ASEE 2022 ANNUAL CONFERENCE
Excellence Through Diversity
MINNEAPOLIS, MINNESOTA, JUNE 26TH-29TH, 2022 ASEE
Paper ID #36790

Work-in-Progress: Undergraduate Courses in Quantum Computing: A Proposal based on our Experience Building a Python-based Quantum Computer Simulator

David Hoe (Associate Professor)

Dr. Hoe completed his undergraduate and graduate studies at the University of Toronto in electrical engineering. Prior to his current position as an Associate Professor at Loyola University Maryland, he worked at GE Research in Niskayuna, NY and was an Assistant Professor at the University of Texas at Tyler. His research interests include high performance computing using FPGAs, quantum computing, and engineering education.

Mary Lowe (Loyola University Maryland)

Dave Binkley

Dave Binkley is a Professor of Computer Science at Loyola University Maryland where he has worked since earning his doctorate from the University of Wisconsin-Madison in 1991. He has been a visiting faculty researcher at the National Institute of Standards and Technology (NIST), worked with Grammatech Inc. on CodeSurfer development, was a member of the Crest Centre at Kings' College London, and a Fulbright scholar working with the researchers at Simula Research, Oslo Norway. Dr. Binkley's current research interests include tools and techniques to help software engineers understand and improve their code.

Work-in-Progress: Undergraduate Courses in Quantum Computing: A Proposal based on our Experience Building a Python-based Quantum Computer Simulator

Abstract

There is a growing need for scientists and engineers with Quantum Information Science and Technology (QIST) skills. To address this need for 'quantum aware' graduates we describe a two-course sequence in quantum computing suitable for undergraduate students studying electrical engineering, physics, and computer science. Our approach is unique in that it aims to take full advantage of the diverse backgrounds of second and third year undergraduate students from these three STEM disciplines. Students will work in interdisciplinary teams to (1) understand the physics and the computational theory relevant to quantum computing, (2) develop computer code that simulates a quantum computer, (3) understand the relevance and importance of existing quantum computing algorithms, and (4) appreciate the need for future research in quantum computing. Our approach is informed in part by our experience working with a pair of undergraduate students this past summer on the development of a Python-based quantum computer simulator. This experience showed that building a simulator was an effective way to teach the theory underlying quantum computing. Building a simulator also provides an excellent foundation upon which to explore student-accessible research projects in quantum computing, which is a high-impact educational practice.

1. Introduction

As current semiconductor-based integrated circuits reach the limits of scaling, researchers are turning their attention to novel device technologies and new computing architectures to obtain continued improvements in computing performance [1]. A promising emerging technology is quantum computing, which is theoretically predicted to provide exponential increases in speed over classical computers for certain problems, such as solving linear systems of equations [2] and factoring integers via Shor's algorithm [3]. The potential to factor large integers in polynomial time with quantum computers makes current encryption methods vulnerable – this initially fueled much of the interest in quantum computing. Combined with recent advances in quantum hardware which have produced experimental quantum machines on the order of 50 to 100 qubits [4], there is now widespread interest from academia and industry in quantum computing as a viable technology for the post-Moore's Law era. The National Quantum Initiative Act, signed into law in December 2018, calls for an accelerated commercialization of quantum technologies and highlights the need to train the next generation of scientists and engineers with skills in quantum information science and technology (QIST) [5].

Surveys of the nascent quantum industry reveal a need for scientists and engineers with a broad spectrum of QIST expertise. Hughes *et al*. surveyed 57 companies in the quantum industry to determine their hiring needs for the next two to five years [6]. While the greatest need was for theoretical physicists and quantum algorithm developers, there was also a significant need for engineers knowledgeable in circuit design, test and measurement, and control systems as well as

data scientists and software developers. While many of these positions did not specifically require a background in quantum computing, clearly being knowledgeable of the QIST field would be an asset when working in the quantum industry. Another recent study of 21 US quantum companies found that 95% of them employed engineers, with 71% having a background in electrical engineering [7]. In particular, there is a need for electrical engineers with expertise in control systems and electronics to build the interface circuits to the quantum circuits, especially for measuring quantum states. This requires knowledge of Field Programmable Gate Array (FPGA) technology for processing the data [8], [9] as well as expertise in designing CMOS circuits operating at cryogenic temperatures for implementing low-noise electronics that interface with quantum processors [10], [11], [12]. A review of the anticipated engineering positions needed in the quantum technology market reveals that most of the openings will be for 'quantum-aware engineer[s]' and 'non-quantum engineer[s] employed in a quantum company' [13]. A number of employers noted the best way to increase this work force is to simply augment a classical engineering curriculum with a one or two semester course in quantum computing [7].

Several universities have offered courses in quantum computing at the graduate and undergraduate levels. Due to the growing demand for 'quantum-aware' specialists, there is a need for introductory-level QIST courses. In this paper, we describe a pair of introductory courses in quantum computing for undergraduate students studying electrical engineering, physics, and computer science. By requiring a modest set of prerequisites, the courses will be accessible to a broader range of STEM students. Our approach features an interdisciplinary learning environment, which prepares students for modern work environments where engineers and scientists routinely work with other disciplines to solve complex and multifaceted design and research problems [14], [15]. This is especially true in the QIST industry, where theoretical physicists, electrical engineers, and data scientists routinely interact [6]. Having interdisciplinary teams of students collaborating on in-class exercises, coding assignments, and projects is expected to enhance the learning experience by promoting high-level cognitive skills such as problem-solving and critical-thinking [16]. When this high-level thinking is combined with metacognitive reflection, students are able to develop expert learning skills by being able to use the appropriate reasoning strategies and concepts to solve new problems [17].

Our first offering for these courses is planned for the Spring 2023 semester. We expect to have 10 to 15 students enrolled but will cap enrollment at 20 students for the initial offering. Rather than a separate laboratory section, some lecture time will be allocated for students to work on their code in interdisciplinary teams. These teams will also be assigned joint exercises to be completed outside of class.

The outline of our paper is as follows. The next section provides the context, motivation, and goals for the courses. The following section details the content for the proposed courses in quantum computing. An innovation in the courses, the development of a Python-based quantum computer simulator, is then detailed, followed by a discussion of learning outcomes and selected undergraduate research projects.

2. Background, Motivation, and Goals for the Course

Our university, Loyola University Maryland, is a private liberal arts institution of approximately 3800 students. Loyola focuses primarily on undergraduate studies and features STEM departments in engineering, computer science, physics, chemistry, biology, and mathematics. The proposed courses in quantum computing will be taught by professors in electrical engineering, physics, and computer science. The goals of the courses are as follows:

1. The courses will have prerequisites that can be typically met in the freshman year. This will make our courses accessible to students with more diverse backgrounds earlier in their college careers, which will help with inclusion and retention of students in the STEM fields.
2. The courses will have students write Python code to reinforce their learning of the theory underlying quantum computing.
3. The courses will prepare students to do research in quantum computing as undergraduates by introducing them to and giving them the opportunity to work on authentic research questions in the course. This is widely viewed as a high-impact educational practice because it encourages "inquiry-based, student-centered activities" [18].

Unlike other courses on quantum computing, our students will develop a quantum computer simulator. A survey of textbooks revealed that many books are either too mathematical or not sufficiently mathematical for an introductory, majors-level, college course for STEM students. If the book or course included programming, the approach relied on existing simulators which do not necessarily challenge students to grapple with the underlying theory. We are choosing a more practical approach in our courses by having students learn the theory and concurrently code their own quantum computer simulator from the ground up.

At Loyola University Maryland, undergraduate students have the opportunity to do research during the summer and academic year. During Summer 2021, the authors collaboratively mentored two undergraduate students, one majoring in physics and the other in computer science. Our observations and assessment of this summer research effort served as the genesis for the proposed interdisciplinary courses in quantum computing.

3. Two-semester course sequence on quantum computing

The first course in the sequence is targeted primarily at sophomore and junior level students majoring in physics, computer science, and electrical engineering. Because their preparations vary and we would like undergraduates to take quantum computing courses from different entry points, we limit the prerequisites for the first course to Calculus I and an introductory programming course. We do not assume prior background in college physics, quantum mechanics, linear algebra, or complex numbers, but we do assume a solid knowledge of high school algebra and trigonometry. Calculus I is a prerequisite to ensure a base level of mathematical maturity. Table 1 summarizes the relevant course background for our proposed courses and discusses whether or not they are prerequisites.

Table 1. Prerequisites and related courses for our quantum computing course

Course	Comments
Calculus I	Required: even though calculus is not necessary for understanding quantum computing, having the course ensures a basic level of mathematical maturity
Introductory Programming	Required: all students will be expected to a have basic level of programming competence
Object-Oriented Programming (OOP)	Not required: code templates along with tutorials will allow students to pick up the required skills. CS students will typically have OOP in their third semester.
Linear Algebra	Not required: understanding of vector spaces and matrix mathematics will be developed in the intro course
Complex Mathematics	Not required: the first course will emphasize computations using real numbers, with complex math introduced towards the end and developed further in the second course
Quantum Physics	Not required: an understanding of the physics of, for example, superposition and measurement will be introduced in the course. Quantum computer hardware will be described conceptually.

Our quantum computing courses will integrate paper-and-pencil exercises covering the physics and mathematical background with writing Python code to simulate quantum gates and circuits. Table 2 shows the topics to be covered in the first course. The paper-and-pencil exercises will use Dirac notation (bra-ket) starting from a 1-qubit system to an N-qubit register. Quantum mechanics concepts for understanding quantum computing will be taught along with basic concepts in probability and statistics. To develop a quantum computer simulator, the students will concurrently learn an alternate representation of quantum states by writing them as column vectors and gates as matrices. The necessary linear algebra will be taught in the course. A more detailed description of the simulator appears in Section 4. The combination of basic quantum computing concepts and mathematical formalism is at a level higher than most of the approaches used in high school courses or with a general audience, but at a lower level than current books targeting graduate and upper division undergraduate physics courses.

A complete understanding of quantum computing requires complex numbers, but in the first course the students will start with algorithms that require only real numbers such as binary addition and Grover's search [19]. The aim is for students to quickly gain an understanding of many of the building blocks used in more sophisticated quantum algorithms. Later, complex numbers and algorithms will be introduced and thoroughly covered in the second course (Table 3) so that the key building blocks such as quantum phase estimation and its inverse, the quantum Fourier transform can be understood. These are important, for example, for comprehending Shor's algorithm.

Table 2. Outline for the first quantum computing course

Topic	Quantum computer simulator activities
Introduction: importance of quantum computing. Traditional vs quantum computing, conventional logic gates vs quantum computing gates. Quantum circuit diagram. Hardware using 2-state physical systems.	
1-qubit quantum circuits. Gates include NOT, Hadamard, phase-shift of 0°and 180°. Successive gate operations. Hand calculations to understand basic physics concepts such as superposition and relative phase using real numbers. Introduction to quantum measurements.	2x2 matrices, matrix multiplication, calculating 1-qubit states as they evolve in the circuit. Simulate quantum measurements at the 1-qubit level.
2-qubit quantum circuits. Hand calculations on superposition, entanglement, orthonormal states, tensor products, orthogonal and symmetric matrices. More on measurements.	4x4 matrices. Ex. CNOT, $H^{\otimes 2}$, controlled-phase shift, etc. Write code for tensor products to calculate matrices and vectors; quantum state vector calculations. Measurement simulation
Multiqubit quantum circuits. Gates involving 3 or more qubits. Construct matrices where control and target are on arbitrary lines. Construct other matrices acting on multiqubits. More complex circuits.	8x8 and larger matrices, e.g., Toffoli, controlled-U, custom gates, etc. Construct circuits with partial quantum gate sets, such as addition, random byte generator, Deutsch, etc. Use simulator to assist with hand calculations to understand circuits.
Algorithms that can be handled with real numbers. Theory and coding.	Deutsch algorithm, Simon's algorithm, Grover's database search algorithm, and Deutsch-Jozsa algorithm
Intro to circuits involving complex numbers (to be continued in Course 2)	Phase-shift gate (rotation gate), Hadamard test
Mini-course project	Research topic in quantum computing. Simulation of basic algorithm and/or addition of new features to simulator

Students should develop a conceptual understanding of qubits and how they are implemented in real quantum computing hardware. For both courses, we will incorporate active learning in the classroom. For example, there will be a day when students will work with a real quantum computer available through IBM Quantum [20] and begin to appreciate noise in quantum computations. To understand two-state physical phenomena, students could study examples from quantum optics by using worksheets, graphical constructions, and hands-on activities pertaining to the polarization of light [21], [22].

In the process of creating the first course, the instructors will develop intermediate-level quantum circuits to help scaffold the learning process. There tends to be a large jump in complexity from algorithms such as the two-qubit Deutsch algorithm to the next algorithm in many textbooks. Another need is a consistent design methodology for implementing arbitrary, multiqubit matrices

frequently found in published algorithms for pedagogical purposes (e.g., see Candela's tutorial paper [23]). Such matrices support the high-level presentation of an algorithm, but typically cannot be directly implemented on a physical quantum computer. While we can incorporate such matrices in the simulator, we need to search for methods to decompose them into a sequence of simpler gates. We note in Section 6 that studying this problem would be a good student research project.

After completing the first course, students will have a solid foundation in quantum computing from which they can pursue more advanced topics through research under the guidance of a faculty mentor. By developing their own code to simulate quantum circuits, the students will be well positioned to make extensions that enable them to study meaningful research questions through simulation. This is particularly advantageous for doing research that requires modification of the code to extend its capabilities. A collection of such future research topics is considered in Section 6.

Table 3. Outline for the second quantum computing course

Topic	**Quantum computer simulator activities**
Circuits involving complex numbers. Complex number manipulations, relative phase (general treatment), unitary matrices. Algorithms requiring complex numbers.	Quantum phase estimation, Quantum Fourier transform, Shor's factoring algorithm, and Solving system of linear equations
Is there a finite universal gate set? Is there a set of gates from which it is possible to construct any quantum circuit? Learn how to write arbitrary multiqubit, unitary matrices.	Build a more complete gate set based on a survey of quantum gates used in the literature (see Section 6 as a possible research topic).
Physical quantum computers based on 2-state systems: Qualitative introduction to photon polarization, ion trapping, superconducting systems, etc.	Lab day: Work with a real quantum computer provided by IBM Quantum. Understand concepts in noise and decoherence.
Additional quantum computing topics Reversibility, no-cloning theorem, ancilla bits, oracle, Bloch sphere, quantum error correction, quantum communication, quantum cryptography, machine learning	Implement selected topics on simulator
Course projects	Research topic in quantum computing. Simulation of algorithm and/or addition of new features to simulator

4. Development of a Python-based quantum computer simulator

4.1 The Integration of the Simulator into the Course

While a variety of open-source simulators exist for simulating quantum circuits, such as Cirq developed by Google and Qiskit by IBM [24], [25], our expectation is that students will better internalize the details of quantum computing by writing their own simulator from the ground up.

There are several reasons. First, through their work going into the design and development of the code, students will gain a deeper understanding of the differences between traditional computing and quantum computing. As the renowned computer science professor Donald Knuth stated: "The truth is you don't understand something until you've taught it to a computer, until you've been able to program it" [26]. Second, having a simulator facilitates the study of complex circuits. As the size of the matrices grows as $2^n \times 2^n$, for n qubits, analyzing a system with more than three qubits becomes challenging when done purely by hand. Third, writing the code for one's own simulator enables the students to tinker with it more easily, which also serves to improve their understanding. For example, they can organize the code to be optimal for each algorithm or add functions to improve their understanding such as inspecting the quantum state at arbitrary internal points in the circuits or selecting how to display the outputs.

Tables 2 and 3 give examples of how the development of the simulator will augment the theory and the paper-and-pencil exercises. Students will use the Python library NumPy, which provides the necessary matrix representations and operations in a form accessible to beginning programmers. Python *classes* will be used to structure the code requiring teaching some elements of object-oriented programming (OOP). While not all students will have been exposed to OOP prior to the first quantum computing course, the faculty already have developed a framework for a simulator and will guide the students to write the code for selected methods. Faculty will also provide a basic introduction to the OOP features of the Python programming language. The students will work in interdisciplinary teams, where the CS students will be helpful in guiding their peers in this area. Thus, previous knowledge of OOP will not be necessary. Other programming possibilities that will enhance student learning include writing their own tensor product function for combining vector spaces instead of using the NumPy function, using a random number generator to simulate the outcomes of quantum measurements, and displaying the measured results in a histogram. Finally, the students will use their simulator to explore the interaction between a quantum computer and a conventional (classical) computer.

4.2 Lessons Learned from the Simulator Developed by Students

As a research project in Summer 2021, two students worked on developing a quantum computer simulator. One student was a physics major who had just finished his freshman year. He had taken two programming courses. The other student was a computer science major who had just finished his sophomore year with four programming courses completed. The students selected for our summer research program are usually exceptional students and so they would not be representative of the typical student in our proposed courses. Nevertheless, it is noteworthy that without any prior knowledge of quantum computing, but through reading the relevant literature and working through tutorials developed by their mentors, the students obtained sufficient understanding of quantum computing to create the simulator from scratch. Also, it should be noted that in our proposed courses, the students will have code templates as guides for developing their simulator. The instructors gained insight on the best ways to integrate coding activities into the proposed courses and where students need more assistance.

1. *An object-oriented approach is a viable and the preferred method for organizing the code*. The learning curve of the summer students was instructive in designing the goals of the course simulator. For example, the student's initial approach was to use string encoding

rather than Python classes and objects. While this naive approach enabled them to get up-and-running quickly, it would have become an increasing burden as the code base grew. Fortunately, the suggestion that they switch to the use of gate *objects* was sufficient for them to produce a much more modular, object-based simulator. Thus, we expect it to be feasible for the course simulator to be object-based from the beginning, assuming we provide some initial examples.

2. *Multiqubit gates, especially when applied to non-adjacent qubits, are challenging.*
   In the initial stages of learning about a 2-qubit gate such as CNOT, the gate acts on adjacent qubits. To handle most algorithms, there has to be a method to work with gates where the qubits are not adjacent. The students were able to develop code by themselves to 'swap' qubits until they were adjacent, but the final code lacked design thought. The instructors realized that course time needs to be spent on how to handle multiqubit gates.
3. *In the simulator, one must think about how to best specify the sequence in which gates are executed*.
   In quantum circuit diagrams, time flows from left to right. In the student simulator, each new gate was placed in the leftmost available position of the circuit diagram. This resulted in code that was difficult to read and impossible to edit. The course simulator will support gates placed at user-specified moments in time. Tests with the newly designed simulator show that it is flexible and easy to edit.

5. Learning Outcomes

Table 4 summarizes the learning objectives for our proposed courses. The objectives are informed by our work with students and the literature [27], [28], [29].

Table 4. Learning Outcomes of the Courses

From taking these courses, the student will be able to
1. describe the basic concepts in quantum mechanics such as wavefunction (state), superposition, entanglement, and measurement
2. perform the fundamental mathematical operations associated with quantum gates and qubits
3. write computer code to implement the matrix operations associated with quantum gates
4. articulate the advantages of quantum computing over traditional computing
5. write code to make measurements, display them, and interpret the results
6. simulate a quantum circuit of 'moderate complexity' on their quantum computer simulator
7. describe the theory and implementation of the Quantum Fourier Transform and Shor's algorithm
8. design simple quantum circuits
9. describe the limitations of near-term quantum computing systems and effects of noise
10. list some important areas where quantum computing can make a contribution
11. list some areas of future research in quantum computing
12. describe qualitatively several examples of quantum computing hardware and their physical principles

6. Ideas for Student Research Projects

Getting students to the point where they have enough background to do research with faculty in the area of quantum computing is an important goal of these courses. Giving students the experience of working on an open-ended research problem, even though it may be somewhat limited within the context and timeframe of the course, will be an important motivating activity where students gain "the sense of excitement that comes from working to answer important questions" [30]. There are several benefits of involving undergraduate students in research including improvements in student learning and cognition, practical experience in developing into a researcher, and retention in the STEM fields [18], [31]. Having students work on a research project in quantum computing in the last part of each course will give them a good introductory experience into what scientists and engineers do in terms of current and future problems and the interdisciplinary nature of the teams [32].

Suitable mini-student research projects for the first course involve two types: first, to design and simulate simplified versions of key algorithms to improve student understanding, and second, to extend the features of the simulator to make it more efficient. A good example of the first would be implementing a 3-qubit version of Grover's search algorithm. In a simplified scenario, the required oracle is given as an 8x8 matrix where the answer is known beforehand [23]. The students would be challenged to implement a more realistic scenario where a quantum database is designed to be searched [33]. Part of their assignment would be to read the scholarly literature, such as Grover's original paper [19] and other articles that use amplitude amplification, and to be able to explain the basic principles. For the second type of project, students might look at implementing features to allow larger qubit circuits to be simulated. One approach is to implement functions to handle sparse matrices which often appear in larger qubit circuits – for an example see the work of Candela [23]. Another approach involves studying quantum circuits with relatively 'low' entanglement to see how they can be simplified [34].

For the second course, students will work on projects that introduce them to problems and issues in quantum computing that could be studied later in more detail as a research project with a faculty mentor. One example would be studying the computational complexity of quantum algorithms and understanding the conditions and assumptions under which quantum speedup can be expected. For example, by comparing the quantum Fourier transform (QFT) with its classical counterpart through simulation and analysis, there is a risk of overestimating the quantum speedup if specific assumptions are not valid [35]. Another type of project would involve studying recent papers that propose quantum computing solutions to the most challenging optimization problems posited by classical complexity theory [36]. A good example is a proposed solution to the traveling salesman problem that encodes the distances between cities as phases. By using quantum phase estimation (QPE), a quadratic speedup over classical approaches is theorized [37]. Demonstrating that these NP-hard optimization problems can be solved more efficiently on a quantum computer is important because it would show the supremacy of quantum solutions over classical approaches. Another project would involve adding a noise model to the simulator and evaluate key algorithms such as QFT and Grover. Adding this capability would open up a variety of interesting research that involves evaluating how well proposed algorithms function on noisy intermediate-scale quantum (NISQ) machines. Finally, students can survey the literature on two related issues: how to design gates given the

constraints posed by actual quantum computer hardware, and how to decompose a large unitary matrix, as specified by an algorithm, into a sequence of simpler quantum gates.

7. Summary and Conclusions

This paper the described a proposal for a pair of introductory courses in quantum computing that will satisfy the growing need in the quantum industry for 'quantum aware' graduates with a bachelor's degree in engineering, physics, or computer science. As advanced mathematical prerequisites will not be required, these courses will be accessible to a more diverse population of students earlier in their college careers. The excitement that comes from learning advanced concepts early in their academic careers combined with the positive experience of working in interdisciplinary teams will not only help with the retention of students in the STEM fields, but also help motivate them to pursue potential graduate studies. The successful development of a Python-based quantum computer simulator from 'scratch' by two of our research students over the course of one summer gives us a high degree of confidence that this code writing activity can be successfully translated into the classroom. The assessment of student learning of quantum computing concepts during this process supports our hypothesis that developing such a simulator will be an effective method for enhancing the learning of fundamental quantum computing concepts. Understanding how to modify and extend the code of the simulator makes possible the study of some important open questions in quantum computing at the undergraduate level.

References

[1] T. N. Theis and H.-S. P. Wong, "The End of Moore's Law: A New Beginning for Information Technology," *Computing in Science & Engineering*, vol. 19, no. 2, pp. 41-50, 2017.

[2] A. W Harrow, A. Hassidim, and S. Lloyd, "Quantum algorithm for linear systems of equations," *Physical Review Letters*, vol. 103, no. 15, p. 150502, 2009.

[3] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," *Proceedings of the IEEE 35th Annual Symposium on Foundations of Computer Science*, Nov. 1994, pp. 124–134.

[4] F. Arute, *et al*., "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 505–510, 2019.

[5] M. G. Raymer and C. Monroe, "The US National Quantum Initiative," *Quantum Science and Technology*, vol. 4, no. 2, p. 020504, 2019, [Online]. Available: https://iopscience.iop.org/article/10.1088/2058-9565/ab0441/meta [Accessed Feb. 5, 2022]

[6] C. Hughes, *et al*., "Assessing the Needs of the Quantum Industry," *arXiv preprint arXiv:2109.03601*. [Online]. Available: https://arxiv.org/abs/2109.03601 [Accessed Feb. 5, 2022].

[7] M. F. Fox, B. M. Zwickl, and H. J. Lewandowski, "Preparing for the quantum revolution: What is the role of higher education?" *Physical Review Physics Education Research*, vol. 16, no. 2, p. 020131, 2020.

[8] N. Ofek, *et al*., "Extending the lifetime of a quantum bit with error correction in superconducting circuits," *Nature*, vol. 536, no. 7617, pp. 441-445, 2016.

[9] C. Lamb, *et al*., "An FPGA-based instrumentation platform for use at deep cryogenic temperatures," *Review of Scientific Instruments*, vol. 87, no. 1, p. 014701, 2016.

[10] J. C. Bardin, *et al*., "Design and characterization of a 28-nm bulk-CMOS cryogenic quantum controller dissipating less than 2 mW at 3 K," *IEEE Journal of Solid-State Circuits*, vol. 54, no. 11, pp. 3043-3060, Nov. 2019.

[11] S. J. Pauka *et al*., "A cryogenic CMOS chip for generating control signals for multiple qubits," *Nature Electronics*, vol. 4, no. 1, pp. 64–70, Jan. 2021.

[12] B. Patra *et al*., "Cryo-CMOS circuits and systems for quantum computing applications," *IEEE Journal of Solid-State Circuits*, vol. 53, no. 1, pp. 309-321, Jan. 2018.

[13] A. Asfaw, *et al*., "Building a quantum engineering undergraduate program," *arXiv preprint arXiv:2108.01311*, [Online]. Available: https://arxiv.org/abs/2108.01311 [Accessed Feb. 5, 2022].

[14] L. Ivanitskaya, *et al*., "Interdisciplinary learning: Process and outcomes," *Innovative Higher Education*, vol. 27, no. 2, pp. 95 – 111, Winter 2002.

[15] V. B. Mansilla, and E. D. Duraising, "Targeted assessment of students' interdisciplinary work: An empirically grounded framework proposed," *The Journal of Higher Education*, vol. 78, no. 2, pp. 215-237, 2007.

[16] H. S. You, "Why teach science with an interdisciplinary approach: history, trends, and conceptual frameworks," *Journal of Education and Learning*, vol. *6, no.* 4, pp. 66-77, 2017.

[17] J. S. Gouvea, V. Sawtelle, B. D. Geller, and C. Turpen, "A framework for analyzing interdisciplinary tasks: Implications for student learning and curricular design," *CBE—Life Sciences Education*, vol. 12, no. 2, pp. 187-205, 2013.

[18] D. Lopatto, "Undergraduate research as a high-impact student experience," *Peer review*, vol. 12, no. 2, pp. 27-31, Spring 2010.

[19] L. K. Grover, "A fast quantum mechanical algorithm for database search," *Proceedings 28th Annual Symposium on the Theory of Computing (STOC)*, July 1996, pp. 212-219.

[20] "IBM Quantum." https://quantum-computing.ibm.com/ [Accessed Feb. 5, 2022].

[21] M. Raymer, "Teaching Quantum Information Science to Undergraduate Science and Nonscience Students," invited talk, *2021 AAPT Virtual Winter Meeting*.

[22] M. Raymer, *Quantum Physics: What Everyone needs to Know*, New York: Oxford University Press, 2017.

[23] D. Candela, "Undergraduate computational physics projects on quantum computing," *American Journal of Physics*, vol. 83, no. 8, pp. 688-702, 2015.

[24] R. Wille, R. Van Meter and Y. Naveh, "IBM's Qiskit tool chain: Working with and developing for real quantum computers," *IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE*), March 2019, pp. 1234-1240.

[25] M. Fingerhuth, T. Babej, and P. Wittek, "Open source software in quantum computing," *PLoS ONE*, vol. 13, no. 12, p. e0208561, 2018.

[26] "BBVA Foundation Frontiers of Knowledge Award in the Information and Communication Technologies." https://www.frontiersofknowledgeawards-fbbva.es/galardonados/donald-e-knuth-2/ [Accessed Feb. 5, 2022].

[27] S. E. Economou, T. Rudolph, and E. Barnes, "Teaching quantum information science to high-school and early undergraduate students," *arXiv preprint arXiv:2005.07874,* [Online]. Available: https://arxiv.org/abs/2005.07874 [Accessed Feb. 5, 2022].

[28] "QIS Key Concepts for Early Learners: K-12 Framework." q12education.org. https://q12education.org/learning-materials/framework [Accessed Feb. 5, 2022].

[29] J. K. Perron, et al., "Quantum undergraduate education and scientific training," *arXiv preprint arXiv:2109.13850*. [Online]. Available: https://arxiv.org/abs/2109.13850 [Accessed Feb. 5, 2022].

[30] G. D. Kuh, "High-Impact Educational Practices," in *What They Are, Who Has Access to Them, and Why They Matter*, Washington, DC: Association of American Colleges and Universities, 2008.

[31] K. M. Kortz and K. J. van der Hoeven Kraft, "Geoscience education research project: Student benefits and effective design of a course-based undergraduate research experience," *Journal of Geoscience Education*, vol. 64, no. 1, pp. 24-36, 2016.

[32] S. N. Davis, D. Mahatmya, P. W. Garner, and R. M. Jones, “Mentoring undergraduate scholars: A pathway to interdisciplinary research?,” *Mentoring & Tutoring: Partnership in Learning*,” vol. 23, no. 5, pp. 427-440, 2015.

[33] B. Kain, “Searching a quantum database with Grover’s search algorithm,” *American Journal of Physics*, vol. 89, no. 618, pp. 681-626, 2021.

[34] G. Vidal, “Efficient classical simulation of slightly entangled quantum computations,” *Physical review letters*, vol. 91, no. 14, p. 147902, 2003.

[35] D. R. Musk, “A comparison of quantum and traditional Fourier transform computations,” *Computing in Science & Engineering*, vol. 22, no. 6, pp. 103-110, 2020.

[36] J. Woeginger, “Exact algorithms for NP-hard problems: A survey,” in *Combinatorial optimization eureka*, *you shrink!*, Berlin: Springer, 2003, pp. 185-207.

[37] K. Srinivasan, S. Satyajit, B. K. Behera, and P. K. Panigrahi, “Efficient quantum algorithm for solving travelling salesman problem: An IBM quantum experience,” *arXiv preprint arXiv:1805.10928*. [Online]. Available: https://arxiv.org/abs/1805.10928 [Accessed Feb. 5, 2022].