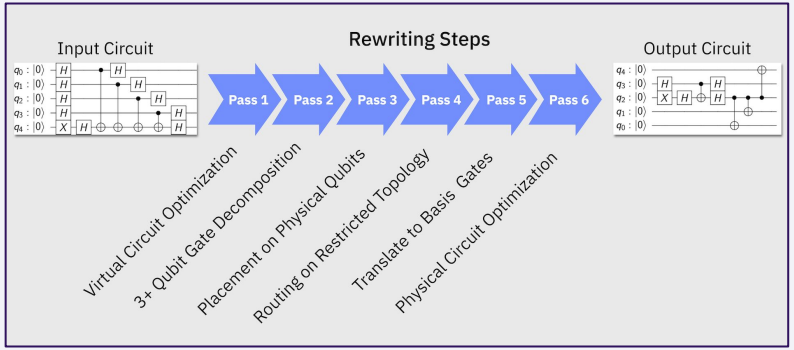
Qiskit Advocate Mentorship Program

Timeline debugger for the Qiskit transpiler

Mentor: Kevin Krsulich

What is a Transpiler

- Quantum algorithms and applications are usually written as **abstract, device agnostic** quantum circuits that may contain any unitary operations.
- However, real quantum devices can only execute a limited set of **hardware-specific, physically calibrated** quantum gates.
- We need to rewrite an abstract quantum circuit into a functionally equivalent one that matches the constraints and characteristics of a specific target quantum device.
- This essential process is known as transpilation. And the tool responsible for it is called the **transpiler**.



Functions of a Transpiler

The transpiler is built as a collection of single purpose passes and a PassManager which is responsible for collecting those passes, and coordinating their execution in order to achieve two main goals:

- **Compatibility**: transform a given quantum circuit into one which is executable on a specific device, preserving measurement outcomes.
- **Optimization**: find an implementation which takes maximum advantage of device resources, while minimizing influence of decoherence and errors.

Compatibility

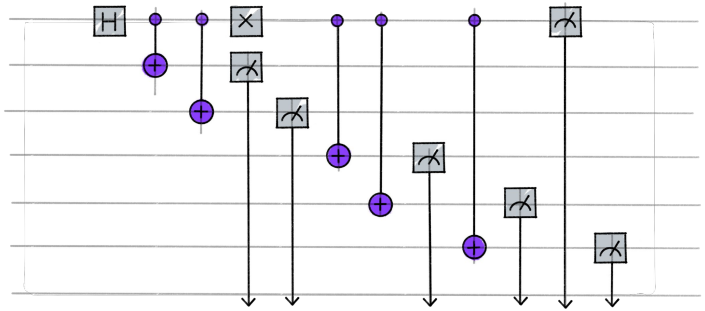
- Expand high level instructions.
- Device's native gate set.
- Layout virtual to physical qubits.
- Device architecture (superconducting – ion trap – ...)
- Device-specific constraints (no mid-circuit measurements – resets – ...)

Optimization

- Remove gate-inverse pairs
- Compact chains of single-qubit gates.
- Commutation analysis and adjacent gate cancellation
- Noise-aware layout selection
- Optimal synthesis of two-qubit blocks

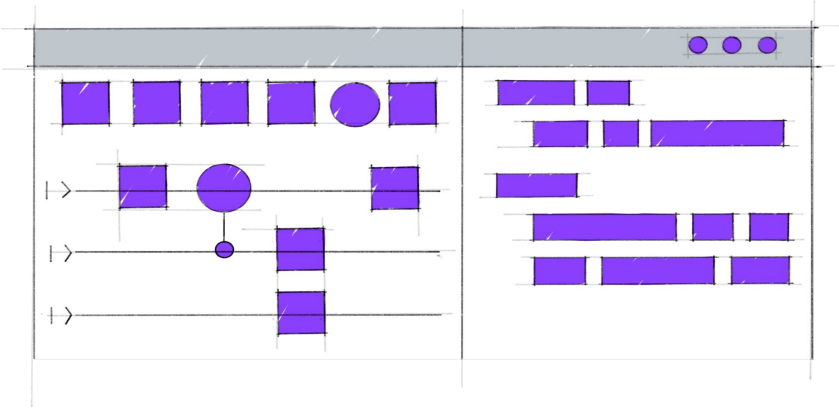
What is Transpiler Debugger

- The Qiskit transpiler and `PassManager` have built-in logging and callback mechanisms to help users understand what changes are being made to their circuit, at which stage in the process, and why.
- These tools are invaluable in investigating and debugging issues in the transpiler.
- However, most users don’t know about these tools, or don’t have (and shouldn’t need!) the deep transpiler knowledge to know how to use them.
- This is where this project can have substantial impact, providing an insight into the qiskit transpiler.



Goals of the Transpiler Debugger

- Provides users with an understandable interface to **interact** with the transpiler.
- Helping users to **find** which passes are responsible for the large changes in overall circuit properties: depth, basis, duration, or seeing these properties (and their changes pass by pass)
- Helping users to **understand** the transpilation process (which passes ran when, were responsible for which changes to a circuit, ...)
- Guiding users during debugging sessions by collecting all the data they need to **investigate** the issue, identify the root cause, and fix it.



Qiskit

“Programming allows you to think about thinking, and while debugging you learn learning”

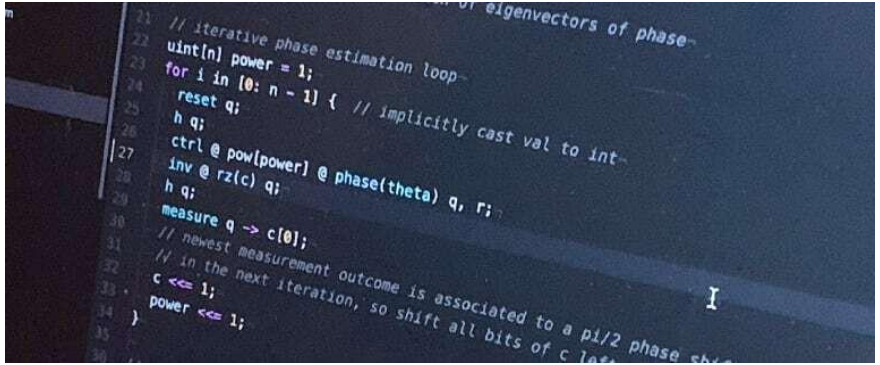
Nicholas Negroponte
Architect and Computer Scientist

Progress

- Kickoff meeting
  - Project scope.
  - Functional requirements discussed in detail.
  - Agreed on how and when the progress meetings will be conducted.
- Learning ipywidgets module used for developing interactive widgets for Jupyter notebooks and the IPython kernel.
- Prototype and UI design (WIP)
  - Iterative design approach (Design – Prototype – Evaluate – Repeat)
  - Two prototypes have been proposed.

Challenges

- Designing an intuitive, user-friendly interface for the debugger.
- In case of multi-circuit transpilation, how to separate data related to each circuit.
- Highlighting changes made by each transpiler pass in case of circuits with large number of circuits without suffering from bad performance.



Resources

- [Transpiling Quantum Circuits - Kevin Krsulich]
- [How Does The Qiskit Transpiler Work? - Qiskit blog]
- [Transpiler Passes and Pass Manager - Qiskit Tutorials]

Thank You!