

# Transpilation (level 3 + custom Coupling Map) change my register's name:

[Edit](#)[New issue](#)

## CircuitError: 'qargs not in this circuit' #XXXX

[Open](#)

ANONYMOUS opened this issue 23 days ago · 2 comments

ANONYMOUS commented 23 days ago



### Environment

- Qiskit Terra version: 0.19.1
- Python version: 3.8
- Operating system: Ubuntu 18.04.6 LTS

### What is happening?

The transpilation pass with optimization level 3 and coupling map passed returns a circuit with a default name "q" for its quantum register, so if I try to add other gates (e.g. measurement) after, it fails with error: ``

### How can we reproduce the issue?

Run this script:

```
from qiskit import QuantumCircuit, ClassicalRegister, QuantumRegister, transpile
qr = QuantumRegister(3, name='qr')
cr = ClassicalRegister(3, name='cr')
qc = QuantumCircuit(qr, cr, name='qc')
qc.h(0)
print("BEFORE:")
print(qc._qubit_indices.keys())
qc = transpile(qc, optimization_level=3, coupling_map=[[0, 1], [1, 0], [1, 2], [2, 1]])
print("AFTER:")
print(qc._qubit_indices.keys())
qc.draw(fold=-1)
```

### Output

```
BEFORE:
dict_keys([Qubit(QuantumRegister(3, 'qr'), 0), Qubit(QuantumRegister(3, 'qr'), 1), Qubit(QuantumRegister(3, 'qr'), 2)])
AFTER:
dict_keys([Qubit(QuantumRegister(3, 'q'), 0), Qubit(QuantumRegister(3, 'q'), 1), Qubit(QuantumRegister(3, 'q'), 2)])

qr_0 -> 0 { H }
qr_1 -> 1 -----
qr_2 -> 2 -----
cr: 3/=====
```

In the output you see that circuit's internal `_qubit_indices.keys()` has been changed as side effect of the optimization.

In practice if we add a measurement we get the error: `CircuitError: 'qargs not in this circuit'`

```
qc.measure(qr, cr)
```

### Output:

```
`CircuitError: 'qargs not in this circuit'`
```

### What should happen?

I expect the optimization not to change the name of the registers I use in my circuit as an unwanted side-effect.

### Any suggestions?

The culprit might be this pass `qiskit.transpiler.passes.layout.apply_layout.ApplyLayout` since it changes the internal register name:

[qiskit-terra/qiskit/transpiler/passes/layout/apply\\_layout.py](#)  
Line 53 in 6eb1296

```
53 n = QuantumRegister(len(layout), "q")
```

#### Assignees

No one assigned

#### Labels

bug

#### Projects

None yet

#### Milestone


No milestone

#### Development

No branches or pull requests

#### Notifications

Customize

 Unsubscribe

You're receiving notifications because you authored the thread.

2 participants

Moreover, if this pass is followed by `qiskit.transpiler.passes.layout.enlarge_with_ancilla.EnlargeWithAncilla` it seems to restore the original register name.

It is a quite strange behavior, I welcome any additional suggestion. Thanks in advance



**ANONYMOUS** added the **bug** label 23 days ago

**QISKIT DEV** commented 23 days ago • edited

Contributor

It is a slightly strange and non-ideal behaviour, for sure. There are a few places that for legacy reasons create new registers or create new qubit instances unnecessarily, and we do need to go through and squash those cases.

Just in case you didn't know, though: you almost invariably shouldn't be modifying a circuit after transpilation. That should be the final step in any processing pipeline - the transpilation is meant to prepare a complete quantum program to run on an actual backend, and any modifications you make to it after that will invalidate the operation. Think of it like trying to modify the executable of a program written in C after you compile it.

**ANONYMOUS** commented 22 days ago

Author

@**QISKIT DEV** thanks for commenting.

Indeed, my use of the API can be considered a bit borderline. Anyway, have you already considered in the past to use an immutable version of the `QuantumCircuit` after the transpilation so to enforce this rule (aka don't use the circuit after transpilation) if that is the intended behavior?

Write Preview

H B I

Leave a comment

Attach files by dragging & dropping, selecting or pasting them.



Close issue

Comment

Remember, contributions to this repository should follow its [contributing guidelines](#) and [code of conduct](#).