Documentation for developers

This explanation serves as documentation for developer. File names are added with their paths relative to openql root directory. Addition of a feature mainly requires:

- Interface specification in openql/openql.h
- API Documentation in openql/openql.i
- C++ implementation in ql/*.h
- If the feature requires implementation related to some architecture (cbox, cc, cc_light), then update corresponding arch/*.h file
- Update of ql/options.h file to add an option, if required
- Add/update python tests in tests/test_*.py

In the following, a specific example is mentioned in which support for controlled gate is added to openql.

Example: Support for controlled kernel generation

This document explains the steps taken to add the necessary support for controlled kernel generation.

- openql/openql.h file was updated to add the following API in the Kernel class:

```
void controlled(Kernel &k,
    std::vector<size_t> control_qubits,
    std::vector<size_t> ancilla_qubits)
{
    kernel->controlled(k.kernel, control_qubits, ancilla_qubits);
}
```

- Following was added to openql/openql.i for API documentation:

```
%feature("docstring") Kernel::controlled
""" generates controlled version of the kernel from the input kernel.

Parameters
----------

arg1 : ql::Kernel
    input kernel. Except measure, Kernel to be controlled may contain a

arg2 : []
    list of control qubits.

arg3 : []
    list of ancilla qubits. Number of ancilla qubits should be equal to
```

- Bulk of the implemenation to actually support controlled kernel generation logic was added to ql/kernel.h file in the function `controlled()`. This in turn required addition of single and multi qubit controlled logic generation functions

(in the same file ql/kernel.h).

-- `controlled_single()` implements controlled version of gates when single control qubit is involved. The circuit is extracted from the kernel and controlled version of indicidual gate is generated.

-- In case of multi-qubit control, a network based on Fig. 4.10, p.p 185, Nielson & Chuang is utilized.

- This feature did not require any changes specific to any backend architecutre, so no ql/arch/*.h was modified.
- This feature did not require any addition of options provided to user, so ql/options.h is not modified.
- test_controlled.py was written to contain tests for controlled kernel implementation.