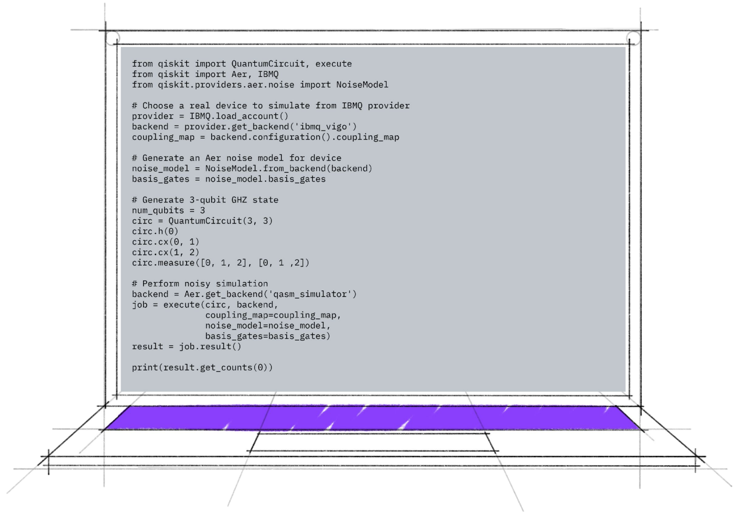
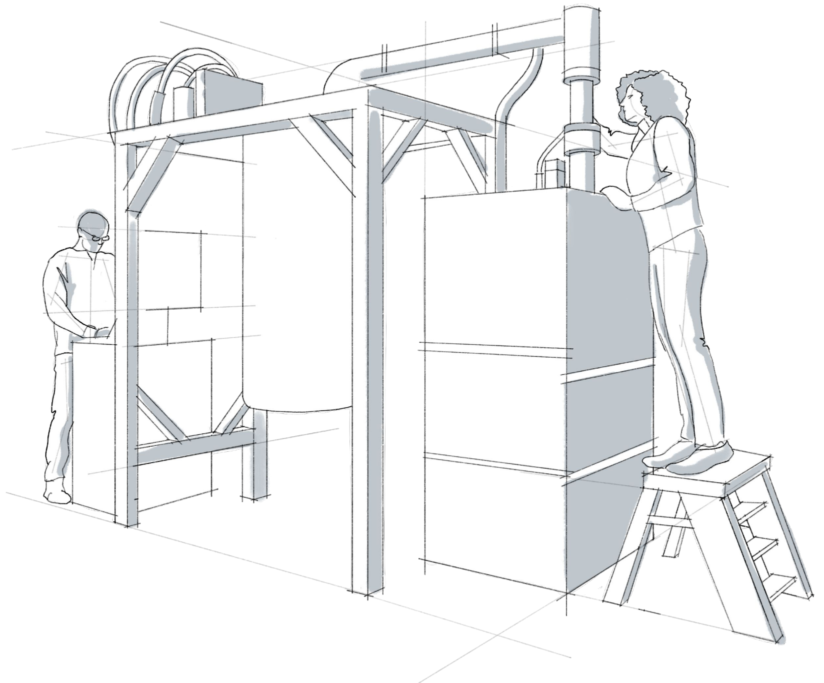
#23: OpenQASM 3.0 Reference Python Implementation | “QAMP-21”

Mentored by – Jack Woehr

Abeer Vaishnav, Adrien Suau, Vishal Bajpe



```
from qiskit import QuantumCircuit, execute
from qiskit import Aer, IBMQ
from qiskit.providers.aer.noise import NoiseModel

# Choose a real device to simulate from IBMQ provider
provider = IBMQ.load_account()
backend = provider.get_backend('ibmq_vigo')
coupling_map = backend.configuration().coupling_map

# Generate an Aer noise model for device
noise_model = NoiseModel.from_backend(backend)
basis_gates = noise_model.basis_gates

# Generate 3-qubit GHZ state
num_qubits = 3
circ = QuantumCircuit(3, 3)
circ.h(0)
circ.cx(0, 1)
circ.cx(1, 2)
circ.measure([0, 1, 2], [0, 1 ,2])

# Perform noisy simulation
backend = Aer.get_backend('qasm_simulator')
job = execute(circ, backend,
              coupling_map=coupling_map,
              noise_model=noise_model,
              basis_gates=basis_gates)
result = job.result()

print(result.get_counts(0))
```



Why OpenQASM?

Qiskit

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

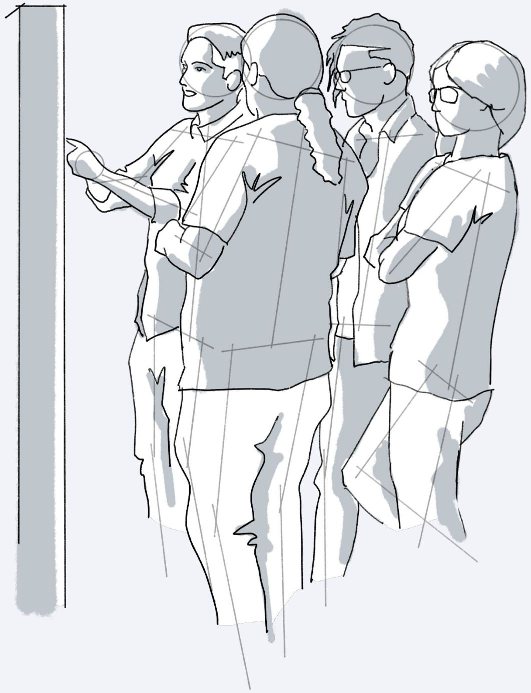
Support for **`int`**, **`uint`**, **`float`**, **`bool`**, and **`bit`** for classical types with functionality to specify a type with exact bit-precision for **low-level** and **bitwise** operations

Qiskit

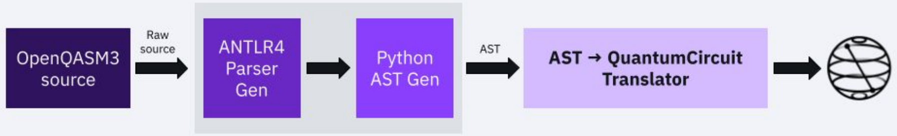
Checkpoint #3 Update

Qiskit

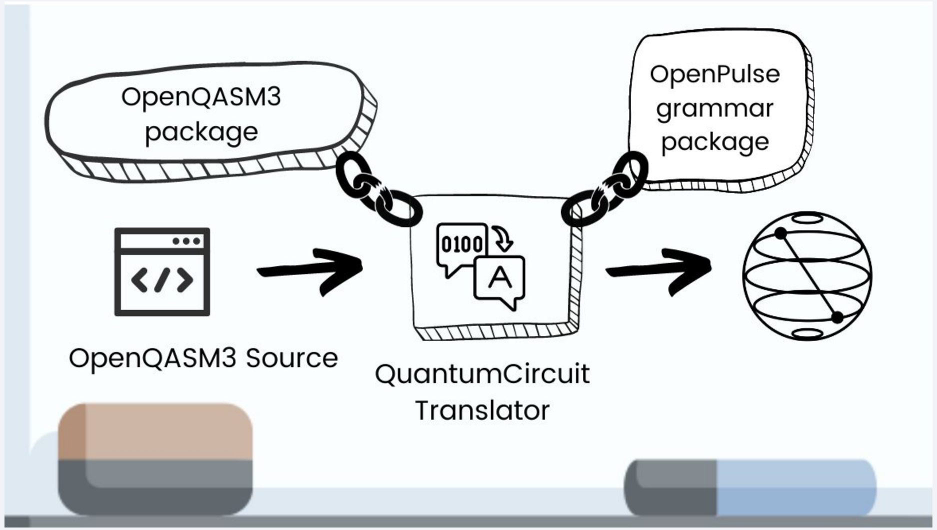
Architecture



Current



Proposed `defcal` support (incomplete)

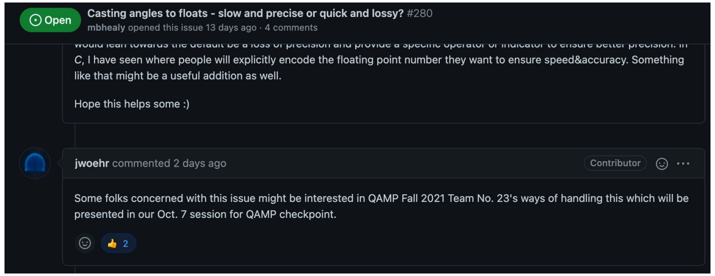


IBM Quantum / © 2021 IBM Corporation

How are we bringing about changes?

- Tight integration with the OpenQASM 3 Technical Steering Committee (TSC) OpenQASM3 reference AST

- Several contributions to Qiskit/openqasm repo 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

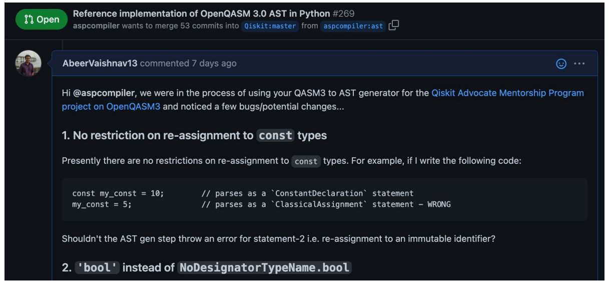
C, I have seen where people will explicitly encode the floating point number they want to ensure speed&accuracy. Something like that might be a useful addition as well.

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.

👍 2



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...

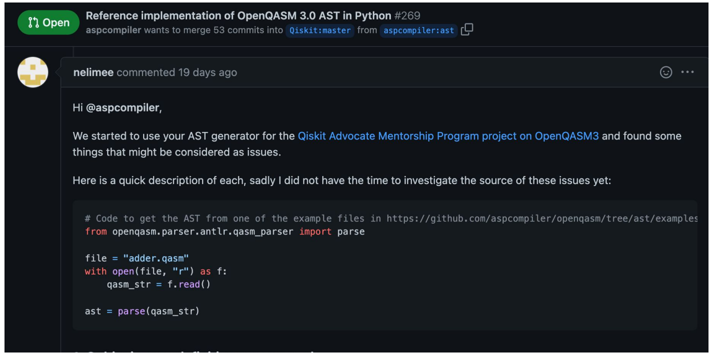
1. No restriction on re-assignment to `const` types

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`

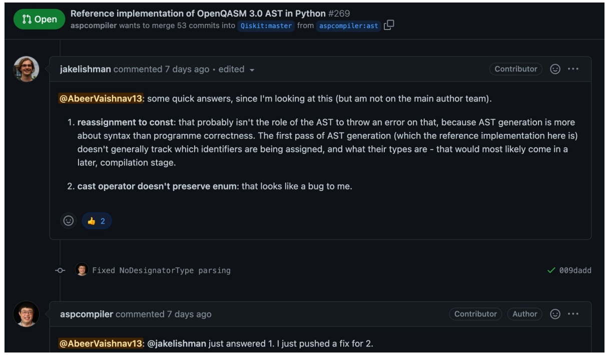
nelimee commented 19 days ago

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`

jakelishman commented 7 days ago • edited — Contributor

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

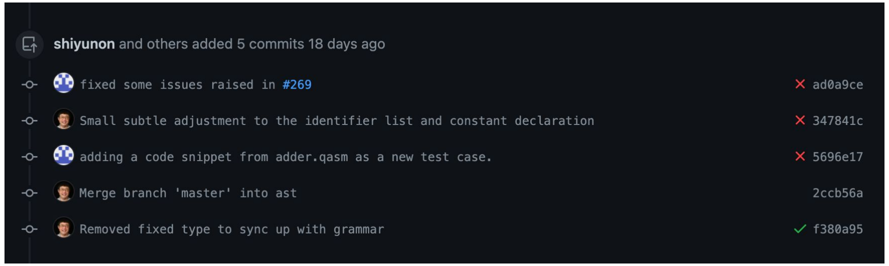
1. **reassignment to const**: that probably isn't the role of the AST to throw an error on that, because AST generation is more about syntax than programme correctness. The first pass of AST generation (which the reference implementation here is) doesn't generally track which identifiers are being assigned, 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.

👍 2

```
Fixed NoDesignatorType parsing                                  ✓ 009dadd
```

aspcompiler commented 7 days ago — Contributor — Author

@AbeerVaishnav13: **@jakelishman** 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
```

Qiskit/openqasm Pull Request #269: https://github.com/Qiskit/openqasm/pull/269

PR Contributions

PR#295 :

- Identifying an important hole in the specification about mathematical functions
- Addition of the power (**) operator for complex numbers

PR#269 :

- Extensive testing on various inputs
- Remarks to improve overall ease of use
- A few defects raised to the attention of the authors

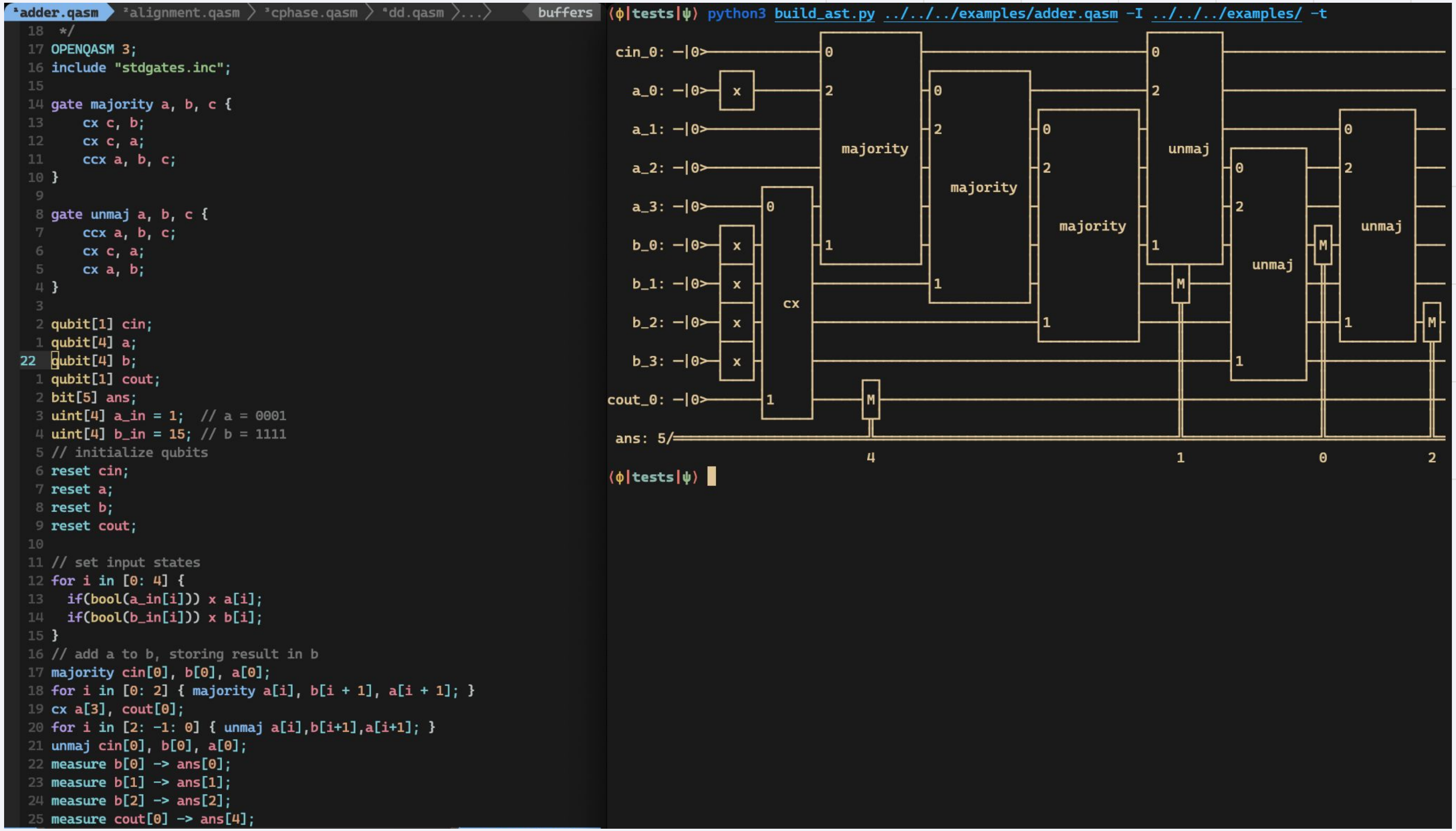
PR#296 :

- Following the progressive definition of OpenPulse grammar
- Waiting for advances from the main team

PR#1 (jakelishman/openqasm):

- Automatic packaging of the OpenQASM 3.0 parser using GitHub Actions
- Waiting for the “openqasm” package to be available

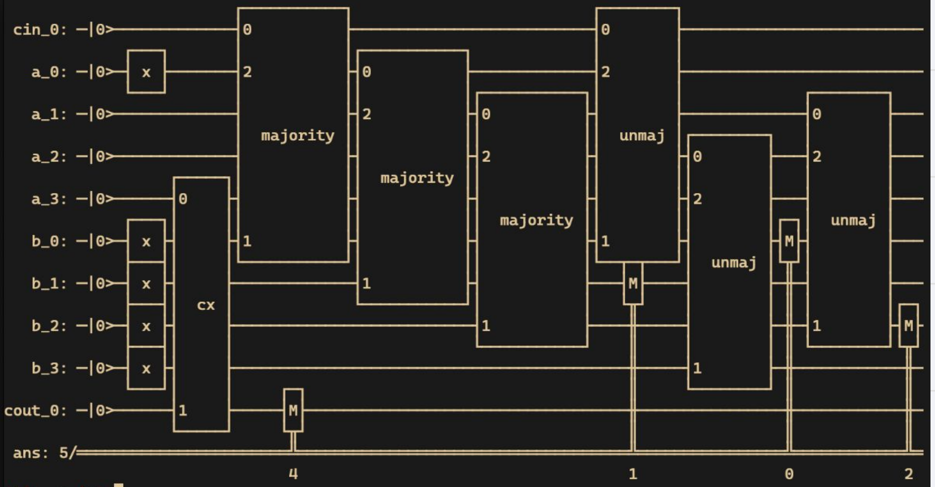
Demo



```
¹adder.qasm  ²alignment.qasm  ³cphase.qasm  ⁴dd.qasm  ...   buffers
```

```
 */
OPENQASM 3;
include "stdgates.inc";

gate majority a, b, c {
    cx c, b;
    cx c, a;
    ccx a, b, c;
}

gate unmaj a, b, c {
    ccx a, b, c;
    cx c, a;
    cx a, b;
}

qubit[1] cin;
qubit[4] a;
qubit[4] b;
qubit[1] cout;
bit[5] ans;
uint[4] a_in = 1;  // a = 0001
uint[4] b_in = 15; // b = 1111
// initialize qubits
reset cin;
reset a;
reset b;
reset cout;

// set input states
for i in [0: 4] {
  if(bool(a_in[i])) x a[i];
  if(bool(b_in[i])) x b[i];
}
// add a to b, storing result in b
majority cin[0], b[0], a[0];
for i in [0: 2] { majority a[i], b[i + 1], a[i + 1]; }
cx a[3], cout[0];
for i in [2: -1: 0] { unmaj a[i],b[i+1],a[i+1]; }
unmaj cin[0], b[0], a[0];
measure b[0] -> ans[0];
measure b[1] -> ans[1];
measure b[2] -> ans[2];
measure cout[0] -> ans[4];
```

```
⟨ϕ|tests|ψ⟩ python3 build_ast.py ../../../examples/adder.qasm -I ../../../examples/ -t
```

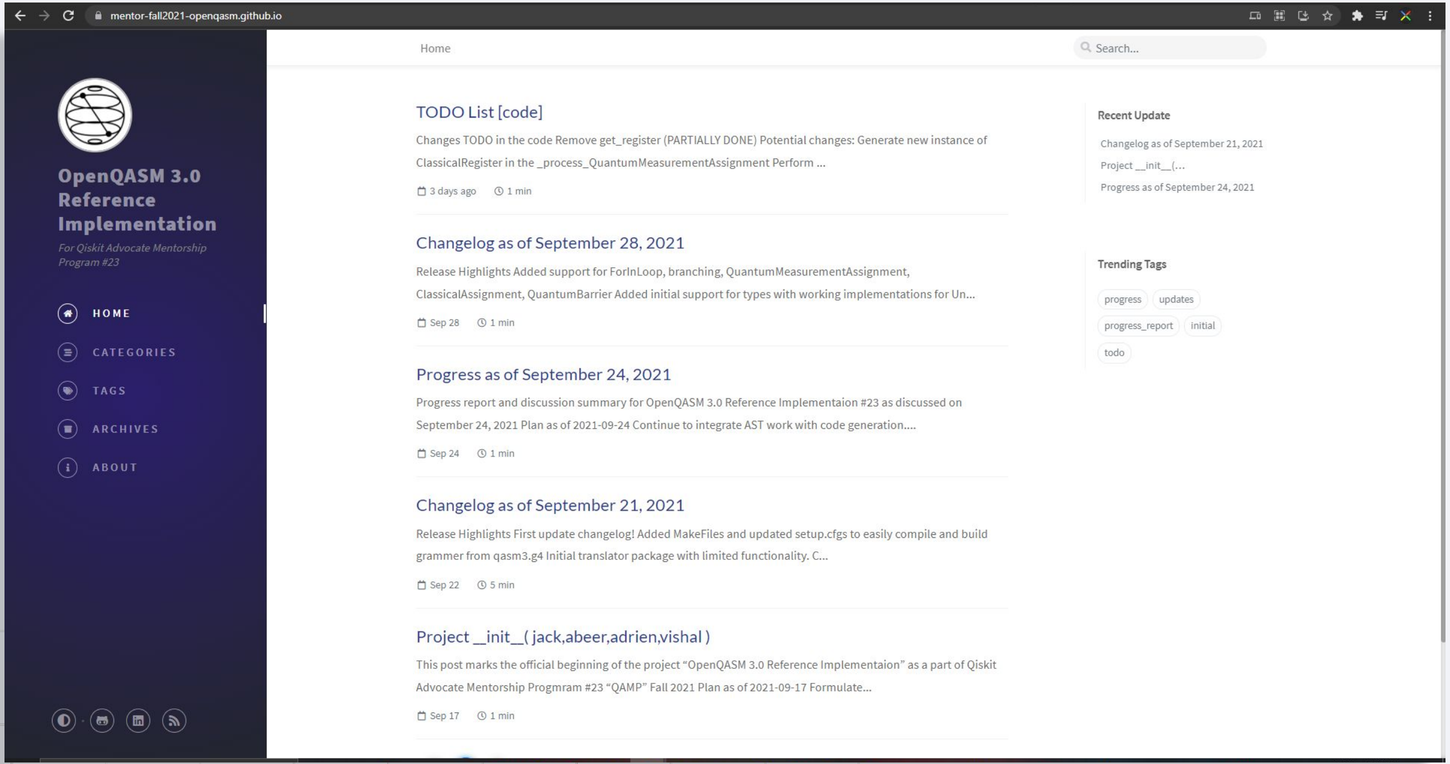


```
⟨ϕ|tests|ψ⟩ █
```

Dedicated blog website

Check it out here:

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



mentor-fall2021-openqasm.github.io

OpenQASM 3.0 Reference Implementation

For Qiskit Advocate Mentorship Program #23

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.cfgs 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 Progmram #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__(...

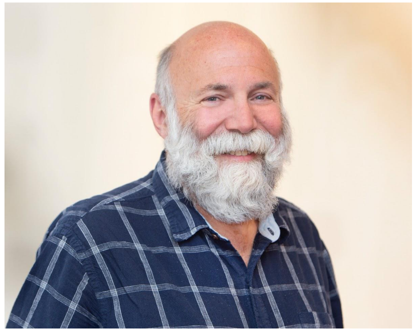
Progress as of September 24, 2021

Trending Tags

progress updates progress_report initial todo

QAMP #23 - Team

- Jack Woehr (Mentor)
IBM Champion 2021



- Abeer Vaishnav

- Adrien Suau

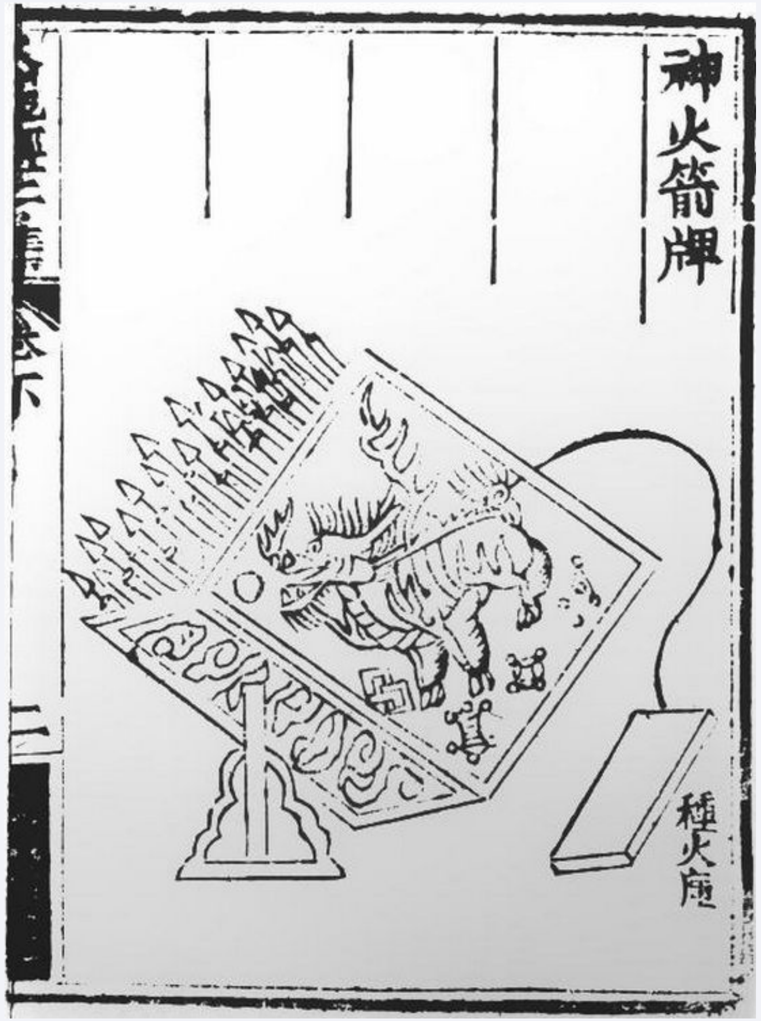
- Vishal Bajpe

Today
Is
Yesterday’s Tomorrow

In the course of 20 years’ dilettante interest in Quantum Computing, I have seen the field progress from what Nobel laureate Bill Phillips called “A 50-50 chance: 50% chance in 50 years” to a world-wide collaborative research project involving the brightest young minds of every habitable continent.

It has been a pleasure and a privilege to participate in QAMP Fall 2021 with this stellar team! Best of luck in your careers!

- Jack Woehr



Depiction of a fire arrow rocket launcher, or shen huo chien phai, from the Ming Dynasty book Huo Long Jing Wikimedia Public Domain

Thank you

for listening