(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
08 October 2020 (08.10.2020)

WIPO | PCT

(10) International Publication Number
WO 2020/200758 A1

(51) International Patent Classification:
G06N 10/00 (2019.01)

(21) International Application Number:
PCT/EP2020/057239

(22) International Filing Date:
17 March 2020 (17.03.2020)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
16/370,062 29 March 2019 (29.03.2019) US

(71) Applicant: **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, New York 10504 (US).

(71) Applicant *(for MG only)*: **IBM UNITED KINGDOM LIMITED** [GB/GB]; PO Box 41, North Harbour, Portsmouth Hampshire PO6 3AU (GB).

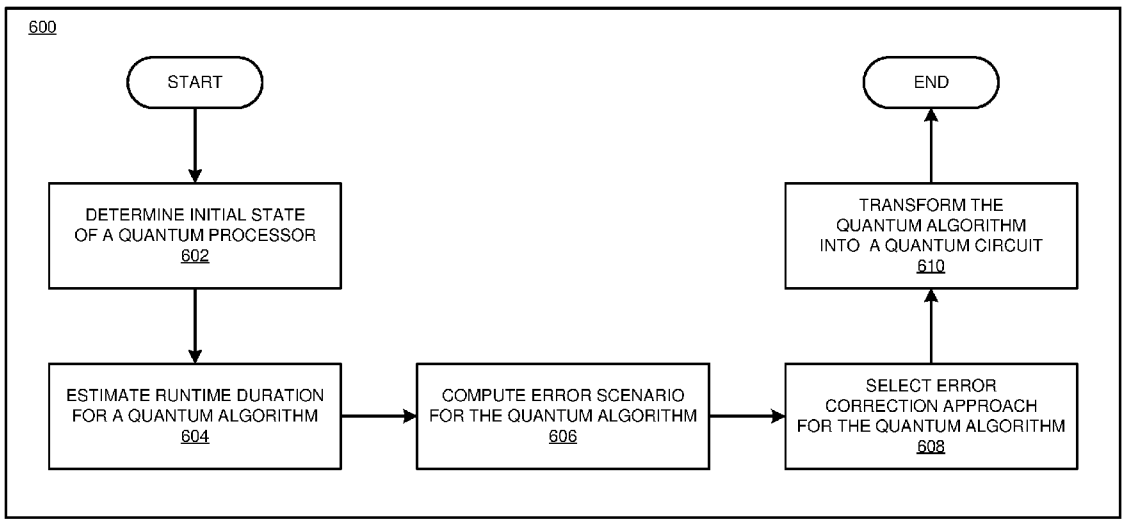
(72) Inventors: **GUNNELS, John**; IBM Corporation, PO Box 218, 1101 Kitchawan Road, Yorktown Heights, New York 10598 (US). **WEGMAN, Mark**; IBM Corporation, PO Box 218, 1101 Kitchawan Road, Yorktown Heights, New York 10598 (US). **KAMINSKY, David**; IBM Corporation, 4205 South Miami Boulevard, Durham, North Carolina 27703-9141 (US).

(74) Agent: **GASCOYNE, Belinda**; IBM United Kingdom Limited, Intellectual Property Law, Hursley Park, Winchester Hampshire SO21 2JN (GB).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO,

(54) Title: ADAPTIVE ERROR CORRECTION IN QUANTUM COMPUTING

FIGURE 6



(57) Abstract: A method for adaptive error correction in quantum computing includes executing a calibration operation on a set of qubits, the calibration operation determining an initial state of a quantum processor. In an embodiment, the method includes estimating, responsive to determining an initial state of the quantum processor, a runtime duration for a quantum circuit design corresponding to a quantum algorithm, the quantum processor configured to execute the quantum circuit design. In an embodiment, the method includes computing an error scenario for the quantum circuit design. In an embodiment, the method includes selecting, using the error scenario and the initial state of the quantum processor, a quantum error correction approach for the quantum circuit design. In an embodiment, the method includes transforming the quantum algorithm into the quantum circuit design, the quantum circuit design including a set of quantum logic gates.

WO 2020/200758 A1

[Continued on next page]

DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

ADAPTIVE ERROR CORRECTION IN QUANTUM COMPUTING

TECHNICAL FIELD

[0001] The present invention relates generally to a method for quantum computing. More particularly, the present invention relates to a method for adaptive error correction in quantum computing.

BACKGROUND

[0002] Hereinafter, a "Q" prefix in a word of phrase is indicative of a reference of that word or phrase in a quantum computing context unless expressly distinguished where used.

[0003] Molecules and subatomic particles follow the laws of quantum mechanics, a branch of physics that explores how the physical world works at the most fundamental levels. At this level, particles behave in strange ways, taking on more than one state at the same time, and interacting with other particles that are very far away. Quantum computing harnesses these quantum phenomena to process information.

[0004] The computers we commonly use today are known as classical computers (also referred to herein as "conventional" computers or conventional nodes, or "CN"). A conventional computer uses a conventional processor fabricated using semiconductor materials and technology, a semiconductor memory, and a magnetic or solid-state storage device, in what is known as a Von Neumann architecture. Particularly, the processors in conventional computers are binary processors, i.e., operating on binary data represented by 1 and 0.

[0005] A quantum processor (q-processor) uses the unique nature of entangled qubit devices (compactly referred to herein as "qubit," plural "qubits") to perform computational tasks. In the particular realms where quantum mechanics operates, particles of matter can exist in multiple states—such as an "on" state, an "off" state, and both "on" and "off" states simultaneously. Where binary computing using semiconductor processors is limited to using just the on and off states (equivalent to 1 and 0 in binary code), a quantum processor harnesses these quantum states of matter to output signals that are usable in data computing.

[0006] Conventional computers encode information in bits. Each bit can take the value of 1 or 0. These 1s and 0s act as on/off switches that ultimately drive computer functions. Quantum computers, on the other hand, are based on qubits, which operate according to two key principles of quantum physics: superposition and entanglement. Superposition means that each qubit can represent both a 1 and a 0 inference between possible outcomes for an event. Entanglement means that qubits in a superposition can be correlated with each other in a non-classical way; that is, the state of one (whether it is a 1 or a 0 or both) can depend on the state of another, and that there is more information contained within the two qubits when they are entangled than as two individual qubits.

[0007] Using these two principles, qubits operate as processors of information, enabling quantum computers to function in ways that allow them to solve certain difficult problems that are intractable using conventional computers.

[0008] A superconducting qubit includes a Josephson junction. A Josephson junction is formed by separating two thin-film superconducting metal layers by a non-superconducting material. When the metal in the superconducting layers is caused to become superconducting – e.g. by reducing the temperature of the metal to a specified cryogenic temperature – pairs of electrons can tunnel from one superconducting layer through the non-superconducting layer to the other superconducting layer. In a qubit, the Josephson junction – which functions as a dispersive nonlinear inductor - is electrically coupled in parallel with one or more capacitive devices forming a nonlinear microwave oscillator. The oscillator has a resonance/transition frequency determined by the value of the inductance and the capacitance in the qubit circuit.

[0009] The information processed by qubits is carried or transmitted in the form of microwave signals/photons in the range of microwave frequencies. The microwave signals are captured, processed, and analyzed to decipher the quantum information encoded therein. A readout circuit is a circuit coupled with the qubit to capture, read, and measure the quantum state of the qubit. An output of the readout circuit is information usable by a q-processor to perform computations.

[0010] A superconducting qubit has two quantum states – |0> and |1>. These two states may be two energy states of atoms, for example, the ground (|g>) and first excited state (|e>) of a superconducting artificial atom (superconducting qubit). Other examples include spin-up and spin-down of the nuclear or electronic spins, two positions of a crystalline defect, and two states of a quantum dot. Since the system is of a quantum nature, any combination of the two states are allowed and valid.

[0011] For quantum computing using qubits to be reliable, quantum circuits, e.g., the qubits themselves, the readout circuitry associated with the qubits, and other parts of the quantum processor, must not alter the energy states of the qubit, such as by injecting or dissipating energy, in any significant manner or influence the relative phase between the |0> and |1> states of the qubit. This operational constraint on any circuit that operates with quantum information necessitates special considerations in fabricating semiconductor and superconducting structures that are used in such circuits.

[0012] In conventional circuits, Boolean logic gates arranged in succession manipulate a series of bits. The technology for optimizing the gate-logic for binary computations is well-known. Circuit optimization software for conventional circuits aims to increase efficiency and decrease complexity of conventional circuits. Circuit optimization software for conventional circuits functions in part by decomposing the overall desired behavior of the conventional circuit into simpler functions. The conventional circuit optimization software more easily manipulates and processes the simpler functions. The circuit optimization software generates an efficient layout of design

elements on the conventional circuit. As a result, circuit optimization software for conventional circuits significantly reduces resource demands, thereby increasing efficiency and decreasing complexity.

[0013] The illustrative embodiments recognize that in quantum circuits, quantum gates manipulate qubits to perform quantum computations. Quantum gates are unitary matrix transformations acting on qubits. Due to the superposition and entanglement of qubits, quantum gates represent a 2^n by 2^n matrix, where n is the number of qubits the quantum gate manipulates. The illustrative embodiments recognize that the decomposition of such matrix transformations quickly becomes too complex to perform by hand due to the exponential increase in the size of the matrix transformations with the number of qubits. For example, quantum computers with 2 qubits require a 4 by 4 matrix operator for quantum gate representation. As a result of the exponential increase, manual quantum logic gate matrix transformations quickly become unmanageable as the number of qubits increases. The depth of a quantum circuit is the total number of time steps. Each time step corresponds to a quantum logic gate acting on a set of qubits.

[0014] The illustrative embodiments recognize that quantum processors exhibit varying physical characteristics over a period of time. The illustrative embodiments recognize that calibration of quantum processors determines error rates for associated qubits and quantum gates of the quantum processor. The illustrative embodiments further recognize that quantum processors are often calibrated only once or twice daily.

[0015] The illustrative embodiments recognize that quantum computers are more susceptible to errors than classical computers due to the sensitivity of quantum computers resulting in unwanted interactions with the environment. The illustrative embodiments recognize that quantum states and quantum operations are continuous, i.e. belong to continuous intervals, which allows small errors to accumulate over an entire quantum circuit. The illustrative embodiments further recognize that classical error correcting schemes perform a measurement to inquire about the nature and type of error. The illustrative embodiments recognize that performing a measurement on a quantum system collapses the quantum system and can result in loss of information encoded in a quantum state from collapsing the quantum state to a classical state.

[0016] The illustrative embodiments recognize that presently available quantum error correction techniques require varying amounts of quantum computing resources to operate. A three-qubit bit flip code uses three qubits to determine and correct errors in one of the three qubits. The first, or target qubit, is transmitted through a noisy channel which either flips the state of the first qubit, with probability p, or leaves the state of the first qubit unchanged. Thus, the state of the first qubit is correctly transmitted with probability $1-p$.

[0017] To improve this probability, the state of the first qubit is encoded into additional qubits. The first qubit is entangled with two ancillary qubits, and all three qubits are initialized in state $|0\rangle$. As the encoded state of three qubits is transmitted through the noisy channel, the channel acts on the encoded state and flips a subset of the

three qubits. The probability no qubit is flipped is $(1-p)^3$. The probability a single qubit is flipped is $3p*(1-p)^2$. Two qubits are flipped with probability $3*p^2*(1-p)$ and all three qubits are flipped with probability p^3.

[0018] The illustrative embodiments recognize that p is generally a very small number. By determining whether one qubit differs from the others, detection of whether one of the qubits was flipped is determined. Since all qubits began in the same state, if any qubit differs from the others, the qubit was flipped while transmitting through the channel. If the qubit was flipped, a Pauli X gate or quantum bit-flip gate is applied to the flipped qubit.

[0019] The illustrative embodiments further recognize that sign flip produces another error in quantum computers. A Hadamard gate maps basis state $|0\rangle$ to $(|0\rangle + |1\rangle)/\sqrt{2}$ and basis state $|1\rangle$ to $(|0\rangle - |1\rangle)/\sqrt{2}$. Using the three qubit bit flip code above and adding a layer of Hadamard gates on each qubit before and after transmission through the noisy channel, the three qubit bit flip code is transformed to a three qubit sign flip code.

[0020] The illustrative embodiments further recognize that more complex quantum error correction techniques require additional quantum computing resources. The Shor code requires nine total qubits, one target qubit and eight ancillary qubits. The Shor code corrects for bit slips, sign flips, and any arbitrary single-qubit errors. The Steane code requires seven total qubits, one target qubit and six ancillary qubits to correct for any arbitrary single-qubit errors. Calderbank-Shor-Steane (CSS) codes require a minimum of five total qubits, one target qubit and four ancillary qubits to correct for any arbitrary single-qubit errors. The illustrative embodiments recognize that choosing an appropriate error correction technique depends on the quantum computing resources available.

SUMMARY

[0021] The illustrative embodiments provide a method for adaptive error correction in quantum computing. In an embodiment, a method includes executing a calibration operation on a set of qubits, the calibration operation determining an initial state of a quantum processor. In an embodiment, a method includes estimating, responsive to determining an initial state of the quantum processor, a runtime duration for a quantum circuit design corresponding to a quantum algorithm, the quantum processor configured to execute the quantum circuit design.

[0022] In an embodiment, a method includes computing an error scenario for the quantum circuit design. In an embodiment, a method includes selecting, using the error scenario and the initial state of the quantum processor, a quantum error correction approach for the quantum circuit design. In an embodiment, a method includes transforming the quantum algorithm into the quantum circuit design, the quantum circuit design including a set of quantum logic gates.

[0023] In an embodiment, a subset of the set of quantum logic gates corresponds to the selected quantum error correction approach. In an embodiment, the initial state of the quantum processor includes a total number of qubits of the quantum processor. In an embodiment, a method includes determining a depth of the quantum circuit design.

[0024] In an embodiment, a method includes assigning a set of weights to the set of quantum logic gates. In an embodiment, a method includes comparing the quantum circuit design to a set of previously executed quantum circuit designs. In an embodiment, a method includes determining a first computation cost for the selected quantum error correction approach.

[0025] In an embodiment, a method includes determining a second computation cost for a probabilistic quantum error correction approach. In an embodiment, a method includes comparing the quantum circuit design to a set of previously executed quantum circuit designs. In an embodiment, a method includes selecting a quantum error correction approach for a first quantum circuit design of the set of previously executed quantum circuit designs. In an embodiment, a method includes transforming the quantum algorithm into a new quantum circuit design, the new quantum circuit design including a set of quantum logic gates corresponding to the first quantum circuit design.

[0026] An embodiment includes a computer usable program product. The computer usable program product includes a computer-readable storage device, and program instructions stored on the storage device.

[0027] In an embodiment, the computer usable code is stored in a computer readable storage device in a data processing system, and wherein the computer usable code is transferred over a network from a remote data processing system. In an embodiment, the computer usable code is stored in a computer readable storage device in a server data processing system, and wherein the computer usable code is downloaded over a network to a remote data processing system for use in a computer readable storage device associated with the remote data processing system

[0028] An embodiment includes a computer system. The computer system includes a processor, a computer-readable memory, and a computer-readable storage device, and program instructions stored on the storage device for execution by the processor via the memory.

BRIEF DESCRIPTION OF THE DRAWINGS

[0029] The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objectives and advantages thereof, will best be understood by reference to the following detailed description of the illustrative embodiments when read in conjunction with the accompanying drawings, wherein:

Figure 1 depicts a block diagram of a network of data processing systems in which illustrative embodiments may be implemented;

Figure 2 depicts a block diagram of a data processing system in which illustrative embodiments may be implemented;

Figure 3 depicts an example configuration of adaptive error correction in quantum computing in accordance with an illustrative embodiment;

Figure 4 depicts an example configuration for calibration of a quantum processor in accordance with an illustrative embodiment;

Figure 5 depicts an example configuration for quantum circuit construction in accordance with an illustrative embodiment;

Figure 6 depicts a flowchart of an example method for adaptive error correction in quantum computing in accordance with an illustrative embodiment;

Figure 7 depicts a flowchart of an example method for estimating runtime of a quantum circuit design corresponding to a quantum algorithm in accordance with an illustrative embodiment;

Figure 8 depicts a flowchart of an example method for computing an error scenario in accordance with an illustrative embodiment; and

Figure 9 depicts a flowchart of an example method for selecting an error correction approach in accordance with an illustrative embodiment.

DETAILED DESCRIPTION

[0030] The illustrative embodiments used to describe the invention generally address and solve the above-described needs for reducing error rates on qubits and quantum gates in a quantum processor. The illustrative embodiments provide a method for adaptive error correction in quantum computing.

[0031] In an embodiment, an initiation of a quantum algorithm on a quantum computer is detected. In the embodiment, an initial state of degradation for the quantum computer is determined. In particular embodiments, randomized benchmarking is used to determine the initial state of degradation for the quantum computer. In particular embodiments, a set of pre-determined operations on a plurality of qubits are performed on the quantum computer. The set of pre-determined operations generate a set of values for each qubit in response to performing the set of pre-determined operations. In an embodiment, at least one of the set of values for each qubit are compared to an expected answer of at least one of the set of pre-determined operations.

[0032] In an embodiment, a time period for the quantum algorithm is estimated. In particular embodiments, the depth of the quantum circuit is determined. In an embodiment, an application determines a frequency of gates performed on a set of qubits. In particular embodiments, weights are assigned to quantum logic gates of a quantum circuit corresponding to the quantum algorithm. In particular embodiments, quantum logic gates with greater error rates are assigned greater weights. In an embodiment, an application compares the estimated time period to a decoherence time of a set of qubits of the quantum processor.

[0033] In an embodiment, the quantum algorithm is compared to a second quantum algorithm, the second quantum algorithm corresponding to a compiled quantum circuit. In particular embodiments, the initial state of the quantum computer is compared to a second initial state corresponding to the second quantum algorithm.

[0034] In an embodiment, an error correction approach is selected responsive to the comparison. In particular embodiments, an error correction approach includes executing a plurality of iterations of the quantum algorithm, the plurality of iterations producing a plurality of outputs. In particular embodiments, the plurality of outputs are averaged. In particular embodiments, a second time period for the selected error correction approach is selected.

[0035] An operation described herein as occurring with respect to a frequency of frequencies should be interpreted as occurring with respect to a signal of that frequency or frequencies. All references to a "signal" are references to a microwave signal unless expressly distinguished where used.

[0036] For the clarity of the description, and without implying any limitation thereto, the illustrative embodiments are described using some example configurations. From this disclosure, those of ordinary skill in the art will be able to conceive many alterations, adaptations, and modifications of a described configuration for achieving a described purpose, and the same are contemplated within the scope of the illustrative embodiments.

[0037] Furthermore, simplified diagrams of the example resistors, inductors, capacitors, and other circuit components are used in the figures and the illustrative embodiments. In an actual fabrication or circuit, additional structures or component that are not shown or described herein, or structures or components different from those shown but for a similar function as described herein may be present without departing the scope of the illustrative embodiments.

[0038] The illustrative embodiments are described with respect to certain types of materials, electrical properties, steps, numerosity, frequencies, circuits, components, and applications only as examples. Any specific manifestations of these and other similar artifacts are not intended to be limiting to the invention. Any suitable manifestation of these and other similar artifacts can be selected within the scope of the illustrative embodiments.

[0039] The examples in this disclosure are used only for the clarity of the description and are not limiting to the illustrative embodiments. Any advantages listed herein are only examples and are not intended to be limiting to the illustrative embodiments. Additional or different advantages may be realized by specific illustrative embodiments. Furthermore, a particular illustrative embodiment may have some, all, or none of the advantages listed above.

[0040] With reference to the figures and in particular with reference to **Figures 1** and **2**, these figures are example diagrams of data processing environments in which illustrative embodiments may be implemented. **Figures 1** and **2** are only examples and are not intended to assert or imply any limitation with regard to the environments in which different embodiments may be implemented. A particular implementation may make many modifications to the depicted environments based on the following description.

[0041] **Figure 1** depicts a block diagram of a network of data processing systems in which illustrative embodiments may be implemented. Data processing environment **100** is a network of computers in which the illustrative embodiments may be implemented. Data processing environment **100** includes network **102**. Network

102 is the medium used to provide communications links between various devices and computers connected together within data processing environment **100**. Network **102** may include connections, such as wire, wireless communication links, or fiber optic cables.

[0042] Clients or servers are only example roles of certain data processing systems connected to network **102** and are not intended to exclude other configurations or roles for these data processing systems. Classical processing system **104** couples to network **102**. Classical processing system **104** is a classical processing system. Software applications may execute on any quantum data processing system in data processing environment **100**. Any software application described as executing in classical processing system **104** in **Figure 1** can be configured to execute in another data processing system in a similar manner. Any data or information stored or produced in classical processing system **104** in **Figure 1** can be configured to be stored or produced in another data processing system in a similar manner. A classical data processing system, such as classical processing system **104**, may contain data and may have software applications or software tools executing classical computing processes thereon.

[0043] Server **106** couples to network **102** along with storage unit **108**. Server **106** is a conventional data processing system. Storage unit **108** can store a set of quantum algorithms, a set of quantum circuits, a set of quantum logic gates, and a set of quantum error correction codes as described herein. Quantum processing system **140** couples to network **102**. Quantum processing system **140** is a quantum data processing system. Software applications may execute on any quantum data processing system in data processing environment **100**. Any software application described as executing in quantum processing system **140** in **Figure 1** can be configured to execute in another quantum data processing system in a similar manner. Any data or information stored or produced in quantum processing system **140** in **Figure 1** can be configured to be stored or produced in another quantum data processing system in a similar manner. A quantum data processing system, such as quantum processing system **140**, may contain data and may have software applications or software tools executing quantum computing processes thereon.

[0044] Clients **110**, **112**, and **114** are also coupled to network **102**. A conventional data processing system, such as server **106**, or client **110**, **112**, or **114** may contain data and may have software applications or software tools executing conventional computing processes thereon.

[0045] Only as an example, and without implying any limitation to such architecture, **Figure 1** depicts certain components that are usable in an example implementation of an embodiment. For example, server **106**, and clients **110**, **112**, **114**, are depicted as servers and clients only as example and not to imply a limitation to a client-server architecture. As another example, an embodiment can be distributed across several conventional data processing systems, quantum data processing systems, and a data network as shown, whereas another embodiment can be implemented on a single conventional data processing system or single quantum data processing system within the

scope of the illustrative embodiments. Conventional data processing systems **106**, **110**, **112**, and **114** also represent example nodes in a cluster, partitions, and other configurations suitable for implementing an embodiment.

[0046] Device **132** is an example of a conventional computing device described herein. For example, device **132** can take the form of a smartphone, a tablet computer, a laptop computer, client **110** in a stationary or a portable form, a wearable computing device, or any other suitable device. Any software application described as executing in another conventional data processing system in **Figure 1** can be configured to execute in device **132** in a similar manner. Any data or information stored or produced in another conventional data processing system in **Figure 1** can be configured to be stored or produced in device **132** in a similar manner.

[0047] Server **106**, storage unit **108**, classical processing system **104**, quantum processing system **140**, and clients **110**, **112**, and **114**, and device **132** may couple to network **102** using wired connections, wireless communication protocols, or other suitable data connectivity. Clients **110**, **112**, and **114** may be, for example, personal computers or network computers.

[0048] In the depicted example, server **106** may provide data, such as boot files, operating system images, and applications to clients **110**, **112**, and **114**. Clients **110**, **112**, and **114** may be clients to server **106** in this example. Clients **110**, **112**, **114**, or some combination thereof, may include their own data, boot files, operating system images, and applications. Data processing environment **100** may include additional servers, clients, and other devices that are not shown.

[0049] In the depicted example, memory **144** may provide data, such as previously compiled quantum algorithms, quantum logic gates, quantum error correction codes, boot files, operating system images, and applications to quantum processor **142**. Quantum processor **142** may include its own data, boot files, operating system images, and applications. Data processing environment **100** may include additional memories, quantum processors, and other devices that are not shown. Memory **144** includes application **105** that may be configured to implement one or more of the quantum processor functions described herein in accordance with one or more embodiments.

[0050] In the depicted example, data processing environment **100** may be the Internet. Network **102** may represent a collection of networks and gateways that use the Transmission Control Protocol/Internet Protocol (TCP/IP) and other protocols to communicate with one another. At the heart of the Internet is a backbone of data communication links between major nodes or host computers, including thousands of commercial, governmental, educational, and other computer systems that route data and messages. Of course, data processing environment **100** also may be implemented as a number of different types of networks, such as for example, an intranet, a local area network (LAN), or a wide area network (WAN). **Figure 1** is intended as an example, and not as an architectural limitation for the different illustrative embodiments.

[0051] Among other uses, data processing environment **100** may be used for implementing a client-server environment in which the illustrative embodiments may be implemented. A client-server environment enables software applications and data to be distributed across a network such that an application functions by using the interactivity between a conventional client data processing system and a conventional server data processing system. Data processing environment **100** may also employ a service oriented architecture where interoperable software components distributed across a network may be packaged together as coherent business applications. Data processing environment **100** may also take the form of a cloud, and employ a cloud computing model of service delivery for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, network bandwidth, servers, processing, memory, storage, applications, virtual machines, and services) that can be rapidly provisioned and released with minimal management effort or interaction with a provider of the service.

[0052] With reference to **Figure 2**, this figure depicts a block diagram of a data processing system in which illustrative embodiments may be implemented**.** Data processing system **200** is an example of a computer, such as servers **104** and **106**, or clients **110**, **112**, and **114** in **Figure 1**, or another type of device in which computer usable program code or instructions implementing the processes may be located for the illustrative embodiments.

[0053] Data processing system **200** is also representative of a data processing system or a configuration therein, such as data processing system **132** in **Figure 1** in which computer usable program code or instructions implementing the processes of the illustrative embodiments may be located. Data processing system **200** is described as a computer only as an example, without being limited thereto. Implementations in the form of other devices, such as device **132** in **Figure 1**, may modify data processing system **200**, such as by adding a touch interface, and even eliminate certain depicted components from data processing system **200** without departing from the general description of the operations and functions of data processing system **200** described herein.

[0054] In the depicted example, data processing system **200** employs a hub architecture including North Bridge and memory controller hub (NB/MCH) **202** and South Bridge and input/output (I/O) controller hub (SB/ICH) **204**. Processing unit **206**, main memory **208**, and graphics processor **210** are coupled to North Bridge and memory controller hub (NB/MCH) **202**. Processing unit **206** may contain one or more processors and may be implemented using one or more heterogeneous processor systems. Processing unit **206** may be a multi-core processor. Graphics processor **210** may be coupled to NB/MCH **202** through an accelerated graphics port (AGP) in certain implementations.

[0055] In the depicted example, local area network (LAN) adapter **212** is coupled to South Bridge and I/O controller hub (SB/ICH) **204**. Audio adapter **216**, keyboard and mouse adapter **220**, modem **222**, read only memory (ROM) **224**, universal serial bus (USB) and other ports **232**, and PCI/PCIe devices **234** are coupled to South Bridge and I/O controller hub **204** through bus **238**. Hard disk drive (HDD) or solid-state drive (SSD) **226** and CD-ROM **230** are coupled to South Bridge and I/O controller hub **204** through bus **240**. PCI/PCIe devices **234** may include,

for example, Ethernet adapters, add-in cards, and PC cards for notebook computers. PCI uses a card bus controller, while PCIe does not. ROM **224** may be, for example, a flash binary input/output system (BIOS). Hard disk drive **226** and CD-ROM **230** may use, for example, an integrated drive electronics (IDE), serial advanced technology attachment (SATA) interface, or variants such as external-SATA (eSATA) and micro- SATA (mSATA). A super I/O (SIO) device **236** may be coupled to South Bridge and I/O controller hub (SB/ICH) **204** through bus **238**.

[0056] Memories, such as main memory **208,** ROM **224**, or flash memory (not shown), are some examples of computer usable storage devices. Hard disk drive or solid state drive **226**, CD-ROM **230**, and other similarly usable devices are some examples of computer usable storage devices including a computer usable storage medium.

[0057] An operating system runs on processing unit **206**. The operating system coordinates and provides control of various components within data processing system **200** in **Figure 2**. The operating system may be a commercially available operating system for any type of computing platform, including but not limited to server systems, personal computers, and mobile devices. An object oriented or other type of programming system may operate in conjunction with the operating system and provide calls to the operating system from programs or applications executing on data processing system **200**.

[0058] Instructions for the operating system, the object-oriented programming system, and applications or programs, such as application **105** in **Figure 1**, are located on storage devices, such as in the form of code **226A** on hard disk drive **226**, and may be loaded into at least one of one or more memories, such as main memory **208**, for execution by processing unit **206**. The processes of the illustrative embodiments may be performed by processing unit **206** using computer implemented instructions, which may be located in a memory, such as, for example, main memory **208**, read only memory **224**, or in one or more peripheral devices.

[0059] Furthermore, in one case, code **226A** may be downloaded over network **201A** from remote system **201B**, where similar code **201C** is stored on a storage device **201D**. in another case, code **226A** may be downloaded over network **201A** to remote system **201B**, where downloaded code **201C** is stored on a storage device **201D**.

[0060] The hardware in **Figures 1-2** may vary depending on the implementation. Other internal hardware or peripheral devices, such as flash memory, equivalent non-volatile memory, or optical disk drives and the like, may be used in addition to or in place of the hardware depicted in **Figures 1-2**. In addition, the processes of the illustrative embodiments may be applied to a multiprocessor data processing system.

[0061] In some illustrative examples, data processing system **200** may be a personal digital assistant (PDA), which is generally configured with flash memory to provide non-volatile memory for storing operating system files and/or user-generated data. A bus system may comprise one or more buses, such as a system bus, an I/O bus, and a PCI bus. Of course, the bus system may be implemented using any type of communications fabric or

architecture that provides for a transfer of data between different components or devices attached to the fabric or architecture.

[0062] A communications unit may include one or more devices used to transmit and receive data, such as a modem or a network adapter. A memory may be, for example, main memory **208** or a cache, such as the cache found in North Bridge and memory controller hub **202**. A processing unit may include one or more processors or CPUs.

[0063] The depicted examples in **Figures 1-2** and above-described examples are not meant to imply architectural limitations. For example, data processing system **200** also may be a tablet computer, laptop computer, or telephone device in addition to taking the form of a mobile or wearable device.

[0064] Where a computer or data processing system is described as a virtual machine, a virtual device, or a virtual component, the virtual machine, virtual device, or the virtual component operates in the manner of data processing system **200** using virtualized manifestation of some or all components depicted in data processing system **200**. For example, in a virtual machine, virtual device, or virtual component, processing unit **206** is manifested as a virtualized instance of all or some number of hardware processing units **206** available in a host data processing system, main memory **208** is manifested as a virtualized instance of all or some portion of main memory **208** that may be available in the host data processing system, and disk **226** is manifested as a virtualized instance of all or some portion of disk **226** that may be available in the host data processing system. The host data processing system in such cases is represented by data processing system **200**.

[0065] With reference to **Figure 3**, this figure depicts an example configuration of adaptive error correction in quantum computing in accordance with an illustrative embodiment. The example embodiment includes an application **302**. In a particular embodiment, application **302** is an example of application **105** in **Figure 1**. Application **302** includes a calibration component **304** and a quantum circuit construction component **306**. Calibration component **304** calibrates an input quantum processor **308** in accordance with an example method described herein. Quantum circuit construction component **306** compiles an output quantum circuit design **310** in accordance with an example method described herein.

[0066] With reference to **Figure 4**, this figure depicts an example configuration for calibration of a quantum processor in accordance with an illustrative embodiment. The example embodiment includes an application **402**. In a particular embodiment, application **402** is an example of application **105** in **Figure 1**.

[0067] Application **402** includes calibration component **404**. In a particular embodiment, component **404** is an example of component **304** in **Figure 3**. Component **404** includes calibration operator component **406**, qubit parameter analysis component **408**, and quantum gate parameter analysis component **410**. Application **402** receives a calibration operation **412**.

[0068] Calibration operator **406** executes calibration operation **412**. In an embodiment, calibration operation **412** performs a set of operations a plurality of qubits Q1, Q2, Q3, . . ., Qn of the quantum processor **414.** In an embodiment, calibration operation **412** performs a method of randomized benchmarking. For example, calibration operation **412** can perform a set of pre-determined operations on a plurality of qubits of the quantum processor **414**. The set of pre-determined operations generate a set of values for each qubit in response to performing the set of pre-determined operations. In an embodiment, calibration operator **412** compares the set of values for each qubit to an expected answer of at least one of the set of pre-determined operations.

[0069] In an embodiment, calibration operation **412** returns a set of qubit parameter values **416** for the plurality of qubits of the quantum processor **414.** For example, qubit coherence time, qubit relaxation time, measurement error, and other qubit parameter values can be determined by the calibration operation. Each qubit of the quantum processor **414** can include a subset of the set of parameter values. For example, qubit Q1 can include associated parameter values P1, P2, . . ., Pn, etc. These examples of qubit parameter values are not intended to be limiting. From this disclosure, those of ordinary skill in the art will be able to conceive of many other qubit parameter values suitable for calibrating a set of qubits and the same are contemplated within the scope of the illustrative embodiments.

[0070] In an embodiment, calibration operation **412** returns a set of quantum gate parameters. For example, calibration operation **412** can return a parameter corresponding to an error rate for each quantum gate in the quantum processor **414**. In an embodiment, calibration operation **412** returns a parameter corresponding to an error rate for each one and two qubit gate (primitive gate) in the quantum processor **414.**

[0071] Component **408** analyzes the set of qubit parameter values **416**. In an embodiment, component **408** analyzes the set of qubit parameter values **416** according to at least one of a set of qubit acceptability criteria **418**. For example, component **408** can compare a parameter value of a qubit to a qubit acceptability criterion. For example, component **408** can determine a coherence time of a qubit fails to satisfy a threshold coherence time to perform a set of operations. As another example, component **408** can determine a coherence time of another qubit meets a threshold coherence time to perform the set of operations.

[0072] Component **410** analyzes a set of quantum gate parameters. In an embodiment, quantum gate parameters correspond to the set of qubits forming the quantum gate and the layout of the qubits on the quantum processor. In an embodiment, calibration operation **412** returns a set of quantum gate parameters values for a plurality of quantum gates of the quantum processor **414.** For example, gate error rates, gate speeds, gate cross talk matrix, and other quantum gate parameter values can be determined by the calibration operation. Each quantum gate of the quantum processor **414** can include a subset of the set of quantum gate parameter values. These examples of quantum gate parameter are not intended to be limiting. From this disclosure, those of ordinary skill in the art will be able to conceive of many other quantum gate parameter values suitable for calibrating a set of quantum gates and the same are contemplated within the scope of the illustrative embodiments.

[0073] Component **410** analyzes the set of quantum gate parameter values. In an embodiment, component **410** analyzes the set of quantum gate parameter values according to at least one of a set of quantum gate acceptability criteria **420**. For example, component **410** can compare a parameter value of a quantum gate to a quantum gate acceptability criterion. For example, component **410** can determine a gate error rate of a quantum gate fails to satisfy a threshold error rate to perform the quantum gate.

[0074] In an embodiment, component **410** generates a composite gate error rate for a composite gate from a set of primitive gate error rates, the composite gate formed from a set of primitive gates corresponding to the primitive gate error rates. In an embodiment, component **410** generates a composite gate error rate for a composite gate formed using three qubits from the formula

$\alpha = \frac{A_{123}B_{12}B_{13}B_{23}}{21}\left(\left(A_{23}B_{23}^{-1} + A_{13}B_{13}^{-1} + A_{12}B_{12}^{-1} + \frac{A_1+A_2+A_3}{3} + 9\right)\right)$, where A_{xy} and B_{xy} are primitive gate error rates for a quantum gate formed from qubits x and y during a first calibration operation and a second calibration operation, respectively.

[0075] With reference to **Figure 5**, this figure depicts an example configuration for quantum circuit construction in accordance with an illustrative embodiment. The example embodiment includes an application **502**. In a particular embodiment, application **502** is an example of application **105** in **Figure 1**.

[0076] Application **502** includes quantum circuit construction component **504**. In a particular embodiment, component **504** is an example of component **306** in **Figure 3.** Component **504** includes a quantum circuit compiler component **506**. Component **506** is configured to transform an input quantum algorithm **514** into a quantum circuit design **530**. Component **516** includes an algorithm transformation component **508**.

[0077] Component **508** transforms the quantum algorithm code into a first quantum circuit design corresponding to the operations performed by the quantum algorithm. Application **502** receives a set of quantum circuit designs **528** stored in database **524**. In particular embodiments, each quantum circuit design **528** includes a previously determined runtime estimation. In particular embodiments, the runtime estimation for each of the stored set of quantum circuit designs is determined from previously executing each of the set of quantum circuit designs on a quantum processor. Database **524** is an example of storage unit **108** in **Figure 1**. In an embodiment, runtime estimation component **510** determines a runtime for the quantum algorithm **522**. In an embodiment, component **510** compares the quantum algorithm **522** to a set of quantum algorithms corresponding to the set of quantum circuit designs **528**. In particular embodiments, component **510** compares the total number of quantum logic gates in the quantum circuit design **530** and the total number of quantum logic gates in at least one of the stored set of quantum circuit designs.

[0078] Component **510** includes circuit depth component **512** and gate weight factor **514**. In an embodiment, circuit depth component **514** determines a depth of the quantum circuit design **530**. In an embodiment, gate weight

factor **514** assigns a set of weights to a set of quantum logic gates in the quantum circuit design **530**. In particular embodiments, a subset of the set of quantum logic gates with comparatively higher error rates have greater weight factors than quantum logic gates with lower error rates.

[0079] Error scenario component **516** determines a worst case error scenario for the quantum circuit design **530**. In an embodiment, circuit comparison component **518** analyzes the quantum circuit design **530** to determine the number of quantum logic gates, the type of quantum logic gates, and the state of the quantum processor configured to execute the quantum circuit design **530**. In particular embodiments, circuit comparison component **518** determines decoherence times for a set of qubits of the quantum processor. In particular embodiments, each of the set of quantum circuit designs **528** includes a state of the quantum processor determined before executing the corresponding quantum circuit design.

[0080] In an embodiment, circuit comparison component **518** compares the quantum circuit design **530** and the state of the quantum processor to the set of quantum logic gates and the corresponding set of states of the quantum processor. In an embodiment, component **518** selects a similar quantum circuit design to the quantum circuit design **530** from the set of quantum circuit designs **528**. In particular embodiments, component **518** compares the total number of quantum logic gates, the type of quantum logic gates, and the initial state of the quantum processor between the quantum circuit designs.

[0081] In an embodiment, error correction selection component **520** analyzes the quantum circuit design **530**, the runtime estimation, and the state of the quantum processor to select an appropriate error correction approach. In an embodiment, component **520** analyzes the number of qubits necessary to execute the quantum circuit design **530**. In an embodiment, component **520** analyzes the total number of qubits available for performing computations on the quantum processor. In particular embodiments, component **520** determines a remaining set of qubits available for quantum error correction on the quantum processor. For example, component **520** can determine five qubits are necessary to execute the quantum circuit design **530**. Component **520** can determine eight total qubits are available for computations on the quantum processor. Component **520** can determine a remaining three qubits are available for quantum error correction.

[0082] In an embodiment, component **520** selects an appropriate quantum error correction technique in response to determining the subset of the set of qubits on the quantum processor available for quantum error correction. In particular embodiments, component **520** selects a quantum error correction technique which produces the highest level of accuracy while also fitting the available qubits for quantum error correction. In particular embodiments, component **520** selects a quantum error correction technique based on the type of quantum logic gates in the quantum circuit design **530**.

[0083] Probability comparison component **520** determines a number of iterations of performing the quantum circuit design **530** on the quantum processor necessary to reach a threshold level of accuracy. For example,

component **520** can determine 120 iterations are necessary to produce a 99% level of accuracy. Component **520** determines a runtime for the selected quantum error correction approach. Component **520** compares the runtime for the selected quantum error correction approach to a runtime for executing additional iterations to reach a threshold level of accuracy. For example, component **520** can determine performing additional iterations produces a similar level of accuracy in less time than the selected quantum error correction approach. Application **502** outputs the quantum circuit design **530** with a corresponding selected quantum error correction approach **532**.

[0084] With reference to **Figure 6**, this figure depicts a flowchart of an example method for adaptive error correction in quantum computing in accordance with an illustrative embodiment. Application **105** performs method **600** in an embodiment.

[0085] In block **602**, application **105** determines an initial state of a quantum processor. In an embodiment, application **105** executes a calibration operation on a set of qubits in a first iteration. In an embodiment, the calibration operation produces a set of parameters, a first subset of the set of parameters corresponding to a first qubit of the set of qubits, and a second subset of the set of parameters corresponding to a second qubit of the set of qubits. In an embodiment, application **105** executes a pre-determined operation on the first qubit. In an embodiment, application **105** compares an output of the pre-determined operation to an expected output.

[0086] In block **604**, application **105** estimates a runtime duration for a quantum circuit design corresponding to a quantum algorithm. In block **606**, application **105** computes an error scenario for the quantum circuit design corresponding to the quantum algorithm. In block **608**, application **105** selects an appropriate error correction approach for the quantum circuit design corresponding to the quantum algorithm. In block **610**, application **105** transforms the quantum algorithm into the quantum circuit design, the quantum circuit design including a set of quantum logic gates. In an embodiment, a subset of the set of quantum logic gates corresponds to the selected error correction approach. Application **105** ends process **600** thereafter.

[0087] With reference to **Figure 7**, this figure depicts a flowchart of an example method for estimating runtime of a quantum circuit design corresponding to a quantum algorithm in accordance with an illustrative embodiment. Application **105** performs method **700** at block **604** in an embodiment. In block **702**, application **105** determines a depth of the quantum circuit corresponding to the quantum algorithm. In block **704**, application **105** assigns a set of weights to a set of quantum logic gates in the quantum circuit design. Application **105** ends process **700** thereafter.

[0088] With reference to **Figure 8**, this figure depicts a flowchart of an example method for computing an error scenario in accordance with an illustrative embodiment. Application **105** performs method **800** at block **606** in an embodiment. In block **802**, application **105** assigns a set of weights to a set of error scenarios. In block **804**, application **105** compares the quantum circuit design to a set of previously executed quantum circuit designs stored in a database. Application **105** ends process **800** thereafter.

[0089] With reference to **Figure 9**, this figure depicts a flowchart of an example method for selecting an error correction approach in accordance with an illustrative embodiment. Application **105** performs method **900** at block **608** in an embodiment. In block **902**, application **105** determines a first computation (evaluation) cost for a selected error correction approach. In block **904**, application **105** determines a second computation (evaluation) cost for a probabilistic error correction approach. In block **906**, application **105** compares the first computation cost and the second computation cost. In an embodiment, application **105** chooses the error correction approach with a lower computation cost. Application **105** ends process **900** thereafter.

[0090] Various embodiments of the present invention are described herein with reference to the related drawings. Alternative embodiments can be devised without departing from the scope of this invention. Although various connections and positional relationships (e.g., over, below, adjacent, etc.) are set forth between elements in the following description and in the drawings, persons skilled in the art will recognize that many of the positional relationships described herein are orientation-independent when the described functionality is maintained even though the orientation is changed. These connections and/or positional relationships, unless specified otherwise, can be direct or indirect, and the present invention is not intended to be limiting in this respect. Accordingly, a coupling of entities can refer to either a direct or an indirect coupling, and a positional relationship between entities can be a direct or indirect positional relationship. As an example of an indirect positional relationship, references in the present description to forming layer "A" over layer "B" include situations in which one or more intermediate layers (e.g., layer "C") is between layer "A" and layer "B" as long as the relevant characteristics and functionalities of layer "A" and layer "B" are not substantially changed by the intermediate layer(s).

[0091] The following definitions and abbreviations are to be used for the interpretation of the claims and the specification. As used herein, the terms "comprises," "comprising," "includes," "including," "has," "having," "contains" or "containing," or any other variation thereof, are intended to cover a non-exclusive inclusion. For example, a composition, a mixture, process, method, article, or apparatus that comprises a list of elements is not necessarily limited to only those elements but can include other elements not expressly listed or inherent to such composition, mixture, process, method, article, or apparatus.

[0092] Additionally, the term "illustrative" is used herein to mean "serving as an example, instance or illustration." Any embodiment or design described herein as "illustrative" is not necessarily to be construed as preferred or advantageous over other embodiments or designs. The terms "at least one" and "one or more" are understood to include any integer number greater than or equal to one, i.e. one, two, three, four, etc. The terms "a plurality" are understood to include any integer number greater than or equal to two, i.e. two, three, four, five, etc. The term "connection" can include an indirect "connection" and a direct "connection."

[0093] References in the specification to "one embodiment," "an embodiment," "an example embodiment," etc., indicate that the embodiment described can include a particular feature, structure, or characteristic, but every embodiment may or may not include the particular feature, structure, or characteristic. Moreover, such phrases are

not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the art to affect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

[0094] The terms "about," "substantially," "approximately," and variations thereof, are intended to include the degree of error associated with measurement of the particular quantity based upon the equipment available at the time of filing the application. For example, "about" can include a range of ± 8% or 5%, or 2% of a given value.

[0095] The descriptions of the various embodiments of the present invention have been presented for purposes of illustration, but are not intended to be exhaustive or limited to the embodiments disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope of the described embodiments. The terminology used herein was chosen to best explain the principles of the embodiments, the practical application or technical improvement over technologies found in the marketplace, or to enable others of ordinary skill in the art to understand the embodiments described herein.

[0096] The descriptions of the various embodiments of the present invention have been presented for purposes of illustration, but are not intended to be exhaustive or limited to the embodiments disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art without departing from the scope of the described embodiments. The terminology used herein was chosen to best explain the principles of the embodiments, the practical application or technical improvement over technologies found in the marketplace, or to enable others of ordinary skill in the art to understand the embodiments described herein.

[0097] Thus, a computer implemented method, system or apparatus, and computer program product are provided in the illustrative embodiments for managing participation in online communities and other related features, functions, or operations. Where an embodiment or a portion thereof is described with respect to a type of device, the computer implemented method, system or apparatus, the computer program product, or a portion thereof, are adapted or configured for use with a suitable and comparable manifestation of that type of device.

[0098] Where an embodiment is described as implemented in an application, the delivery of the application in a Software as a Service (SaaS) model is contemplated within the scope of the illustrative embodiments. In a SaaS model, the capability of the application implementing an embodiment is provided to a user by executing the application in a cloud infrastructure. The user can access the application using a variety of client devices through a thin client interface such as a web browser (e.g., web-based e-mail), or other light-weight client-applications. The user does not manage or control the underlying cloud infrastructure including the network, servers, operating systems, or the storage of the cloud infrastructure. In some cases, the user may not even manage or control the capabilities of the SaaS application. In some other cases, the SaaS implementation of the application may permit a possible exception of limited user-specific application configuration settings.

[0099] The present invention may be a system, a method, and/or a computer program product at any possible technical detail level of integration. The computer program product may include a computer readable storage medium (or media) having computer readable program instructions thereon for causing a processor to carry out aspects of the present invention.

[0100] The computer readable storage medium can be a tangible device that can retain and store instructions for use by an instruction execution device. The computer readable storage medium may be, for example, but is not limited to, an electronic storage device, a magnetic storage device, an optical storage device, an electromagnetic storage device, a semiconductor storage device, or any suitable combination of the foregoing. A non-exhaustive list of more specific examples of the computer readable storage medium includes the following: a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), a static random access memory (SRAM), a portable compact disc read-only memory (CD-ROM), a digital versatile disk (DVD), a memory stick, a floppy disk, a mechanically encoded device such as punch-cards or raised structures in a groove having instructions recorded thereon, and any suitable combination of the foregoing. A computer readable storage medium, as used herein, is not to be construed as being transitory signals per se, such as radio waves or other freely propagating electromagnetic waves, electromagnetic waves propagating through a waveguide or other transmission media (e.g., light pulses passing through a fiber-optic cable), or electrical signals transmitted through a wire.

[0101] Computer readable program instructions described herein can be downloaded to respective computing/processing devices from a computer readable storage medium or to an external computer or external storage device via a network, for example, the Internet, a local area network, a wide area network and/or a wireless network. The network may comprise copper transmission cables, optical transmission fibers, wireless transmission, routers, firewalls, switches, gateway computers and/or edge servers. A network adapter card or network interface in each computing/processing device receives computer readable program instructions from the network and forwards the computer readable program instructions for storage in a computer readable storage medium within the respective computing/processing device.

[0102] Computer readable program instructions for carrying out operations of the present invention may be assembler instructions, instruction-set-architecture (ISA) instructions, machine instructions, machine dependent instructions, microcode, firmware instructions, state-setting data, configuration data for integrated circuitry, or either source code or object code written in any combination of one or more programming languages, including an object oriented programming language such as Smalltalk, C++, or the like, and procedural programming languages, such as the "C" programming language or similar programming languages. The computer readable program instructions may execute entirely on the user's computer, partly on the user's computer, as a stand-alone software package, partly on the user's computer and partly on a remote computer or entirely on the remote computer or server. In the latter scenario, the remote computer may be connected to the user's computer through any type of network,

including a local area network (LAN) or a wide area network (WAN), or the connection may be made to an external computer (for example, through the Internet using an Internet Service Provider). In some embodiments, electronic circuitry including, for example, programmable logic circuitry, field-programmable gate arrays (FPGA), or programmable logic arrays (PLA) may execute the computer readable program instructions by utilizing state information of the computer readable program instructions to personalize the electronic circuitry, in order to perform aspects of the present invention.

[0103] Aspects of the present invention are described herein with reference to flowchart illustrations and/or block diagrams of methods, apparatus (systems), and computer program products according to embodiments of the invention. It will be understood that each block of the flowchart illustrations and/or block diagrams, and combinations of blocks in the flowchart illustrations and/or block diagrams, can be implemented by computer readable program instructions.

[0104] These computer readable program instructions may be provided to a processor of a general purpose computer, special purpose computer, or other programmable data processing apparatus to produce a machine, such that the instructions, which execute via the processor of the computer or other programmable data processing apparatus, create means for implementing the functions/acts specified in the flowchart and/or block diagram block or blocks. These computer readable program instructions may also be stored in a computer readable storage medium that can direct a computer, a programmable data processing apparatus, and/or other devices to function in a particular manner, such that the computer readable storage medium having instructions stored therein comprises an article of manufacture including instructions which implement aspects of the function/act specified in the flowchart and/or block diagram block or blocks.

[0105] The computer readable program instructions may also be loaded onto a computer, other programmable data processing apparatus, or other device to cause a series of operational steps to be performed on the computer, other programmable apparatus or other device to produce a computer implemented process, such that the instructions which execute on the computer, other programmable apparatus, or other device implement the functions/acts specified in the flowchart and/or block diagram block or blocks.

[0106] The flowchart and block diagrams in the Figures illustrate the architecture, functionality, and operation of possible implementations of systems, methods, and computer program products according to various embodiments of the present invention. In this regard, each block in the flowchart or block diagrams may represent a module, segment, or portion of instructions, which comprises one or more executable instructions for implementing the specified logical function(s). In some alternative implementations, the functions noted in the blocks may occur out of the order noted in the Figures. For example, two blocks shown in succession may, in fact, be executed substantially concurrently, or the blocks may sometimes be executed in the reverse order, depending upon the functionality involved. It will also be noted that each block of the block diagrams and/or flowchart illustration, and combinations of blocks in the block diagrams and/or flowchart illustration, can be implemented by special purpose

hardware-based systems that perform the specified functions or acts or carry out combinations of special purpose hardware and computer instructions.

CLAIMS

1. A method comprising:

executing a calibration operation on a set of qubits, the calibration operation determining an initial state of a quantum processor;

estimating, responsive to determining an initial state of the quantum processor, a runtime duration for a quantum circuit design corresponding to a quantum algorithm, the quantum processor configured to execute the quantum circuit design;

computing an error scenario for the quantum circuit design;

selecting, using the error scenario and the initial state of the quantum processor, a quantum error correction approach for the quantum circuit design; and

transforming the quantum algorithm into the quantum circuit design, the quantum circuit design including a set of quantum logic gates.

2. The method of claim 1, wherein a subset of the set of quantum logic gates corresponds to the selected quantum error correction approach.

3. The method of claim 1 or 2, wherein the initial state of the quantum processor includes a total number of qubits of the quantum processor.

4. The method of claim 1, 2 or 3, further comprising:
determining a depth of the quantum circuit design.

5. The method of any of claims 1 to 4, further comprising:
assigning a set of weights to the set of quantum logic gates.

6. The method of any of claims 1 to 5, further comprising:
comparing the quantum circuit design to a set of previously executed quantum circuit designs.

7. The method of any of claims 1 to 6, further comprising:
determining a first computation cost for the selected quantum error correction approach; and
determining a second computation cost for a probabilistic quantum error correction approach.

8. The method of any of claims 1 to 7, further comprising:
comparing the quantum circuit design to a set of previously executed quantum circuit designs;

selecting a quantum error correction approach for a first quantum circuit design of the set of previously executed quantum circuit designs; and

transforming the quantum algorithm into a new quantum circuit design, the new quantum circuit design including a set of quantum logic gates corresponding to the first quantum circuit design.

9. A computer usable program product comprising a computer-readable storage device, and program instructions stored on the storage device, the stored program instructions comprising:

program instructions to execute a calibration operation on a set of qubits, the calibration operation determining an initial state of a quantum processor;

program instructions to estimate, responsive to determining an initial state of the quantum processor, a runtime duration for a quantum circuit design corresponding to a quantum algorithm, the quantum processor configured to execute the quantum circuit design;

program, instructions to compute an error scenario for the quantum circuit design;

program instructions to select, using the error scenario and the initial state of the quantum processor, a quantum error correction approach for the quantum circuit design; and

program instructions to transform the quantum algorithm into the quantum circuit design, the quantum circuit design including a set of quantum logic gates.

10. The computer usable program product of claim 9, wherein the program instructions are stored in a computer readable storage device in a data processing system, and wherein the computer usable code is transferred over a network from a remote data processing system.

11. The computer usable program product of claim 9 or 10, wherein the program instructions are stored in a computer readable storage device in a server data processing system, and wherein the computer usable code is downloaded over a network to a remote data processing system for use in a computer readable storage device associated with the remote data processing system.

12. The computer usable program product of any of claims 9, 10 or 11, wherein a subset of the set of quantum logic gates corresponds to the selected quantum error correction approach.

13. The computer usable program product of any of claims 9 to 12, wherein the initial state of the quantum processor includes a total number of qubits of the quantum processor.

14. The computer usable program product of any of claims 9 to 13, further comprising:

program instructions to determine a depth of the quantum circuit design.

15. The computer usable program product of any of claims 9 to 14, further comprising:

program instructions to assign a set of weights to the set of quantum logic gates.

16. The computer usable program product of any of claims 9 to 15, further comprising:

program instructions to compare the quantum circuit design to a set of previously executed quantum circuit designs.

17. The computer usable program product of any of claims 9 to 16, further comprising:

program instructions to determine a first computation cost for the selected quantum error correction approach; and

program instructions to determine a first computation cost for the selected quantum error correction approach; and

program instructions to determine a second computation cost for a probabilistic quantum error correction approach.

18. The computer usable program product of claims 9 to 17, further comprising:

program instructions to compare the quantum circuit design to a set of previously executed quantum circuit designs;

program instructions to select a quantum error correction approach for a first quantum circuit design of the set of previously executed quantum circuit designs; and

program instructions to transform the quantum algorithm into a new quantum circuit design, the new quantum circuit design including a set of quantum logic gates corresponding to the first quantum circuit design.

19. A computer system comprising a processor, a computer-readable memory, and a computer-readable storage device, and program instructions stored on the storage device for execution by the processor via the memory, the stored program instructions comprising:

program instructions to execute a calibration operation on a set of qubits, the calibration operation determining an initial state of a quantum processor;

program instructions to estimate, responsive to determining an initial state of the quantum processor, a runtime duration for a quantum circuit design corresponding to a quantum algorithm, the quantum processor configured to execute the quantum circuit design;

program, instructions to compute an error scenario for the quantum circuit design;

program instructions to select, using the error scenario and the initial state of the quantum processor, a quantum error correction approach for the quantum circuit design; and

program instructions to transform the quantum algorithm into the quantum circuit design, the quantum circuit design including a set of quantum logic gates.

20. The computer system of claim 19, wherein a subset of the set of quantum logic gates corresponds to the selected quantum error correction approach.

FIGURE 1

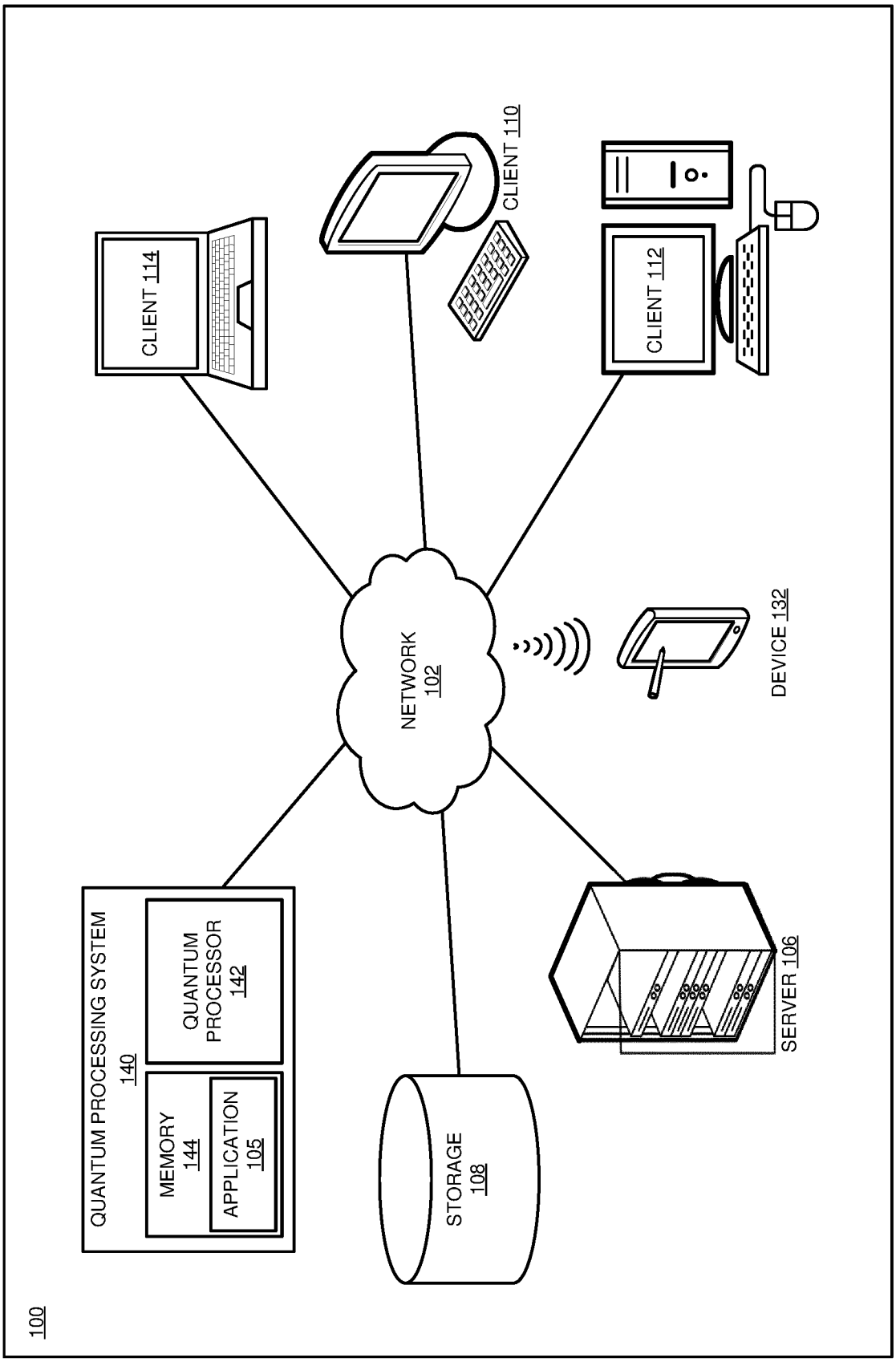
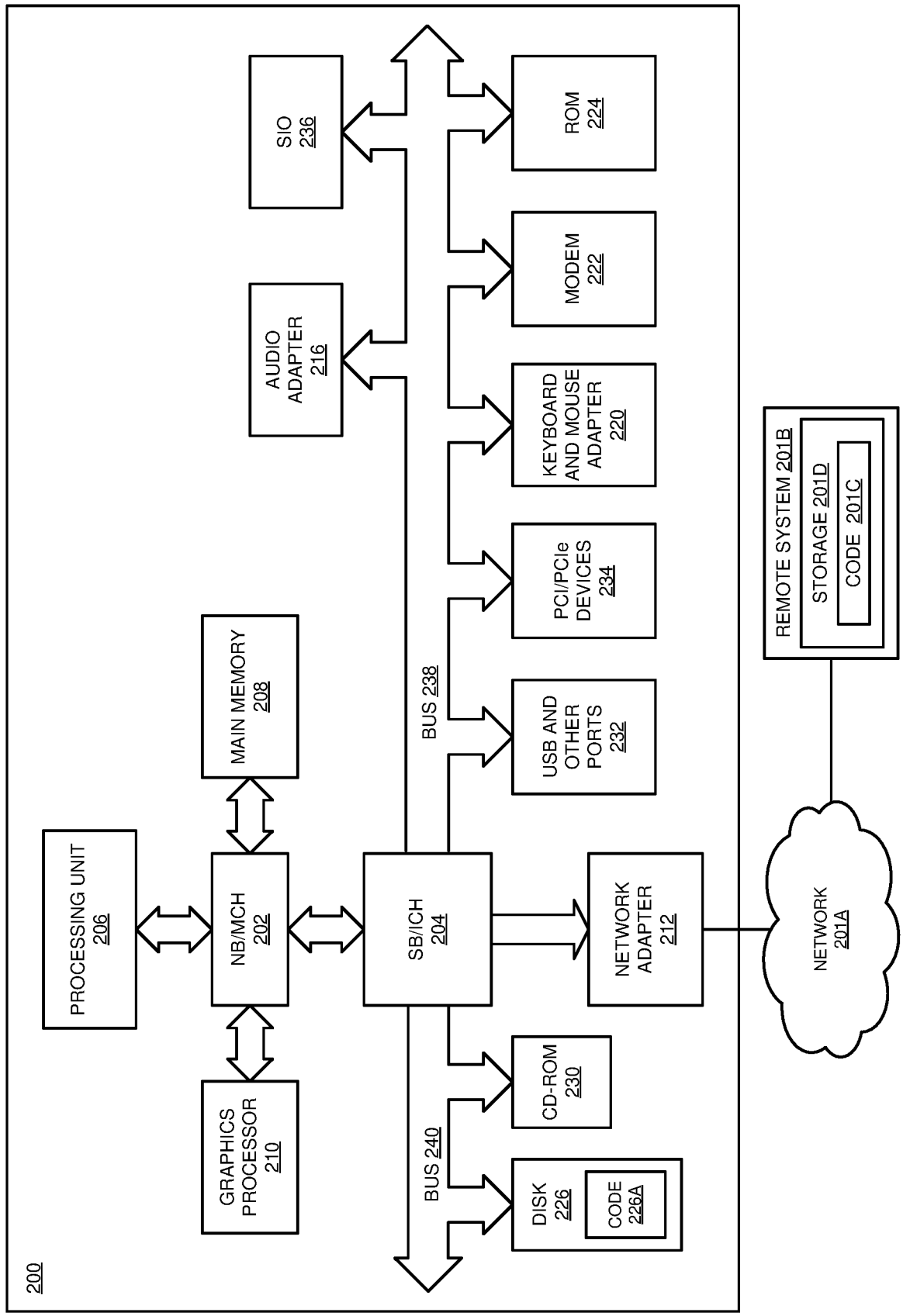

100
CLIENT 114
CLIENT 110
CLIENT 112
NETWORK 102
DEVICE 132
QUANTUM PROCESSING SYSTEM 140
QUANTUM PROCESSOR 142
MEMORY 144
APPLICATION 105
STORAGE 108
SERVER 106

FIGURE 2


200
PROCESSING UNIT 206
NB/MCH 202
MAIN MEMORY 208
GRAPHICS PROCESSOR 210
SB/ICH 204
BUS 238
BUS 240
SIO 236
AUDIO ADAPTER 216
ROM 224
MODEM 222
KEYBOARD AND MOUSE ADAPTER 220
PCI/PCIe DEVICES 234
USB AND OTHER PORTS 232
CD-ROM 230
DISK 226
CODE 226A
NETWORK ADAPTER 212
NETWORK 201A
REMOTE SYSTEM 201B
STORAGE 201D
CODE 201C

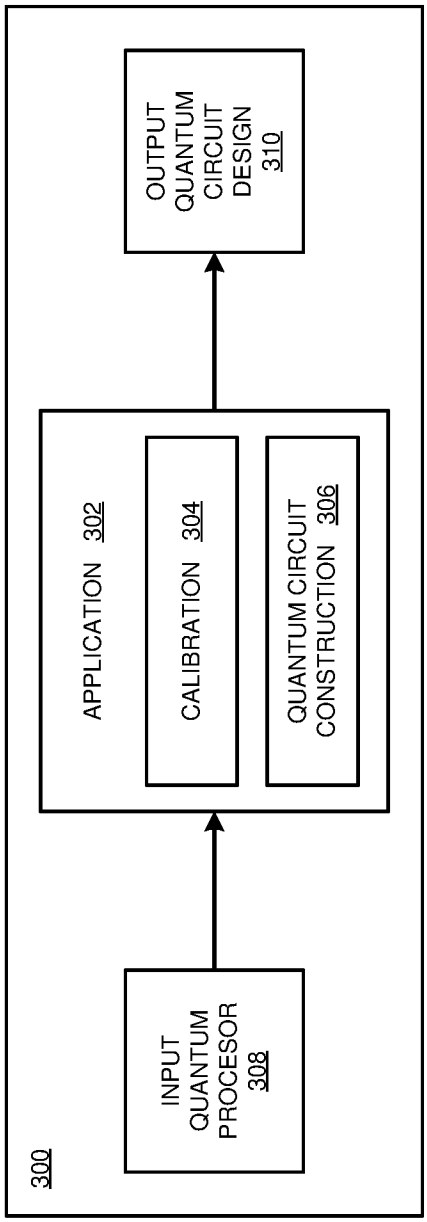

300
INPUT QUANTUM PROCESOR 308
APPLICATION 302
CALIBRATION 304
QUANTUM CIRCUIT CONSTRUCTION 306
OUTPUT QUANTUM CIRCUIT DESIGN 310

FIGURE 3

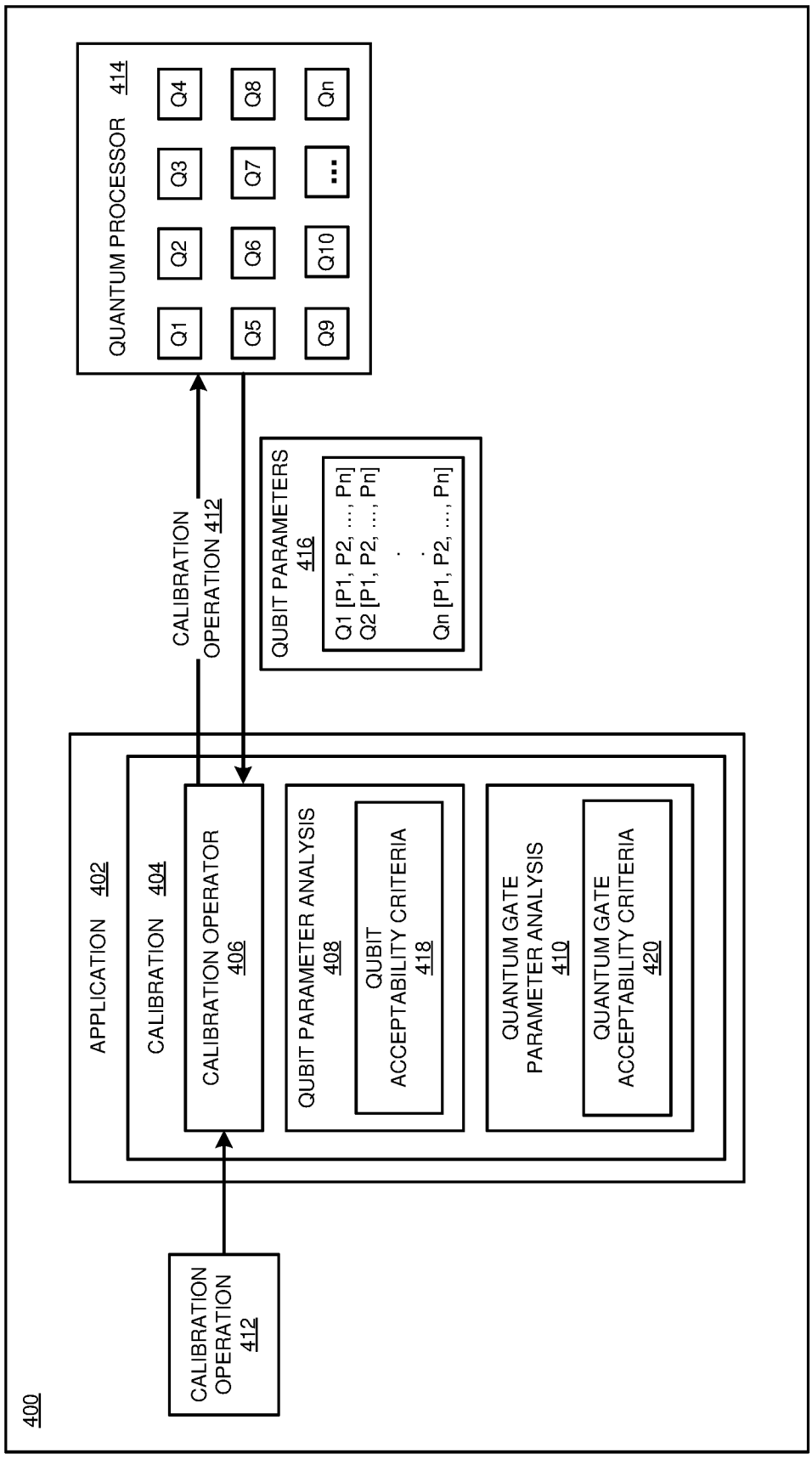
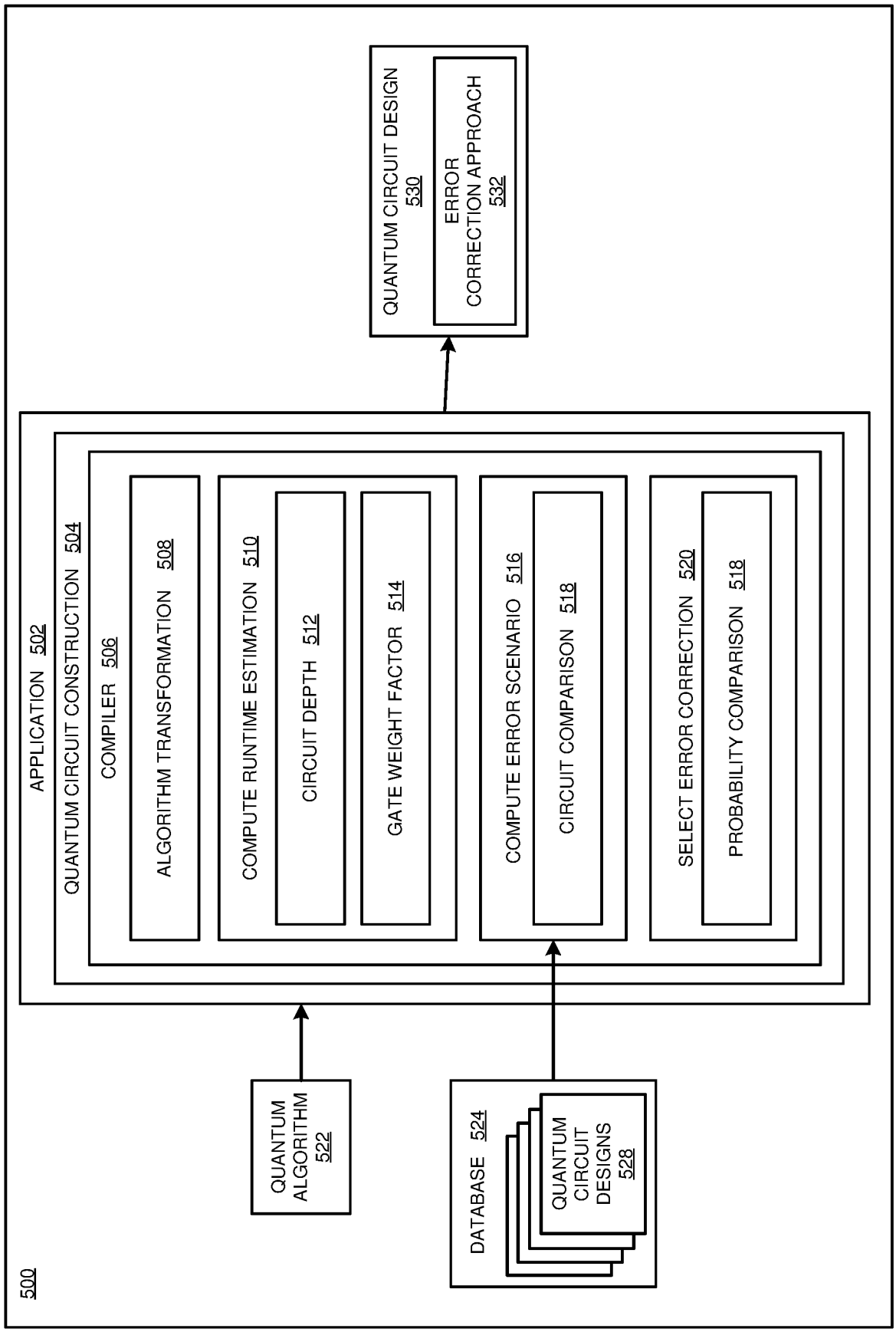

400
CALIBRATION OPERATION 412
APPLICATION 402
CALIBRATION 404
CALIBRATION OPERATOR 406
QUBIT PARAMETER ANALYSIS 408
QUBIT ACCEPTABILITY CRITERIA 418
QUANTUM GATE PARAMETER ANALYSIS 410
QUANTUM GATE ACCEPTABILITY CRITERIA 420
CALIBRATION OPERATION 412
QUBIT PARAMETERS 416
Q1 [P1, P2, ..., Pn]
Q2 [P1, P2, ..., Pn]
Qn [P1, P2, ..., Pn]
QUANTUM PROCESSOR 414
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q8
Q9
Q10
...
Qn

FIGURE 4

FIGURE 5


500
QUANTUM ALGORITHM 522
DATABASE 524
QUANTUM CIRCUIT DESIGNS 528
APPLICATION 502
QUANTUM CIRCUIT CONSTRUCTION 504
COMPILER 506
ALGORITHM TRANSFORMATION 508
COMPUTE RUNTIME ESTIMATION 510
CIRCUIT DEPTH 512
GATE WEIGHT FACTOR 514
COMPUTE ERROR SCENARIO 516
CIRCUIT COMPARISON 518
SELECT ERROR CORRECTION 520
PROBABILITY COMPARISON 518
QUANTUM CIRCUIT DESIGN 530
ERROR CORRECTION APPROACH 532

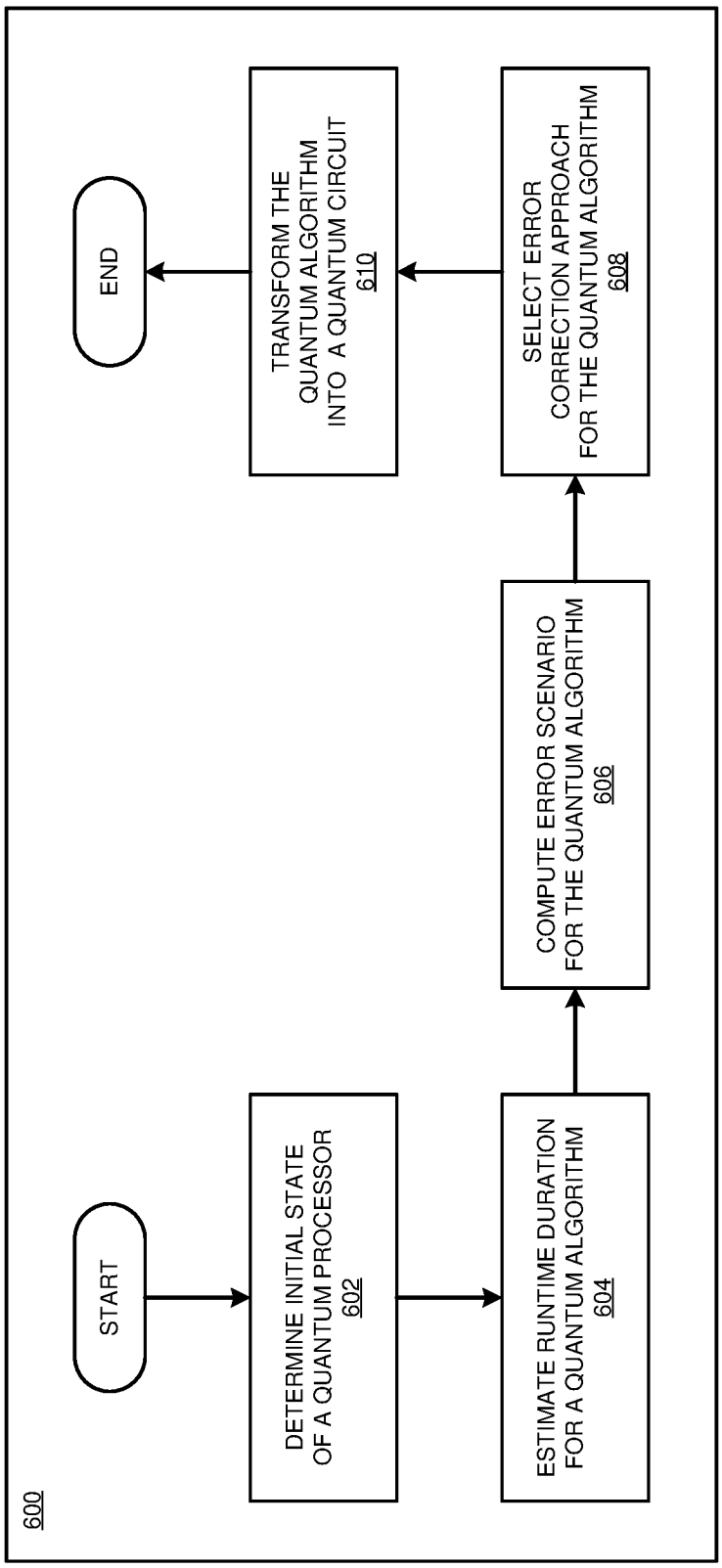

600
START
DETERMINE INITIAL STATE OF A QUANTUM PROCESSOR 602
ESTIMATE RUNTIME DURATION FOR A QUANTUM ALGORITHM 604
COMPUTE ERROR SCENARIO FOR THE QUANTUM ALGORITHM 606
SELECT ERROR CORRECTION APPROACH FOR THE QUANTUM ALGORITHM 608
TRANSFORM THE QUANTUM ALGORITHM INTO A QUANTUM CIRCUIT 610
END

FIGURE 6

FIGURE 7

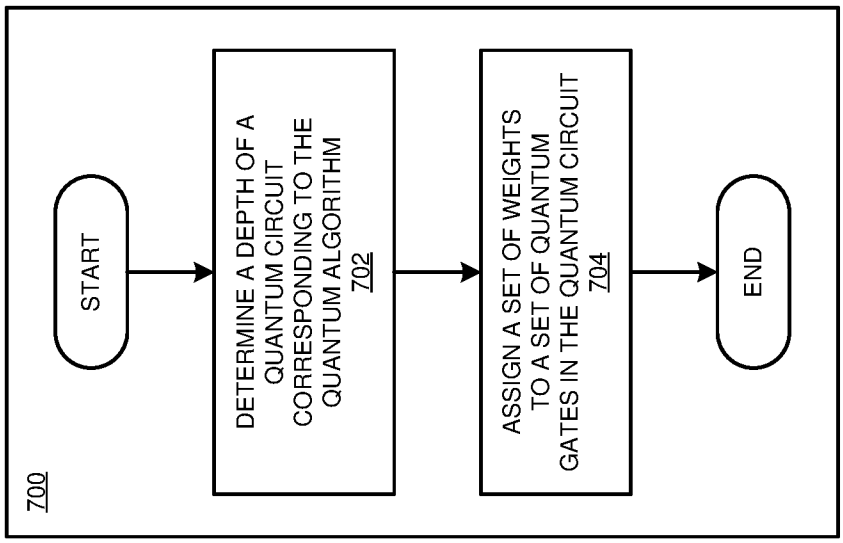

700
START
DETERMINE A DEPTH OF A QUANTUM CIRCUIT CORRESPONDING TO THE QUANTUM ALGORITHM
702
ASSIGN A SET OF WEIGHTS TO A SET OF QUANTUM GATES IN THE QUANTUM CIRCUIT
704
END

FIGURE 8

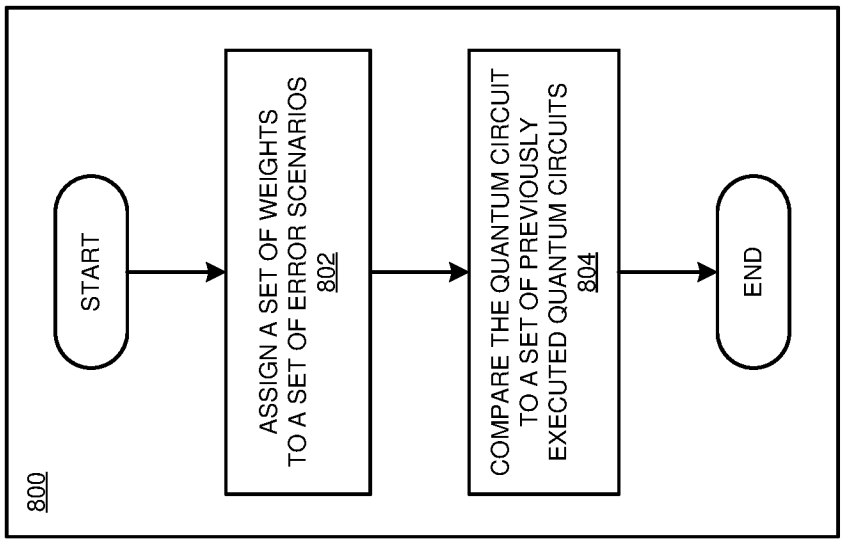
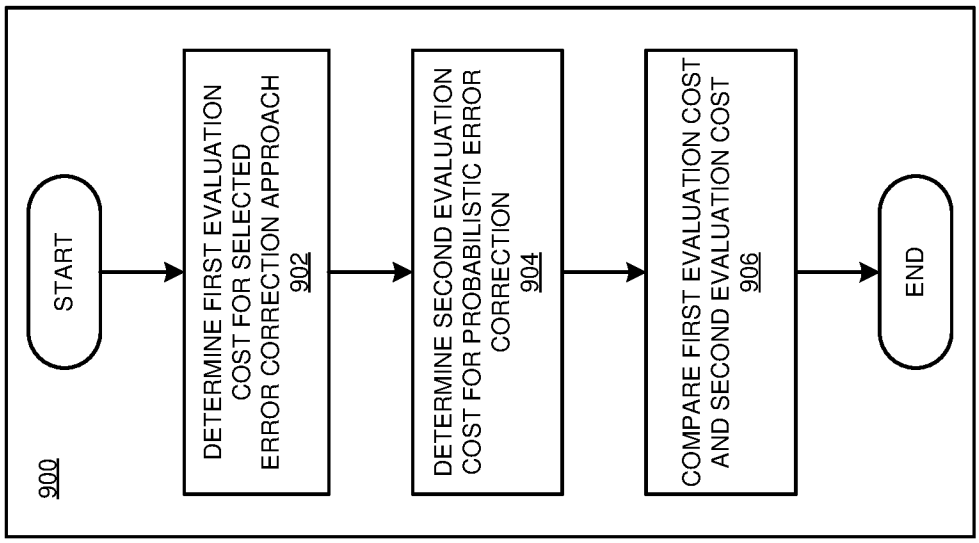

800
START
ASSIGN A SET OF WEIGHTS TO A SET OF ERROR SCENARIOS
802
COMPARE THE QUANTUM CIRCUIT TO A SET OF PREVIOUSLY EXECUTED QUANTUM CIRCUITS
804
END

FIGURE 9


900
START
DETERMINE FIRST EVALUATION COST FOR SELECTED ERROR CORRECTION APPROACH
902
DETERMINE SECOND EVALUATION COST FOR PROBABILISTIC ERROR CORRECTION
904
COMPARE FIRST EVALUATION COST AND SECOND EVALUATION COST
906
END

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2020/057239

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06N10/00
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06N

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	SPITZ S T ET AL: "Adaptive weight estimator for quantum error correction", ARXIV.ORG, CORNELL UNIVERSITY LIBRARY, 201 OLIN LIBRARY CORNELL UNIVERSITY ITHACA, NY 14853, 6 December 2017 (2017-12-06), XP081302837, DOI: 10.1002/QUTE.201800012	1-3,6,7, 9-13,16, 17,19,20
Y	abstract; figures 1-4 page 1, left-hand column, line 1 - page 5, right-hand column, last line	4,5,8, 14,15,18

-/--

[X] Further documents are listed in the continuation of Box C.

[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
29 May 2020	09/06/2020

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Totir, Felix

1

Form PCT/ISA/210 (second sheet) (April 2005)

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	ANDREW W CROSS ET AL: "Validating quantum computers using randomized model circuits", ARXIV.ORG, CORNELL UNIVERSITY LIBRARY, 201 OLIN LIBRARY CORNELL UNIVERSITY ITHACA, NY 14853, 30 November 2018 (2018-11-30), XP081042610, abstract; figures 1-3 page 1, left-hand column, line 1 - page 5, right-hand column, last line page 6, left-hand column, last line - page 11, last line	4,5,8, 14,15,18
A	WO 2019/058161 A1 (IBM [US]; IBM UK [GB]; IBM CHINA INVESTMENT CO LTD [CN]) 28 March 2019 (2019-03-28) abstract; figures 1-7,10-11B paragraph [0003] paragraph [0040] - paragraph [0061] paragraph [0070] - paragraph [0082]	1-20
A	BARBERIS-BLOSTEIN P ET AL: "From quantum feedback to probabilistic error correction: Manipulation of quantum beats in cavity QED", ARXIV.ORG, CORNELL UNIVERSITY LIBRARY, 201 OLIN LIBRARY CORNELL UNIVERSITY ITHACA, NY 14853, 6 January 2010 (2010-01-06), XP080389362, DOI: 10.1088/1367-2630/12/2/023002 abstract; figures 1-4 page 1, line 1 - page 14, last line	1-20

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/EP2020/057239

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 2019058161 A1	28-03-2019	CN 111052122 A	21-04-2020
		DE 112017007826 T5	23-04-2020
		GB 2579008 A	03-06-2020
		US 2019095561 A1	28-03-2019
		WO 2019058161 A1	28-03-2019