TKET USER FEATURES

Kathrin Spendier
Quantum Evangelist

ORNL Quantum Computing User Program Workshop

Get started with TKET, a universal quantum SDK

TKET is available for free on GitHub (https://gitub.com/CQCL/pytket) and is installed by

```
pip install pytket
```

The extension module interfacing PyTKET with the H-series is installed by

```
pip install pytket-quantinuum
```

and for interfacing with Qiskit install

```
pip install pytket-qiskit
```

TYPICAL QUANTUM ALGORITHM WORKFLOW ON A GATE-MODEL QUANTUM COMPUTER

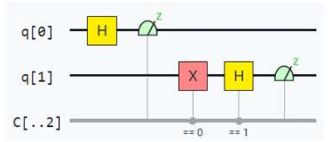
e.g. Traveling Salesman

```
circuit = QuantumCircuit(q,c)
normal = NormalDistribution(num_target_qubits = 5, mu=0, sigma=1,
low=-1, high=1)
normal.build(circuit,q)
circuit.measure(q,c)

job = execute(circuit, backend, shots=8192)
job_monitor(job)
counts = job.result().get_counts()

print(counts)
sortedcounts = []
sortedkeys = sorted(counts)

for i in sortedkeys:
    for j in counts:
        if(i == j):
            sortedcounts.append(counts.get(j))
```



e.g. TKET

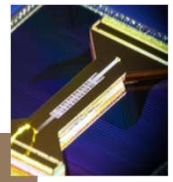
Problem Definition

Quantum Algorithm

Quantum Circuit

Quantum Compiler

Quantum Processor

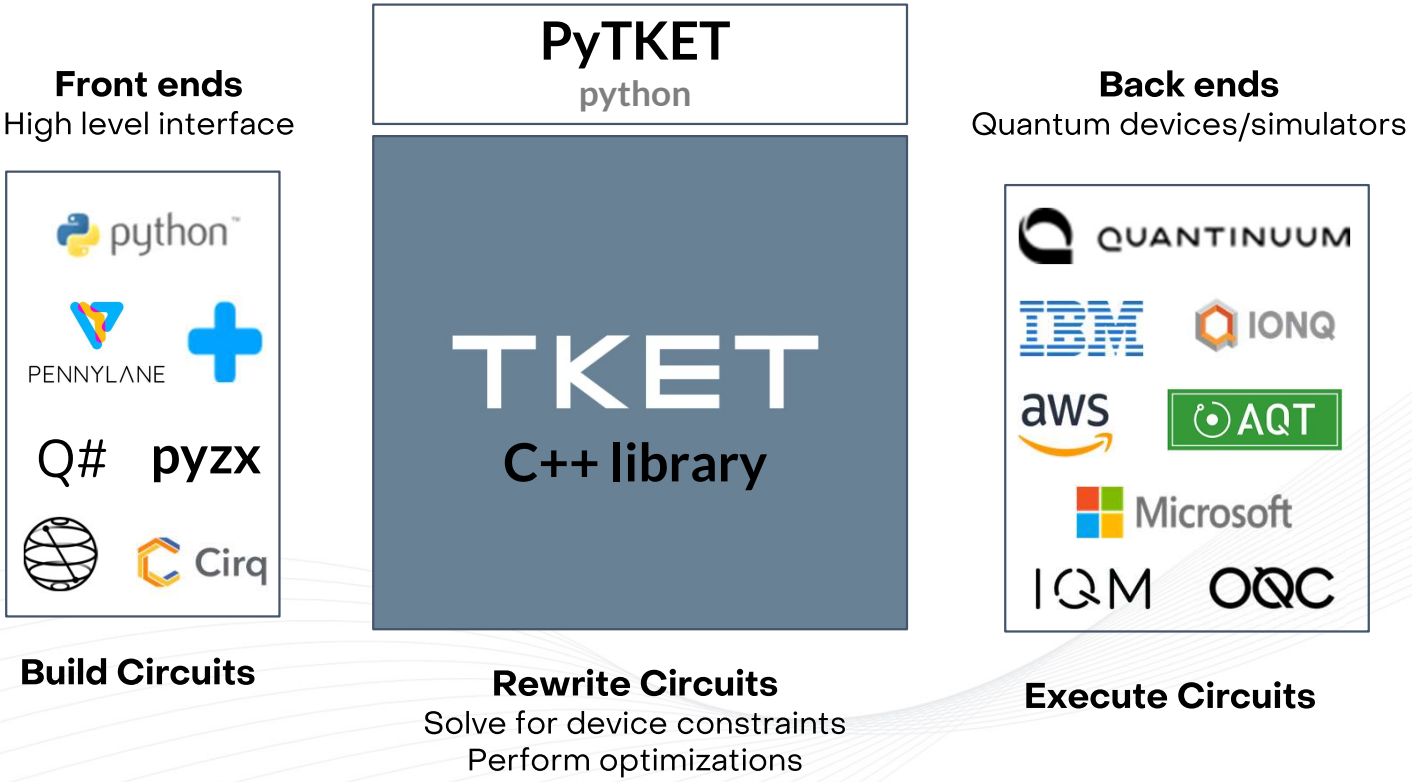


Quantum Simulator

Full-Stack

TKET as a universal SDK

TKET optimizes quantum circuits, reducing the number of required operations – essential for NISQ devices.



TKET EXTENSIONS

Device & Simulators

- pytket-quantinuum
- pytket-qiskit (IBM)
- pytket-ionq
- pytket-aqt
- pytket-braket (AWS)
- pytket-qsharp (Azure)
- pytket-pyquil (Rigetti)
- pytket-iqm

Simulators

- pytket-qujax
- pytket-project
- pytket-pysimplex
- pytket-qulacs
- pytket-stim
- pytket-cutensornet*

* Under development

Transpilers

- pytket-pennylane
- pytket-pyzx
- pytket-cirq
- pytket-qir*

* Under development

TKET SIMPLIFIES THE INTEGRATION OF DIFFERENT QUANTUM TOOLKITS.

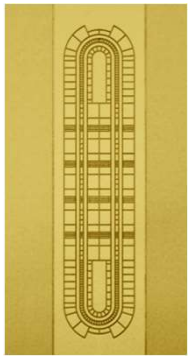
qiskit_to_tk(circuit)

Other bidirectional conversions exist: Cirq, Bracket etc.

tk_to_qiskit(circuit)

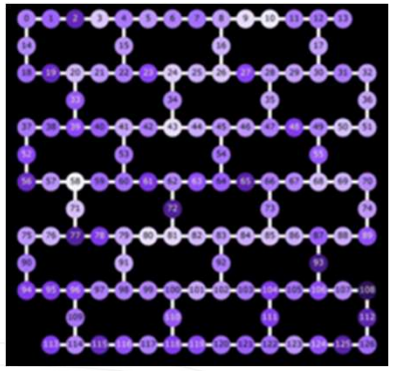
Quantum hardware architectures

H-2



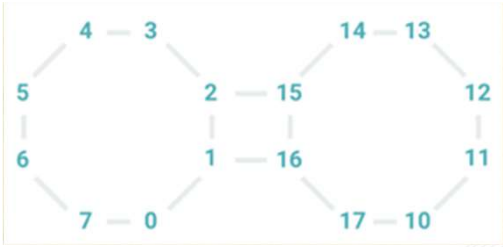
Basis Gates:
$U1q(\theta,\varphi)$, R_Z, ZZ, RZZ

EAGLE



Basis Gates:
ECR, ID, R_Z, S_X, X

ASPEN



Basis Gates:
CZ, R_X, R_Z

TKET-SPECIFIC COMPILER PASSES

- Synthesize many qubit operations to 1-, 2-qubit gates
- Local graph rewrites, pattern-replacement
- Resynthesize sub-circuits via special representations
  - ZX-terms, Clifford tableaux, Phase-polynomial / Phase gadget
- Architecture-aware synthesis
- Mapping to chosen gate basis
- Mapping and routing circuits to fixed architectures
- Symbolic expression optimization
- …. more!

TKET has a default pass manager for each backend

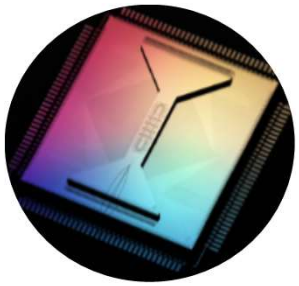
```
get_compiled_circuit(circuit,optimization_level)
```

Level 0	Level 1	Level 2 (default)
Solves the device constraints without optimizing.	Additionally performs some light optimizations.	Adds more intensive optimizations that can increase compilation time for large circuits.

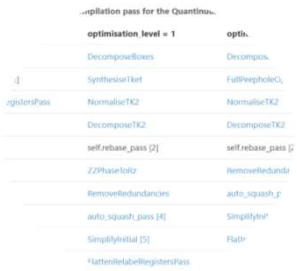
CODING EXAMPLE OF SOME TKET FEATURES

ADD TKET TO YOUR WORKFLOW IF COMPATIBLE – IT COULD HELP YOU!

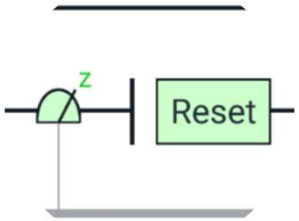
PYTKET-QUANTINUUM



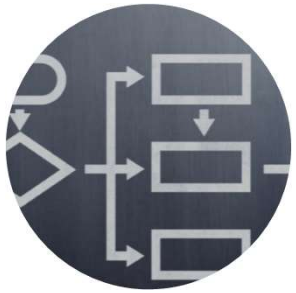
Provides access to various H-series QPUs, emulators, and syntax checkers. Emulators and syntax checkers are available for specific devices, and each device has its own specifications.



Offers a default compilation pass that optimizes circuits based on different levels of optimization. The optimization levels range from 0 to 2, with level 2 being the default, applying more intensive optimizations.



Provides predicates that circuits must satisfy to run on H-series devices. It Supports mid-circuit measurements, fast classical feedforward, cost calculation, and partial results retrieval.



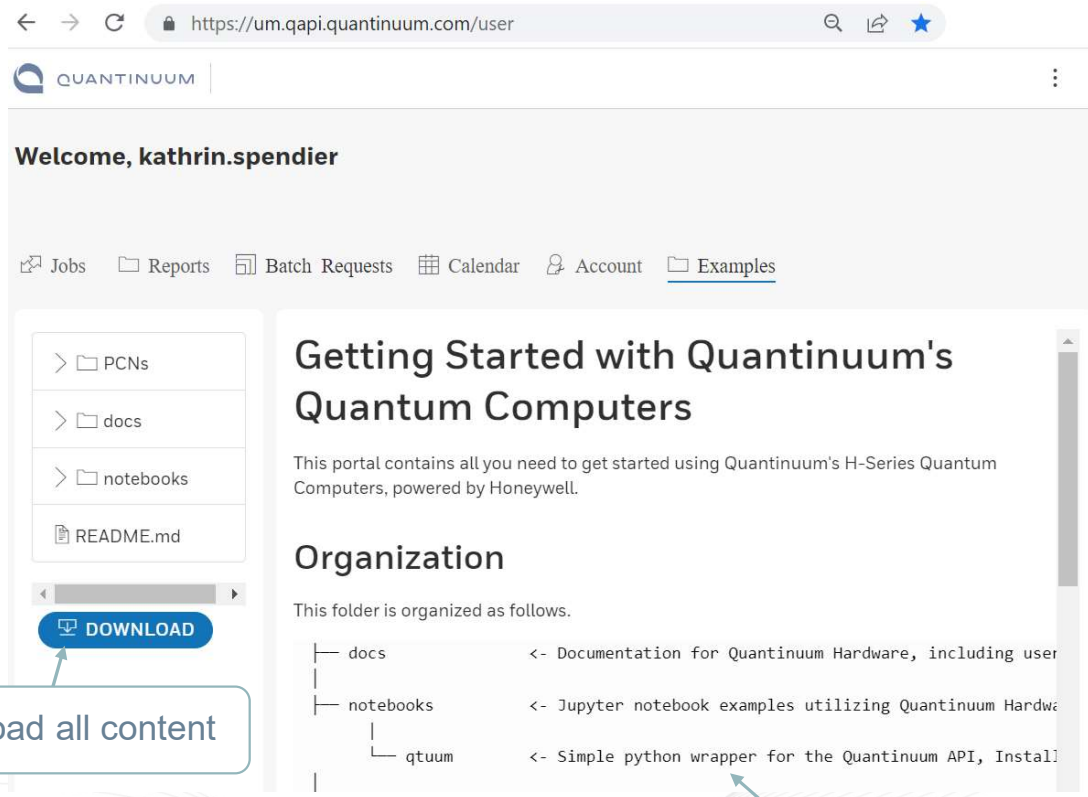
Supports batching of jobs (circuits), allowing submission of multiple circuits together as a batch, which will be executed one after another on the QPU.

Allows control of the language used for circuit submission such as QASM and QIR*.

* Under development

Getting Started with the H-series

- Web Interface: https://um.qapi.quantinuum.com
- Download the set of examples
- Set up Python Environment on your computer following the instructions
- Open Jupyter Notebook



Download all content

Follow installation instructions

JUPYTER NOTEBOOKS

Code examples include:

- Circuit Submission (conditional gates, parametrized circuits)
- Using the Emulator
- Quantinuum OpenQASM Extension
- Circuit Batching
- Mid-Circuit Measurement
- Arbitrary Angle ZZ Gates
- Using Qiskit with Quantinuum Devices
- Using the Leakage Gadget
- Qubit Reuse Compilation (see packages folder)

Quantinuum Systems

Workflow

Syntax Checker (i.e **`H1-1SC`**)

- Ensure that your quantum circuit will run on Quantinuum hardware before submitting jobs
- Checks the quantum circuit syntax against a device's compiler
- Free to use, does not require H-System Quantum Credits (HQCs)

Emulator (i.e **`H1-1E`**)

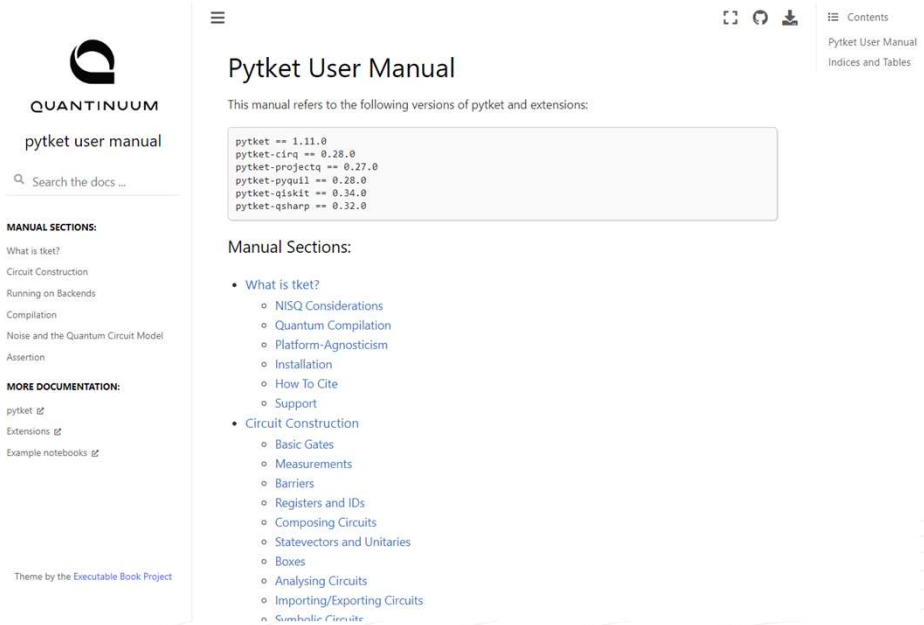
- Classical emulation of the H-Series quantum computers
- Realistic physical and noise models of the devices
- Requires HQCs

Hardware (i.e **`H1-1`**)

- Trapped ion quantum computers
- Requires HQCs

CODING EXAMPLE FOR H-SERIES WORKFLOW

PyTKET



- PyTKET User Manual: https://cqcl.github.io/pytket/manual/index.html
- PyTKET API: https://cqcl.github.io/tket/pytket/api/index.html
- Github: https://github.com/CQCL/pytket-extensions
- Slack: https://app.slack.com