Quantum Simulators Catalog

Overview of important quantum simulators

Table of contents

OVERVIEW 2

Analog Quantum Simulators 2
Quantum Computer Emulator 2
Quantum Circuit Simulators 3
Quantum Network Simulators 3
QKD Simulators 3

CHAPTER A: SIMULATION WITH PLATFORMS 4

Amazon Braket 4
Azure Quantum (Q#/QKD) 5
Google Quantum AI (Cirq) 6
IBM Quantum (Qiskit) 7
QUANTASTICA (QPS /Qubit Toaster) 8
Quantum Programming Studio 8

Quantum Inspire 10
Quantum Network Explorer (NetSquid) 11
Rigetti QCS (pyQuil) 12

CHAPTER B: SELF-INSTALLATION SIMULATORS 13

Quantum Circuit Simulators 13
blueqat 13
Cirq 13
myQLM 14
Ocean 14
ProjectQ 15
pyQuil 15
Microsoft QKD (Q#) 16
QCircuits 16
Qiskit 17
Quantum Computing Playground 17
Quantum JavaScript (Q.js) 18
Qubit Toaster 19
Quest 19
Qibo 20
QuTip 20
Quirk 20
Silq 21
Strawberry Fields 22
XACC 22

Quantum Network Simulators 23
Interlin-q 23
NetSquid 23
OpenQL 24
QuISP 24
QuNetSim 25
SeQUeNCe 25
SimulaQron 26
SQUANCH 26

QKD Simulators 27
QKDNetSim 27
QKDSimulator 27

Quantum annealing 28
D-Wave Ocean 28

OTHER SIMULATORS 29

Overview

The term quantum simulator is ambiguous and is used for analog quantum simulators that work on the basis of quantum phenomena, as well as for software for simulating quantum computers or effects. Therefore, it is first necessary to present the correct technical terms and their meaning.

Analog Quantum Simulators

Analog quantum simulators are systems to simulate the behavior of a quantum system and its quantum effects by another, more controllable system.

Analog quantum simulators are used, among other things, to analyze the behavior of molecules, which can lead to faster drug development. In materials research, the potential of quantum simulators lies in the research of new catalysts, e.g. for the Haber-Bosch process for the production of ammonia: With better catalysts, the energy costs for this process could be significantly reduced.

Analog simulators have the advantage that they are subject to the same physical systems and laws as the target system to be analyzed. Therefore, they work more effectively compared to (digital) supercomputers and can solve problems faster.

A special feature are the so-called quantum annealers from the company D-Wave from Canada. These exploit quantum mechanical effects to obtain solutions for optimization tasks that would require too much time for a classical computer. The advantage of quantum annealers, in contrast to analog quantum simulators, is the free adjustability. However, these cannot be used universally, but are only suitable for optimization tasks.

An example of the development of an analogue quantum simulator is the goal of the PASQUanS project within the framework of the EU Quantum Flagship project. As part of PASQUanS, an analog quantum simulator with 1000 atoms or ions is to be created, which is to be universally programmable.

Quantum Computer Emulator

Since there is still no quantum computer that can be universally programmed for any type of problem and the use of such computers to solve practical tasks is only in its infancy, so-called quantum computer emulators are used. These simulate the behavior of a quantum computer and the associated QuBits on a classical computer and allow software solutions and algorithms for (future) quantum computers to be tested and developed now. In the following, the terms quantum simulator or simulation refer exclusively to quantum computer emulators; pure analog simulators are not considered further in this document.

Quantum simulators are used, for example, to study the behavior and influence of quantum technologies on areas such as communication networks, cyber security, and computing power. Since the integration of applications and concepts based on quantum hardware into conventional technologies is still in its infancy, quantum simulators already offer the possibility of analyzing predictions and effects of this future technology.

There are now a large number of providers and software for quantum simulators. Above all, large corporations (IBM, Google, Amazon, Microsoft), which themselves produce and further develop existing quantum computers, offer quantum simulators with comprehensive libraries and graphical tools – mostly free of charge. After prior free registration, for example, programs can be run on an IBM quantum computer with the Qiskit simulator developed by IBM.

In general, there are some special features when working with QuBits (not simulated): These include, for example, that quantum computers are susceptible to errors resulting from thermal noise, loss of photons, etc. The consequence is that the states of QuBits are manipulated and consequently errors occur in the calculations. Although there are also software packages for simulating quantum errors, simulation software can only reproduce the real hardware system to a limited extent due to external influences.

Depending on the area of application, quantum simulators can be divided into different categories:

Quantum Circuit Simulators

These types of simulators are suitable for simulating fundamental properties of quantum technologies such as single qubits, entanglement or quantum teleportation.

Quantum Network Simulators

These simulators are intended to simulate networks that are (should) be operated with quantum hardware components in a simplified way and are also able to model partly the transfer of quantum states up to the physical layer.

QKD Simulators

QKD (Quantum Key Distribution) simulators are used at the application layer to simulate key generation, transmission and exchange. Since a QKD key is a random number, this simulation method is also suitable as a random number generator.

Chapter A: Simulation with Platforms

This chapter describes online platforms that make it possible to create simulations online and, if necessary, upload them from a private PC. In addition, many platform operators offer access to real quantum hardware, e.g. quantum processors and "classical" simulators for several QuBits, after registration. Depending on the provider, there are also costs for using this service.

Amazon Braket

Platform	Amazon Braket
Properties	• Amazon Braket is a platform from AWS (Amazon Web Services) that enables working with various types of quantum computers and circuit simulators • Creating Quantum Projects in a Cloud • Can be run on real quantum hardware • Provides advice and support to users and businesses through the Quantum Solutions Lab • Enables the use of quantum computers or processors from D:Wave, IonQ, Rigetti, OQC and soon also from QuEra • AWS offers a Cloud Credit program for scientists
Applications	• To develop and build quantum algorithms • Testing algorithms with quantum simulators • Linking to other AWS products • AWS Braket Services are also used by companies to develop quantum technologies or products for customers (see S. here)
Language	• The Braket SDK (Software Development Kit) is a library written in Python • Braket SDK is OpenSource
License	• Amazon Braket SDK: Apache 2.0 License
Access	• Installation and download of the Braket SDK is free of charge • Use of simulators and quantum hardware usually subject to a fee
OS	• Platform-independent: Python interpreter required
Repository	• GitHub repository of the Amazon-Braket SDK
Documents	• Examples of Braket

Azure Quantum (Q#/QKD)

Platform	Azure Quantum
Properties	• Platform of the Microsoft cloud service Azure, which allows users to create quantum software • Provides access to computers based on ion traps from the manufacturers IonQ and Honeywell • Provides optimization algorithms for quantum annealing (without hardware)
Applications	• Simulation of quantum circuits, optimization tasks, simulation of a complete quantum computer with the Fullstack Simulator (QKD)
Language	• Q# • Python
License	
Access	• Azure account can be created for free here • Access to quantum hardware or booking of computing resources for a fee
OS	• Azure packages can be installed platform-independent
Repository	• Libraries of Q#
Documentation	• Q Tutorials and Documentation#

Google Quantum AI (Cirq)

Platform	Google Quantum AI
Properties	• Google Quantum AI (GQA) includes a collection of tools for developing quantum algorithms • Quantum algorithms for simulators and quantum hardware are written using the Python library Cirq • It is possible to test programs written in Cirq via the Quantum Computing Service on real quantum hardware. • GQA also contains numerous links to current research areas and publications on the topic of quantum technology and educational offers • In addition to Cirq, GQA also offers libraries for simulating fermionic systems (OpenFermion) and another for hybrid (classical/quantum) machine learning (TensorFlow Quantum)
Applications	• Creation and simulation of quantum algorithms or quantum circuits
Language	• Python
License	• Cirq: Apache 2.0 License
Access	• Download Cirq for free • The Quantum Computing Service is not currently available to the public • Other libraries such as OpenFermion and TensorFlow Quantum are also available for free
OS	• Cirq can be used platform-independently, as long as Python is supported
Repository	• Cirq on GitHub
Documentation	• Cirq Documentation and Tutorials • Other software and packages from GQA: see here

IBM Quantum (Qiskit)

Platform	IBM Quantum
Properties	• IBM Quantum is an IBM platform for quantum technologies • The Quantum Composer can be used to create and test quantum circuits online • The IBM Quantum Lab service allows users to create and test programs created with Qiskit online (without installation) in Jupyter Notebook • The website also contains an overview of quantum simulators and (hybrid) hardware systems operated by IBM • Also includes programs to support scholars and faculty in the field • Provides access to real quantum hardware on which quantum circuits can be tested, for example
Applications	• Online Modeling and Simulation of Circuits • Execution of the programs on real quantum hardware developed by IBM • Simulation of (existing) quantum hardware
Language	• Python Library
License	• Apache 2.0 License
Access	• Circuits can be tested on real hardware after registration • No installation required for online use, but registration is required • Registration is free
OS	• Platform-independent: only a browser is required to access the platform
Repository	• Overview of Qiskit Repositories
Documentation	• Page of Qiskit • IBM Quantum Project Documents

QUANTASTICA (QPS /Qubit Toaster)

Platform	QUANTASTICA
Properties	• Organization that develops and deploys various software tools and solutions in the field of quantum computing • Since May 2019, QUANTASTICA has been a partner of Rigetti in the field of application development • Good networking with other partners: Unitary Zero Space, icebreaker and Unitary Fund • Simulators developed by QUANTASTICA include the Qubit Toaster and quantum circuit • The Quantum Programming Studio can be used to carry out graphic simulations online and create quantum algorithms • The QConverter - as an online or commandline version - is a tool that can be used to convert quantum programming languages for specific quantum computers into others, e.g. QASM according to PyQuil • Quantum Algorithm Generator: This tool is used to create quantum circuits based on state vectors (reverse engineering). Installation instructions can be found here.
Applications	• Depending on the software package used, different tasks such as code conversion, creation of simulations or algorithms or quantum circuits can be derived from state vectors
Language	• Quantum Circuit Simulator, QConvert: Javascript • Quantum Algorithm Generator: Python • Qubit toaster: n/a
License	• Quantum Algorithm Generator: Apache 2.0 License • QConverter, Quantum Circuit: MIT License • Qubit toaster: n/a
Access	• No restrictions except Quantum Algorithm: Login to Quantum Programming Studio required
OS	• Platform-independent: Python interpreter or browser (Javascript) required
Repository	QConverter: GitHub Quantum Algorithm Generator, Qubit Toaster: not available Quantum Circuit: GitHub
Documentation	Qubit toaster: s. here Quantum Circuit: s. here Quantum Algorithm Generator: see here QConverter: s. here

Quantum Programming Studio

Simulator	Quantum Programming Studio

Properties	• The Quantum Programming Studio (QPS) is a web-based graphical development environment for quantum algorithms and simulations • Circuits can be easily created via drag and drop • QPS is a partner application from Rigetti • Quantum circuits can be exported as a format for different languages or frameworks and run on different simulators and quantum computers such as TensorFlow Quantum or Amazon Braket • QPS is a package extension of the opensource Quantum Circuit Simulator
Applications	• Running simulations, creating algorithms for different platforms
Language	• Quantum Circuit Simulator: JavaScript
License	• MIT License
Access	• A free registration is required to use the web interface
OS	• OS independent: All you need is a browser with Javascript
Respository	• Available on GitHub
Documentation	• Documentation on different topics of QPS

Quantum Inspire

Platform	Quantum Inspire
Properties	• Quantum Inspire (QI) is a multi-hardware quantum technology platform developed by QuTech • Other partners of the project are listed here • Provides the ability to test your own quantum algorithms on simulators or real-world quantum hardware • Algorithms must be written in the cQASM language, which can also be displayed graphically (e.g. in the form of quantum circuits). • QX is a quantum computer emulator on which up to 34 QuBits can be simulated, depending on the account status (emulator backends) • Two quantum processors are available as hardware backends: the Spin-2 with 2 qubits and the Starmon-5 with five qubits
Applications	• Testing of (quantum) algorithms on real quantum hardware or quantum computer emulators
Language	• cQASM
License	• User guidelines and terms of use can be found here.
Access	• There are three types of user accounts: • **Anonymous:** no registration required, use of the cQASM Online Editor and simulation of up to five QuBits on simulator backend possible. No saving of projects possible • **Basic Account**: Registration required. Project storage possible. Access to quantum hardware or processors and use of different emulator backends possible • **Advanced/Special Account**: for special requests such as registering a group of people or expanding the previous account, e.g. for access to more QuBits. This can be requested via the account management in the user account
OS	• Platform-independent: browser required
Repository	• no repository available
Documentation	• API reference, quick start guide, cQASM manual and code examples are available here

Quantum Network Explorer (NetSquid)

Simulator	Quantum Network Explorer
Properties	• Enables interactive testing of applications in quantum networks such as QKD or distributed CNOT operations • Developed by employees of the Dutch research project QuTech on quantum technology • Also includes documentation on various topics related to quantum networks such as QKD, security, quantum cloud etc... • For the development of your own applications for the Quantum Network Explorer, the QNE-ADK software can be downloaded
Applications	• Suitable for getting a quick overview of the topic of quantum networks and to carry out your own small experiments, e.g. QKD transmission from Amsterdam to Rotterdam
Language	Written in Python - CLI Interface
License	MIT License
Access	• Login required if attempts with the QNE are to be saved online • Registration for the NetSquid simulator is required if QNE is to be installed locally. The reason for this is that QNE uses libraries of this simulator
OS	• Browser required for online access • MacOS or Linux with Python 3.7 and pip version 19
Respository	• QNE-ADK in GitHub
Documentation	• Information on quantum networks, quantum technology and user guides on QNE and QNE-ADK can be found here

Rigetti QCS (pyQuil)

Platform	Rigetti QCS
Properties	• The Rigetti Quantum Cloud Service provides access to Rigetti QPUs (quantum processors) • The Forest SDK contains software tools for creating programs in Quil that are then executed using QCS or a simulator • The Forest SDK consists of three components: • pyQuil: Python library for building and running programs in Quil • quilc: Compiler for Quil Programs • QVM: virtual machine for simulating a quantum computer
Applications	• Creating simulations or programs in Quil • Use of Rigetti's QPUs
Language	• pyQuil: Python • quilc, QVM: Common Lisp
License	• pyQuil, quilc : Apache 2.0 License • QVM: Rigetti License File
Access	• Registration is required for QCS • Otherwise no restrictions
OS	• Platform-independent: Python interface or browser required
Repository	• pyQuil in GitHub • quilc in GitHub • QVM in GitHub
Documentation	• User guides on QCS homepage • (API) References on homepage

Chapter B: Self-Installation Simulators

Simulators for self-installation are usually free of charge and freely available without registration. They are particularly suitable for getting an initial overview of programming and simulation with QuBits. In addition to the creation of (graphical) quantum circuits and algorithms, QKD and quantum network simulators offer the opportunity to investigate the impact and influence of this new technology on current computer networks in more detail.

Quantum Circuit Simulators

blueqat

Simulator	blueqat
Properties	• Simulator written in Python with graphical output • Well suited to control circuits, qubit states etc... to illustrate
Applications	• bluegat is suitable for defining circuits, machine learning applications and optimization tasks
Language	• Python / Jupyter Notebook
License	• Apache 2.0 License
Access	• Free access (GitHub)
OS	• Platform-independent: Python required
Repository	• bluegat in GitHub
Documentation	• Has well-described tutorials

Cirq

Simulator/Library	Cirq
Properties	• Cirq can be used to build and simulate quantum circuits • Google's quantum simulator with extensive documentation • Used in Google Quantum AI
Applications	• Simulation of quantum circuits • Testing algorithms (on a quantum basis)
Language	• Python
License	• Apache 2.0 License
Access	• Code freely available on GitHub
OS	• Platform-independent: requires Python
Repository	• GitHub
Documentation	• Good documentation, tutorials and numerous examples available

myQLM

Simulator	myQLM
Properties	• Simulator developed by Atos for designing and simulating programs in the field of quantum technology • In addition to myQLM, Atos offers other services in the field of quantum technology such as QLaaS (Quantum Learning as a Service), Q-Score and the QLM User Club • Provides interfaces to other well-known simulators such as ProjectQ and Qiskit • API for implementing plugins
Applications	• Creation of quantum circuits and simulations
Language	• Python, Jupyter Notebook, HTML, OpenQASM
License	• Atos myQLM EULA
Access	• Installation possible without registration
OS	• Platform-independent: requires Python interpreter
Repository	• Available on GitHub
Documentation	• Documentation and examples can be found on the myQLM homepage

Ocean

Simulator	Ocean
Properties	• Ocean is a toolbox developed by D-Wave
Applications	• Used to solve difficult optimization problems with quantum computers. • Application examples or problems are Map Coloring, Vertex Cover, Postprocessing with Greedy Solver
Language	• Python
License	• Apache 2.0 License (Ocean SDK)
Access	• SDK installation possible without restrictions • Registration is required to use the Leap Quantum Cloud Service
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• Documents on different D-Wave toolboxes and further information are available here

ProjectQ

Simulator	ProjectQ
Properties	• ETH Zurich project • "High Level" Language for Quantum Programs • Modular and customizable compiler • Backend interface can be selected: e.g. classic or Q hardware • Also has a "Fermilib" library for solving fermion problems on quantum computers • There is also the option in the backend to have the program translated for IBM Quantum hardware
Applications	• Suitable for creating Q applications that are to run on several backends (e.g. classic computer, Q hardware)
Language	• Python
License	• OpenSource: Apache 2 License
Access	• No restrictions:
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• ProjectQ documentation page

pyQuil

Simulator	pyQuil
Properties	• Is part of the Rigetti Forest SDK • Allows you to create and run Quil programs in Python • Quil requires the installation of the Quil compiler and QVM (Quantum Virtual Machine)
Applications	• The Forest SDK or pyQuil is used as part of the QCS (Quantum Cloud Service) to offer users the possibility to use QPU's (quantum processors) via QCS
Language	• Python
License	• pyQuil: Apache 2.0 License
Access	• Installation of pyQuil possible without registration • Registration is required to use QCS
OS	• pyQuil: platform-independent. Python interpreter required
Repository	• pyQuil: Available on GitHub
Documentation	• various topics and tutorials in QCS are available here • Documentation and examples of pyQuil can be found on the pyQuil homepage page

Microsoft QKD (Q#)

Simulator	QDK
Properties	• The Microsoft Quantum Development Kit (QDK) contains four simulators for Q# programs • Q# is a programming language developed by Microsoft with its own syntax specifically for programming in the field of quantum technology • Q# and Quantum Development Kit can also be used without an Azure subscription • Q# or QDK can be used to build applications for Quantum Azure • Q# programs can also be run with Jupyter Notebook
Applications	• Focus is on the development of complex applications or algorithms (not just simple circuits)
Language	• Q# has its own syntax based on Python, C#, F#
License	• Microsoft (QDK)
Access	• Available for free download • Q# is open source • Can also be used online (without downloading) via Visual Studio Codespaces, but this service is not free of charge • QDK can be used via Visual Studio and Visual Studio Code
OS	• Platform-independent when used via Visual Studio Code • Only via Windows when used via Visual Studio
Repository	• Quantum Libraries by Microsoft on GitHub
Documentation	• Download QDK • Q# and QDK documentation

QCircuits

Simulator	QCircuits
Properties	• based on the quantum circuit model • Simple interface for ease of use
Applications	• Simulation and investigation of the behavior of quantum computers • Investigation and construction of the behavior of Q-circuits
Language	• Python package
License	• WITH License
Access	• Installation without registration
Repository	• Available on GitHub
Documentation	• Tutorials, examples, documentation available on the QCircuits homepage

Qiskit

Simulator	Qiskit
Properties	• IBM-Developed Quantum Simulator Framework • Developed circuits can also be tested on real IBM hardware after registration (IBM Quantum) • Also includes a module that can be used to graph circuits • Other circuit simulators (such as "extended Clifford, Schrödinger") are available in the IBM Cloud • Noise models like Pauli, depolarization etc... available
Applications	• Modeling and Simulation of Circuits • Execution of the programs on real quantum hardware developed by IBM • Simulation of (existing) quantum hardware
Language	• Framework written primarily in Python • Other languages used: C++, OpenQASM, Typescript, Vu
License	• Open Source Framework • Apache 2.0 License
Access	• Qiskit is free to download and install and does not require registration • Registration required when using the IBM Cloud and/or testing programs on real quantum hardware
OS	• Platform-independent: Python interpreter required
Repository	• Simulators, other tools and projects for Qiskit are available on GitHub
Documentation	• Qiskit Textbook with numerous examples and explanations

Quantum Computing Playground

Simulator	Quantum Computing Playground
Properties	• WebGL Project in Chrome • Still in the experimental stage • Simulates a quantum computer (on GPU) with its own scripting language • Up to 22 qbits can be simulated • 3D visualization of Q states
Applications	• 3D simulations of quantum states and gates
Language	• Chrome Project: Source code not accessible
License	• n/a
Access	• Targeted at google chrome
OS	• Platform-independent: browser required
Repository	• no repository available
Documentation	• Examples available on the homepage under Examples

Quantum JavaScript (Q.js)

Simulator	Quantum JavaScript (Q.js)
Properties	• Simulator for quantum circuits • Drag and drop editor for web browsers • Circuits can also be defined via source code, which is then displayed as a circuit • Output of the circuits is displayed graphically
Applications	• Graphical creation of Q-circuits
Language	• Javascript, HTML, CSS
License	• WITH License
Access	• Via web browser or as a download via GitHub
OS	• Platform-independent: browser required
Repository	• Available on GitHub
Documentation	• Examples and API documentation available on the Q.js homepage

Qubit Toaster

Simulator	Qubit Toaster
Properties	• Part of the QUANTASTICA project, which focuses on the development of software tools and solutions in the field of quantum computing • The Qubit Toaster is a high-performance quantum circuit simulator designed for speed. • It is based on algorithms for circuit optimization and efficient execution. • It can be used standalone or together with common quantum programming frameworks, e.g. Qiskit, Quantum Programmingt Studio.
Applications	• For quantum simulations that are to be executed at higher or improved speed (HPC)
Language	• n/a
License	• n/a
Access	• No restrictions
OS	• available for Linux, MacOS and Windows
Repository	• no repository: program is installed via precompiled binary
Documentation	• Use of QuBit Toaster described on homepage

Quest

Simulator	Quest
Properties	• distributed, GPU-accelerated simulator of universal quantum circuits, state vectors, and density matrices. • QuEST is an open-source and standalone C/C++ library • Simulation of dephasing and depolarizing noise possible • The same code can be used seamlessly on all hardware backends, and the simulation cost and accuracy can be changed at compile time. • QuEST is currently the only active distributed QC simulator and the first and only one to support a distributed density matrix.
Applications	• Simulation of quantum circuits, status vectors and density matrices
Language	• C/C++, Cuda, JavaScript
License	• WITH License
Access	• No restrictions: download and install for free
OS	• Available for MacOS, Linux and Windows
Repository	• Available on GitHub
Documentation	• Tutorials and examples are available here

Qibo

Simulator	Qibo
Properties	• QIBO is an API for quantum simulations and control of quantum hardware
Applications	• Consideration of the properties of hardware components such as NISQs when performing simulations
Language	• Python
License	• Apache License Version 2.0
Access	• No restrictions
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• Documentation page of Qibo

QuTip

Simulator	QuTIP
Properties	• QuTiP (Quantum Toolbox in Python) is an open-source software for simulating the dynamics of open quantum systems. • graphical output via Matplotlib.
Applications	• is used for efficient numerical simulations of Hamiltonian functions in areas such as quantum optics, ion traps
Language	• Python, HTML. Shell
License	• BSD-3 Clause License, No License Fees
Access	• No restrictions
OS	• Linux, MacOS, Windows
Repository	• Available on GitHub
Documentation	• Documentation and tutorials available on the homepage

Quirk

Simulator	Quirk
Properties	• Quirk is an online graphical simulator for (simple) quantum circuits. • There are also prefabricated circuits that can be used to simulate quantum teleportation, for example
Applications	• Suitable for demonstrating or analyzing the behavior of quantum circuits • Generation of quantum circuits via drag and drop
Language	• Source code available on GitHub (JavaScript)
License	• Apache 2.0 License
Access	• No restrictions (OpenSource)
OS	• Platform-independent: Browser required
Repository	• Available on GitHub
Documentation	• User guide and video tutorial available

Silq

Simulator	Silq
Properties	• Developed by ETH Zurich • Silq is a high-level language among the programming languages for quantum circuits and computers and is easier to use than, for example, OpenQASM • In addition to the simple and plain design, quantum mechanical properties are already taken into account that the user does not have to worry about (anymore), so that the code is less error-prone • An important criterion in quantum computing is the so-called uncomputation, i.e. the reset of QuBits to the initial state, which are used to store intermediate results. These bits are also known as ancilla bits. This reset has to be done with quantum computers, as there are usually only very limited QuBits available and they are to be reused. In classic computers, this task is performed by a garbage collector after a program has been executed • Intuitive variable types for quantum states such as entanglement, superposition states, etc...
Applications	• Creating programs for quantum computers • Investigation of quantum circuits
Language	• Q#, D, Tex, Python
License	FreeBSD License
Access	• No restrictions: download and install without registration
OS	• VS Code is required to install the Silq plug-in
Repository	• Available on GitHub
Documentation	• Documentation and examples available on the homepage

Strawberry Fields

Simulator	Strawberry Fields
Properties	• Python library developed by Xanadu • Used to simulate and execute programs on quantum hardware based on photons • Large selection of tutorials and examples from the field of quantum photonics • Provides access to the first fully programmable photon quantum computer via the Xanadu Cloud
Applications	• Programs/simulations for quantum hardware based on photons
Language	• Python Libraries
License	• Apache 2.0 License
Access	• No restrictions
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• Documentation and examples can be found on the Strawberry Fields homepage under Documentation

XACC

Simulator	XACC
Properties	• Framework for hybrid (classical-quant) computer architectures • Supports programming in the classical and quantum environment • Provides the ability to execute quantum code on quantum processors from Rigetti, IBM, IonQ, and others • Based on the C++ Micro Services project
Applications	• Hybrid computer architectures, executing quantum code on different computer architectures or processors
Language	• C++
License	• Eclipse Public License and Eclipse Distribution License, BSD-3 Clause
Access	• No restrictions
OS	• Ubuntu 16.04/18.04, Centos 7, Fedora 7/30, MacOS X • C++ Compiler and CMake Required
Repository	• Available on GitHub
Documentation	• Tutorials, examples and further information can be found on the XACC documentation page

Quantum Network Simulators

Interlin-q

Simulator	Interlin-q
Properties	• Simulator using a master-slave control structure in a quantum network • With Interlin-q, individual circuits as well as distributed algorithms can be defined and simulated in a quantum network topology • Quantum circuits are transferred to a quantum computer architecture and the communication between the individual nodes is regulated according to the master-slave principle
Applications	• Build and test distributed quantum algorithms on different quantum computer architectures
Language	• Python
License	• WITH License
Access	• No restrictions
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• Interlin-q Documentation Page

NetSquid

Simulator	NetSquid
Properties	• Models the influence of time in quantum networks and computer systems • Modular structure: Individual components can be nested inside each other (Quantum Computing Library)
Applications	• For the design/simulation of a quantum-based internet • For the design of modular quantum computer architectures • Performance Investigation of the Physical Layer (Quantum Hardware) of Quantum Networks
Language	• Python Package
License	• WITH License
Access	• Free, but registration required for download
OS	• Platform-independent: Python interpreter required
Repository	• GitHub repository only available with examples
Documentation	• Available on Netsquid's homepage

OpenQL

Simulator	OpenQL
Properties	• Framework for quantum programming in Python/C++ • Unlike Qiskit, for example, the focus is on generating assembly code for various (micro-) architectures of QuTech • The format of the assembly code is cQASM (Quantum Assembly Language) • Format of the output code depends on the platform
Applications	• To generate code for QuTech architectures
Language	• C++, Python, HTML, JavaScript, OpenQASM
License	• Apache 2.0 License
Access	• No restrictions
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• OpenQL documentation page

QuISP

Simulator	QuISP
Properties	• Event-controlled simulator for quantum repeater networks • The aim of the project is to simulate 100 networks with 100 nodes each • With such large networks, it is not possible to simulate them at the Hamiltonian level or CNOT gate; therefore, only Q misstates are recorded and not the complete Q status (error basis) • Supports all except "Pauli" error types (different from other simulators) • Setting link lengths in the network, gate error rates, memory states of individual Q-bits • Requires OmNET++ and Eigen, a matrix calculator for C/C++ • QuISP is a product of the AQUA (Advances Quantum Architecture) research group
Applications	• Protocol Design • Investigation of the behavior of large complex heterogeneous networks
Language	• C++, Python, Shell
License	• BSD 3 Clause License
Access	• None: Download and install for free
OS	• Available for Linux, Windows and MacOS
Repository	• Available on Github
Documentation	• Documentation OmNET++ • Documentation QuISP

QuNetSim

Simulator	QuNetSim
Properties	• Simulator for quantum networks • Provides framework for the development of protocols in quantum networks • Already contains quantum basic technologies such as quantum teleportation, QKD generation etc... • Uses the Python graphical networkx library to display and generate networks • Event-controlled simulator • Treats quantum networks like classical networks in terms of the layer model
Applications	• For example, suitable for tracking (step by step) a QuBIt teleportation in the Q-network • For trying out or creating network protocols at a "high level" level • Not suitable for accurate simulation of quantum effects • Suitable for beginners in quantum networks
Language	• Python
License	• WITH License
Access	• No restrictions: download and install for free
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• Documentation available here

SeQUeNCe

Simulator	SeQUeNCe
Properties	• Used to analyze effects in quantum networks on the lower network layers, such as the caching of quantum states
Applications	• To test protocols, network parameters, and topologies
Language	• C++, Python, Makefile
License	• Open Source License
Access	• No restrictions
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• SeQUeNCe Documentation Page

SimulaQron

Simulator	SimulaQron
Properties	• Simulator for application layer in quantum networks • Developed by QuTech • Provides infrastructure of distributed Q processors connected via Q communication channels • Each Q processor is available via a server that runs on a normal PC (also distributed on different PCs); SimulaQron connects the processors to transmit Q-bits and entanglement over distances • The simulated hardware can then be accessed via Python, C libraries or the CQC interface
Applications	• Creation of own applications for the "quantum internet" • Development of software engineering concepts and libraries for Quantum Internet
Language	• Python
License	• See license file
Access	• No restrictions
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• Installation • SimulaQron Documentation

SQUANCH

Simulator	SQUANCH
Properties	• developed for the simulation of quantum networks • was developed as part of the INQNET program • contains classical and quantum error models • Protocols such as Quantum Error Correction included in examples (as source code)
Applications	• For testing network protocols and quantum transmission • Simulation of multiparty networks
Language	• Python
License	• WITH License
Access	• No restrictions
OS	• Platform-independent: Python interpreter required
Repository	• Available on GitHub
Documentation	• SQUANCH Documentation Page

QKD Simulators

QKDNetSim

Simulator	QKDNetSim
Properties	• It is a QKD simulation module that is embedded in the established NS-3 simulator for networks • Therefore, no complete quantum simulator • Networks can be scanned on a QKD basis in overlay or TCP/IP mode • Based on the NS-3 Network Simulator (s. here) • QKDNetSim can also be used via a web interface
Applications	• Suitable for investigating the effects or behaviour in conventional networks
Language	• C/C++, Python, Perl
License	• GPL 2.0 License
Access	• No restrictions
OS	• Linux
Repository	• Available on GitHub
Documentation	• QKDNetSim Documentation

QKDSimulator

Simulator	QKDSimulator
Properties	• Online simulator that can be used to analyze and simulate QKD protocols • Parameters (number of QBits, fault tolerance etc...) can be set via sliders • There are 4 simulator types to choose from: *Complete QKD Stack*, *Sifting*, *Biased Error Estimation* and *Shannon Bound* • After simulation, results are summarized under their own tab • Plots for the results are also available
Applications	• Simulator for certain initial values for QKD
Language	• Python Program
License	• N/A
Access	• No restrictions
OS	• Platform-independent: web browser required
Repository	• no respository
Documentation	• available on website

Quantum annealing

D-Wave Ocean

Simulator	D-Wave Ocean
Properties	• D-Wave Ocean is a software suite consisting of several independent packages • Each package is designed for an individual application: For example, dwave-hybrid is suitable for solving mathematical problems on hybrid (classical + based on quantum) architectures
Applications	• Solving hard-to-calculate mathematical problems. Depending on the software package, the code can also be executed on quantum annealers from D-Wave
Language	• Different programming languages. Depending on the respective D-Wave Ocean package. Most packages are in Python or C++
License	• Apache 2.0 license or MIT license. Depending on the respective software package
Access	• No restrictions
OS	• Installation tested for Linux, Mac OS and Windows
Repository	• Available on GitHub
Documentation	• D-Wave Ocean Software Products Documentation Page

Other simulators

Other simulators can be found on the following URLs:

https://www.win-labor.dfn.de/quantentechnologien/quantensimulation/

https://quantiki.org/wiki/list-qc-simulators

https://github.com/qosf/awesome-quantum-software

https://qosf.org/project_list/