Claims:

1. A virtual quantum computer system, comprising:

a main memory (610);

one or more memory bus systems (650) coupled to the main memory (610);

one or more physical processing units (620) that have access to the main memory (610), preferably via at least one memory bus system;

a data processing unit (640) coupled to at least one of the one or more physical processing units (620), the data processing unit (640) serving as a bridge between internal systems of the virtual quantum computer system and external systems;

one or more cache coherency interconnects (630) connecting the one or more physical processing units (620); and

an information process stack that comprises:

a hardware layer, an operating system coupled to the hardware layer and a container environment coupled to the operating system;

wherein the information process stack is configured to:

(A) initialize qubits with classical meta information;

(B) initialize gate circuits between the qubits with the classical meta information;

(C) process a given quantum circuit by transforming all qubits by unitary matrices;

(D) measure the qubits to retrieve classical information; and

(E) process classical information.

2. A virtual quantum computer system, comprising:

a main memory (610);

one or more memory bus systems (650) coupled to the main memory (610);

one or more physical processing units (620) that have access to the main memory (610), preferably via at least one memory bus system;

a data processing unit (640) coupled to at least one of the one or more physical processing units (620), the data processing unit (640) serving as a bridge between internal systems of the virtual quantum computer system and external systems; and

wherein a Bloch sphere is built into an intermediate representation of a memory pattern within the main memory (610) in such a way that associated same state vectors $|\psi\rangle$ are fully represented within the memory pattern alongside or along with the classical information $|0\rangle$ and $|1\rangle$.

3. The system of claim 1 or 2, wherein:

the one or more cache coherency interconnects (630) comprise physical hardware components; and/or

the initialization of the qubits with classical meta information includes performing memory pattern translation (720), wherein, preferably, the memory pattern translation comprises translation to one or both of Bloch registers and/or neural networks.

4. The system of any of the preceding claims, wherein optimization parameters in a memory representation of quantum information are provided as meta information from an application layer (750) via a kernel scheduler (API 5D) to an inner core of the operating system (770) to efficiently use overall transactional computation power of the information process stack; and/or

wherein the information process stack is further configured to:

transfer, via a kernel scheduler API (740, 830), the classical meta information required for the quantum gate circuit to a memory pattern translation layer (720) associated with the operating system (770); and/or

the system further comprising physical qubit registers, wherein the classical meta information required for the quantum gate circuitry is computed by a gate control unit of a native quantum processor; and/or

the system further comprising a virtual quantum processor, wherein gate matrices within the main memory are constructed with the classical meta information required for the quantum gate circuit, such that processor hardware is implemented with an agnostic architecture of the information process stack, preferably above the kernel scheduler API; and/or

the system further comprising a quantum processing unit (QPU) that is configured to utilize an arithmetic and logic unit (ALU) as a piece of software in processor kernel extensions of a hybrid quantum computing operating system, preferably carried out by physical resources of the hardware layer (780); and/or

wherein the container environment (760) includes a kernel scheduler API having MPI overlay functionality (740) that provides known methods of thread parallelization to a programmer, such that the programmer can distribute applications and/or tasks within one application over arbitrary numbers of different processors and compute nodes that run a same version of the operating system (770).

5. The system of any of the preceding claims, wherein, when running an application, the information process stack is configured to:

run parallel versions of the application on both virtual quantum processing units (virtual QPUs) and native quantum processing units (native QPUs),

wherein, preferably, when the application is being run via a native QPU, a memory pattern translation unit (720) is configured to read out state vectors; and/or

wherein, preferably, when the application is being run via a virtual QPU, the system is configured to utilize advanced quantum inspired computing (AQIC) shortcuts automatically; and/or

wherein the information process stack is configured to:

store data utilized in different instances of native and virtual processing in shared memory using virtual processor instances (VPIs) comprised of processor kernel extensions.

6. The system of any of the preceding claims, wherein a memory pattern itself within the main memory (720) functions as meta-information for the respective processing units to control their execution of the data processed, such as to realize cache coherency, implement inherent

intermediate representations or deliver information about topological dependencies to the control units of the processors, which, preferably, is configured for utilization by software developers to orchestrate different processing units accessing the same main memory most efficiently in parallel; and/or

wherein a Bloch sphere (100) is built into an intermediate representation of a memory pattern within the main memory (720) in such a way that the same state vectors |ψ> (111) are fully represented within the memory pattern alongside or along with the classical information |0> and |1> (112).

7. The system of any of the preceding claims, wherein the information process stack is configured to:

store data in shared memory and process the data using virtual processor instances (VPIs), wherein the VPIs are processor kernel extensions comprising:

a plurality of subcomponents including a multi-protocol driver (MPD) (870), a meta protocol controller (MPC) (880), and an arithmetic and logic unit (ALU) (890); and

a bus system (860) configured to provide the subcomponents access to shared memory in the main memory and to handle direct memory access (DMA) exchanges involving the subcomponents;

wherein, optionally, the multi-protocol driver (MPD) (870) is configured to:

function as a driver interface to the operating system (830);

handle communication between subcomponents of virtual processor instances (VPIs), including a multi-protocol driver (MPD) (870), a meta protocol controller (MPC) (880), and an arithmetic and logic unit (ALU);

translate different protocols;

switch between the subcomponents of the VPIs and the external systems; and/or

hold cache for the virtual processing unit; and/or

wherein the virtual processing unit is configured to:

be built by the multi-protocol driver in memory, if there is no physical implementation of the VPIs; and/or

map physical cache of a physical (quantum) processing unit into the main memory and thereby provide cache coherency throughout the system; and/or

wherein the meta protocol controller (880) is configured to:

handle the meta information exchanged over the multi-protocol driver (MPD) and hold an Intermediate Representation (IR) for the information processing structures, such as quantum circuits; and/or

wherein, when the system utilizes a gate-based quantum processor, the arithmetic and logic unit (ALU) (890) is configured to carry out logic and arithmetic operations using linear algebra representation with matrix operations.

8. A method of performing virtualized quantum processing, the method comprising:

initializing qubits with classical meta information;

initializing gate circuits between the qubits with the classical meta information;

processing a given quantum circuit by transforming all the qubits by unitary matrices;

measuring the qubits to retrieve classical information; and

processing the classical information;

wherein, preferably, the method is implemented via an information process stack, or computational stack, comprising: a hardware layer, an operating system coupled to the hardware layer, and a container environment coupled to the operating system.

9. The method of claim 8, wherein the processing of the classical information is performed by virtual hardware processors that are hardware agnostic, and wherein, preferably, the virtual hardware processors are implemented via a virtual quantum computer system comprising:

a main memory (610);

one or more memory bus systems (650) coupled to the main memory (610);

one or more physical processing units (620) that have access to the main memory (610), preferably via at least one memory bus system;

a data processing unit (640) coupled to at least one of the one or more physical processing units (620), the data processing unit (640) serving as a bridge between internal systems of the virtual quantum computer system and external systems;

one or more cache coherency interconnects (630) connecting the one or more physical processing units (620); and

the information process stack.

10. The method of claim 8 or 9, further comprising:

building a Bloch sphere (100) into an intermediate representation of a memory pattern within memory, such as the main memory (720), in such a way that the same state vectors $|\psi\rangle$ (111) are fully represented within the memory pattern alongside or with the classical information $|0\rangle$ and $|1\rangle$ (112); and/or

implementing a memory pattern within the main memory (720) to function as meta-information for respective processing units to control their execution of the data processed, such as, but not restricted to realize cache coherency, implement inherent intermediate representations or deliver information about topological dependencies to the control units of the processors, which, preferably, is configured for utilization by software developers to orchestrate different processing units accessing the same main memory most efficiently in parallel.

11. The method of claim 9, optionally in combination with claim 10, wherein the one or more cache coherency interconnects comprise physical hardware components.

12. The method of any of the claims 8 to 11, wherein the initialization of the qubits with classical meta information includes performing memory pattern translation (720), wherein, preferably, the memory pattern translation comprises translation to one or both of Bloch registers and/or neural networks; and/or

wherein optimization parameters in a memory representation of quantum information are provided as meta information from an application layer (750) via a kernel scheduler API (740,

830) to an inner core of the operating system (770), preferably to efficiently use overall transactional computation power of the information process stack; and/or

wherein the information process stack is further configured to:

transfer, via a kernel scheduler API (740, 830), the classical meta information required for the quantum gate circuit to a memory pattern translation layer (720) associated with the operating system (770); and/or

wherein the information process stack is further configured for implementation with physical qubit registers, wherein the classical meta information required for quantum gate circuitry is computed by a gate control unit of a native quantum processor; and/or

wherein the method is further implemented via a virtual quantum processor, wherein gate matrices within the main memory are constructed with the classical meta information required for the quantum gate circuit, such that processor hardware is implemented with an agnostic architecture of the information process stack, preferably above the kernel scheduler API; and/or

wherein the method is further implemented via a quantum processing unit (QPU) that is configured to utilize an arithmetic and logic unit (ALU) as a piece of software in processor kernel extensions of a hybrid quantum computing operating system, preferably carried out by physical resources of the hardware layer (780); and/or

the method further comprising: implementing the container environment (760) with a kernel scheduler API having MPI overlay functionality (740) configured to provide known methods of thread parallelization, preferably to a programmer, such that the programmer can distribute applications and/or tasks within one application over arbitrary numbers of different processors and compute nodes that run a same version of the operating system (770).

13. The method of any of the claims 8 to 12, wherein, when running an application, the information process stack is configured for:

running parallel versions of the application on both virtual quantum processing units (virtual QPUs) and native quantum processing units (native QPUs),

wherein, preferably, when the application is being run via a native QPU, a memory pattern translation unit (720) is configured to read out state vectors; and/or

wherein, preferably, when the application is being run via a virtual QPU, the system is configured to utilize advanced quantum inspired computing (AQIC) shortcuts automatically; and/or

wherein the information process stack is configured for:

storing data utilized in different instances of native and virtual processing in shared memory using virtual processor instances (VPIs) comprised of processor kernel extensions; and/or

wherein the information process stack is configured for:

storing data in shared memory and process the data using virtual processor instances (VPIs), wherein the VPIs are processor kernel extensions comprising:

a plurality of subcomponents including a multi-protocol driver (MPD) (870), a meta protocol controller (MPC) (880), and an arithmetic and logic unit (ALU) (890); and

a bus system (860) configured to provide the subcomponents access to shared memory in the main memory and to handle direct memory access (DMA) exchanges involving the subcomponents.

14. The method of claim 13, wherein the multi-protocol driver (MPD) (870) is configured for:

functioning as a driver interface to the operating system (830);

handling communication between subcomponents of virtual processor instances (VPIs), including a multi-protocol driver (MPD) (870), a meta protocol controller (MPC) (880), and an arithmetic and logic unit (ALU);

translating different protocols;

switching between the subcomponents of the VPIs and the external systems; and/or

holding cache for the virtual processing unit;

wherein, preferably, the virtual processing unit is configured for:

being built by the multi-protocol driver in memory, if there is no physical implementation of the VPIs; and/or

mapping physical cache of a physical (quantum) processing unit into the main memory and thereby provide cache coherency throughout the system; and/or

wherein the meta protocol controller (880) is configured for:

handling the meta information exchanged over the multi-protocol driver (MPD) and hold an intermediate representation (IR) for the information processing structures, such as quantum circuits; and/or

wherein, during implementations that utilize a gate-based quantum processor, the arithmetic and logic unit (ALU) (890) is configured to carry out logic and arithmetic operations using linear algebra representation with matrix operations.

15. A method of performing virtualized quantum processing, the method comprising:

implementing a virtual quantum computer system comprising a main memory (610, 720), one or more memory bus systems (650) coupled to the main memory (610), one or more physical processing units (620) that have access to the main memory (610), and a data processing unit (640) coupled to at least one of the one or more physical processing units (620);

implementing a memory pattern within the main memory (610, 720), including building a Bloch sphere into an intermediate representation of the memory pattern within the main memory in such a way that associated same state vectors $|\psi\rangle$ are fully represented within the memory pattern alongside or along with the classical information $|0\rangle$ and $|1\rangle$.

.