FockMap: A Composable Functional Framework for Fermion-to-Qubit Encodings in F#

John Azariah[*1]

[1]University of Technology Sydney

February 2026

Summary

Simulating fermionic systems on quantum computers requires mapping fermionic creation and annihilation operators to qubit (Pauli) operators — the fermion-to-qubit encoding problem. Different encodings (Jordan–Wigner, Bravyi–Kitaev, Parity, tree-based) trade off Pauli weight, circuit depth, and implementational complexity. Existing libraries implement each encoding as a separate, non-composable transform, obscuring the shared mathematical structure.

We present `FockMap`, an open-source F# library that unifies all known fermion-to-qubit encodings under two composable abstractions: *index-set schemes*, in which an encoding is specified by three set-valued functions (`Update`, `Parity`, `Occupation`), and *path-based tree encodings*, in which any rooted labelled tree automatically induces a valid encoding. The index-set framework captures Jordan–Wigner, Bravyi–Kitaev, and Parity in 3–5 lines each; the path-based framework handles arbitrary tree topologies including balanced binary and balanced ternary trees that achieve provably optimal $O(\log n)$ Pauli weight.

Crucially, the entire operator pipeline is *symbolic*: Pauli strings are multiplied exactly using an algebraic phase-tracking rule—no matrices are instantiated and no floating-point arithmetic occurs until the final Hamiltonian coefficients are assembled. This makes correctness verifiable by inspection and eliminates an entire class of numerical-precision bugs that plague matrix-based approaches.

The library is built entirely from algebraic data types and pure functions, features a persistent Fenwick tree ADT, includes 303 unit and property-based tests, and provides a running example encoding the complete H_2 STO-3G Hamiltonian across all five built-in encodings.

Statement of Need

Quantum simulation of molecular electronic structure is widely regarded as one of the most promising near-term applications of quantum computing [Feynman, 1982, Aspuru-Guzik et al., 2005]. Between the molecular Hamiltonian in second quantization and the measurements performed on quantum hardware lies a critical middleware step: the fermion-to-qubit encoding. The choice of encoding determines the Pauli weight of each operator (and hence circuit depth), the number of measurement terms, and ultimately whether a simulation is feasible on a given device.

Current tools for this step — OpenFermion [McClean et al., 2020], Qiskit Nature [Qiskit contributors, 2023], and PennyLane [Bergholm et al., 2022] — implement each encoding as a monolithic function mapping `FermionOperator` → `QubitOperator`. Adding a new encoding requires writing hundreds of lines of bespoke code. The mathematical structure shared across encodings (Majorana decomposition, parity tracking, update sets) is duplicated rather than abstracted. There is no mechanism for users to define, compose, or compare custom encodings programmatically.

`FockMap` addresses this gap. An encoding is a *value* — a record of three functions — not a class hierarchy. This design enables:

- **Exploration:** researchers can define and test novel encodings in 3–5 lines of code.

[*]ORCID: 0009-0007-9870-1970

- **Comparison:** all encodings share the same verification pipeline (anti-commutation tests, eigenspectrum comparison).
- **Pedagogy:** the code *is* the mathematics — an `EncodingScheme` is the literal definition from the literature, not an opaque implementation of it.

The library serves quantum computing researchers exploring encoding-aware circuit synthesis, students learning the algebraic structure of encodings, and developers building quantum chemistry simulation pipelines who need a correct and composable encoding layer.

Functionality

Encoding Schemes (Index-Set Framework)

The `EncodingScheme` record type captures the three index-set functions that define a fermion-to-qubit encoding. Three concrete schemes are provided: `jordanWignerScheme`, `bravyiKitaevScheme`, and `parityScheme`. User-defined schemes are first-class values of the same type:

```
let myScheme : EncodingScheme =
    { Update = fun j n -> set [ j + 1 .. n - 1 ]
      Parity = fun j -> if j > 0 then Set.singleton (j - 1)
                        else Set.empty
      Occupation = fun j -> if j > 0 then set [j-1; j]
                            else Set.singleton j }
```

The framework automatically constructs Majorana operators c_j and d_j from these three functions, then derives ladder operators $a_j^\dagger$ and a_j by linear combination.

Tree Encodings (Path-Based Framework)

Any rooted labelled tree defines a fermion-to-qubit encoding. The library provides `balancedBinaryTree` and `balancedTernaryTree` constructors; users can build arbitrary trees from `TreeNode` values. The path-based encoding function traverses the tree to construct Majorana operators without requiring the index-set monotonicity constraint, making it strictly more general than the index-set framework.

Symbolic Algebra Engine

A distinguishing feature of `FockMap` is that operator multiplication is entirely symbolic. The `PauliRegister` type represents a Pauli string (e.g. $XZIY$) together with an exact `Phase` drawn from $\{+1, -1, +i, -i\}$. Multiplying two Pauli registers applies the single-qubit multiplication table ($X \cdot Y = iZ$, etc.) position-wise and accumulates the phase algebraically—no $2^n \times 2^n$ matrices are ever constructed. The resulting `PauliRegisterSequence` (a weighted sum of Pauli strings) is the symbolic representation of an encoded operator.

This design means that encoding a ladder operator on 100 modes produces a compact list of Pauli strings, not a $2^{100} \times 2^{100}$ sparse matrix. Correctness can be verified symbolically: the anti-commutation tests in the verification suite check $\{a_i, a_j^\dagger\} = \delta_{ij}$ by Pauli string cancellation, not by matrix eigenvalue comparison.

The library also includes a persistent `FenwickTree<'a>` (parameterised over any monoid) and a `Hamiltonian` module for constructing molecular Hamiltonians from one-body and two-body integrals.

Verification Suite

All 303 tests pass across three categories:

- **Anti-commutation:** $\{a_i, a_j^\dagger\} = \delta_{ij}$ verified symbolically for all mode pairs.
- **Number conservation:** $a_j^\dagger a_j$ produces diagonal Pauli operators.
- **Cross-encoding agreement:** all five encodings produce isospectral Hamiltonians for H_2 (eigenvalue agreement to 5×10^{-16}).

Design Principles

Encodings as data. An `EncodingScheme` is a value, not a class hierarchy. Jordan–Wigner, Bravyi–Kitaev, and Parity are different values of the same type. This enables algebraic reasoning: one can ask whether two schemes agree on a given mode without running a full encoding.

Two complementary frameworks. The index-set framework (`MajoranaEncoding.fs`) is fast and algebraically transparent but requires a monotonicity condition on ancestor indices. The path-based framework (`TreeEncoding.fs`) works for *any* tree topology. Both produce the same output type (`PauliRegisterSequence`), so downstream code is encoding-agnostic.

Symbolic over numerical. Existing libraries represent operators as sparse matrices or coefficient dictionaries indexed by opaque integer keys. `FockMap` represents them as typed Pauli strings with exact algebraic phases. This makes the intermediate representation human-readable, composable, and free of floating-point error accumulation. Numerical coefficients enter only at the Hamiltonian assembly stage, where they multiply symbolic Pauli terms.

Pure functions, no mutation. All data structures are immutable: persistent Fenwick trees, recursive tree ADTs, and Pauli register sequences. The library has zero mutation and no side effects in its core modules.

Discovered constraints. Implementation and testing revealed that the index-set framework's monotonicity requirement (ancestor indices must exceed descendant indices) is satisfied only by star-shaped trees — a structural constraint not previously documented in the literature. This discovery motivated the path-based framework as a universal alternative and is explored further in a companion paper.

Comparison with Related Software

Feature	**OpenFermion**	**Qiskit Nature**	**PennyLane**	**FockMap**
JW / BK / Parity	✓/✓/✓	✓/✓/✓	✓/✓/—	✓/✓/✓
Tree encodings	Steiner ext.	—	—	Binary, Ternary
User-defined encodings	—	—	—	✓
User-defined trees	—	—	—	✓
Generic encoding abstraction	—	—	—	✓
Symbolic Pauli algebra	—	—	—	✓
Typed / functional	—	—	—	✓
Persistent Fenwick tree	—	—	—	✓

Table 1: Feature comparison with existing fermion-to-qubit libraries.

OpenFermion [McClean et al., 2020] is the most comprehensive existing tool, offering extensive support for operator manipulation and circuit synthesis. Qiskit Nature [Qiskit contributors, 2023] integrates tightly with IBM quantum hardware. PennyLane [Bergholm et al., 2022] excels at differentiable quantum computing. `FockMap` does not compete on scope; it provides the *framework* abstraction that these libraries lack, enabling systematic exploration and comparison of encodings.

Acknowledgements

This work is dedicated to Dr. Guang Hao Low, whose encouragement to explore Bravyi–Kitaev encodings seven years ago led to the development of this symbolic algebra processing library. We also thank the F# Software Foundation and the .NET open-source community for the language and runtime that make this work possible.

References

Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21:467–488, 1982. doi: 10.1007/BF02650179.

Alán Aspuru-Guzik, Anthony D. Dutoi, Peter J. Love, and Martin Head-Gordon. Simulated quantum computation of molecular energies. *Science*, 309:1704–1707, 2005. doi: 10.1126/science.1113479.

Jarrod R. McClean, Nicholas C. Rubin, Kevin J. Sung, et al. OpenFermion: the electronic structure package for quantum computers. *Quantum Science and Technology*, 5(3):034014, 2020. doi: 10.1088/2058-9565/ab8ebc.

Qiskit contributors. Qiskit: An open-source framework for quantum computing. 2023. doi: 10.5281/zenodo.2573505.

Ville Bergholm, Josh Izaac, Maria Schuld, et al. PennyLane: Automatic differentiation of hybrid quantum-classical computations. *arXiv preprint arXiv:1811.04968*, 2022.