ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

MSc in QUANTUM COMPUTING AND QUANTUM TECHNOLOGIES

DEMOKRITOS
ΕΘΝΙΚΟ ΚΕΝΤΡΟ ΕΡΕΥΝΑΣ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ "ΔΗΜΟΚΡΙΤΟΣ"

Μείωση της διαρροής της πύλης *X* σε μεταβαλλόμενα με ροή Transmon qubits: DRAG, FAST-DRAG και HD-DRAG

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Βύρων Βασιλειάδης, Α.Ε.Μ.: 60625

Επιβλέπων Καθηγητής: Εμμανουήλ, Πασπαλάκης, Καθηγητής, Τμήμα Επιστήμης των Υλικών, Πανεπιστήμιο Πατρών

Ξάνθη, 2025

MSc in QUANTUM COMPUTING AND QUANTUM TECHNOLOGIES

DEMOKRITOS

ΕΘΝΙΚΟ ΚΕΝΤΡΟ
ΕΡΕΥΝΑΣ ΦΥΣΙΚΩΝ
ΕΠΙΣΤΗΜΩΝ
"ΔΗΜΟΚΡΙΤΟΣ"

ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ:

QUANTUM COMPUTING AND QUANTUM TECHNOLOGIES

ΜΕΤΑΠΤΥΧΙΑΚΗ ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Μείωση της διαρροής της πύλης X σε μεταβαλλόμενα με ροή Transmon qubits: DRAG, FAST-DRAG και HD-DRAG

Βύρων Βασιλειάδης, Α.Ε.Μ.: 60625

Η παρούσα Μεταπτυχιακή Διπλωματική Εργασία (Μ.Δ.Ε.) υποβλήθηκε στο Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Δημοκρίτειου Πανεπιστημίου Θράκης για την απόκτηση του τίτλου Μεταπτυχιακών Σπουδών Ειδίκευσης στις Επιστήμες της Κβαντικής Υπολογιστικής και των Κβαντικών Τεχνολογιών.

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ

Επιβλέπων Καθηγητής: Εμμανουήλ, Πασπαλάκης, Καθηγητής, Τμήμα Επιστήμης των Υλικών, Πανεπιστήμιο Πατρών

2ο Μέλος: Ιωάννης, Θανόπουλος, Καθηγητής, Τμήμα Επιστήμης των Υλικών, Πανεπιστήμιο Πατρων

3ο Μέλος Διονύσης, Στεφανάτος, Αναπληρωτής Καθηγητής, Τμήμα Επιστήμης των Υλικών, Πανεπιστήμιο Πατρων

Ξάνθη, 2025

DEMOCRITUS UNIVERSITY OF THRACE
SCHOOL OF ENGINEERING
DEPARTMENT ELECTRICAL AND COMPUTER ENGINEERING
SECTOR of ELECTRONICS AND INFORMATION SYSTEMS TECHNOLOGY

MSc in QUANTUM COMPUTING AND QUANTUM TECHNOLOGIES

NATIONAL CENTRE FOR SCIENTIFIC RESEARCH "DEMOKRITOS"

POSTGRADUATE PROGRAM:

QUANTUM COMPUTING AND QUANTUM TECHNOLOGIES

POSTGRADUATE DIPLOMA DISSERTATION

Reducing Leakage of *X*-Gate in Flux-Tunable Transmon Qubits: DRAG, FAST-DRAG and HD-DRAG

Vyron Vasileiadis, Registration Number: 60625

Postgraduate Diploma Dissertation submitted in partial fulfilment of the requirements for the Master's degree in Quantum Computing and Quantum Technologies, Department of Electrical and Computer Engineering, Democritus University of Thrace

COMMITTEE OF EXAMINERS

Supervisor: Emmanuel, Paspalakis, Professor, Department of Materials Science, University of Patras

Member 2: Ioannis, Thanopulos, Professor, Department of Materials Science, University of Patras

Member 3: Dionisis, Stefanatos, Associate Professor, Department of Materials Science, University of Patras

Xanthi, 2025

ΑΝΑΦΟΡΑ ΣΤΗΝ ΤΗΡΗΣΗ ΤΩΝ ΑΚΑΔΗΜΑΪΚΩΝ ΑΡΧΩΝ ΔΕΟΝΤΟΛΟΓΙΑΣ

Η παρούσα εργασία με τίτλο Μείωση της διαρροής της πύλης X σε μεταβαλλόμενα με ροή Transmon qubits: DRAG, FAST-DRAG και HD-DRAG είναι πρωτότυπη και πραγματοποιήθηκε από τον φοιτητή Βύρων Βασιλειάδη του Τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών με Αρ. Μητρώου ΑΕΜ 60625 υπό την επίβλεψη του Εμμανουήλ Πασπαλάκη. Η συγγραφή της εργασίας/διατριβής πραγματοποιήθηκε εξολοκλήρου από τον φοιτητή Βύρων Βασιλειάδη, υπό την καθοδήγηση και τις υποδείξεις του επιβλέποντά του.

Βεβαιώνεται ότι, κατά την εκπόνηση και τη συγγραφή της εργασίας/διατριβής του, ο/η φοιτητής τήρησε τα προβλεπόμενα από τον νόμο και τον αντίστοιχο εσωτερικό κανονισμό του Τμήματος, σεβάστηκε πλήρως τις Αρχές της Ακαδημαϊκής Ηθικής και του Κώδικα Δεοντολογίας, οι οποίες απαγορεύουν την παραποίηση των ερευνητικών/πειραματικών αποτελεσμάτων, την αναφορά ψευδών στοιχείων, την κατάχρηση της διανοητικής ιδιοκτησίας τρίτων και τη λογοκλοπή και ότι έγινε με σεβασμό στις αρχές.

REPORT ON COMPLIANCE WITH ACADEMIC ETHICS

This work, entitled *Reducing Leakage of X-Gate in Flux-Tunable Transmon Qubits: DRAG, FAST-DRAG and HD-DRAG* is original and was conducted by Vyron Vasileiadis, Department of Electrical and Computer Engineering, AEM Registration No. 60625, under the supervision of Emmanuel Paspalakis. The thesis was written solely by the student, under the guidance of the supervisor.

It is confirmed that the student complied with all legal provisions and Departmental regulations, adhered to the Principles of Academic Ethics and the Code of Conduct, and refrained from falsification of results, reporting of false data, misuse of intellectual property, and plagiarism.

ΠΕΡΙΛΗΨΗ

Η ασθενής ανυρμονικότητα στα transmon qubits καθιστά τις γρήγορες μονο-qubit περιστροφές ευάλωτες σε φασματική επικάλυψη με μεταβάσεις διαρροής, με αποτέλεσμα η αύξηση της ταχύτητας συχνά να επιδεινώνει τη μεταφορά πληθυσμού έξω από τον υπολογιστικό υπόχωρο. Η παρούσα διπλωματική εργασία αντιμετωπίζει αυτήν την ένταση στο πλαίσιο της κυκλωματικής κβαντικής ηλεκτροδυναμικής (cQED) συγκρίνοντας συστηματικά και επεκτείνοντας οικογένειες αναλυτικών παλμομορφών για συντονισμένο έλεγχο. Ξεκινώντας από τους καθιερωμένους Gaussian και Derivative Removal by Adiabatic Gate (DRAG) παλμούς, αναπτύσσουμε δύο αναλυτικά διαχειρίσιμες επεκτάσεις — τους Fourier–ansatz spectrum–tuning DRAG (FAST–DRAG) και higher–derivative DRAG (HD–DRAG) — που καταστέλλουν περαιτέρω τη διαρροή χωρίς αύξηση της διάρκειας της πύλης. Αυτές οι οικογένειες παλμών ενσωματώνονται σε μια πλήρως αυτοματοποιημένη cQED ροή εργασίας που μεταγλωττίζει προγράμματα σε επίπεδο παλμών, τα εκτελεί σε ένα flux-tunable transmon, και αξιολογεί τις πιστότητες και τη διαρροή υπό ίδιες συνθήκες για όλες τις στρατηγικές διαμόρφωσης. Οι σχεδιασμοί αντιστοιχούν καθαρά σε πειραματικές παραμέτρους, οι οποίες βαθμονομούνται μέσω μονοδιάστατης και δισδιάστατης φασματοσκοπίας, ταλαντώσεων Rabi, Single-Shot Readout (SSRO), και randomized benchmarking (RB), όλα υλοποιημένα σε μια ενοποιημένη στοίβα λογισμικού που διαχωρίζει τη γλώσσα προγραμματισμού παλμών από τα instrument backends και τυποποιεί την απόκτηση δεδομένων. Η προκύπτουσα υπολογιστική ροή επιτρέπει μια άμεση, ισότιμη σύγκριση των Gaussian, DRAG, FAST–DRAG και HD–DRAG για σταθερό χρόνο πύλης. Πειραματικά, τόσο το FAST–DRAG όσο και το HD–DRAG μειώνουν τη διαρροή και το συνολικό σφάλμα σε σχέση με το συμβατικό DRAG σε παρόμοιο εύρος ζώνης και προσεγγίζουν το όριο συνοχής για την επιλεγμένη διάρκεια πύλης, ενώ παραμένουν ανθεκτικά σε μέτριες παραλλαγές παραμέτρων.

Λέξεις-κλειδιά: Κβαντική Υπολογιστική, Κβαντικός Έλεγχος, Υπεραγώγιμα Qubits, Transmon Qubits, cQED, Προγραμματισμός Παλμών, Πειραματική Φυσική

ABSTRACT

Weak anharmowenicity in transmon qubits makes fast single–qubit rotations vulnerable to spectral overlap with leakage transitions, so that improving speed often aggravates population transfer outside the computational subspace. This thesis addresses that tension within the circuit quantum electrodynamics (cQED) framework by systematically comparing and extending analytical pulse–shaping families for resonant control. Starting from the standard Gaussian and Derivative Removal by Adiabatic Gate (DRAG) pulses, we develop two analytically tractable extensions—Fourier–ansatz spectrum–tuning DRAG (FAST–DRAG) and higher–derivative DRAG (HD–DRAG)—that further suppress leakage without increasing gate duration. These pulse families are integrated into a fully scripted cQED workflow that compiles pulse–level programs, executes them on a flux–tunable transmon, and benchmarks fidelities

and leakage under identical conditions across shaping strategies. The designs map cleanly onto experimental knobs, which are calibrated through 1-D and 2-D spectroscopy, Rabi oscillations, Single-Shot Readout (SSRO), and randomized benchmarking (RB), all implemented in a unified software stack that separates a pulse–programming language from instrument backends and standardizes data acquisition. The resulting pipeline enables a head–to–head, apples–to–apples comparison of Gaussian, DRAG, FAST–DRAG, and HD–DRAG at fixed gate time. Experimentally, both FAST–DRAG and HD–DRAG reduce leakage and total error relative to conventional DRAG at comparable bandwidth and approach the coherence floor at the chosen gate duration, while remaining robust under moderate parameter variations.

Keywords: Quantum Computing, Quantum Control, Superconducting Qubits, Transmon Qubits, cQED, Pulse Programming, Experimental Physics

Abstract

Weak anharmowenicity in transmon qubits makes fast single–qubit rotations vulnerable to spectral overlap with leakage transitions, so that improving speed often aggravates population transfer outside the computational subspace. This thesis addresses that tension within the circuit quantum electrodynamics (cQED) framework by systematically comparing and extending analytical pulse–shaping families for resonant control. Starting from the standard Gaussian and Derivative Removal by Adiabatic Gate (DRAG) pulses, we develop two analytically tractable extensions—Fourier–ansatz spectrum–tuning DRAG (FAST–DRAG) and higher–derivative DRAG (HD–DRAG)—that further suppress leakage without increasing gate duration. These pulse families are integrated into a fully scripted cQED workflow that compiles pulse–level programs, executes them on a flux–tunable transmon, and benchmarks fidelities and leakage under identical conditions across shaping strategies. The designs map cleanly onto experimental knobs, which are calibrated through 1-D and 2-D spectroscopy, Rabi oscillations, Single-Shot Readout (SSRO), and randomized benchmarking (RB), all implemented in a unified software stack that separates a pulse–programming language from instrument backends and standardizes data acquisition. The resulting pipeline enables a head–to–head, apples–to–apples comparison of Gaussian, DRAG, FAST–DRAG, and HD–DRAG at fixed gate time. Experimentally, both FAST–DRAG and HD–DRAG reduce leakage and total error relative to conventional DRAG at comparable bandwidth and approach the coherence floor at the chosen gate duration, while remaining robust under moderate parameter variations.

All stable processes we shall predict.
All unstable processes we shall control.

— John von Neumann

Contents

Contents 2

List of Figures 5

Acknowledgments 6

1 Introduction 7

2 Flux-Tunable Transmon Qubits 10
- 2.1 LC Circuit ... 10
- 2.2 Transmon Hamiltonian ... 12
- 2.3 Driving the Qubit ... 14
- 2.4 Coupling Schemes ... 14
  - 2.4.1 Oscillator–Oscillator Coupling ... 14
  - 2.4.2 Oscillator–Qubit Coupling ... 15
  - 2.4.3 Qubit–Qubit Coupling ... 15
- 2.5 Frequency Control via SQUID Tuning ... 15
- 2.6 Typical Transmon Parameters ... 16

3 Theoretical Foundations of *X*-Gate Calibration 17
- 3.1 Overview ... 17
- 3.2 Spectroscopy and Qubit Frequency Identification ... 18
  - 3.2.1 From device parameters to a frequency prior ... 18
  - 3.2.2 Dispersive readout and two-tone spectroscopy: signal model ... 18
  - 3.2.3 Optical Bloch steady state, Lorentzian lineshape, and power broadening ... 19
  - 3.2.4 Higher levels, AC–Stark shift, and multi-photon features ... 19
  - 3.2.5 Practical sweep design and frequency planning ... 19
  - 3.2.6 Analytical fit models and parameter extraction ... 20
  - 3.2.7 Connecting spectroscopy to coherence and Purcell limits ... 20
- 3.3 Rabi and Ramsey for Pulse Amplitude, Duration, and Frequency ... 21
  - 3.3.1 Driven Two-Level Dynamics with Square and Shaped Pulses ... 21
  - 3.3.2 Small-Detuning Effects: Toggling-Frame Analysis and Error Axes ... 22
  - 3.3.3 Rabi Oscillations with Decoherence: Bloch-Equation Solution ... 22
  - 3.3.4 Ramsey Interferometry: Phase, Contrast, and Noise ... 23
  - 3.3.5 Spectral Selectivity and Leakage Metrics for Envelopes ... 23
  - 3.3.6 Calibration Formulas and Small-Error Fidelity Budgets ... 24

3.3.7 Decoherence-Limited Infidelity (Integral Estimates) 24
3.4 Pulse Optimization: DRAG to Suppress Leakage and AC–Stark Errors 25
3.4.1 Three-Level Transmon and Leakage Mechanism 25
3.4.2 DRAG Ansatz and Analytic Rationale 25
3.4.3 FAST DRAG: Fourier–Ansatz Spectrum Tuning 27
3.4.4 Higher–Derivative DRAG 28
3.4.5 Experimental Tuning Strategies 30
3.5 T_1 and T_2 Measurements 31
3.5.1 T_1 (Energy Relaxation) 31
3.5.2 T^* (Ramsey) and T_2 (Echo) 32
3.5.3 Relation Between T_1, T_ϕ, and T_2 34
3.6 Randomized Benchmarking (RB) and Gate-Fidelity Verification 34
3.6.1 Standard RB (SRB) 34
3.6.2 Interleaved RB (IRB) 35
3.6.3 SPAM robustness, model diagnostics, and fitting 36
3.6.4 Decoherence limit and relation to T_1, T_2 36
3.6.5 Leakage-robust RB and extraction of leakage/seepage 36
3.6.6 Unitarity benchmarking and coherence of noise 36
3.6.7 Simultaneous and two-qubit RB 37
3.6.8 Experimental procedure 37
3.6.9 Gate dependence, drift, and model limitations 37
3.6.10 Connecting RB to per–primitive errors 37
3.7 Putting It All Together: A Practical Calibration Loop 38
3.7.1 Recommended order of operations 38
3.7.2 Gate error budget 39

4 Experimental Setup **41**
4.1 Overview 41
4.2 Quantum processor: QuantWare *Soprano* 41
4.3 Cryogenic platform 42
4.4 Room-temperature instrumentation and signal flow 43
4.4.1 Qubit drive path: QCM-RF 44
4.4.2 Readout path: QRM-RF 44
4.4.3 Flux-control path: QCM (baseband) 44
4.4.4 Timing, references, and triggers 44
4.4.5 LO and IF planning in the Qblox signal chain 44
4.4.6 Digital–analog–digital flow (end-to-end) 45
4.5 Calibrated signal levels and line models 46
4.6 Pulse families and on-hardware processing 46

5 Experiment and Results **47**
5.1 Overview 47
5.2 Analytical synthesis of control pulses 48
5.2.1 Pulse families 48
5.2.2 Spectral analysis and figures 49
5.2.3 Implementation details 50
5.3 Qililab: Quantum Control Library 51
5.4 Resonator Spectroscopy (One-time) 52

5.5 Resonator Spectroscopy vs Qubit Flux Offsets (One-time) 54
5.6 Two-tone Spectroscopy (f_{01}) (One-time) 58
5.7 Two-Tone Spectroscopy vs Flux Offsets (One-time) 60
5.8 Rabi oscillations (Per-variant) 64
5.9 Single-Shot Readout (SSRO) (Per-variant) 69
5.10 Randomized benchmarking (RB) (Per-variant) 76

6 Conclusions **79**

A Gate synthesis and spectral analysis **81**

B Lorentzian fit **91**

Bibliography **95**

List of Figures

2.1 **Circuit schematics and energy potentials for LC oscillator and transmon.** 12

4.1 **Schematic of the experimental setup.** 42
4.2 **QuantWare *Soprano* chip under the microscope.** 43

5.1 **Time-domain envelopes for four analytical variants.** 50
5.2 **Amplitude spectrum of the complex envelope.** 51
5.3 **Resonator spectroscopy** 54
5.4 **Resonator spectroscopy versus static qubit-flux bias.** 58
5.5 **Two-tone qubit spectroscopy** 60
5.6 **Two-tone qubit spectroscopy versus DC flux offset.** 63
5.7 **Rabi oscillations for each pulse variant.** 69
5.8 **Single-shot readout characterization for each pulse variant.** 75
5.9 **Randomized benchmarking for each pulse variant.** 78

Acknowledgments

I am deeply grateful to my supervisor for steady guidance, rigorous scientific feedback, and unwavering support throughout the course of this research. Their insight and high standards have shaped the scope of the project and strengthened the clarity and reproducibility of the results.

I wish to thank the professors and lecturers of the Master's programme for exemplary teaching and mentorship. The breadth and depth of the coursework, seminars, and discussions provided the theoretical and methodological foundation upon which this thesis is built.

I also extend my sincere appreciation to my colleagues at Qilimanjaro Quantum Tech for their generosity with time and expertise. Their constructive discussions, practical advice on experiments and software, and collegial spirit created a stimulating environment that greatly advanced this work.

I am especially indebted to my family and friends for their patience, encouragement, and emotional support. Their faith in me, understanding during long workdays, and steadfast presence kept me grounded and motivated through challenges.

Any remaining errors are my own.

Chapter 1

Introduction

Quantum computation seeks to manipulate well–isolated quantum two–level systems with error rates low enough to enable fault tolerance [1, 2, 3, 4, 5, 6, 7, 8]. In superconducting platforms based on weakly anharmonic Josephson circuits (transmons), the dominant challenge for single–qubit control is the tension between *speed* and *selectivity* [9, 10, 11, 12, 13]. Short pulses reduce exposure to amplitude damping and dephasing [14, 15], yet their broad spectra can resonantly or near–resonantly populate noncomputational levels, producing *leakage* [16, 9] that is not captured by simple Pauli error models [17, 18] and that compounds across gate sequences [19]. This thesis addresses this trade–off by implementing, and experimentally validating analytic pulse–design frameworks [16, 11] that suppress leakage while preserving fast gate times, together with a measurement–informed calibration pipeline that connects the microwave control layer to device–level Hamiltonian parameters and noise processes.

We consider a tunable–frequency transmon operated in the dispersive regime of circuit quantum electrodynamics (cQED) and parked at, or close to, a flux–insensitive sweet spot during single–qubit gates [20, 21]. To capture leakage, we truncate to the lowest three levels $\{|0\rangle, |1\rangle, |2\rangle\}$ and write, in a frame rotating at the $|0\rangle \leftrightarrow |1\rangle$ drive frequency,

$$\frac{H(t)}{\hbar} = \frac{\alpha}{2}\,|2\rangle\langle 2| + \frac{1}{2}\Big[\Omega_{\mathrm{I}}(t)\,X + \Omega_{\mathrm{Q}}(t)\,Y\Big], \qquad \alpha = \omega_{12} - \omega_{01} < 0, \tag{1.1}$$

where $X = \sum_{j} \lambda_{j,j+1}\big(|j\rangle\langle j+1| + |j+1\rangle\langle j|\big)$ and $Y = -i\sum_{j} \lambda_{j,j+1}\big(|j\rangle\langle j+1| - |j+1\rangle\langle j|\big)$ encode the ladder structure, following standard multilevel modeling approaches [22, 23]. The control envelopes (Ω_{I}, Ω_{Q}) are synthesized by a Radio-Frequency Arbitrary Waveform Generator (RF AWG) with deterministic timing and are subject to mild, well–characterized filtering along the control chain [24]. Our target is a resonant X_π gate that in the closed–system limit maps $|0\rangle' \to |1\rangle$ and $|1\rangle' \to |0\rangle$, with total duration t_g short relative to T_1 and T_ϕ and with leakage and coherent phase errors suppressed below the decoherence floor [25].

We study three analytic families for the realization of the resonant X_π gate. The first is *Derivative Removal by Adiabatic Gate* (DRAG) [16], in which a quadrature proportional to the time derivative of a smooth in–phase envelope cancels, to leading order in $\|\Omega\|/|\alpha|$ [26, 27, 15], the diabatic coupling into $|2\rangle$:

$$\Omega_{\mathrm{DRAG}}(t) = \Omega_{\mathrm{I}}(t) - i\beta\,\dot{\Omega}_{\mathrm{I}}(t), \qquad \beta \simeq \frac{1}{\alpha} \quad (\alpha \text{ in rad/s}), \tag{1.2}$$

with a small additional detuning or phase pre–distortion compensating the concomitant AC–Stark shift [26]. In practice, β is tuned around the analytic value to account for line filtering and model mismatch [28, 9].

Building on the DRAG mechanism, we introduce and evaluate two analytical extensions tailored to fast gates on weakly anharmonic transmons. The first, *Fourier–Ansatz Spectrum Tuning DRAG* (FAST–DRAG) [11], designs the in–phase envelope as a short cosine series and minimizes its spectral energy over user–specified baseband intervals (in particular a neighborhood of $|\alpha|/2\pi$ and a high–frequency stopband) before applying the DRAG quadrature, which places a spectral zero at the leakage line. The second, *Higher–Derivative DRAG* (HD–DRAG) [11], augments (1.2) by linear combinations of higher time derivatives of the in–phase template so that the *complex* envelope exhibits a root of prescribed multiplicity at $|\alpha|/2\pi$, yielding multi–order cancellation of spectral weight at the leakage transition while preserving the rotation area [29].

A central feature of the present work is the tight integration between model–based pulse design and measurement–informed calibration. We implement a complete bring–up and calibration loop that starts from two–tone spectroscopy and resonator characterization, proceeds through T_1/T_2 measurements, Rabi and Ramsey calibrations, and DRAG sweeps, and culminates in leakage diagnostics (shelving and flipping) and single–qubit randomized benchmarking (RB). Each experiment is accompanied by an explicit signal model that connects measured observables (demodulated *I/Q* samples, complex transmission S_{21}, population probabilities, or sequence survival rates) to Hamiltonian parameters and noise spectra. By comparing the observed gate errors to the coherence floor $r_{\mathrm{dec}} \approx \frac{t_g}{2T_1} + c_\phi\, t_g/T_\phi$, we disentangle decoherence–dominated regimes from control– or leakage–limited regimes and iteratively refine pulse parameters [30, 31, 17, 32, 33].

All measurements are orchestrated with Qilimanjaro Quantum Tech's qililab library, for which the author is one of the lead maintainers. The library provides three capabilities essential for this thesis. First, a platform abstraction builds the instrument graph from a configuration file (runcard) and exposes a single entry point to execute experiments deterministically on hardware. Second, a pulse–programming domain–specific language (DSL) separates algorithmic intent from instrument specifics: a *QProgram* describes time–ordered operations in a single shot (frequency updates, pulse plays, measurements, waits, inner loops and averages), while an *Experiment* composes one or more QPrograms and sweeps hardware parameters such as attenuations or DC biases in outer loops. Third, a results layer standardizes data acquisition: executing an Experiment returns a path to a Hierarchical Data Format version 5 (HDF5) file containing raw data and metadata, and the helper class ExperimentResults exposes multidimensional arrays whose trailing dimension encodes the *I/Q* outcomes. For analysis and plotting we form the complex transmission $S_{21} = I + iQ$ and its modulus $|S_{21}| = \sqrt{I^2 + Q^2}$; we refer to $|S_{21}|$ as the *modulus* (amplitude) [34, 35] and use $|S_{21}|_{\mathrm{dB}} = 20\log_{10}|S_{21}|$ when working in decibels, consistent with standard microwave practice [36, 37]. This software stack enables fully scripted, reproducible calibration and measurement campaigns that can be re–run end–to–end on different devices and hardware revisions.

The scientific contributions of this thesis are both analytical and experimental. On the analytical side, we derive closed–form FAST–DRAG envelopes by solving a constrained quadratic program in the frequency domain to minimize spectral power over tunable intervals while preserving the target rotation area; we also present HD–DRAG as a binomial higher–derivative construction that imprints an R–fold spectral zero at $|\alpha|/2\pi$ with smooth endpoints, quantifying leakage–probability scaling $\propto (\|\Omega\|/|\alpha|)^{2R+2}$ under mild regularity, analogous to the analyses in [11]. On the experimental side, we (i) commission a full cQED stack around a tunable–frequency transmon, (ii) implement a measurement–informed calibration pipeline that maps each knob (amplitude, duration, β, phase pre–distortion) to a physical target (area, spectral selectivity, Stark compensation), and (iii) carry out a head–to–head comparison of conventional DRAG, FAST–DRAG, and HD–DRAG at fixed gate time across the same device and setup. The combination of analytic solvability, small–parameter calibration, and reproducible scripting yields gates whose observed errors approach the coherence floor at the chosen t_g, with demonstrably lower leakage tails than standard DRAG at comparable bandwidth [9].

This work is organized as follows: Chapter 2 introduces the weakly anharmonic transmon model, control

Hamiltonians, and decoherence channels, establishing the filter–function formalism used to interpret Ramsey/echo. Chapter 3 presents DRAG, FAST–DRAG, and HD–DRAG in a self–contained manner, deriving the leading–order cancellations and AC–Stark compensation and clarifying the regime of validity for each method. Chapter 4 documents the cryogenic and room–temperature platform, the readout and control chains, and the software stack. Chapter 5 reports the calibration and characterization campaign together with the Python code that implements each experiment. We conclude with limitations of this study and with an outlook on improving and extending these techniques. These chapters are further supplemented by two appendices, providing the source code for the analytical pulse generation in Appendix A and the common fitting procedures in Appendix B.

Throughout, we set $\hbar = 1$ unless stated otherwise, and express all frequencies in angular units unless "$/2\pi$" is explicitly written. The transmon anharmonicity is denoted by $\alpha < 0$. For coherence times, we adopt the convention $T_2^{-1} = (2T_1)^{-1} + T_\phi^{-1}$.

Chapter 2

Flux-Tunable Transmon Qubits

In this chapter we introduce the basic building block for our experiments, the transmon qubit. We begin with a review of quantum mechanics as it applies to electrical oscillators and then apply similar concepts to derive the energy-level structure of a transmon. Next, we discuss methods for coupling quantum circuits to control and measure them as well as coupling between different quantum devices. Finally, we review various design constraints such as frequency tuning and coupling strengths and list typical parameters used in our experiments. Primary references include the thesis of Daniel Sank [38], lecture notes by Girvin [39] and Devoret [40], and the original paper by Koch *et al.* [20].

2.1 LC Circuit

To understand how quantum mechanics applies to circuits, we start with a simple LC oscillator [see Fig. 2.1 (a)]. In this circuit the energy is stored both in the capacitor and the inductor. The *potential energy* stored in a capacitor with capacitance C and charge Q is

$$U = \frac{Q^2}{2C}, \tag{2.1}$$

and the *kinetic energy* (associated with the current $\dot{Q}$ flowing through an inductor of inductance L) is

$$T = \frac{L\dot{Q}^2}{2}. \tag{2.2}$$

The Lagrangian is then

$$\mathrm{L} = T - U = \frac{L\dot{Q}^2}{2} - \frac{Q^2}{2C}. \tag{2.3}$$

Applying the Euler–Lagrange equation

$$\frac{\partial \mathrm{L}}{\partial Q} - \frac{d}{dt}\frac{\partial \mathrm{L}}{\partial \dot{Q}} = 0, \tag{2.4}$$

we obtain

$$\frac{Q}{C} + L\ddot{Q} = 0, \tag{2.5}$$

or equivalently,

$$\ddot{Q} + \omega^2 Q = 0, \tag{2.6}$$

where the resonance (angular) frequency is

$$\omega = \frac{1}{\sqrt{LC}}. \tag{2.7}$$

The conjugate momentum corresponding to the generalized coordinate Q is

$$\Phi \equiv \frac{\partial \mathrm{L}}{\partial \dot{Q}} = L\dot{Q}, \tag{2.8}$$

which has the physical interpretation of the magnetic flux threading the inductor.

The Hamiltonian, obtained by the Legendre transform,

$$H = \Phi\dot{Q} - \mathrm{L}, \tag{2.9}$$

thus becomes

$$H = \frac{\Phi^2}{2L} + \frac{Q^2}{2C}, \tag{2.10}$$

which is the familiar harmonic oscillator Hamiltonian.

For quantization, we promote Q and Φ to operators obeying the canonical commutation relation

$$[\Phi, Q] = i\hbar. \tag{2.11}$$

It is convenient to define the ladder operators

$$a = \sqrt{\frac{1}{2\hbar Z}}\,\Phi + iZQ\,, \tag{2.12}$$

$$a^\dagger = \sqrt{\frac{1}{2\hbar Z}}\,\Phi - iZQ\,, \tag{2.13}$$

where the characteristic impedance is

$$Z = \sqrt{\frac{L}{C}}. \tag{2.14}$$

In terms of these operators, the Hamiltonian [Eq. (2.10)] reads

$$H = \hbar\omega \quad a^\dagger a + \frac{1}{2} \quad . \tag{2.15}$$

Moreover, the charge and flux operators can be expressed as

$$Q = i\,Q_{\mathrm{zpf}}\,(a^\dagger - a), \tag{2.16}$$

$$\Phi = \Phi_{\mathrm{zpf}}\,(a^\dagger + a), \tag{2.17}$$

with the zero-point fluctuations defined by

$$Q_{\mathrm{zpf}} = \sqrt{\frac{\hbar}{2Z}}, \tag{2.18}$$

$$\Phi_{\mathrm{zpf}} = \sqrt{\frac{\hbar Z}{2}}. \tag{2.19}$$

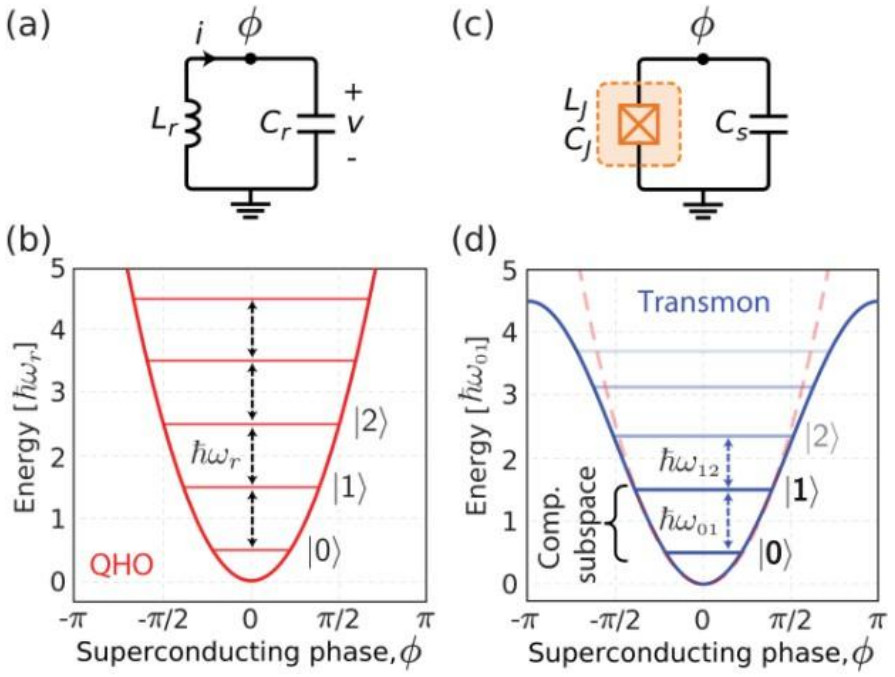


Figure 2.1: Circuit schematics and energy potentials for LC oscillator and transmon. Reproduced from [41]. (a) Lumped-element LC resonator used as the canonical linear mode in circuit QED. The node variable is the superconducting phase ϕ (reduced flux $\Phi = (\Phi_0/2\pi)\,\phi$, with Φ_0 the flux quantum), the current through the inductor is i, and the voltage across the capacitor is v. (b) Corresponding quadratic potential energy $U_r(\phi) = \frac{1}{2}E_L\,\phi^2$ with $E_L \equiv (\Phi_0/2\pi)^2/L_r$. The spectrum is that of a quantum harmonic oscillator, $E_n = \hbar\omega_r\,(n + \frac{1}{2})$, with equally spaced levels and allowed transitions of energy $\hbar\omega_r$. (c) Transmon circuit: a Josephson junction (JJ, orange) with Josephson energy E_J and critical current I_c, shunted by a large capacitance C_s (often in parallel with the junction capacitance C_J, giving $C_\Sigma = C_s + C_J$). The effective two-terminal element replaces the linear inductor by the nonlinear JJ whose current–phase relation is $I = I_c \sin\phi$ and whose energy is $U_J(\phi) = -E_J \cos\phi$ with $E_J = \Phi_0 I_c/2\pi$. (d) Transmon potential $U_t(\phi) = -E_J \cos\phi$ (blue) with weakly anharmonic level spacings: the qubit transition $\hbar\omega_{01}$ and the next transition $\hbar\omega_{12}$ differ by the negative anharmonicity $\alpha = \omega_{12} - \omega_{01} < 0$. The lowest two eigenstates $\{|0\rangle, |1\rangle\}$ form the computational subspace; the slight deviation from equal spacing is what enables selective microwave control while retaining insensitivity to charge noise at large E_J/E_C.

2.2 Transmon Hamiltonian

A linear LC oscillator, having equally spaced energy levels, is not suitable for quantum computation since selective addressing of only two levels becomes impossible [2, 39]. To obtain a nonlinear energy level structure, one replaces the linear inductor with a nonlinear element: the Josephson junction (JJ) [40], which is the workhorse nonlinearity in circuit QED [42]. A JJ is formed by two superconducting leads separated by a

thin insulating barrier and is characterized by the Josephson relations [43, 44, 45]:

$$V(t) = \frac{\hbar}{2e}\frac{d\phi}{dt}, \tag{2.20}$$

$$I(t) = I_c \sin\phi(t), \tag{2.21}$$

where $\phi(t)$ is the phase difference across the junction and I_c is its critical current. The energy stored in the Josephson junction is obtained by integrating the power:

$$U = \int_0^t V(t')I(t')\,dt' = -E_J \cos\phi, \tag{2.22}$$

with the Josephson energy defined as

$$E_J = \frac{\Phi_0 I_c}{2\pi}, \tag{2.23}$$

and the flux quantum

$$\Phi_0 = \frac{h}{2e}. \tag{2.24}$$

Replacing the linear inductor in the LC circuit with the nonlinear Josephson element (see Fig. 2.1 (c)), the transmon Hamiltonian becomes

$$H = \frac{Q^2}{2C} - E_J \cos\left(\frac{2\pi\Phi}{\Phi_0}\right). \tag{2.25}$$

It is common to express this Hamiltonian in terms of the number operator n, defined by

$$n = \frac{Q}{2e}, \tag{2.26}$$

and the charging energy

$$E_C = \frac{e^2}{2C}. \tag{2.27}$$

Thus, Eq. (2.25) can be rewritten as

$$H = 4E_C n^2 - E_J \cos\phi, \tag{2.28}$$

where the operators n and ϕ obey the commutation relation

$$[\phi, n] = i. \tag{2.29}$$

To gain insight into the energy level structure, we expand the cosine term assuming small fluctuations,

$$\cos\phi \approx 1 - \frac{\phi^2}{2} + \frac{\phi^4}{24} + \ldots, \tag{2.30}$$

and dropping the constant term, we obtain (to lowest order)

$$H \approx 4E_C n^2 + \frac{E_J}{2}\phi^2 - \frac{E_J}{24}\phi^4. \tag{2.31}$$

Retaining only the quadratic term yields the harmonic oscillator with energy spacing

$$\hbar\omega_{10} \approx \sqrt{8E_J E_C}, \tag{2.32}$$

while the quartic term introduces an anharmonicity of approximately

$$\alpha \equiv \omega_{21} - \omega_{10} \approx -E_C. \tag{2.33}$$

Thus, the transmon effectively uses the two lowest states of an anharmonic oscillator as its qubit states.

2.3 Driving the Qubit

Control of the transmon is achieved by applying microwave drives [46, 47, 8]. If a LC circuit is coupled to a voltage source through a drive capacitance C_d, then with the addition of a time-dependent voltage $V_d(t)$, the total Lagrangian becomes

$$L_{\text{driven}} = \frac{1}{2} C_\Sigma \dot{\Phi}^2 + \frac{1}{2} C_d \left(V_d(t) - \dot{\Phi} \right)^2 - \frac{\Phi^2}{2L'} \tag{2.34}$$

where the total capacitance is $C_\Sigma = C + C_d$. Neglecting constant terms and redefining the conjugate momentum accordingly, one finds that the drive couples predominantly to the momentum (or equivalently the charge) degree of freedom. Treating the drive as a perturbation and truncating to the two-level system, we obtain the driven qubit Hamiltonian

$$H_{\text{qubit}} = -\frac{\hbar \omega_{10}}{2} \sigma_z + \hbar \Omega f(t) \sigma_y, \tag{2.35}$$

where Ω is the Rabi frequency given approximately by

$$\Omega = \frac{C_d V_0 Q_{\text{zpf}}}{\hbar \sqrt{C + C_d}}, \tag{2.36}$$

and $f(t)$ denotes the pulse envelope. An appropriate choice of phase in the drive defines the axis of rotation on the Bloch sphere [40, 39].

2.4 Coupling Schemes

Coupling between elements in superconducting circuits is crucial for both readout and multi-qubit operations. Here we briefly describe three types of coupling.

2.4.1 Oscillator–Oscillator Coupling

Consider two LC oscillators with capacitances C_1 and C_2 respectively, coupled by a capacitor C_g. The kinetic energy part of the Lagrangian is

$$T = \frac{1}{2}(C_1 + C_g)\dot{\Phi}_1^2 + \frac{1}{2}(C_2 + C_g)\dot{\Phi}_2^2 - C_g \dot{\Phi}_1 \dot{\Phi}_2. \tag{2.37}$$

This may be written in matrix form as

$$T = \frac{1}{2} \begin{pmatrix} \dot{\Phi}_1 & \dot{\Phi}_2 \end{pmatrix} \begin{pmatrix} C_1 + C_g & -C_g \\ -C_g & C_2 + C_g \end{pmatrix} \begin{pmatrix} \dot{\Phi}_1 \\ \dot{\Phi}_2 \end{pmatrix}. \tag{2.38}$$

Upon Legendre transformation and subsequent quantization, the coupling Hamiltonian between the oscillators is found to be

$$H_g = \hbar g \left(a_1^\dagger - a_1 \right) \left(a_2 - a_2^\dagger \right), \tag{2.39}$$

with the coupling strength

$$g = \sqrt{\omega_1 \omega_2} \frac{C_g'}{2\sqrt{C_1' C_2'}}, \tag{2.40}$$

where C_1', C_2' and C_g' are the effective capacitances derived from C_1, C_2, and C_g.

2.4.2 Oscillator–Qubit Coupling

When coupling a transmon qubit to a microwave resonator (used for readout), a similar capacitive interaction yields the Jaynes–Cummings Hamiltonian:

$$H = \hbar\omega_r\, a^\dagger a - \frac{\hbar\omega_{10}}{2}\sigma_z + \hbar g\left(a^\dagger\sigma^- + a\,\sigma^+\right), \tag{2.41}$$

where ω_r is the resonator frequency, ω_{10} is the qubit transition frequency, and g is the coupling strength. In the dispersive regime ($|\Delta| \gg g$, with $\Delta = \omega_{10} - \omega_r$) the interaction reduces to

$$H_{\text{disp}} \approx \hbar\chi\sigma_z\, a^\dagger a, \tag{2.42}$$

with the dispersive shift defined as [42, 40]

$$\chi = -\frac{g^2}{\Delta}. \tag{2.43}$$

2.4.3 Qubit–Qubit Coupling

Capacitive coupling between two transmon qubits results in an interaction term of the form

$$H_{qq} = \hbar g\left(\sigma_1^+\sigma_2^- + \sigma_1^-\sigma_2^+\right), \tag{2.44}$$

which, in the appropriate rotating frame, effectively swaps excitations between the qubits.

2.5 Frequency Control via SQUID Tuning

For dynamic control of the qubit frequency, the single Josephson junction is replaced by a Superconducting QUantum Interference Device (SQUID) comprising two junctions in parallel [45, 44]. For identical junctions (with Josephson energy E_J), the total Josephson energy becomes

$$U = -E_J\left(\cos\phi_1 + \cos\phi_2\right) = -2E_J\cos\left(\frac{\phi_1+\phi_2}{2}\right)\cos\left(\frac{\phi_1-\phi_2}{2}\right). \tag{2.45}$$

following the standard derivation of the SQUID potential energy [48, 40]. The phase difference between the two junctions is constrained by the fluxoid quantization condition,

$$\phi_2 - \phi_1 = \frac{2\pi}{\Phi_0}\left(\Phi_{\text{ext}} + n\Phi_0\right). \tag{2.46}$$

with $\Phi_0 = h/2e$ the superconducting flux quantum [43, 45]. For simplicity and by choosing the appropriate branch (i.e., $n = 0$), the effective Josephson energy becomes

$$E_{J,\text{eff}} = 2E_J\cos\left(\frac{\pi\Phi_{\text{ext}}}{\Phi_0}\right), \tag{2.47}$$

and consequently the qubit transition frequency is tunable according to

$$\hbar\omega_{10} \approx \sqrt{8E_C\,E_{J,\text{eff}}} - E_C. \tag{2.48}$$

as derived in the seminal transmon work [20] and subsequent circuit-QED reviews [42, 39]. This tunability underpins modern flux-biased qubits such as the Xmon and tunable-coupler architectures [49, 50].

2.6 Typical Transmon Parameters

Practical design of the transmon entails balancing high coherence with sufficient anharmonicity to enable selective driving [20, 42]. In the transmon regime, characterized by a large ratio $E_J / E_C \sim$ 50–100, charge dispersion is exponentially suppressed while maintaining adequate anharmonicity for microwave control [51, 13]. Typical parameters for a flux-tunable transmon qubit implemented in planar or 3D architectures are consistent with experimental reports from superconducting circuit platforms [49, 52, 53]. Representative design values are:

- **Capacitance:** $C \approx 80\,\mathrm{fF}$
- **Charging Energy:** $E_C \approx 240\,\mathrm{MHz}$
- **Junction Critical Current:** $I_c \approx 40\,\mathrm{nA}$
- **Josephson Energy:** $E_J \approx 20\,\mathrm{GHz}$
- **Characteristic Impedance:** $Z \approx 320\,\Omega$
- **Typical Qubit Frequency:** $f_{10} \sim 5\text{–}7\,\mathrm{GHz}$
- **Anharmonicity:** $\eta \approx -220\,\mathrm{MHz}$

Additional parameters, such as coupling capacitances for the drive and readout lines and qubit–qubit coupling strengths, are engineered to optimize gate fidelity and dispersive readout performance [54, 42]. These values strike a compromise between coherence times exceeding $T_1, T_2 > 20\,\mu\mathrm{s}$ and control bandwidth sufficient for 10–20 ns single-qubit gates.

Chapter 3

Theoretical Foundations of X-Gate Calibration

3.1 Overview

This chapter assembles the theoretical models and analysis underlying a measurement-informed, end-to-end calibration pipeline for realizing a high-fidelity single-qubit X gate on a *flux-tunable* transmon biased at a chosen DC flux Φ_{bias} [20, 42, 13]. It provides priors, fit models, and error budgets that motivate and guide the experimental procedure, while taking a deliberately broader view than the next chapters: some methods and variants are included for completeness and context even when not used directly in the subsequent experiments.

We proceed from *spectroscopy*—including coarse flux–frequency mapping to locate $\omega_{01}(\Phi)$ and select a sweet-spot bias—through *T_1/T_2 characterization*, *Rabi/Ramsey calibration*, *pulse-shape optimization* (DRAG), and finally *randomized benchmarking* (RB) to verify and fine-tune the gate fidelity [55], with emphasis on models that connect measured observables to Hamiltonian parameters and noise processes so that every calibration knob has a clear physical target.

A resonant X gate is a π rotation about $\hat{x}$ on the Bloch sphere that, in the ideal, closed-system limit, implements

$$X_\pi : \quad |0\rangle \mapsto |1\rangle, \qquad |1\rangle \mapsto |0\rangle.$$

In a weakly anharmonic superconducting circuit, the achievable fidelity depends on (i) *spectral targeting* of the $|0\rangle \leftrightarrow |1\rangle$ transition, (ii) *coherence limits* set by T_1 and T_ϕ, (iii) *control* of amplitude, duration, and phase to suppress coherent errors, and (iv) *leakage* out of the computational subspace [20, 13]. We adopt the convention $T_2^{-1} = (2T_1)^{-1} + T_\phi^{-1}$ [2, 56] and quantify the decoherence-limited gate error for a gate of duration t_g by the back-of-the-envelope estimate

$$r_{\text{dec}} \approx \frac{t_g}{2T_1} + c_\phi \frac{t_g}{T_\phi}, \qquad c_\phi \in \left[\tfrac{1}{3}, \tfrac{1}{2}\right], \tag{3.1}$$

which serves as a practical floor when choosing t_g and optimizing pulse shape [13].

Notation. We treat the transmon as a weakly anharmonic ladder with $\{|0\rangle, |1\rangle, |2\rangle\}$ and flux-dependent transition frequencies $\omega_{01}(\Phi)$ and $\omega_{12}(\Phi) = \omega_{01}(\Phi) + \alpha(\Phi)$ (anharmonicity $\alpha(\Phi) < 0$). A microwave control tone at $\omega_d \approx \omega_{01}(\Phi_{bias})$ has a complex envelope $\varepsilon(t)$. The static flux bias is denoted Φ_{bias} and is typically chosen at (or near) a sweet spot where $\partial_\Phi \omega_{01}(\Phi)\big|_{\Phi_{bias}} \approx 0$ to minimize first-order flux-noise dephasing [20, 56]. We set $\hbar = 1$ unless noted; frequencies are in angular units unless "$/2\pi$" is explicitly written.

3.2 Spectroscopy and Qubit Frequency Identification

3.2.1 From device parameters to a frequency prior

For a flux-tunable transmon (split Josephson junction forming a SQUID), the effective Josephson energy depends on the applied flux as

$$E_J^{\text{eff}}(\Phi) \simeq E_{J,\Sigma} \cos\left(\pi \frac{\Phi}{\Phi_0}\right), \tag{3.2}$$

for a nearly symmetric SQUID; small junction asymmetry produces a smooth nonvanishing minimum but does not change the discussion below [45]. In the regime $E_J^{\text{eff}}(\Phi)/E_C \gg 1$, leading-order transmon perturbation theory yields

$$\omega_{01}(\Phi) \approx \sqrt{8E_J^{\text{eff}}(\Phi)E_C} - E_C, \quad \omega_{12}(\Phi) \approx \sqrt{8E_J^{\text{eff}}(\Phi)E_C} - 2E_C, \quad \alpha(\Phi) \equiv \omega_{12}(\Phi) - \omega_{01}(\Phi) \approx -E_C. \tag{3.3}$$

Thus, while $\alpha(\Phi)$ is only weakly flux dependent at leading order, the qubit frequency $\omega_{01}(\Phi)$ can be tuned over hundreds of MHz by varying Φ [20, 13]. Sweet spots occur near integer multiples of Φ_0 (for a symmetric SQUID), where $\partial_\Phi \omega_{01}(\Phi) = 0$ and the dephasing sensitivity to low-frequency flux noise is minimized [56, 57]. These formulas set practical priors for two-dimensional spectroscopy over (Φ, f) when locating the operating point and the carrier for time-domain control.

3.2.2 Dispersive readout and two-tone spectroscopy: signal model

We model a single-mode readout resonator of frequency ω_r coupled to the transmon with strength g in the dispersive limit $\Delta_r \equiv \omega_{01} - \omega_r$ satisfying $|\Delta_r| \gg |\alpha|$. Eliminating the qubit-resonator exchange to second order gives the familiar effective Hamiltonian

$$H_{\text{eff}} \approx \omega_r a^\dagger a + \frac{\tilde{\omega}_{01}}{2}\sigma_z + \chi a^\dagger a \sigma_z, \qquad \chi \approx -\frac{g^2\alpha}{\Delta_r(\Delta_r + \alpha)}. \tag{3.4}$$

A weak probe near ω_r therefore acquires a qubit-state-dependent phase shift; simultaneously driving the qubit at ω_d modulates $\langle\sigma_z\rangle$ and hence the demodulated readout quadrature [42]. In the linear-response limit,

$$y(\omega_d) \approx y_0 + K P_e(\Delta), \qquad \Delta \equiv \omega_d - \omega_{01}(\Phi_{bias}), \tag{3.5}$$

where P_e is the steady-state excited-state population under the qubit drive and K collects the readout gain and dispersive response. For a tunable device, one often acquires $y(\omega_d, \Phi)$ to map $\omega_{01}(\Phi)$ coarsely, and then fixes $\Phi = \Phi_{bias}$ (typically a sweet spot) for fine spectroscopy in ω_d only.

3.2.3 Optical Bloch steady state, Lorentzian lineshape, and power broadening

Under a continuous-wave (CW) drive with Rabi frequency Ω and detuning Δ, the optical Bloch equations (with longitudinal relaxation time T_1 and transverse coherence time T_2) give the steady-state solution

$$P_e(\Delta) = \frac{\frac{1}{2}\Omega^2 T_1 T_2}{1 + \Delta^2 T_2^2 + \Omega^2 T_1 T_2} = \frac{\frac{1}{2}s}{1 + \Delta^2 T_2^2 + s}, \qquad s \equiv \Omega^2 T_1 T_2. \tag{3.6}$$

Hence the spectroscopy signal (3.5) is a Lorentzian in Δ whose height saturates with increasing drive due to the finite T_1, and whose width exhibits *power broadening* [58, 59]. Defining the half width at half maximum (HWHM) by $P_e(\Delta_{1/2}) = \frac{1}{2}P_e(0)$ gives

$$\Delta_{1/2} = \frac{1}{T_2}\sqrt{1+s} \quad \Longrightarrow \quad \mathrm{FWHM} = \frac{2}{T_2}\sqrt{1+s}. \tag{3.7}$$

In the weak-drive limit ($s \ll 1$) this reduces to FWHM $\to 2/T_2$, so that a sufficiently gentle CW scan estimates T_2 directly. At stronger drive ($s \gtrsim 1$) the width grows as $\propto \sqrt{1+s}$ and the apparent center can shift slightly due to higher levels (see below).

If ω_{01} slowly fluctuates (e.g., $1/f$ flux or charge noise), the observed lineshape is a convolution of the Lorentzian (*homogeneous*) with a Gaussian distribution (standard deviation σ_f in Hz), i.e. a *Voigt* profile. A widely used approximation for its full width at half maximum is [60, 61]

$$\mathrm{FWHM}_V \approx 0.5346\,\Gamma_L + \sqrt{0.2166\,\Gamma_L^2 + \Gamma_G^2}, \qquad \Gamma_L = \frac{2}{T_2}, \quad \Gamma_G = 2\sqrt{2\ln 2}\,\sigma_f. \tag{3.8}$$

For flux-tunable transmons biased away from a sweet spot, slow $1/f$ flux noise induces qubit-frequency wander with standard deviation $\sigma_f \simeq |\partial_\Phi f_{01}|\sigma_\Phi$, where $f_{01} = \omega_{01}/2\pi$ and σ_Φ is the low-frequency RMS flux noise. This broadens the Gaussian component of the Voigt profile by increasing the frequency fluctuation σ_f . At a sweet spot with $\partial_\Phi f_{01} = 0$, the leading coupling to flux noise is quadratic [56, 57] and the lineshape distortion is correspondingly weaker, motivating operation at Φ_{bias} near a sweet spot for single-qubit gates.

3.2.4 Higher levels, AC–Stark shift, and multi-photon features

Because the transmon is only weakly anharmonic, a strong drive near ω_{01} admixes $|2\rangle$ and induces an AC–Stark shift of the $|0\rangle \leftrightarrow |1\rangle$ frequency. To leading order (treating $|2\rangle$ perturbatively),

$$\delta\omega_{01}^{(\mathrm{AC})} \approx -\frac{\Omega^2}{2\alpha}, \tag{3.9}$$

which shifts the apparent resonance to lower frequency for $\alpha < 0$. Additionally, a two-photon transition near $\omega_{02}/2$ with

$$\omega_{02} \approx \omega_{01} + \omega_{12} \approx 2\omega_{01} + \alpha \tag{3.10}$$

can produce a secondary dip [26, 16]. Both effects argue for using modest drive (small s) during coarse frequency identification and for verifying that the fitted f_{01} is stable versus drive power.

3.2.5 Practical sweep design and frequency planning

A typical superheterodyne arrangement uses a fixed local oscillator at f_{LO} and a single-sideband (SSB) $\pm f_{IF}$ generated by an AWG. The qubit-drive *carrier* is thus $f_d = f_{LO} \pm f_{IF}$. With a tunable transmon one benefits from a two-stage plan:

- **Flux–frequency map.** Sweep the DC flux (or the DAC value commanding the flux line) over a coarse range while stepping f_d coarsely to obtain a 2D map $y(f_d, \Phi)$ that reveals the chevron of $\omega_{01}(\Phi)$. Identify usable sweet spots (local extrema).
- **Fine spectroscopy at fixed flux.** Fix $\Phi = \Phi_{bias}$ at the selected sweet spot and perform a low-power two-tone sweep versus f_d only. As for fixed-frequency devices, pick f_{IF} away from DC (e.g., 50 MHz–250 MHz), null I/Q imbalance and LO leakage, and choose a frequency step $\delta f \lesssim \mathrm{FWHM}/6$ to oversample the linewidth.

A coarse-to-fine strategy (logarithmic zooms in Φ and f_d) reduces time while ensuring that $f_{01}(\Phi)$ is bracketed and a sweet-spot bias is selected [13].

3.2.6 Analytical fit models and parameter extraction

With $f = \omega/2\pi$ and a measured baseband observable $y(f)$, a minimal yet robust fit is

$$y(f) = y_0 + \frac{A}{1 + \left(\frac{f - f_{01}}{\gamma}\right)^2}, \qquad \mathrm{FWHM} = 2\gamma. \tag{3.11}$$

Under weak drive ($s \ll 1$), identify $T_2 \approx 1/(\pi\ \mathrm{FWHM}) = 1/(2\pi\gamma)$. If power broadening is non-negligible, introduce the saturation parameter s as a second shape parameter by fitting the *amplitude* and *width* jointly via (3.6)–(3.7), or use a Voigt profile when slow fluctuations dominate [58, 60].

Assuming Gaussian i.i.d. noise of standard deviation σ on $y(f)$, the Cramér–Rao scaling for the frequency-center estimator is approximately

$$\mathrm{SE}(f_{01}) \sim \sqrt{\frac{\gamma}{N\ \mathrm{SNR}}}, \qquad \mathrm{SNR} \equiv \frac{A}{\sigma}, \tag{3.12}$$

with N effective samples across the resonance. For small drifts $\delta f(t)$ between spectroscopy and subsequent time-domain calibrations, a software Z-frame update

$$\phi_Z(t) = \int_0^t \delta\omega(\tau)\, d\tau \tag{3.13}$$

keeps the control frame aligned without changing the physical carrier [13]. In tunable devices, one may also servo the DC flux (slow feedback on Φ) to maintain the sweet-spot condition that maximizes ${}_2T^*$.

3.2.7 Connecting spectroscopy to coherence and Purcell limits

The observed linewidth places a bound on T_2; combined with T_1 (from inversion recovery), it implies a pure-dephasing time via

$$\frac{1}{T_\phi} = \frac{1}{T_2} - \frac{1}{2T_1}. \tag{3.14}$$

For a tunable transmon, both T_1 and T_2 can be flux dependent. In particular, the Purcell rate varies with the detuning $\Delta_r(\Phi) = \omega_{01}(\Phi) - \omega_r$:

$$\Gamma_{\mathrm{Purcell}}(\Phi) \approx \kappa \left(\frac{g}{\Delta_r(\Phi)}\right)^2 \quad \Longrightarrow \quad T_{1,\mathrm{Purcell}}(\Phi) \approx \Gamma_{\mathrm{Purcell}}(\Phi)^{-1}, \tag{3.15}$$

so moving the bias changes the Purcell limit. Likewise, dephasing from flux noise grows as $|\partial_\Phi \omega_{01}(\Phi)|$ away from a sweet spot, broadening the low-power spectroscopy line beyond the homogeneous width [42, 62, 63]. These considerations jointly motivate choosing Φ_{bias} to balance long coherence with the desired frequency placement.

3.3 Rabi and Ramsey for Pulse Amplitude, Duration, and Frequency

3.3.1 Driven Two-Level Dynamics with Square and Shaped Pulses

We analyze the control of a single transmon transition as an effective two-level system in the rotating frame, with Hamiltonian

$$H_{\mathrm{rot}}(t) = \frac{\Delta}{2}\sigma_z + \frac{\Omega(t)}{2}\sigma_x, \tag{3.16}$$

where Δ is the detuning between the drive and the qubit transition, and $\Omega(t)$ is the (real) Rabi rate proportional to the drive envelope. For *on-resonance* control ($\Delta = 0$), $H_{\mathrm{rot}}(t)$ generates a rotation about $\hat{x}$ with time-dependent angular speed $\Omega(t)/2$. The unitary evolution is

$$U(t) = \exp\left[-\frac{i}{2}\vartheta(t)\,\sigma_x\right], \qquad \vartheta(t) \equiv \int_0^t \Omega(t')\,dt'. \tag{3.17}$$

Starting from $|0\rangle$, the excited-state probability at the gate end t_g is therefore

$$P_e(t_g) = \sin^2\left[\tfrac{1}{2}\vartheta(t_g)\right]. \tag{3.18}$$

Equation (3.18) yields the *area rule*: to realize an X_π gate (a π rotation about $\hat{x}$), the total rotation angle must satisfy $\vartheta(t_g) = \pi$ [2, 58].

Square pulse (rectangular envelope). With $\Omega(t) \equiv \Omega_0$ for $t \in [0, t_g]$, one has $\vartheta(t_g) = \Omega_0 t_g$. The π-pulse condition is

$$\Omega_0 t_g = \pi \quad \Longrightarrow \quad \Omega_0 = \frac{\pi}{t_g}. \tag{3.19}$$

Gaussian and flat-top envelopes. Let $\Omega(t) = A f(t)$, where A is a scalar amplitude and f is a fixed, dimensionless shape with $\int_0^{t_g} f(t)\,dt = \mathrm{A_f}$. Then $\vartheta(t_g) = A\,\mathrm{A_f}$, and the π-amplitude is

$$A_\pi = \frac{\pi}{\mathrm{A_f}}. \tag{3.20}$$

Two common choices are:

1. **Gaussian:** $f(t) = \exp\left[-(t - t_g/2)^2/2\sigma^2\right]$ (truncated at $[0, t_g]$ with $t_g \gtrsim 4\sigma$). Neglecting truncation, $\mathrm{A_f} = \sqrt{2\pi}\,\sigma$ and $A_\pi = \pi/(\sqrt{2\pi}\sigma)$.
2. **Flat-top with cosine ramps (Tukey window):** let $f(t) = 1$ on a flat segment of duration T_{flat}, and $f(t) = \frac{1}{2}\left[1 - \cos(\pi t/\tau_r)\right]$ on each ramp of duration τ_r at the beginning and end. The area is $\mathrm{A_f} = T_{\mathrm{flat}} + \tau_r$, hence $A_\pi = \pi/(T_{\mathrm{flat}} + \tau_r)$.

Shaped envelopes control spectral selectivity via the Fourier transform $\tilde{\Omega}(\omega) = \int_0^{t_g} \Omega(t) e^{i\omega t} dt$. For a square pulse,

$$\tilde{\Omega}_{\mathrm{sq}}(\omega) = \Omega_0\, e^{i\omega t_g/2}\, t_g\, \mathrm{sinc}\,\frac{\omega t_g}{2}, \qquad \mathrm{sinc}(x) \equiv \frac{\sin x}{x}, \tag{3.21}$$

so the main-lobe width scales as $\sim 2\pi/t_g$ and side-lobes decay only algebraically, which can drive near-resonant spectator transitions. In contrast, a (truncated) Gaussian has a Gaussian spectrum,

$$\tilde{\Omega}_{\mathrm{Gauss}}(\omega) \propto e^{-(\omega\sigma)^2/2}, \tag{3.22}$$

with exponentially suppressed wings, strongly reducing spectral leakage (e.g., to the $|1\rangle \leftrightarrow |2\rangle$ transition separated by the anharmonicity α).

3.3.2 Small-Detuning Effects: Toggling-Frame Analysis and Error Axes

When $\Delta = 0$ during a nominal X_π rotation, the rotation axis tilts away from $\hat{x}$ and the net gate accumulates coherent error. A convenient analysis uses the control-toggling frame defined by

$$U_c(t) = \exp\left(-\frac{i}{2}\vartheta(t)\,\sigma_x\right), \qquad \vartheta(t) = \int_0^t \Omega(t')\,dt'. \tag{3.23}$$

In this frame, the detuning term transforms as

$$H_I(t) = \frac{\Delta}{2}U_c^\dagger(t)\sigma_z U_c(t) = \frac{\Delta}{2}\left(\sigma_z\cos\vartheta(t) + \sigma_y\sin\vartheta(t)\right). \tag{3.24}$$

To first order in Δ (Magnus expansion), the error unitary at t_g is

$$U_{\rm err}^{(1)} = \exp\left(-\frac{i}{2}\left[\varepsilon_z\sigma_z + \varepsilon_y\sigma_y\right]\right), \qquad \varepsilon_z = \Delta\int_0^{t_g}\cos\vartheta(t)\,dt, \quad \varepsilon_y = \Delta\int_0^{t_g}\sin\vartheta(t)\,dt. \tag{3.25}$$

For a *square* π pulse with $\vartheta(t) = \Omega_0 t$ and $t_g = \pi/\Omega_0$, one obtains

$$\varepsilon_z = \Delta\int_0^{\pi/\Omega_0}\cos(\Omega_0 t)\,dt = 0, \qquad \varepsilon_y = \Delta\int_0^{\pi/\Omega_0}\sin(\Omega_0 t)\,dt = \frac{2\Delta}{\Omega_0}. \tag{3.26}$$

Thus a small detuning produces, to leading order, a spurious rotation about $\hat{y}$ of angle $\varepsilon_y = 2\Delta/\Omega_0$. For a *generic* symmetric π envelope, (3.25) still yields $\varepsilon_z \approx 0$ by symmetry, while ε_y is controlled by the "sine-area" accumulated during the pulse. The corresponding average gate infidelity for a small error unitary $U_{\rm err} = \exp[-i(\epsilon\hat{n}\cdot\boldsymbol{\sigma})/2]$ scales as [64, 28]

$$r_\Delta \approx \frac{\epsilon^2}{6} \quad\Longrightarrow\quad r_\Delta \approx \frac{\varepsilon_y^2}{6} = \frac{2}{3}\left(\frac{\Delta}{\Omega_0}\right)^2 \quad \text{(square } \pi \text{ pulse)}. \tag{3.27}$$

For shaped pulses, replace ε_y by (3.25); larger $|\varepsilon_y|$ correlates with greater spectral weight in the control envelope near DC in the toggling frame, which is minimized by smooth ramps.

3.3.3 Rabi Oscillations with Decoherence: Bloch-Equation Solution

Including relaxation and dephasing with rates $\Gamma_1 \equiv 1/T_1$ and $\Gamma_2 \equiv 1/T_2$, the Bloch equations for a *resonant* ($\Delta = 0$) step-on drive $\Omega(t) = \Omega_0\,\Theta(t)$ read

$$\dot{u} = -\Gamma_2\,u, \tag{3.28}$$

$$\dot{v} = -\Omega_0\,w - \Gamma_2\,v, \tag{3.29}$$

$$\dot{w} = \Omega_0\,v - \Gamma_1\,(w - 1), \tag{3.30}$$

with $(u, v, w) = (\langle\sigma_x\rangle, \langle\sigma_y\rangle, \langle\sigma_z\rangle)$ and initial condition $(0, 0, 1)$. Solving for v and w (the relevant subspace) yields an underdamped response when

$$\Omega_0 > \frac{|\Gamma_1 - \Gamma_2|}{2}, \qquad \Omega_d \equiv \sqrt{\Omega_0^2 - \frac{(\Gamma_1 - \Gamma_2)^2}{4}}, \qquad \gamma \equiv \frac{\Gamma_1 + \Gamma_2}{2}. \tag{3.31}$$

Writing the steady state (v_{ss}, w_{ss}) as

$$w_{ss} = \frac{\Gamma_1\Gamma_2}{\Gamma_1\Gamma_2 + \Omega_0^2}, \qquad v_{ss} = -\frac{\Omega_0\Gamma_1}{\Gamma_1\Gamma_2 + \Omega_0^2}, \tag{3.32}$$

the exact transient for $w(t)$ is

$$w(t) = w_{ss} + e^{-\gamma t}\left[A\cos(\Omega_d t) + \frac{\Omega_0 B + \frac{1}{2}(\Gamma_2 - \Gamma_1)A}{\Omega_d}\sin(\Omega_d t)\right], \qquad A \equiv 1 - w_{ss}, \quad B \equiv -v_{ss}. \tag{3.33}$$

The excited-state population is $P_e(t) = \frac{1}{2}[1 - w(t)]$. In the weak-damping limit ($\Gamma_{1,2} \ll \Omega_0$), (3.33) reduces to the familiar *damped Rabi* form,

$$P_e(t) \approx P_e^{(ss)} - \left[P_e^{(ss)} - P_e(0)\right] e^{-\gamma t}\cos(\Omega_d t - \phi), \qquad P_e^{(ss)} = \frac{\Omega_0^2 T_1 T_2}{2\left(1 + \Omega_0^2 T_1 T_2\right)}, \tag{3.34}$$

with a small phase lag $\phi \approx \arctan[(\Gamma_2 - \Gamma_1)/(2\Omega_0)]$ [59].

3.3.4 Ramsey Interferometry: Phase, Contrast, and Noise

A Ramsey sequence prepares an equatorial superposition, idles for time τ, then maps phase to population with a second $\pi/2$ pulse (phase-referenced). For a static detuning Δ, the excited-state probability is

$$P_e(\tau) = \tfrac{1}{2}\left[1 + V\cos(\Delta\tau + \phi_0)\right] W(\tau), \tag{3.35}$$

where $V \le 1$ captures pulse contrast and SPAM, and $W(\tau)$ is the coherence envelope due to dephasing. The *filter-function* description for phase noise $\beta(t)$ with one-sided PSD $S_\beta(\omega)$ gives

$$W(\tau) = \exp\left[-\chi(\tau)\right], \qquad \chi(\tau) = \frac{1}{\pi}\int_0^\infty d\omega\, \frac{S_\beta(\omega)}{\omega^2}\, 4\sin^2\frac{\omega\tau}{2}. \tag{3.36}$$

Two important limits are:

- **Quasi-static Gaussian detuning noise:** β constant over a shot with variance $\sigma_\Delta^2 \Rightarrow W(\tau) = \exp\left[-(\sigma_\Delta\tau)^2/2\right]$ (Gaussian decay).
- **White frequency noise:** $S_\beta(\omega) = 2\Gamma_\phi \Rightarrow W(\tau) = \exp[-\Gamma_\phi\tau]$ (exponential decay with pure-dephasing rate $\Gamma_\phi = 1/T_\phi$) [65].

Fitting (3.35) yields both Δ (slope of the fringe phase) and a dephasing time T_2^* via $W(\tau)$. A frequency update that nulls detuning in subsequent operations is

$$\omega_d^{(\mathrm{new})} = \omega_d^{(\mathrm{old})} - \Delta. \tag{3.37}$$

In the presence of strong pulses, additional *AC–Stark* phase shifts appear during driven segments [26]. For a weakly anharmonic ladder, the instantaneous shift is approximately $\delta\omega_{01}(t) \approx -\Omega^2(t)/(2\alpha)$ (with $\alpha < 0$ the anharmonicity), yielding a net phase $\phi_{AC} \approx \int \delta\omega_{01}(t)\, dt$. For a square π pulse, $\phi_{AC} \approx -(\Omega_0\pi)/(2\alpha)$; this can be compensated by a small frame update or by slightly retuning ω_d.

3.3.5 Spectral Selectivity and Leakage Metrics for Envelopes

Let α (rad/s) denote the anharmonicity separating $|1\rangle \leftrightarrow |2\rangle$ from $|0\rangle \leftrightarrow |1\rangle$. A crude spectral-leakage proxy is the ratio of the drive spectrum at $\omega = \alpha$ to its DC component,

$$\mathrm{L}_{\mathrm{spec}} \equiv \frac{|\tilde{\Omega}(\alpha)|}{|\tilde{\Omega}(0)|}. \tag{3.38}$$

For a square π pulse, $\tilde{\Omega}(0) = \Omega_0 t_g = \pi$ by (3.19), so

$$L_{\text{spec}}^{(\text{sq})} = \operatorname{sinc}\left(\frac{\alpha t_g}{2}\right). \tag{3.39}$$

For a Gaussian π pulse with area π and width σ, $\tilde{\Omega}(0) = \pi$ and

$$L_{\text{spec}}^{(\text{Gauss})} = \exp\left[-\tfrac{1}{2}(\alpha\sigma)^2\right]. \tag{3.40}$$

Hence smoother envelopes substantially suppress spectral weight at $\omega = \alpha$, reducing leakage and AC–Stark-induced phase errors [20].

3.3.6 Calibration Formulas and Small-Error Fidelity Budgets

The area rule (3.18) implies that for a fixed shape $f(t)$ and gate time t_g, the π-amplitude is given by (3.20). Two ubiquitous small coherent-error channels admit simple leading-order estimates:

Let the implemented rotation be $\vartheta = \pi + \delta\vartheta$. The unitary error is $U_{\text{err}} = \exp[-i(\delta\vartheta/2)\sigma_x]$, giving

$$r_{\text{ampl}} \approx \frac{(\delta\vartheta)^2}{6}. \tag{3.41}$$

For a fractional amplitude error ϵ on a square pulse, $\delta\vartheta = \epsilon\pi$ and $r_{\text{ampl}} \approx \epsilon^2\pi^2/6$.

Using (3.26), a square π pulse accumulates a spurious y^-rotation of angle $\varepsilon_y = 2\Delta/\Omega_0$, giving

$$r_\Delta \approx \frac{1}{6}\varepsilon_y^{\,2} = \frac{2}{3}\left(\frac{\Delta}{\Omega_0}\right)^2, \tag{3.42}$$

and analogously $r_\Delta \approx \varepsilon_y^2/6$ with ε_y from (3.25) for shaped pulses.

3.3.7 Decoherence-Limited Infidelity (Integral Estimates)

During a *resonant* X_π gate, the instantaneous excited-state probability for a square envelope is $P_e(t) = \sin^2\frac{\pi t}{2t_g}$. The integrated excited exposure is

$$\int_0^{t_g} P_e(t)\,dt = \int_0^{t_g} \sin^2\frac{\pi t}{2t_g}\,dt = \frac{t_g}{2}. \tag{3.43}$$

For $t_g \ll T_1$, the energy-relaxation contribution is therefore

$$r_{T_1} \approx \frac{1}{T_1}\int_0^{t_g} P_e(t)\,dt = \frac{t_g}{2T_1}. \tag{3.44}$$

Pure dephasing contributes an additional error proportional to the time spent in superposition; an orientation-averaged estimate yields

$$r_\phi \approx c_\phi \frac{t_g}{T_\phi}, \qquad c_\phi \in \left[\tfrac{1}{3}, \tfrac{1}{2}\right], \tag{3.45}$$

with the exact prefactor depending on the rotation axis and the dephasing spectrum. Combining both,

$$r_{\text{dec}} \approx \frac{t_g}{2T_1} + c_\phi \frac{t_g}{T_\phi}, \tag{3.46}$$

which provides a practical lower bound for the achievable gate error at duration t_g and guides the trade-off between speed and spectral selectivity.

3.4 Pulse Optimization: DRAG to Suppress Leakage and AC–Stark Errors

3.4.1 Three-Level Transmon and Leakage Mechanism

Include the lowest three levels $\{|0\rangle, |1\rangle, |2\rangle\}$ with $\omega_{12} = \omega_{01} + \alpha$ and $\alpha = \alpha(\Phi_{bias}) < 0$. In the frame rotating at ω_d and under the rotating-wave approximation (RWA), the control couples both $0 \leftrightarrow 1$ and $1 \leftrightarrow 2$:

$$\frac{H}{\hbar} = \Delta\,|1\rangle\langle 1| + (2\Delta + \alpha)\,|2\rangle\langle 2| + \frac{\varepsilon(t)}{2}\,|0\rangle\langle 1| + \sqrt{2}\,|1\rangle\langle 2| + \text{h.c.}\ , \tag{3.47}$$

with complex envelope $\varepsilon(t) = I(t) + iQ(t)$ and detuning $\Delta = \omega_d - \omega_{01}$. A narrow pulse spectrum keeps spectral weight near ω_{01}, but finite bandwidth leaks power to ω_{12}, driving $|1\rangle \leftrightarrow |2\rangle$ and inducing both (i) population leakage and (ii) AC–Stark shifts (phase errors).

3.4.2 DRAG Ansatz and Analytic Rationale

The Derivative Removal by Adiabatic Gate (DRAG) [16] ansatz augments a base in–phase envelope $I(t) = \Omega(t)$ with a quadrature proportional to its time derivative.

$$I(t) = \Omega(t), \qquad Q(t) = -\beta\,\dot{\Omega}(t), \tag{3.48}$$

for a real coefficient β. For smooth $\Omega(t)$ with vanishing endpoints, $\dot{\Omega}(t)$ shapes the spectrum to cancel the leading spectral weight at ω_{12}, thereby suppressing $|1\rangle \leftrightarrow |2\rangle$ off–resonant excitation. A comprehensive review of DRAG and its variants is given in Ref. [66].

To make this quantitative, model a weakly anharmonic superconducting qubit (e.g. a transmon) by its lowest three levels $\{|0\rangle, |1\rangle, |2\rangle\}$ and work in a frame rotating at the $|0\rangle \leftrightarrow |1\rangle$ drive frequency with the rotating–wave approximation. Denoting the anharmonicity by $\alpha = \omega_{12} - \omega_{01}$ (in angular–frequency units), the control–line couples $|0\rangle \leftrightarrow |1\rangle$ and, with a matrix–element factor $\sqrt{2}$, also $|1\rangle \leftrightarrow |2\rangle$. Introducing Pauli–like operators on subspaces (ij),

$$\sigma_x^{(ij)} = |i\rangle\langle j| + |j\rangle\langle i|, \quad \sigma_y^{(ij)} = i(|i\rangle\langle j| - |j\rangle\langle i|), \quad \sigma_z^{(ij)} = |i\rangle\langle i| - |j\rangle\langle j|,$$

the RWA Hamiltonian reads

$$\frac{H(t)}{\hbar} = \frac{\Omega(t)}{2}\,\sigma_x^{(01)} + \frac{Q(t)}{2}\,\sigma_y^{(01)} + \frac{\sqrt{2}\,\Omega(t)}{2}\,\sigma_x^{(12)} + \frac{\sqrt{2}\,Q(t)}{2}\,\sigma_y^{(12)} - \frac{\alpha}{2}\,\sigma_z^{(12)}. \tag{3.49}$$

The goal is to eliminate the $|1\rangle \leftrightarrow |2\rangle$ terms to leading order in the small parameter $|\Omega|/|\alpha| \ll 1$, such that the effective dynamics is confined to $\{|0\rangle, |1\rangle\}$.

Now, consider a time–dependent unitary $U(t) = e^{-iS(t)}$ with *small* generator $S(t)$ chosen to cancel the $|1\rangle \leftrightarrow |2\rangle$ couplings to first order. The transformed Hamiltonian is

$$\frac{H_{\text{eff}}}{\hbar} = \frac{H}{\hbar} + i\,[S,\, H/\hbar] + \dot{S} + O\left(\frac{\Omega^2}{\alpha^2}\right).$$

Take

$$S(t) = -\frac{\lambda(t)}{2}\,\sigma_y^{(12)}, \qquad \lambda(t) = O\left(\frac{\Omega}{\alpha}\right), \tag{3.50}$$

and use the commutators on the (12) subspace, $[\sigma_y^{(12)}, \sigma_z^{(12)}] = 2i\sigma_x^{(12)}$, $[\sigma_y^{(12)}, \sigma_x^{(12)}] = 2i\sigma_z^{(12)}$. Projecting the (12) block of H_{eff} and collecting $\sigma_{x,y}^{(12)}$ terms gives the cancellation conditions

$$\sigma_x^{(12)}: \quad \frac{\sqrt{2}}{2}\Omega(t) - \frac{\alpha}{2}\lambda(t) = 0, \tag{3.51}$$

$$\sigma_y^{(12)}: \quad \frac{\sqrt{2}}{2}Q(t) + \frac{1}{2}\dot{\lambda}(t) = 0. \tag{3.52}$$

Equation (3.51) fixes the instantaneous angle of the decoupling transformation,

$$\lambda(t) = \frac{\sqrt{2}}{\alpha}\Omega(t). \tag{3.53}$$

Because S is time–dependent, the inertial term $\dot{S}$ produces a residual (12)–y coupling; enforcing (3.52) then yields the quadrature that removes it,

$$Q(t) = -\frac{1}{\sqrt{2}}\dot{\lambda}(t) = -\frac{1}{\alpha}\dot{\Omega}(t). \tag{3.54}$$

Comparing (3.54) with the ansatz $Q(t) = -\beta\dot{\Omega}(t)$ identifies the *first–order optimal* DRAG coefficient at the chosen bias,

$$\beta_\star \approx \frac{1}{\alpha(\Phi_{\text{bias}})}, \qquad \text{(with } \alpha \text{ in rad/s)}. \tag{3.55}$$

Because $\alpha(\Phi)$ varies weakly with flux at leading order, $\beta_\star$ is typically only mildly flux dependent but should be rechecked if Φ_{bias} is changed by design.

Once (3.51)–(3.52) are imposed, the leading leakage terms are removed and the remaining $\{|0\rangle, |1\rangle\}$ dynamics is generated by H_{eff} projected onto (01). Retaining second–order terms in Ω/α gives an instantaneous AC–Stark detuning of the qubit transition of order $O(\Omega^2/\alpha)$,

$$\delta_{\text{AC}}(t) = -\frac{\left|\frac{\sqrt{2}}{2}\Omega(t)/2\right|^2 + \left|\frac{\sqrt{2}}{2}Q(t)/2\right|^2}{\alpha} = -\frac{\Omega^2(t) + Q^2(t)}{2\alpha}, \tag{3.56}$$

so that, up to an irrelevant identity shift,

$$\frac{H_{\text{eff}}^{(01)}(t)}{\hbar} = \frac{\Omega(t)}{2}\sigma_x^{(01)} + \frac{Q(t)}{2}\sigma_y^{(01)} - \frac{\delta_{\text{AC}}(t)}{2}\sigma_z^{(01)} + O\left(\frac{\Omega^3}{\alpha^2}, \frac{\Omega\ddot{\Omega}}{\alpha^2}\right). \tag{3.57}$$

A common refinement is to add a small, time–dependent detuning $\Delta(t) \approx -\delta_{\text{AC}}(t)$ to the drive or an equivalent phase pre–distortion, which cancels the Stark shift to the same order as (3.54).

In a tunable device, these microwave-drive-induced Stark shifts are independent of the static flux bias to leading order; nevertheless, if operation at different Φ_{bias} also changes α, the numerical compensation should be revisited.

Consequences and regime of validity. Without DRAG, the leakage amplitude into $|2\rangle$ scales as $O(\Omega/\alpha)$, giving a leakage probability $O(\Omega/\alpha)^2$. Choosing Q according to (3.54) cancels this order and pushes the leading leakage to $O(\Omega/\alpha)^2$ in amplitude, i.e. $O(\Omega/\alpha)^4$ in probability, consistent with the original DRAG analysis [16] and subsequent reviews [66]. The derivation assumes $|\Omega|/|\alpha| \ll 1$ and envelopes that are sufficiently smooth that $|\dot{\Omega}|/|\alpha| \ll |\Omega|$; Gaussian or cosine families with zero endpoints satisfy these

assumptions and automatically ensure that the net Y–area vanishes, so the gate axis is not tilted after the pulse.

Equation (3.54) is the time–domain manifestation of a *spectral zero* at ω_{12}: for smooth Ω, the DRAG choice engineers the complex envelope $\Omega(t) + iQ(t)$ so that its Fourier transform has a first–order notch at the undesired transition, while (3.56) accounts for the concomitant light shift that must be countered (either analytically via $\Delta(t)$ or experimentally via phase calibration).

3.4.3 FAST DRAG: Fourier–Ansatz Spectrum Tuning

Fourier–ansatz spectrum tuning DRAG (FAST DRAG) extends the DRAG idea by *designing the in–phase envelope in the frequency domain* before applying the derivative quadrature [11]. Instead of fixing $I(t)$ to a single parametric shape (Gaussian, cosine) and then adding $Q(t) \propto \dot{I}(t)$, FAST writes $I(t)$ as a short cosine series whose coefficients are chosen to *minimize the spectral energy* of the pulse over frequency intervals that contain undesired transitions (most notably the $|1\rangle \leftrightarrow |2\rangle$ leakage line near $\omega_{12} = \omega_{01} + \alpha$). After this frequency–domain shaping, the usual DRAG quadrature is applied to place a first–order spectral zero precisely at the leakage line. The result is a complex envelope whose spectrum exhibits a configurable *notch* of controllable width and depth around ω_{12}, yielding leakage suppression that is both stronger and more robust than with conventional DRAG.

Let t_p denote the pulse duration and $\Pi(x)$ the unit rectangular window supported on $[-\tfrac{1}{2}, \tfrac{1}{2}]$. FAST expands $I(t)$ in a compact raised–cosine basis,

$$I(t) = \sum_{n=1}^{N} c_n g_n(t), \qquad g_n(t) = \left[1 - \cos(2\pi n t/t_p)\right] \Pi\left(\frac{t}{t_p} - \frac{1}{2}\right), \tag{3.58}$$

with real coefficients $\{c_n\}$. Each g_n satisfies $g_n(0) = g_n(t_p) = \dot{g}_n(0) = \dot{g}_n(t_p) = 0$, promoting fast spectral roll–off and ensuring that the DRAG quadrature $Q(t) \propto \dot{I}(t)$ also vanishes smoothly at the boundaries.

Adopting the Fourier convention $\hat{f}(f) = \int_{-\infty}^{\infty} f(t) e^{-i2\pi ft}\, dt$, one has the closed–form transform

$$\hat{g}_n(f) = \int_0^{t_p} \left[1 - \cos(2\pi n t/t_p)\right] e^{-i2\pi ft}\, dt$$
$$= t_p \left[e^{-i\pi t_p f} \operatorname{sinc}(\pi t_p f) - \tfrac{1}{2} e^{-i\pi(f - n/t_p)t_p} \operatorname{sinc}\pi(f - n/t_p)t_p - \tfrac{1}{2} e^{-i\pi(f+n/t_p)t_p} \operatorname{sinc}\pi(f + n/t_p)t_p \right], \tag{3.59}$$

where $\operatorname{sinc}(x) = \sin x/x$. Equations (5.8)–(3.59) provide an analytic map between time– and frequency–domain parameters.

Let $\{[f_{\ell,j}, f_{h,j}]\}_{j=1}^{K}$ be baseband frequency intervals whose spectral power we wish to suppress (e.g. a finite–width neighborhood of $|\alpha|/2\pi$ to cover the $|1\rangle \leftrightarrow |2\rangle$ line and, optionally, a high–frequency stopband $[f_c, \infty)$ to limit bandwidth). With nonnegative weights $\{w_j\}$, define the Hermitian *Gram matrix*

$$K_{nm} = \sum_{j=1}^{K} w_j \int_{f_{\ell,j}}^{f_{h,j}} \hat{g}_n(f)\, \hat{g}_m(f)^*\, df, \qquad 1 \le n, m \le N. \tag{3.60}$$

Since $\hat{I}(f) = \sum_n c_n \hat{g}_n(f)$, the total spectral energy over the protected bands is

$$J(\mathbf{c}) = \sum_{j=1}^{K} w_j \int_{f_{\ell,j}}^{f_{h,j}} \left|\hat{I}(f)\right|^2 df = \mathbf{c}^{\mathsf{T}} \operatorname{Re} K\, \mathbf{c}, \qquad \mathbf{c} = (c_1, \ldots, c_N)^{\mathsf{T}} \in \mathbb{R}^N. \tag{3.61}$$

The gate area constraint fixing the rotation angle ϑ is linear because

$$\int_0^{t_p} g_n(t)\,dt = t_p \quad \Longrightarrow \quad \int_0^{t_p} I(t)\,dt = t_p \sum_{n=1}^{N} c_n = \vartheta, \tag{3.62}$$

where we absorb any overall amplitude into the coefficients. The coefficient vector $\mathbf{c}$ is thus obtained from

$$\min_{\mathbf{c}\in\mathbb{R}^N} \mathbf{c}^\top S\,\mathbf{c} \quad \text{subject to} \quad \mathbf{b}^\top\mathbf{c} = \vartheta/t_p, \qquad S = \tfrac{1}{2}\left(K + K^\top\right), \quad \mathbf{b} = \mathbf{1}. \tag{3.63}$$

Introducing a Lagrange multiplier μ gives $2S\mathbf{c} - \mu\mathbf{b} = \mathbf{0}$ and $\mathbf{b}^\top\mathbf{c} = \vartheta/t_p$, whence the *closed–form* solution

$$\boxed{\mathbf{c}^\star = \frac{\vartheta/t_p}{\mathbf{b}^\top S^{-1}\mathbf{b}}\, S^{-1}\mathbf{b},} \tag{3.64}$$

provided S is invertible (which holds for nondegenerate interval choices and N large enough to satisfy the area constraint).

After solving (3.64) to obtain $I(t)$, one applies the DRAG quadrature

$$Q(t) = -\beta\,\dot{I}(t), \qquad \beta_\star \approx \frac{1}{\alpha} \quad \text{(in angular–frequency units)}, \tag{3.65}$$

so that the complex envelope $E(t) = I(t) + iQ(t)$ has the frequency–domain form

$$\hat{E}(f) = \hat{I}(f) - i\hat{Q}(f) = \left[1 - 2\pi\beta f\right]\hat{I}(f). \tag{3.66}$$

Equation (3.66) shows that DRAG multiplies the FAST–shaped spectrum by a first–order polynomial that *zeros* the complex envelope at

$$f_{\text{zero}} = \frac{1}{2\pi\beta} \approx \frac{|\alpha|}{2\pi}, \tag{3.67}$$

i.e. at the leakage line in the rotating frame. Because $I(t)$ has already been optimized to minimize $|\hat{I}(f)|$ over a *finite* interval around f_{zero}, the product (3.66) produces a deep, wide spectral notch in $|\hat{E}(f)|$ around the undesired transition.

There are several benefits of using FAST-DRAG over conventional DRAG. First, FAST provides *interval–wise* suppression of spectral power, not merely a single–frequency cancellation. This yields robustness against detuning errors, power–dependent Stark shifts, and line broadening: even if the effective leakage frequency drifts within the protected band, $|\hat{E}(f)|$ remains small. Second, the interval set can include a high–frequency stopband to explicitly limit bandwidth and reduce ringing or DAC–front–end compression, improving pulse regularity without sacrificing area. Third, the calibration parameters are effectively decoupled: the area constraint (3.62) fixes the rotation angle, the intervals and weights determine notch location/width and high–frequency roll–off through S in (3.63), and β independently sets the DRAG zero via (3.66). This separation simplifies tuning and shortens calibration time in practice. Finally, because the basis (5.8) enforces $I(0) = I(t_p) = \dot{I}(0) = \dot{I}(t_p) = 0$, the DRAG quadrature (3.65) inherits smooth endpoints, mitigating transients and further reducing spectral leakage.

3.4.4 Higher–Derivative DRAG

Higher–derivative DRAG (HD–DRAG) extends the first–order DRAG construction by adding a finite series of time derivatives of the in–phase envelope to both the in–phase and quadrature channels, with coefficients

chosen so that the *complex* envelope exhibits a root of prescribed multiplicity at the leakage frequency. This realizes multi–order cancellation of the spectral density at the $|1\rangle\leftrightarrow|2\rangle$ line while preserving the target rotation area and smooth endpoints.

Let t_p be the pulse duration and let $I(t)$ denote a smooth base in–phase template with $I^{(m)}(0) = I^{(m)}(t_p) = 0$ for $m = 0, 1, \ldots, 2R - 1$. Define the order–R HD–DRAG controls by

$$I_R(t) = \sum_{k=0}^{\lfloor R/2 \rfloor} c_{2k}^{(R)} \frac{d^{2k} I}{dt^{2k}}(t) \frac{1}{\alpha^{2k}}, \qquad Q_R(t) = -\sum_{k=0}^{\lfloor (R-1)/2 \rfloor} b_{2k+1}^{(R)} \frac{d^{2k+1} I}{dt^{2k+1}}(t) \frac{1}{\alpha^{2k+1}}, \tag{3.68}$$

where $\alpha = \omega_{12} - \omega_{01}$ is the anharmonicity (in rad/s). The complex envelope $E_R(t) = I_R(t) + iQ_R(t)$ obeys, for baseband frequency f and the dimensionless variable $\xi = (2\pi f)/\alpha$,

$$\hat{E}_R(f) = \left[\sum_{k=0}^{\lfloor R/2 \rfloor} c_{2k}^{(R)} (-1)^k \xi^{2k} + \sum_{k=0}^{\lfloor (R-1)/2 \rfloor} b_{2k+1}^{(R)} (-1)^{k+1} \xi^{2k+1} \right] \hat{I}(f). \tag{3.69}$$

Choosing the coefficients so that the bracket equals $(1 - \xi)^R$ yields an *R-fold zero* of $\hat{E}_R$ at $\xi = 1$ (i.e. $f = \alpha/2\pi$) while maintaining $\hat{E}_R(0) = \hat{I}(0)$ and thus preserving the area to leading order. This is achieved by the *binomial* choice

$$c_{2k}^{(R)} = (-1)^k \binom{R}{2k}, \quad 0 \le 2k \le R, \qquad b_{2k+1}^{(R)} = (-1)^k \binom{R}{2k+1}, \quad 1 \le 2k+1 \le R, \tag{3.70}$$

for which

$$\sum_{k=0}^{\lfloor R/2 \rfloor} c_{2k}^{(R)} (-1)^k \xi^{2k} + \sum_{k=0}^{\lfloor (R-1)/2 \rfloor} b_{2k+1}^{(R)} (-1)^{k+1} \xi^{2k+1} = \sum_{m=0}^{R} (-1)^m \binom{R}{m} \xi^m = (1 - \xi)^R. \tag{3.71}$$

Hence $\hat{E}_R(f) = (1 - \xi)^R \hat{I}(f)$ with $\xi = (2\pi f)/\alpha$.

For $R = 1$ one recovers standard DRAG:

$$I_1(t) = I(t), \qquad Q_1(t) = -\frac{dI}{dt}(t)\frac{1}{\alpha}, \qquad \hat{E}_1(f) = \left(1 - \frac{2\pi f}{\alpha}\right)\hat{I}(f). \tag{3.72}$$

For $R = 2$ one obtains a *second–order* zero at $f = |\alpha|/2\pi$:

$$I_2(t) = I(t) - \frac{d^2 I}{dt^2}(t)\frac{1}{\alpha^2}, \qquad Q_2(t) = -2\frac{dI}{dt}(t)\frac{1}{\alpha}, \qquad \hat{E}_2(f) = \left(1 - \frac{2\pi f}{\alpha}\right)^2 \hat{I}(f). \tag{3.73}$$

For $R = 3$:

$$I_3(t) = I(t) - 3\frac{d^2 I}{dt^2}(t)\frac{1}{\alpha^2}, \qquad Q_3(t) = -3\frac{dI}{dt}(t)\frac{1}{\alpha} + \frac{d^3 I}{dt^3}(t)\frac{1}{\alpha^3}, \qquad \hat{E}_3(f) = \left(1 - \frac{2\pi f}{\alpha}\right)^3 \hat{I}(f). \tag{3.74}$$

At fixed envelope bandwidth, the on–resonance leakage amplitude without DRAG scales as $O(\varepsilon)$ with $\varepsilon = \|\Omega\|/|\alpha|$, yielding leakage probability $O(\varepsilon^2)$. First–order DRAG ($R = 1$) pushes the amplitude to $O(\varepsilon^2)$ (probability $O(\varepsilon^4)$). In general, the multiplicity–R zero implies an amplitude scaling $O(\varepsilon^{R+1})$ and thus leakage probability $O(\varepsilon^{2R+2})$, up to smoothness–controlled prefactors.

Equation (3.68) preserves the rotation area to leading order because $\hat{E}_R(0) = \hat{I}(0)$ by construction. The smooth–endpoint assumption $I^{(m)}(0) = I^{(m)}(t_p) = 0$ for $m \le 2R - 1$ ensures I_R and Q_R vanish at the boundaries, avoiding axis tilt and spurious high–frequency components. To the same accuracy, the instantaneous AC–Stark shift entering the effective qubit Hamiltonian,

$$\delta_{AC}(t) = -\frac{I_R^2(t) + Q_R^2(t)}{2\alpha} + O\left(\frac{\|I\|^4}{\alpha^3}\right), \tag{3.75}$$

can be compensated by a small phase pre–distortion $\phi(t)$ (or detuning schedule $\Delta(t) = \dot{\phi}(t)$) chosen such that $\Delta(t) \approx -\delta_{AC}(t)$, generalized to include higher–order terms consistently with R when needed. Because I_R and Q_R are linear in derivatives of I, δ_{AC} can be evaluated analytically from (3.68).

The asymptotic elimination underpinning HD–DRAG remains controlled when

$$\max_{0 \le m \le 2R} \frac{\|I^{(m)}\|}{|\alpha|^m} \ll 1, \qquad \text{and} \qquad \frac{\|\dot{I}\|}{|\alpha|} \ll 1, \tag{3.76}$$

with norms taken over $[0, t_p]$. Increasing R sharpens the spectral notch but raises sensitivity to analog bandwidth and digital sampling, since high–order derivatives amplify high–frequency noise. In practice, $R \in \{2, 3\}$ provides a favorable trade–off between leakage suppression and hardware robustness.

When the base envelope is the FAST ansatz $I_{FAST}(t)$ obtained by spectrum shaping over intervals containing $f_\alpha = \alpha/2\pi$, one simply substitutes $I \to I_{FAST}$ in (3.68) with the same binomial coefficients (3.70). The resulting spectrum is

$$\hat{E}_R^{FAST}(f) = \left(1 - \frac{2\pi f}{\alpha}\right)^R \hat{I}_{FAST}(f). \tag{3.77}$$

Thus HD–FAST DRAG inherits the *finite–width* spectral suppression of FAST while imprinting an R–fold *pointwise* zero at f_α. In a small neighborhood $f = f_\alpha + \delta f$, Taylor expansion gives

$$\hat{E}_R^{FAST}(f) = \hat{I}_{FAST}(f_\alpha + \delta f)\left(\frac{2\pi\,\delta f}{\alpha}\right)^R + O\left(|\delta f|^{R+1}\right), \tag{3.78}$$

so the integrated spectral weight leaking into the $|2\rangle$ line scales as $\int_{|\delta f| \le W} |\delta f|^{2R} |\hat{I}_{FAST}|^2 \, d\delta f$, which benefits multiplicatively from the FAST notch (reducing $|\hat{I}_{FAST}|$ across the interval) and the HD polynomial factor (raising the power of $|\delta f|$). Operationally, this improves robustness to detuning drifts, line broadening, and transient Stark shifts compared to first–order DRAG with a single–shape envelope.

Relative to $R = 1$, the order–R construction produces an R–fold zero at f_α, yielding leakage probability scaling $O(\varepsilon^{2R+2})$ versus $O(\varepsilon^4)$ and reducing the sensitivity to small frequency offsets by a factor that grows with R. For FAST envelopes, HD–FAST DRAG compounds this pointwise cancellation with interval–wise spectral minimization, delivering stronger leakage suppression at fixed gate time and improved tolerance to calibration errors without sacrificing the analytic solvability of the design.

3.4.5 Experimental Tuning Strategies

- **Leakage scan:** Prepare $|1\rangle$, apply N repetitions of the candidate X_π, measure a leakage proxy (e.g., shelving $|2\rangle$ via a two-tone readout or selective $|1\rangle \to |2\rangle$ π pulse). Sweep β and pick the minimum.
- **Phase calibration:** Use a Ramsey-with-X_π-in-the-gap sequence to detect phase shifts induced by the pulse; co-tune β and the drive frequency (or a small Z-frame correction) to null the phase error.
- **AllXY:** Run the 21-sequence AllXY diagnostic; minimize pattern-specific deviations by fine-tuning amplitude, frequency, and β.

3.5 T_1 and T_2 Measurements

3.5.1 T_1 (Energy Relaxation)

Consider a weakly driven two–level system with transition frequency ω_{01} coupled to a thermal bosonic bath of mean occupancy $n_{\mathrm{th}} = \left(e^{\hbar\omega_{01}/\mathrm{k}_B\mathrm{T}} - 1\right)^{-1}$. In the absence of drive, the dynamics is well described by a Lindblad master equation with upward and downward rates

$$\Gamma_\uparrow = n_{\mathrm{th}}\,\Gamma_1^0, \qquad \Gamma_\downarrow = (n_{\mathrm{th}} + 1)\,\Gamma_1^0, \tag{3.79}$$

where Γ_1^0 is the spontaneous emission rate in the vacuum limit. The longitudinal relaxation of the excited–state population $P_{\mathrm{e}}(t) = \langle e|\rho(t)|e\rangle$ obeys

$$\dot{P}_{\mathrm{e}}(t) = -(\Gamma_\uparrow + \Gamma_\downarrow)\,P_{\mathrm{e}}(t) + \Gamma_\uparrow, \qquad \Rightarrow \qquad P_{\mathrm{e}}(t) = P_{\mathrm{e}}^{(\infty)} + \left(P_{\mathrm{e}}(0) - P_{\mathrm{e}}^{(\infty)}\right) e^{-\mathrm{t}/\mathrm{T}_1}, \tag{3.80}$$

with approach–to–equilibrium time

$$T_1 = \frac{1}{\Gamma_\uparrow + \Gamma_\downarrow}, \qquad P_{\mathrm{e}}^{(\infty)} = \frac{\Gamma_\uparrow}{\Gamma_\uparrow + \Gamma_\downarrow} = \frac{1}{1 + e^{\hbar\omega_{01}/\mathrm{k}_B\mathrm{T}}}. \tag{3.81}$$

Inversion recovery prepares $P_{\mathrm{e}}(0) \approx 1$ by a calibrated π pulse, idles for a variable delay τ, and measures in the computational basis; repeating this for a grid of τ yields $P_{\mathrm{e}}(\tau)$, which is fit to (3.80). At millikelvin temperatures, $n_{\mathrm{th}} \ll 1$ and $P_{\mathrm{e}}^{(\infty)}$ is typically $\ll 10^{-2}$ but can be nonzero due to stray photons; retaining the floor $P_{\mathrm{e}}^{(\infty)}$ is therefore essential [59, 42, 62, 63].

Heterodyne dispersive readout has finite assignment fidelity. If $\epsilon_{\mathrm{g}\to\mathrm{e}}$ is the probability to misassign $|g\rangle$ as $|e\rangle$ and $\epsilon_{\mathrm{e}\to\mathrm{g}}$ the reverse, the measured excited–outcome probability is

$$P_{\mathrm{e}}^{\mathrm{meas}}(\tau) = \left(1 - \epsilon_{\mathrm{e}\to\mathrm{g}} - \epsilon_{\mathrm{g}\to\mathrm{e}}\right) P_{\mathrm{e}}(\tau) + \epsilon_{\mathrm{g}\to\mathrm{e}}. \tag{3.82}$$

Calibrations of $(\epsilon_{\mathrm{g}\to\mathrm{e}}, \epsilon_{\mathrm{e}\to\mathrm{g}})$ allow one to invert (3.82) and recover P_{e} from $P_{\mathrm{e}}^{\mathrm{meas}}$. In the absence of an explicit readout calibration, it is common (and usually sufficient) to absorb the affine distortion into two fit parameters:

$$P_{\mathrm{e}}^{\mathrm{meas}}(\tau) = P_\infty + \left(P_0 - P_\infty\right) e^{-\tau/\mathrm{T}_1}, \tag{3.83}$$

where $P_0 \simeq P_{\mathrm{e}}^{\mathrm{meas}}(0)$ captures the combined state–preparation and readout offset and $P_\infty \simeq P_{\mathrm{e}}^{\mathrm{meas}}(\tau \to \infty)$ the long–delay floor.

At each delay τ_{k} we acquire N_{k} single–shot outcomes, with n_{k} "excited" results. Under independent Bernoulli sampling with success probability $p_{\mathrm{k}} = P_{\mathrm{e}}^{\mathrm{meas}}(\tau_{\mathrm{k}})$, the log–likelihood of parameter vector $\vartheta = (T_1, P_0, P_\infty)$ is

$$\mathrm{L}(\vartheta) = \sum_{\mathrm{k}} \left[n_{\mathrm{k}} \ln p_{\mathrm{k}} + (N_{\mathrm{k}} - n_{\mathrm{k}}) \ln(1 - p_{\mathrm{k}})\right], \qquad p_{\mathrm{k}} = P_\infty + (P_0 - P_\infty) e^{-\tau_k/\mathrm{T}_1}. \tag{3.84}$$

Maximizing L (or equivalently minimizing the negative log–likelihood) yields an efficient estimator; asymptotic confidence intervals follow from the inverse observed Fisher information, i.e. the Hessian of $-\mathrm{L}$ at the optimum.

For accurate T_1, the π–pulse amplitude, duration, and DRAG parameter are first calibrated to maximize $P_{\mathrm{e}}(0)$ while minimizing leakage. The set of delays $\{\tau_{\mathrm{k}}\}$ should cover at least 0.1–5 times the expected T_1 on a quasi–logarithmic grid. To mitigate slow drifts, one interleaves delays rather than sweeping monotonically, and optionally includes frequent references at $\tau = 0$ and a long $\tau \gg T_1$ to track P_0 and P_∞. Average powers

are kept low enough to avoid heating; if "measurement–induced mixing" is a concern, the readout amplitude is reduced or the qubit is briefly parked away from the readout resonator frequency between repetitions. The principal microscopic channels contributing to $1/T_1$ include dielectric loss, quasiparticles, radiative loss through control and readout lines, and Purcell decay. In the dispersive regime of a qubit coupled to a readout resonator with coupling g, detuning $\Delta = \omega_{01} - \omega_r$, and resonator linewidth κ, the Purcell contribution scales approximately as

$$\Gamma_{\text{Purcell}} \approx \kappa \frac{g^2}{\Delta^2} \left(1 + \bar{n}_r\right), \tag{3.85}$$

where $\bar{n}_r$ is the mean photon number, highlighting the importance of Purcell filters and cryogenic attenuation.

3.5.2 T_2^* (Ramsey) and T_2 (Echo)

A Ramsey experiment applies a $\pi/2$ pulse, idles for τ, and applies a second $\pi/2$ of phase φ. In a frame rotating at the drive frequency ω_d, the qubit accumulates a phase

$$\Phi(\tau) = \Delta\tau + \int_0^\tau \delta\omega(t)\, dt, \qquad \Delta = \omega_d - \omega_{01}, \tag{3.86}$$

where $\delta\omega(t)$ is the stochastic frequency noise. Averaging over realizations defines the *coherence function*

$$C(\tau) = \left\langle e^{\,i\int_0^\tau \delta\omega(t)dt} \right\rangle = e^{-\chi_R(\tau)}, \tag{3.87}$$

so that for a final $\pi/2$ about x (i.e. $\varphi = 0$) the excited–state probability reads

$$P_e(\tau) = \frac{1}{2}\left[1 + V\,\mathrm{Re}\left\langle e^{-\chi_R(\tau)} e^{i(\Delta\tau+\phi_0)}\right\rangle\right] = \frac{1}{2}\left[1 + V\, e^{-\chi_R(\tau)}\cos(\Delta\tau + \phi_0)\right], \tag{3.88}$$

with visibility $V \le 1$ accounting for state–preparation and readout imperfections and ϕ_0 an overall phase. For Gaussian, stationary noise, the cumulant expansion yields the *filter–function* representation

$$\chi_R(\tau) = \frac{1}{\pi}\int_0^\infty \frac{S_{\delta\omega}(\omega)}{\omega^2} F_R(\omega,\tau)\, d\omega, \qquad F_R(\omega,\tau) = 4\sin^2\!\left(\frac{\omega\tau}{2}\right), \tag{3.89}$$

where $S_{\delta\omega}(\omega)$ is the one–sided power spectral density (PSD) of $\delta\omega(t)$. White frequency noise, $S_{\delta\omega}(\omega) = 2\Gamma_\phi$ (constant), gives $\chi_R(\tau) = \Gamma_\phi \tau$ and hence the exponential envelope $e^{-\tau/T_2^*}$ with $1/T_2^* = 1/(2T_1) + \Gamma_\phi$. Quasi–static (low–frequency) noise with Gaussian dispersion $\langle\delta\omega\rangle = 0$ and variance σ^2 yields $\chi_R(\tau) = \frac{1}{2}\sigma^2\tau^2$ and the Gaussian envelope $e^{-(\tau/T_G)^2}$ with $T_G = \sqrt{2}/\sigma$. For $1/f$–type noise, $S_{\delta\omega}(\omega) = 2\pi A/|\omega|$ between infrared and ultraviolet cutoffs (ω_ℓ, ω_h),

$$\chi_R(\tau) \approx \frac{A}{\pi}\tau^2 \ln\frac{1}{\omega_\ell \tau}, \qquad \omega_\ell\tau \ll 1 \ll \omega_h\tau, \tag{3.90}$$

which is well captured by a compressed–exponential envelope with $\beta \simeq 2$ over practical time windows.

For tunable transmons, quasi-static flux noise $\delta\Phi$ converts to detuning noise via $\delta\omega \approx (\partial_\Phi \omega_{01})\,\delta\Phi$ at fixed Φ_{bias}. If this dominates and is Gaussian, the Ramsey envelope is $e^{-(\tau/T_G)^2}$ with

$$T_G = \frac{\sqrt{2}}{\sigma_\Delta} = \frac{\sqrt{2}}{2\pi\,\sigma_f} = \sqrt{\frac{1}{2}}\,\frac{1}{\pi\,|\partial_\Phi f_{01}(\Phi_{\text{bias}})|\,\sigma_\Phi}, \tag{3.91}$$

where σ_Φ is the low-frequency RMS flux noise and $\sigma_f = |\partial_\Phi f_{01}|\,\sigma_\Phi$. At a sweet spot, $\partial_\Phi f_{01} = 0$ and first-order flux dephasing is suppressed; residual dephasing then arises from quadratic coupling (second derivative) and from non-flux sources, and is well probed by echo.

In a Hahn echo, a π pulse at $\tau/2$ reverses the sign of phase accumulation from noise components slower than $1/\tau$, strongly suppressing inhomogeneous dephasing. The corresponding filter function reads

$$\chi_{\mathrm{E}}(\tau) = \frac{1}{\pi}\int_0^{\infty} \frac{S_{\delta\omega}(\omega)}{\omega^2} F_{\mathrm{E}}(\omega,\tau)\, d\omega, \qquad F_{\mathrm{E}}(\omega,\tau) = 8\sin^4\frac{\omega\tau}{4}, \tag{3.92}$$

so that

$$P_{\mathrm{e}}(\tau) = \frac{1}{2}\left[1 + V' e^{-\chi_{\mathrm{E}}(\tau)}\right], \tag{3.93}$$

with no detuning oscillation ideally. For $1/f$ noise one finds, in the same asymptotic regime,

$$\chi_{\mathrm{E}}(\tau) \approx \frac{A}{\pi}\tau^2 \ln 2, \tag{3.94}$$

demonstrating a dramatic improvement over (3.90) by eliminating the infrared divergence. When white dephasing dominates, both Ramsey and echo decay exponentially with the same pure–dephasing rate, but echo removes slow, quasi–static contributions, leading to $T_2 \geq T_2^*$ with equality only when low–frequency noise is negligible.

After calibrating $\pi/2$ pulses (amplitude, duration, and DRAG parameter) and aligning the drive frequency ω_{d} close to ω_{01}, one acquires Ramsey data over a range of delays $\{\tau_k\}$ that spans several oscillation periods $2\pi/|\Delta|$ and extends to the decay time. It is advantageous to use phase cycling (e.g. alternating $\varphi = 0$ and $\varphi = \pi$) and to interleave delays to suppress drifts. A small, known detuning $\Delta/2\pi \sim 1$–2 MHz yields robust fringes and allows precise extraction of both Δ and the envelope $e^{-\chi_{\mathrm{R}}(\tau)}$. Echo is performed by inserting a calibrated π pulse at $\tau/2$; higher–order sequences (CP/CPMG or Uhrig) generalize the switching function $y(t) \in \{\pm 1\}$ that modulates phase accumulation.

For a total free-evolution time τ and N refocusing π pulses, the Uhrig dynamical decoupling (UDD) sequence places the flips of $y(t)$ at [67]

$$t_j^{\mathrm{UDD}} = \tau \sin^2\frac{\pi j}{2N+2}, \qquad j = 1, 2, \ldots, N, \tag{3.95}$$

so that $y(t)$ alternates sign at each t_j^{UDD} (starting with $y = +1$ at $t = 0$). For comparison, the CPMG sequence uses equally spaced flips at

$$t_j^{\mathrm{CPMG}} = \left(j - \tfrac{1}{2}\right)\frac{\tau}{N}, \qquad j = 1, 2, \ldots, N. \tag{3.96}$$

The UDD timings enforce N-th order cancellation of low-frequency noise in the filter function, thereby suppressing decoherence from slowly varying detuning [67, 68].

In general,

$$\chi(\tau) = \frac{1}{\pi}\int_0^{\infty} \frac{S_{\delta\omega}(\omega)}{\omega^2} \left|\int_0^{\tau} y(t)\, e^{i\omega t}\, dt\right|^2 d\omega, \tag{3.97}$$

with $y(t) \equiv 1$ for Ramsey and $y(t) = \mathrm{sgn}(t - \tau/2)$ for Hahn echo; CPMG corresponds to $y(t)$ flipping sign at each π pulse. Equation (3.97) is the basis for PSD reconstruction from coherence–decay data using different $y(t)$ and τ. As for T_1, measured probabilities should be corrected for readout assignment using (3.82) (or fit visibilities V, V'), and maximum–likelihood fits to (3.88) and the echo envelope $e^{-\chi_{\mathrm{E}}(\tau)}$ produce statistically efficient estimates with well–defined uncertainties.

3.5.3 Relation Between T_1, T_ϕ, and T_2

In the Markovian (white–noise) limit, the Bloch equations give the familiar identity

$$\frac{1}{T_2} = \frac{1}{2T_1} + \frac{1}{T_\phi}, \tag{3.98}$$

where $T_\phi = 1/\Gamma_\phi$ is the pure–dephasing time arising from white frequency noise. When inhomogeneous broadening dominates (e.g. quasi–static flux noise), the Ramsey envelope is better captured by a Gaussian or compressed exponential $e^{-(\tau/T_2^*)^\beta}$ with $\beta \simeq 2$, and one often reports an *effective* T_2^* from such a fit. Echo removes inhomogeneous broadening and recovers the Markovian relation (3.98) for the residual (fast) noise; measuring T_1 and T_2 then yields

$$\frac{1}{T_\phi} = \frac{1}{T_2} - \frac{1}{2T_1}, \tag{3.99}$$

which isolates the contribution of fast phase noise to dephasing. If the Ramsey decay is nonexponential due to $1/f$ noise, (3.98) does not strictly apply to T_2^*; in that case, (3.90) and (3.94) provide a more faithful connection between the noise PSD and the observed envelopes [2, 56].

Accurate extraction of T_1, T_2^*, and T_2 benefits from interleaved calibration points, randomized delay ordering, and sufficient averaging to make binomial uncertainties subdominant to systematics. Slow drift of ω_{01} biases Δ and can artificially shorten T_2^*; frequent re–centering of ω_d or virtual–Z phase tracking stabilizes the Ramsey phase. Heating or photon shot noise in the readout resonator can elevate $P_e^{(\infty)}$ and introduce extra dephasing via $S_{\delta\omega}(\omega) = 4\chi^2 S_n(\omega)$, where χ is the dispersive shift and S_n the photon–number PSD. Finally, the π–pulse in echo must be spectrally clean (e.g. DRAG–shaped) to avoid leakage; otherwise the recovered T_2 may be limited by control errors rather than environmental noise.

3.6 Randomized Benchmarking (RB) and Gate-Fidelity Verification

3.6.1 Standard RB (SRB)

Randomized benchmarking estimates the average performance of a gate set while being provably insensitive, to leading order, to state–preparation and measurement (SPAM) errors [55]. For a single qubit, SRB samples a sequence of m uniformly random Clifford gates $\{C_i\}_{i=1}^m$ from the single–qubit Clifford group Cl_1, followed by the unique inversion $C_{m+1} = (C_m \cdots C_1)^{-1}$ so that, in the absence of noise, $C_{m+1} C_m \cdots C_1 = \mathrm{I}$. Repeating this over many independent random sequences and measurement shots yields, as a function of m, the "survival" probability of returning to the initial state.

Model each implemented Clifford C as the ideal unitary channel $\mathcal{U}_C(\rho) = C\rho C^\dagger$ followed by a fixed (gate–independent and Markovian) noise channel Λ. The average over random sequences leverages the fact that the Clifford group forms a unitary 2–design. Consequently, the averaged channel

$$\mathcal{T}(\cdot) = \mathrm{E}_{C\in\mathrm{Cl}_1} \mathcal{U}_C^\dagger \circ \Lambda \circ \mathcal{U}_C(\cdot) \tag{3.100}$$

is a depolarizing channel on the traceless subspace. In the Pauli–Liouville representation, $\mathcal{T}$ acts as the identity on the identity component and as $p\,\mathrm{I}_3$ on the three–dimensional traceless subspace, i.e.

$$\mathcal{D}_p(\rho) = p\rho + (1-p)\frac{\mathrm{I}}{2}, \qquad -\tfrac{1}{3} \le p \le 1. \tag{3.101}$$

After m noisy Cliffords followed by the noisy inverse, the averaged effective channel is $\mathcal{D}_{p^m}$ acting on the prepared state. If ρ denotes the (possibly imperfect) prepared state and E the POVM effect corresponding to the measurement outcome labeled "survival", the predicted survival probability is

$$P(m) = \mathrm{Tr}\left[E\,\mathcal{D}_{p^m}(\rho)\right] = p^m \mathrm{Tr}(E\rho) + (1-p^m)\,\frac{\mathrm{Tr}(E)}{2}$$
$$= Ap^m + B, \qquad A = \mathrm{Tr}\left[E\left(\rho - \tfrac{I}{2}\right)\right], \quad B = \tfrac{1}{2}\mathrm{Tr}(E). \tag{3.102}$$

The parameters A and B absorb all SPAM imperfections, while the decay parameter p depends only on the average gate noise.

For a qudit of dimension d, the depolarizing channel $\mathcal{D}_p(\rho) = p\rho + (1-p)\,I/d$ has average gate fidelity to the identity

$$F_{\mathrm{avg}}(\mathcal{D}_p) = \frac{1+(d-1)p}{d}. \tag{3.103}$$

Accordingly, the average Clifford fidelity and error per Clifford are

$$F_C = \frac{1+p}{2}, \qquad r_C = 1 - F_C = \frac{1-p}{2} \qquad (d=2), \tag{3.104}$$

and, more generally, $r_C = \frac{d-1}{d}(1-p)$. If a compiled Clifford uses on average n_g primitive pulses, a common first–order attribution is $r_g \simeq r_C/n_g$, assuming gate–independent noise per primitive.

RB estimates the average performance of an entire gate set with just three fit parameters (A, B, p) and is robust to SPAM by construction, as seen in (3.102). Because uniform randomization "twirls" coherent unitary errors into stochastic depolarization, the single parameter p faithfully captures the *entanglement infidelity* averaged over the Clifford set. Compared to process tomography, RB scales more gently with system size, is less sensitive to model mismatch in SPAM, and provides a metric directly tied to circuit–level performance.

3.6.2 Interleaved RB (IRB)

To estimate the fidelity of a particular gate G (e.g. X_π), IRB interleaves G between the random Cliffords, producing a new decay parameter p_G [55]. Under gate–independent noise on the Cliffords and independent, gate–independent noise on G, the interleaved decay factors multiply:

$$p_{\mathrm{inter}} = p_G\, p_{\mathrm{ref}}, \tag{3.105}$$

where p_{ref} is the SRB decay without interleaving. Hence

$$p_G = \frac{p_{\mathrm{inter}}}{p_{\mathrm{ref}}}, \qquad F_G = \frac{1+(d-1)p_G}{d}, \qquad r_G = 1 - F_G = \frac{d-1}{d}\left(1 - p_G\right). \tag{3.106}$$

For $d = 2$ this reduces to $F_G = \frac{1}{2}\left(1 + \frac{p_{\mathrm{inter}}}{p_{\mathrm{ref}}}\right)$ as in the introductory summary. In the presence of mild gate dependence, (3.106) remains accurate up to corrections of order r_{ref}, and rigorous bounds exist in terms of p_{ref} and the spread of Clifford errors; in practice one reports the central estimate together with uncertainty bands reflecting these corrections.

3.6.3 SPAM robustness, model diagnostics, and fitting

Equation (3.102) shows that SPAM contributes only to (*A, B*) and not to *p*. This can be seen algebraically by expanding D_{p^m} as $p^{\mathrm{m}}(\ )+(1\ \ p^{\mathrm{m}})\ \mathrm{I}/d$ and grouping the SPAM–dependent constants. In experiments, one typically acquires *K* independently sampled random sequences per length *m* and *N* single–shot measurements per sequence. If $n_{\mathrm{m,k}}$ "survivals" are observed for sequence *k* at length *m*, a binomial maximum–likelihood fit to (3.102) with heteroscedastic weights is statistically efficient:

$$\mathrm{L}(A, B, p) = \sum_{\mathrm{m,k}}^{\mathrm{h}} n_{\mathrm{m,k}} \ln P(m) + (N - n_{\mathrm{m,k}}) \ln\ 1 - P(m)\ \ ^{\mathrm{i}}, \qquad P(m) = Ap^{\mathrm{m}} + B. \tag{3.107}$$

Residuals stratified by *m* and *k* serve as diagnostics: systematic deviations from a single exponential can indicate drift, gate dependence, leakage, or time–correlated noise, motivating the variants discussed below.

3.6.4 Decoherence limit and relation to T_1, T_2

If gate errors are dominated by Markovian amplitude damping and dephasing over a gate of duration t_{g}, the Bloch–vector contraction factors are $e^{-\mathrm{t}_g/\mathrm{T}_2}$ on *x, y* and $e^{-\mathrm{t}_g/\mathrm{T}_1}$ on *z*. The average gate infidelity of this noise relative to the identity is

$$r_{\mathrm{decoh}} \ = \ 1 - F_{\mathrm{avg}} \ = \ \frac{1}{2} - \frac{1}{6}\ 2e^{-\mathrm{t}_g/\mathrm{T}_2} + e^{-\mathrm{t}_g/\mathrm{T}_1} \ \approx \ \frac{t_{\mathrm{g}}}{3T_2} + \frac{t_{\mathrm{g}}}{6T_1} \ = \ \frac{t_{\mathrm{g}}}{3T_\phi} + \frac{t_{\mathrm{g}}}{3T_1}, \tag{3.108}$$

to leading order in $t_{\mathrm{g}}/T_{1,2}$. Equation (3.108) provides a coherence "floor" against which RB–extracted errors can be compared; values near this floor indicate noise dominated by T_1/T_ϕ, while significant excess points to control or leakage errors.

3.6.5 Leakage-robust RB and extraction of leakage/seepage

If population can escape the computational subspace (e.g. $|2\rangle$ in a transmon), the survival curve can acquire an additional exponential associated with the net population flow between the computational subspace ($\mathrm{H}_{\mathrm{comp}}$) and the leakage manifold ($\mathrm{H}_{\mathrm{leak}}$) [69]. Let ℓ denote the per–step leakage probability out of $\mathrm{H}_{\mathrm{comp}}$ and *s* the seepage probability back into it. Averaged over random Cliffords acting trivially on $\mathrm{H}_{\mathrm{leak}}$, the population in $\mathrm{H}_{\mathrm{comp}}$ obeys

$$P_{\mathrm{comp}}(m) = \frac{s}{\ell + s} + P_{\mathrm{comp}}(0) - \frac{s}{\ell + s}\ (1 - \ell - s)^{\mathrm{m}}, \tag{3.109}$$

while coherence within $\mathrm{H}_{\mathrm{comp}}$ decays as p^{m} as before. The measured survival thus fits

$$P(m) \ = \ Ap^{\mathrm{m}} \ + \ C\,(1 - \ell - s)^{\mathrm{m}} \ + \ B, \tag{3.110}$$

where the second exponential isolates leakage/seepage. Estimating (ℓ, *s*) is valuable because leakage errors typically compound across circuits, are poorly captured by Pauli error models, and can corrupt error–correcting codes unless explicitly suppressed.

3.6.6 Unitarity benchmarking and coherence of noise

SRB collapses all errors into a single scalar *p*, obscuring whether imperfections are predominantly coherent (unitary miscalibrations) or stochastic [70]. The *unitarity u* of a channel Λ quantifies the coherence of noise via the average squared contraction of Bloch vectors:

$$u \ = \ \frac{1}{d^2 - 1}\ \mathrm{Tr}\ T^{\dagger}T\ , \tag{3.111}$$

where T is the $(d^2-1)\times(d^2-1)$–block of the Pauli–transfer matrix of Λ acting on traceless operators. Purity/Unitary RB protocols access u by replacing the final projective measurement with an estimate of the output state purity (or by introducing tomography over random bases); the resulting observable decays as $A' u^{m-1} + B'$. Combining r from SRB and u bounds the coherent component of the noise and guides whether further calibration (reducing coherent over/under–rotations) or passive suppression (e.g. through randomized compiling) is likely to help.

3.6.7 Simultaneous and two-qubit RB

To probe crosstalk and addressability, concurrent SRB is run on multiple qubits by drawing independent random Clifford sequences and executing them simultaneously [55]. Degradation of p relative to isolated RB quantifies spectator–induced errors. For entangling operations, two–qubit RB over Cl_2 yields p from which the average two–qubit Clifford fidelity follows via (3.103) with $d = 4$. Interleaving a particular two–qubit gate (e.g. CZ, iSWAP) within two–qubit RB then estimates its fidelity in the same manner as (3.106).

3.6.8 Experimental procedure

A practical SRB/IRB experiment proceeds as follows. First, calibrate the primitive gate set (including DRAG parameters and virtual-Z phases) and compile a lookup table mapping each single–qubit Clifford to a short sequence of primitives. Next, choose a set of sequence lengths $\mathrm{M} = \{m_j\}$ spanning the dynamic range of the decay; geometric spacing (e.g. $m \in \{1, 2, 4, 8, 16, 32, \ldots\}$) is convenient. For each $m \in \mathrm{M}$, sample K independent random sequences, compute the inverse Clifford, and execute with N measurement shots per sequence. Interleave the order of (m, k) to mitigate drift, and include $m = 0$ references to monitor SPAM drifts in A and B. For IRB, repeat with the chosen G_i inserted between random Cliffords. Fit the resulting binomial data to (3.102) (and to its IRB analogue), extract (A, B, p) and $(A', B', p_{\mathrm{inter}})$, and convert to fidelities via (3.103)–(3.106). Bootstrap resampling across sequences provides conservative confidence intervals that reflect sequence–to–sequence fluctuations beyond shot noise.

3.6.9 Gate dependence, drift, and model limitations

The single–exponential SRB model assumes gate–independent, time–stationary noise. Mild gate dependence manifests as excess scatter across sequences for fixed m and, if strong, as systematic curvature in semi–log plots of the averaged survival. Slow drift can bias p low; interleaving sequence execution and re–randomizing frequently mitigates this. Time–correlated noise can be diagnosed by varying the total experiment duration at fixed m or by inserting idles between gates; deviations motivate extensions such as cycler benchmarking or randomized compiling, which tailor twirling to the native gate–cycle structure of the hardware and render coherent errors effectively stochastic at the circuit level.

3.6.10 Connecting RB to per–primitive errors

While $r_g \approx r_c / n_g$ is a useful first–order attribution, it presumes independent and identically distributed errors per primitive. A sharper analysis fits SRB using multiple compiler decompositions with differing primitive counts (e.g. $(n_X, n_Y, n_Z, n_{\mathrm{virt\,Z}})$) and solves a linear regression for per–primitive entanglement infidelities. Interleaving each primitive in IRB further isolates its contribution. These strategies help diagnose whether errors are dominated by a specific control primitive or by idles and frame updates.

3.7 Putting It All Together: A Practical Calibration Loop

In this section we assemble the preceding techniques into a concrete, reproducible loop that carries a device from first light to reportable single-qubit gate fidelities. The loop is written for a tunable transmon at a chosen flux bias Φ_{bias} and a standard cQED readout chain (Chapter 4), but the same logic applies to fixed-frequency qubits with the obvious omissions. Where relevant, we state both the physical quantity inferred at each step and the control artifact updated in software (drive frequency, amplitudes, phases, or pulse-shape parameters). The sequence is ordered to minimize backtracking, and each step yields quantitative diagnostics that propagate into a gate-error budget at the end of the section.

3.7.1 Recommended order of operations

Flux–frequency mapping and sweet-spot selection. Starting from millikelvin base temperature, we map $f_{01}(\Phi)$ via two-tone spectroscopy over a coarse grid of DC flux biases. This identifies flux-insensitive sweet spots (vanishing first derivative $\partial f_{01}/\partial\Phi \approx 0$) and guards against spurious avoided crossings. We pick Φ_{bias} to balance coherence (maximizing T_1, T_2^* at or near a sweet spot) and frequency placement (spectral avoidances relative to neighboring qubits and readout modes). Deliverables: $f_{01}(\Phi)$, candidate Φ_{bias}, and a blacklist of crowding regions.

Two-tone (CW) spectroscopy at fixed Φ_{bias}. Holding the bias fixed, we sweep a weak drive to pin f_{01} and—where feasible—locate the $|e\rangle \leftrightarrow |f\rangle$ transition to estimate the anharmonicity $\alpha/2\pi = f_{12} - f_{01}$. This sets the initial drive carrier $f_d \approx f_{01}$ and provides a first estimate of the detuning landscape for leakage-aware pulse design.

Ramsey (coarse carrier and dephasing). We run short-interval Ramsey sequences to refine the qubit drive frequency ω_d and obtain T_2^* at the chosen bias. Fitting the Ramsey fringe frequency yields residual detuning $\Delta = \omega_d - \omega_{01}$; we update the software frame or carrier to null Δ within a few kHz. Deliverables: (ω_d, T_2^*).

Rabi calibration (amplitude and duration). Using Gaussian (or cosine) reference envelopes, we scan amplitude and/or duration to determine $\pi/2$ and π pulse settings $(A_{\pi/2}, t_{\pi/2})$ and (A_π, t_π). We preferentially report and reuse the $t_{\pi/2}$ setting and derive other rotations by phase updates plus repetition to avoid nonlinearity at high amplitudes. The extracted Rabi rate also serves as a line-gain proxy (cf. Eq. (4.2)).

Fine amplitude & frequency via error amplification. We refine $(A_{\pi/2}, \omega_d)$ using error-amplifying sequences that make the observed probability linearly sensitive to over/under-rotation or residual detuning (e.g., Bang–Bang for amplitude; $X_{\pi/2}$, $X_{\pi/2}$, $X_{-\pi/2}$, $X_{-\pi/2}$ followed by a $Y_{\pi/2}$ readout for phase). The objective is to push systematic errors below the incoherent floor set by $T_{1,2}$.

Leakage suppression and phase control: DRAG, FAST-DRAG, and HD-DRAG. At this point we switch from a reference envelope to a leakage-aware pulse family. Classical DRAG uses an out-of-quadrature component proportional to the time derivative of the in-phase envelope to suppress diabatic excitation of the nearest leakage transition while compensating AC–Stark phase (first-order counterdiabatic correction) *in situ*. We perform a sweep of the DRAG scaling coefficient to minimize measured leakage and then co-tune the Z frame (virtual-Z) or a small carrier detuning to cancel any residual phase error (DRAG-P vs. DRAG-L workflow). Modern spectrum-engineering variants—FAST-DRAG and HD-DRAG—extend this step by shaping the *frequency-domain* content of the envelope to enforce deep spectral notches around $|\alpha|/2\pi$

(and optionally beyond a bandwidth cutoff), or by adding higher time derivatives in a controlled way to sharpen the notch and reduce leakage at short gate times. We calibrate the corresponding parameters (notch center/widths and weights for FAST-DRAG; higher-derivative weights for HD-DRAG) using leakage-sensitive RB and phase-sensitive interleaves, keeping virtual-Z for deterministic phase control. These choices follow the DRAG framework and its generalizations to multi-diabatic suppression while preserving implementability on bandwidth-limited hardware.

Coherence floor at operating point. We measure T_1, Hahn-echo T_2 (and optionally CPMG for noise spectroscopy) at Φ_{bias}. These set the best-case, decoherence-limited error for a gate of duration t_g and determine whether further pulse speeding is beneficial (until leakage or control distortion dominates).

Randomized benchmarking and leakage quantification. Finally, we benchmark with standard (SRB) and interleaved RB to extract the depolarizing parameter p and per-gate error ϵ_g, optionally complemented by purity RB to separate coherent from incoherent components. We run leakage RB or three-state readout to quantify the $|f\rangle$population growth per Clifford and convert it to leakage per native gate. Small iterative adjustments (amplitude, derivative scaling, notch position/widths, virtual-Z offsets) are applied to maximize p (and p_C/p_{ref} in IRB). The outcome is a calibrated pulse set and a defensible error budget.

3.7.2 Gate error budget

We decompose the average gate infidelity $r_{\text{gate}} = 1 - F_{\text{avg}}$ into physically interpretable contributions

$$r_{\text{gate}} \approx r_{\text{dec}} + r_{\text{leak}} + r_{\text{ampl}} + r_{\text{detune}} + r_{\text{phase}} + r_{\text{xtalk}} + r^{(\text{tomog})}_{,\text{SPAM}} \tag{3.112}$$

with the following operational definitions and estimators [13, 66, 11, 55, 64, 69, 70]:

Decoherence (r_{dec}). For a single-qubit gate of length t_g subject to amplitude damping and dephasing, a first-order bound is

$$r_{\text{dec}} \simeq \frac{t_g}{3T_1} + \frac{t_g}{3T_\phi},$$

where $T_\phi^{-1} = T_2^{-1} - (2T_1)^{-1}$. This sets the irreducible floor given the chosen t_g and measured (T_1, T_2).

Leakage (r_{leak}). Estimated from the slope of the $|f\rangle$population in leakage RB or from direct interleaved leakage amplifiers (start in $|1\rangle$). DRAG suppresses first-order diabatic coupling to $|f\rangle$; FAST-DRAG/HD-DRAG further enforce spectral zeros around $|\alpha|/2\pi$ and can reduce leakage by an order of magnitude at sub-10-ns gate times when properly tuned.

Amplitude miscalibration (r_{ampl}). Extracted from amplitude error–amplification sequences (e.g., repeated $X_{\pi/2}$) that are linearly sensitive to over/under-rotation; corrected by re-scaling the envelope amplitude or duration.

Residual detuning (r_{detune}). Inferred from phase error–amplification sequences and short Ramsey interleaves. Mitigated by updating the software frame and, for DRAG-L/FAST-DRAG/HD-DRAG, by small virtual-Z adjustments pre/post gate.

Phase systematics (r_{phase}). AC–Stark shifts or imperfect DRAG balancing produce deterministic phase offsets. These are nulled either by DRAG-P tuning (shared with HD-DRAG variants) or by explicit virtual-Z calibration alongside leakage minimization.

Crosstalk and crowding (r_{xtalk}). Assessed by driving spectators while monitoring the target qubit (and vice versa). FAST-DRAG allows placement of multiple spectral notches and a bandwidth cutoff, which helps mitigate crowding/crosstalk without lengthening t_g.

SPAM (for tomography only) ($r_{\mathrm{SPAM}}^{(\mathrm{tomog})}$). Not part of RB's ϵ_g, but necessary when converting process tomography into gate fidelity. Three-state readout calibration and assignment-matrix inversion reduce bias if leakage is non-negligible.

Bias dependence. For tunable devices, each term should be regarded as a function of Φ_{bias}. In particular, $r_{\mathrm{dec}}(\Phi)$ tracks $T_{1,2}(\Phi)$ and $r_{\mathrm{leak}}(\Phi)$ follows $\alpha(\Phi)$ and local spectral crowding. Running the loop at neighboring biases provides a Pareto front between speed (short t_g), coherence, and spectral isolation.

Chapter 4

Experimental Setup

4.1 Overview

This chapter documents the full control-and-measurement stack used to implement the calibration protocols of Chapter 3. The hardware follows the standard cQED architecture [42, 12, 13]: a weakly anharmonic transmon qubit is dispersively coupled to a readout resonator, coherent microwave pulses realize unitary control through a dedicated drive line, and the qubit state is inferred from the state-dependent resonator response, which is amplified cryogenically and at room temperature before coherent demodulation [71, 72]. Figure 4.1 summarizes the cryogenic wiring and the room-temperature instrumentation.

All instruments are phase-locked to a common 10 MHz reference and share deterministic triggers and time tags. Pulse synthesis and data acquisition are orchestrated by a Python DSL (Section 4.4) that compiles experiment descriptions into hardware-specific sequencer programs with cycle-accurate timing. In the remainder of this chapter, we detail the quantum processor, the cryogenic signal chain, and the room-temperature modules that generate and digitize pulses, with particular attention to the flow of information (digital instructions $\rightarrow$ analog voltages $\rightarrow$ microwave fields $\rightarrow$ digitized I/Q samples) along the control and readout paths.

4.2 Quantum processor: QuantWare *Soprano*

The experiments target a QuantWare *Soprano*[1] chip: a multi-qubit transmon processor in which each qubit is provided with a dedicated microwave drive line, an individual flux-bias line, and a dispersively coupled readout resonator. QuantWare reports typical single-qubit energy-relaxation times on the order of 60 µs and specifies infrastructure requirements that align with our setup (RF sources for qubit control and baseband/low-frequency sources for flux biasing), which guided our instrument selection and wiring. These device-level capabilities—especially independent flux lines—are essential for this thesis, where we evaluate pulse-engineering techniques at fixed bias points of a *flux-tunable* transmon. Figure 4.2 shows the optical micrograph of the QuantWare *Soprano* die used in this work.

[1] https://www.quantware.com/product/soprano

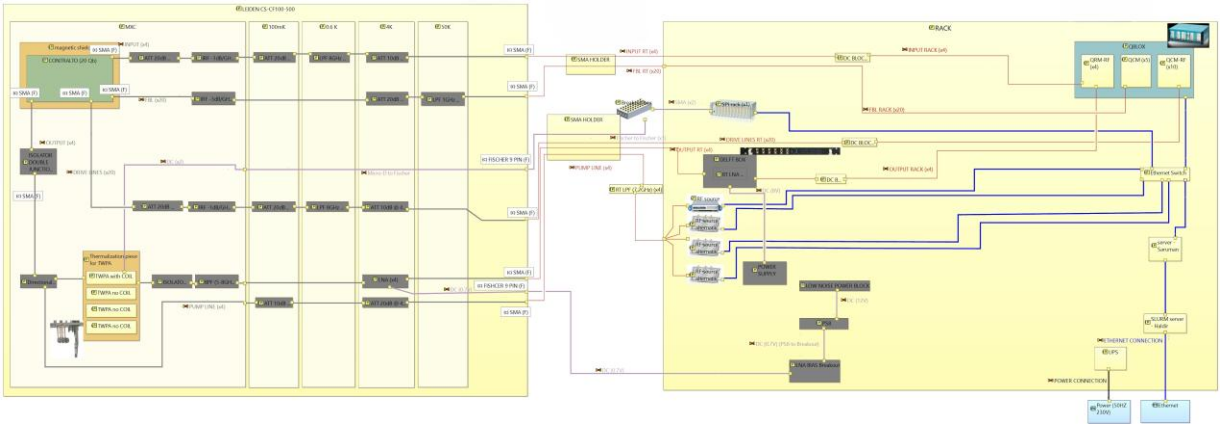


Figure 4.1: **Schematic of the experimental setup.** Left: cryogenic wiring inside a dilution refrigerator with input (control) lines, a dedicated pump line, and output (readout) lines including a traveling-wave parametric amplifier (TWPA), cryogenic isolators, and a high-electron-mobility transistor (HEMT) amplifier anchored at 4 K. Right: room-temperature instrumentation based on Qblox Cluster modules for RF control and readout, DC/low-frequency sources for flux biasing, and the control server.

4.3 Cryogenic platform

The device is hosted in a Leiden Cryogenics CF-CS110[2] cryogen-free dilution refrigerator. In its standard configurations the CF-CS110 provides a base temperature below 9 mK with cooling power exceeding 1000 µW at 100 mK, enabled by two pulse-tube stages (≥ 5.4 W at 4.2 K) and a high-throughput pumping unit; typical options include large mixing-chamber plates and fast sample-change top-loading probes. These specifications set the thermal budget for input attenuation and output isolation and ensure that resonator and qubit modes can be thermalized close to the quantum limit [73].

Thermal stages at approximately 50 K, 4 K, ∼0.8 K (still), and ∼100 mK (mixing chamber) provide successive heat-sinking points for attenuators, infrared/low-pass filters, bias tees, and passive microwave components. The chip is mounted in a gold-plated copper enclosure within a Cryoperm shield at MXC to suppress magnetic noise.

Each qubit drive line is heavily attenuated and filtered to thermalize room-temperature Johnson noise and suppress out-of-band radiation [13]:

$$P_{\mathrm{qb}} \approx P_{\mathrm{RT}} - \sum_{i} A_i - \sum_{j} L_j, \tag{4.1}$$

where P_{RT} is the room-temperature generator output in dBm, A_i collects cryogenic attenuators anchored at intermediate stages, and L_j accounts for line losses and filter insertion. The effective on-chip Rabi rate,

$$\Omega_R = \frac{2}{\hbar} \left| \langle 0 | \hat{\mu} | 1 \rangle \right| E_{\mathrm{qb}}, \tag{4.2}$$

implicitly calibrates the net line gain through the measured field E_{qb} at the qubit.

[2] https://www.leidencryogenics.nl/cf-cs110

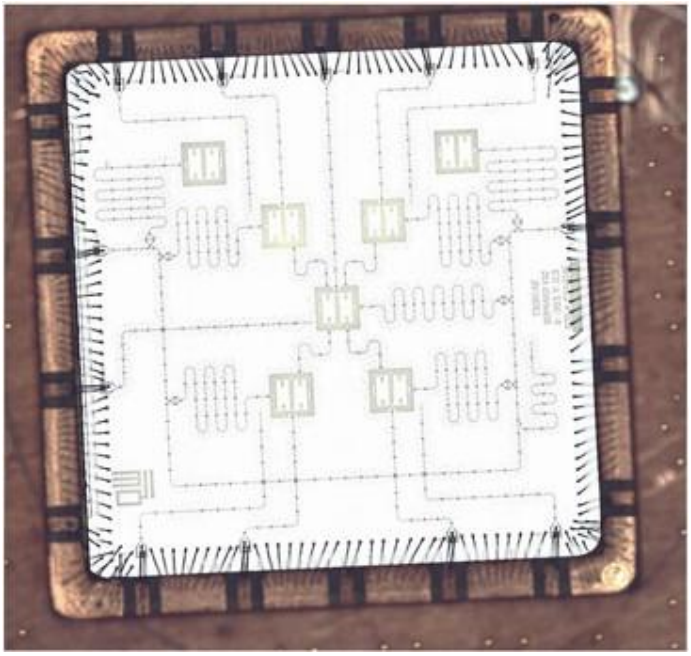


Figure 4.2: **QuantWare *Soprano* chip under the microscope.** The die hosts up to five superconducting transmon qubits with dedicated drive and flux lines and individual readout resonators.

Additionally, each qubit includes a DC/low-frequency flux line that sets the operating point Φ_{bias} and allows nanosecond-scale flux pulses where required. Bias sources are filtered and thermalized at multiple stages (RC and copper-powder filters), and the room-temperature control (Section 4.4) provides both static biasing and shaped flux pulses.

The readout resonator is probed through a separate line routed via two or more cryogenic circulators/isolators at MXC and 4 K. The first active element is a traveling-wave parametric amplifier (TWPA) [71, 72] pumped through a dedicated line; a 4 K HEMT amplifier [62] raises the signal well above the noise floor of room-temperature electronics. The overall system noise is dominated by the effective input-referred noise of the first stage, which motivates careful TWPA biasing and pump-tone conditioning.

4.4 Room-temperature instrumentation and signal flow

The room-temperature control stack is built around the Qblox *Cluster* [3] platform, which integrates RF generation/digitization and baseband AWG channels under a common timing and reference architecture. The Cluster's modules execute compiled sequencer programs (Qblox *Q1* sequence processors and the Q1ASM instruction set) that schedule pulses, phase updates, digital markers, and acquisitions with deterministic timing; this tight hardware–software loop is central to our measurement-informed calibration workflow.

[3] https://docs.qblox.com/en/main/cluster/cluster.html

4.4.1 Qubit drive path: QCM-RF

Single-qubit gates are generated by Qblox *QCM-RF* [4] modules, which synthesize microwave control pulses directly at RF without external mixers or LOs. The QCM-RF specifies two RF output channels in the 2 GHz to 18.5 GHz band and provides digital markers per channel; its front-end is driven by on-board Q1 sequence processors, enabling phase-coherent pulses, virtual-Z updates, and envelope shaping with sub-cycle granularity. In our rack, the QCM-RF outputs are routed to the cryostat drive lines via appropriate attenuation and filtering, so that the compiled envelopes $\Omega(t)$ appear at the chip as control fields that implement the rotations analyzed in Chapter 3.

4.4.2 Readout path: QRM-RF

Dispersive readout and heterodyne acquisition are performed by Qblox *QRM-RF* [5] modules. The QRM-RF integrates up- and down-conversion stages and operates directly in the 2 GHz to 18.5 GHz band, with on-board DSP for weighted integration, thresholding, and averaging of I/Q samples. This architecture enables multiplexed resonator readout and reduces host-side data rates: the module computes integrated (I, Q) outcomes per shot using user-defined weights and returns averages/bins to the control server over Ethernet. In our flow, the readout pulse and weights are compiled into the QRM-RF sequencer; upon trigger, the module generates the probe, digitizes the returned signal, applies the integration weights [cf. Eq. (4.3)], and reports the complex acquisition $S_{21} = I + iQ$ used in fits and calibration loops.

4.4.3 Flux-control path: QCM (baseband)

Static flux bias and flux pulses are delivered by Qblox *QCM* [6] modules operating in the baseband regime up to 400 MHz. Each module offers multiple analog outputs and digital markers with high channel density, again controlled by Q1 sequence processors. A programmable DC offset provides the static Φ_{bias}, while shaped baseband pulses superimposed on the bias allow controlled excursions when required. The baseband outputs pass through bias-tees and cryogenic filters before reaching the on-chip flux coils, and their amplitude–current transfer function is calibrated against spectroscopy and Ramsey data as part of the flux-mapping routine.

4.4.4 Timing, references, and triggers

All Cluster modules share a phase-coherent reference and accept deterministic cross-module triggers so that control and readout sequences are aligned to within the instrument latency. A single experiment thus corresponds to a synchronized set of Q1 sequencers: the QCM-RF emits a drive waveform, the QCM applies (optional) flux pulses, and the QRM-RF emits a readout probe and digitizes the returning field under common start and marker signals. Sequencer-level phase accumulation is tracked in software, enabling virtual-Z corrections and dynamic frame updates.

4.4.5 LO and IF planning in the Qblox signal chain

In heterodyne RF systems, a *local oscillator* (LO) at frequency f_{LO} is mixed with a baseband or *intermediate–frequency* (IF) signal f_{IF} to translate spectra [13, 42]. Ideal linear mixing produces sum and difference components at $f_{\text{RF}} = f_{\text{LO}} \pm f_{\text{IF}}$; IQ (in–phase/quadrature) mixers select a single sideband so that the RF carrier is precisely f_{RF} while the unwanted image and LO leakage are suppressed [28]. These are the same

[4] https://docs.qblox.com/en/main/cluster/qcm_rf.html
[5] https://docs.qblox.com/en/main/cluster/qrm_rf.html
[6] https://docs.qblox.com/en/main/cluster/qcm.html

Table 4.1: Role of LO and IF in the Qblox modules.

Module	Where is the LO?	What is the IF?	Conversion path
QCM	*External* mixer LO near f_{RF}	AWG/NCO frequency (0–400 MHz)	Upconvert: $f_{LO} \pm f_{IF}$ via external IQ mixer.
QCM-RF	*Internal* LO per RF output	Digital NCO (programmable IF)	On-board DUC ⇒ IQ mixer ⇒ RF at 2 GHz to 18.5 GHz.
QRM-RF	*Internal* LO per RF I/O	Digital NCO (TX/RX IF)	TX: DUC ⇒ upconvert; RX: downconvert ⇒ DDC ⇒ I/Q.

superheterodyne ideas used in classical receivers, where signals are down–converted from f_{RF} to a fixed IF for filtering and demodulation.

The Qblox *Cluster* family implements these principles slightly differently across its three module types:

- **QCM.** The QCM outputs shaped analog *baseband* waveforms (0–400 MHz). Reaching microwave control frequencies requires an *external* IQ upconverter driven by an external LO near the desired carrier. In this architecture, the IF is the programmed QCM/NCO frequency (typically 50 MHz to 250 MHz to avoid DC), and the RF is $f_{RF} = f_{LO} \pm f_{IF}$.
- **QCM-RF.** The QCM-RF integrates the *IQ mixer and LO* per RF output. The user programs the IF digitally via an on-board numerically controlled oscillator (NCO); the module mixes this IF with its *internal LO* to synthesize the RF tone directly in the 2 GHz to 18.5 GHz band, again yielding $f_{RF} = f_{LO} \pm f_{IF}$. In addition, the module provides fine frequency/phase resolution for both LO and IF and supports automated mixer calibration to suppress LO leakage and the unwanted sideband.
- **QRM-RF.** On transmit, the QRM-RF works like the QCM-RF: a shaped digital IF is upconverted with the *internal LO* to the RF probe band for dispersive readout. On receive, the analog front end *downconverts* the incoming resonator response from RF to an IF with the same LO, and on-board DSP performs digital down-conversion (DDC) and weighted integration to return I/Q samples.

Keeping f_{LO} near the target carrier and using a modest f_{IF} offers three practical benefits: (i) the LO can be shared or held fixed across many shots/sequences for phase-stable operation while f_{IF} is stepped digitally with sub-Hz resolution; (ii) the IF can be placed away from DC to avoid flicker/offset impairments in DACs/ADCs; and (iii) image/LO-leakage calibration (sideband suppression) becomes well-posed and repeatable. In Qblox's RF modules, the IF is purely digital (NCO-based) and the LO is generated on-module, so frequency updates and mixer calibrations can be issued deterministically by the sequencers without external microwave sources.

For both control and readout we program f_{LO} close to the qubit or resonator frequency and choose $f_{IF} \sim 50–250$ MHz to avoid DC while staying well inside the analog baseband; the final carrier is then $f_{RF} = f_{LO} \pm f_{IF}$. With QCM-RF/QRM-RF, f_{IF} is changed by the NCO, whereas with a baseband QCM the same change is achieved by retuning the AWG frequency feeding the external IQ mixer. Mixer-calibration routines on the RF modules (automatic or manual) are used to minimize LO leakage and the unwanted sideband before fine experiments.

4.4.6 Digital–analog–digital flow (end-to-end)

An experiment description in the Python DSL specifies a sequence of pulses and measurements parameterized by loop variables (frequency, amplitude, flux bias, delay). The compiler maps this description to per-module Q1ASM programs, uploads them to the QCM-RF/QCM/QRM-RF, and arms the Cluster. Upon a global trigger, (i) the QCM-RF generates the microwave control envelope at the programmed carrier, (ii) the QCM outputs the static bias plus any shaped flux pulse, (iii) the QRM-RF emits the readout probe and acquires

the response, applying on-board demodulation with user-defined integration weights,

$$I = \int_0^{T_{int}} V(t) w_I(t)\, dt, \qquad Q = \int_0^{T_{int}} V(t) w_Q(t)\, dt, \tag{4.3}$$

and (iv) the resulting (I, Q) arrays (and metadata) are streamed to the control server and archived. This closed loop underlies the measurement-informed calibration routines of the next chapter.

4.5 Calibrated signal levels and line models

Quantitative control requires a mapping from programmed amplitudes to on-chip fields and from digitizer codes to physical quadratures. For the drive line we use Eq. (4.1) to propagate generator power to the MXC plane, and Eq. (4.2) to relate the observed Rabi rate Ω_R to the field amplitude, which implicitly calibrates the net line transfer function (including cable loss, attenuators, and cryogenic components). For the readout line we calibrate the absolute phase and complex gain by measuring the resonator transmission S_{21} across frequency at low power and referencing the off-resonant response as unity. For flux we combine DC transport measurements (room-temperature conversion from DAC codes to current) with qubit spectroscopy, using the measured $f_{01}(\Phi)$ chevron to infer the on-chip mutual inductance. These calibrations feed directly into the frequency planning, bias selection, and pulse-shape optimization strategies of Chapter 3.

4.6 Pulse families and on-hardware processing

Digital envelope generation allows us to realize shaped pulses such as Gaussian, cosine flat-top, and DRAG-corrected waveforms,

$$\Omega(t) = \Omega_0\, e^{-t^2/2\sigma^2}, \qquad \Omega_{DRAG}(t) = \Omega(t) - i\beta\, \dot{\Omega}(t), \tag{4.4}$$

with derivative-quadrature scaling β tuned in the experiment. The QCM-RF/QRM-RF execute these envelopes with sub-cycle phase control, while the QRM-RF's on-board DSP performs the weighted demodulation and binning used throughout this work. These module-level capabilities—direct-RF synthesis in 2 GHz to 18.5 GHz and baseband generation up to 400 MHz, sequencer-driven timing, and on-module demodulation—are the enabling technologies for the measurement-informed calibration and for the comparative evaluation of DRAG, FAST-DRAG, and HD-DRAG reported later.

Chapter 5

Experiment and Results

5.1 Overview

This chapter documents the end-to-end characterization and calibration campaign used to commission the platform introduced in chapter 4. Each experiment is presented with:

i) its goal and measured parameters,

ii) the pulse-level program written in qililab's DSL,

iii) an analysis pipeline that loads .h5 results, fits models, and produces publication-quality figures.

The workflow is intentionally split into a *one-time* device characterization block (run once per cooldown) and a *per-variant* calibration block (repeated for each control-pulse family). One-time experiments determine static or slowly varying device parameters and populate a small cache; per-variant experiments then reuse that cache while tuning the control pulses and benchmarking gates.

One-time block. 1-D and 2-D spectroscopy that identify the working point:

1. **Resonator spectroscopy** $\Rightarrow f_r$, linewidth κ.
2. **Resonator vs. flux** $\Rightarrow$ sweet-spot flux and cross-talk cues.
3. **Two-tone spectroscopy** $\Rightarrow f_{01}$.
4. **Two-tone vs. flux** $\Rightarrow$ tunability and an initial $|\alpha|$ (anharmonicity) estimate.

Per-variant block. For each analytical pulse family (below), we:

1. **Synthesize** (Ω_I, Ω_Q) for target angle ϑ and duration T.
2. **Calibrate** π by Rabi.
3. **Acquire SSRO** clouds and thresholds.
4. **Benchmark** with RB.

Table 5.1: Control-pulse variants and their primary knobs.

Variant	In-phase $I(t)$	Quadrature $Q(t)$ / knobs
Gaussian-only	$I = g$	$Q = 0$
DRAG	$I = g$	$Q = -(\beta/\alpha)\,\dot{I}$., tune $\beta \simeq 1$ (leakage) or $\beta \simeq 0.5$ (phase).
HD-DRAG	$I = g + \beta_2 \ddot{g}$	$Q = -(\beta/\alpha)\,\dot{I}$; $\beta_2 \approx \alpha^{-2}$ for a double zero.
FAST-DRAG	$I(t) = \sum_n c_n[1 - \cos(2\pi n t/T)]$	$Q = -(\beta/\alpha)\,\dot{I}$; $\{c_n\}$ from band-suppression KKT.

5.2 Analytical synthesis of control pulses

We formalize and implement four pulse–shaping variants for resonant single–qubit rotations in the rotating–wave approximation (RWA): Gaussian-only (no quadrature), DRAG (first derivative), HD-DRAG (higher derivative), and FAST-DRAG. The driven weakly–anharmonic oscillator (transmon) under a resonant drive of duration T is modeled, in the frame rotating at the carrier, by

$$H_R(t) = \frac{\hbar}{2} \sum_{j\geq 1} \Omega_I(t)\, \sigma_x^{(j-1,j)} + \Omega_Q(t)\, \sigma_y^{(j-1,j)} \Big] + \hbar \sum_{j\geq 0} \delta_j\, |j\rangle\langle j|, \tag{5.1}$$

with $\delta_1 = 0$ and $\delta_2 = \alpha$ the transmon anharmonicity (in angular units). For on–resonance single–qubit rotations, the area rule gives

$$\vartheta \simeq \int_0^T \Omega_I(t)\, dt, \tag{5.2}$$

within the RWA. Every envelope pair (Ω_I, Ω_Q) is scaled so that (5.2) yields the target angle ϑ, and the same scalar is applied to Ω_Q. A virtual-Z phase ϕ is a frame choice implemented by

$$\begin{pmatrix} \Omega_I' \\ \Omega_Q' \end{pmatrix} = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} \Omega_I \\ \Omega_Q \end{pmatrix}. \tag{5.3}$$

For frequency-domain diagnostics, we use the complex envelope

$$\Omega_{IQ}(t) = \Omega_I(t) - i\,\Omega_Q(t), \qquad \hat{\Omega}_{IQ}(f) = \int_0^T \Omega_{IQ}(t)\, e^{-i2\pi f t}\, dt, \tag{5.4}$$

and plot the amplitude spectrum $|\hat{\Omega}_{IQ}(f)|$ on the baseband axis $f \in \mathbb{R}$, where the leakage transition $|e\rangle \leftrightarrow |f\rangle$ appears at $f = \alpha/(2\pi)$. Implementations of all helper functions (envelope generators, normalization, spectrum utilities, and plotting) are listed in Appendix A.

5.2.1 Pulse families

Gaussian-only. The in-phase waveform is a truncated, centered Gaussian $g(t)$ on $[0, T]$ with $\Omega_Q(t) = 0$. This baseline exposes the spectral crowding around $\pm\alpha/(2\pi)$ without leakage mitigation.

DRAG. Derivative-Removal-by-Adiabatic-Gate augments the drive with an off-quadrature component proportional to the time derivative of the in-phase envelope:

$$\Omega_Q(t) = -\frac{\beta}{\alpha}\dot{\Omega}_I(t), \tag{5.5}$$

where β is the DRAG coefficient and α the anharmonicity. Choosing $\beta \approx 1$ creates a first-order spectral zero of $\hat{\Omega}_{IQ}(f)$ near $f = \alpha/(2\pi)$ and hence suppresses leakage. Alternatively, $\beta \approx 1/2$ is shown to cancel AC-Stark phase errors. [66]

HD-DRAG. Following the higher-derivative extension introduced in Ref. [11] we decouple the in-phase and quadrature design by

$$\Omega_I(t) = g(t) + \beta_2\, g''(t), \qquad \Omega_Q(t) = -\frac{\beta}{\alpha}\frac{d}{dt}\Omega_I(t) = -\frac{\beta}{\alpha}\, g'(t) + \beta_2\, g(t)\,. \tag{5.6}$$

In Fourier space this factors as

$$\hat{\Omega}_{IQ}(f) = \left[1 - \beta_2(2\pi f)^2\right]\left[1 - 2\pi\beta f/\alpha\right]\hat{g}(f), \tag{5.7}$$

so that β (DRAG) and β_2 (higher-derivative shaping) are decoupled. The leakage-optimized choice $\beta = 1$, $\beta_2 = \alpha^{-2}$ produces a second-order zero at $f = \alpha/(2\pi)$ while leaving the I/Q calibration simple. For this setting, the frequency-domain form reads

$$\hat{\Omega}_{IQ}(f)\Big|_{\beta=1,\,\beta_2=\alpha^{-2}} = \left[1 - \frac{(2\pi f)^2}{\alpha^2}\right]\left[1 - \frac{2\pi f}{\alpha}\right]\hat{g}(f) = (1+x)(1-x)^2\,\hat{g}(f), \qquad x \equiv \frac{2\pi f}{\alpha},$$

which explicitly reveals the double zero at $x = 1$ (i.e., $f = \alpha/2\pi$). The additional prefactor $(1 + x)$ arises from the even-derivative contribution and does not alter the order or position of the zero compared with the expression $(1 - x)^2$ derived in Chapter 3.

FAST-DRAG. The in-phase envelope is synthesized as a short cosine series

$$\Omega_I(t) = A\sum_{n=1}^{N} c_n\left[1 - \cos(2\pi n t/T)\right], \tag{5.8}$$

with coefficients c_n chosen to *analytically* minimize the spectral energy of $\hat{\Omega}_I$ over user-specified stop-bands $\{[f_{\ell,j}, f_{h,j}]\}$ (e.g. a band centered at $|\alpha|/2\pi$ and a high-frequency cut-off) under the area constraint $\sum_n c_n = \vartheta$. In this work we use $N = 4$ basis terms, which provides sufficient spectral flexibility to shape a deep notch around the leakage transition while keeping the time-domain envelope smooth and experimentally well-conditioned. This convex quadratic problem with a single linear constraint reduces to the KKT system

$$\begin{bmatrix} A + A^{\top} & -\mathbf{b} \\ \mathbf{b}^{\top} & 0 \end{bmatrix}\begin{bmatrix} \mathbf{c} \\ \mu \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \vartheta/T \end{bmatrix}, \qquad A_{nm} = \sum_j w_j \int_{f_{\ell,j}}^{f_{h,j}} g_n(f)\, g_m^*(f)\, df, \quad \mathbf{b} = (1, \ldots, 1)^{\top}. \tag{5.9}$$

We then set $\Omega_Q(t) = -(\beta/\alpha)\,\dot{\Omega}_I(t)$ as in (5.5). FAST-DRAG shapes a *wide* notch around $\pm\alpha/(2\pi)$ and simultaneously regularizes bandwidth by suppressing high-frequency content, enabling short, low-leakage gates.

5.2.2 Spectral analysis and figures

Figure 5.1 shows the time–domain envelopes $[\Omega_I(t), \Omega_Q(t)]$ for the four variants at identical duration T and target angle ϑ; the Gaussian-only panel uses $\Omega_Q \equiv 0$. Figure 5.2 overlays the corresponding amplitude spectra $|\hat{\Omega}_{IQ}(f)|$ over the bi-sided baseband axis (negative and positive frequencies). A dashed vertical line marks $f = \alpha/(2\pi)$. Consistent with (5.5)–(5.7), DRAG produces a single spectral zero at the anharmonicity,

HD-DRAG raises the order of that zero by one, and FAST-DRAG yields a broad engineered stop-band whose width is set by $[f_{\ell,1}, f_{h,1}]$; the latter also reduces spurious peaks at larger $|f|$ by design. These frequency-domain features mirror the expected leakage trends for fast gates in weakly anharmonic systems.

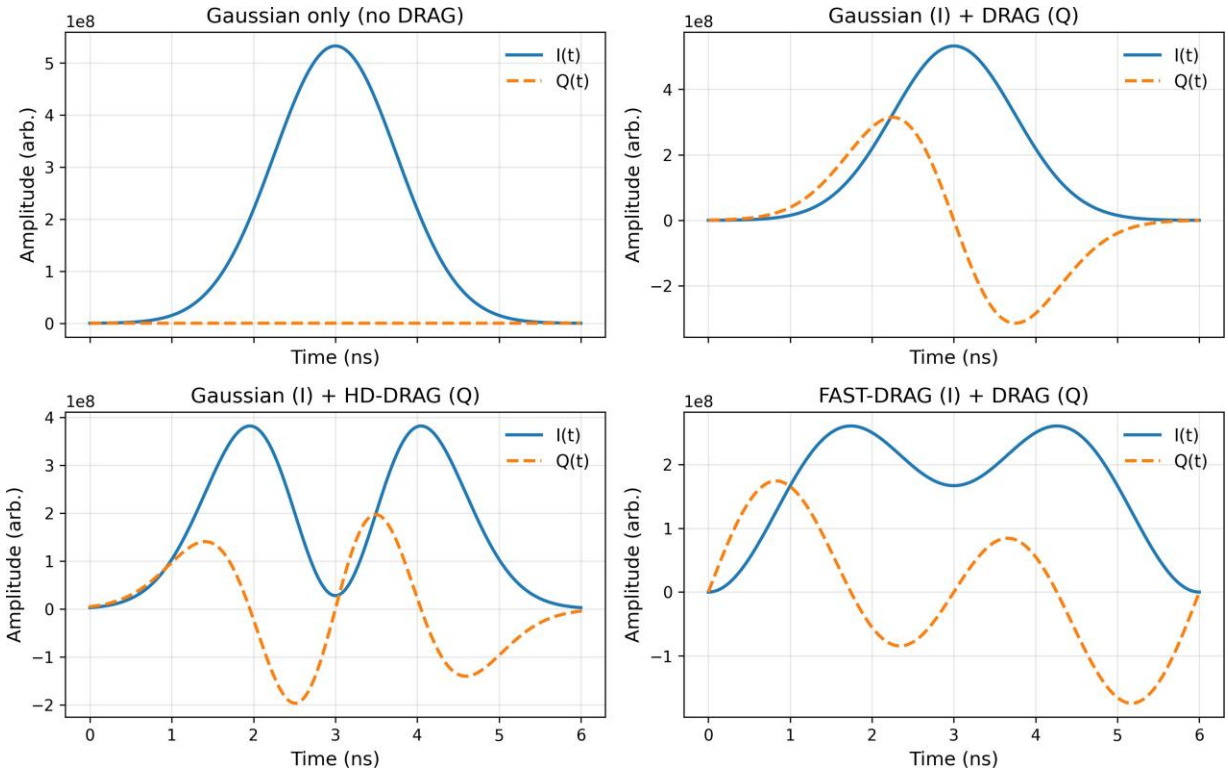


Figure 5.1: **Time-domain envelopes for four analytical variants.** Each panel shows the in-phase $\Omega_I(t)$ (solid) and quadrature $\Omega_Q(t)$ (dashed) at identical gate length T and target area $\int_0^T \Omega_I(t)\,dt = \vartheta$. From left to right: Gaussian-only ($\Omega_Q = 0$); DRAG with $\Omega_Q = -(\beta/\alpha)\dot{\Omega}_I$; HD-DRAG with $\Omega_I = g + \beta_2 g^{\cdot\cdot}$ and $\Omega_Q = -(\beta/\alpha)\dot{\Omega}_I$; FAST-DRAG (cosine series for Ω_I solving (5.9), plus DRAG quadrature).

5.2.3 Implementation details

All four variants are implemented with NumPy arrays on a uniform grid $t_k = k\,\Delta t$ ($k = 0, \ldots, N-1$); time derivatives are computed with centered finite differences of order two. Envelopes are normalized so that $\int_0^T \Omega^I(t)\,dt = \vartheta$; the same scale is applied to Ω^Q. The API phase parameter applies the rotation (5.3) so that arbitrary X_ϕ/Y_ϕ axes are synthesized without re-sampling. For HD-DRAG we provide a leakage-optimized default $(\beta, \beta_2) = (1, \alpha^{-2})$ as well as a phase-optimized variant ($\beta \approx 0.5$) with β_2 tunable; the factorization (5.7) decouples these degrees of freedom and simplifies calibration. FAST-DRAG exposes the number of basis terms N, stop-bands $[f_{\ell,j}, f_{h,j}]$ with weights w_j, and an optional high-frequency band $[f_c, f_{Nyq}]$ to regularize bandwidth and peak amplitude. Detailed Python implementations of the envelopes and spectrum utilities are provided in Appendix A.

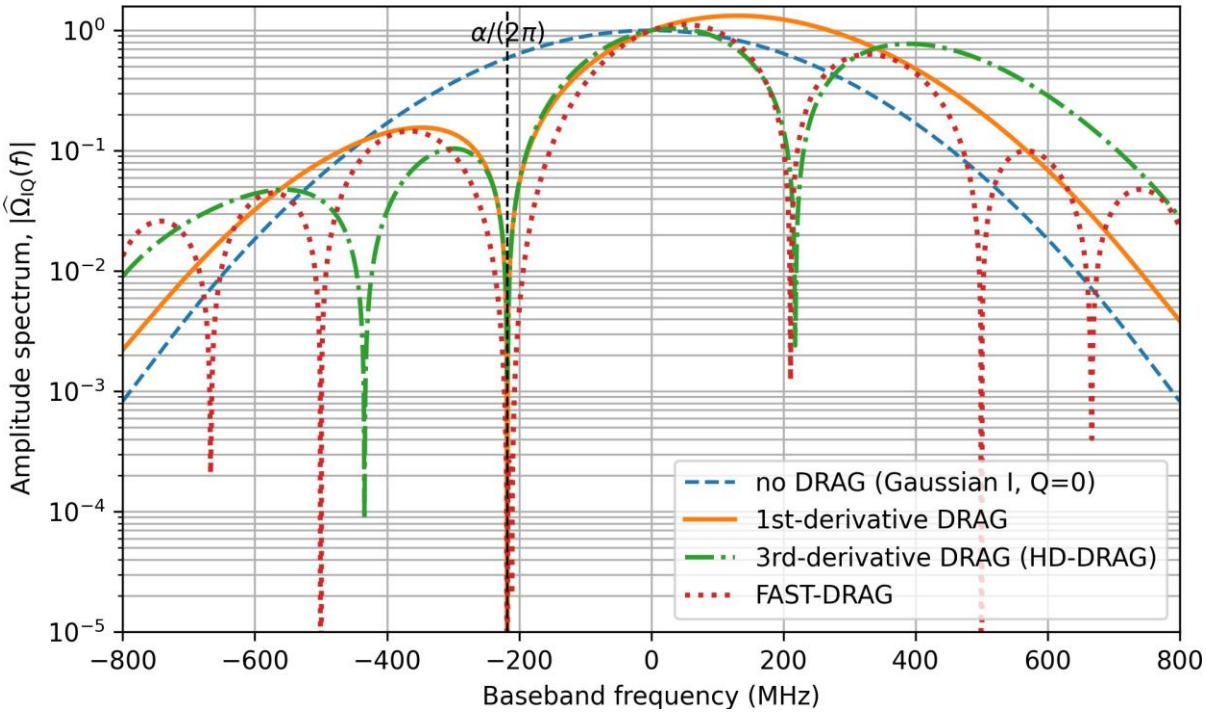


Figure 5.2: **Amplitude spectrum of the complex envelope.** Amplitude spectra $|\widehat{\Omega}_{IQ}(f)|$ for the pulses in Fig. 5.1, displayed over negative and positive baseband frequencies (MHz). The dashed vertical marker indicates $f = \alpha/(2\pi)$. Gaussian-only concentrates substantial spectral weight near $\pm\ \alpha/(2\pi)$; DRAG creates a first-order zero there; HD-DRAG strengthens the zero to second order; FAST-DRAG constructs a broad stop-band around $\alpha/(2\pi)$ and suppresses high-frequency content set by the design cut-off. Spectral notches at $\pm\alpha/(2\pi)$ correlate with reduced $|e\rangle \rightarrow |f\rangle$ leakage in short gates.

5.3 Qililab: Quantum Control Library

All measurements are orchestrated with `qililab`[1] library owned by Qilimanjaro Quantum Tech.[2] The author is the main maintainer and technical lead of `qililab`. The library, among other features, offers:

- A **platform abstraction** that builds the instrument graph from a runcard and exposes a single entry point to execute experiments deterministically on hardware.
- A **domain-specific language (DSL)** to express pulse programs independent of the underlying AWGs and digitizers. A QProgram describes time-ordered actions (set frequency, play, measure, wait, loops and averages). An Experiment composes one or more QPrograms, optionally sweeping hardware parameters (e.g. attenuations, DC biases) across outer loops.

[1] https://github.com/qilimanjaro-tech/qililab
[2] https://qilimanjaro.tech

- A **results layer**. Executing an Experiment returns a path to an HDF5 file with raw data and metadata. The helper class ExperimentResults streamlines loading of multidimensional results.

Execution & pattern. Every experiment in this chapter follows the same pattern:

1. Build a QProgram (inner control/readout loop).
2. Wrap it in an Experiment (outer sweeps over hardware conditions).
3. Run on the platform and load results:

```
import qililab as ql

# Define constants
EXPERIMENT_NAME = ...
QUBIT_IDX = 0
NSHOTS = 10_000
RELAXATION_DURATION = 600_000
DRIVE_BUS = f"drive_q{QUBIT_IDX}_bus"
READOUT_BUS = f"readout_q{QUBIT_IDX}_bus"
FLUX_BUS = f"flux_q{QUBIT_IDX}_bus"

# Build Platform
platform = ql.build_platform(runcard="path/to/runcard.yml")

# ... define experiment ...

# Execute Experiment
with platform.session():
    results_path = platform.execute_experiment(experiment)

# Retrieve data and metadata from .h5 file
with ExperimentResults(results_path) as results:
    data, dims = results.get()        # data shape: (Loop1, Loop2, ... LoopN, 2) with (..., 0|1) = I|Q
```

Throughout, we compute the complex transmission

$$S_{21} = I + iQ, \qquad |S_{21}| = \sqrt{I^2 + Q^2}, \qquad |S_{21}|_{\mathrm{dB}} = 20 \log_{10} |S_{21}|,$$

where we refer to $|S_{21}|$ as the *modulus (amplitude)* of S_{21} and use $|S_{21}|_{\mathrm{dB}}$ when working in decibels.

```
s21 = data[..., 0] + 1j * data[..., 1]
s21_modulus = np.abs(s21)
s21_db = 20 * np.log10(s21_modulus)
```

5.4 Resonator Spectroscopy (One-time)

Goal

The Resonator Spectroscopy experiment characterizes the readout resonator attached to qubit by sweeping a weak probe across the resonator while measuring the demodulated response on the readout line. A fixed readout local oscillator (LO) is used and the intermediate frequency (IF) on the readout bus is scanned over a narrow window centered at the current configuration:

$$f = f_{\mathrm{LO}} + \mathrm{IF}, \qquad \mathrm{IF} \in [-10\ \mathrm{MHz}, +10\ \mathrm{MHz}],$$

with uniformly spaced points (step ≈ 10 kHz). At each probe frequency the system executes a readout pulse and integrates the in-phase and quadrature components using uniform weights. The loop is averaged over N_{shots} = NSHOTS to improve SNR, and a relaxation delay is inserted between points to avoid power broadening and history effects.

The goal is to extract the resonator center frequency f_0 near the LO.

Program

```
LO = platform.get_parameter(alias=READOUT_BUS, parameter=Parameter.LO_FREQUENCY)
IF = platform.get_parameter(alias=READOUT_BUS, parameter=Parameter.IF)
IF_VALUES = np.linspace(-10e6, 10e6, 201) + IF

qp = QProgram()
frequency = qp.variable(domain=Domain.Frequency, label="Frequency")
with qp.average(NSHOTS):
    with qp.loop(variable=frequency, values=IF_VALUES):
        qp.set_frequency(bus=READOUT_BUS, frequency=frequency)
        qp.measure(bus=READOUT_BUS, waveform=readout_wf, weights=weights)
        qp.wait(bus=READOUT_BUS, duration=RELAXATION_DURATION)

experiment = Experiment(label="Resonator Spectroscopy")
experiment.execute_qprogram(qp)
```

Analysis

The spectrum $|S_{21}|$(IF) is fitted to a Lorentzian model (see Appendix B),

$$L(f; f_0, \Gamma, A, C) = C + \frac{A}{1 + 4\left(\frac{f - f_0}{\Gamma}\right)^2}, \tag{5.10}$$

where f_0 is the center (in IF units), Γ the full width at half maximum, A the amplitude (peak or dip, depending on sign), and C a background offset. The fit returns $\hat{f}_{IF}$, the fitted spectrum $L(f)$, and a coefficient of determination R^2 used as a quality metric. The resonator frequency is then reported as

$$f_{01} = f_{LO} + \hat{f}_{IF}.$$

```
fitted_if, fitvals, rsquared = lorentzian_fit(s21_modulus, IF_VALUES)

plt.figure(figsize=(10, 6))
plt.plot((IF_VALUES)*1e-6, s21_modulus, "--o", color="blue",label = 'Data')
plt.plot((IF_VALUES)*1e-6, fitvals, "--", color="red", label='Fit')
plt.title(f"Fitted IF = {fitted_if*1e-6:.3f}MHz, Resonator Frequency={(fitted_if + LO)*1e-9:.3f}GHz")

plt.suptitle(EXPERIMENT_NAME)
plt.xlabel('IF (MHz)')
plt.ylabel('Integrated Voltage (V)')
plt.legend(loc='lower right',bbox_to_anchor=(0.9, 0.15),fontsize=14)
plt.savefig(f'{data_folder}/{EXPERIMENT_NAME}.png', dpi=300, bbox_inches='tight')
plt.show()
```

Results

The measured transmission spectrum $|S_{21}|$ as a function of probe IF shows a clear Lorentzian dip, corresponding to the resonator's fundamental mode. The fit to Eq. (5.13) yields a center frequency consistent with the expected resonator frequency, with a coefficient of determination $R^2 \approx 1$, indicating an excellent fit. The extracted value of $\hat{f}_{\mathrm{IF}}$ determines the resonator frequency to be $f_{01} = f_{\mathrm{LO}} + \hat{f}_{\mathrm{IF}}$. This calibration is subsequently used as input for qubit spectroscopy and readout optimization.

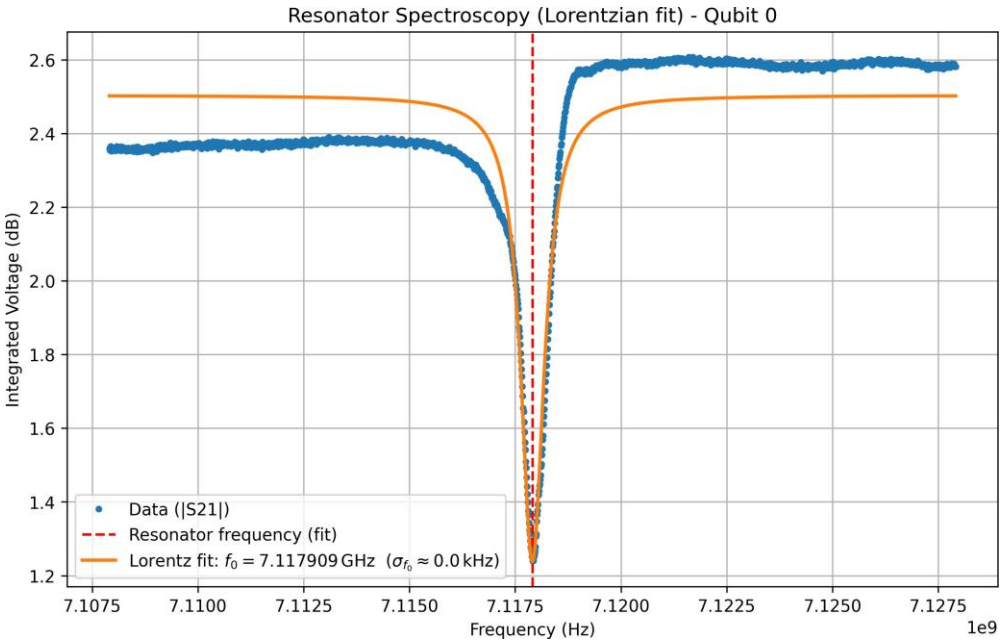


Figure 5.3: **Resonator spectroscopy.** The integrated response S_{21} (blue markers) is measured as a function of the probe intermediate frequency (IF) and fitted to a Lorentzian model (red line). The fit returns the resonator center frequency $\hat{f}_{\mathrm{IF}}$, which together with the fixed LO frequency yields the calibrated resonator frequency f_{01}.

5.5 Resonator Spectroscopy vs Qubit Flux Offsets (One-time)

Goal

This experiment maps how the readout resonator responds to a static flux bias applied to the qubit. For each value of a DC bias voltage V on the qubit flux line, we perform a short resonator spectroscopy by sweeping the readout tone around the current configuration,

$$f = f_{\mathrm{LO}} + \mathrm{IF}, \qquad \mathrm{IF} \in [-4\ \mathrm{MHz}, +4\ \mathrm{MHz}],$$

and measure the demodulated response (I, Q). The dispersive interaction makes the resonator's apparent center frequency f_r depend on the qubit frequency $f_q(V)$, which is flux-periodic. By stacking many 1D resonator traces across V, we obtain a 2D dataset $S_{21}(f; V)$ whose ridge (or valley) traces $f_r(V)$.

The aim is to (i) extract $f_r(V)$ for each bias, (ii) capture its periodic dependence with a compact model, and

(iii) identify a flux "sweet spot" $V^{\star}$ (a point of vanishing first-order sensitivity) to guide later qubit-level calibrations. The same analysis provides the period in voltage (useful for flux-line calibration) and the amplitude of the dispersive shift.

Program

```
LO = platform.get_parameter(alias=READOUT_BUS, parameter=Parameter.LO_FREQUENCY)
IF = platform.get_parameter(alias=READOUT_BUS, parameter=Parameter.IF)
OFFSET = platform.get_parameter(alias=FLUX_BUS, parameter=Parameter.OFFSET)

IF_VALUES = np.linspace(-4e6, 4e6, 51) + IF
FLUX_VALUES = np.linspace(-0.6, 0.6, 51)

qp = QProgram()
frequency = qp.variable(domain=Domain.Frequency, label="Frequency")
with qp.average(NSHOTS):
    with qp.loop(variable=frequency, values=IF_VALUES):
        qp.set_frequency(bus=READOUT_BUS, frequency=frequency)
        qp.measure(bus=READOUT_BUS, waveform=readout_wf, weights=weights)
        qp.wait(bus=READOUT_BUS, duration=RELAXATION_DURATION)

experiment = Experiment(label="Resonator Spectroscopy vs Qubit Flux Offsets")
offset = experiment.variable(domain=Domain.Voltage, label="Offset")
with experiment.loop(variable=offset, values=FLUX_VALUES):
    experiment.set_parameter(bus=FLUX_BUS, parameter=Parameter.OFFSET, value=offset)
    experiment.execute_qprogram(qp)

results_path = platform.execute_experiment(experiment)
```

Analysis

For each bias V_j we estimate the resonator center frequency by the location of the transmission dip,

$$\hat{f}_r(V_j) = \arg\min_f S_{21}^{dB}(f; V_j),$$

yielding a discrete curve $\{(V_j, \hat{f}_r(V_j))\}$ over the scanned range. (If higher precision is required, a 1D Lorentzian-dip fit around the local minimum can replace the argmin.)

Over moderate ranges the dispersive shift of the resonator follows a smooth, periodic pattern versus flux. We model the extracted centers phenomenologically by a single-harmonic fit

$$\hat{f}_r(V) \approx a \sin(bV + c) + d, \tag{5.11}$$

where a is the half peak-to-peak shift, b encodes the period in voltage $T_V = 2\pi/b$, c is a phase, and d the mean frequency. This captures the periodicity imposed by the SQUID loop while avoiding over-parameterization.

A flux sweet spot is a bias where first-order sensitivity to flux vanishes. With the model in (5.11),

$$\frac{d\hat{f}_r}{dV}(V) = a\,b\cos(bV + c),$$

so any solution of $\cos(bV^\star + c) = 0$ within the scan window is a candidate sweet spot:

$$V^\star = \frac{\frac{\pi}{2} - c + k\pi}{b}, \qquad k \in \mathbb{Z}.$$

We report the $V^\star$ that lies in the measured interval and mark it on the plot.

```
import lmfit
from scipy.optimize import root

res_freq = IF_VALUES[np.argmin(dB(s),axis=0)]

x_data = OFFSETS
y_data = res_freq

tt = np.array(x_data)
yy = np.array(y_data)
ff = np.fft.fftfreq(len(tt), (tt[1] - tt[0])) * 2 * np.pi  # assume uniform spacing
Fyy = abs(np.fft.fft(yy))
guess_freq = abs(ff[np.argmax(Fyy[1:]) + 1])  # excluding the zero frequency "peak", which is related to offse
guess_amp = np.std(yy) * 2.0**0.5
guess_offset = np.mean(yy)

mod   =   lmfit.Model(sinus)

mod.set_param_hint("a", value=guess_amp)
mod.set_param_hint("b", value=guess_freq / 1.5)
mod.set_param_hint("c", value=0)
mod.set_param_hint("d", value=guess_offset)

params = mod.make_params()
fit = mod.fit(data=y_data, params=params, x=x_data, method="basinhoping", max_nfev=100000)

fitted_a = fit.params["a"].value
fitted_freq = fit.params["b"].value
fitted_c = fit.params["c"].value
fitted_t2 = fit.params["d"].value

popt = [fitted_a, fitted_freq, fitted_c, fitted_t2]
fit_values = sinus(np.linspace(x_data[0], x_data[-1], 101), *popt)
fit_cosine = cosine(np.linspace(x_data[0], x_data[-1], 101), *popt)

def func(x):
    return cosine(x, *popt) - popt[3]

# Make an initial guess for the root
x0 = [0]

# Find the root
sol = root(func, x0)

plt.figure(figsize=(10, 8))
```

```
plt.plot(OFFSETS,res_freq , "o", color="red")
plt.plot(np.linspace(x_data[0], x_data[-1], 101),fit_values , "r-")
plt.pcolor(OFFSETS, IF_VALUES, dB(s))
plt.axvline(x=sol.x[0], label=f"Sweet Spot\n {sol.x[0]:.4f}V", color="b")
plt.legend(bbox_to_anchor=(1.16, 1), loc="upper left")
plt.colorbar()
plt.ylabel("Tone Frequency (Hz)")
plt.xlabel("Biasing Voltage (V)")
plt.title(EXPERIMENT_NAME)
plt.savefig(f'./{EXPERIMENT_NAME}', dpi=300, bbox_inches='tight')
```

Results

The two–dimensional map $S_{21}(f; V)$ exhibits a single dispersive ridge whose minimum traces the resonator center $\hat{f}_r(V)$. The point–wise minima (red markers) are well described by the single–harmonic model in (5.11), and the fitted curve (solid red) captures both the periodicity and the amplitude of the flux-induced shift. From the fit we extract the voltage period $T_V = 2\pi/b$ and the half peak-to-peak excursion a, which quantify, respectively, the flux-line calibration in volts per period and the magnitude of the resonator's dispersive response. The sweet-spot condition $d\hat{f}_r/dV = 0$ selects a bias V^* inside the scan range; in this dataset the corresponding operating point is $V^* \simeq 0.0244$ V (vertical blue line). This bias is subsequently used as a robust parking point for qubit-level calibrations.

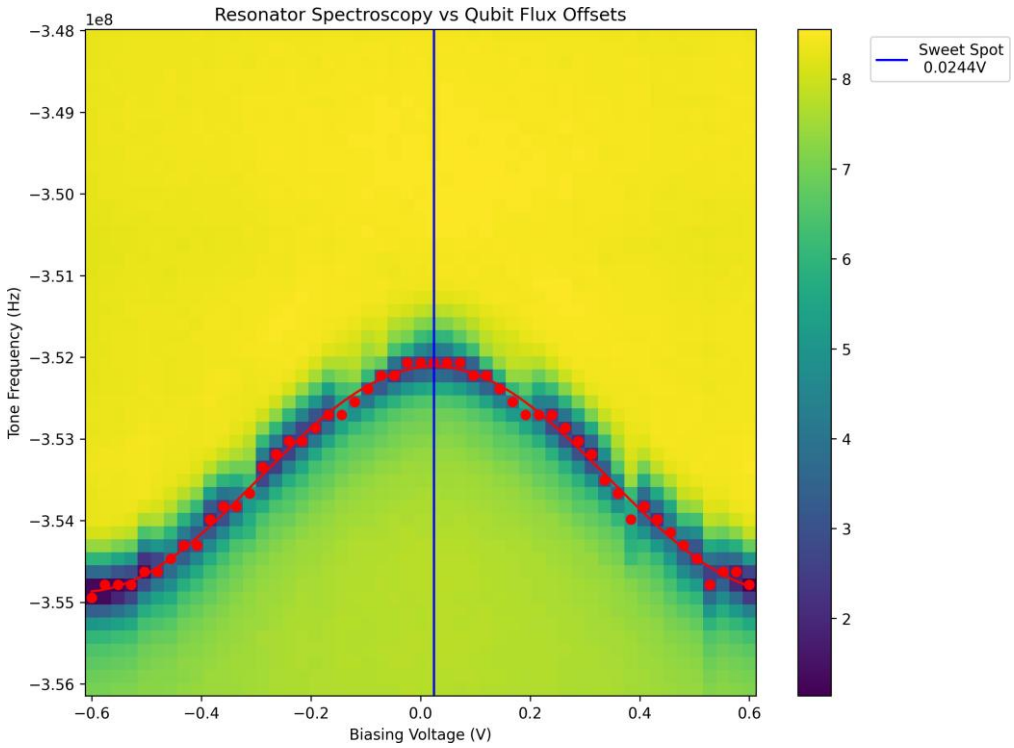


Figure 5.4: **Resonator spectroscopy versus static qubit-flux bias.** Shaded background: colormap of the measured transmission $S_{21}(f\,;V)$ (dB). Red dots: argmin estimates $\hat{f}_r(V_j)$ of the resonator center for each bias V_j. Red curve: sinusoidal fit $a\sin(bV+c)+d$ used to extract the voltage period $T_V = 2\pi/b$ and the amplitude a. Blue vertical line: flux sweet spot V^* (here $V^* \approx 0.0244$ V) obtained from the zero-slope condition $d\hat{f}_r/dV = 0$.

5.6 Two-tone Spectroscopy (f_{01}) (One-time)

Goal

The objective of the Two-Tone Spectroscopy experiment is to determine the qubit transition frequency f_{01} by sweeping a coherent drive near the qubit resonance while simultaneously performing a dispersive readout. Operationally, a fixed local oscillator (LO) on the qubit drive line is used and the intermediate frequency (IF) of a short pulse is swept over a broad window,

$$\text{IF} \in \{-400\ \text{MHz}, -399\ \text{MHz}, \ldots, 399\ \text{MHz}\},$$

while a readout pulse measures the integrated response on the readout resonator. The experiment returns a single 1D spectrum capturing the qubit's resonant response versus IF, from which we extract:

- the best-fit IF center $\hat{f}_{\text{IF}}$ of the qubit resonance,
- the absolute qubit frequency $f_{01} = f_{\text{LO}} + \hat{f}_{\text{IF}}$,
- line-shape parameters (width, contrast) and a goodness-of-fit metric for quality control.

Program

```
LO = platform.get_parameter(alias=READOUT_BUS, parameter=Parameter.LO_FREQUENCY)
IF = platform.get_parameter(alias=READOUT_BUS, parameter=Parameter.IF)

IF_VALUES = np.arange(-50e6, 50e6, 2e6) + IF

qp = QProgram()
frequency = qp.variable(domain=Domain.Frequency, label="Frequency")
with qp.average(NSHOTS):
    with qp.loop(variable=frequency, values=IF_VALUES):
        qp.set_frequency(bus=READOUT_BUS, frequency=frequency)
        qp.play(bus=DRIVE_BUS, waveform=drive_wf)
        qp.sync()
        qp.measure(bus=READOUT_BUS, waveform=readout_wf, weights=weights)
        qp.wait(bus=READOUT_BUS, duration=RELAXATION_DURATION)

experiment = Experiment(label="Resonator Spectroscopy")
experiment.execute_qprogram(qp)

results_path = platform.execute_experiment(experiment)
```

Analysis

The spectrum $|S_{21}|$(IF) is fitted to a Lorentzian model,

$$L(f; f_0, \Gamma, A, C) = C + \frac{A}{1 + 4\left(\frac{f - f_0}{\Gamma}\right)^2}, \tag{5.12}$$

where f_0 is the center (in IF units), Γ the full width at half maximum, A the amplitude (peak or dip, depending on sign), and C a background offset. The fit returns $\hat{f}_{\mathrm{IF}}$, the fitted spectrum $L(f)$, and a coefficient of determination R^2 used as a quality metric. The qubit frequency is then reported as

$$f_{01} = f_{\mathrm{LO}} + \hat{f}_{\mathrm{IF}}.$$

```
fitted_if, fitvals, rsquared = lorentzian_fit(s21_modulus, IF_VALUES)

plt.figure(figsize=(10, 6))
plt.plot((IF_VALUES)*1e-6, s21_modulus, "--o", color="blue",label = 'Data')
plt.plot((IF_VALUES)*1e-6, fitvals, "--", color="red", label='Fit')
plt.title(f"Fitted IF = {fitted_if*1e-6:.3f}MHz, Qubit Frequency={(fitted_if + LO)*1e-9:.3f}GHz")

plt.suptitle(EXPERIMENT_NAME)
plt.xlabel('IF (MHz)')
plt.ylabel('Integrated Voltage (V)')
plt.legend(loc='lower right',bbox_to_anchor=(0.9, 0.15),fontsize=14)
plt.savefig(f'{data_folder}/{EXPERIMENT_NAME}.png', dpi=300, bbox_inches='tight')
plt.show()
```

Results

The one–dimensional spectrum exhibits a single dominant Lorentzian feature corresponding to the $|0\rangle \rightarrow |1\rangle$ transition. The fit of Eq. (5.13) returns a center $\hat{f}_{\mathrm{IF}}$ located on the negative–IF side of the sweep (panel

header), from which we infer the absolute qubit frequency $f_{01} = f_{LO} + \hat{f}_{IF}$. The fitted width Γ captures both intrinsic dephasing and power broadening at the chosen drive amplitude, while the offset C accounts for the slowly varying measurement baseline. A small residual feature at large positive IF is off–resonant and does not affect the estimation of f_{01}. The fit quality (large R^2) confirms that a single-pole Lorentzian adequately describes the response and that the reported f_{01} can be used to seed subsequent fine calibrations.

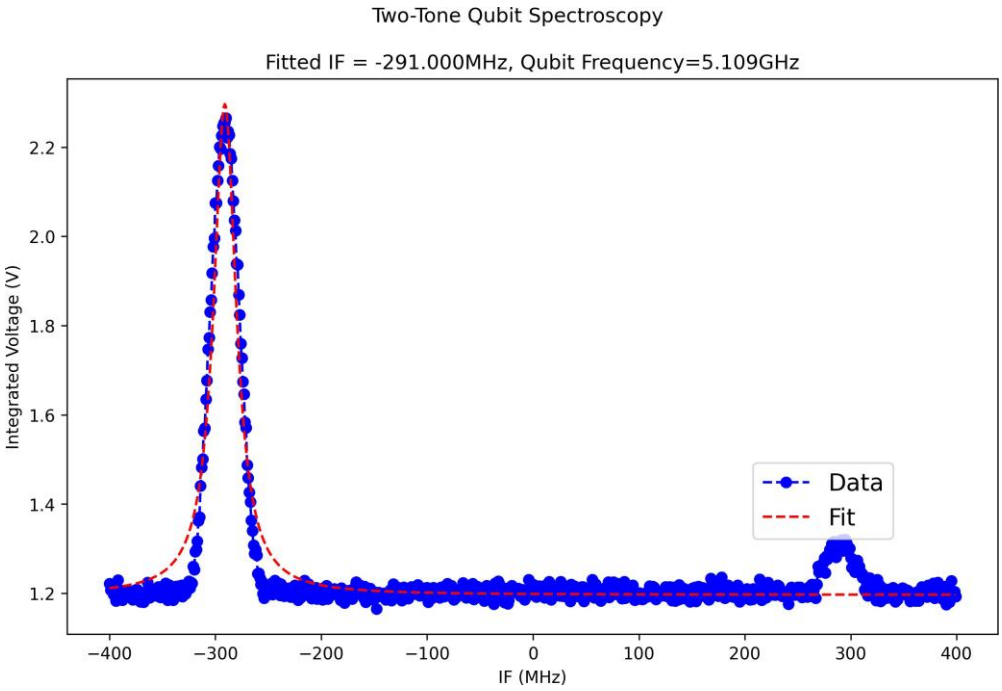


Figure 5.5: **Two-tone qubit spectroscopy.** Blue markers: measured integrated readout response versus swept intermediate frequency (IF) of the qubit drive. Red dashed curve: Lorentzian fit $L(f\,; f_0, \Gamma, A, C)$. The fit provides the resonance center $\hat{f}_{IF}$ (displayed in the title) and thus $f_{01} = f_{LO} + \hat{f}_{IF}$.

5.7 Two-Tone Spectroscopy vs Flux Offsets (One-time)

Goal

This experiment performs two–tone qubit spectroscopy as a function of a DC flux (bias) offset in order to map the qubit transition frequency f_{01} versus flux and identify the flux "sweet spot" (where $\partial f_{01}/\partial\Phi = 0$). The protocol uses a fixed local oscillator (LO) on the qubit drive line and sweeps the intermediate frequency (IF) of a short drive pulse while stepping a static flux offset on the flux line. A readout pulse is used to measure the demodulated response (I, Q) on the readout line.

Concretely:

- For each flux value $\Phi \in$ FLUX_VALUES, the drive frequency is swept across

$$f = f_{LO} + \text{IF}, \qquad \text{IF} \in \text{IF_VALUES} = \{-50\ \text{MHz} : 2\ \text{MHz} : +50\ \text{MHz}\} + \text{IF},$$

a window centered on the current IF setting.

- At each (Φ, f) point, a pulse is played on the drive bus, followed by a readout, and the integrated complex response $S_{21} = I + iQ$ is recorded.

The outcome is a 2D dataset $S_{21}(\Phi, \text{IF})$ (stored as I, Q), from which we extract $|S_{21}|$ resonance curves versus frequency at each flux bias. This enables locating the sweet-spot flux and the corresponding transition frequency.

Program

```
LO = platform.get_parameter(alias=READOUT_BUS, parameter=Parameter.LO_FREQUENCY)
IF = platform.get_parameter(alias=READOUT_BUS, parameter=Parameter.IF)
OFFSET = platform.get_parameter(alias=FLUX_BUS, parameter=Parameter.OFFSET)

IF_VALUES = np.arange(-50e6, 50e6, 2e6) + IF
FLUX_VALUES = np.linspace(-0.05, 0.05, 51) + OFFSET

qp = QProgram()
frequency = qp.variable(domain=Domain.Frequency, label="Frequency")
with qp.average(NSHOTS):
    with qp.loop(variable=frequency, values=IF_VALUES):
        qp.set_frequency(bus=READOUT_BUS, frequency=frequency)
        qp.play(bus=DRIVE_BUS, waveform=drive_wf)
        qp.sync()
        qp.measure(bus=READOUT_BUS, waveform=readout_wf, weights=weights)
        qp.wait(bus=READOUT_BUS, duration=RELAXATION_DURATION)

experiment = Experiment(label="Resonator Spectroscopy vs Qubit Flux Offsets")
offset = experiment.variable(domain=Domain.Voltage, label="Offset")
with experiment.loop(variable=offset, values=OFFSETS):
    experiment.set_parameter(bus=FLUX_BUS, parameter=Parameter.OFFSET, value=offset)
    experiment.execute_qprogram(qp)
```

Analysis

For each flux row Φ_i, the 1D trace $|S_{21}|(\Phi_i, \text{IF})$ is fitted to a Lorentzian line shape

$$L(f;\, f_0, \Gamma, A, C) = C + \frac{A}{1 + 4\left(\frac{f - f_0}{\Gamma}\right)^2}, \tag{5.13}$$

to obtain an estimate of the local transition frequency $\hat{f}_0(\Phi_i)$ (reported in IF; the absolute tone is $f_{LO} + \hat{f}_0$). Fits whose coefficient of determination falls below a threshold ($R^2 < 0.40$) are discarded; the remainder form the set $\{(\Phi_i, \hat{f}_0(\Phi_i))\}$.

Provided at least three valid points are available, a quadratic model is fit to the extracted centers:

$$\hat{f}_0(\Phi) \approx a\,\Phi^2 + b\,\Phi + c. \tag{5.14}$$

The sweet spot is taken as the vertex

$$\Phi^\star = -\frac{b}{2a}, \qquad f_{01}^\star = f_{LO} + a\,\Phi^{\star 2} + b\,\Phi^\star + c\,, \tag{5.15}$$

and the quality of the quadratic description is summarized by R^2 computed between the accepted $\hat{f}_0(\Phi_i)$ and the polynomial evaluated at the same Φ_i.

```
    # --- Fit |S21| row-by-row (one Lorentzian per flux) ---
    fit_values = np.empty((len(FLUX_VALUES), len(IF_VALUES)))
    fitted_ifs = np.empty(len(FLUX_VALUES))
    r_squareds = np.empty(len(FLUX_VALUES))
    mask = np.empty(len(FLUX_VALUES), dtype=bool)

    R2_THRESHOLD = 0.40

    for ii in range(len(FLUX_VALUES)):
        fitted_if, fitvals, rsq = lorentzian_fit(s21_modulus[ii], IF_VALUES)

        if (rsq is None) or (rsq < R2_THRESHOLD) or (not np.isfinite(fitted_if)):
            fitted_if = np.nan
            mask[ii] = False
        else:
            mask[ii] = True

        fitted_ifs[ii] = fitted_if
        fit_values[ii] = fitvals
        r_squareds[ii] = rsq

    # --- Quadratic trend vs flux to find sweet spot (only if enough good points) ---
    if np.count_nonzero(mask) >= 3:
        coeffs = np.polyfit(FLUX_VALUES[mask], fitted_ifs[mask], 2)
        sweetspot = -coeffs[1] / (2 * coeffs[0])

        # --- Plot |S21| colormap with quadratic curve ---
        fig, ax = plt.subplots(figsize=(10, 7))

        im = ax.pcolormesh(IF_VALUES + LO, FLUX_VALUES, s21_modulus)
        cbar = fig.colorbar(im, ax=ax)
        cbar.set_label('|S21|')
        flux_line = np.linspace(FLUX_VALUES[0], FLUX_VALUES[-1], 201)
        freq_fit = np.polyval(coeffs, flux_line)

        ax.plot(freq_fit + LO, flux_line, 'r', label='Quadratic fit')

        ax.axhline(sweetspot, color='black', label='Sweet spot')

        ax.set_title(
            f"Flux Offset = {sweetspot:.6f} V\n"
            f"Frequency = {(np.polyval(coeffs, sweetspot) + LO) * 1e-9:.3f} GHz"
        )

        ax.set_xlabel("Tone Frequency")
        ax.set_ylabel("Biasing Voltage (V)")
        ax.legend(loc='upper left')

        fig.suptitle(EXPERIMENT_NAME)
        fig.savefig(f'./{EXPERIMENT_NAME}.png', dpi=300, bbox_inches='tight')
```

```
plt.show()
```

Results

The two–tone map $|S_{21}|(\Phi, \mathrm{IF})$ shows a well-defined dispersive ridge whose maximum tracks the qubit transition. Row-wise Lorentzian fits yield a set of local centers $\{(\Phi_i, \hat{f}_0(\Phi_i))\}$; after rejecting low-quality rows ($R^2 < 0.40$), the remaining points are accurately captured by the quadratic trend (5.14), consistent with the local parabolic curvature of a flux-tunable transmon around its sweet spot. The vertex of the parabola gives the operating bias and frequency,

$$\Phi^\star \simeq -4.57 \times 10^{-4}\,\mathrm{V}, \qquad f_{01}^\star \simeq 5.108\ \mathrm{GHz},$$

as annotated on the panel. Operating at $\Phi^\star$ minimizes first-order flux sensitivity and is standard practice for fixed-bias calibrations and fast-gate operation on tunable transmons, where subsequent control focuses on microwave-only pulse shaping rather than DC retuning.

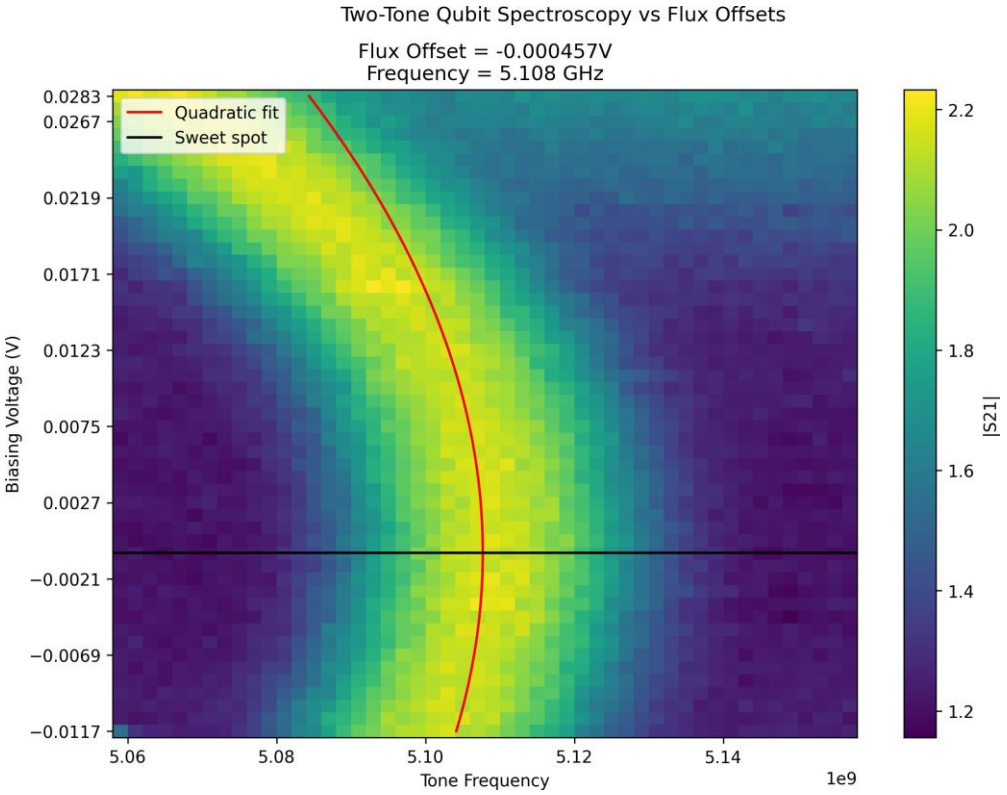


Figure 5.6: **Two-tone qubit spectroscopy versus DC flux offset.** Background: colormap of the measured magnitude $|S_{21}|$ as a function of absolute tone frequency and flux bias. For each flux row, a Lorentzian is fitted to $|S_{21}|$ to obtain $\hat{f}_0(\Phi)$ (in IF), then converted to absolute frequency $f_{\mathrm{LO}} + \hat{f}_0$. The accepted centers are summarized by a quadratic fit $a\Phi^2 + b\Phi + c$ (red curve), whose vertex $\Phi^\star = -b/(2a)$ defines the sweet-spot bias (black line). The header reports the extracted $\Phi^\star$ and the corresponding transition frequency $f_{01}^\star$.

5.8 Rabi oscillations (Per-variant)

Goal

The objective of the Rabi experiment is to determine, for a fixed drive duration T, the mapping between the dimensionless amplitude control $a \in [0, 1]$ applied on the drive line and the rotation angle $\vartheta(a)$ effected on the qubit Bloch vector. In particular, we extract the amplitude a_π that implements a π-rotation under the specific pulse envelope and hardware transfer function used in the experiment. The measurement is performed by sweeping the drive amplitude over a dense grid of 101 points between 0 and 1, keeping all other parameters constant, and recording the integrated readout quadratures (I, Q) after a single drive pulse and subsequent dispersive measurement. For each amplitude we form the complex demodulated signal and use $|s_{21}(a)|$ as the scalar response to be fitted. Hardware averaging over $N_{\text{avg}} = 10^4$ repetitions per amplitude point is used to suppress sampling noise.

The Rabi experiment is *repeated* for all four control-pulse variants used throughout this work: a baseline Gaussian envelope, the standard DRAG pulse, and the two frequency-domain analytical envelopes HD-DRAG and FAST-DRAG.

All Rabi scans are executed with identical measurement settings (readout pulse amplitude and duration, integration weights, repetition period) to ensure comparability across pulse variants. The output of this section is the set of calibrated amplitudes $\{a_\pi\}$ (and, when needed, $a_{2\pi}$) for each envelope, which serve as starting points for subsequent gate calibration and leakage characterization.

Program

```
AMPLITUDE_VALUES = np.linspace(0, 1.0, 101)

for pulse_gen in [gaussian_only_gate, gaussian_drag,gate, hd_drag_gate, fast_drag_gate]:
    I, Q, _ = pulse_gen(theta=np.pi, phase=0.0)

    qp = ql.QProgram()
    gain = qp.variable(label="Gain", domain=ql.Domain.Voltage)
    with qp.average(NSHOTS):
        qp.set_gain(bus=READOUT_BUS, gain=1.0)
        with qp.for_loop(variable=gain, values=AMPLITUDE_VALUES):
            qp.reset_phase(bus=DRIVE_BUS)
            qp.set_gain(bus=DRIVE_BUS, gain=gain)
            qp.play(bus=DRIVE_BUS, waveform=ql.IQPair(I=ql.Arbitrary(I), Q=ql.Arbitrary(Q)))
            qp.sync()
            qp.measure(bus=READOUT_BUS, waveform=square_wf, weights=weights)
            qp.wait(bus=READOUT_BUS, duration=RELAXATION_DURATION)

    experiment = ql.Experiment(EXPERIMENT_NAME)
    experiment.execute_qprogram(qp)
```

Analysis

On resonance ($\Delta = 0$) and for a fixed pulse duration T, the ideal excited-state population after a pulse of (effective) Rabi frequency Ω is

$$P_e(a) = \sin^2\left(\tfrac{1}{2}\Omega(a)\,T\right), \qquad \Omega(a) \approx \gamma\, a,$$

so that $P_e(a)$ is π-periodic in a with first maximum at $a_\pi^{(\mathrm{ideal})} = \pi/(\gamma T)$. In practice, finite anharmonicity, AC-Stark shifts, mixer imbalance, and the readout chain distort this simple law, and different pulse envelopes alter both γ and higher-order corrections in a. DRAG compensates the dominant spectral coupling to $|2\rangle$ by an off-quadrature derivative term, thereby restoring near-sinusoidal Rabi behavior and suppressing leakage; HD-/FAST-DRAG go further by imposing spectral zeros around the ef-transition, which improves the regularity of $P_e(a)$ at short T while enabling markedly faster gates. Moreover, phase-modulated (chirped) control can produce robust, plateau-like transfer curves; this informs our choice of fit models and thresholding when extracting a_π from $|s_{21}(a)|$.

We therefore model the measured scalar response $y(a) \equiv |s_{21}(a)|$ with two complementary families and select the one that best describes the data by the coefficient of determination R^2.

$$R^2 \equiv 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}, \qquad \bar{y} = \frac{1}{N}\sum_i y_i.$$

i) *Sinusoidal fit.* We fit

$$f_{\sin}(a) = A \sin(\omega a + \varphi) + B,$$

with parameters (A, ω, φ, B). To avoid local minima and accommodate non-uniform contrast, ω is initialized by the dominant non-DC Fourier component of $y(a)$ over the amplitude grid (i.e., by the argmax of $|\mathrm{F}\{y\}|$ excluding $k = 0$), and the full parameter vector is refined by a globalized local optimizer. The first set of extrema of $f_{\sin}$ occurs at

$$\omega a + \varphi = \frac{\pi}{2} + 2\pi k \quad \text{(maxima)}, \qquad \omega a + \varphi = \frac{3\pi}{2} + 2\pi k \quad \text{(minima)},$$

yielding candidate abscissae $a_k^{(\sin)} = \left(\frac{\pi}{2} - \varphi + 2\pi k\right)/\omega$ and $a_k^{(\sin)} = \left(\frac{3\pi}{2} - \varphi + 2\pi k\right)/\omega$. Among these, we select the smallest a within the sweep range that lies above an empirical guard band

$$a_{\mathrm{thr}} = a_{\min} + \frac{1}{12}\left(a_{\max} - a_{\min}\right),$$

to reject spurious near-zero extrema induced by offsets and residual mixer transients. This yields the sinusoidal estimate $a_\pi^{(\sin)}$.

ii) *Polynomial fit.* As a non-parametric baseline, we fit a degree-8 polynomial

$$f_{\mathrm{poly}}(a) = \sum_{n=0}^{8} c_n a^n,$$

and locate its interior turning points by the discrete sign-change pattern of the first-difference sequence over the amplitude grid (equivalently, by the roots of f'_{poly} projected onto the sampled abscissae). The smallest turning point above a_{thr} furnishes $a_\pi^{(\mathrm{poly})}$.

We compute $R^2_{\sin}$ and R^2_{poly} on $y(a)$ and retain $a_\pi = a_\pi^{(\sin)}$ if $R^2_{\sin} \geq R^2_{\mathrm{poly}}$, otherwise $a_\pi = a_\pi^{(\mathrm{poly})}$. The corresponding fitted curve $\hat{y}(a)$ is used solely for visualization and quality control; the reported π-amplitude is the scalar a_π (in the instrument's normalized amplitude units), and the optimal drive set-point is obtained by setting the drive gain to a_π for the given pulse variant.

A few remarks are in order. First, because we fit $|s_{21}|$ rather than a direct estimator of P_e, the fitted baseline B (and, for the polynomial, the offset c_0) absorb readout-chain gain and quadrature rotation; this

is inconsequential for the location of a_π. Second, DRAG-type pulses alter the effective γ and suppress leakage without invalidating the sinusoidal ansatz at modest durations; this is precisely the regime where $R^2_{\sin}$ dominates and the sinusoidal estimator is preferred. Third, at very short T or with aggressively notched spectra (HD-/FAST-DRAG), the response can deviate from a pure sinusoid; in this case the polynomial model often provides a more faithful local interpolation around the first peak, yet still yields a unique and stable a_π. Finally, for phase- or frequency-modulated pulses that realize adiabatic or quasi-adiabatic population transfer, the Rabi curve can display plateau-like regions.

The outcome of the analysis is the set of four calibrated amplitudes $\{a_\pi^{(\text{Gauss})}, a_\pi^{(\text{DRAG})}, a_\pi^{(\text{HD-DRAG})}, a_\pi^{(\text{FAST-DRAG})}\}$, each with an associated goodness-of-fit score and diagnostic plot.

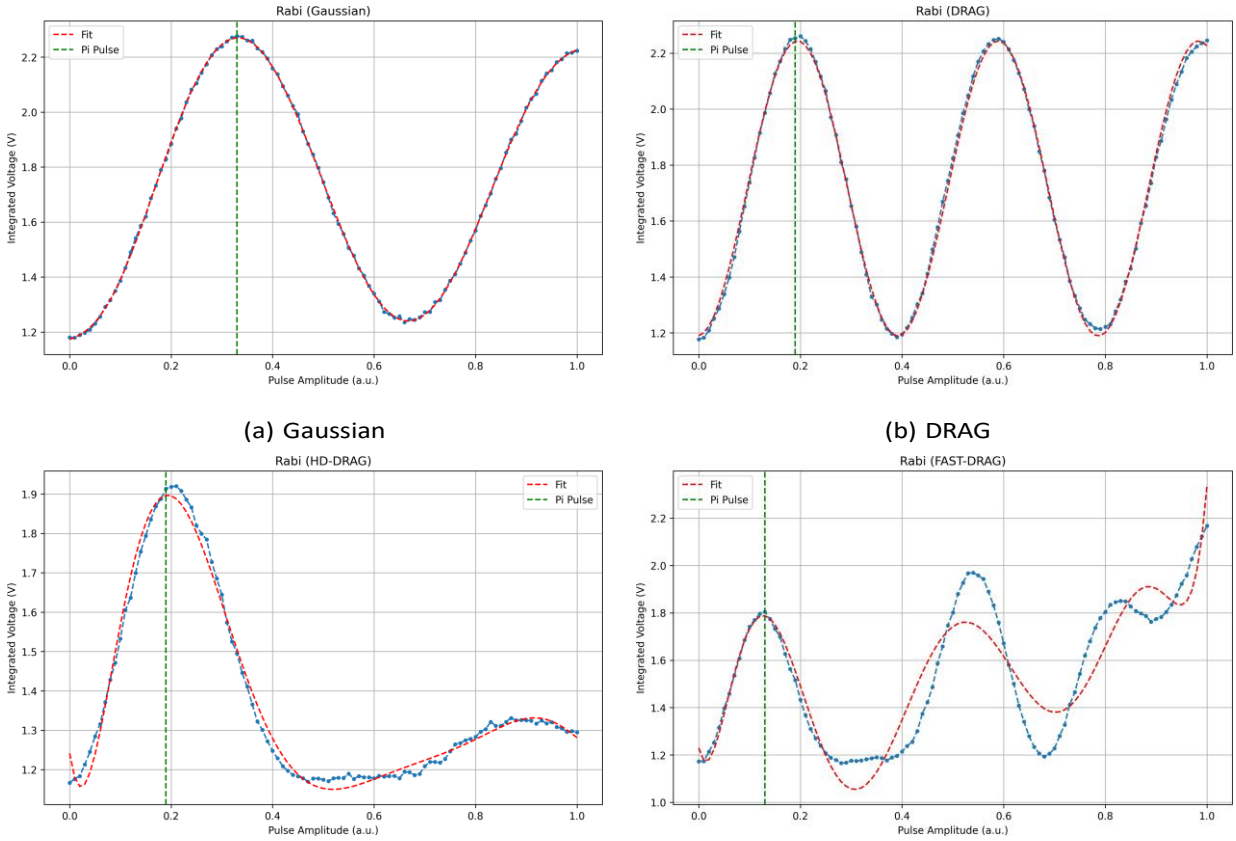
```
def sinus(x, a, b, c, d):
    return a * np.sin(b * np.array(x) + c) + d

def cosine(x, a, b, c, d):
    return a * np.cos(b * np.array(x) + c) + d

# Poly fit (8th order)
z = np.polyfit(AMPLITUDE_VALUES, s21_modulus, 8)
p = np.poly1d(z)
fit_poly = p(AMPLITUDE_VALUES)

# Sinus fit with FFT-based initialization
mod = lmfit.Model(sinus)

tt = np.array(AMPLITUDE_VALUES)
yy = np.array(s21_modulus)
ff = np.fft.fftfreq(len(tt), (tt[1] - tt[0])) * 2 * np.pi
Fyy = np.abs(np.fft.fft(yy))
guess_freq = np.abs(ff[np.argmax(Fyy[1:]) + 1])
guess_amp = np.std(yy) * np.sqrt(2.0)
guess_offset = np.mean(yy)

# Initial parameter hints
mod.set_param_hint("a", value=guess_amp)
mod.set_param_hint("b", value=guess_freq)
mod.set_param_hint("c", value=0.0)
mod.set_param_hint("d", value=guess_offset)

params = mod.make_params()
fit = mod.fit(data=s21_modulus, params=params, x=AMPLITUDE_VALUES, method="basinhopping")

a_value = fit.params["a"].value
b_value = fit.params["b"].value
c_value = fit.params["c"].value
d_value = fit.params["d"].value
popt = [a_value, b_value, c_value, d_value]

# T/8 with T as the sweep interval (kept as in original implementation)
peak_threshold = AMPLITUDE_VALUES[0] + np.abs((AMPLITUDE_VALUES[-1] - AMPLITUDE_VALUES[0]) / 12)

fit_sinus = sinus(AMPLITUDE_VALUES, *popt)
```

```
peaks_max_sinus, _ = find_peaks(fit_sinus)
peaks_min_sinus, _ = find_peaks(-fit_sinus)

all_peaks_indices_sinus = np.concatenate([peaks_max_sinus, peaks_min_sinus])
all_peaks_indices_sinus.sort()

# Assumes a helper function `find_peaks_poly` is available in your environment
all_peaks_indices_poly = find_peaks_poly(fit_poly)

peak_sinus = AMPLITUDE_VALUES[all_peaks_indices_sinus[0]] if len(all_peaks_indices_sinus) != 0 else AMPLITUDE_
peak_2_sinus = min((peak_sinus + np.abs(1 / (popt[1]))), AMPLITUDE_VALUES[-1])

peak_poly = (
    AMPLITUDE_VALUES[all_peaks_indices_poly[0]]
    if len(all_peaks_indices_poly) != 0
    else AMPLITUDE_VALUES[-1]
)
peak_2_poly = (
    AMPLITUDE_VALUES[all_peaks_indices_poly[1]]
    if len(all_peaks_indices_poly) > 1
    else AMPLITUDE_VALUES[-1]
)

fitted_pi_pulse_amplitude_sinus = peak_sinus if peak_sinus >= peak_threshold else peak_2_sinus
fitted_pi_pulse_amplitude_poly = peak_poly if peak_poly >= peak_threshold else peak_2_poly

# Compare and select fittings (R²)
r2_poly = r2_score(s21_modulus, fit_poly)
r2_sinus = r2_score(s21_modulus, fit_sinus)

chosen_fit = "poly" if r2_poly > r2_sinus else "sinus"

# Save results in the same structure as before
fits = {
    "poly": [fitted_pi_pulse_amplitude_poly, fit_poly, r2_poly],
    "sinus": [fitted_pi_pulse_amplitude_sinus, fit_sinus, r2_sinus],
}

# Final outputs (kept consistent with previous variable names)
f_popt = popt
pi_amp = fits[chosen_fit][0]
fit_values = fits[chosen_fit][1]
r_squared = fits[chosen_fit][2]

pi_fitted = float(pi_amp)

plt.figure(figsize=(10, 6))
plt.plot(AMPLITUDE_VALUES, s21_modulus,'.--')
plt.plot(AMPLITUDE_VALUES, fit_values, "--", color="red", label="Fit")
plt.axvline(pi_fitted, color="green", linestyle="--", label="Pi Pulse")
plt.legend()
plt.grid(which='both')
```

```
plt.xlabel('Pulse Amplitude (a.u.)')
plt.ylabel('Integrated Voltage (V)')
plt.title(EXPERIMENT_NAME)
plt.savefig(f"./{EXPERIMENT_NAME}", dpi=300, bbox_inches='tight')
plt.show()
```

Results

The four Rabi scans exhibit the expected envelope-dependent modulation of the response $y(a) = |s_{21}(a)|$ and yield a unique π-amplitude a_π for each control shape (vertical green dashed lines). For the Gaussian and standard DRAG pulses the curves are close to sinusoidal over the first period; the selection rule therefore picks the sinusoidal model, which achieves $R^2 \approx 1$ and provides a clean estimate of a_π. In contrast, HD-DRAG and FAST-DRAG show pronounced deviations from a pure sinusoid (secondary lobes and asymmetry) caused by the engineered spectral zeros and the finite transfer function; in these cases the polynomial interpolant gives the higher R^2 and is used to locate a_π. Across variants, the DRAG-family pulses require a smaller drive amplitude to reach π rotation than the baseline Gaussian, consistent with their larger effective on-resonant Rabi rate under identical duration and hardware settings. These calibrated $\{a_\pi\}$ are used as the drive set-points in subsequent calibrations and leakage benchmarks.



(a) Gaussian (b) DRAG

(c) HD-DRAG (d) FAST-DRAG

Figure 5.7: **Rabi oscillations for four pulse envelopes.** Blue points: measured $|s_{21}(a)|$ versus normalized drive amplitude a. Red dashed line: selected fit $\hat{y}(a)$ (sinusoidal for Gaussian/DRAG; degree-8 polynomial for HD-/FAST-DRAG). Green dashed line: extracted π-amplitude a_π used for calibration. All scans share the same duration and readout configuration to enable direct comparison across variants.

5.9 Single-Shot Readout (SSRO) (Per-variant)

Goal

The Single-Shot Readout (SSRO) experiment quantifies how well a single, fixed readout pulse discriminates the computational basis states $|0\rangle$ and $|1\rangle$. We acquire large ensembles of single-shot outcomes by repeatedly preparing $|0\rangle$ (no excitation) and $|1\rangle$ (a calibrated π pulse), then performing a single dispersive measurement that yields one complex datum (I, Q) per shot. The deliverables are:

- an optimal linear discriminator defined by a rotation angle $\vartheta^\star$ in the IQ plane and a scalar threshold $\tau^\star$ on the rotated axis;
- an average assignment fidelity F_{avg} that reflects real preparation & measurement imperfections;

- a discrimination-limited fidelity $F_{\text{avg}}^{(\text{ideal})}$ that factors out thermal population and measurement relaxation;
- diagnostic plots (2D histograms, 1D histograms/CDFs along the discriminator) and summary statistics (cluster widths, goodness-of-fit).

Program

```
for pulse_gen in [gaussian_only_gate, gaussian_drag,gate, hd_drag_gate, fast_drag_gate]:
    I0, Q0, _ = pulse_gen(theta=0.0, phase=0.0)
    Ipi, Qpi, _ = pulse_gen(theta=np.pi, phase=0.0)

    qp = ql.QProgram()
    bins = qp.variable(label="Bins", domain=ql.Domain.Scalar, type=int)
    qp.set_gain(bus=READOUT_BUS, gain=1.0)
    qp.set_gain(bus=DRIVE_BUS, gain=1.0)
    with qp.for_loop(variable=bins, start=1, stop=N_BINS):
        qp.play(bus=DRIVE_BUS, waveform=ql.IQPair(ql.Arbitrary(I0), ql.Arbitrary(Q0)))
        qp.sync()
        qp.measure(bus=READOUT_BUS, waveform=square_wf, weights=ql.IQPair(I=WEIGHTS_I, Q=WEIGHTS_Q))
        qp.wait(bus=DRIVE_BUS,duration=RELAXATION_DURATION)
        qp.play(bus=DRIVE_BUS, waveform=ql.IQPair(ql.Arbitrary(Ipi), ql.Arbitrary(Qpi)))
        qp.sync()
        qp.measure(bus=READOUT_BUS, waveform=square_wf, weights=ql.IQPair(I=WEIGHTS_I, Q=WEIGHTS_Q))
        qp.wait(bus=DRIVE_BUS,duration=RELAXATION_DURATION)

    experiment = ql.Experiment(EXPERIMENT_NAME)
    experiment.execute_qprogram(qp)
```

Analysis

Let $\{(I_0, Q_0)\}$ and $\{(I_1, Q_1)\}$ denote the single-shot clouds obtained after preparing $|0\rangle$ and $|1\rangle$, respectively. We first center the data at the $|0\rangle$ centroid $(\bar{I}_0, \bar{Q}_0)$:

$$x_k^{(b)} = I_k^{(b)} - \bar{I}_0, \qquad y_k^{(b)} = Q_k^{(b)} - \bar{Q}_0, \qquad b \in \{0, 1\}.$$

Define the separation vector $\Delta = (\bar{I}_1 - \bar{I}_0, \bar{Q}_1 - \bar{Q}_0)$ and its angle

$$\vartheta = \arctan 2(\Delta_Q, \Delta_I).$$

Project each centered point onto the axis parallel to Δ,

$$u_k^{(b)} = x_k^{(b)} \cos\vartheta + y_k^{(b)} \sin\vartheta,$$

so that discrimination reduces to a one-dimensional threshold test on u (linear discriminant in the IQ plane).

From the projected data we form empirical CDFs $F_0(u)$ and $F_1(u)$ for $|0\rangle$ and $|1\rangle$. The optimal scalar threshold $\tau^\star$ maximizing the single-threshold separation is chosen by

$$\tau^\star = \arg\max_u \; F_1(u) - F_0(u) \,,$$

which maximizes Youden's J = TPR − FPR and yields the average assignment fidelity

$$F_{\text{avg}} = \tfrac{1}{2}\,\text{TPR} + \text{TNR} = \tfrac{1}{2}\,1 + \max_u F_1(u) - F_0(u) \,.$$

The corresponding threshold line in the original IQ plane is obtained by de-projecting $\tau^\star$:

$$\tau_{\mathrm{IQ}}^{\star} = \bar{I}_0 \cos\vartheta + \bar{Q}_0 \sin\vartheta + \tau^\star .$$

Classification is then $u \geq \tau^\star \Rightarrow |1\rangle$, else $|0\rangle$.

To separate intrinsic readout overlap from state-preparation and measurement (SPAM) effects, we fit a simple generative model to the CDFs along u:

$$F_0(u) \approx (1 - p_{\mathrm{th}})\,\Phi\left(\frac{u-\mu_0}{\sigma_0}\right) + p_{\mathrm{th}}\,\Phi\left(\frac{u-\mu_1}{\sigma_1}\right),$$

$$F_1(u) \approx (1 - p_{\mathrm{rel}})\,\Phi\left(\frac{u-\mu_1}{\sigma_1}\right) + p_{\mathrm{rel}}\,\Phi\left(\frac{u-\mu_0}{\sigma_0}\right),$$

where Φ is the Gaussian CDF, μ_b, σ_b are state-dependent means and widths, p_{th} captures thermal excitation when preparing $|0\rangle$, and p_{rel} captures measurement-induced relaxation when preparing $|1\rangle$. The fit returns $\{\mu_b, \sigma_b, p_{\mathrm{th}}, p_{\mathrm{rel}}\}$ and a reduced χ^2 as a quality metric. Setting $p_{\mathrm{th}} = p_{\mathrm{rel}} = 0$ in the fitted model yields the discrimination-limited figure of merit $F_{\mathrm{avg}}^{(\mathrm{ideal})}$.

We report:

- the discriminator orientation $\vartheta^\star$ and threshold $\tau^\star$ (in integrated-signal units);
- F_{avg} from the empirical CDFs at $\tau^\star$ and $F_{\mathrm{avg}}^{(\mathrm{ideal})}$ from the generative fit;
- cluster widths (σ_0, σ_1) and their ratio (noise imbalance);
- reduced χ^2 of the CDF fit.

```
import lmfit

with ql.ExperimentResults(results_path) as results:
    data0, dims0 = results.get(0, 0)
    data1, dims1 = results.get(0, 1)

i0, q0 = data0[..., 0], data0[..., 1]
i1, q1 = data1[..., 0], data1[..., 1]

center_0_i, center_0_q = np.mean(i0), np.mean(q0)
center_1_i, center_1_q = np.mean(i1), np.mean(q1)

dist_i = center_1_i - center_0_i
dist_q = center_1_q - center_0_q

# x,y are I,Q centered in 0
x_0 = i0 - center_0_i
y_0 = q0 - center_0_q
x_1 = i1 - center_0_i
y_1 = q1 - center_0_q

# u,v are x,v rotated
theta = np.arctan2(dist_q, dist_i)
u_0 = x_0 * np.cos(theta) + y_0 * np.sin(theta)
u_1 = x_1 * np.cos(theta) + y_1 * np.sin(theta)
```

```
perc_vec = np.linspace(0, 100, 101)
cdf_0 = np.percentile(u_0, perc_vec)
cdf_1 = np.percentile(u_1, perc_vec)

dummy_x = np.linspace(np.min((cdf_0, cdf_1)), np.max((cdf_0, cdf_1)), 101)
interp_cdf_0 = np.interp(dummy_x, cdf_0, perc_vec)
interp_cdf_1 = np.interp(dummy_x, cdf_1, perc_vec)
diff_cdf = interp_cdf_1 - interp_cdf_0

idx_max = np.argmax(np.abs(diff_cdf))
threshold = dummy_x[idx_max]  # threshold in u axis
# i want threshold in the u axis, but without the shifts
threshold_d = threshold + center_0_i * np.cos(theta) + center_0_q * np.sin(theta)
f_asg = np.abs(diff_cdf[idx_max])
f_avg = (100 + f_asg) / 2

cdfs_mod = lmfit.Model(joint_model)

v0_guess = np.mean(u_0)
std0_guess = np.std(u_0)
v1_guess = np.mean(u_1)
std1_guess = np.std(u_1)

y = joint_model(
    dummy_x,
    std0=std0_guess, v0=v0_guess, std1=std1_guess, v1=v1_guess,
    mmt_relax=0.7, thermal_pop=0.03, N=100
)

cdfs_mod.set_param_hint("mmt_relax", value=0.7, min=0, max=1, vary=True)
cdfs_mod.set_param_hint("thermal_pop", value=0.03, min=0, max=1, vary=True)
cdfs_mod.set_param_hint("std0", value=std0_guess, vary=True)
cdfs_mod.set_param_hint("std1", value=std1_guess, vary=True)
cdfs_mod.set_param_hint("v0", value=v0_guess, vary=True)
cdfs_mod.set_param_hint("v1", value=v1_guess, vary=True)
cdfs_mod.set_param_hint("N", value=100, min=0, vary=False)
params = cdfs_mod.make_params()
fit_res = cdfs_mod.fit(data=np.concatenate((interp_cdf_1, interp_cdf_0)), x=dummy_x, params=params)

chi_square = fit_res.redchi
std_0=fit_res.best_values["std0"]
std_1=fit_res.best_values["std1"]

y = joint_model(dummy_x, **fit_res.best_values)
ideal_dict = deepcopy(fit_res.best_values)
ideal_dict["mmt_relax"] = 0
ideal_dict["thermal_pop"] = 0

y_ideal = joint_model(dummy_x, **ideal_dict)
f_asg_ideal = np.max(np.abs(y_ideal[: len(y_ideal) // 2] - y_ideal[len(y_ideal) // 2 :]))
f_avg_ideal = (100 + f_asg_ideal) / 2
```

```
fig, axs = plt.subplots(1, 3, figsize=(17, 4))
ax = axs[0]
ax.tick_params(axis="both",  labelsize=12)
_ = ax.hist(u_0, bins=101, alpha=0.5, label="|1>")
_ = ax.hist(u_1, bins=101, alpha=0.5, label="|0>")
# plot fits as pdf
fit_0_u = np.linspace(u_0.min(), u_0.max(), 101)
prep0_m0 = 1 - fit_res.best_values["thermal_pop"]
prep1_m0 = fit_res.best_values["mmt_relax"]
fit_0_y = 100 * two_gaussians(
    x=fit_0_u / np.cos(theta),
    a0=prep0_m0,
    std0=fit_res.best_values["std0"],
    v0=fit_res.best_values["v0"],
    std1=fit_res.best_values["std1"],
    v1=fit_res.best_values["v1"],
)
cdf_0_fit = y[len(y) // 2 :]
ax.plot(
    (dummy_x[1:] + dummy_x[:-1]) / 2,
    100 * (cdf_0_fit[1:] - cdf_0_fit[:-1]) / 4,
    "--",
    label="Fit |0>",
    color="purple",
)
cdf_1_fit = y[: len(y) // 2]
ax.plot(
    (dummy_x[1:] + dummy_x[:-1]) / 2,
    100 * (cdf_1_fit[1:] - cdf_1_fit[:-1]) / 4,
    "--",
    label="Fit |1>",
    color="r",
)
ax.set_xlabel("Integrated Voltage (mV)", fontsize=14)
ax.set_ylabel("Counts", fontsize=14)
ax.set_title(
    f'Mmt relaxation={100*(1-fit_res.best_values["thermal_pop"]):.1f} % | Thermal pop={100*(1-fit_res.best_val
    fontsize=14,
)
ax = axs[1]
ax.set_xlabel("Integrated Voltage (mV)", fontsize=14)
ax.tick_params(axis="both", labelsize=12)
ax.plot(cdf_0, perc_vec, "o", label="|0>", alpha=0.3)
ax.plot(dummy_x, y[len(y) // 2 :], "--", label="Fit |0>", color="purple")
ax.plot(cdf_1, perc_vec, "o", label="|1>", alpha=0.3)
ax.plot(dummy_x, y[: len(y) // 2], "--", label="Fit |1>", color="r")
ax.axvline(
    threshold, linestyle="--", color="k", label=f"Threshold={threshold_d:.5f}; theta={180*theta/np.pi:.1f} deg
)
ax.set_title(f"Average Fidelity = {f_avg:.1f} % \n Assignment Fidelity = {f_avg_ideal:.1f} %", fontsize=14)
ax.legend()
```

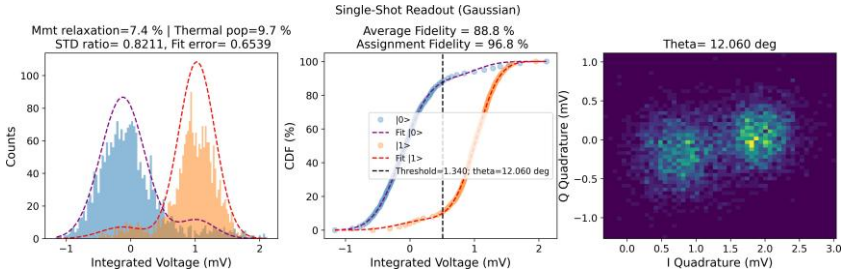
```
ax.set_ylabel("CDF (%)", fontsize=14)
ax = axs[2]
ax.set_xlabel("I Quadrature (mV)", fontsize=14)
ax.set_ylabel("Q Quadrature (mV)", fontsize=14)
ax.tick_params(axis="both", labelsize=12)
ax.set_title(f"Theta= {theta*180/np.pi} deg", fontsize=14)
ax.hist2d(np.concatenate((i0, i1)), np.concatenate((q0, q1)), bins=51)
plt.title(EXPERIMENT_NAME)
plt.savefig(f"./{EXPERIMENT_NAME}", dpi=300, bbox_inches='tight')
plt.show()
```

Results

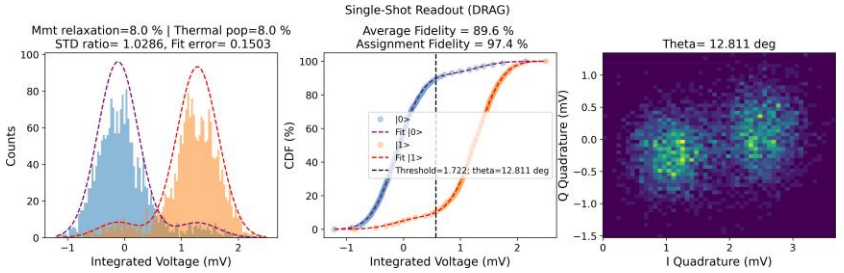
Across all four control envelopes the single–shot clouds $\{(I_0, Q_0)\}$ and $\{(I_1, Q_1)\}$ are well separated by a linear discriminant obtained from the empirical centroids. The projection–based threshold $\tau^\star$ (chosen to maximize $|F_1 - F_0|$) yields average assignment fidelities in the 88–93% range, while the discrimination–limited figures (obtained from the Gaussian–mixture CDF fit with $p_{\mathrm{th}} = p_{\mathrm{rel}} = 0$) lie near 96–99%. Representative values extracted from the panels are:

$$
\begin{aligned}
&\text{Gaussian:} && F_{\mathrm{avg}} \approx 88.8\%,\ F_{\mathrm{avg}}^{(\mathrm{ideal})} \approx 96.8\%,\ \vartheta^\star \approx 12.1^\circ,\\
&\text{DRAG:} && F_{\mathrm{avg}} \approx 89.6\%,\ F_{\mathrm{avg}}^{(\mathrm{ideal})} \approx 97.4\%,\ \vartheta^\star \approx 12.8^\circ,\\
&\text{HD-DRAG:} && F_{\mathrm{avg}} \approx 90.6\%,\ F_{\mathrm{avg}}^{(\mathrm{ideal})} \approx 98.0\%,\ \vartheta^\star \approx 13.6^\circ,\\
&\text{FAST-DRAG:} && F_{\mathrm{avg}} \approx 91.3\%,\ F_{\mathrm{avg}}^{(\mathrm{ideal})} \approx 98.5\%,\ \vartheta^\star \approx 0.63^\circ.
\end{aligned}
$$

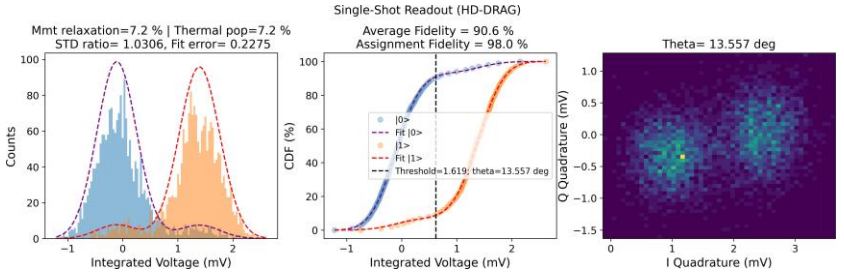
The Gaussian–mixture fit further reports small but non-negligible SPAM contributions (thermal population and measurement relaxation of order 6–10%), explaining the gap between F_{avg} and $F_{\mathrm{avg}}^{(\mathrm{ideal})}$. The cluster-width ratio σ_1/σ_0 is close to unity for all variants, indicating near-balanced noise on the two classes. Among the tested envelopes, the FAST-DRAG preparation yields the highest F_{avg} and the most axis-aligned discriminator (small $\vartheta^\star$), whereas DRAG and HD-DRAG provide comparable separations with slightly larger discriminator angles. These calibrated $(\vartheta^\star, \tau^\star)$ are used as the fixed readout discriminator for the remainder of the experiments.



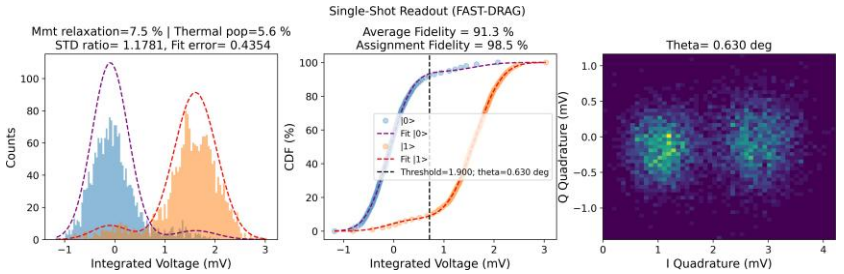
(a) Gaussian



(b) DRAG



(c) HD-DRAG



(d) FAST-DRAG

Figure 5.8: **Single-shot readout characterization for each pulse variant.** *Left:* 1D histograms along the optimal projection axis u with Gaussian–mixture PDFs from the CDF fit (dashed). *Middle:* empirical CDFs $F_0(u)$, $F_1(u)$ and the optimal threshold $\tau^\star$ (vertical dashed line); the panel annotates the empirical average fidelity F_{avg} and the discrimination-limited value $F_{\text{avg}}^{(\text{ideal})}$. *Right:* 2D histogram of (I, Q) with the fitted discriminator orientation $\vartheta^\star$.

5.10 Randomized benchmarking (RB) (Per-variant)

Goal

The goal of this experiment is to quantify the average performance of single-qubit control under realistic noise by means of randomized benchmarking (RB). We implement RB by applying sequences of uniformly random single-qubit Clifford operations and measuring the survival probability as a function of sequence length. For each sequence length m (number of Cliffords) we execute N_{seed} independently sampled random sequences and record the ground-state population after measurement. Sequence lengths are chosen to span the initial linear-response regime through the asymptotic baseline, providing dynamic range sufficient to identify a single-parameter decay. The average number of native physical pulses per Clifford is fixed to $N_{\text{G}} = 1.875$, consistent with the compilation used in control. All acquisition and analysis steps (including fitting) are performed identically for the four pulse variants to enable a direct, apples-to-apples comparison of fidelities and inferred error rates across shaping strategies.

Analysis

Under gate- and time-independent noise twirled by the single-qubit Clifford group, the mean survival probability obeys the standard RB ansatz

$$P(m) = Ap^{m} + B = Ae^{-\lambda m} + B, \qquad \lambda \equiv -\ln p, \tag{5.16}$$

where m is the Clifford sequence length, $p \in [0, 1]$ is the depolarizing parameter, and A, B absorb state-preparation-and-measurement (SPAM) effects. We fit (5.16) to the seed-averaged data $P(m)$ using nonlinear least squares with (λ, A, B) as free parameters.[3] For a single qubit ($d = 2$), the average Clifford fidelity is

$$F_{\text{Cl}} = \frac{1 + p}{2}, \tag{5.17}$$

and the average *native* gate fidelity (for the chosen compilation) follows by distributing the Clifford over N_{G} physical pulses,

$$F_{\text{g}} = F_{\text{Cl}}^{\,1/N_G} = \left(\frac{1 + e^{-\lambda}}{2}\right)^{1/N_G}, \qquad \varepsilon_{\text{g}} \equiv 1 - F_{\text{g}}. \tag{5.18}$$

Uncertainties on $\overline{P}(m)$ are estimated from the empirical spread across seeds at fixed m; these are propagated through the fit to obtain intervals for (λ, A, B) and, via (5.17)–(5.18), for F_{g} and ε_{g}. The offset B captures residual SPAM bias and, in practice, reflects the asymptotic polarization of the randomized sequence; omitting an explicit recovery Clifford (as done here) changes A, B but leaves the estimator for p valid under the usual RB assumptions, since randomization drives the effective channel toward the depolarizing fixed point.

Two features of the analysis are pulse-shape dependent. First, for leakage-suppressing variants (DRAG-L, FAST DRAG-L, HD DRAG-L), the fitted p is typically closer to unity at short m because spectral notches at the anharmonic detuning reduce population transfer out of the computational subspace, thereby mitigating non-Pauli errors that would otherwise degrade the effective depolarizing parameter. Second, for very short gates the observed improvement in p is consistent with frequency-domain shaping that explicitly zeros or strongly attenuates the drive spectrum at $\alpha/(2\pi)$ (the *ef* transition), a design objective of FAST DRAG and HD DRAG that strengthens leakage resilience without closed-loop optimization.

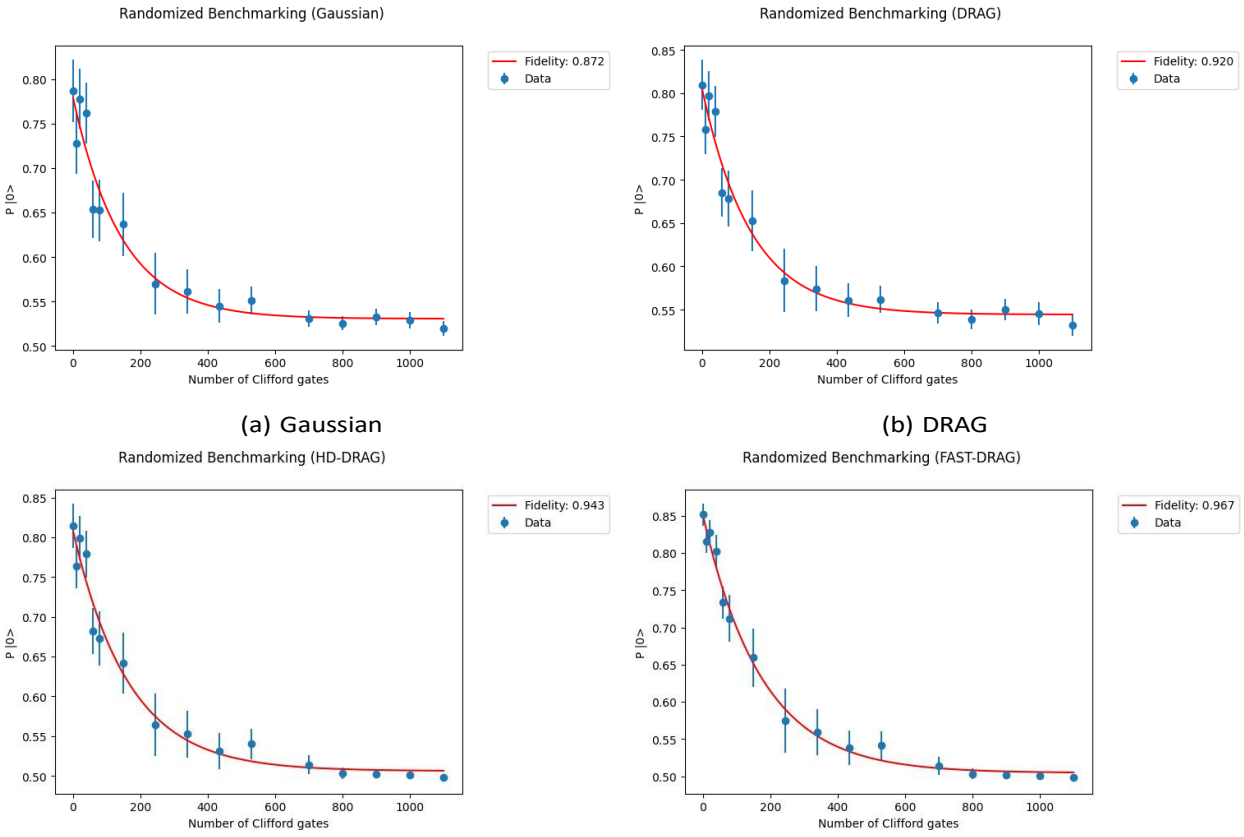
Concretely, our fits use the grid of sequence lengths $m \in$ CLIFFORD_GATES and report ε_{g} per pulse by applying (5.18) with $N_{\text{G}} = 1.875$. The entire pipeline—decay fit (5.16), conversion to F_{Cl} via (5.17), and to

[3] In code we parametrize the decay via $\ln p = \lambda$ and compute $p = e^{-\lambda}$.

F_g and ε_g via (5.18)—is rerun independently for each of the four pulse variants. This yields a like-for-like comparison that attributes differences in ε_g to control shaping rather than compilation or readout, thereby isolating the gains expected from DRAG and its FAST/HD extensions in weakly-anharmonic superconducting qubits.

Results

For all four pulse families the seed–averaged survival probability $\overline{P}(m)$ follows a single-exponential decay over the entire range of sequence lengths, validating the RB ansatz in Eq. (5.16). The fitted depolarizing parameter increases monotonically from the baseline Gaussian to DRAG, HD-DRAG, and FAST-DRAG, indicating progressively slower decay (higher fidelity) as spectral shaping suppresses leakage and non-Pauli errors. The offsets B cluster near the expected randomized baseline (≈ 0.5), and the spread across seeds is well captured by the error bars. Converting the fits via Eqs. (5.17)–(5.18) (with $N_G = 1.875$) yields the smallest per-gate error ε_g for FAST-DRAG, followed by HD-DRAG, then DRAG, with the Gaussian envelope performing worst. These results are consistent with frequency-domain shaping that zeros or attenuates spectral weight near $|\alpha|/(2\pi)$, thereby reducing leakage during short gates.



(a) Gaussian (b) DRAG

(c) HD-DRAG (d) FAST-DRAG

Figure 5.9: **Randomized benchmarking for each pulse variant.** Blue points (with error bars): seed-averaged survival probability $\overline{P}(m)$ vs. Clifford length m. Red line: nonlinear least-squares fit to $Ae^{-\lambda m}+B$. The legend in each panel reports the fitted average fidelity (converted from $p = e^{-\lambda}$); per-gate metrics follow from Eqs. (5.17)–(5.18) using $N_G = 1.875$. The decay becomes progressively slower from Gaussian to FAST-DRAG, reflecting improved control with leakage-suppressing spectral shaping.

Chapter 6

Conclusions

This thesis introduced and assessed an analytic route to fast, low–leakage single–qubit control on weakly anharmonic superconducting qubits. The central contribution is a frequency–domain design of pulse envelopes that enforces spectral selectivity under a fixed rotation area. Two complementary families were developed. In the Fourier–ansatz spectrum–tuning DRAG (FAST–DRAG), the in–phase envelope is synthesized to place and weight spectral notches at user–defined bands, while the quadrature follows from DRAG so that leakage suppression and AC–Stark compensation are disentangled at the level of calibration knobs. In the higher–derivative DRAG (HD–DRAG), smooth base envelopes are augmented by higher–order time derivatives so as to realize higher–order spectral zeros at the leakage transition while preserving endpoint smoothness. These constructions were embedded in a reproducible experimental stack with standardized compilation, acquisition and analysis, and were benchmarked *like–for–like* against conventional Gaussian/cosine DRAG through leakage scans, diagnostic sequences, and randomized benchmarking at fixed gate time.

Across the experiments, both analytic families reduced spectral weight at the primary leakage transition and thereby lowered seepage compared with conventional DRAG at comparable bandwidth. The resulting gates exhibited the expected robustness to moderate hyperparameter variation and, at the shortest calibrated durations, leakage was no longer the dominant limitation. Instead, residual coherent error consistent with distortions in the control delivery chain determined the observed performance. In other words, the analytic shaping achieved its goal of moving the limiting mechanism from spectral selectivity to hardware–imposed distortions and calibration systematics.

A key limitation of the present study is that we did *not* mitigate control–line distortions [74, 75], nor did we execute the full calibration pipeline required to remove systematic phase errors and other coherent offsets [76, 31]. In particular, we did not identify and invert the effective transfer function of the I/Q chain [77, 78], did not compensate mixer skew and LO leakage beyond minimal setup [79], and did not run the complete sequence of frame, detuning, and DRAG–parameter calibrations that are standard when pushing towards coherence–limited performance [16, 28, 80]. These omissions explain the lower–than–target gate fidelities obtained in the end: with leakage already suppressed by design [9, 11], uncorrected cross–quadrature memory and phase miscalibration became the dominant contributors to the error per gate [70, 69]. The bandwidth and sampling budgets of the electronics further constrained the usable harmonic content for FAST–DRAG and the effective derivative order for HD–DRAG, limiting the achievable notch sharpness without reintroducing high–frequency power [66, 81].

Methodologically, the results validate the premise that *analytic* spectrum shaping provides pulse families

that are both interpretable in the frequency domain and tunable in experiment [16, 26, 66, 11]. Stop–band endpoints and weights map directly to notches [81, 82]; the overall amplitude fixes the rotation area [20, 42]; and a small number of envelope parameters control the balance between leakage suppression and Stark compensation [28, 9]. Because the design is not bespoke to a particular transfer function, it is portable across devices and electronics [21, 12]. The observed performance plateau thus reflects correctable hardware effects rather than a limit of the analytic construction itself [77, 78].

The immediate next steps are to close the calibration loop that was intentionally left open here. First, characterize and predistort the control path using qubit–as–sensor protocols to extract the complex transfer function [77, 79], followed by FIR/IIR deconvolution on both quadratures together with static compensation of mixer skew and LO leakage [79]. Second, run the full phase–calibration pipeline, including fine tuning of the DRAG coefficient for AC–Stark compensation [16, 28], closed–loop detuning optimization, and robust virtual–Z frame management [80], all embedded in restless calibration to track slow drift [76, 31]. Third, augment benchmarking with unitarity and leakage–robust variants [70, 55, 69, 32] to separate coherent, incoherent, and seepage contributions within the same automated workflow. On the control–design side, extend the FAST–DRAG formulation to multi–interval notch sets that include device–specific spectator and crosstalk frequencies [81, 82, 9], enabling concurrent single–qubit operations in crowded spectra, and co–design the target spectra with measured electronics bandwidth to avoid aliasing of suppressed power. Finally, port the formalism to flux–modulated and microwave–activated two–qubit gates, where frequency–domain constraints are even more stringent and analytic notching can act directly on entangling interactions [50, 29, 42]. Broader control–theoretic insights from open–system dynamics [15, 14] can guide the extension of these analytic approaches toward noise-adaptive and feedback-assisted operation.

As a closing remark, this work established a clear causal chain from pulse spectrum to leakage suppression on hardware by pairing analytically solvable, frequency–selective envelopes with a reproducible experimental stack. The lower–than–desired final fidelities are attributable not to the analytic method but to deliberately unmitigated line distortions and incomplete phase calibration. Implementing transfer–function–aware predistortion and the full calibration pipeline is expected to unlock the headroom revealed here, turning the demonstrated low–leakage gates into fast, high–fidelity primitives that operate at, or very near, the coherence limit.

Appendix A

Gate synthesis and spectral analysis

This appendix collects the NumPy-only helper routines used throughout the thesis to synthesize microwave single-qubit gates and to compute their baseband amplitude spectra. The implementation follows the standard DRAG relation for the quadrature channel, with the qubit anharmonicity α treated as an *angular* frequency (rad/s), and adopts the higher-derivative DRAG (HD-DRAG) construction and the FAST-DRAG cosine-series ansatz developed in recent work. In the rotating frame, and under the rotating-wave approximation (RWA), the control Hamiltonian reads

$$H_{\mathrm{R}}(t) = \frac{\hbar}{2}\left[\Omega_{\mathrm{I}}(t)\,\sigma_{\mathrm{x}} + \Omega_{\mathrm{Q}}(t)\,\sigma_{\mathrm{y}}\right], \qquad \vartheta \approx \int_0^{\mathrm{T}} \Omega_{\mathrm{I}}(t)\,dt,$$

so that the rotation angle ϑ is set by the in-phase area. DRAG adds an off-quadrature term proportional to $\dot{\Omega}_{\mathrm{I}}$ to cancel diabatic couplings to the nearest leakage level $|f\rangle$,

$$\Omega_{\mathrm{Q}}(t) = -\frac{\beta}{\alpha}\frac{d}{dt}\Omega_{\mathrm{I}}(t),$$

with β the DRAG coefficient and α the anharmonicity, here taken negative for a transmon. This first-order relation and its role in the multi-transition DRAG framework are reviewed in [**?**] (see also the spectral and superadiabatic interpretations there).

HD-DRAG extends the in-phase envelope by even derivatives and the quadrature by odd derivatives so as to create higher-order spectral zeros at the leakage frequency. The special case used in this thesis is

$$\Omega_{\mathrm{I}}(t) = g(t) + \beta_2\, g^{(3)}(t), \tag{A.1}$$

$$\Omega_{\mathrm{Q}}(t) = -\frac{\beta}{\alpha}\frac{d}{dt}\Omega_{\mathrm{I}}(t) = -\frac{\beta}{\alpha}\left[\dot{g}(t) + \beta_2\, g^{(3)}(t)\right], \tag{A.2}$$

which yields a factorized baseband spectrum $\hat{\Omega}_{\mathrm{IQ}}(f) = \left[1 - \beta_2(2\pi f)^2\right]\left[1 - 2\pi\beta f/\alpha\right]\hat{g}(f)$. Choosing $\beta = 1$ and $\beta_2 = \alpha^{-2}$ produces a strong spectral notch at $f = |\alpha|/(2\pi)$ while keeping the β– and β_2–tunings decoupled, as advocated in [11].

FAST-DRAG uses a short cosine series,

$$g(t) = \sum_{n=1}^{N} c_n\left[1 - \cos(2\pi n t/T)\right],$$

with coefficients c_n obtained from a small KKT system that minimizes the spectral energy of g in prescribed baseband stop-bands (e.g. a window around $|\alpha|/(2\pi)$ and a high-frequency cutoff) subject to the area constraint $\sum_n c_n T = \vartheta$. We then set $\Omega_Q = -(\beta/\alpha)\,\dot{\Omega}_I$. The construction and calibration strategy are detailed in [11].

Throughout the code below we: (i) sample on a uniform grid $t_k = k\,dt$ over $k = 0, \ldots, N-1$ with duration $T = N\,dt$; (ii) normalize so that $\int \Omega_I(t)\,dt = \vartheta/\pi$ to match the convention used in our Rabi calibrations; and (iii) apply a final IQ rotation by the drive phase ϕ, i.e. $[I', Q']^\top = R_z(\phi)[I, Q]^\top$, so the same primitive realizes X– and Y–axis rotations.

Code listing

```
"""
Microwave single-qubit X/Y rotations with Gaussian / DRAG / HD-DRAG / FAST-DRAG envelopes.

Conventions:
- Time in seconds, frequency in Hz (alpha is angular frequency, rad/s).
- I(t) dt is scaled to pi; Q is scaled coherently.
- Phase rotates the IQ pair: [I',Q'] = Rz(phase)[I,Q].

References:
- DRAG review: Theis et al., 2018.
- FAST-DRAG and HD-DRAG: Hyypp¨a et al., 2024.
"""

import numpy as np

# ---------------------------------------------------
# Helpers
# ---------------------------------------------------

def _time_grid(duration: float, dt: float) -> np.ndarray:
    """Uniform time grid [0, T) with step dt (seconds)."""
    n = int(np.round(duration / dt))
    n = max(n, 2)
    return np.arange(n) * dt

def _gaussian(t: np.ndarray, duration: float, sigma: float) -> np.ndarray:
    """Centered Gaussian envelope on [0, T) with width sigma."""
    t0 = duration / 2.0
    g = np.exp(-0.5 * ((t - t0) / sigma) ** 2)
    # Subtract edge value to make it ~0 at boundaries (common in DRAG practice)
    edge = np.exp(-0.5 * ((-t0) / sigma) ** 2)
    return g - edge

def _nth_derivative(y: np.ndarray, dt: float, n: int) -> np.ndarray:
    """n-th time derivative via repeated central differences (2nd-order)."""
    d = y.copy()
    for _ in range(n):
        d = np.gradient(d, dt, edge_order=2)
    return d
```

```python
def _phase_rotate(I: np.ndarray, Q: np.ndarray, phase: float) -> tuple[np.ndarray, np.ndarray]:
    """Rotate I/Q by 'phase' (radians): equivalent to drive phase shift."""
    c, s = np.cos(phase), np.sin(phase)
    I_rot = c * I - s * Q
    Q_rot = s * I + c * Q
    return I_rot, Q_rot

def _normalize_area(I: np.ndarray, Q: np.ndarray, dt: float, target_area: float) -> tuple[np.ndarray, np.ndarr
    """Scale envelopes so that integral of I equals 'target_area'."""
    area = np.sum(I) * dt
    if area == 0:
        return I, Q
    s = target_area / area
    return s * I, s * Q

# ---------------------------------------------------
# 1) Gaussian-I only (no Q)
# ---------------------------------------------------

def gaussian_only_gate(
    theta: float,
    phase: float,
    *,
    duration: float,
    dt: float,
    num_sigmas: float = 4.0,
) -> tuple[np.ndarray, np.ndarray, np.ndarray]:
    """
    Microwave Rotation(theta, phase) using a Gaussian I envelope and no Q (i.e. Q = 0).

    Parameters
    ----------
    theta : float
        Rotation angle (radians) for an on-resonance X/Y rotation; area(I)  theta.
    phase : float
        Rotation axis phase in the XY plane (radians). Applied as IQ rotation.
    duration : float
        Total pulse length (seconds).
    dt : float
        Sampling interval (seconds).
    num_sigmas : float
        Truncation setting for Gaussian: T/2 = num_sigmas * sigma.

    Returns
    --------
    I, Q, t : np.ndarrays
        Time-domain I/Q envelopes (arrays) and the time grid t (seconds).
    """
    t = _time_grid(duration, dt)
    sigma = duration / (2.0 * num_sigmas)
    I = _gaussian(t, duration, sigma)
```

```
    Q = np.zeros_like(I)

    # Normalize area(I) to theta/pi (convention matching many Rabi calibrations)
    I, Q = _normalize_area(I, Q, dt, target_area=theta / np.pi)

    # Apply drive phase
    I, Q = _phase_rotate(I, Q, phase)
    return I, Q, t

def gaussian_drag_gate(
    theta: float,
    phase: float,
    *,
    duration: float,
    dt: float,
    alpha: float,
    num_sigmas: float = 4.0,
    beta: float = 1.0,
) -> tuple[np.ndarray, np.ndarray, np.ndarray]:
    """
    Microwave Rotation(theta, phase) using a Gaussian I envelope and no Q (i.e. Q = 0).

    Parameters
    ----------
    theta : float
        Rotation angle (radians) for an on-resonance X/Y rotation; area(I)  theta.
    phase : float
        Rotation axis phase in the XY plane (radians). Applied as IQ rotation.
    duration : float
        Total pulse length (seconds).
    dt : float
        Sampling interval (seconds).
    num_sigmas : float
        Truncation setting for Gaussian: T/2 = num_sigmas * sigma.

    Returns
    --------
    I, Q, t : np.ndarrays
        Time-domain I/Q envelopes (arrays) and the time grid t (seconds).
    """
    t = _time_grid(duration, dt)
    sigma = duration / (2.0 * num_sigmas)
    I = _gaussian(t, duration, sigma)

    dI = _nth_derivative(I, dt, 1)

    Q = - (beta / alpha) * dI

    # Normalize area(I) to theta/pi (convention matching many Rabi calibrations)
    I, Q = _normalize_area(I, Q, dt, target_area=theta / np.pi)

    # Apply drive phase
```

```
    I, Q = _phase_rotate(I, Q, phase)
    return I, Q, t

# ---------------------------------------------------
# 2) Gaussian-I + HD-DRAG-Q
# ---------------------------------------------------

def gaussian_hd_drag_gate(
    theta: float,
    phase: float,
    *,
    duration: float,
    dt: float,
    alpha: float,
    num_sigmas: float = 4.0,
    beta: float = 1.0,
    beta2: float | None = None,
):
    """
    Higher-derivative DRAG following Hyypp¨a et al. (2024), Eqs. (6)-(7):
        I(t) = g(t) + beta2 * d^2 g/dt^2,
        Q(t) = -(beta/alpha) * dI/dt  = -(beta/alpha) * [dg/dt + beta2 * d^3 g/dt^3].

    Here g(t) is taken to be a (truncated) Gaussian on [0, duration).
    The envelopes are finally scaled so that I(t) dt = theta/pi, and then
    rotated by 'phase' in the IQ-plane.

    Parameters
    ----------
    theta : target rotation angle (rad); area(I) = theta/pi after normalization.
    phase : axis angle in the XY plane (rad).
    duration : pulse length (s).
    dt : sample spacing (s).
    alpha : anharmonicity (rad/s), e.g. alpha = -2*pi*200e6.
    num_sigmas : Gaussian truncation parameter: duration/2 = num_sigmas * sigma.
    beta : DRAG coefficient (dimensionless).
    beta2 : HD-DRAG in-phase coefficient; if None, set to 1/alpha**2 (s^2).

    Returns
    --------
    I, Q, t : np.ndarray
        Sampled in-phase and quadrature envelopes and the time grid.
    """
    if alpha == 0.0:
        raise ValueError("alpha must be non-zero (rad/s).")
    if beta2 is None:
        beta2 = 1.0 / (alpha ** 2)

    # time grid
    t = _time_grid(duration, dt)

    # base Gaussian g(t)
```

```
    sigma = duration / (2.0 * num_sigmas)
    g = _gaussian(t, duration, sigma)

    # required derivatives of g
    dg = _nth_derivative(g, dt, 1)
    d2g = _nth_derivative(g, dt, 2)
    d3g = _nth_derivative(g, dt, 3)

    # HD{DRAG construction
    I = g + beta2 * d2g

    # enforce the DRAG relation exactly: Q = -(beta/alpha) * dI/dt
    dI = _nth_derivative(I, dt, 1)  # equals dg + beta2 * d3g numerically
    Q = - (beta / alpha) * dI

    # normalize area of I to theta/pi and scale Q coherently
    I, Q = _normalize_area(I, Q, dt, target_area=theta / np.pi)

    # IQ phase rotation
    I, Q = _phase_rotate(I, Q, phase)
    return I, Q, t

# ---------------------------------------------------
# 3) FAST-DRAG (Fourier-ansatz I + DRAG Q)
# ---------------------------------------------------

def _fast_basis_gn(t: np.ndarray, duration: float, n: int) -> np.ndarray:
    """Basis g_n(t) = 1 - cos(2 n t / T) used by FAST (continuous with zero slope at edges)."""
    return 1.0 - np.cos(2.0 * np.pi * n * t / duration)

def _interval_energy_matrix(
    Ghat: np.ndarray,
    freqs: np.ndarray,
    intervals: list[tuple[float, float]],
    weights: list[float],
) -> np.ndarray:
    """
    Compute H_{nm} = sum_j w_j * _{fl_j}^{fh_j} ghat_n(f) ghat_m*(f) df (numerically).
    Uses rfft (non-negative freqs), doubles non-DC contributions to account for symmetry.
    """
    N = Ghat.shape[0]
    H = np.zeros((N, N), dtype=float)
    df = (freqs[1] - freqs[0]) if len(freqs) > 1 else 1.0

    for (fl, fh), w in zip(intervals, weights):
        mask = (freqs >= max(0.0, fl)) & (freqs <= fh)
        # Contribution matrix over masked frequencies
        # Note: Ghat has shape (N, F). We select columns in 'mask' and integrate.
        Gh = Ghat[:, mask]  # shape (N, K)
        # _k G_n(fk) * conj(G_m(fk)) * df
        block = (Gh @ Gh.conj().T).real * df
        # Double for negative frequencies (except DC at f=0)
```

```
        if np.any(mask) and freqs[mask][0] > 0:
            block *= 2.0
        H += w * block
    return H

def _fast_solve_coeffs(
    t: np.ndarray,
    duration: float,
    dt: float,
    N_terms: int,
    intervals: list[tuple[float, float]],
    weights: list[float],
    area_target: float = 1.0,
) -> tuple[np.ndarray, list[np.ndarray]]:
    """
    Solve KKT system for coefficients 'c' minimizing summed spectral energy over intervals
    subject to ∫I(t) dt = area_target. Basis I(t) = sum_{n=1..N} c_n g_n(t),
    g_n(t) = 1 - cos(2 n t / T).
    """
    # Build basis in time
    G_time = np.stack([_fast_basis_gn(t, duration, n) for n in range(1, N_terms + 1)], axis=0)  # (N, T)

    # FFT (non-negative freqs)
    Ghat = np.fft.rfft(G_time, axis=1)
    freqs = np.fft.rfftfreq(t.size, d=dt)  # Hz

    # Interval energy matrix
    H = _interval_energy_matrix(Ghat, freqs, intervals, weights)

    # Linear constraint: b_n = ∫ g_n dt
    b = (G_time.sum(axis=1) * dt)  # shape (N,)

    # KKT system:
    # [ H+H^T  -b ] [ c ] = [ 0 ]
    # [  b^T    0 ] [ λ ]   [ area_target ]
    K11 = H + H.T  # symmetric
    K12 = -b[:, None]
    K21 = b[None, :]
    K22 = np.zeros((1, 1), dtype=float)
    KKT = np.block([[K11, K12], [K21, K22]])

    rhs = np.zeros(N_terms + 1, dtype=float)
    rhs[-1] = area_target

    # Regularize if near-singular
    eps = 1e-12
    KKT[:N_terms, :N_terms] += eps * np.eye(N_terms)

    sol = np.linalg.solve(KKT, rhs)
    c = sol[:N_terms]  # coefficients

    # Return also the basis functions to avoid recomputation
```

```
    basis = [G_time[k, :] for k in range(N_terms)]
    return c, basis

def fast_drag_gate(
    theta: float,
    phase: float,
    *,
    duration: float,
    dt: float,
    alpha: float,
    N_terms: int = 4,
    suppress_intervals: list[tuple[float, float]] | None = None,
    suppress_weights: list[float] | None = None,
    cutoff: float | None = None,
    beta: float = 1.0,
) -> tuple[np.ndarray, np.ndarray, np.ndarray]:
    """
    FAST-DRAG: Fourier-ansatz I (minimize spectral energy over specified intervals) + DRAG Q.

    I(t) = sum_{n=1..N} c_n [1 - cos(2 n t / T)]  (coeffs c from convex KKT)
    Q(t) = - (beta / alpha) * dI/dt

    Parameters
    ----------
    theta : float
        Target rotation angle (radians); area(I) is scaled to theta/pi.
    phase : float
        Rotation axis phase (radians).
    duration : float
        Pulse duration (seconds).
    dt : float
        Sample time (seconds).
    alpha : float
        Anharmonicity (rad/s), e.g. alpha = -2*np.pi*212e6.
    N_terms : int
        Number of Fourier-ansatz basis functions.
    suppress_intervals : list of (fl, fh) in Hz
        Baseband frequency intervals to suppress (e.g. around |alpha|/(2)).
        If None, defaults to [(max(0, |alpha|/2 - 10e6), |alpha|/2 + 10e6)].
    suppress_weights : list of floats
        Weights per interval; if None, all ones.
    cutoff : float or None
        If provided, also suppress everything above 'cutoff' (Hz): add interval [cutoff, f_Nyquist].
    beta : float
        DRAG coefficient for Q (beta1.0 for leakage suppression; beta0.5 for AC-Stark cancellation).

    Returns
    --------
    I, Q, t : np.ndarrays
    """
    if alpha == 0.0:
        raise ValueError("alpha must be non-zero (rad/s).")
```

```
    t = _time_grid(duration, dt)

    # Default suppression: a band around |alpha|/(2), e.g. ±10 MHz
    if suppress_intervals is None:
        f_alpha = abs(alpha) / (2.0 * np.pi)
        suppress_intervals = [(max(0.0, f_alpha - 10e6), f_alpha + 10e6)]
    if suppress_weights is None:
        suppress_weights = [1.0] * len(suppress_intervals)

    # Optional high-frequency cutoff suppression
    if cutoff is not None:
        f_nyq = 0.5 / dt
        if cutoff < f_nyq:
            suppress_intervals = list(suppress_intervals) + [(cutoff, f_nyq)]
            suppress_weights = list(suppress_weights) + [1.0]

    # Solve for coefficients c_n under area constraint = 1.0 (we'll rescale to theta/pi later)
    c, basis = _fast_solve_coeffs(
        t=t,
        duration=duration,
        dt=dt,
        N_terms=N_terms,
        intervals=suppress_intervals,
        weights=suppress_weights,
        area_target=1.0,
    )

    # Build I(t)
    I = np.zeros_like(t)
    for cn, gn in zip(c, basis):
        I += cn * gn

    # DRAG Q from dI/dt
    dI = _nth_derivative(I, dt, 1)
    Q = - (beta / alpha) * dI

    # Scale area(I) to theta/pi
    I, Q = _normalize_area(I, Q, dt, target_area=theta / np.pi)

    # Phase rotation
    I, Q = _phase_rotate(I, Q, phase)
    return I, Q, t

def complex_envelope(I, Q):
    """_IQ(t) = I(t) - i Q(t)"""
    return I - 1j * Q

def amplitude_spectrum(I, Q, t, freqs):
    """
    Compute |  IQ(t) e^{-i2 f t} dt | with IQ(t)=I(t)-iQ(t).
    'freqs' is an array of baseband frequencies [Hz] (can include ±).
```

```
"""
dt = t[1]-t[O]
omegaIQ = I - 1j*Q
# Explicit quadrature at arbitrary frequencies (smooth, \few peaks")
phase = np.exp(-2j*np.pi*np.outer(freqs, t))  # shape (F, T)
spec = phase @ omegaIQ * dt                   # Riemann sum approximation
return np.abs(spec)
```

Appendix B

Lorentzian fit

Spectroscopy is a foundational primitive in the calibration stack: by sweeping a control parameter (typically the drive frequency) and recording a scalar response (e.g. resonator transmission, qubit excited-state probability, or I/Q quadrature magnitude), one obtains a lineshape from which the transition frequency and linewidth can be inferred. In the weak-drive, Markovian limit, power broadening is negligible and homogeneous dissipation leads to a Lorentzian profile. This appendix documents the Lorentzian estimator used throughout the thesis to extract a best-estimate center frequency (used, for example, as an updated intermediate frequency), a full width at half maximum (FWHM), and a goodness-of-fit statistic from raw scan data.

The model assumed for a single, isolated feature is the four-parameter Lorentzian

$$L(x; A, x_0, \Gamma, B) = \frac{A}{1 + \left(x - x_0\right)/(\Gamma/2)^2} + B, \tag{B.1}$$

where x is the swept variable (typically frequency), $A \in \mathrm{R}$ is the peak amplitude ($A > 0$ for a peak and $A < 0$ for a dip), x_0 is the line center, $\Gamma > 0$ is the full width at half maximum (FWHM), and B is a constant baseline offset. By construction $L(x_0) = A + B$, and the half-maximum points $x_\pm = x_0 \pm \Gamma/2$ satisfy $L(x_\pm) = A/2 + B$. The integrated area of the *baseline-subtracted* line is $\int_{-\infty}^{\infty} L(x) - B \; dx = A\pi\,(\Gamma/2)$, a quantity sometimes used to benchmark drive saturation.

The routine below implements a pragmatic, calibration-friendly fit using the lmfit library. Given vectors (x_i, y_i), it (i) constructs the model (B.1); (ii) infers the likely sign of A by comparing the relative position of the sample mean to the extrema, so that both peaks and dips are handled without a separate branch; (iii) sets physically motivated bounds,

$$\begin{aligned} A &\in [-\Delta_y,\ \Delta_y], & \Delta_y &= \max(y) - \min(y), \\ x_0 &\in [\min(x),\ \max(x)], & & \\ \Gamma &\in \left[\tfrac{1}{100}\Delta_x,\ \Delta_x\right], & \Delta_x &= \max(x) - \min(x), \\ B &\in [\min(y),\ \max(y)], & & \end{aligned}$$

to regularize the optimization; and (iv) invokes a global, derivative-free optimizer (differential evolution) to avoid poor local minima when the initial guesses are crude. From the optimized parameters $\hat{A}, \hat{x}_0, \hat{\Gamma}, \hat{B}$,

the code evaluates the fitted curve $L(x_i; \hat{A}, \hat{x}_0, \hat{\Gamma}, \hat{B})$ and reports the coefficient of determination,

$$R^2 = 1 - \frac{\sum_i \left(y_i - L(x_i; \hat{A}, \hat{x}_0, \hat{\Gamma}, \hat{B})\right)^2}{\sum_i (y_i - \bar{y})^2},$$

as a compact quality metric. For convenience in downstream calibration logic, the function additionally returns a discrete arg-extremum,

$$x^\star = \begin{cases} \arg\max_{x_i} L(x_i; \hat{A}, \hat{x}_0, \hat{\Gamma}, \hat{B}), & \hat{A} > 0, \\ \arg\min_{x_i} L(x_i; \hat{A}, \hat{x}_0, \hat{\Gamma}, \hat{B}), & \hat{A} < 0, \end{cases}$$

which coincides with $\hat{x}_0$ up to sampling resolution for an ideal Lorentzian. In the thesis codebase this value is used as the "fitted IF" (updated drive point) when re-centering subsequent scans.

The estimator is intentionally minimal: it assumes a single, isolated, symmetric feature over a slowly varying baseline with homoscedastic noise. When these assumptions are violated—e.g. overlapping lines, dispersive Fano-like asymmetry, inhomogeneous broadening (Voigt profiles), or baseline drifts—one should (a) precondition the data (baseline removal; restriction to a local window), (b) provide frequency-dependent weights to lmfit if noise is heteroscedastic, or (c) upgrade the model (doublet, Voigt) while retaining the same fitting interface. Within its domain of validity, the routine gives robust point estimates for center and width and a diagnostic R^2 sufficient for automated calibration loops.

Code listing

```
import lmfit
import numpy as np


def lorentzian(x, amplitude, center, width, offset):
    """
    Lorentzian function.

    Parameters
    ----------
    x : array_like
        Input values (typically frequency or time).
    amplitude : float
        Peak height (positive for a peak, negative for a dip).
    center : float
        Center position of the peak (x-coordinate).
    width : float
        Full width at half maximum (FWHM) of the peak.
    offset : float
        Constant baseline offset.

    Returns
    --------
    np.ndarray
        The Lorentzian function evaluated at `x`.
```

```
    """
    return amplitude / (1 + ((x - center) / (0.5 * width)) ** 2) + offset


def lorentzian_fit(y_values, x_values):
    """
    Fit a Lorentzian model to spectroscopy data using non-linear optimization.

    Parameters
    ----------
    Y_VALUES : array_like
        Measured signal values (e.g., I or Q amplitude).
    X_VALUES : array_like
        Independent variable values (e.g., frequency sweep points).

    Returns
    --------
    fitted_if : float
        The frequency (or x-value) at which the Lorentzian reaches its minimum or maximum.
        This is used as the "fitted IF" (intermediate frequency).
    fit_values : np.ndarray
        The fitted Lorentzian curve evaluated at all X_VALUES.
    r_squared : float
        Coefficient of determination (R²) for the fit, indicating quality.

    Notes
    ------
    - Uses the `lmfit` package with the `differential_evolution` global optimization method.
    - Automatically detects whether the peak is a dip or bump by checking amplitude sign.
    - Suitable for resonator or qubit spectroscopy data where a Lorentzian line shape is expected.
    """
    fit_signal = y_values

    # Create model
    mod = lmfit.Model(lorentzian)

    # Estimate initial amplitude sign (dip or peak)
    initial_amp = np.max(fit_signal) - np.min(fit_signal)
    if (np.mean(fit_signal) - np.min(fit_signal)) ** 2 > (np.mean(fit_signal) - np.max(fit_signal)) ** 2:
        initial_amp = -initial_amp

    # Set parameter hints
    mod.set_param_hint("amplitude", value=initial_amp,
                       min=-abs(np.max(fit_signal) - np.min(fit_signal)),
                       max=abs(np.max(fit_signal) - np.min(fit_signal)))
    mod.set_param_hint("center", value=np.mean(x_values),
                       min=np.min(x_values), max=np.max(x_values))
    mod.set_param_hint("width", value=(np.max(x_values) - np.min(x_values)) / 10,
                       min=(np.max(x_values) - np.min(x_values)) / 100,
                       max=(np.max(x_values) - np.min(x_values)))
    mod.set_param_hint("offset", value=np.mean(fit_signal),
                       min=np.min(fit_signal), max=np.max(fit_signal))
```

```
# Fit the model
params = mod.make_params()
fit = mod.fit(data=fit_signal, params=params, x=x_values, method="differential_evolution")
r_squared = fit.rsquared

# Extract parameters
fitted_amplitude = fit.params["amplitude"].value
fitted_center = fit.params["center"].value
fitted_width = fit.params["width"].value
fitted_offset = fit.params["offset"].value
popt = [fitted_amplitude, fitted_center, fitted_width, fitted_offset]

# Determine peak/dip position
if fitted_amplitude < 0:
    fitted_if = x_values[np.argmin(lorentzian(x_values, *popt))]
else:
    fitted_if = x_values[np.argmax(lorentzian(x_values, *popt))]

fit_values = lorentzian(x_values, *popt)
return fitted_if, fit_values, r_squared
```

Bibliography

[1] Riccardo Manenti and Mario Motta. *Quantum Information Science*. Oxford University Press, illustrated edition, 2023.

[2] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, Cambridge, 10th anniversary edition, 2010.

[3] Frank Gaitan. *Quantum Error Correction and Fault Tolerant Quantum Computing*. CRC Press, 2008. Comprehensive exposition of the accuracy threshold theorem and practical fault tolerance.

[4] Daniel Gottesman. An introduction to quantum error correction and fault-tolerant quantum computation. *arXiv:0904.2557*, 2009. Lecture-style review; gives the threshold theorem and conditions for fault tolerance.

[5] R. Barends, J. Kelly, A. Megrant, A. Veitia, D. Sank, E. Jeffrey, T. C. White, J. Mutus, A. G. Fowler, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, C. Neill, P. O'Malley, P. Roushan, A. Vainsencher, J. Wenner, A. N. Korotkov, A. N. Cleland, and John M. Martinis. Logic gates at the surface-code threshold: Superconducting qubits poised for fault-tolerant quantum computing. *arXiv:1402.4848*, 2014. Demonstrates experimentally high fidelities approaching thresholds for fault tolerance in superconducting qubits.

[6] Yuval R. Sanders, Joel J. Wallman, and Barry C. Sanders. Bounding quantum gate error rate based on reported average fidelity. *arXiv:1501.04932*, 2015. Connects experimental gate fidelities to error rates relevant for fault tolerance.

[7] Abhinav Kandala, Kristan Temme, Antonio D. Corcoles, Antonio Mezzacapo, Jerry M. Chow, and Jay M. Gambetta. Extending the computational reach of a noisy superconducting quantum processor. *arXiv:1805.04492*, 2018. Discusses the necessity of very low error rates and error mitigation and the overhead of fault tolerance.

[8] E. Paspalakis, D. Stefanatos, D. Koutromanos, and I. Thanopulos. Notes of the course "quantum control". Lecture notes, Democritus University of Thrace (DUTH), 2025.

[9] Max Werninghaus, Daniel J. Egger, Federico Roy, Shai Machnes, Frank K. Wilhelm, and Stefan Filipp. Leakage reduction in fast superconducting qubit gates via optimal control. *npj Quantum Information*, 7:14, 2021.

[10] Aravind P. Babu, Jani Tuorila, and Tapio Ala-Nissila. State leakage during fast decay and control of a superconducting transmon qubit. *npj Quantum Information*, 7:30, 2021. "Coherent driving of multilevel systems inherently induces excitations out of the qubit subspace . . . short gates increase leakage errors.".

[11] Eric Hyyppä, Antti Vepsäläinen, Miha Papič, Chun Fai Chan, Sinan Inel, Alessandro Landra, Wei Liu, Jürgen Luus, Fabian Marxer, Caspar Ockeloen-Korppi, Sebastian Orbell, Brian Tarasinski, and Johannes Heinsoo. Reducing leakage of single-qubit gates for superconducting quantum processors using analytical control pulse envelopes. *PRX Quantum*, 5:030353, Sep 2024.

[12] Morten Kjaergaard, Matthew E. Schwartz, Jochen Braumüller, Philip Krantz, Joel I.-J. Wang, Simon Gustavsson, and William D. Oliver. Superconducting qubits: Current state of play. *Annual Review of Condensed Matter Physics*, 11:369–395, 2020.

[13] Philip Krantz, Morten Kjaergaard, Fei Yan, Terry P. Orlando, Simon Gustavsson, and William D. Oliver. A quantum engineer's guide to superconducting qubits. *Applied Physics Reviews*, 6(2):021318, 2019.

[14] Heinz-Peter Breuer and Francesco Petruccione. *The Theory of Open Quantum Systems*. Oxford University Press, 2002. Standard reference on amplitude damping, dephasing, master equations and Lindblad formalism.

[15] Howard M. Wiseman and Gerard J. Milburn. *Quantum Measurement and Control*. Cambridge University Press, 2009. Covers continuous measurement, open quantum systems, amplitude damping and dephasing channels, and general decoherence theory.

[16] Felix Motzoi, J. M. Gambetta, Pedro Rebentrost, and Frank K. Wilhelm. Simple pulses for elimination of leakage in weakly nonlinear qubits. *Phys. Rev. Lett.*, 103:110501, 2009.

[17] Z. Chen, J. Kelly, C. Quintana, R. Barends, B. Campbell, Y. Chen, B. Chiaro, A. Dunsworth, A. G. Fowler, E. Jeffrey, et al. Measuring and suppressing quantum state leakage in a superconducting qubit. *Phys. Rev. Lett.*, 116:020501, 2016.

[18] Hidetaka Manabe, Yasunari Suzuki, and Andrew S. Darmawan. Efficient simulation of leakage errors in quantum error correcting codes using tensor network methods. *arXiv preprint*, 2023. arXiv:2308.08186; emphasizes that leakage errors cannot be captured by simple Pauli approximations and must be simulated more carefully.

[19] Boris M. Varbanov, Francesco Battistel, Brian M. Tarasinski, Viacheslav P. Ostroukh, Thomas E. O'Brien, Leonardo DiCarlo, and Barbara M. Terhal. Leakage detection for a transmon-based surface code. *npj Quantum Information*, 6, 2020. Discusses how leakage is "outside of the domain of the qubit stabilizer formalism" and how it can persist across cycles.

[20] Jens Koch, Terri M. Yu, Jay Gambetta, A. A. Houck, David I. Schuster, Johannes Majer, Alexandre Blais, Michel H. Devoret, Steven M. Girvin, and Robert J. Schoelkopf. Charge-insensitive qubit design derived from the Cooper pair box. *Phys. Rev. A*, 76:042319, 2007.

[21] S. E. Rasmussen, K. S. Christensen, S. P. Pedersen, L. B. Kristensen, T. Bækkegaard, N. J. S. Loft, and N. T. Zinner. The superconducting circuit companion — an introduction with worked examples. *PRX Quantum*, 2(4):040204, 2021.

[22] Author et al. Practical methods for efficient analytical control in transmon systems. *Unpublished Technical Report / preprint*, 2023. Uses three-level Hamiltonian truncation in deriving DRAG-type pulses.

[23] R. Bianchetti et al. Control and tomography of a three level superconducting artificial atom. *Physical Review Letters / associated article*, 2010. Demonstrates control in three-level transmon models.

[24] M. Cykiert. Robust control of the transmon qubit. Master's / PhD thesis, 2022. Discusses leakage and three-level modeling for DRAG control.

[25] Yvonne Gao et al. Practical guide for building superconducting quantum devices. *PRX Quantum*, 2:040202, 2023.

[26] Jay M. Gambetta, Felix Motzoi, Seth T. Merkel, and Frank K. Wilhelm. Analytic control methods for high-fidelity unitary operations in a weakly anharmonic oscillator. *Physical Review A*, 83:012308, 2011.

[27] Albert Messiah. *Quantum Mechanics*. North-Holland / Wiley, 1962. Classic text covering adiabatic theorem and perturbation methods that underpin DRAG's expansion in $||\Omega||/|\alpha|$.

[28] J. M. Chow, L. DiCarlo, J. M. Gambetta, F. Motzoi, L. Frunzio, S. M. Girvin, M. H. Devoret, and R. J. Schoelkopf. Optimized driving of superconducting artificial atoms for improved single-qubit gates. *Phys. Rev. A*, 82:040305(R), 2010.

[29] D. A. Rower, L. Ding, H. Zhang, M. Hays, J. An, P. M. Harrington, I. T. Rosen, J. M. Gertler, T. M. Hazard, B. M. Niedzielski, M. E. Schwartz, S. Gustavsson, K. Serniak, J. A. Grover, and W. D. Oliver. Suppressing counter-rotating errors for fast single-qubit gates with circuit qed. *PRX Quantum*, 5(4):040342, 2024. Addresses corrections for counter-rotating (off-resonant) drive errors, relevant when pushing to very fast and strong drives.

[30] Shin Li, Justin Ruths, Tsyr Yan Yu, Haribabu Arthanari, and Gerhard Wagner. Optimal pulse design in quantum control: A unified computational method. *Proceedings of the National Academy of Sciences*, 108(5):1879–1884, 2011.

[31] Nicolas Wittler, Federico Roy, Kevin Pack, Max Werninghaus, Anurag Saha Roy, Daniel J. Egger, Stefan Filipp, Frank K. Wilhelm, and Shai Machnes. Integrated tool set for control, calibration, and characterization of quantum devices applied to superconducting qubits. *Physical Review Applied*, 15:034080, 2021.

[32] T. Chasseur et al. Complete randomized benchmarking protocol accounting for leakage errors. *Physical Review A*, 92:042333, 2015.

[33] Yi-Hsiang Chen and Charles H. Baldwin. Randomized benchmarking with leakage errors. *arXiv preprint arXiv:2502.00154*, 2025.

[34] David M. Pozar. *Microwave Engineering*. Wiley, 4th edition, 2012.

[35] John Choma. Scattering parameters: Concept, theory, and applications. *USC Department of Electrical Engineering Course Notes*, 2002. Technical report on S-parameter theory and measurement.

[36] University of South Florida. S-parameters in decibels. Online lecture notes, 2003. Definition of $20\log_{10}|S_{21}|$ conversion.

[37] F. Caspers. Rf engineering basic concepts: S-parameters. CERN Lecture Notes / PDF, 2012.

[38] Daniel Thomas Sank. *Fast, Accurate State Measurement in Superconducting Qubits*. PhD thesis, University of California, Santa Barbara, 2014.

[39] Steven M. Girvin. Circuit qed: Superconducting qubits coupled to microwave photons. In M. H. Devoret, R. J. Schoelkopf, B. Huard, and L. F. Cugliandolo, editors, *Quantum Machines: Measurement and Control of Engineered Quantum Systems (Les Houches Summer School Proceedings, Vol. 96)*. Oxford University Press, 2014.

[40] Michel H. Devoret. Quantum fluctuations in electrical circuits. In S. Reynaud, E. Giacobino, and J. Zinn-Justin, editors, *Quantum Fluctuations (Les Houches Session LXIII, 1995)*, pages 351–386. Elsevier Science B.V., 1997.

[41] S. E. Rasmussen, K. S. Christensen, S. P. Pedersen, L. B. Kristensen, T. Bækkegaard, N. J. S. Loft, and N. T. Zinner. Superconducting circuit companion — an introduction with worked examples. *PRX Quantum*, 2(4):040204, 2021.

[42] Alexandre Blais, A. L. Grimsmo, S. M. Girvin, and A. Wallraff. Circuit quantum electrodynamics. *Reviews of Modern Physics*, 93(2):025005, 2021.

[43] B. D. Josephson. Possible new effects in superconductive tunnelling. *Physics Letters*, 1(7):251–253, 1962.

[44] Antonio Barone and Gianfranco Patern. *Physics and Applications of the Josephson Effect*. John Wiley & Sons, 1982.

[45] Michael Tinkham. *Introduction to Superconductivity*. Dover Publications, 2nd edition, 2004.

[46] Bruce W. Shore. *Manipulating Quantum Structures Using Laser Pulses*. Cambridge University Press, Cambridge, UK, 2011. Includes bibliographical references and index.

[47] Juan Jos´e Garc´ıa Ripoll. *Quantum Information and Quantum Optics with Superconducting Circuits*. Cambridge University Press, 2022.

[48] John Clarke and Alex I. Braginski. *The SQUID Handbook: Fundamentals and Technology of SQUIDs and SQUID Systems*. Wiley-VCH, 2004.

[49] Rami Barends, J. Kelly, A. Megrant, D. Sank, E. Jeffrey, Y. Chen, Y. Yin, B. Chiaro, J. Mutus, C. Neill, P. O'Malley, P. Roushan, J. Wenner, T. C. White, A. N. Cleland, and John M. Martinis. Coherent josephson qubit suitable for scalable quantum integrated circuits. *Physical Review Letters*, 111(8):080502, 2013.

[50] Yu Chen, C. Neill, P. Roushan, N. Leung, M. Fang, R. Barends, J. Kelly, B. Campbell, Z. Chen, B. Chiaro, A. Dunsworth, E. Jeffrey, A. Megrant, J. Mutus, P. J. J. O'Malley, C. M. Quintana, D. Sank, A. Vainsencher, J. Wenner, T. C. White, M. R. Geller, A. N. Cleland, and John M. Martinis. Qubit architecture with high coherence and fast tunable coupling. *Physical Review Letters*, 113(22):220502, 2014.

[51] J. A. Schreier, A. A. Houck, Jens Koch, D. I. Schuster, B. R. Johnson, J. M. Chow, J. Majer, Alexandre Blais, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf. Suppressing charge noise decoherence in superconducting charge qubits. *Physical Review B*, 77(18):180502, 2008.

[52] A. P. M. Place, L. V. H. Rodgers, P. Mundada, B. M. Smitham, M. Fitzpatrick, Z. Leng, A. Premkumar, J. Bryon, A. Vrajitoarea, S. Sussman, M. K. Y. Fung, D. K. Kim, X. Zhang, L. Frunzio, S. M. Girvin, L. Jiang, M. H. Devoret, and A. A. Houck. New materials platform for superconducting transmon qubits with coherence times exceeding 0.3 milliseconds. *Nature Communications*, 12:1779, 2021.

[53] Chad Rigetti, Jay M. Gambetta, Stefano Poletto, B. L. T. Plourde, J. M. Chow, A. D. Córcoles, J. A. Smolin, S. T. Merkel, J. R. Rozen, G. A. Keefe, M. B. Rothwell, M. B. Ketchen, and M. Steffen. Superconducting qubit in a waveguide cavity with a coherence time approaching 0.1 ms. *Physical Review B*, 86(10):100506, 2012.

[54] Jay Gambetta, Jerry M. Chow, and Matthias Steffen. Building logical qubits in a superconducting quantum computing system. *npj Quantum Information*, 3(2):2, 2017.

[55] Easwar Magesan, Jay M. Gambetta, and Joseph Emerson. Scalable and robust randomized benchmarking of quantum processes. *Physical Review Letters*, 106:180504, 2011.

[56] G. Ithier, E. Collin, P. Joyez, P. J. Meeson, D. Vion, D. Esteve, F. Chiarello, A. Shnirman, Y. Makhlin, J. Schriefl, and G. Schön. Decoherence in a superconducting quantum bit circuit. *Physical Review B*, 72:134519, 2005.

[57] Jonas Bylander, Simon Gustavsson, Fei Yan, Fumiki Yoshihara, Khalil Harrabi, George Fitch, David G. Cory, Yasunobu Nakamura, Jaw-Shen Tsai, and William D. Oliver. Noise spectroscopy through dynamical decoupling with a superconducting flux qubit. *Nature Physics*, 7(7):565–570, 2011.

[58] L. Allen and J. H. Eberly. *Optical Resonance and Two-Level Atoms*. Dover, New York, 1987.

[59] Marlan O. Scully and M. Suhail Zubairy. *Quantum Optics*. Cambridge University Press, Cambridge, 1997.

[60] J. J. Olivero and R. L. Longbothum. Empirical fits to the voigt line width: A brief review. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 17:233–236, 1977.

[61] Ellis E. Whiting. An empirical approximation to the voigt profile. *Journal of Quantitative Spectroscopy and Radiative Transfer*, 8(6):1379–1384, 1968.

[62] M. D. Reed, B. R. Johnson, A. A. Houck, L. DiCarlo, J. M. Chow, D. I. Schuster, L. Frunzio, and R. J. Schoelkopf. Fast reset and suppressing spontaneous emission of a superconducting qubit. *Applied Physics Letters*, 96(20):203110, 2010.

[63] A. A. Houck, D. I. Schuster, J. M. Gambetta, J. A. Schreier, B. R. Johnson, J. M. Chow, J. Majer, L. Frunzio, M. H. Devoret, S. M. Girvin, and R. J. Schoelkopf. Controlling the spontaneous emission of a superconducting transmon qubit. *Physical Review Letters*, 101:080502, 2008.

[64] Easwar Magesan, Jay M. Gambetta, Blake R. Johnson, Colm A. Ryan, Jerry M. Chow, Seth T. Merkel, Marcus P. da Silva, George A. Keefe, Mary Beth Rothwell, Thomas A. Ohki, Mark B. Ketchen, and Matthias Steffen. Efficient measurement of quantum gate fidelity via randomized benchmarking. *Physical Review Letters*, 109:080505, 2012.

[65] Łukasz Cywiński, Roman M. Lutchyn, Chetan P. Nave, and S. Das Sarma. How to enhance dephasing time in superconducting qubits. *Physical Review B*, 77:174509, 2008.

[66] L. S. Theis, F. Motzoi, S. Machnes, and F. K. Wilhelm. Counteracting systems of diabaticities using drag controls: The status after 10 years. *EPL (Europhysics Letters)*, 123(6):60001, 2018.

[67] Götz S. Uhrig. Keeping a quantum bit alive by optimized π-pulse sequences. *Phys. Rev. Lett.*, 98:100504, 2007.

[68] Wen Yang and Ren-Bao Liu. Quantum many-body theory of qubit decoherence in a finite-size spin bath. *Phys. Rev. B*, 84:054301, 2011. Review of dynamical decoupling sequences and filter functions.

[69] Christopher J. Wood and Jay M. Gambetta. Quantification and characterization of leakage errors. *Physical Review A*, 97:032306, 2018.

[70] Joel J. Wallman, Christopher Granade, Robin Harper, and Steven T. Flammia. Estimating the coherence of noise. *New J. Phys.*, 17:113020, 2015.

[71] Christopher Macklin et al. A near–quantum-limited josephson traveling-wave parametric amplifier. *Science*, 350(6258):307–310, 2015.

[72] Amit Roy and Michel Devoret. Introduction to parametric amplification of quantum signals with josephson circuits. *C. R. Physique*, 17:740–755, 2016.

[73] Fei Yan et al. The flux qubit revisited to enhance coherence and reproducibility. *Nat. Commun.*, 7:12964, 2016.

[74] S. Gustavsson, O. Zwier, J. Bylander, F. Yan, F. Yoshihara, Y. Nakamura, T. P. Orlando, and W. D. Oliver. Improving quantum gate fidelities by using a qubit to measure microwave pulse distortions. *Phys. Rev. Lett.*, 110:040502, 2013.

[75] M. Jerger, A. Kulikov, T. Walter, O.-P. Saira, J. Bylander, A. Shnirman, G. S. Paraoanu, A. Wallraff, and A. Fedorov. In situ characterization of qubit control lines: A qubit as a vector network analyzer. *Phys. Rev. Lett.*, 123:150501, 2019.

[76] J. Kelly, R. Barends, B. Campbell, Y. Chen, Z. Chen, B. Chiaro, A. Dunsworth, A. G. Fowler, I.-C. Hoi, E. Jeffrey, A. Megrant, J. Mutus, C. Neill, P. J. J. O'Malley, C. Quintana, P. Roushan, D. Sank, A. Vainsencher, J. Wenner, T. C. White, A. N. Cleland, and J. M. Martinis. Optimal quantum control using randomized benchmarking. *Phys. Rev. Lett.*, 112:240504, 2014.

[77] M. F. Gely et al. In situ characterization of qubit-drive phase distortions. *Phys. Rev. Applied*, 22, 2024.

[78] L. L. Guo et al. Universal scalable characterization and correction of pulse distortions in superconducting qubit lines. *Phys. Rev. Applied*, 2024.

[79] K. R. Chaves et al. Nonlinear signal distortion corrections through quantum sensing. *(preprint / technical report)*, 2021.

[80] D. C. McKay, C. J. Wood, S. Sheldon, J. M. Chow, and J. M. Gambetta. Efficient *z* gates for quantum computing. *Phys. Rev. A*, 96:022330, 2017.

[81] R. Schutjens, F. Abu Dagga, D. J. Egger, and F. K. Wilhelm. Single-qubit gates in frequency-crowded transmon systems. *Phys. Rev. A*, 88:052330, 2013.

[82] L. S. Theis, F. Motzoi, and F. K. Wilhelm. Simultaneous gates in frequency-crowded multilevel systems using fast, robust, analytic control shapes. *Phys. Rev. A*, 93:012324, 2016.