Large-Scale Distributed State-Vector Simulation of IQPE for Early-FTQC

Shoma Hiraoka[1] · Yusuke Teranishi[1,2] · Sho Nishida[3] · Wataru Mizukami[4]

[1] Graduate School of Information Science and Technology, The University of Osaka · [2] QunaSys Inc. · [3] School of Engineering Science, The University of Osaka · [4] Center for Quantum Information and Quantum Biology, The University of Osaka

1. The simulation challenge

Simulating molecular energy estimation

Molecular energies help predict chemical stability and reactivity. We simulate quantum phase estimation (QPE) circuits for these energies at scale, ahead of practical fault-tolerant machines.

Each extra qubit doubles the data

A state vector stores 2^q complex amplitudes for q qubits. A 41-qubit state needs **32 TiB** in complex-FP64. Multiple GPUs provide the required aggregate memory, but operations spanning GPU partitions require data exchange.

IQPE reduces the memory requirement

Iterative QPE (IQPE) extracts phase digits with one reused auxiliary qubit (**ancilla**). This saves qubits in early fault-tolerant quantum computing (Early FTQC) and simulation memory. Long circuits still demand efficient communication.

Our approach reduces data movement

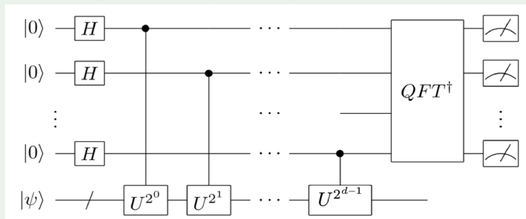
We extend existing qubit reordering with two IQPE optimizations. They avoid unnecessary state transfers and combine per-GPU probability sums for measurement.

LOWER MEMORY USE · REDUCE COMMUNICATION

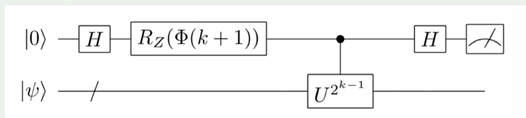
Why use IQPE? A smaller state vector

For n molecular qubits and d binary phase digits, IQPE stores 2^{n+1} rather than 2^{n+d} amplitudes.

QPE d-qubit phase register — $n+d$ qubits



IQPE measure · feed forward · reuse — $n+1$ qubits

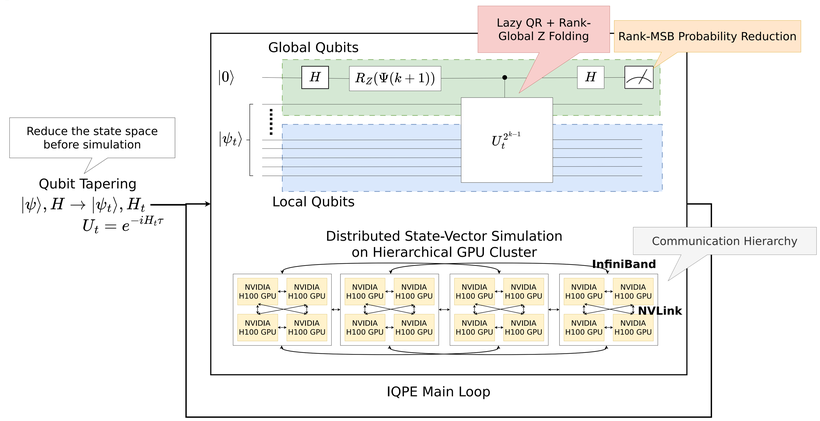


One recycled ancilla → 2^{d-1}× smaller state vector
Less memory and less simulation work per gate

2. Communication-aware IQPE execution

Each GPU holds a slice of the state vector. Compatible operations stay on that slice. Data moves only when needed.

Distributed IQPE workflow



Lazy Qubit Reordering (QR) [1] reuses data layouts across operations. We add rank-global Z folding to handle eligible phase operations without moving state data. Rank-MSB places the ancilla at the most significant bit of the MPI rank. Each GPU holds amplitudes for ancilla 0 or 1. Measurement probabilities then follow from per-GPU probability sums.

Communication-Aware IQPE Execution

MINIMIZE DATA MOVEMENT

NAIVE BASELINE — **QR BEFORE EACH NON-LOCAL ROTATION**

Naive reference: reorder independently before every non-local Pauli rotation

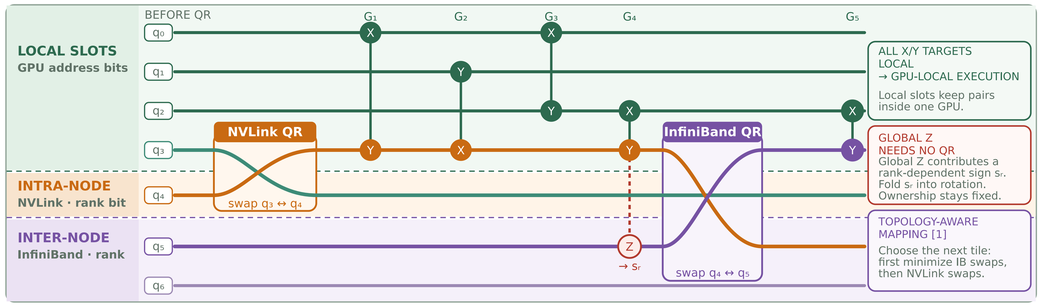
QR = SWAP STATE-INDEX BITS
BEFORE LOCAL q_4 GLOBAL q_3
AFTER LOCAL q_3 GLOBAL q_4
Example: one cross-boundary swap

QR → G_1 (make P_1 local) → QR → G_2 (make P_2 local) → QR → G_3 (make P_3 local)

REFERENCE BASELINE ONLY
QR frequency PER ROTATION

INTEGRATED PATH — **LAZY QR [1] + RANK-GLOBAL Z FOLDING** — **ONE MAPPING → MANY ROTATIONS**

Lazy QR [1] reuses one mapping across compatible rotations. Rank-global Z folding widens each compatible run.



KEEP ANCILLA AT RANK MSB — **REDUCE MASSES, NOT AMPLITUDES**

Ancilla a stays at rank MSB and out of QR → one ancilla value per GPU. Reduce one scalar mass $w_r = \|\psi_r\|_2$

a=0: GPU 00 • → w_{00} + GPU 01 • → w_{01} → P_0
a=1: GPU 10 • → w_{10} + GPU 11 • → w_{11} → P_1

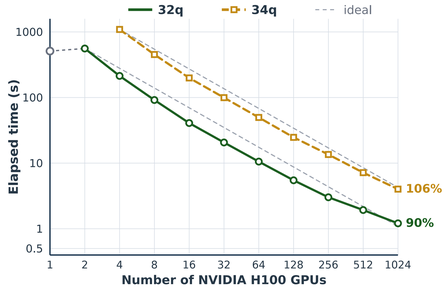
SELECT MOST-PROBABLE BIT $j_k = \arg\max_b P_b$ → **FEEDBACK BIT** exact P_0, P_1 · no sampling

3. Results

Experimental setup

Hardware: ABCI-Q · 4× H100/node · 80 GB/GPU · 1 MPI rank/GPU
Software: chemqulacs-gpu · cuStateVec 1.11.1 · CUDA 12.9 · HPC-X 2.22.1 · complex-FP64

Cl_2 strong scaling for one IQPE digit



Single-digit runtime · strong-scaling efficiency

32q · 22,487 terms · 90.2%: 558.91 → 1.21 s (2 → 1,024 GPUs)
34q · 28,770 terms · 106.0%: 1,091.26 → 4.02 s (4 → 1,024 GPUs)

Distributed efficiency is $E(p;p_0) = p_0 \frac{T(p_0)}{pT(p)}$. Here $p_0 = 2$ (32q) and 4 (34q).
1 GPU uses the single-GPU path and is shown only for reference.

Scaling behavior

From the first multi-GPU configuration, both workloads scale nearly ideally to 1,024 GPUs, reaching **90.2%** (32q) and **106.0%** (34q) efficiency. The series use Cl_2/STO-3G (34e,18o) and Cl_2/6-31G (34e,19o), respectively. Both are tapered and include the ancilla. The 34q endpoint is superlinear relative to its 4-GPU baseline. Timing alone does not identify the mechanism.

Large-scale execution · all runs on 1,024 GPUs

A 41-qubit complex-FP64 state requires 32 TiB. In our implementation, U^{2^k} repeats the base evolution, so d IQPE digits execute $2^d - 1$ evolution units. Smaller states leave memory headroom for more phase digits. For wider states, wall time grows rapidly with the number of digits.

Molecule	Qubits	Basis / source	Pauli terms	Digits	Time (min)
Cl_2	34	6-31G · (34e,19o) · taper	28,770	11	120.6
SO_2	39	STO-3G · (32e,19o)	52,270	2	9.3
H_2O	39	cc-pVTZ · (10e,21o) · taper	76,068	1	4.1
Fe_2S_2	39	FCIDUMP · (30e,20o) · taper	**232,793**	1	11.9
Fe_2S_2	**41**	FCIDUMP · (30e,20o)	**232,793**	1	49.5

All counts include one IQPE ancilla. Parentheses give active-space (electrons, orbitals). Pauli terms are runtime Hamiltonian counts. Both Fe_2S_2 rows share the (30e,20o) FCIDUMP. The 39q state is tapered and the 41q state is untapered.

SYSTEMS-SCALE TAKEAWAY

Strong scaling reaches 1,024 GPUs with 90.2% (32q) and 106.0% (34q) efficiency relative to the first multi-GPU baselines.

A 41-qubit simulation completes end to end. The complex-FP64 Fe_2S_2 run uses 1,024 GPUs.

The integrated path reduces data movement by combining Lazy QR [1], rank-global Z folding, and rank-MSB probability reduction.

We report problem size, end-to-end wall time, and distributed efficiency at scale. IQPE digits denote executed phase-estimation depth. Lazy QR is benchmarked in [1].

Acknowledgment. Computational resources were provided by the ABCI-Q Grand Challenge Program of AIST. OpenAI Codex (GPT-5) assisted with prose, figure composition, and layout throughout this poster [2]. The authors verified all claims. · **Code:** github.com/qiqb-osaka/chemqulacs-gpu · [1] Y. Teranishi et al., "Lazy Qubit Reordering for Accelerating Parallel State-Vector-based Quantum Circuit Simulation," ACM TQC 6(4), 2025. · [2] OpenAI, Codex (GPT-5), 2026.