CLAIMS

1. A digital computer system for compiling a computer program for a target quantum processor comprising a plurality of qubits, qudits, or quantum continuous variables, the computer system comprising:

one or more digital processors;

a memory; and

one or more programs in the form of a compiler, wherein the compiler comprises:

(A) instructions for obtaining a computer program written in a unified language and performing code refactoring on all or a portion of the computer program to form a refactored code and converting the refactored code into a first code;

(B) instructions for compiling the first code into a second code, wherein the second code comprises a plurality of data elements in one or more quantum data structures determined by the compiler from one or more digital datatypes determined by the compiler from one or more digital datatypes in the first code; and

(C) instructions for compiling the second code into a third code expressed in a quantum gate-level language in accordance with an instruction set and one or more gate locality constraints of the target quantum processor,

wherein the quantum gate-level language uses a plurality of different gates, wherein each respective gate in the plurality of different gates is defined in a different data structure that includes a name of the respective gate, a dimension of the respective gate in terms of number of qubits or qudits, and a corresponding plurality of Kraus operators for the respective gate.

2. The computer system of claim 1, wherein the compiler further comprises instructions for converting the second code to a fourth code comprising a series of quantum gate level operations, and wherein the instructions for compiling the second code into the third code further includes instructions for compiling the fourth code into the third code.

3. The computer system according to claim 1, wherein

the compiler performs code refactoring at least in part by replacing a source loop in the computer program written in the unified language with a plurality of loops that collectively accomplish a result of the source loop, and wherein a complexity of each loop in the plurality of loops is less than that of the source loop, and

the compiler performs code refactoring at least in part by analytically solving a loop in the plurality of loops.

4. The computer system of claim 1, wherein the compiler:

provides access to quantum gate level commands specified in the first code, and

supports pointers that are encoded on qubits, qudits, or quantum continuous variables in the plurality of qubits, qudits or quantum continuous variables, functions, loops that cannot be unrolled, recursion, pointers, data structure definitions, class object definitions located in the first code, or a combination thereof.

5. The computer system of claim 1, wherein the first code defines a quantum data structure.

6. The computer system of claim 5, wherein the quantum data structure is a graph data structure or a matrix.

7. The computer system of claim 1, wherein the unified language is limited to digital instructions.

8. The computer system of claim 1, wherein the second code is written in a gate level language that is augmented with a conditional and wherein the second code includes a conditional.

9. The computer system of claim 8, wherein the conditional is an if condition or repeat condition in which an action is repeated until a predetermined condition is achieved.

10. The computer system of claim 9, wherein the predetermined condition is a digital measurement outcome for one or more qubits, qudits, or quantum continuous variables defined by the second code.

11. The computer system of claim 9, wherein the gate level language is augmented with support for a subroutine, in which one portion of the second code is repeatedly called by another portion of the second code until a predetermined outcome is achieved.

12. The computer system of claim 11, wherein the subroutine is a measurement subroutine.

13. The computer system of claim 12, wherein the plurality of qubits, qudits, or quantum continuous variables is a plurality of qubits or qudits and the measurement subroutine makes use of ancilla qubits or qudits in the plurality of qubits or qudits to improve a measurement outcome.

14. The computer system of claim 11, wherein the predetermined outcome is a measurement outcome for one or more qubits or qudits defined by the second code.

15. The computer system of claim 8, wherein the conditional is a parameter of an increment loop.

16. The computer system of claim 8, wherein the compiler converts the second code to the third code at least in part by optimizing a quantum operation that serves to perform a portion of the second code, thereby reducing a number of gates, a number of qubits, a number of qudits, or a number of quantum continuous variables in the third code that are needed to perform the quantum operation.

17. The computer system of claim 1, wherein the second code defines a qubit, qudit, or quantum continuous variable within the target quantum processor.

18. The computer system of claim 1, wherein the second code defines a register of qubits or a register of qudits within the target quantum processor.

19. The computer system of claim 8, wherein the gate level language supports qubit, qudit, or quantum continuous variable measurement.

20. The computer system of claim 1, wherein a subset of Kraus operators in the corresponding plurality of Kraus operators for the respective gate is specified by a function handle.

21. The computer system of claim 2, wherein the fourth code supports general positive operator-valued measures of one or more qubits, qudits, or quantum continuous variables in the plurality of qubits, qudits, or quantum continuous variables.

22. The computer system of claim 2, wherein the compiler compiles the fourth code into the third code at least in part by performing gate synthesis in which a sequence of ideal gates, which implement at least a portion of the fourth code, are swapped with a sequence of gates that differ from the sequence of ideal gates but will perform the function, or substantially the same function, of the sequence of ideal gates on the target quantum processor.

23. The computer system of claim 22, wherein the sequence of gates that differ from the sequence of ideal gates are identified by a stored hardware characterization of the target quantum processor.

24. A digital computer system of claim 1 for compiling a computer program for a target quantum processor comprising a plurality of qubits, qudits, or quantum continuous variables, the computer system comprising:

one or more digital processors;

a memory; and

one or more programs in the form of a compiler, wherein the compiler comprises:

(A) instructions for obtaining a computer program written in a unified language and performing code refactoring on all or a portion of the computer program to form a refactored code and converting the refactored code into a first code;

(B) instructions for compiling the first code into a second code, wherein the second code comprises a plurality of data elements in one or more quantum data structures determined by the compiler from one or more digital datatypes determined by the compiler from one or more digital datatypes in the first code; and

(C) instructions for compiling the second code into a third code expressed in a quantum gate-level language in accordance with an instruction set and one or more gate locality constraints of the target quantum processor, wherein

the compiler further comprises instructions for circuit embedding of the third code in accordance with one or more locality constraints of the target quantum processor, and

the instructions for circuit embedding of the third code in accordance with the one or more locality constraints of the target quantum processor comprises taking into account a determined quality of the target quantum processor at performing 2-qubit gates or 2-qudit gates.

25. The computer system of claim 1, wherein the compiler further comprises instructions for counting a number of gates, qubits, qudits, quantum continuous variables or wall clock time needed to implement the computer program on the target quantum processor.

26. The computer system of claim 1, wherein the compiler further comprises instructions for implementing a first portion of the computer program on the target quantum processor and a second portion of the computer program on a digital central processing unit or a graphical processing unit of a digital computer system.

27. The computer system of claim 1, wherein the compiler further comprises instructions for simulating quantum code generated by the compiler on a digital central processing unit or a graphical processing unit of a digital computer system.

28. A method for compiling a computer program that can be run on a target quantum processor, the target quantum processor comprising a plurality of qubits, qudits, or quantum continuous variables, the method comprising:

at a digital computer system comprising one or more digital processors and a memory, the memory comprising non-transitory instructions configured to perform a procedure using the one or more digital processors, the procedure comprising:

(A) obtaining a computer program written in a unified language;

(B) performing code refactoring on all or a portion of the computer program to form a refactored code;

(C) converting the refactored code into a first code;

(D) compiling the first code into a second code, wherein the second code comprises a plurality of data elements in one or more quantum data structures determined by a compiler from one or more digital datatypes in the first code; and

(E) compiling the second code into a third code expressed in a quantum gate-level language in accordance with an instruction set and one or more gate locality constraints of the target quantum processor,

wherein the quantum gate-level language uses a plurality of different gates, wherein each respective gate in the plurality of different gates is defined in a different data structure that includes a name of the respective gate, a dimension of the respective gate in terms of number of qubits or qudits, and a corresponding plurality of Kraus operators for the respective gate.

29. A non-transitory computer readable storage medium storing one or more programs, the one or more programs comprising instructions, which when executed by a digital computer system with one or more digital processors, cause the digital computer system to:

(A) obtain a computer program written in a unified language;

(B) perform code refactoring on all or a portion of the computer program to form a refactored code;

(C) convert the refactored code into a first code;

(D) compile the first code into a second code, wherein the second code comprises a plurality of data elements in one or more quantum data structures determined by a high level module from one or more digital datatypes in the first code; and

(E) compiling the second code into a third code expressed in a quantum gate-level language in accordance with an instruction set and one or more gate locality constraints of a target quantum processor,

wherein the quantum gate-level language uses a plurality of different gates, wherein each respective gate in the plurality of different gates is defined in a different data structure that includes a name of the respective gate, a dimension of the respective gate in terms of number of qubits or qudits, and a corresponding plurality of Kraus operators for the respective gate.