Field-Oriented Control of a PMSM: A Practical Implementation Guide

From Motor Characterization to Modulation

Vedant K. Naik

March 2026

Contents

1 Overview and Philosophy

This document walks through the full pipeline of getting a PMSM spinning under Field-Oriented Control (FOC): starting from bench characterization of the motor's electrical parameters, building up the mathematical control layer in the d-q frame, and then translating voltage commands into physical gate duties through the modulation layer. The Hall sensor-specific caveats are covered at the end, which may not be relevant to all readers.

The goal is a single coherent reference that bridges the "why" and the "how" at each step, so that the math and the implementation never feel disconnected.

Note

For background, the machine used for testing throughout this document is a 5 kW radial flux outrunner hub BLDC motor used by the MSU Solar Racing Team. Whether to treat a BLDC and a PMSM as equivalent is a debate we will not get into here. The practical distinction between the two is largely one of winding topology: a BLDC typically uses concentrated windings that produce a trapezoidal back-EMF, while a PMSM uses sinusoidally distributed windings that produce a sinusoidal back-EMF. That difference matters for NVH and torque ripple analysis, neither of which is the focus of this document. For the purposes of control design, FOC is widely regarded as a superior method for both machine types precisely because it provides decoupled control over the torque-producing component (i_q) and the pseudo-excitation field component (i_d), regardless of the exact back-EMF waveform shape. The framework developed here applies to both.

2 Motor Characterization

Before writing a single line of control code, we need to largely understand three numbers that describe the motor electrically: stator resistance R_s, phase inductance L_s, and permanent magnet flux linkage ψ. Everything downstream, controller gains, torque estimates, voltage feed-forward, depends on these. Also, for a lot of implementations, ignoring the feed-forward term won't be a big deal since the idea is not to always achieve zero steady-state error.

2.1 What We Are Actually Measuring

A three-phase PMSM has three stator windings. When we put probes across two terminals (say A and B), we are exciting two windings in series. So every measured "series" quantity gets halved to recover the per-phase value. Keep that halving in mind throughout.

2.2 DC Resistance Test

Procedure. Connect a DC supply across terminals A and B. Sweep through several voltage set-points (e.g. 0.5 V, 1 V, 2 V, 3 V) and record the resulting steady-state current at each point.

Why it works. At DC, inductance plays no role (no dI/dt). Ohm's law applies directly:

$$V = R_{\text{series}} \cdot I. \tag{1}$$

Fit a line through the (I, V) cloud using least squares. The slope is R_{series}. In practice, you may see small elliptical clusters of points at each set-point rather than perfect dots. This is because the supply has some 60/50 Hz ripple and the winding inductance introduces a small phase lag, spreading the instantaneous (i, v) samples. The linear fit is still excellent ($R^2 > 0.999$) because the effect is largely symmetric and averages out.

Result. From our measurements:

$$R_{\text{series}} = 386.94\,\text{m}\Omega \quad \Longrightarrow \quad R_s = \frac{R_{\text{series}}}{2} = 193.47\,\text{m}\Omega. \tag{2}$$

2.3 Locked-Rotor AC Impedance Test

Procedure. Mechanically lock the rotor (so no back-EMF is generated). Apply a sinusoidal voltage at 60 Hz across terminals A and B using a variac with an isolation transformer. Capture the voltage and current time-domain waveforms on an oscilloscope. Run an FFT on both to extract the fundamental phasors V_{60} and I_{60}.

Why we lock the rotor. If the rotor spins, it generates a back-EMF that adds to the measured voltage, corrupting the impedance estimate. Locking it ensures $V = Z \cdot I$ with no extra EMF term.

Why 60 Hz. It is a conveniently available frequency from the mains via a variac, and the resulting inductive reactance (ωL) is large enough to measure accurately with standard lab equipment. At DC, L contributes nothing; at 60 Hz it contributes a measurable reactive component.

Math. The complex impedance between the two terminals at frequency $\omega = 2\pi \times 60$ is:

$$Z(\omega) = \frac{V_{60}}{I_{60}} = R_{\text{series}} + j\omega L_{\text{series}}. \tag{3}$$

The real part should match R_s (it does, to within measurement uncertainty, a good cross-check). The imaginary part gives inductance:

$$L_{\text{series}} = \frac{\Im\{Z\}}{\omega} = \frac{0.332}{2\pi \times 60} = 0.88\,\text{mH}. \tag{4}$$

The Lissajous plot of V vs. I will be an ellipse tilted to the right, confirming the inductive phase lag. A collapsed ellipse (straight line) would mean purely resistive; a perfect circle would mean purely inductive. The THD of the current waveform was $< 3\%$, confirming the test setup is clean.

Result.

$$L_s = \frac{L_{\text{series}}}{2} = 0.44\,\text{mH}. \tag{5}$$

Note

In principle, this measurement should be carried out for multiple locked rotor positions and graphed as a function of rotor position. The resulting periodic variation reveals machine saliency: the maximum inductance corresponds to the q-axis, while the minimum corresponds to the d-axis (magnet-aligned).
For a surface-mounted PMSM, this distinction is typically negligible since $L_d \approx L_q$, as the permeabilities of the airgap and magnet are similar.

2.4 Back-EMF Test for Flux Linkage

Procedure. With the motor terminals open-circuit (no current flows), manually spin the rotor at a few known speeds. Record the line-to-line open-circuit voltage waveform. Extract the fundamental frequency f_e via FFT to find the electrical speed $\omega_e = 2\pi f_e$, and extract the peak phase voltage $V_{\phi,\text{pk}}$.

Why it works. When no current flows ($i_d = i_q = 0$), the motor's q-axis voltage equation reduces to:

$$v_q = \omega_e \psi. \tag{6}$$

In other words, the back-EMF is directly proportional to speed, with ψ as the proportionality constant. Given the peak phase voltage and electrical speed:

$$\psi = \frac{V_{\phi,\text{pk}}}{\omega_e}. \tag{7}$$

Note on the factor of 2. Some formulations write $\psi = V_{\phi,\text{pk}}/(2\omega_e)$ depending on whether ψ is defined as the peak or RMS-equivalent flux linkage and how the dq model is normalized. The key is consistency with the torque equation used later. The paper uses the convention:

$$\psi = \frac{V_{\phi,\text{pk}}}{2\omega_e}, \tag{8}$$

which corresponds to treating $V_{\phi,\text{pk}}$ as the line-to-line peak, effectively halving to get per-phase. Verify this against your own FOC convention.

Linearity check. Repeating the measurement at three different speeds and plotting $V_{\phi,\text{pk}}$ vs. ω_e should give a straight line through the origin. Our data gives $R^2 = 0.998$, confirming consistent magnet flux over the tested speed range.

Result.

$$\psi = 98.05\,\text{mWb} \pm 3.25\,\text{mWb}. \tag{9}$$

Note

Hand-spinning the machine gave us decently uniform values of the voltage and current sinusoids, due to the fact that we had a 16-pole pair machine, meaning even a single mechanical rotation would result in 16 electrical rotations, giving clean data. Still, if the machine you are using has a small number of pole pairs, it might be worth having a more sophisticated controlled setup to spin and induce BEMF.

2.5 Parameter Summary

Table 1: Characterized PMSM Electrical Parameters

Parameter	Symbol	Value
Stator resistance	R_s	$193.47\,\mathrm{m}\Omega$
Phase inductance	L_s	$0.44\,\mathrm{mH}$
Flux linkage	ψ	$98.05\,\mathrm{mWb}$
Number of pole pairs	p	16

With R_s, L_s, and ψ in hand, we can now design the control system from first principles.

3 The PMSM Model in the d-q Frame

Before designing controllers, we need the governing equations in the rotating reference frame where control is easy.

3.1 Why the d-q Frame

In the three-phase (abc) frame, the motor voltages and currents are sinusoids rotating at electrical frequency ω_e. Designing a PI controller to track a sinusoidal reference is painful integral windup, limited bandwidth, and phase error all become problems.

The Park transform rotates the abc frame into a frame that is *locked to the rotor*. In this rotor-flux (d-q) frame, the steady-state currents are *DC values*. A PI controller that needs to track a DC reference is straightforward and well-understood. So, in a sense, you can think of using the nice geometrical properties of the machine we have, transformations to realize the 3-phase sine-wave tracking problem into a DC-reference tracking problem.

3.2 Voltage Equations

In the d-q frame, the stator voltage equations are:

$$v_d = R_s i_d + L_d \frac{di_d}{dt} - \omega_e L_q i_q, \tag{10}$$

$$v_q = R_s i_q + L_q \frac{di_q}{dt} + \omega_e L_d i_d + \omega_e \psi. \tag{11}$$

The terms $-\omega_e L_q i_q$ and $\omega_e L_d i_d + \omega_e \psi$ are the *cross-coupling* (or back-EMF) terms. They couple the d and q axes together. At high speed or high current, these terms dominate and a naive single-axis PI loop will fight against them constantly.

For a surface-mounted PMSM (SPMSM), magnetic saliency is negligible: $L_d \approx L_q \triangleq L_s$.

3.3 Torque Equation

Electromagnetic torque is:

$$\tau = \frac{3p}{2}\left(\psi\, i_q + (L_d - L_q)\, i_d i_q\right). \tag{12}$$

For an SPMSM, $(L_d - L_q) \approx 0$, so reluctance torque vanishes:

$$\tau = \frac{3p}{2}\,\psi\, i_q = K_t\, i_q, \quad K_t = \frac{3p\psi}{2}. \tag{13}$$

This is the central insight of FOC: **torque is linearly proportional to** i_q. Controlling torque reduces to controlling a single current component. The d-axis current i_d is set to zero (Maximum Torque Per Ampere, MTPA) unless field weakening is needed.

4 Control Layer: From References to Voltage Commands

4.1 The Clarke Transform (abc $\rightarrow \alpha\beta$)

The first step converts the three measured phase currents I_a, I_b, I_c into a stationary two-dimensional representation. Using the amplitude-invariant ($\frac{2}{3}$) Clarke transform and dropping the zero-sequence row (balanced three-phase system):

$$\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}\begin{bmatrix} I_a \\ I_b \\ I_c \end{bmatrix}. \tag{14}$$

The $\alpha\beta$ frame is still stationary the vectors rotate at ω_e. We need one more rotation to lock into the rotor frame.

4.2 The Park Transform ($\alpha\beta \rightarrow dq$)

Using the rotor electrical angle θ_e:

$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \begin{bmatrix} \cos\theta_e & \sin\theta_e \\ -\sin\theta_e & \cos\theta_e \end{bmatrix}\begin{bmatrix} I_\alpha \\ I_\beta \end{bmatrix}. \tag{15}$$

In implementation, Clarke and Park are almost always fused into one matrix multiply:

$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \frac{2}{3}\begin{bmatrix} \cos\theta_e & \cos\left(\theta_e - \frac{2\pi}{3}\right) & \cos\left(\theta_e + \frac{2\pi}{3}\right) \\ -\sin\theta_e & -\sin\left(\theta_e - \frac{2\pi}{3}\right) & -\sin\left(\theta_e + \frac{2\pi}{3}\right) \end{bmatrix}\begin{bmatrix} I_a \\ I_b \\ I_c \end{bmatrix}. \tag{16}$$

I_d and I_q are now DC in steady state. The PI controllers can track them without error.

4.3 Feedback Linearization (Decoupling)

Looking at (10)–(11), the cross-coupling terms mean that a voltage applied on the d-axis also affects the q-axis current dynamics (and vice versa). We remove this coupling analytically.

Define virtual control inputs:

$$u_d = v_d + \omega_e L_s i_q, \qquad u_q = v_q - \omega_e (L_s i_d + \psi). \tag{17}$$

Substituting into (10)–(11):

$$L_s \dot{i}_d + R_s i_d = u_d, \tag{18}$$

$$L_s \dot{i}_q + R_s i_q = u_q. \tag{19}$$

The two axes are now independent first-order systems. Each can be controlled by a separate PI loop without interference. The cross-coupling terms are cancelled by injecting them as feed-forwards into the voltage commands.

4.4 PI Controller Design

In the Laplace domain, each decoupled plant is a first-order low-pass:

$$G(s) = \frac{1}{L_s s + R_s}. \tag{20}$$

We place a PI controller on each axis:

$$C(s) = K_p + \frac{K_i}{s}. \tag{21}$$

The open-loop transfer function is:

$$L(s) = C(s)G(s) = \frac{K_p s + K_i}{s(L_s s + R_s)}. \tag{22}$$

The closed-loop characteristic polynomial is:

$$L_s s^2 + (R_s + K_p)s + K_i = 0. \tag{23}$$

Matching to the standard second-order form $s^2 + 2\zeta\omega_n s + \omega_n^2 = 0$:

$$K_p = 2\zeta\omega_n L_s - R_s, \tag{24}$$

$$K_i = \omega_n^2 L_s. \tag{25}$$

Design choice: $\zeta = 0.707$ (underdamped, $\approx 5\%$ overshoot) and $\omega_n = 2000\,\text{rad/s}$ (318 Hz bandwidth). How these two numbers are arrived at is discussed in detail in Section 5.

Computed gains:

$$K_p = 2(0.707)(2000)(0.44 \times 10^{-3}) - 0.19347 = 1.05\,\Omega, \tag{26}$$

$$K_i = (2000)^2(0.44 \times 10^{-3}) = 1760\,\Omega/\text{s}. \tag{27}$$

5 Bandwidth, Sampling, and Discrete Implementation

The gains K_p and K_i derived above are only valid choices if $\omega_n = 2000\,\text{rad/s}$ is itself a valid choice. This section explains where that number comes from, why $\zeta = 0.707$ is a reasonable but not unique selection, how motor speed enters the constraint chain, and how the continuous-domain gains translate into what actually executes on the microcontroller.

5.1 Damping Ratio ζ: Transient Shape, Not Speed

The standard second-order step response has the form:

$$y(t) = 1 - \frac{e^{-\zeta\omega_n t}}{\sqrt{1-\zeta^2}}\sin(\omega_d t + \phi)\,, \quad \omega_d = \omega_n\sqrt{1-\zeta^2}, \tag{28}$$

where ω_d is the damped natural frequency. The damping ratio ζ controls the *shape* of the transient, not how fast the system settles (that is set by ω_n):

- $\zeta < 1$: underdamped. The response overshoots and rings. The percent overshoot is:

$$\%OS = 100 \cdot \exp\left(\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}\right). \tag{29}$$

- $\zeta = 1$: critically damped. No overshoot, fastest approach to the setpoint without oscillation.
- $\zeta > 1$: overdamped. No overshoot, but slower than critically damped.

For $\zeta = 0.707$:

$$\%OS = 100 \cdot \exp\left(\frac{-\pi \times 0.707}{\sqrt{1-0.707^2}}\right) = 100 \cdot \exp(-\pi) \approx 4.3\%. \tag{30}$$

So $\zeta = 0.707$ is **underdamped** with approximately 4–5% overshoot. It is a common default in motor current control because it gives a fast rise time at the cost of a small, brief overshoot in current (and therefore torque). Critically damped ($\zeta = 1$) would eliminate the overshoot entirely but slow the rise. The choice is an engineering tradeoff; $\zeta = 0.707$ is conventional, not sacred.

5.2 The Plant's Natural Time Constant: The Floor on ω_n

The decoupled plant after feedback linearization is a first-order RL circuit:

$$G(s) = \frac{1}{L_s s + R_s} = \frac{1/R_s}{\tau_{\text{elec}}\, s + 1}, \quad \tau_{\text{elec}} = \frac{L_s}{R_s}. \tag{31}$$

For our motor:

$$\tau_{\text{elec}} = \frac{0.44 \times 10^{-3}}{0.19347} \approx 2.27\,\text{ms}, \tag{32}$$

which corresponds to a plant pole at:

$$\omega_{\text{plant}} = \frac{1}{\tau_{\text{elec}}} = \frac{R_s}{L_s} \approx 440 \text{ rad/s}. \tag{33}$$

The closed-loop bandwidth ω_n must be substantially larger than ω_{plant}. If it is not, the PI integrator does most of the work, the proportional term barely contributes, and the loop is sluggish. A practical minimum is $\omega_n \geq 5\,\omega_{\text{plant}} \approx 2{,}200\,\text{rad/s}$. This sets the **floor**.

5.3 Motor Maximum Speed: Another Floor on ω_n

The dq currents are DC only in perfect steady state at constant speed. In reality the electrical frequency $\omega_e = p\,\omega_m$ is always present. The current loop must be fast enough to track its reference over the full electrical cycle. As a rule of thumb:

$$\omega_n \;\geq\; 5 \cdot \omega_{e,\max} = 5 \cdot p \cdot \omega_{m,\max}. \tag{34}$$

For our motor with $p = 16$ pole pairs and a mechanical top speed of approximately 150 RPM (wheel speed at racing velocity for a hub motor):

$$\omega_{e,\max} = 16 \times \frac{150 \times 2\pi}{60} \approx 251 \text{ rad/s}. \tag{35}$$

The speed-derived floor is $5 \times 251 \approx 1{,}255\,\text{rad/s}$, which is less binding than the plant-pole floor of $\sim 2{,}200\,\text{rad/s}$ for this particular motor. However, if the motor ran at 1000 RPM with 16 pole pairs, $\omega_{e,\max}$ would be $\sim 1{,}676\,\text{rad/s}$ and the speed floor would be $\sim 8{,}380\,\text{rad/s}$ suddenly the dominant constraint. **High pole count and high speed tighten this floor aggressively.**

5.4 PWM and Sampling Frequency: The Ceiling on ω_n

The microcontroller samples currents and executes the FOC algorithm once per PWM period $T_{\text{sw}} = 1/f_{\text{sw}}$. The sample-and-hold process introduces a half-sample delay on average, and the zero-order hold (ZOH) nature of the PWM output introduces additional phase lag. Combined, the effective delay is approximately T_s (one full sample period in a single-update implementation).

A pure delay e^{-sT_s} contributes phase lag of ωT_s radians at frequency ω. At the bandwidth frequency ω_n, the phase loss is:

$$\Delta\phi_{\text{delay}} = \omega_n \cdot T_s \quad \text{[radians]}. \tag{36}$$

The continuous PI design with $\zeta = 0.707$ has a phase margin of approximately 65°. To keep the phase margin above a safe minimum of $\sim 45°$, the delay-induced loss must stay below $\sim 20° \approx 0.35\,\text{rad}$:

$$\omega_n \cdot T_s \leq 0.35 \quad \Longrightarrow \quad \omega_n \leq \frac{0.35}{T_s} = 0.35 \cdot f_{\text{sw}} \cdot 2\pi \cdot \frac{1}{2\pi} = \frac{0.35}{T_s}. \tag{37}$$

More commonly stated as a frequency ratio: bandwidth should not exceed roughly $f_{\text{sw}}/5$ to $f_{\text{sw}}/10$:

$$\omega_n \;\leq\; \frac{2\pi\, f_{\text{sw}}}{5 \text{ to } 10}. \tag{38}$$

For $f_{\text{sw}} = 10\,\text{kHz}$:

$$\omega_n \leq \frac{2\pi \times 10{,}000}{5} \approx 12{,}566 \text{ rad/s} \quad \text{(liberal)} \qquad \text{or} \qquad \frac{2\pi \times 10{,}000}{10} \approx 6{,}283 \text{ rad/s} \quad \text{(conservative)}. \tag{39}$$

5.5 The Full Constraint Chain

Combining all constraints:

$$\underbrace{\max(5\,\omega_{\text{plant}},\; 5\,\omega_{e,\max})}_{\text{floor}} \;\leq\; \omega_n \;\leq\; \underbrace{\frac{2\pi\, f_{\text{sw}}}{5 \text{ to } 10}}_{\text{ceiling}}. \tag{40}$$

In numbers for this motor at $f_{\text{sw}} = 10\,\text{kHz}$:

$$2{,}200 \text{ rad/s} \;\leq\; \omega_n \;\leq\; 6{,}283 \text{ rad/s}. \tag{41}$$

The choice $\omega_n = 2{,}000\,\text{rad/s}$ sits just below the floor technically slightly tight relative to the plant pole constraint. A more comfortable choice for this system is $\omega_n = 3{,}000$–$4{,}000\,\text{rad/s}$. The value of $2{,}000\,\text{rad/s}$ is used in this document because the back-EMF feed-forward compensates for much of the speed-dependent coupling, relaxing how hard the PI must work, making a slightly lower bandwidth acceptable in practice.

Note

The chain also runs in reverse as a design tool. If you know your motor's top speed and pole count, you can compute the minimum ω_n needed. That minimum ω_n then implies a minimum f_{sw} via (38). Higher f_{sw} means more switching losses and more heat in the power devices so motor speed, pole count, bandwidth, and thermal design are all coupled.

5.6 Computational Delay and Phase Margin Verification

With $\omega_n = 2{,}000\,\text{rad/s}$ and $T_s = 100\,\mu\text{s}$ ($10\,\text{kHz}$ switching):

$$\Delta\phi_{\text{delay}} = \omega_n \cdot T_s = 2{,}000 \times 100 \times 10^{-6} = 0.2 \text{ rad} \approx 11.5^\circ. \tag{42}$$

Starting phase margin of the continuous design: $\approx 65^\circ$. After delay: $65^\circ - 11.5^\circ = 53.5^\circ$. Comfortably above the 45° minimum. At $\omega_n = 6{,}000\,\text{rad/s}$ the delay loss would be 34°, leaving only 31° marginal and potentially unstable with any additional unmodelled lag.

5.7 From Continuous Gains to the Discrete Difference Equation

The continuous PI controller is:

$$u(t) = K_p\, e(t) + K_i \int_0^t e(\tau)\, d\tau. \tag{43}$$

In the microcontroller, time is discrete. The integral must be approximated numerically. The standard choice for motor control is the **Tustin (bilinear) method**, which substitutes:

$$s \;\longleftarrow\; \frac{2}{T_s} \cdot \frac{z-1}{z+1}. \tag{44}$$

This is equivalent to trapezoidal integration averaging the current and previous error sample before accumulating. Applying this to $C(s) = K_p + K_i/s$ and converting to a time-domain difference equation (velocity form):

$$\boxed{u[k] = u[k-1] \;+\; K_p\big(e[k] - e[k-1]\big) \;+\; K_i \frac{T_s}{2}\big(e[k] + e[k-1]\big).} \tag{45}$$

This is the equation that runs in the interrupt service routine every PWM period.

Are the gains the same in continuous and discrete domains?

K_p appears multiplying a *difference* of errors ($e[k] - e[k-1]$), with no T_s factor. So yes the proportional gain computed in the s-domain is used numerically unchanged in the difference equation.

K_i does not appear alone. It always multiplies $T_s/2$. The quantity that lives in hardware and does the accumulating is:

$$K_i^{\text{discrete}} = K_i \cdot \frac{T_s}{2}. \tag{46}$$

For our values:

$$K_i^{\text{discrete}} = 1760 \times \frac{100 \times 10^{-6}}{2} = 0.088\ \Omega. \tag{47}$$

The number stored in the chip is 0.088, not 1760. If you naively substituted the s-domain K_i directly into a discrete accumulator without the $T_s/2$ scaling, the integrator gain would be $2/T_s = 20{,}000\times$ too large and the loop would blow up instantly.

The physical reason is straightforward: K_i has units of Ω/s in continuous time, meaning "rate of voltage buildup per unit error per unit time." In discrete time there is no "per second" there is only "per sample." The sample period T_s converts between these two notions. The factor of 1/2 from Tustin arises because we average the error at two time points (trapezoidal rule) rather than using just one (rectangular rule as in forward Euler).

Why Tustin and not forward Euler?

Forward Euler uses $s \leftarrow (z-1)/T_s$, giving:

$$u[k] = u[k-1] + K_p\big(e[k] - e[k-1]\big) + K_i \cdot T_s \cdot e[k]. \tag{48}$$

The discrete integral gain becomes $K_i \cdot T_s$ (no factor of 1/2, uses only current error). Forward Euler is known to map stable continuous poles to potentially unstable discrete poles when the ratio $\omega_n T_s$ is not small. Tustin, by contrast, maps the imaginary axis of the s-plane to the unit circle of the z-plane exactly it cannot create instability from a stable continuous design due to the discretization alone. For this reason Tustin is the preferred method for digital motor control implementations.

Anti-windup in the velocity form

When v_d^* or v_q^* saturates against the DC bus voltage limit, the integrator must stop accumulating. In the velocity form (45), anti-windup is straightforward: after computing $u[k]$, clamp it to the allowed range and only update the stored value $u[k-1]$ with the clamped result. The integral state is implicitly embedded in $u[k-1]$, so clamping that value stops the windup naturally no separate integrator state variable needs to be frozen.

5.8 Where Motor Speed Appears in the Running Controller

Speed ω_e appears in two operationally distinct places each control cycle.

In the feed-forward decoupling terms (computed fresh each cycle using the current $\hat{\omega}_e$):

$$v_d^* = u_d - \omega_e L_s i_q, \tag{49}$$

$$v_q^* = u_q + \omega_e L_s i_d + \omega_e \psi. \tag{50}$$

At low speed these terms are small and the PI loops operate almost as two independent DC circuits. At high speed the back-EMF term $\omega_e\psi$ in (50) dominates it is the voltage the motor generates against the applied v_q^* and the PI must overcome it. If the feed-forward is accurate, the PI only sees a small residual error and responds easily. If ψ is poorly characterized, the error the PI must correct grows linearly with speed, the integrator has to work harder, and at some speed it saturates. This is why ψ accuracy matters most at high speed.

In the voltage ceiling (base speed): the stator voltage is bounded by:

$$\sqrt{(v_d^*)^2 + (v_q^*)^2} \le \frac{V_{\mathrm{DC}}}{\sqrt{3}}. \tag{51}$$

At rated current with $i_d = 0$, the q-axis voltage is dominated by $v_q^* \approx R_s I_{\max} + \omega_e \psi$. This grows linearly with speed and eventually hits the ceiling. The speed at which this happens is the base speed:

$$\omega_{\mathrm{base}} = \frac{V_{\mathrm{DC}}/\sqrt{3} - R_s I_{\max}}{\psi}. \tag{52}$$

Below ω_{base}: constant torque region, PI has voltage headroom, i_q is thermally limited. Above ω_{base}: field weakening is required negative i_d is commanded to reduce effective flux and lower the back-EMF, allowing higher speed at reduced torque capability.

5.9 Numerical Summary for This Motor

Table 2: Constraint Chain and Discrete Implementation Summary

Quantity	**Symbol / Expression**	**Value**
Switching / sampling frequency	f_{sw}	10 kHz
Sample period	T_s	100 μs
Electrical time constant	$\tau_{\text{elec}} = L_s/R_s$	2.27 ms
Plant pole	$\omega_{\text{plant}} = R_s/L_s$	440 rad/s
Floor from plant pole ($\times 5$)	$5\,\omega_{\text{plant}}$	2,200 rad/s
Ceiling ($f_{\text{sw}}/10$)	$2\pi f_{\text{sw}}/10$	6,283 rad/s
Chosen bandwidth	ω_n	2,000 rad/s
Damping ratio	ζ	0.707
Step overshoot	$\%OS$	$\approx 4.3\%$
Phase loss from 1-period delay	$\omega_n T_s$	$\approx 11.5^\circ$
Remaining phase margin (est.)		$\approx 53^\circ$
Continuous K_p	$2\zeta\omega_n L_s - R_s$	1.05 Ω
Continuous K_i	$\omega_n^2 L_s$	1760 Ω/s
Discrete K_p (unchanged)	same	1.05 Ω
Discrete integral gain	$K_i \cdot T_s/2$	0.088 Ω

5.10 Voltage Command Reconstruction

The PI outputs u_d and u_q are the *virtual* decoupled commands. To get the actual voltage commands, we add back the cross-coupling feed-forward terms (inverting (17)):

$$v_d^* = u_d - \omega_e L_s i_q, \tag{53}$$

$$v_q^* = u_q + \omega_e (L_s i_d + \psi). \tag{54}$$

These are the voltage references in the d-q frame. They now need to be transformed back to phase voltages for the modulator.

5.11 Full Control Loop Summary

The complete control execution sequence each PWM period is:

1. Read I_a, I_b, I_c from current sensors.
2. Read θ_e from position sensor (or estimate it).
3. Apply (16) to get I_d, I_q.

4. Compute PI errors: $e_d = I_d^* - I_d$, $e_q = I_q^* - I_q$.
5. Run PI controllers via (45) to get u_d, u_q.
6. Apply feed-forward (53)–(54) to get v_d^*, v_q^*.
7. Pass v_d^*, v_q^*, θ_e to the modulation layer.
8. Modulation layer outputs gate duties d_A, d_B, d_C.

6 Modulation Layer: From Voltage Commands to Gate Duties

The modulation layer's job is to convert the abstract voltage references (v_d^*, v_q^*) in the rotating frame into three duty cycles $d_A, d_B, d_C \in [0, 1]$ for the high-side switches of the inverter.

6.1 Inverse Park and Inverse Clarke

Reverse the transforms to get back to phase voltages:

$$\begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix} = \begin{bmatrix} \cos\theta_e & -\sin\theta_e \\ \sin\theta_e & \cos\theta_e \end{bmatrix} \begin{bmatrix} v_d^* \\ v_q^* \end{bmatrix}, \tag{55}$$

$$\begin{bmatrix} V_a \\ V_b \\ V_c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix} \begin{bmatrix} V_\alpha \\ V_\beta \end{bmatrix}. \tag{56}$$

The resulting V_a, V_b, V_c are zero-mean sinusoidal references (they represent the desired phase voltages relative to the DC bus midpoint). Their magnitude is:

$$|\mathbf{V}^*| = \sqrt{V_\alpha^2 + V_\beta^2}. \tag{57}$$

6.2 The Space Vector Hexagon and Modulation Regions

The inverter's six switches can produce eight distinct voltage vectors: six active vectors of magnitude $\frac{2}{3}V_{\mathrm{DC}}$, and two zero vectors. These six active vectors form a regular hexagon in the $\alpha\beta$ plane.

Hexagon geometry

Each active vector has length (the hexagon circumradius):

$$R = \frac{2}{3}V_{\mathrm{DC}}. \tag{58}$$

The largest circle that fits entirely inside the hexagon (the inscribed circle, radius = apothem) has radius:

$$r = R\cos 30^\circ = \frac{2}{3}V_{\mathrm{DC}} \cdot \frac{\sqrt{3}}{2} = \frac{V_{\mathrm{DC}}}{\sqrt{3}} \approx 0.577\, V_{\mathrm{DC}}. \tag{59}$$

Region 1 Sinusoidal PWM (no injection)

If the reference vector stays inside the inscribed circle at all times, the three sinusoidal phase references never saturate:

$$|\mathbf{V}^*| \leq \frac{V_{\mathrm{DC}}}{\sqrt{3}} \quad \Longrightarrow \quad \text{linear (SPWM)}. \tag{60}$$

Region 2 Min-Max Injection (≈ SVPWM)

We can push the linear modulation range out to the hexagon vertex by injecting a common-mode (zero-sequence) offset. The offset is invisible to the motor because it only sees *line-to-line* voltages. The optimal zero-sequence offset is:

$$V_{\mathrm{OF}} = -\frac{\max(V_a, V_b, V_c) + \min(V_a, V_b, V_c)}{2}. \tag{61}$$

This centres the three references within $[-V_{\mathrm{DC}}/2,\ +V_{\mathrm{DC}}/2]$, extending the linear range:

$$|\mathbf{V}^*| \leq \frac{2}{3}V_{\mathrm{DC}} \quad \Longrightarrow \quad \text{linear (SVPWM)}. \tag{62}$$

The gain in DC bus utilisation over plain SPWM is:

$$\frac{2/3}{1/\sqrt{3}} = \frac{2}{\sqrt{3}} \approx 1.155 \quad (15.5\% \text{ improvement}). \tag{63}$$

Region 3 Overmodulation

For $|\mathbf{V}^*| > \frac{2}{3}V_{\mathrm{DC}}$, the reference exceeds the hexagon boundary during part of the electrical cycle. No combination of switch states can synthesize the exact reference. The duties computed below will fall outside $[0, 1]$ and must be saturated (clipped), introducing harmonic distortion.

6.3 Duty Cycle Computation

After applying the min-max injection ((61)):

$$V_X^* = V_X + V_{\mathrm{OF}}, \qquad X \in \{A, B, C\}. \tag{64}$$

The duty cycle for each phase is:

$$d_X = \frac{1}{2} + \frac{V_X^*}{V_{\mathrm{DC}}}. \tag{65}$$

The $\frac{1}{2}$ offset maps the zero-mean voltage reference to the $[0, 1]$ duty range of the PWM timer. In the linear regions, $d_X \in [0, 1]$ naturally. In overmodulation, apply saturation:

$$d_X^{\text{sat}} = \text{clip}(d_X,\ 0,\ 1). \tag{66}$$

Equations (65) and (66) are the complete duty computation. The formula does not change between regions only whether clipping activates.

7 Hall Sensor Caveats

Everything above assumed a known, accurate θ_e. In practice, with Hall-effect sensors, θ_e is estimated and carries systematic errors that must be understood and corrected.

7.1 Formal Definition of θ_e

The electrical angle is the angle between the stator α-axis and the rotor d-axis (permanent magnet north pole), measured positive in the forward-rotation direction:

$$\theta_e = p \cdot \theta_m, \tag{67}$$

where p is the number of pole pairs and θ_m is the mechanical rotor angle. The Park transform is only correct if θ_e refers to this quantity. An error in θ_e directly corrupts the dq decomposition.

7.2 Hall State to Angle Mapping

Three Hall sensors placed 120° electrical apart produce a 3-bit state ($H_1H_2H_3$) that transitions six times per electrical revolution, dividing it into six 60° sectors. Each sector boundary corresponds to a known electrical angle.

Between transitions, θ_e is interpolated using the estimated electrical speed from the previous transition period:

$$\theta_e(t) = \theta_{e,\text{sector}}[\text{state}] + \hat{\omega}_e \cdot \Delta t, \tag{68}$$

where $\theta_{e,\text{sector}}[\cdot]$ is a lookup table of the six sector entry angles, $\hat{\omega}_e$ is the electrical speed estimated from the last Hall transition period, and Δt is time elapsed since that transition.

This linear interpolation is exact at constant speed. At accelerating or decelerating conditions it introduces lag or lead error. For slow enough dynamics (relative to ω_e), this is acceptable.

7.3 The Electrical Offset δ

Due to physical Hall sensor placement, there is a constant angular offset δ between the Hall-derived angle $\theta_{e,\text{hall}}$ and the true rotor d-axis:

$$\theta_e = \theta_{e,\text{hall}} + \delta. \tag{69}$$

If $\delta \neq 0$, the Park transform rotates into a frame displaced from the true dq frame by δ. The measured currents relate to the true dq currents as:

$$\begin{bmatrix} I_d^{\text{meas}} \\ I_q^{\text{meas}} \end{bmatrix} = \begin{bmatrix} \cos\delta & \sin\delta \\ -\sin\delta & \cos\delta \end{bmatrix} \begin{bmatrix} I_d^{\text{true}} \\ I_q^{\text{true}} \end{bmatrix}. \tag{70}$$

An offset of δ reduces the effective torque by a factor of $\cos\delta$ and cross-couples the d and q axes, degrading current controller performance. A 30° offset already loses 13% of torque.

7.4 Step 1: Verify Commutation Order

Before calibrating δ, the commutation order (the sequence of Hall state transitions as the motor rotates forward) must be correct. An incorrect commutation order means the Hall lookup table maps states to wrong sectors, producing systematic torque errors that no δ calibration can fix.

Procedure:

1. Apply a fixed open-loop voltage vector along the α-axis: $V_a = V_{\text{cal}}$, $V_b = V_c = -V_{\text{cal}}/2$. Hold for $\sim$500 ms until the rotor settles. Confirm the Hall state is valid (not 000 or 111).
2. Slowly sweep the voltage vector through one full electrical revolution (360°) while recording each Hall state transition and the associated angle.
3. Compare the observed transition sequence against the expected forward sequence $1 \to 2 \to 3 \to 4 \to 5 \to 6 \to 1$.
   - *Forward match*: commutation order is correct, proceed to δ calibration.
   - *Reverse match*: phase order is swapped; swap any two motor phase wires (or swap B/C output assignments in firmware), then repeat from step 1.
   - *No match*: a Hall input is misassigned in firmware; permute the three Hall input mappings (6 possibilities) until a forward match is obtained, then repeat from step 1.

7.5 Step 2: Measure and Store δ

Once commutation order is confirmed, lock the rotor to a known electrical angle and read back the Hall-interpolated angle.

Procedure: Command $I_d^* = I_{\text{cal}}$, $I_q^* = 0$ and hold. This forces the rotor to align with the controller's assumed d-axis. The offset is then:

$$\delta = \theta_{e,\text{expected}} - \theta_{e,\text{hall}}\Big|_{\text{rotor locked}}. \tag{71}$$

Store δ in non-volatile memory. Apply it at every subsequent Park and inverse-Park computation via (69). No other changes to the control or modulation equations are needed.

7.6 Summary of Hall Caveats

1. Hall sensors give 60° resolution with linear interpolation between; speed estimation quality directly impacts angle accuracy.
2. The commutation sequence in firmware must match the physical wiring verify this first, before trying to tune δ.
3. The electrical offset δ is a fixed per-motor constant once calibrated, but must be re-calibrated after any mechanical reassembly or firmware Hall-mapping change.
4. Hall states 000 and 111 indicate sensor faults and must be handled in firmware before entering closed-loop control.

8 Complete System Data Flow

The full pipeline from sensor readings to gate outputs is summarized below.

$$\underbrace{I_a, I_b, I_c, \theta_{e,\text{hall}}}_{\text{sensor inputs}} \xrightarrow{(16)} I_d, I_q \xrightarrow{\text{PI + FF}} v_d^*, v_q^* \xrightarrow{\text{inv. Park + inv. Clarke}} V_a, V_b, V_c \xrightarrow{(61),(65)} \underbrace{d_A, d_B, d_C}_{\text{gate duties}}$$

Each arrow is a deterministic, cycle-by-cycle computation. The only external time-varying inputs are the current references I_d^* and I_q^* (from a higher-level torque or speed controller) and the measured angle θ_e.

9 Conclusion

This document covered the end-to-end implementation of FOC for a PMSM:

1. **Motor characterization**: three simple bench tests (DC resistance, locked-rotor AC, back-EMF) yield R_s, L_s, ψ with quantified uncertainty. No special equipment is needed beyond a DC supply, variac, multimeter, and oscilloscope.
2. **Control layer**: the Park/Clarke transforms convert measured currents to DC quantities in the rotor frame; feedback linearization decouples the d and q axes; PI gains are set by pole-placement via (24)–(25); feed-forward terms (53)–(54) cancel cross-coupling.
3. **Bandwidth and discrete implementation**: ω_n is chosen inside a window bounded below by the plant pole and maximum electrical frequency, and above by the sampling rate. The continuous K_p is used unchanged in the discrete difference equation; the continuous K_i must be scaled by $T_s/2$ (Tustin) to form the discrete integral gain that actually runs in the microcontroller.
4. **Modulation layer**: inverse Park and Clarke give three phase voltage references; min-max injection (61) extends linear range by 15.5%; duty cycles follow from (65) with clipping for overmodulation.

5. **Hall caveats**: commutation order must be verified before δ calibration; δ is a fixed offset that must be stored and applied every control cycle.