Qubit control via RFSoC

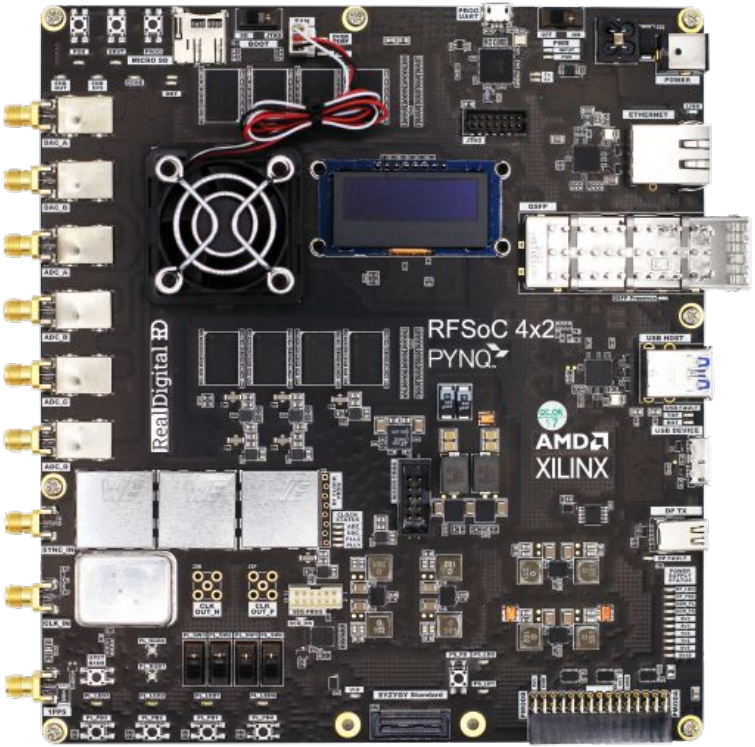
Joint QubIT PNRR October meeting (25/10/2023)

Rodolfo Carobene

Qibosoq software

- Qibosoq: an open-source framework for quantum circuit RFSoC programming,
  https://arxiv.org/abs/2310.05851

- Qibosoq software (GitHub),
  https://github.com/qiboteam/qibosoq



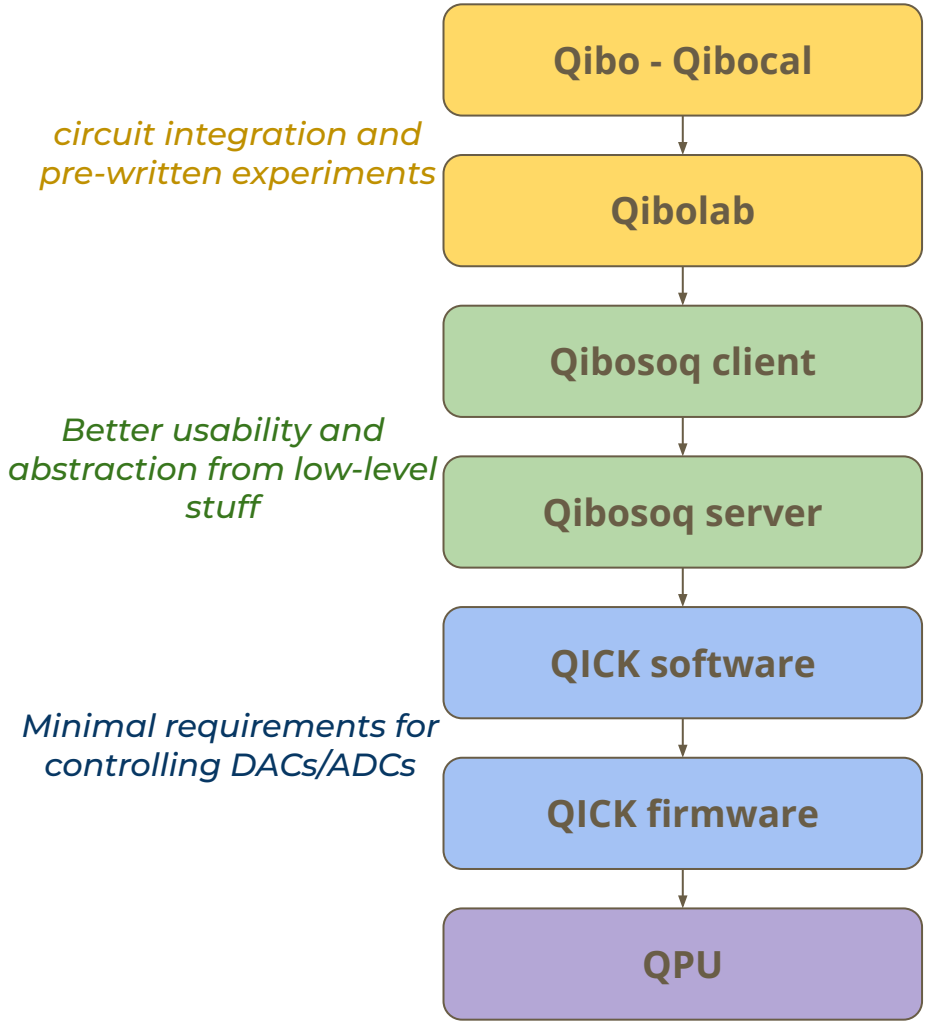
- Tested with RFSoC4x2, ZCU111, ZCU216
- Tested with 1 and 3 qubits

Qibosoq features

- Generation of arbitrary pulses
- “Qubit measurements” (pulse + delayed acquisition)

- Modulated and unmodulated acquisition
- Averaged and non-averaged results

- Limited memory for different pulse shapes



Experiment deployment

Remote client

On board server

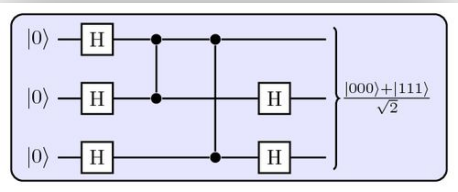
Circuit definition → **Transpilation** → **Hardware deployment** → **Experiment execution**

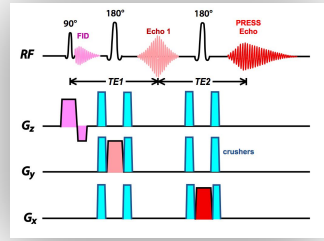
Qibo

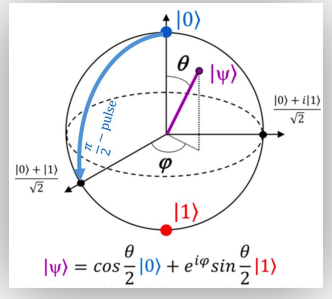
Qibolab

Qibosoq + QICK

From doi.org/10.1017/CBO9780511976667







Qibosoq installation

qibo.science/qibosoq/stable/getting-started/index.html

$ pip install qibosoq (remote client and on-board server)

remote .bashrc

```
# server address
export QIBOSOQ_HOST=192.168.0.81
# server port
export QIBOSOQ_PORT=6000
# bitstream path
export QIBOSOQ_BITSTREAM=/home/xilinx/jupyter_notebooks/qick_111_rfbv1_mux.bit
# is the readout multiplexed?
export QIBOSOQ_IS_MULTIPLEXED=True
```

Qibosoq examples

Pulses directly from qibosoq

```
pulse = Rectangular(
        frequency = 7000, #MHz
        amplitude = 0.5,
        relative_phase = 0,
        start_delay = 0,
        duration = 1,
        name = "readout_pulse",
        type = "readout",
        dac = 1,
        adc = 0
)

sequence = [pulse]
config = Config(
        repetition_duration=50,
        adc_trig_offset=200,
        reps=1000,
        average=True
)
qubit = Qubit()

server_commands = {
   "operation_code": OperationCode.EXECUTE_PULSE_SEQUENCE,
   "cfg": config,
   "sequence": sequence,
   "qubits": [qubit],
}
results = []
for freq in frequencies:
   server_commands["sequence"][0].frequency = freq
   i, q = execute(server_commands, HOST, PORT)
   results.append(np.abs(np.array(i[0][0]) + 1j * np.array(q[0][0])))
```

Pulses directly from qibolab

```
readout_pulse = platform.create_MZ_pulse(qubit=0, start=0)
readout_pulse.amplitude = 0.5
sequence.add(readout_pulse)

options=ExecutionParameters(
   nshots=1000,
   relaxation_time=50,
   acquisition_type=AcquisitionType.INTEGRATION,
   averaging_mode=AveragingMode.CYCLIC,
)
sweeper = Sweeper(
   parameter=Parameter.frequency,
   values=np.arange(-2e8, +2e8, 1e6),
   pulses=[readout_pulse],
   type=SweeperType.OFFSET,
)

results = platform.sweep(sequence, options, sweeper)

frequencies = np.arange(-2e8, +2e8, 1e6) + readout_pulse.frequency
plt.plot(frequencies, amplitudes)
```

Pulses from qibocal

```
- id: resonator high power
  priority: 0
  operation: resonator_spectroscopy
  parameters:
    power_level: high
    freq_width: 400_000_000
    freq_step: 1_000_000
    amplitude: 0.5
    nshots: 1000
```

Current/short-term development

- Development of more experiments compatible with the Qibo/Qibocal interface
- Detach acquisition and measurements
- Documentation improvements

qibo.science/qibosoq/stable

- Installation instructions
- Usage instructions
- Communication protocol
- Biases and Pulses
- Sweepers
- Examples
- Qibosoq - Qibolab - Qibocal