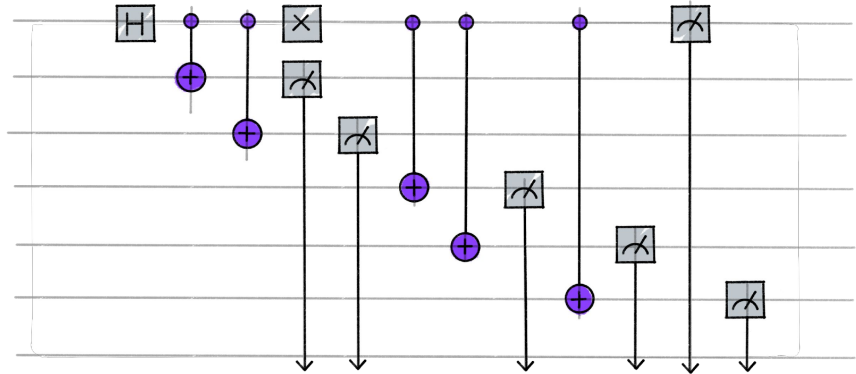


# #23: OpenQASM 3.0 Reference Python Implementation | “QAMP-21”

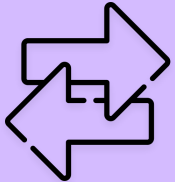
Mentored by – Jack Woehr  
Abeer Vaishnav, Adrien Suau, Vishal Bajpe



# Why OpenQASM 3.0?

---

Convenient &  
standardised format  
for quantum circuits



Hardware-agnostic  
representations



Closer to real  
hardware



Straightforward syntax



# Upgrades in OpenQASM 3.0

A complete language for quantum circuits now with salient features as compared to OpenQASM 2.0:

- Complete type system (constants, variables, operators, casting, expressions, ...)
- Control flow statements
- Support for versatile circuit and operation expression
- Dynamic circuit subroutines and external function calls
- Support for lower level operation definition
- Extended grammar for pulse operations

## Physical level

`delay` statements, adding relative timings to operation

Type `stretch` to resolve concrete durations at compile time for granular calibration

Support for qubit-specific calibration instructions via `defcal` construct

## Design Philosophy & execution

Arbitrary classical control flow, gate modifiers and timings

Ability to perform new kinds of circuits and experiments

**Pulse level calibration** and **multi level optimization**

## Logical Level

Ability to use quantum-classical dependencies in quantum circuits

Native support for classical computation on measurement results

Robusts classical types with support for classical control flow

Support for `int`, `uint`, `float`, `bool`, and `bit` for classical types with functionality to specify a type with exact bit-precision for **low-level** and **bitwise** operations

# Goals for the OpenQASM 3.0 Translator



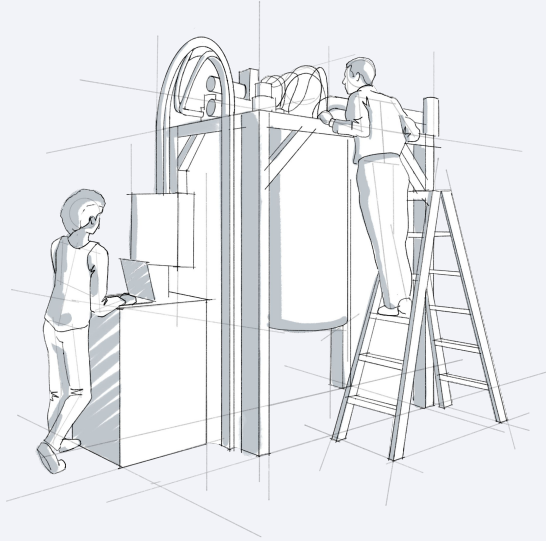
## Goals

- Having a working translator
- Support most of the OpenQASM3 features
- Provide feedback on the proposed reference OpenQASM3 parser implementation (AWS team)
- Provide feedback to the OpenQASM3 TSC about the language specification

## Non- Goals

- Having an **efficient** translator
- Implementation of **all** the OpenQASM3 specification

# Out-of-scope OpenQASM3 features



## Defcal Grammars

- Completely different language embedded in the OpenQASM3 grammar
- Integration with a lot of components required at the same time (Qiskit-terra, Qiskit-Pulse, Qiskit-Runtime, etc)
- Can be implemented at a later stage when the **openpulse** grammar is mature

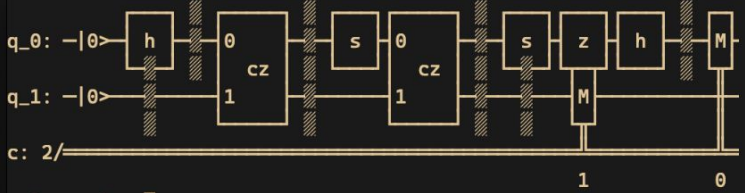
## Timing & duration

- **delay[t]**, **stretch**, **box**, and **duration**
- Computing **stretch** is hard: multi-objective linear programming problem

# Demo

```
...> ^qft.qasm > ^qpt.qasm > ^rb.qasm > ^rus.qasm >... buffers (phi|tests|psi) python3 build_ast.py ../../examples/rb.qasm -I ../../examples/ -t
```

```
20 // One randomized benchmarking sequence
19 OPENQASM 3;
18 include "stdgates.inc";
17
16 qubit[2] q;
15 bit[2] c;
14
13 reset q;
12 h q[0];
11 barrier q;
10 cz q[0], q[1];
9 barrier q;
8 s q[0];
7 cz q[0], q[1];
6 barrier q;
5 s q[0];
4 z q[0];
3 h q[0];
2 barrier q;
1 measure q[0] -> c[0];
21 measure q[1] -> c[1];
```

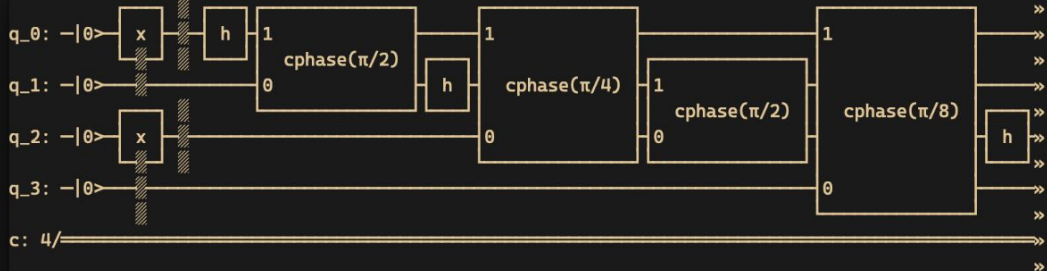


(phi|tests|psi) █

# Demo

```
...> ^1pong.qasm ^2qec.qasm ^3qft.qasm ^4qpt.qasm ^5... buffers <| tests |> python3 build_ast.py ../../examples/qft.qasm -I ../../examples/ -t
```

```
20 // quantum Fourier transform
19 OPENQASM 3;
18 include "stdgates.inc";
17 qubit[4] q;
16 bit[4] c;
15 reset q;
14 x q[0];
13 x q[2];
12 barrier q;
11 h q[0];
10 cphase(pi / 2) q[1], q[0];
9 h q[1];
8 cphase(pi / 4) q[2], q[0];
7 cphase(pi / 2) q[2], q[1];
6 h q[2];
5 cphase(pi / 8) q[3], q[0];
4 cphase(pi / 4) q[3], q[1];
3 cphase(pi / 2) q[3], q[2];
2 h q[3];
1 c[0] = measure q[0];
21 c[1] = measure q[1];
1 c[2] = measure q[2];
2 c[3] = measure q[3];
```



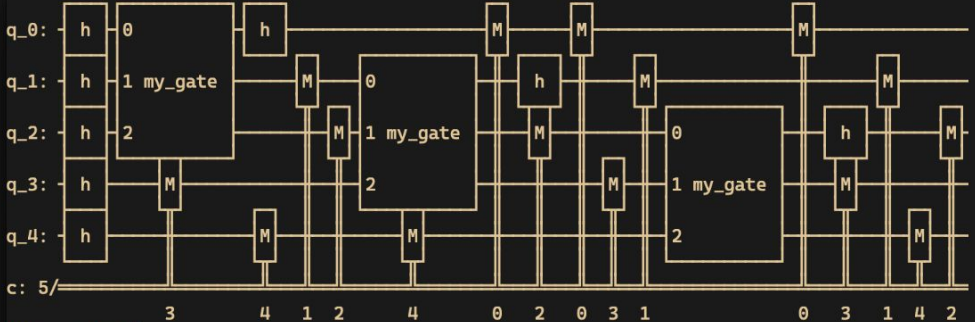
```
q_0: -|0>
q_1: -|0>
q_2: -|0>
q_3: -|0>
c: 4/

«
«q_0:
«
«q_1: 1
«
«q_2: cphase(pi/4)
«
«q_3: 0
«c: 4/
«
0 1 2 3
```

# Demo

```
test.qasm buffers  $\langle \phi | \text{tests} | \psi \rangle$  python3 build_ast.py ../src/openqasm/translator/tests/test.qasm -I ../../examples/ -t
```

```
1 OPENQASM 3.0;
2 include "stdgates.inc";
3
4 gate my_gate a, b, c {
5   cx a, b;
6   cx b, c;
7 }
8
9 const n = 5;
10 qubit[n] q;
11 uint[n] c;
12
13 h q;
14
15 for i in [0: n-2] {
16   my_gate q[i], q[i+1], q[i+2];
17   h q[i];
18   for j in [0: n] { c[j] = measure q[j]; }
19 }
```



$\langle \phi | \text{tests} | \psi \rangle$



```

*adder.qasm *alignment.qasm *cphase.qasm *dd.qasm >... buffers (|tests|ψ) python3 build_ast.py ../../examples/adder.qasm -I ../../examples/ -t
18 */
17 OPENQASM 3;
16 include "stdgates.inc";
15
14 gate majority a, b, c {
13   cx c, b;
12   cx c, a;
11   ccx a, b, c;
10 }
9
8 gate unmaj a, b, c {
7   ccx a, b, c;
6   cx c, a;
5   cx a, b;
4 }
3
2 qubit[1] cin;
1 qubit[4] a;
22 qubit[4] b;
1 qubit[1] cout;
2 bit[5] ans;
3 uint[4] a_in = 1; // a = 0001
4 uint[4] b_in = 15; // b = 1111
5 // initialize qubits
6 reset cin;
7 reset a;
8 reset b;
9 reset cout;
10
11 // set input states
12 for i in [0: 4] {
13   if(bool(a_in[i])) x a[i];
14   if(bool(b_in[i])) x b[i];
15 }
16 // add a to b, storing result in b
17 majority cin[0], b[0], a[0];
18 for i in [0: 2] { majority a[i], b[i + 1], a[i + 1]; }
19 cx a[3], cout[0];
20 for i in [2: -1: 0] { unmaj a[i], b[i+1], a[i+1]; }
21 unmaj cin[0], b[0], a[0];
22 measure b[0] -> ans[0];
23 measure b[1] -> ans[1];
24 measure b[2] -> ans[2];
25 measure cout[0] -> ans[4];
                
```

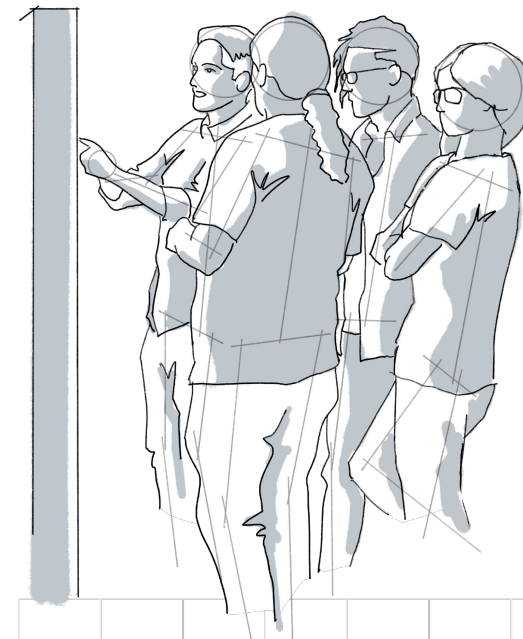
The diagram illustrates a quantum circuit for adding two 4-bit numbers. It features several 'majority' and 'unmaj' gates. The majority gate takes three bits (a, b, c) and outputs their majority. The unmaj gate takes three bits and outputs their minority. The circuit starts with a carry-in (cin\_0) and processes bits a\_0 through a\_3 and b\_0 through b\_3. It uses CX gates to propagate carries and measures the final carry-out (cout\_0) and bits b[0], b[1], b[2] to produce the answer (ans).

# How are we making a change?

Impact of “QAMP” #23 on the shaping of OpenQASM3 compiler Development

Under the guidance of Technical Steering Committee (TSC)

- Providing an **end-user’s perspective** to the Technical Steering Committee (TSC)
- **Reporting inconsistencies** in the Language Specification and Examples
- **Tightening** the AST implementation of OpenQASM3
- Building a platform for **translating** and **running** OpenQASM3 code with the existing Qiskit specifications
- **Implementing features** from the current grammar specification for OpenQASM3



# How are we making a change?

- Tight integration with TSC-Approved OpenQASM3 AST

- Tons of bug fixes while working on the Translator

**Open** Casting angles to floats - slow and precise or quick and lossy? #280  
mbhealy opened this issue 13 days ago · 4 comments

would really appreciate the ability to choose to precision and provide a specific operator or indicator to ensure better precision. In C, I have seen where people will explicitly encode the floating point number they want to ensure speed&accuracy. Something like that might be a useful addition as well.

Hope this helps some :)

**jwoehr** commented 2 days ago Contributor

Some folks concerned with this issue might be interested in QAMP Fall 2021 Team No. 23's ways of handling this which will be presented in our Oct. 7 session for QAMP checkpoint.

**Open** Reference implementation of OpenQASM 3.0 AST in Python #269  
aspcompiler wants to merge 53 commits into @qiskit/master from aspcompiler/ast

**AbeerVaishnav13** commented 7 days ago

Hi @aspcompiler, we were in the process of using your QASM3 to AST generator for the Qiskit Advocate Mentorship Program project on OpenQASM3 and noticed a few bugs/potential changes...

- 1. No restriction on re-assignment to `const` types**

Presently there are no restrictions on re-assignment to `const` types. For example, if I write the following code:

```
const my_const = 10; // parses as a 'ConstantDeclaration' statement
my_const = 5; // parses as a 'ClassicalAssignment' statement - WRONG
```

Shouldn't the AST gen step throw an error for statement-2 i.e. re-assignment to an immutable identifier?

- 2. 'bool' instead of `NoDesignatorTypeName.bool`**

**Open** Reference implementation of OpenQASM 3.0 AST in Python #269  
aspcompiler wants to merge 53 commits into @qiskit/master from aspcompiler/ast

**nelimee** commented 19 days ago

Hi @aspcompiler,

We started to use your AST generator for the Qiskit Advocate Mentorship Program project on OpenQASM3 and found some things that might be considered as issues.

Here is a quick description of each, sadly I did not have the time to investigate the source of these issues yet:

```
# Code to get the AST from one of the example files in https://github.com/aspcompiler/openqasm/tree/ast/examples
from openqasm.parser.antlr.qasm_parser import parse

file = "adder.qasm"
with open(file, "r") as f:
    qasm_str = f.read()

ast = parse(qasm_str)
```

**Open** Reference implementation of OpenQASM 3.0 AST in Python #269  
aspcompiler wants to merge 53 commits into @qiskit/master from aspcompiler/ast

**jakeshman** commented 7 days ago · edited · Contributor

@AbeerVaishnav13, some quick answers, since I'm looking at this (but am not on the main author team).

- 1. reassignment to const:** that probably isn't the role of the AST to throw an error on that, because AST generation is more about syntax than programme correctness. The first pass of AST generation (which the reference implementation here is) doesn't generally track which identifiers are being assigned, and what their types are - that would most likely come in a later, compilation stage.
- 2. cast operator doesn't preserve enum:** that looks like a bug to me.

**Fixed NoDesignatorType parsing** ✓ @93ddd

**aspcompiler** commented 7 days ago Contributor Author

@AbeerVaishnav13: @jakeshman just answered 1. I just pushed a fix for 2.

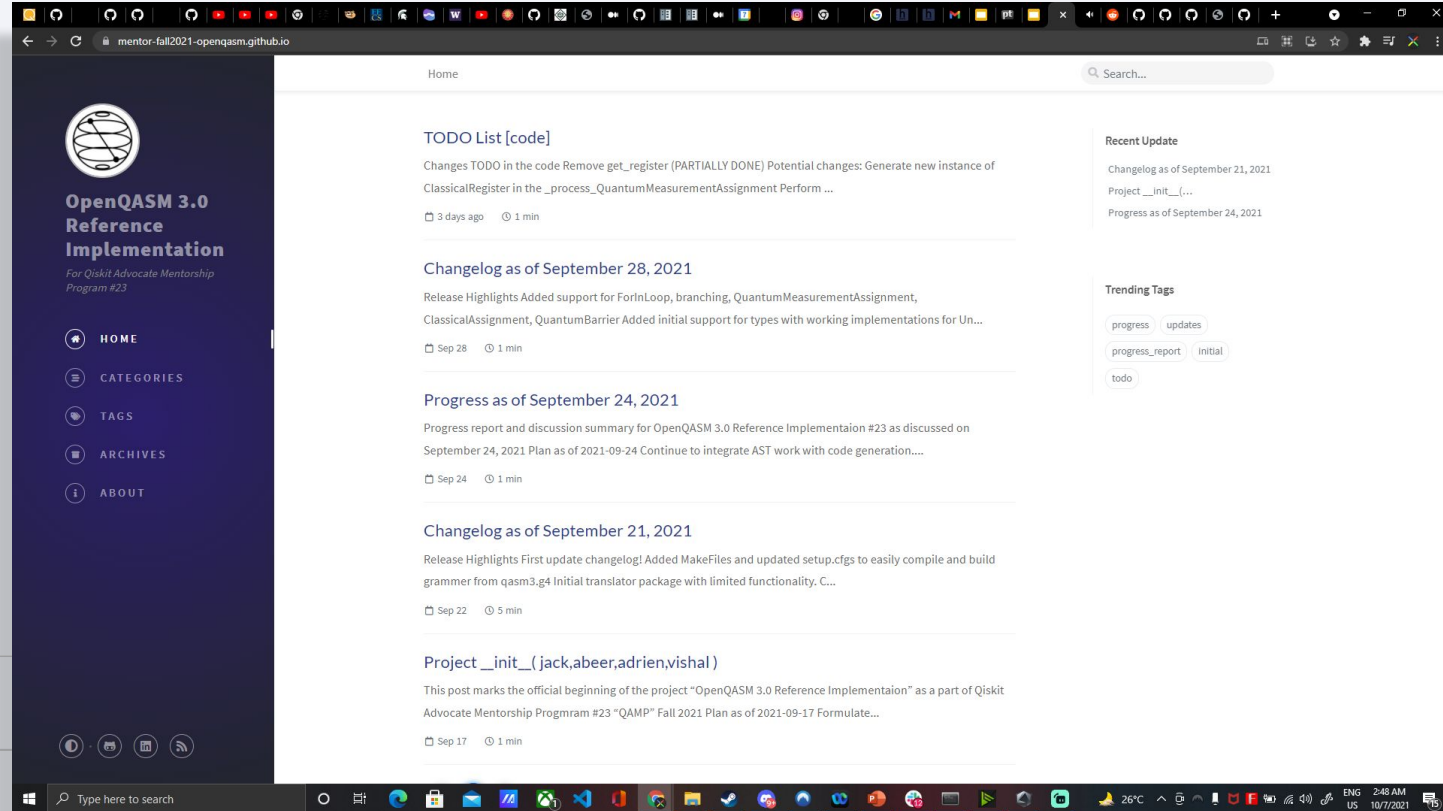
**shiyunon** and others added 5 commits 18 days ago

- fixed some issues raised in #269 ✗ ad0a9ce
- Small subtle adjustment to the identifier list and constant declaration ✗ 347841c
- adding a code snippet from adder.qasm as a new test case. ✗ 5696e17
- Merge branch 'master' into ast 2ccb56a
- Removed fixed type to sync up with grammar ✓ f380a95

# Dedicated blog website

Check it out here:

<https://mentor-fall2021-openqasm.github.io/>



The screenshot shows a web browser displaying the homepage of the OpenQASM 3.0 Reference Implementation blog. The page features a dark blue sidebar on the left with the Qiskit logo and navigation links for HOME, CATEGORIES, TAGS, ARCHIVES, and ABOUT. The main content area is white and contains several blog posts. The top post is titled 'TODO List [code]' and discusses changes to the code, including removing 'get\_register' and adding support for 'ForinLoop'. The second post is titled 'Changelog as of September 28, 2021' and highlights support for 'ForinLoop' and 'QuantumBarrier'. The third post is titled 'Progress as of September 24, 2021' and reports on the progress of the implementation. The fourth post is titled 'Changelog as of September 21, 2021' and mentions updates to 'MakeFiles' and the 'qasm3.g4' translator. The fifth post is titled 'Project \_\_init\_\_(jack,abeer,adrien,vishal)' and marks the official beginning of the project. A right-hand sidebar contains 'Recent Update' and 'Trending Tags' sections.

Home

Search...

### TODO List [code]

Changes TODO in the code Remove `get_register` (PARTIALLY DONE) Potential changes: Generate new instance of `ClassicalRegister` in the `_process_QuantumMeasurementAssignment` Perform ...

3 days ago 1 min

---

### Changelog as of September 28, 2021

Release Highlights Added support for `ForinLoop`, branching, `QuantumMeasurementAssignment`, `ClassicalAssignment`, `QuantumBarrier` Added initial support for types with working implementations for Un...

Sep 28 1 min

---

### Progress as of September 24, 2021

Progress report and discussion summary for OpenQASM 3.0 Reference Implementaion #23 as discussed on September 24, 2021 Plan as of 2021-09-24 Continue to integrate AST work with code generation....

Sep 24 1 min

---

### Changelog as of September 21, 2021

Release Highlights First update changelog! Added `MakeFiles` and updated `setup.cfg`s to easily compile and build grammer from `qasm3.g4` Initial translator package with limited functionality, C...

Sep 22 5 min

---

### Project \_\_init\_\_(jack,abeer,adrien,vishal)

This post marks the official beginning of the project "OpenQASM 3.0 Reference Implementaion" as a part of Qiskit Advocate Mentorship Program #23 "QAMP" Fall 2021 Plan as of 2021-09-17 Formulate...

Sep 17 1 min

Recent Update

Changelog as of September 21, 2021

Project \_\_init\_\_(...

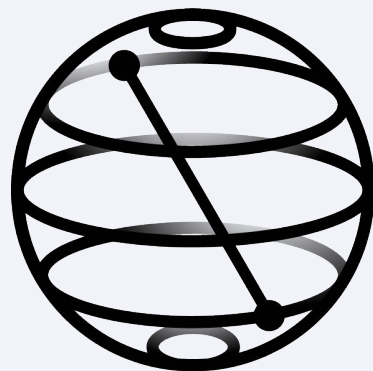
Progress as of September 24, 2021

Trending Tags

progress updates

progress\_report initial

todo



# Thank You!