Qutritium: ternary quantum computation with superconducting qutrits

Qutritium Development Team

Hanoi, 14th June 2023

(well, all presentations are time-reversal invariant, so this doesn't matter much)

Part I

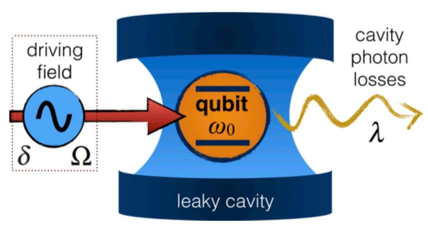
Why we are building Qutritium

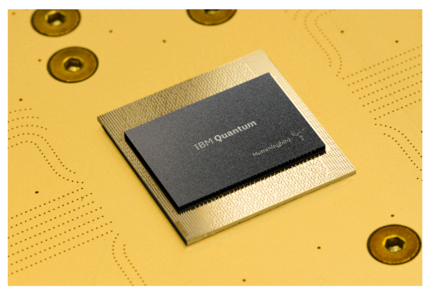
Unexplored avenue: higher dimensions of the Hilbert space

To get a better view, climb higher!

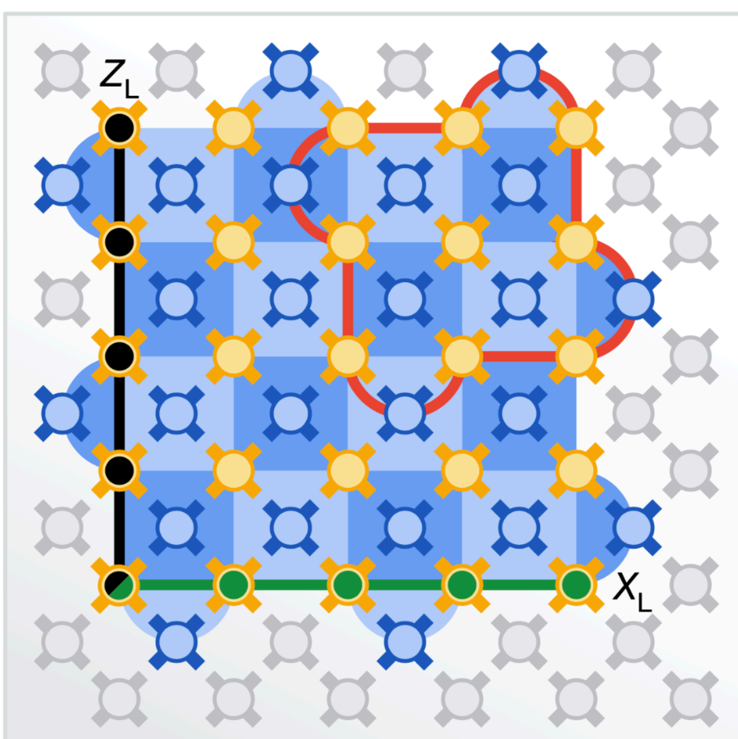
Pathways toward universal quantum advantage

Cavity QED (S. Haroche, ENS Paris)





IBQM's Hummingbird quantum processor



Surface code lattice d^2 qubits QEC (Nature 2023)

Multiple approaches are being actively pursued. *All paths lead to Rome*:

1. Noise suppression techniques
2. Intrinsically novel qubit designs

   Fluxonium (Long et al. PRX 2019), 0-π qubit (Gyenis et al. PRX Quantum 2021), CV qubits, Topologically protected qubits (Microsoft).
3. Quantum error correction

   Theory: Shor, Kitaev, Gottesman, you name it; Experiments: Three-qubit correction (Reed et al. Nature 2012), Surface code (Marques et al. Nature 2022), Surface code at break-even point for cQED (Google Quantum AI 2023).
4. Higher dimensions of Hilbert space

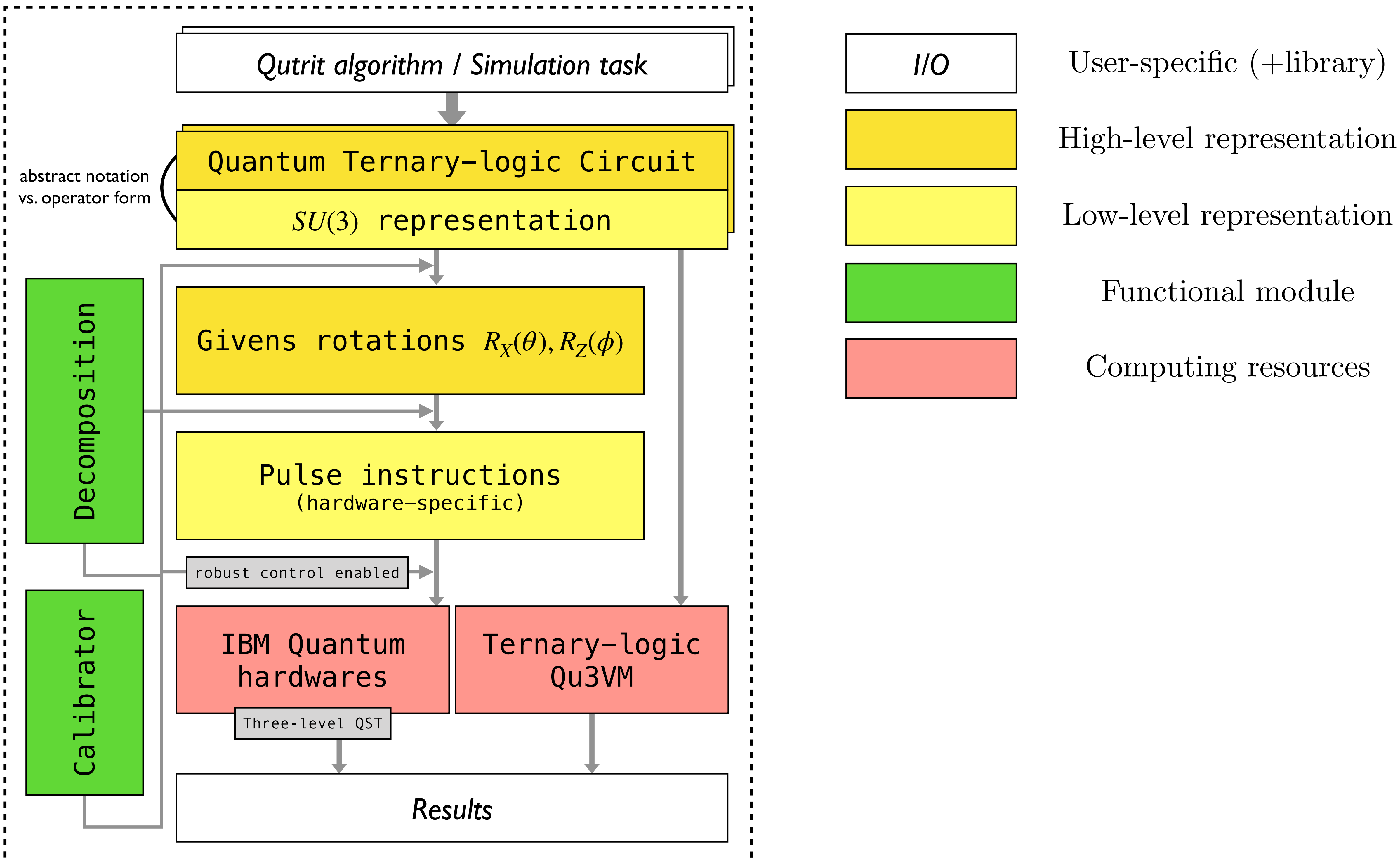
   Improved QEC threshold (Campbell PRL 2014, Majumdar PRA 2018), less redundancy, efficient data encoding & circuit compilation (Kapit PRL 2016).
   *(+) Some **hints** of quantum advantage (Gedik 2015).*

Part II

How we envision Qutritium

The one-stop station for qutrit enthusiasts

high-dimensional quantum computation with superconducting circuits

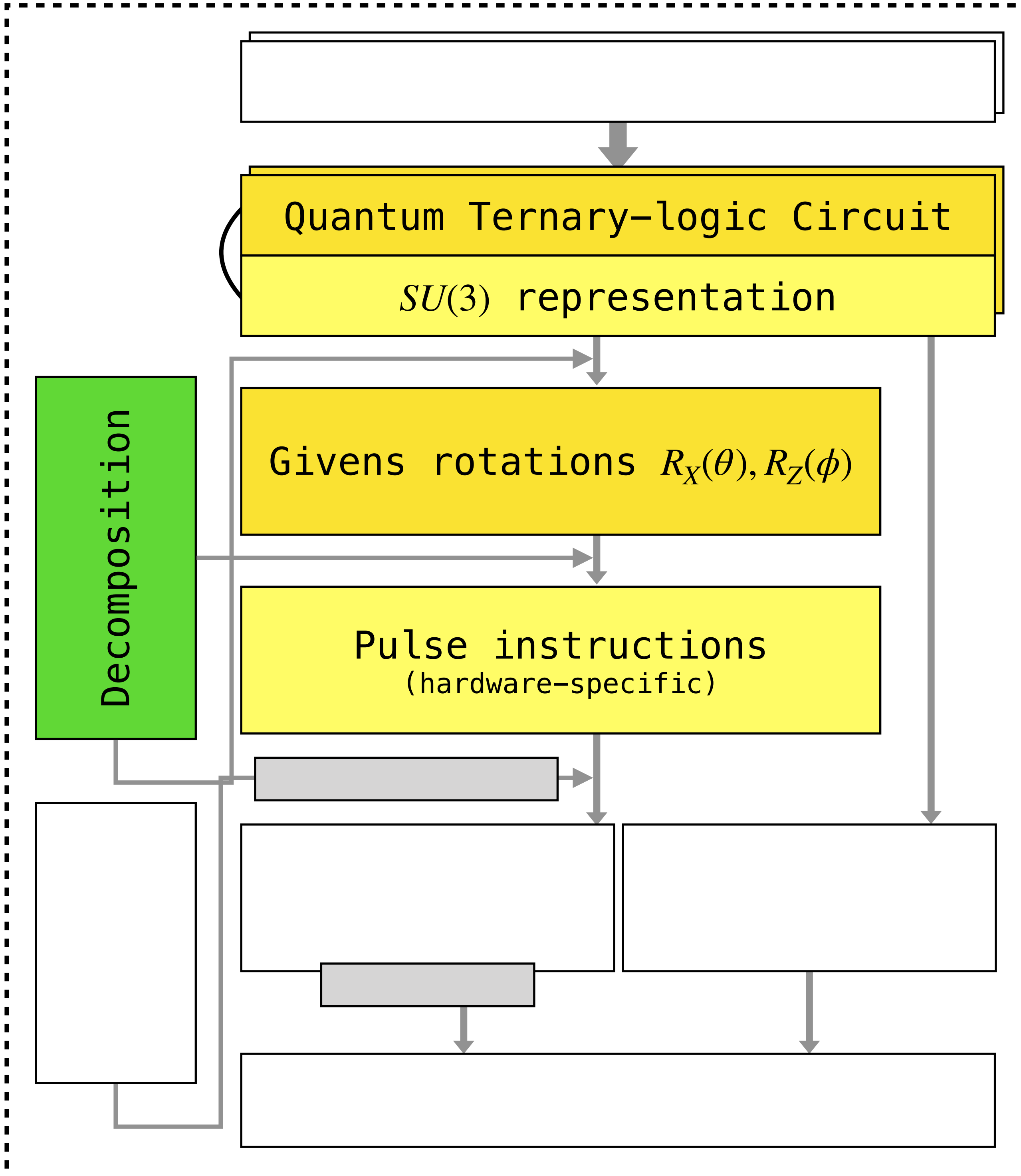


Part III

How we are realizing Qutritium

The need of decomposition

Generalized $SU(3)$ rotations to Givens rotations



$$\begin{pmatrix} u_{11} & u_{12} & u_{13} \\ u_{21} & u_{22} & u_{23} \\ u_{31} & u_{32} & u_{33} \end{pmatrix}$$

'Algorithm' language

$$U_d(\phi_6, \phi_5, \phi_4) R^{01}_{\phi_3}(\theta_3) R^{12}_{\phi_2}(\theta_2) R^{01}_{\phi_1}(\theta_1)$$

'Lie algebra of $SU(3)$' language

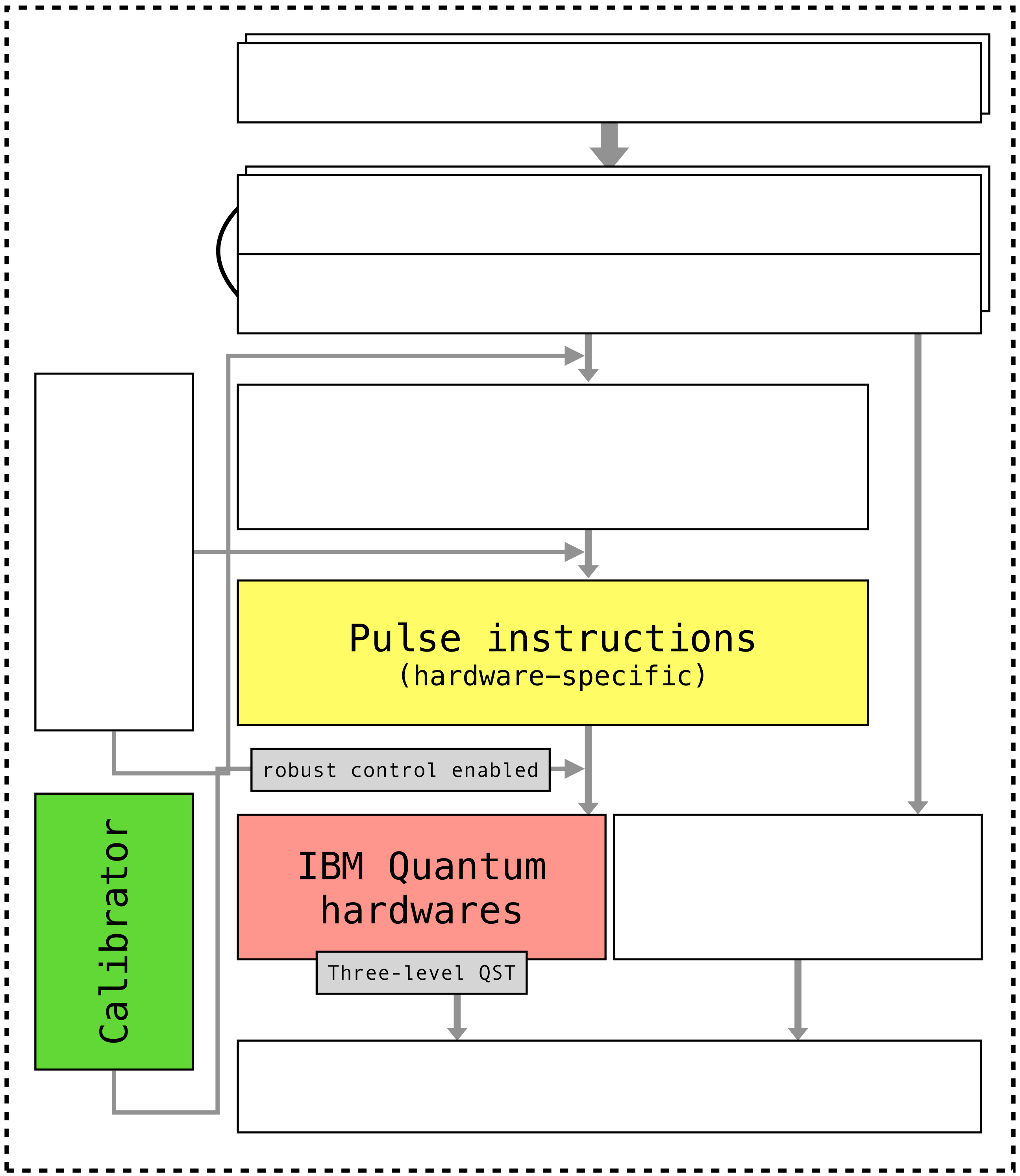
$$\Omega_D(t) = e^{2\pi i \delta_f t} \left(\Omega(t) + i(\alpha/\Delta)\dot{\Omega}(t) \right)$$

'Pulse-level' Qiskit language

+the complication of tracking phase advances between the two subspaces $\{ |n\rangle, |n+1\rangle \}$

The need of calibration

or day in the life of a quantum mechanic



Multiple sources of errors

1. Systematic drifts of the Hamiltonian parameters
2. Amplitude damping T_1/ Markovian pure dephasing T_2'
3. Electronics pulse distorsion
4. Coherent accumulation of over-under/rotation
5. Leakage out of comp. manifold
6. Off-resonant of R_ϕ^{12} pulses
7. Phase advance intrinsic to qudit rotations

Part IV

How we demonstrate Qutritium

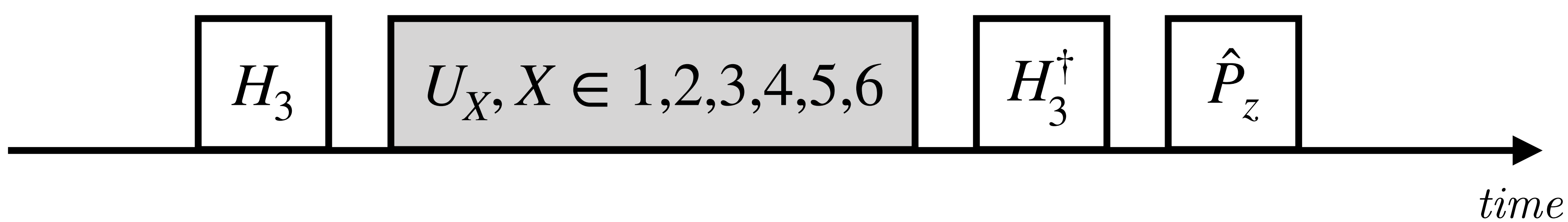
Toy model of single-qutrit computational speed up

Calling the oracle only once

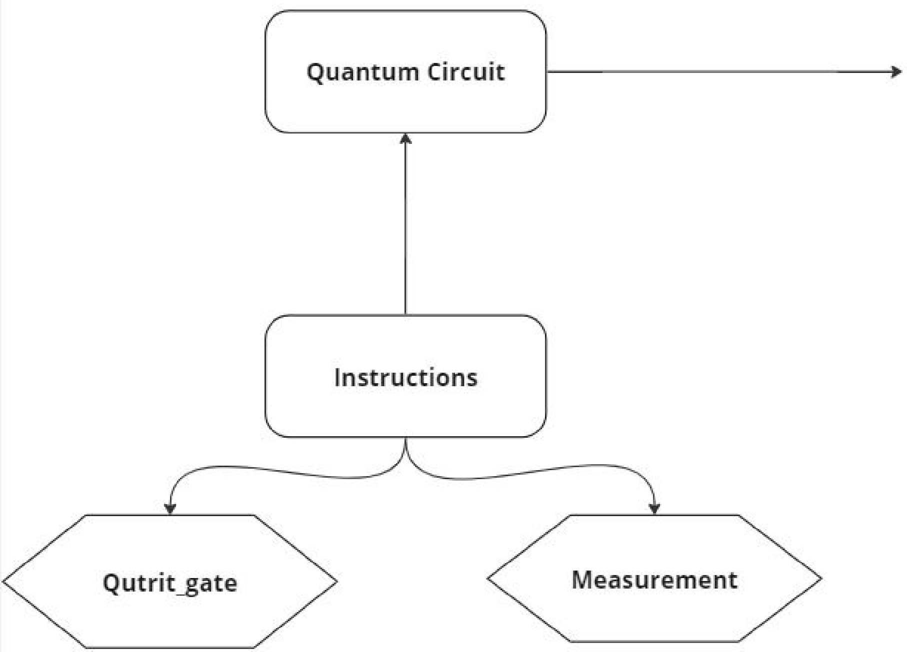
Determining the parity of a permutation $(-1)^{\sigma}$

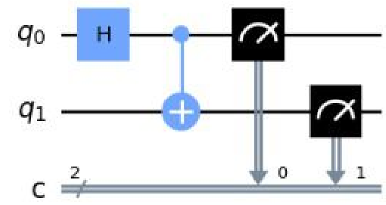
$$\sigma_1 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix} \quad \sigma_2 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix} \quad \sigma_3 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$$

$$\sigma_4 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} \quad \sigma_5 = \begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix} \quad \sigma_6 = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix}$$



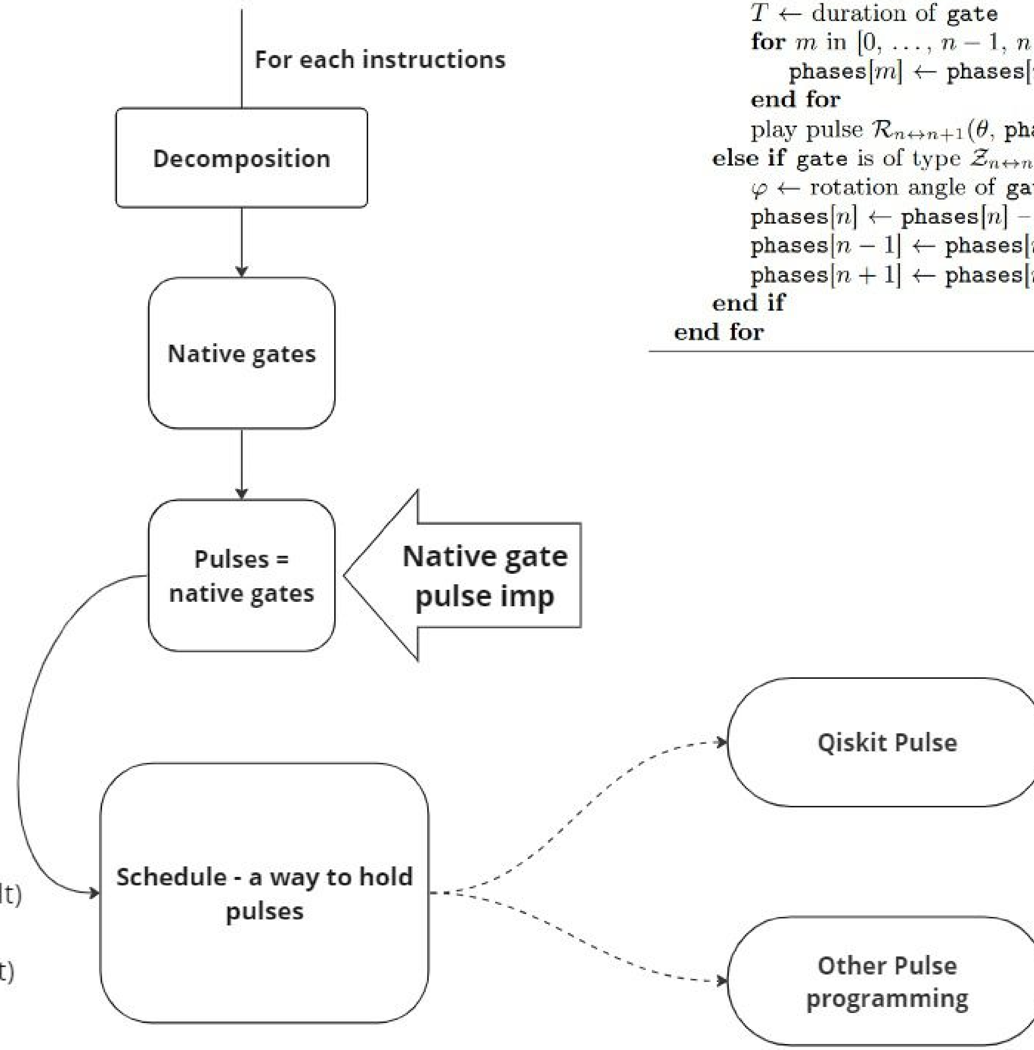
Two possibilities: If state $|0\rangle \rightarrow \sigma_X$ was even; if state $|2\rangle \rightarrow \sigma_X$ was odd!





Algorithm 1 Implementation of a sequence of Givens rotations $\mathcal{G}$ and phase gates $\mathcal{Z}$ via hardware-native pulses $\mathcal{R}$ achieved by keeping track of all necessary phase shifts.

levels $\leftarrow$ number of levels in qudit space
phases $\leftarrow [0, \ldots, 0]$ $\triangleright$ list of length levels-1
gates $\leftarrow$ sequence of $\mathcal{G}_{n\leftrightarrow n+1}(\theta, \phi)$ and $\mathcal{Z}_{n\leftrightarrow n+1}(\varphi)$ gates
for gate in gates **do**
  if gate is of type $\mathcal{G}_{n\leftrightarrow n+1}$ **then**
    $\theta \leftarrow$ rotation angle of gate
    $\phi \leftarrow$ phase of gate
    $T \leftarrow$ duration of gate
    for m in $[0, \ldots, n-1, n+2, \ldots,$ levels$]$ **do**
      phases$[m] \leftarrow$ phases$[m] - (\omega_m - \omega_n)T$
    end for
    play pulse $\mathcal{R}_{n\leftrightarrow n+1}(\theta,$ phases$[n] + \phi)$
  else if gate is of type $\mathcal{Z}_{n\leftrightarrow n+1}$ **then**
    $\varphi \leftarrow$ rotation angle of gate
    phases$[n] \leftarrow$ phases$[n] - \varphi$
    phases$[n-1] \leftarrow$ phases$[n-1] + \frac{\varphi}{2}$
    phases$[n+1] \leftarrow$ phases$[n+1] + \frac{\varphi}{2}$
  end if
end for



Rx_01(theta_1)Rx_12(theta_2)

- Freq , channel
- Gaussian_Pulse(X_amp = amp_ rabi_01) = Rx_01(pi)

--> Gaussian_Pulse(X_amp = amp_ rabi_01/pi*theta_1, duration=160dt)
--> Change Freq of Drive_Channel = freq_12
--> Gaussian_Pulse(X_amp = amp_ rabi_12/pi*theta_2, duration=160dt)