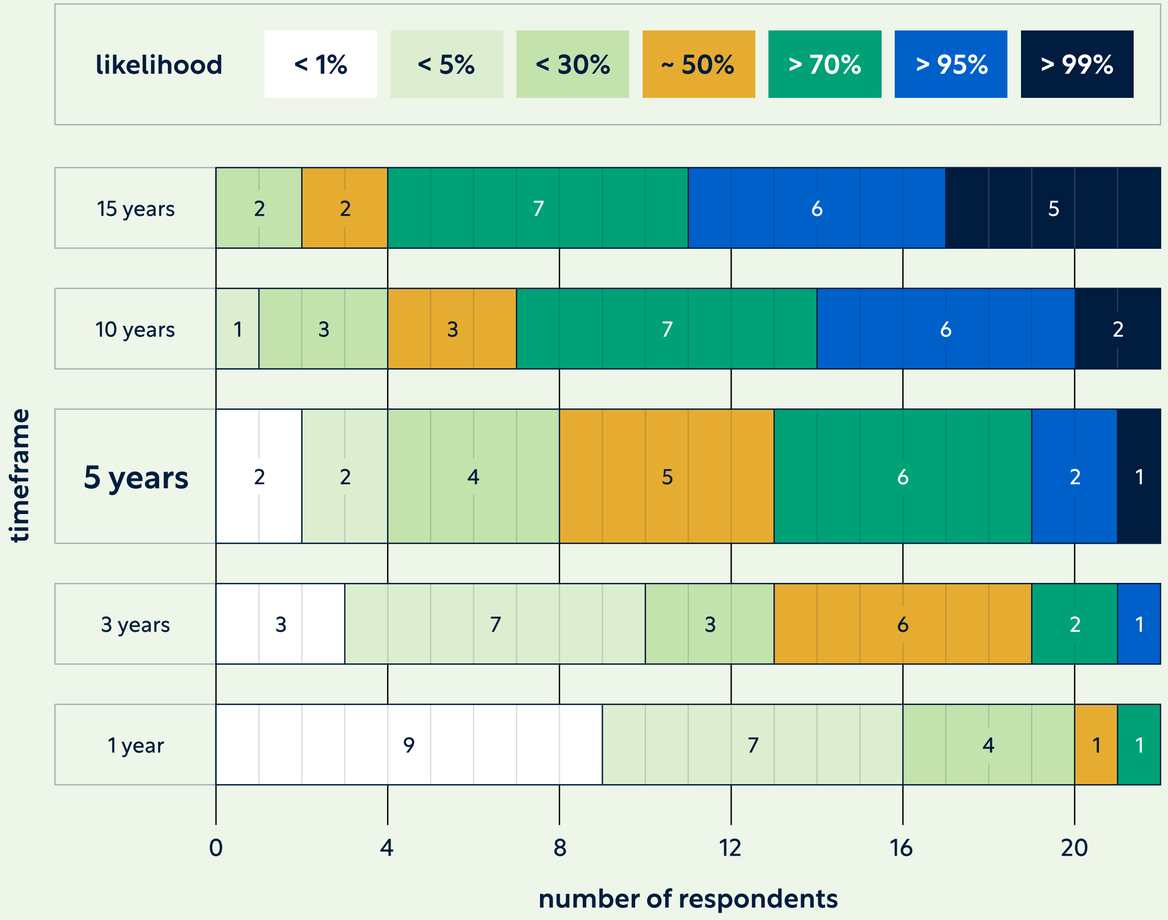
Commercial applications are likely to emerge soon, **the application infrastructure needs to be ready**

In a 2024 report on the state of quantum computing, most of the quantum computing experts polled anticipated commercial applications to emerge within the next 5 years.

Developers and domain experts need the infrastructure to experiment, test, and build applications before commercial demand takes off.

2024 EXPERTS' ESTIMATES OF LIKELIHOOD OF COMMERCIAL APPLICATIONS FOR EARLY QUANTUM COMPUTERS

Number of experts who indicated a certain likelihood in each indicated timeframe



Source: Global Risk Institute — Quantum Threat Timeline Report 2024

Condensed consolidated balance sheet (Unaudited)

(In US$, unless otherwise stated)	June 30, 2026	December 31, 2025
ASSETS		
Current assets		
Cash and cash equivalents	$ 113,254,440	$ 222,939
Prepaid and other current assets	6,868,373	746,372
Total current assets	**120,122,813**	**969,311**
Property and equipment, net	3,489,535	3,204,829
Intangible assets, net	20,392	22,566
Right-of-use assets	361,307	459,982
Other non-current assets	51,716	175,115
TOTAL ASSETS	**$ 124,045,763**	**$ 4,831,803**
LIABILITIES AND STOCKHOLDERS' EQUITY		
Current liabilities		
Derivative liabilities - SAFE	$ -	$ 6,406,878
Other payables	3,224,136	2,602,604
Operating lease liabilities	325,126	396,025
Total current liabilities	**3,549,262**	**9,405,507**
Derivative liabilities - warrants	76,778,574	-
Operating lease liabilities, non-current	42,393	88,921
TOTAL LIABILITIES	**$ 80,370,229**	**$ 9,494,428**

(In US$, unless otherwise stated)	March 31, 2026	December 31, 2025
STOCKHOLDERS' EQUITY		
Seed Preferred Shares, 2,500,000 authorized; 2,500,000 issued and outstanding as of December 31, 2025	$ -	$ 839,602
Seed Plus Preferred Shares, 2,936,828 authorized; 2,936,828 issued and outstanding as of December 31, 2025	-	2,349,212
Series A Preferred Shares, 2,586,522 authorized; 2,586,522 issued and outstanding as of December 31, 2025	-	18,100,000
Ordinary Shares, 8,000,000 authorized; 8,000,000 issued and outstanding as of December 31, 2025	-	3,649
Ordinary Class A Shares, 34,227,495 authorized, 34,227,495 issued and outstanding as of June 30, 2026	187,124,075	-
Ordinary Class B Shares, 19,744,585 authorized, 19,744,585 issued and outstanding as of June 30, 2026	40,110	-
Additional paid-in capital	8,997,535	7,417,778
Equity proceeds receivable	(343,862)	-
Accumulated deficit	(152,439,504)	(33,573,537)
Accumulated other comprehensive (loss) income	297,180	200,671
TOTAL STOCKHOLDERS' EQUITY	**$ 43,675,534**	**$ (4,662,625)**
TOTAL LIABILITIES AND STOCKHOLDERS' EQUITY	**$ 124,045,763**	**$ 4,831,803**

Note: All results presented prior to the closing of the business combination on March 19, 2026, reflect the financial results of Horizon Quantum Pte. Ltd. Results as of June 30, 2026, reflect Horizon Quantum Holdings Ltd.

Condensed consolidated statement of operations and comprehensive loss (Unaudited)

(In US$, except share amount and per share data)	Three Months Ended June 30, 2026	Three Months Ended June 30, 2025	Six Months Ended June 30, 2026	Six Months Ended June 30, 2025
Revenue	$ -	$ 38,462	$ -	$ 38,462
Operating Expenses:				
Research and development	2,614,022	1,206,612	4,742,435	4,527,603
Selling and marketing	375,989	248,716	834,952	576,665
General and administrative	3,849,526	1,146,373	7,451,257	2,047,525
Depreciation and amortization	335,966	180,787	646,233	349,836
Total operating expenses	**7,175,504**	**2,782,488**	**13,674,878**	**7,501,629**
Loss from operations	**(7,175,504)**	**(2,744,027)**	**(13,674,878)**	**(7,463,167)**

(In US$, except share amount and per share data)	Three Months Ended June 30, 2026	Three Months Ended June 30, 2025	Six Months Ended June 30, 2026	Six Months Ended June 30, 2025
Other income and (expense):				
Interest expense	(2,694)	(2,320)	(5,686)	(4,830)
Other income	487,055	16,943	530,443	49,730
Change in fair value of derivative liabilities	(108,294,223)	-	(105,317,692)	-
Foreign exchange (loss) gain	(241,982)	(172,909)	(318,912)	(306,927)
Income tax expense	-	-	-	-
Net loss	**$ (115,227,348)**	**$ (2,902,313)**	**$ (118,786,726)**	**$ (7,725,194)**
Other comprehensive loss:				
Foreign currency translation adjustment	62,202	72,683	96,509	290,671
Comprehensive loss	**$ (115,165,146)**	**$ (2,829,630)**	**$ (118,690,217)**	**$ (7,434,523)**
Basic and diluted weighted average ordinary shares outstanding, as recast	52,367,347	39,015,950	46,630,787	39,015,950
Net (loss) income per ordinary share, basic and diluted, as recast	$ (2.20)	$ (0.07)	$ (2.55)	$ (0.20)

Note: All results presented prior to the closing of the business combination on March 19, 2026, reflect the financial results of Horizon Quantum Pte. Ltd. Results as of June 30, 2026, reflect Horizon Quantum Holdings Ltd.

Reconciliation of Non-GAAP Financial Measures

Below is a reconciliation of net loss (GAAP) to adjusted EBITDA:

	Three Months Ended June 30,		Six Months Ended June 30,	
(In US$, unless otherwise stated)	**2026**	**2025**	**2026**	**2025**
Net loss (GAAP)	**$ (115,227,348)**	**$ (2,902,313)**	**$ (118,786,726)**	**$ (7,725,194)**
Adjustments				
Net interest (income) expense	(477,253	(12,805)	(512,778)	(42,915)
Depreciation and amortization expenses	335,966	180,787	646,233	349,836
EBITDA	**(115,368,635)**	**(2,734,330)**	**(118,653,271)**	**(7,418,272)**
Adjustments				
Share based compensation within				
Research and development	436,808	261,285	676,982	2,778,608
Selling and marketing	46,320	65,299	92,386	170,448
General and administrative	509,959	130,294	839,994	437,675
Change in fair value of derivative liabilities	108,294,223	-	105,317,692	-
Business combination expenses*	622,980	214,101	2,164,930	214,101
Adjusted EBITDA	**$ (5,458,343)**	**$ (2,063,351)**	**$ (9,561,286)**	**$ (3,817,439)**

*Includes $0.27 million in share-based compensation for a vendor in the six months ended June 30, 2026.

Note: All results presented prior to the closing of the business combination on March 19, 2026, reflect the financial results of Horizon Quantum Computing Pte. Ltd. Results as of June 30, 2026, reflect Horizon Quantum Holdings Ltd.

Condensed consolidated statement of cashflow (Unaudited)

(In US$, unless otherwise stated)	Six Months Ended June 30, 2026	Six Months Ended June 30, 2025
Cash flows from operating activities		
Loss for the period	$ (118,786,726)	$ (7,725,194)
Adjustments to reconcile net loss to net cash used for operating activities:		
Depreciation	644,059	349,035
Share based compensation	1,878,363	3,387,475
Change in fair value of derivative liabilities	105,317,692	-
Unrealized foreign currency transaction (gain) loss	85,088	629,163
Amortization	2,174	801
Changes in operating assets and liabilities:		
Accounts receivable	-	270,824
Other payables	1,890,795	(241,801)
Lease liability	(212,283)	(131,790)
Prepaid expenses and other assets	(102,350)	(189,219)
Net cash used in operating activities	**(9,283,188)**	**(3,650,706)**

(In US$, unless otherwise stated)	Six Months Ended June 30, 2026	Six Months Ended June 30, 2025
Cash flows from investing activities		
Purchase of property, equipment including construction in progress	(5,521,276)	(297,706)
Purchase of intangible assets and trademarks	-	-
Net cash used in investing activities	**(5,521,276)**	**(297,706)**
Cash flows from financing activities		
Proceeds from issuance of SAFE notes	2,500,000	-
Proceeds from exercise of warrants	27,186,518	-
Proceeds from merger and PIPE transaction, net of transaction costs	98,167,633	-
Net cash provided by financing activities	**127,854,151**	**-**
Effect of exchange rate changes on cash	(18,186)	(194,118)
Net (decrease) increase in cash and cash equivalents	113,031,501	(4,142,530)
Cash and cash equivalents at beginning of period	222,939	4,848,855
Cash and cash equivalents at end of period	**$ 113,254,440**	**$ 706,325**
Supplemental disclosures of non-cash transactions:		
Initial recognition of warrant liabilities at close of the business combination	$ 20,526,410	$ -
Initial recognition of net assets at close of the business combination	2,458,713	-
Conversion of SAFE liabilities into equity at close of the business combination	11,183,077	-
Issuance of shares to a service provider	269,000	-
Reclassification of warrant liabilities to equity upon exercise	46,789,330	-

Note: All results presented prior to the closing of the business combination on March 19, 2026, reflect the financial results of Horizon Quantum Pte. Ltd. Results as of June 30, 2026, reflect Horizon Quantum Holdings Ltd.

Capital-efficient **software & SaaS model** in **quantum computing**

Although we depend on quantum advantage being achieved, we enjoy a number of strengths:

- Strong technical team with **proven track record in quantum computing**
- **Compelling technology** to power quantum computing applications
- Software business model with **lower capex requirements** than quantum hardware companies
- Hardware agnostic approach designed to make success **independent of winning hardware technology**
- Anticipated **value-based pricing model**
- Business model **designed to be sticky**

horizon quantum

For more information

investors.horizonquantum.com	Katherine Bailon VP of Investor Relations katherine@horizonquantum.com 212-729-1990

Appendix A: Glossary

This Glossary explains several terms used in the Horizon Quantum Investor Presentation. The Glossary, as prepared by Horizon Quantum management, is not a comprehensive explanatory document. Shareholders and investors should not unduly rely on it to make investment decisions.

GLOSSARY

Algorithm — An algorithm is a finite set of step-by-step instructions for solving a type of problem or carrying out a computation. Algorithms are a cornerstone of computer science and form the building blocks of computer programs. They are used for calculation, data processing, and automation, and they range from simple procedures (such as sorting) to advanced methods (such as combinatorial optimization).

Algorithm construction — The process of designing and specifying algorithms to solve problems or perform computations. For conventional computing, this involves creating algorithms that run on classical hardware, following deterministic, step-by-step logic. For quantum computing, algorithm construction requires exploiting quantum effects—such as interference, superposition, and entanglement—to design methods that can sometimes solve problems more efficiently than classical algorithms.

API (Application programming interface): An API is a standard way for different software systems to talk to each other. It lets one program use the functions or data of another without needing to know how that other program is built. APIs make it easier for developers to connect tools, build new applications, and share data across software systems.

Compiler — A compiler is a software tool that translates code written in a high-level programming language into a form that computer's hardware can execute. A compiler may translate step by step from one abstraction level to the next, or all the way down to native instructions for the hardware. Compilation happens before the program runs, allowing developers to code in languages that are easier to read and write while still producing instructions the machine can understand.

Circuit (quantum circuit) — In quantum computing, a circuit is a sequence of operations, consisting of quantum gates and measurements, carried out on qubits within a quantum processing unit (QPU). Much like logic circuits express computations in classical computers, a quantum circuit is the basic way of expressing a computation on a quantum computer. By arranging operations in different sequences, quantum circuits implement algorithms, and the interplay of gates within them gives rise to quantum interference, enabling quantum computers to perform computations that a classical computer cannot. Circuits are limited, however, in that they cannot implement control flow or determine which operation to perform dynamically.

GLOSSARY (CONTINUED)

Concurrent classical functions — In quantum computing, concurrent classical functions are classical computations that run at the same time as a quantum circuit, rather than only before or after it. Traditionally, quantum programs are structured as static circuits: the circuit is designed in advance, executed on the quantum processor, and then the results are processed classically. Concurrent classical functions allow classical and quantum steps to be interwoven more tightly. For example, a classical function may process a mid-circuit measurement result and immediately decide which gate to apply next, or update parameters on the fly to steer the quantum computation dynamically. This concurrent execution is powerful because it allows feedback loops between classical and quantum processes, making quantum algorithms more flexible and efficient. It is also a necessary ingredient for important functionality such as error correction.

Control flow (if/else, while loops) — In classical computing, control flow is the order in which individual instructions in a program are executed. Instead of always running commands in a straight line from top to bottom, programs can make decisions or repeat steps based on conditions. For example, an if/else statement allows a program to take different paths depending on whether a condition is true or false. A while loop makes the program repeat a block of instructions until a condition is no longer met. These control flow structures make it possible to write programs that adapt to different inputs and situations.

In quantum computing, control flow is much more challenging. Traditional quantum programming frameworks have typically been confined to static circuits, where all the operations are fixed in advance. This limits the range of programs expressible in those frameworks.

Control systems — A quantum control system is the combination of hardware and software used to manipulate and measure qubits in a quantum computer or experiment. It relies on classical electronics to generate highly precise signals that drive qubit operations and to process measurement feedback in real time. These systems enable complex tasks such as characterization of the physical properties of the underlying processors, calibration of specific gate operations, error correction protocols, and gate sequences, while also suppressing noise and maintaining the stability of sensitive quantum hardware. Quantum control systems are typically tailored to the underlying qubit technology—for example, superconducting qubits, trapped ions, or photons each require different types of control signals and electronics.

GLOSSARY (CONTINUED)

Dynamic memory allocation — In classical computing, dynamic memory allocation is the process of assigning memory to a program while it is running, rather than fixing the memory size in advance. Instead of the programmer deciding ahead of time how much memory is needed, the program can request and release memory as it runs. For example, with static allocation, if a programmer declares a data structure with a fixed length in C, its size is set at compile time. If the programmer doesn't know how big the data structure needs to be until the program is running, they can use dynamic allocation to get exactly as much memory as is needed.

Dynamic memory allocation is important because it makes programs more flexible, efficient and scalable. It allows them to handle data whose size is not known until runtime, reuse memory by freeing it when it is no longer needed, and adapt to the amount of memory available on a machine. Without it, programmers would have to guess memory needs in advance, which could waste memory if they overestimate or cause the program to fail if they underestimate.

While this type of memory management is standard in classical computing, it remains rare in quantum systems. Traditional quantum circuits pre-label and allocate qubits before execution begins, and most programming frameworks are designed around this static model. However, programs that use control flow may require qubits to be allocated and released on the fly.

Errors (quantum) — Errors in quantum computing arise from the inherent fragility and sensitivity of quantum states to environmental noise, which can cause unintended changes to the state of qubits or the outcome of operations. Errors include bit-flip errors (where a qubit's state changes from 0 to 1 or vice versa) and phase-flip errors (where the relative phase of a superposition is altered), as well as combinations of both. Errors can result from a variety of sources, including gate errors (incorrect operations applied to qubits), measurement errors (mistakes during the readout process), crosstalk (unwanted interactions between nearby qubits), noise (external interference such as electromagnetic signals), and decoherence (the loss of a qubit's quantum state through interactions with its environment). Because qubits are so vulnerable to these effects, errors occur more frequently in quantum systems than in classical computers, making error correction a central challenge for practical quantum computing.

GLOSSARY (CONTINUED)

Error correction — Quantum error correction is a method used to protect quantum information from errors caused by environmental disturbances, hardware imperfections, or unwanted interactions that can cause qubits to change state. Unlike classical error correction, which can rely on duplicating information, quantum error correction must use more sophisticated techniques because quantum information cannot be copied. It is essential for building fault-tolerant quantum computers capable of carrying out long and reliable computations.

Error correction overhead — Error correction overhead refers to the additional resources required to protect information from errors and preserve data integrity. In conventional computing, this takes the form of extra bits added to the original data, which allow errors to be detected and corrected but reduce the efficiency of storage or transmission. In quantum computing, it appears as the large number of physical qubits and operations needed to encode and stabilize a single logical qubit, ensuring the quantum information remains intact in the face of noise or interference. In both cases, stronger protection against errors requires greater overhead, while less overhead means weaker protection.

Gate (classical gate/logic gate) — A logic gate is a basic building block of classical circuits. It takes one or more binary inputs (0 or 1) and produces a single binary output according to a logical rule. For instance, an AND gate outputs 1 only if all inputs are 1. By combining many logic gates, classical circuits can perform complex computations.

Gate (quantum) — A quantum gate is the basic building block of quantum circuits. It changes the state of one or more qubits by applying a reversible mathematical transformation called a unitary operation. Because they are reversible, unitary operations preserve the total probability of all possible outcomes, ensuring that a quantum gate never loses information—unlike many classical logic gates, which discard information. For example, with an AND gate the inputs (binary information of two bits) cannot always be recovered from the output (one bit). The reversible aspect of quantum gates means that when operations are performed on qubits, whether in simple gates or full algorithms, the process can always be reversed, allowing the system to return to an earlier qubit state. Because quantum gates exploit quantum effects such as superposition and entanglement, when combined in circuits, they can perform complex computations beyond the limits of conventional computers.

GLOSSARY (CONTINUED)

Gate-level code — In quantum computing, gate-level code refers to programs written directly in terms of the quantum gates that act on qubits. At this level, the programmer specifies the exact sequence of gate operations—such as Hadamard or CNOT—that make up a quantum circuit. Gate-level code is analogous to assembly language in classical computing: it provides fine-grained control over the hardware but can be difficult to write and maintain for larger algorithms. It requires a specialized skillset in quantum computing, including understanding how gates manipulate qubits and how circuits are constructed.

Hardware Modality — A hardware modality is the physical approach used to build and operate qubits in a quantum computer. Each modality has distinct advantages and challenges. The choice of modality affects factors such as qubit stability, error rates, scalability, and operating conditions, as well as the design and performance of a quantum system. No single modality has yet proven dominant, and multiple approaches are being actively explored in research and industry. Many researchers believe that different modalities may be better suited to different types of problems. Research into the most effective hardware modalities is central to scaling quantum computers and making them practical for real-world applications.

Hardware testbed — In quantum computing, a hardware testbed is an experimental setup where new computing devices and components can be built, tested, and evaluated under real operating conditions. Beyond testing hardware performance, a testbed also provides a platform for integrating hardware with software, allowing researchers to develop, refine, and validate software in a realistic environment. This tight coupling of hardware and software enables more effective testing, faster iteration, and deeper insights into how the two interact.

Integrated Development Environment (IDE) — An integrated development environment is a software suite that provides programmers with the tools they need to write and compile code efficiently. An IDE typically combines features such as a code editor, a compiler, a debugger and project management support into a single application. By bringing these components together, IDEs are designed to maximize developer productivity and streamline the software development process.

GLOSSARY (CONTINUED)

Libraries — In the context of programming quantum computers, a library is a collection of pre-written code that can be called from within a conventional programming language, such as Python. These libraries provide ready-to-use primitives, utilities, and sometimes full algorithms that simplify building quantum workflows. Providing reusable building blocks and standardized interfaces saves users from having to build everything from scratch. While libraries offer convenience and consistency, they can also introduce limitations: they may restrict users to certain programming models, hardware backends, or algorithmic approaches. This makes them easily accessible and well suited for addressing the tasks for which they were designed, but means they are less flexible than custom code.

Logical qubit — A logical qubit is a qubit formed from a group of physical qubits that store an encoding of the information to protect it against errors and enable longer, more reliable computation. Unlike a physical qubit, which refers to the actual hardware, a logical qubit is a higher-level abstraction used in fault-tolerant quantum computing. In classical computers, repetition codes repeat information multiple times to detect and correct errors. Because quantum information cannot be duplicated, a logical qubit distributes the information of a single qubit across multiple physical qubits, allowing errors in individual qubits to be detected and corrected without having to measure and disturb the original qubit's quantum state.

Measurement — In quantum computing, all computations end with measurement, the process of extracting classical information from qubits. Unlike classical bits, which are always 0 or 1, qubits can exist in a superposition—a state where they simultaneously hold probabilities of being both 0 and 1. Measurement collapses this superposition, forcing the qubit into a definite state of either 0 or 1. Measuring one qubit in a system of multiple qubits affects the entire quantum system. Measurement is critical because it returns outputs as usable classical information, which is essential for solving problems. However, the act of measuring can also disturb qubits and introduce errors, making it an inherently imperfect process. Because measurement outcomes are probabilistic, experiments must often be repeated many times to build up reliable statistics. Measurement is also irreversible: once a qubit is measured, its prior superposition is lost.

GLOSSARY (CONTINUED)

Model-based synthesis — In quantum computing, model-based synthesis is a way to generate circuits from a high-level description of an algorithm and explicit design constraints. Instead of assembling fixed gate templates, a developer specifies the algorithm's intent (such as arithmetic or oracle behavior) alongside resource or hardware targets like qubit count, depth limits, gate set, and backend. A synthesis engine then compiles this model into a gate-level circuit that satisfies the stated constraints, potentially producing different circuit structures from the same model as the constraints change.

Noise — In quantum computing, noise refers to unwanted disturbances that affect qubits and the operations performed on them. It can arise from many sources, including environmental factors such as temperature fluctuations or electromagnetic interference, imperfections in the control systems used to operate quantum gates, and unintended interactions between qubits. Because the quantum information stored in qubits is fragile, noise can disrupt their state and corrupt the information. As noise builds up, the likelihood of an algorithm producing the correct result decreases — and if the noise is too high, the quantum computer becomes unusable.

Neutral atoms — Neutral atoms are atoms with no net electric charge, meaning they have an equal number of protons and electrons. In quantum computing, scientists exploit the internal energy levels of neutral atoms to use them as qubits. They trap individual atoms in a vacuum chamber and cool them with lasers to near absolute zero (-273.15°C or 0 Kelvin), reducing their motion. Other lasers, known as optical tweezers, precisely arrange the atoms into specific configurations and can replace misplaced or missing atoms. Optical tweezers also allow scientists to move neutral-atom qubits during computation without disturbing their quantum states. To carry out computation, lasers manipulate the atoms' energy levels in controlled ways that create interactions between neighboring atoms. Scientists then use imaging techniques to measure the neutral-atom qubits.

Neutral atoms offer advantages such as scalability to large numbers of qubits, the ability to hold their quantum state for relatively long periods, flexible qubit interactions, and high gate fidelity. However, neutral atoms pose technically demanding challenges such as requiring highly precise laser cooling, trapping, and manipulation.

GLOSSARY (CONTINUED)

Physical qubit — A physical qubit is the hardware-based implementation of a qubit in a quantum computer. Physical qubits can be realized in many different forms, depending on the hardware modality, such as trapped ions, neutral atoms, or photons. They are the raw qubits built directly from physical systems in materials in contrast to logical qubits, which are error-corrected, conceptual units built from many physical qubits working together. Their states are manipulated using laser pulses, microwave pulses, or other control signals. Because physical qubits are highly susceptible to environmental noise and prone to errors, they are too fragile to perform long or complex computations on their own, which has resulted in significant research into quantum error correction as well as the development of logical qubits.

Pulse control — In quantum computing, pulse control refers to the use of precisely shaped electromagnetic signals—called pulses—to manipulate qubits. These pulses, often in the microwave or laser range depending on the qubit technology, are carefully timed and tuned to implement quantum gates by changing the state of the qubits. Pulse control gives researchers fine-grained access to the hardware, allowing them to optimize performance, correct for imperfections, and explore new gate designs beyond the standard high-level programming abstractions.

Pulse-level programming — In quantum computing, pulse-level programming is the practice of controlling qubits by directly specifying the pulses that drive their operations. Unlike gate-level programming, where programmers work with abstract quantum gates, pulse-level programming exposes the hardware controls themselves. This approach allows researchers to fine-tune qubit behavior, optimize gate performance, and experiment with customized control schemes, but it also requires detailed knowledge of the underlying hardware.

Pulse sequences — In quantum computing, a pulse sequence is an ordered set of control pulses used to manipulate qubits and carry out quantum operations. Instead of describing computation only in terms of abstract quantum gates, a pulse sequence specifies the timing, frequency, and shape of the pulses that directly drive the hardware. This programmable approach gives developers and researchers fine-grained control over the physical behaviour of qubits and other quantum systems, making it possible to design custom gates, compensate for hardware imperfections, and optimise performance for specific algorithms.

GLOSSARY (CONTINUED)

Quantum advantage — In quantum computing, quantum advantage refers broadly to the point at which a quantum computer can solve a problem more efficiently than the best known classical methods running on the best classical hardware. However, the term has no universal definition, and it is not always used consistently.

Most often, quantum advantage refers to the experimental demonstration of a quantum algorithm solving a real-world problem faster than any classical algorithm could. In this sense, it contrasts with quantum supremacy, which is usually defined as a quantum computer solving any problem — even a contrived one with no practical value — that no classical computer can solve in reasonable time. In other words, supremacy is about demonstrating hardness of simulation, while advantage is about solving useful problems. Unlike supremacy, which can be achieved without error correction, demonstrations of quantum advantage are generally expected to rely on quantum error correction to ensure the results are reliable and commercially meaningful

In some contexts, the term covers theoretical speedups not yet demonstrated in practice. In others, it also covers benefits beyond speed alone, such as enhanced precision or more efficient compression of classical data. For industry and business, the most relevant sense of quantum advantage is pragmatic: when a quantum computer solves a problem in significantly less time, energy, or money than classical methods, or enables solutions that would otherwise be out of reach.

Quantum algorithm — A quantum algorithm is a set of step-by-step instructions designed to run on a quantum computer. Using quantum effects such as superposition and entanglement, they can perform certain computations more efficiently than conventional computers, and in some cases solve problems that are otherwise intractable. Quantum algorithms are expressed as a combination of quantum gates, control flow and classical computation. Designing them requires specialized knowledge of both quantum mechanics and computer science, and implementing them on current quantum hardware is challenging due to the presence of errors and the number of qubits required, which has led to research into error correction, fault tolerance, and hardware optimization. Quantum algorithms are central to the promise of quantum computing, with potential applications in areas such as cryptography, logistics, finance, and medicine. Well-known examples include Shor's algorithm, which can factor large numbers efficiently, and Grover's algorithm, which accelerates search.

GLOSSARY (CONTINUED)

Qubit (quantum bit) — A qubit is the basic unit of information in quantum computing, analogous to the bit in classical computing. While a classical bit can only take the value 0 or 1, a qubit can exist in a superposition, meaning it has some chance of being measured as either 0 or 1. Before measurement, these probabilities are captured in a mathematical description called a wave function. When measured, the wave function collapses and the qubit takes on a definite value of 0 or 1, producing classical information that can be used in computation. Qubits are typically realized using physical systems that exhibit quantum behavior, such as photons, electrons, trapped ions, or superconducting circuits. The space needed to describe the state of qubits grows exponentially as they are connected together.

Quantum mechanics — Quantum mechanics is a fundamental branch of physics that describes the behavior of matter and energy at the atomic and subatomic scale. It explains how particles can exist in multiple states at once (superposition), display both wave-like and particle-like properties, and share stronger correlations than exist in classical physics (entanglement). These rules differ sharply from the laws of classical physics and often feel counterintuitive compared with everyday experience, which makes them challenging to grasp, but it is by exploiting these uniquely quantum effects that quantum computing becomes possible.

Quantum processing unit (QPU) — A quantum processing unit (QPU) is a common industry term for the core of a quantum computer: a system made up of physical qubits and the apparatus used to control them, such as lasers, microwave generators, and other supporting electronics. A QPU is where the qubits reside and where computation takes place. It is often described as the “brain” of a quantum computer. Like the central processing unit (CPU) in a classical computer, a QPU requires significant supporting infrastructure — and the nature of that infrastructure can vary widely depending on the underlying hardware design.

Unlike CPUs, QPUs are not standardized in design. Different hardware modalities each bring their own strengths, weaknesses, and engineering challenges. QPU performance also lacks a single standardized metric. The first point of comparison is usually the number of qubits, but true capability is also shaped by other factors, including how reliably qubits maintain their quantum state over time, their connectivity, gate speeds, and error rates.

GLOSSARY (CONTINUED)

Runtime environment — A runtime environment is the support system a program needs while it is running. It provides the behind-the-scenes services that allow code to execute smoothly, such as managing memory, handling input and output, and giving the program access to hardware resources like storage, networks, or displays.

The runtime environment does not translate high-level code into machine code — that's the role of a compiler, but it provides the setting necessary for that program to run once it has been translated. It supplies common functionality, such as garbage collection, system libraries, and error handling, so that programs don't need to re-implement those tasks themselves. In quantum computing, a quantum runtime environment plays a similar role by providing the infrastructure needed to let quantum programs run across different hardware backends. It often handles tasks such as scheduling, error mitigation, and integration with classical computing resources.

Superposition — In quantum mechanics, all quantum systems can be described as a combination of possible states, known as basis states. For a qubit, the basis states are zero and one. Unlike a classical bit, which can only exist as zero or one at a time, a qubit can exist in a superposition—a blend, or linear combination, of both states at once. In other words, the qubit is described by certain weights attached to zero and to one, which determine how likely the computation is to result in each outcome. When qubits are measured, they lose their superposition and collapse into either zero or one. Because measuring a qubit only gives one outcome at a time, quantum algorithms must be run many times to build up the full probability distribution if that is needed.

Superposition is a central feature of quantum computing and a major source of its potential for solving complex problems more efficiently than classical ones. It allows a quantum computer to process many possibilities at once, creating a kind of built-in parallelism that can yield exponential speedups for certain types of tasks. However, such information can only be accessed indirectly, through quantum interference, making the task of designing quantum algorithms particularly tricky.

Maintaining superposition is difficult: interaction with the surrounding environment can quickly destroy it, leaving the qubit in a simple classical state. This fragility makes error correction an essential step toward building practical quantum computers.

GLOSSARY (CONTINUED)

Superconducting qubits — A superconducting qubit is implemented on specialized chips, typically made of silicon or sapphire, using microfabrication techniques similar to those used in classical processors. Superconducting qubits are tiny electrical circuits made from superconducting materials, which exhibit zero electrical resistance when cooled to millikelvin temperatures. This is drastically different from normal conductors, where electrons flow, collide with atoms and defects, and lose energy, creating electrical resistance

These tiny circuits typically include components called Josephson junctions, which are made by placing a very thin layer of insulating material between two superconducting materials. When electrons tunnel across this barrier at low temperatures, the circuit takes on discrete energy levels, meaning it can only occupy certain fixed-energy states, not a continuous range. With careful circuit design, the two lowest of these states can be isolated and used to encode a qubit's 0 and 1 states. Because they behave like atoms, superconducting qubits are often called artificial atoms.

Superconducting qubits are among the most widely developed hardware modalities because they can be manufactured using existing chip-making methods and execute operations quickly. However, they are difficult to scale due to the short lifetimes of their quantum states, the need for cooling to near absolute zero, and their sensitivity to fabrication errors. These errors, introduced during manufacturing, can affect the electrical circuits forming qubits, reducing their performance or reliability.

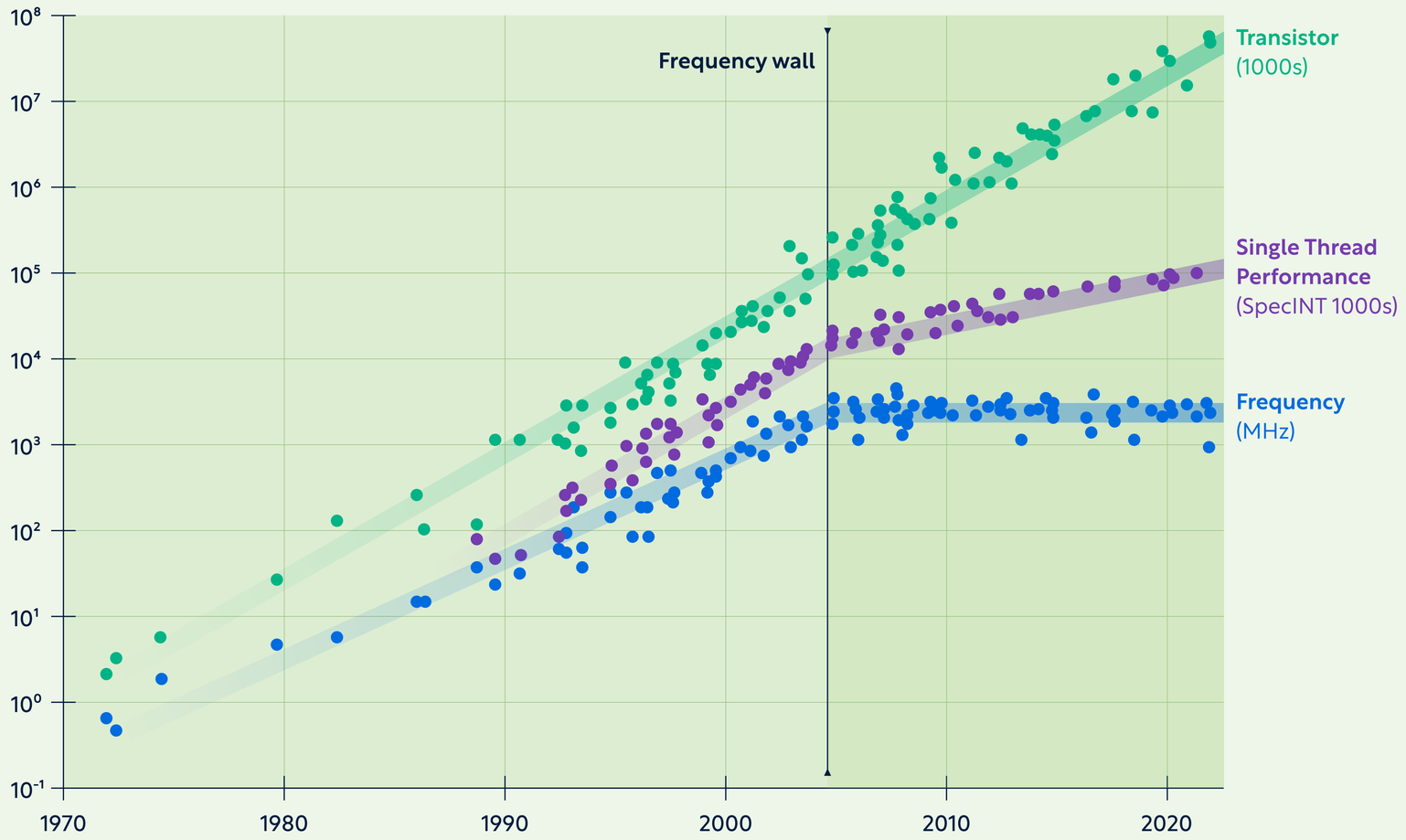
Trapped ions — In quantum computing, scientists use charged atomic particles (ions) as qubits by "trapping" them in a vacuum with an ion trap, which creates electromagnetic fields that confine the ions. Each ion serves as a qubit, with different electronic states corresponding to the 0 and 1 states. Using lasers, scientists cool the ions to near absolute zero to minimize their motion and precisely control their behavior. With lasers or microwave pulses, they manipulate the ions' internal states and interactions to perform quantum operations.

Trapped-ion systems have played a major role in demonstrating fundamental quantum algorithms and advancing quantum computing from theory to experiment. They offer advantages such as long qubit lifetimes (the ability to hold their quantum state for extended periods) and high-fidelity operations. However, scaling to large numbers of ions and integrating the supporting technology into compact, practical systems remain challenging.

GLOSSARY (CONTINUED)

Turing-complete language — A Turing-complete language is a programming language that can be used to express any computation that any computer can perform, given enough time and memory. To be Turing complete, a language must support (or allow the construction of) basic features such as conditional branching (e.g. if/else statements), loops, and recursion. Most general-purpose programming languages, including Python, Java, and C, are Turing-complete. Languages and frameworks that only allow the expression of static circuits are not Turing-complete.

Appendix B: Why Quantum Computing?



Source: Modified from "50 Years of Microprocessor Trend Data", K. Rupp, https://github.com/karlrupp/microprocessor-trend-data

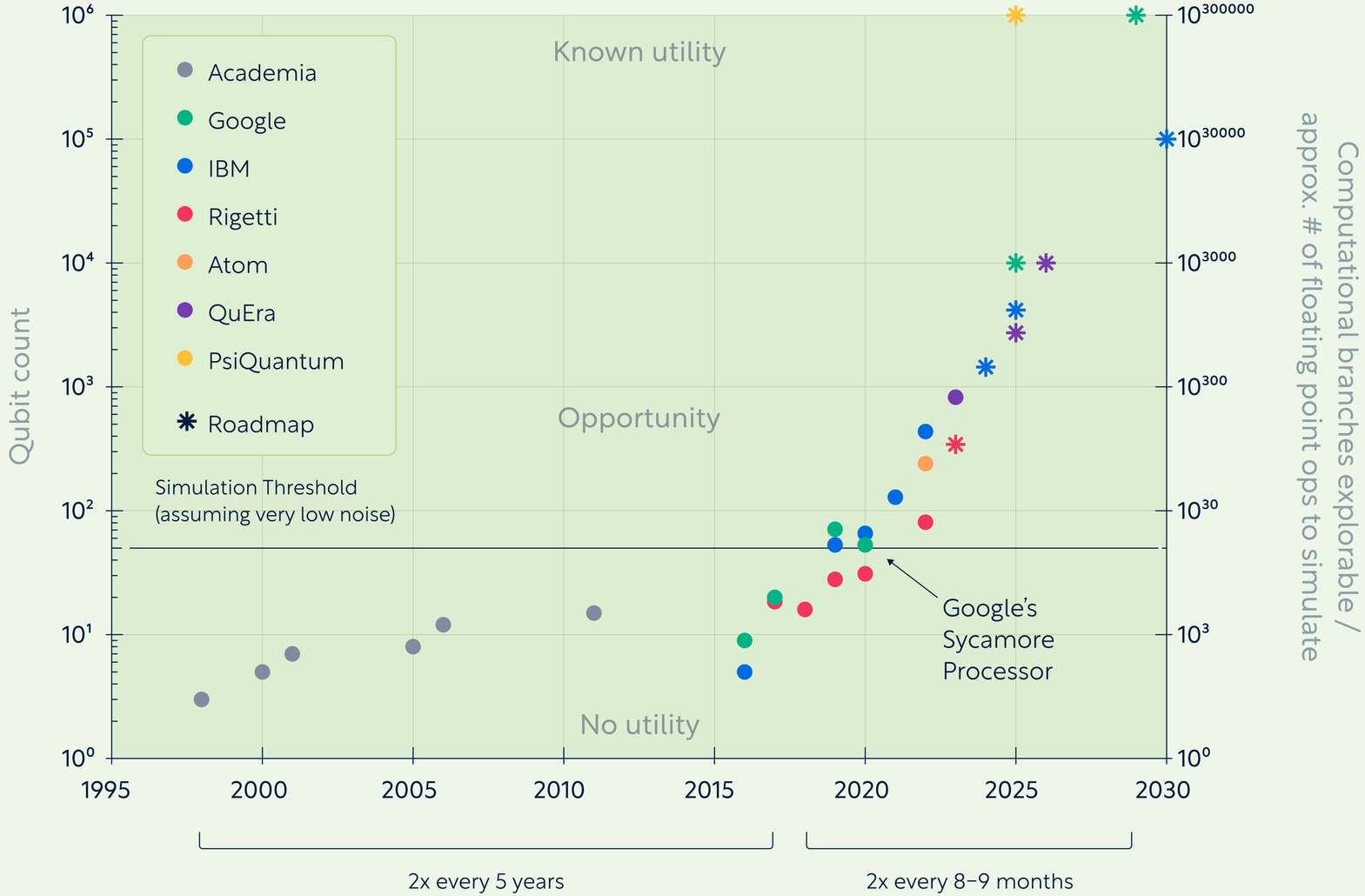
Classical computing is approaching the limits of Moore's Law

The improvement in classical microchips has followed Moore's Law for decades as **the number of transistors on a chip doubling about every 2 years,** resulting in exponential growth.

Although the number of transistors is still increasing, classical computing faces **increasing physical and technological challenges**. The growth of some other performance metrics has already slowed or stopped.

Quantum changes the game, **it's time for Neven's Law**

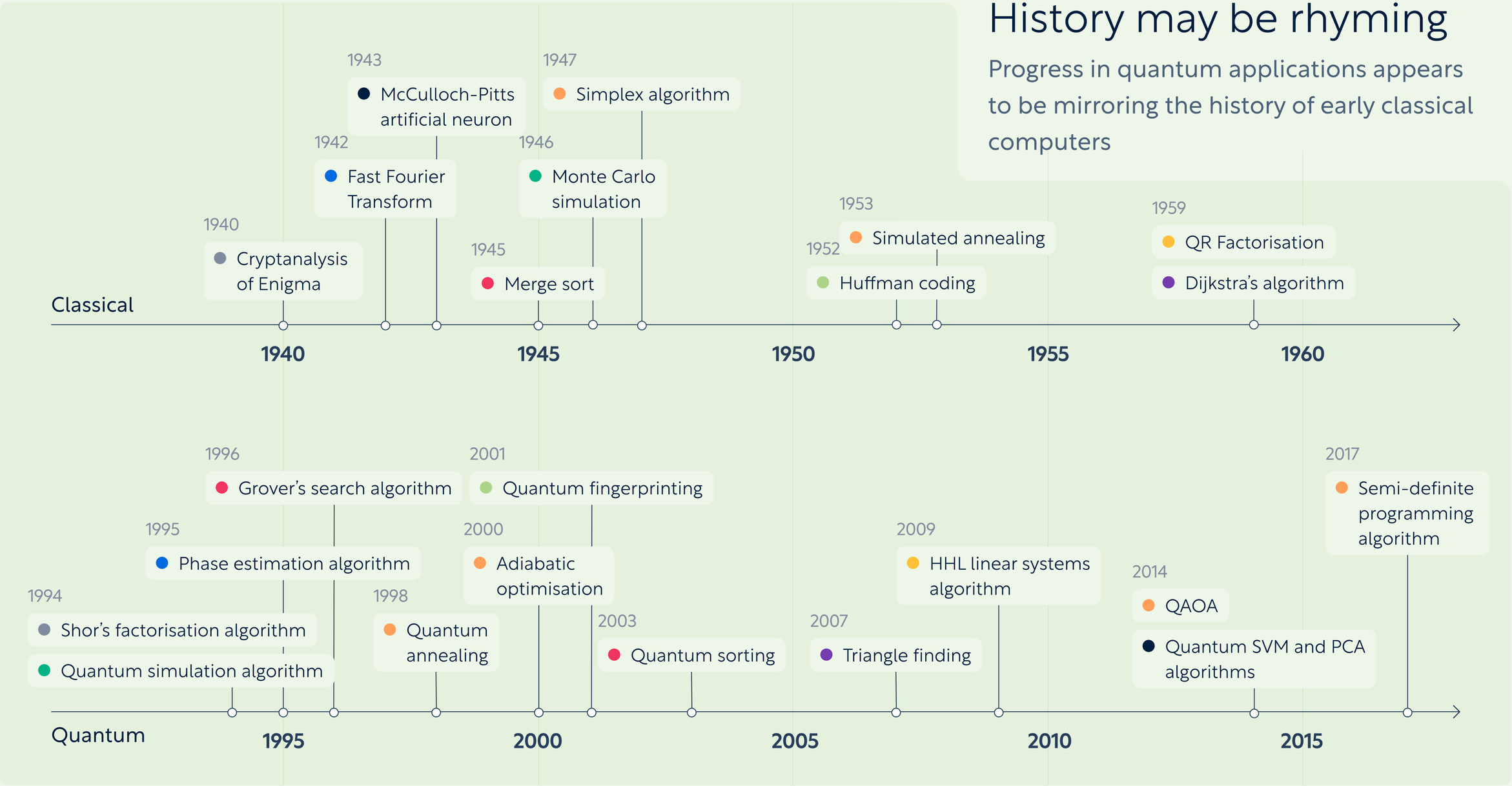
Where Moore's Law predicts exponential growth in the capacity of conventional computers, Neven's Law suggests **double-exponential** growth for quantum computers due to a combination of exponential qubit growth over time and the increase in the difficulty of simulating quantum systems for each incremental qubit.



Source: Neven's Law analysis diagram prepared by Horizon Quantum management for illustrative purposes.

History may be rhyming

Progress in quantum applications appears to be mirroring the history of early classical computers



Algorithm type: Cryptanalysis · Simulation · Fourier methods · Graph algorithms · Sort & search · Optimisation · Compression · Matrix Methods · Machine learning

Source: Timeline prepared by Horizon Quantum management for illustrative purposes.