日本語参考資料
最新版英語データシートはこちら

ANALOG DEVICES

TMC2262

65V/4ARMS+の全機能内蔵型スマート・ステッピング・モーター・ドライバ

概要

TMC2262 は、ステップおよび方向のインターフェース、SPI、および豊富な診断機能を備えた、小型でスマートな高出力 1 軸ステッピング・モーター・ドライバ IC です。高集積、高エネルギー効率、小フォーム・ファクタという特長を備えているため、Nema 17、Nema 23、更には Nema 24 のステッピング・モーターに最適な、小型でスケーラブルかつコスト効率の高いモーター・ドライブ・ソリューションを実現します。

StealthChop+により、静音動作と最大限の効率、最適なモーター・トルク制御、および中間領域でのモーター共振の減衰が実現されており、機械的振動の少ない円滑なモーター動作が可能です。StallGuard+、CoolStep+、µDcStep と組み合わせることにより、非常に効率的で汎用性の高いステッピング・モーター・ドライバが実現されています。無負荷状態の Nema 17 モーターは、わずか数十 mA の供給電流で自動的に動作します。

スマートな 256 マイクロステップ・モーター・ドライバ TMC2262 は、65V、4.25A RMS、6A peak の 2 つの H ブリッジ、および非散逸性の内蔵電流検出（ICS）を統合化しています。ICS によって、かさばる外部センス抵抗が不要になり、スペースと電力を節約できます。

内蔵パワーMOSFET は約 40mΩ と低インピーダンスであるため、高い効率を実現すると共に発熱を低減します。

豊富な診断機能には、チューニングやチェック用のオン・チップ・スコープ・インターフェース（RT-OCSI）、温度および電圧 ADC、短絡保護やサーマル・シャットダウンなどの保護機能一式が含まれます。

このデバイスは、ほぼチップ・スケールのコンパクトな、熱的に最適化された 30 ピン 6mm × 6mm FCQFN パッケージで提供されます。

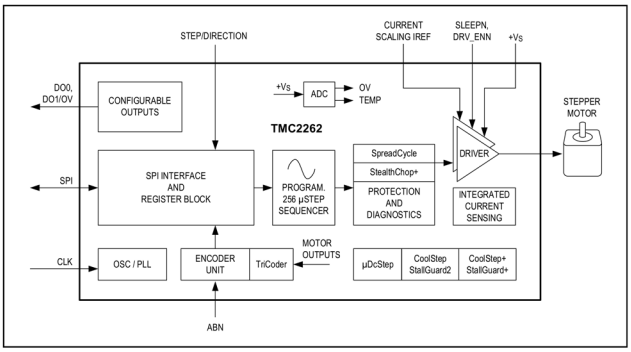
主なアプリケーション

- テキスタイル、ミシン、編み機
- ラボ・オートメーション、IVD
- ロボティクスおよびファクトリ・オートメーション
- 3D プリンタ、ラベル・プリンタ
- POS、マッサージ・チェア
- ATM、現金回収機、紙幣識別機、現金自動支払機
- ポンプおよびバルブ制御
- ステージ照明、アンテナのポジショニング

機能と利点

- 電圧範囲：4.5V～65V DC
- 低 $R_{DS(ON)}$（HS + LS）：0.08Ω（代表値、T_A = 25ºC）
- H ブリッジあたりの電流定格（25ºC での代表値）：
  - I_{RMS} = $4A_{RMS}$（6A フル・スケールのサイン波ピーク）
  - I_{MAX} = 9.0A（ブリッジ・ピーク電流）
- SPI を通じた設定、制御、診断
- ステップおよび方向制御用のインターフェース
- 全機能内蔵無損失電流検出
- インクリメンタル・エンコーダ・インターフェース
- TriCoder センサーレス停止ステップロス検出およびフルステップ・エンコーダ
- 最大 256 マイクロステップの分解能
- モーター整合のためのプログラマブルなサイン波テーブル
- 共振減衰による静音かつ円滑な StealthChop+モーター動作
- SpreadCycle 高ダイナミック・モーター制御・チョッパー
- StallGuard2、StallGuard+によるセンサーレス負荷検出
- アプリケーションで最大 90%のエネルギー節減を実現する CoolStep、CoolStep+電流制御
- µDcStep による負荷に依存した速度適応
- 受動ブレーキおよびフリーホイール・モード
- チップ温度および電源電圧を測定
- すべてを網羅する保護および診断機能
- リアルタイム・オン・チップ・スコープ・インターフェース（RT-OCSI）
- 過電圧保護出力
- 30 ピン 6mm × 6mm のコンパクトな FCQFN パッケージ

簡略ブロック図



※こちらのデータシートには正誤表が付属しています。当該資料の最終ページ以降をご参照ください。

©2026 Analog Devices, Inc. All rights reserved.　商標および登録商標は各社の所有に属します。

19-102038; Rev. 0; 02/26

本　　社／〒105-7323 東京都港区東新橋 1-9-1　東京汐留ビルディング 23F
大　阪営業所／〒532-0003 大阪府大阪市淀川区宮原 3-5-36　新大阪トラストタワー 10F
名古屋営業所／〒451-6038 愛知県名古屋市西区牛島町 6-1　名古屋ルーセントタワー 38F

目次

概要 1
主なアプリケーション 1
機能と利点 1
簡略ブロック図 1
絶対最大定格 10
電気的特性 10
パッケージ 15
FC2QFN30 6mm × 6mm 15
パッケージのマーキングの仕様 15
端子説明 16
ピン配置 18
機能図 19
詳細説明 20
動作原理 20
ステップおよび方向インターフェースを備える SPI ステッピング・モーター・ドライバ 20
主要コンセプト 21
制御インターフェース 21
ステップおよび方向インターフェース 21
自動停止パワー・ダウン 21
StealthChop+および SpreadCycle ドライバ 22
StallGuard2/StallGuard+を用いた機械的負荷のセンシング 22
CoolStep および CoolStep+を用いた負荷適応型電流制御 22
µDcStep 23
リアルタイム・オン・チップ・スコープ・インターフェース 23
TriCoder 23
エンコーダ・インターフェース 23
シリアル・ペリフェラル・インターフェース（SPI） 24
SPI データグラム構造 24
書込み／読出し選択（WRITE_notREAD） 24
各データグラム・リード・バックで転送される SPI ステータス・ビット 25
データ・アライメント 25
SPI の信号 25
SPI のタイミング 25
ステップおよび方向インターフェース 27
タイミング 27
分解能の変更 27
MicroPlyer ステップ・インターポレータおよび停止検出 28

StealthChop+ 30
StealthChop+の動作原理 31
StealthChop+プロシージャのイネーブル 32
StealthChop+の詳細な調整を行う理由 32
StealthChop+電流レギュレータの設定 33
電流レギュレータの P 調整 34
電流レギュレータの I 調整 34
電流レギュレータの PI 調整のための大まかな指針 35
電流レギュレータの PI 調整の例 35
オシロスコープを用いた CUR_P および CUR_I の調整 36
StealthChop+角度レギュレータの設定 37
角度レギュレータの応用範囲 38
角度の P 調整 38
角度の I 調整 39
角度 PI の大まかな指針 40
StealthChop+チョッパーのオプション 41
チョッパー周波数 41
フリーホイーリングおよび受動ブレーキング・モード 42
StallGuard+ 42
SGP_RESULT の解釈 42
ストール検出 43
低速ストール検出 44
COIL_INDUCT の設定 44
COIL_INDUCT の実験による最適化 45
R_COIL の測定 45
R_COIL_AUTO の解釈例 47
R_COIL の測定フローチャート 47
CoolStep+ 49
CoolStep+の設定プロシージャ 50
CoolStep+の調整 50
ターゲット負荷余裕ジェネレータ 50
負荷余裕の大まかな指針 51
CoolStep+レギュレータ 52
CoolStep+の P 調整 53
CoolStep+の I 調整 53
COOL_PI_DOWN_LIMIT の調整 53
CoolStep+の PI および勾配制限についての大まかな指針 54
StealthChop+に関連する速度スレッショルド 55

モーターのバック EMF 定数の解釈 56
速度に依存する機能の概要 57
開放負荷の検出 59
µDcStep 59
µDcStep のイネーブル 61
SpreadCycle および一般的なチョッパー 62
SpreadCycle チョッパー 63
直接モード 65
一般的な定オフ時間チョッパー 65
StallGuard2 の負荷測定 66
StallGuard2 のスレッショルド SGT の調整 67
可変速度制限値 TCOOLTHRS および THIGH 68
高トルク・リップルと共振を伴う小型モーター 69
モーター・コイル抵抗の温度依存性 69
StallGuard2 測定の正確さと再現性 69
StallGuard2 の更新レートおよびフィルタ 69
モーター・ストールの検出 70
StallGuard2 を使用したホーミング 70
StallGuard2 動作の制限 70
CoolStep の負荷適応型電流スケーリング 70
CoolStep 用の設定 70
CoolStep の調整 72
応答時間 72
低速度およびスタンバイ動作 72
内蔵電流検出機能 73
フル・スケール電流レンジの設定 73
ADC 値の解釈 75
勾配制御 75
速度ベースのモード制御 75
TriCoder - バック EMF センサーレス停止ステップロス検出 78
TriCoder BEMF デコーダの動作原理 78
TriCoder 動作に対するモーターおよび速度の条件 79
TriCoder のイネーブルまでの時間 79
TriCoder の設定 79
TriCoder のイネーブル 79
動作設定 79
ABN インクリメンタル・エンコーダ・インターフェース 81
N 信号 81

エンコーダ・カウンタ X_ENC 81
レジスタ ENC_STATUS 81
エンコーダ定数 ENC_CONST 81
モーターの分解能に合うエンコーダ設定 82
サイン波ルックアップ・テーブル 83
マイクロステップ・テーブル 83
位相シフトとモーターのマッチング 84
リセット、ディスエーブル／停止、パワー・ダウン 86
緊急停止 86
外部リセットおよびスリープ・モード 86
診断と保護 87
診断出力 87
リアルタイム・オン・チップ・スコープ・インターフェース（RT-OCSI） 87
RT-OCSI の使用例 90
過電流保護 91
過熱保護およびサーマル・シャットダウン 92
温度測定 92
チップ温度の測定 92
モーター温度の測定 92
過電圧保護と OV 出力 93
短絡保護（地絡および Vs への短絡） 93
開放負荷診断 94
低電圧ロックアウト保護 94
ESD 保護 94
クロック発振器とクロック入力 95
内部 PLL ブロック 95
内部クロックの使用 97
外部クロックの使用 97
クイック設定ガイド 98
PLL の初期化ガイド 98
電流設定ガイド 100
StealthChop+の設定 101
StealthChop+と組み合わせた StallGuard+ 102
StealthChop+と組み合わせた CoolStep+ 103
StealthChop+と組み合わせた µDcStep 動作 104
SpreadCycle の設定 105
SpreadCycle と組み合わせた CoolStep 106
初期化および設定の例 107

サステナビリティのための設計 107
レジスタ・マップ 108
型番 187
改訂履歴 188

図一覧

図 1. パッケージのマーキング（上面図） 15
図 2. バンプ側を下にした TMC2262 のピン配置 18
図 3. ブロック図 19
図 4. 代表的な外部コンポーネントを使用したブロック図 20
図 5. 停止時およびランプアップ時の自動モーター電流制御 22
図 6. StealthChop+および CoolStep+を使用した場合と純粋な SpreadCycle を使用した場合の電力節約の比較（無負荷および高負荷、24V 電源でモーターQSH4218-47-28-040 を使用） 23
図 7. SPI のタイミング図 26
図 8. STEP および DIR 信号のタイミング 27
図 9. STEP および DIR 信号の入力フィルタ構造 27
図 10. 立上がり STEP 周波数での MicroPlyer マイクロステップ補間（例：16～256） 29
図 11. StealthChop+のレギュレータ方式 31
図 12. StealthChop+電流レギュレータ 33
図 13. IHOLD 電流を 0 から増加させた場合の電流ステップ応答のモニタリングによる、電流レギュレータの調整 34
図 14. CUR_P（50% CUR_P$_{50}$）と CUR_I（25% CUR_P$_{50}$）が小さすぎる場合：緩やかな電流ランプアップ、減速時にオーバーシュート 36
図 15. 大きな CUR_P（10 × CUR_P$_{50}$）設定により、起動時と停止時に高速の応答。CUR_I が小さい（CUR_P の 10%）ため、残留している差異は緩やかにのみ補正。 36
図 16. CUR_I（CUR_P の 25%）を増加させた大きな CUR_P（10 × CUR_P$_{50}$）設定により、セット・ポイントまで短時間で電流を安定化 37
図 17. CUR_I（2 × CUR_P$_{50}$）を増加させた CUR_P = CUR_P$_{50}$ 設定においても、少ないオーバーシュートおよびアンダーシュートでセット・ポイントまで短時間で電流を安定化 37
図 18. StealthChop+角度レギュレータ 38
図 19. モーターの中間領域での共振（停止ポイント付近）が ANGLE_P を 250 に増加することにより解決 39
図 20. ANGLE_I を 0 から任意の値に切り替えることで、ANGLE_ERROR を±0 にする補正を開始 39
図 21. モーターの回転方向を 2 回反転。過度に低い値の ANGLE_I で反転させると適応が遅くなるため位相ジャークが発生します。図におけるくぼみは、T_RCOIL_MEAS によって与えられた制限値未満の速度において ANGLE_ERROR が 0 にクリアされたことを示します。 40
図 22. ANGLE_I の値を低値から 50 まで増加させることにより中間領域での共振が減衰 40
図 23. 乱れのない電流測定をチェックするための、StealthChop+において低速での ADC 結果を用いたモーター電流のパターン 41
図 24. StallGuard+の負荷余裕の計算における構造および基本パラメータ 42
図 25. モーター動作の異なる象限での StallGuard+の値 43
図 26. SGP_THRS を使用して高負荷時にモーターを停止（速度 VACTUAL と SGP_RESULT をプロット） 44
図 27. 良好な設定（SGP_RESULT が 0 近くの値になったときにモーターがストールを開始）と低すぎる設定（0 に達する前にストールが開始）との比較 45
図 28. 非常に低速で動作している場合のコイルの温度上昇による、StallGuard+の結果（SGP_RESULT）の変化 46
図 29. RCOIL 抵抗測定のフローチャート（モーター電流設定に続き StealthChop+をイネーブル） 48
図 30. モーターのトルクと負荷角度の関係 49
図 31. 相対的な効率と負荷角度の関係 49
図 32. ジェネレータ動作で効率が低下する非対称設定を示す、ターゲット負荷余裕ジェネレータの機能 51
図 33. CoolStep+レギュレータの構造と基本的なパラメータ 52
図 34. PI レギュレータの設定が低すぎる場合（負荷インクリメント時に SGP_RESULT（緑色）がアンダーシュート）と十分に高い場合との比較 53
図 35. COOL_PI_DOWN_LIMIT で制御される立下がり勾配の制限 54
図 36. 最大電流（CS_ACTUAL = 255）に達するまでの、機械的負荷の増加に伴う SGP_RESULT の減少と COOLSTEP_LOAD_RESERVE の追跡。負荷の減少時の制限された電流減少勾配（COOL_PI_DOWN_LIMIT） 54
図 37. µDcStep を使用する速度プロファイルの例 60

図 38. チョッパーの代表的な減衰フェーズ（センス抵抗は、下側経路の電流測定を表すための記号にすぎません） 62
図 39. 1 チョッパー・サイクルの間のコイル電流を示す、SpreadCycle チョッパーの概略図 64
図 40. オフセットを持つ一般的な定オフ時間チョッパーが示すコイル電流 65
図 41. 一般的なチョッパーのゼロ交差とサイン波オフセットを使用した補正 66
図 42. StallGuard2 の機能原理 67
図 43. 例：サンプル・モーターを用いた SGT の最適設定値と StallGuard2 の読出し値 69
図 44. CoolStep によるモーター電流の負荷への適合 71
図 45. 速度依存モードの選択 76
図 46. モーター動作の検知 78
図 47. インクリメンタル・エンコーダの ABN 信号の概要 81
図 48. LUT プログラミングの例 83
図 49. OFFSET_SIN90 を使用したコサイン波のシフト 84
図 50. DO0 および DO1 の出力の回路図 87
図 51. RT-OCSI を使用した CoolStep+のモニタリング 90
図 52. RT-OCSI を使用した電流 PI レギュレータのモニタリング 90
図 53. RT-OCSI を使用した CoolStep+ PI レギュレータのモニタリング 91
図 54. ブレーキ・チョッパー回路の例 93
図 55. PLL のブロック図 95
図 56. PLL を起動するためのクイック設定ガイド 98
図 57. 電流設定のためのクイック設定ガイド 100
図 58. StealthChop+を動作させるためのクイック設定ガイド 101
図 59. StallGuard+の（StealthChop+の設定に従う）基本的な設定のためのクイック設定ガイド 102
図 60. CoolStep+の（StealthChop+の設定に従う）設定のためのクイック設定ガイド 103
図 61. µDcStep の（StealthChop+および CoolStep+の設定に従う）クイック設定および動作ガイド 104
図 62. SpreadCycle のクイック設定ガイド 105
図 63. SpreadCycle と組み合わせた CoolStep 設定のためのクイック設定ガイド 106

表一覧

表 1. パッケージのマーキングの説明 15
表 2. TMC2262 の機能の概要 21
表 3. SPI データグラム構造 24
表 4. SPI 読出し／書込みフローの例 25
表 5. SPI_STATUS – ビット 39～32 の各 SPI アクセスで転送されるステータス・フラグ 25
表 6. 位相 A/B コイル電流のフルステップ／ハーフステップ・ルックアップ・テーブル値 28
表 7. StealthChop プロシージャのイネーブル 32
表 8. PWM 周波数の選択オプション 41
表 9. CoolStep+の設定プロシージャ 50
表 10. StealthChop+に関連する速度スレッショルド 55
表 11. 速度に依存する StealthChop+の機能の概要 57
表 12. µDcStep のイネーブル・プロシージャ 61
表 13. SpreadCycle および一般的な定オフ時間チョッパーを制御するパラメータ 63
表 14. SpreadCycle モードのパラメータ 64
表 15. 定オフ時間チョッパー・モードを制御するパラメータ 66
表 16. StallGuard2 関連のパラメータ 67
表 17. StallGuard の調整プロシージャ 68
表 18. 低速時に StallGuard を調整するプロシージャ 68
表 19. CoolStep の重要パラメータ 70
表 20. CoolStep のその他のパラメータとステータス情報 71
表 21. R_{REF} = 12kΩ でのフル・スケール電流レジスタ設定 73
表 22. モーター電流を制御するパラメータ 74
表 23. 速度ベースのモード制御パラメータ 76
表 24. TriCoder の制御パラメータ 80
表 25. 200 フルステップのモーターに対して 256 マイクロステップを使用するエンコーダの設定例 82
表 26. リアルタイム・オン・チップ・スコープ・インターフェース（RT-OCSI）の DAC 設定オプション 88
表 27. PLL 制御レジスタ PLL のパラメータ 96

絶対最大定格

V_S～GND ..... −0.3V～70V
V_{DD}～GND ..... −0.3V～Min. (2.2, V_S + 0.3)V
OA1、OA2、OB1、OB2 ..... −0.3V～V_S + 0.3V
V_{CP}～GND .....V_S − 0.3V～Min. (74, V_S + 6)V
CPO～GND ..... V_S − 0.3V～V_{CP} + 0.3V
CPI～GND ..... −0.3V～V_S + 0.3V
IREF～GND ..... − 0.3V～Min. (2.2, V_{DD} + 0.3)V

V_{CC_IO}～GND ..... −0.3V～5.5V
ロジック入力／出力電圧～GND ..... −0.3V～V_{CC_IO} + 0.3V
SLEEPN～GND ..... −0.3V～5.5V
動作温度範囲 ..... −40°C～125°C
ジャンクション温度 ..... +160°C
保管温度範囲 ..... −40°C～125°C
はんだ処理温度（リフロー） ..... 260°C

上記の絶対最大定格を超えるストレスを加えるとデバイスに恒久的な損傷を与えることがあります。これらの規定はストレス定格のみを定めたものであり、この仕様の動作セクションに記載する規定値以上でデバイスが正常に動作することを意味するものではありません。デバイスを長時間絶対最大定格状態に置くとデバイスの信頼性に影響を与えます。

電気的特性

（V_S = 4.5V～65V、R_{REF} = 12kΩ、代表値は T_A = 25°Cおよび V_S = 48V を前提としています。制限値は T_A = +25°Cで 100%テストされています。動作温度範囲および関連する電源電圧範囲における制限値は、設計および特性評価により裏付けられています。）

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
POWER SUPPLY							
Supply Voltage Range	V_S			4.5		65	V
Sleep Mode Current Consumption	I_{VS}	T_J = 25°C	V(SLEEPN) = 0			10	µA
Driver Off Current Consumption	I_{VS}	V(SLEEPN) = 1, V_{CC_IO} = 0V				1.5	mA
Quiescent Current Consumption	I_{VS}	V(SLEEPN) = 1				1.7	mA
1.8V Regulator Output Voltage	V_{VDD}	V_S = 4.5V, I_{LOAD} = 20mA			1.8		V
VDD Current Limit	$IV18_{LIM}$			70			mA
Charge Pump Voltage	V_{CP}				V_S + 2.7		V
Logic I/O Supply Voltage Range	V_{CC_IO}			2.3		5.5	V
V_{CC_IO} UVLO Rising Threshold	UVLOVCCR			2.0	2.1	2.2	V
V_{CC_IO} UVLO Falling Threshold	UVLOVCCF			1.9	2.0	2.1	V
V_{CC_IO} UVLO Hysteresis	UVLOVCCH				100		mV
Sleep Mode Current Consumption	I_{VCC_IO}	V(SLEEPN) = 0			0	5	µA
Quiescent Current Consumption	I_{VCC_IO}	After LDO switchover. 100mA is the current limit of the internal LDO.				100	mA
LOGIC LEVEL INPUTS-OUTPUTS							

（V_S = 4.5V～65V、R_{REF} = 12kΩ、代表値は T_A = 25℃および V_S = 48V を前提としています。制限値は T_A = +25℃で 100%テストされています。動作温度範囲および関連する電源電圧範囲における制限値は、設計および特性評価により裏付けられています。）

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Input Voltage Level - High	V_{IH}		0.7 × V_{CC_IO}			V
Input Voltage Level - Low	V_{IL}				0.3 × V_{CC_IO}	V
Input Hysteresis	V_{HYS}			0.1 × V_{CC_IO}		mV
Internal Pullup/Pulldown Resistance	R_{PULL}	To GND or to V_{CC_IO}	60	100	140	KΩ
Input Leakage	IN_{Leak}	Inputs without pullup/pulldown resistance	-1		1	µA
Open-Drain Output Logic-Low Voltage	V_{OL}	I_{LOAD} = 5mA			0.4	V
Push-Pull Output Logic-High Voltage	V_{OH}	I_{LOAD} = 5mA	V_{CC_IO} - 400mV			
Open-Drain Output Logic-High Leakage Current	I_{OH}	V(PIN) = 5V	-1		1	µA
SLEEPN Voltage Level High	VIH_{SLEEPN}		0.9			V
SLEEPN Voltage Level Low	VIL_{SLEEPN}				0.6	V
SLEEPN Pulldown Input Resistance	RPD_{SLEEPN}		0.85	1.5		MΩ
OUTPUT SPECIFICATIONS						
Output ON-Resistance Low Side	$R_{DS(ON)}$ (LS)	Full-scale bits = 11		0.036	0.070	Ω
		Full-scale bits = 10		0.047	0.095	
		Full-scale bits = 01		0.070	0.140	
		Full-scale bits = 00		0.138	0.280	
Output ON-Resistance High Side	$R_{DS(ON)}$ (HS)			0.040	0.080	Ω
Output Leakage	I_{LEAK}	Driver OFF, T = +125°C	-80		80	µA
		Driver OFF, T = +25°C	-5		5	
Dead Time	t_{DEAD}			100		ns
Output Slew Rate	SR	Slew-rate bits = 00		100		V/µs
		Slew-rate bits = 01		200		
		Slew-rate bits = 10		400		
		Slew-rate bits = 11		800		
PROTECTION CIRCUITS						
Overcurrent Protection Threshold	OCP	Full-scale bits = 11	10.0			A
		Full-scale bits = 10	7.5			

（V_S = 4.5V～65V、R_{REF} = 12kΩ、代表値は T_A = 25℃および V_S = 48V を前提としています。制限値は T_A = +25℃で 100%テストされています。動作温度範囲および関連する電源電圧範囲における制限値は、設計および特性評価により裏付けられています。）

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
		Full-scale bits = 01	5.5			
		Full-scale bits = 00	2.8			
Overcurrent Protection Blanking Time	TOCP		0.9	1.5	2.5	µs
UVLO Threshold on V_S	UVLO	V_S rising	3.85	4	4.15	V
UVLO Threshold on V_S Hysteresis	UVLOHYS			110		mV
Thermal Protection Threshold Temperature	TSD	For temperature rising		165		°C
Thermal Protection Temperature Hysteresis				20		°C
CURRENT REGULATION						
IREF Pin Resistor	R_{REF}			12		kΩ
IREF Output Voltage	V_{REF}		0.882	0.9	0.918	V
Full Scale Current Constant	KIFS	IFS = 1.5A		18		A x kΩ
Full Scale Current Constant	KIFS	IFS = 3A		36		A x kΩ
Full Scale Current Constant	KIFS	IFS = 4.5A		54		A x kΩ
		IFS = 6A		72		
Current Trip Regulation Accuracy	DITRIP1	ITRIG from 30% to 100% FS	-5		5	%
	DITRIP2	ITRIG from 15% to 30% FS	-10		10	
	DITRIP3	ITRIG from 10% to 15% FS	-15		15	
Current Trip Regulation Matching				0.7		%
FUNCTIONAL TIMINGS						
SLEEP Time	t_{SLEEP}	SLEEPN = 0 to OUTxx three state			50	µs
Wake-Up Time from Sleep	t_{WAKE}	SLEEPN = 1 to normal operation		2		ms
Enable Time	t_{ENA}	Time from DRV_ENN pin rising edge to driver on			1	µs
Disable Time	t_{DIS}	Time from DRV_ENN pin falling edge to driver off			1.4	µs
CLOCK OSCILLATOR AND INPUT						
Internal Oscillator Frequency	f_{CLKOSC}	Including variation over temperature	15.3	16	16.7	MHz

（V_S = 4.5V～65V、R_{REF} = 12kΩ、代表値は T_A = 25℃および V_S = 48V を前提としています。制限値は T_A = +25℃で 100%テストされています。動作温度範囲および関連する電源電圧範囲における制限値は、設計および特性評価により裏付けられています。）

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
External Clock Frequency	f_{CLK}	n = 1 … 32		1	n × 1	32	MHz
External Clock High/Low Time	t_{CLKL}			15			ns
External Clock Timeout Detection in Cycles of Internal fCLKOSC		Checked for lower/upper limit.			32		cycles
Allowed Tolerance for External Clock (no clk_1M0_tmo)	f_{CLKTMO}	Related to setting of CLOCK DIVIDER.		-1		1	%
Internal PLL frequency	f_{PLL}	f_{PLL} = 5 × fCLK16			80		MHz
Internal PLL Startup Time (Until Commit Goes Low)	$t_{PLLSTRT}$					170	µs
Internal 16MHz Clock for Internal Controller Logic and Interface Logic	f_{CLK16}	When internal oscillator is used as clock source.	CLK_SYS_SEL = 0	15.3	16	16.7	MHz
Internal 16MHz Clock for Internal Controller Logic and Interface Logic	f_{CLK16}	When external clock and PLL are used as clock source.	CLK_SYS_SEL = 1	15.84	16	16.16	MHz
SPI TIMINGS							
SCK Valid before or after Change of CSN	t_{CC}			TSCLK			ns
CSN High Time	t_{CSH}			4 x 1/fCLK16			ns
SCK Low Time	t_{CL}			20			ns
SCK High Time	t_{CH}			20			ns
SCK Frequency	f_{SCK}					8	MHz
SDI Setup Time before SCK Rising Edge	t_{DU}			10			ns
SDI Hold Time after SCK rising edge	t_{DH}			10			ns
Data Out Valid Time after SCK Falling Edge	t_{DO}	V_{CC_IO} = 3.3V			34	50	ns
		V_{CC_IO} = 2.3V			55	85	
SDI, SCK, and CSN Filter Delay Time	t_{FILT}	Rising and falling edge			10		ns
					10		
STEP/DIR TIMINGS							

（V_S = 4.5V～65V、R_{REF} = 12kΩ、代表値は T_A = 25℃および V_S = 48V を前提としています。制限値は T_A = +25℃で 100%テストされています。動作温度範囲および関連する電源電圧範囲における制限値は、設計および特性評価により裏付けられています。）

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Step Frequency	f_{STEP}	Maximum microstep resolution	Dedge = 1			$f_{CLK16}/8$	MHz
			Dedge = 0			$f_{CLK16}/4$	
Fullstep Frequency	f_{PS}					$f_{CLK16}/512$	MHz
STEP High Time	t_{SH}			100			ns
STEP Low Time	t_{SL}			100			ns
DIR to STEP Setup Time	t_{SU}			100			ns
DIR to STEP Hold Time	t_H			20			ns
ENCODER TIMING							
Encoder Counting Frequency	f_{CNT}				<10	16	MHz
A/B/N Input Low Time	t_{ABNL}			160			ns
A/B/N Input High Time	t_{ABNH}			160			ns
A/B/N Spike Filtering Time	$t_{FILTABN}$	Rising and falling edge			190		ns
ADC/SUPPLY/TEMPERATURE							
ADC Resolution					9		Bit
ADC_VSUPPLY Resolution	V_{ADCFS}				140.9		mV
Driver Temperature Accuracy	T_{driver}				±10		°C
Supply Voltage Measurement Accuracy		V_S ≥ 15V		-5.0		5.0	%
		V_S < 15V		-10		10	
ADC Sample Rate	$F_{SAMPLE,ADC}$				2.45		kHz
DAC OUTPUT							
DAC Resolution		V_{CC_IO} > 2.9V			8		Bit
Output Voltage Range		V_{CC_IO} > 2.9V		0.2		2.7	V
Output Current		V_{CC_IO} > 2.9V				5	mA
DAC Conversion Frequency		V_{CC_IO} > 2.9V				100	kHz

パッケージ

FC2QFN30 6mm × 6mm

Package Code	F306A6F+1F
Outline Number	21-100708
Land Pattern Number	90-100245
Thermal Resistance, Four-Layer Board, Ambient of 25°C, 4W Power Dissipation:	
Junction to Ambient (θ_{JA})	23 C/W
Junction to Case Thermal Resistance (θ_{JC})	2.6 C/W
Ψ_{JT}	1.9 C/W

パッケージのマーキングの仕様

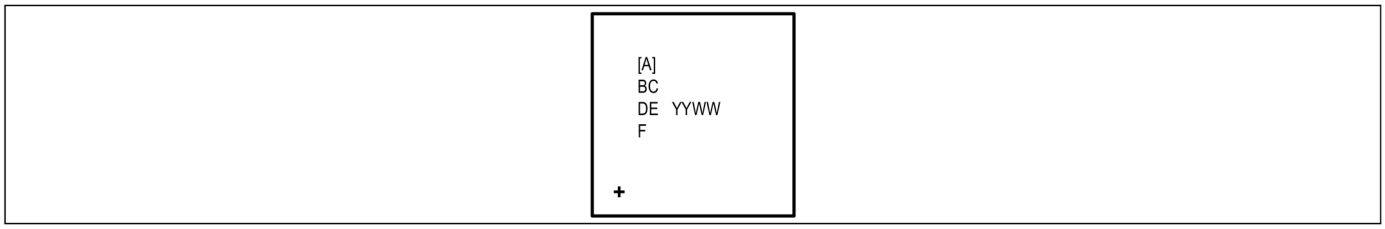


図 1. パッケージのマーキング（上面図）

表 1. パッケージのマーキングの説明

記号	説明
[A]	アナログ・デバイセズのロゴ
BC	B – 製品番号のプレフィックス （3桁、通常は「TMC」） C – 製品番号（4 桁）
DE	D – テスト・フロー E – 製品番号のサフィックス（2 桁）
YYWW	YY – 製造年 WW – 製造作業週
F	G – アッセンブリおよび製造を行った工場の コード（7 桁）
+	鉛フリー・インジケータ（1 番ピン付近に配置）

端子説明

ピン	名称	機能	タイプ
3, 18	GND	グラウンド。グラウンド・プレーンに接続します。	GND
1, 5, 16, 20	V_S	モーター電源電圧。GND プレーン／露出パッドへ最短のループを形成してピン近くでのフィルタによる除去機能を提供します。	Supply
26	V_{DD}	内蔵 1.8V レギュレータの出力。最高性能を発揮させるには、AGND との間のピン近くに 2.2µF 以上のセラミック・コンデンサを接続します。	Supply
6	V_{CP}	チャージ・ポンプ電圧。1.0µF、10V のコンデンサを用いて V_S に接続します。	Output
7	CPO	チャージ・ポンプ・コンデンサ出力。	Output
8	CPI	チャージ・ポンプ・コンデンサ入力。22nF、100V のコンデンサを用いて CPO に接続します。	Output
11	CLK	CLK 入力。内部クロック用に短い配線を用いて GND に接続します。あるいは外部クロックを供給します。外部クロック信号が失われた場合には内蔵のクロック・フェイル・オーバー回路が保護します。	DI
12	CSN	SPI チップ・セレクト入力（負でアクティブ）。	DI (pd)
13	SCK	SPI シリアル・クロック入力。	DI (pd)
14	SDI	SPI データ入力。	DI (pd)
15	SDO	SPI データ出力（トライステート）。	DIO (pd)
27	I_{REF}	電流スケーリング用のアナログ・リファレンス電流。外付け抵抗を介して GND に接続します。	AI
23	ENCB	エンコーダの B チャンネル入力。	DI
24	ENCA	エンコーダの A チャンネル入力。	DI
28	ENCN/OVN	エンコーダの N チャンネル入力／過電圧出力（負でアクティブ）。	DI/DO
30	DRV_ENN	イネーブル入力。このピンがハイ・レベルに駆動されると、電力段がオフ（すべてのモーター出力がフロート状態）に切り替わります。	DI (pu)
22	DO0	設定可能出力／診断出力0。 内蔵モーション・コントローラから外部ドライバへの割込みまたはSTEP/DIR出力。 オープン・ドレイン・モードでは外部プルアップ抵抗を使用します。 システムがリセット状態の場合は、このピンがアクティブにローになり、リセット状態であることを外部コントローラに通知します。 アプリケーションの調整用の設定可能オン・チップ・スコープ・インターフェース（OCSI）アナログ出力。	DO/AO
19	OB2	モーター・コイル B の出力 2。	A
17	OB1	モーター・コイル B の出力 1。	A
2	OA2	モーター・コイル A の出力 2。	A
4	OA1	モーター・コイル A の出力 1。	A
21	SLEEPN	ロー・アクティブのパワー・ダウン入力／リセット入力。 デバイスをスリープ・モードにするには、連続的なロー・レベルを印加します。 ICがスリープ・モード／リセットから回復した後は、再度使用する前に再設定を行う必要があります。スリープ・モードに入る前の最後のレジスタ内容は保存されません。 ICの再設定の間、DRV_ENNを用いてブリッジ・ドライバをディスエーブルしたままにすることを推奨します。 モーター速度が高い場合には決して使用しないでください。 使用しない場合は、V_{CC_IO}に接続します。 内部プルダウン。	AI
29	DO1	設定可能出力／診断出力1。 内蔵モーション・コントローラから外部ドライバへの割込みまたはSTEP/DIR出力。 オープン・ドレイン・モードでは外部プルアップ抵抗を使用します。 アプリケーションの調整用の設定可能オン・チップ・スコープ・インターフェース（OCSI）アナログ出力。	DO/AO
EP	GND	露出ダイ・パッド 露出ダイ・パッドは GND プレーンに接続します。GND プレーンとの間には熱伝導のためにできるだけ多くのビアを使用します。電力段および内部回路には GND ピンとして機能します。	GND

25	V_{CC_IO}	回路の IO レベルを定義するために外部電源から供給されるデジタル IO 電源電圧。出力ピンに適切な電圧レベルを設定するために必要です。	AI
10	STEP	STEP 入力	DI
9	DIR	DIR 入力	DI

ピン配置

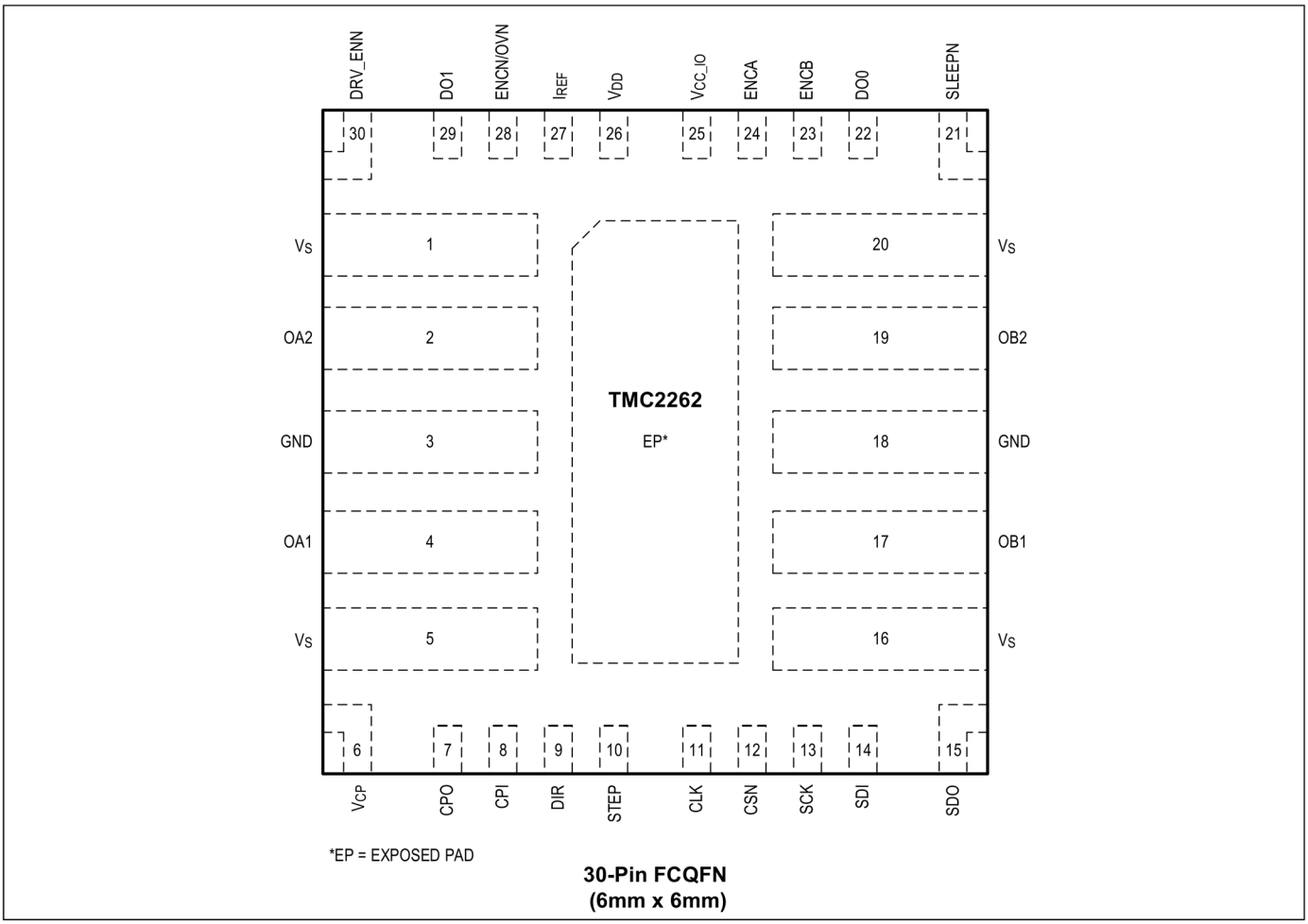


図 2. バンプ側を下にした TMC2262 のピン配置

機能図

TMC2262

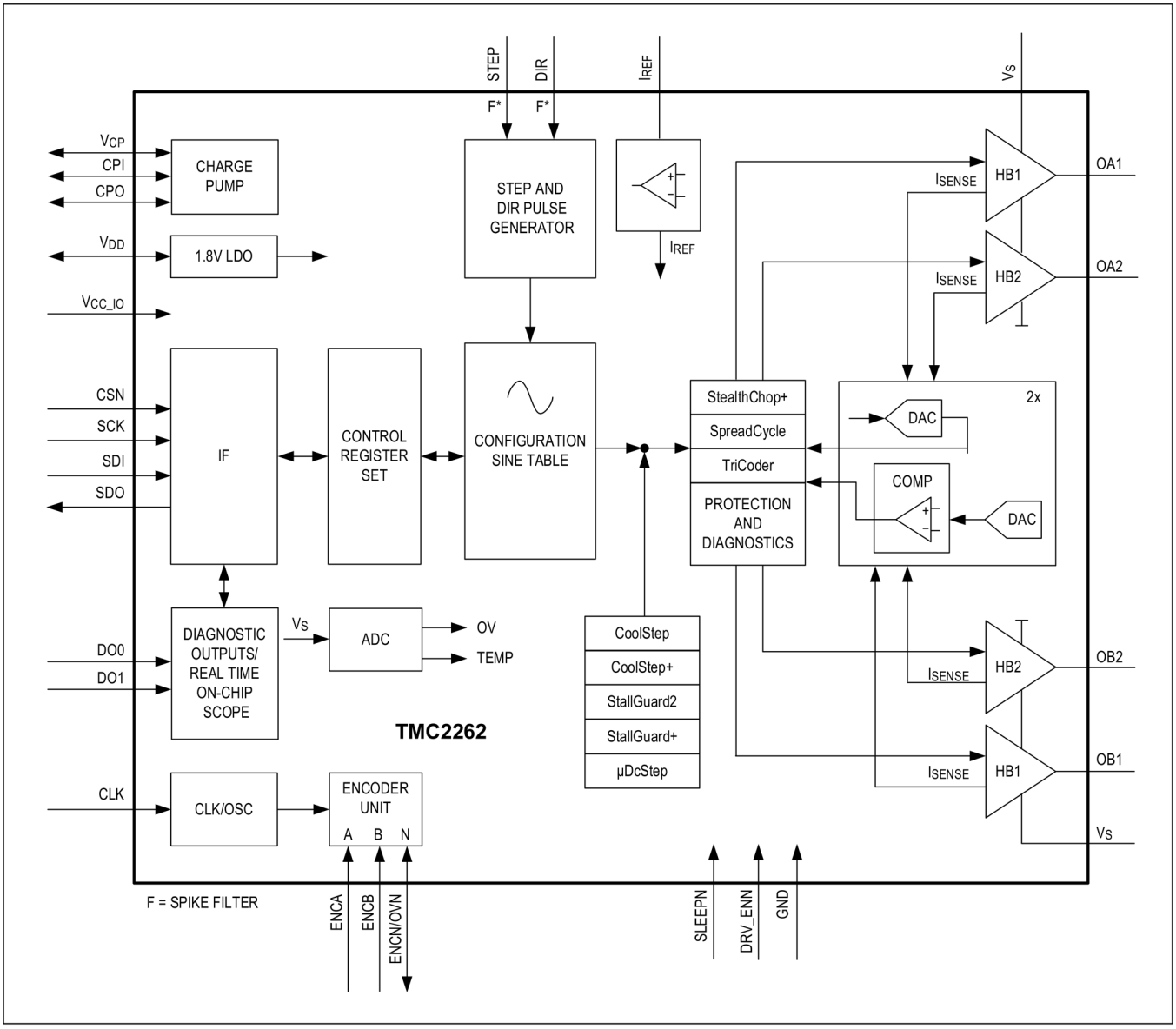


図 3. ブロック図

詳細説明

動作原理

ステップおよび方向インターフェースを備える SPI ステッピング・モーター・ドライバ

TMC2262 はインテリジェントなステッピング・モーター・ドライバ・チップです。このスマート電力変換コンポーネントは 2 相ステッピング・モーターを直接駆動し、また、設定、制御、および診断フィードバックのために CPU とのインターフェースを行います。ステッピング・モーターを駆動するための電流制御機能はすべて内蔵されています。TMC2262 は、アプリケーション内でモーターの動作をサポートするユニークな機能を数多く備えています。

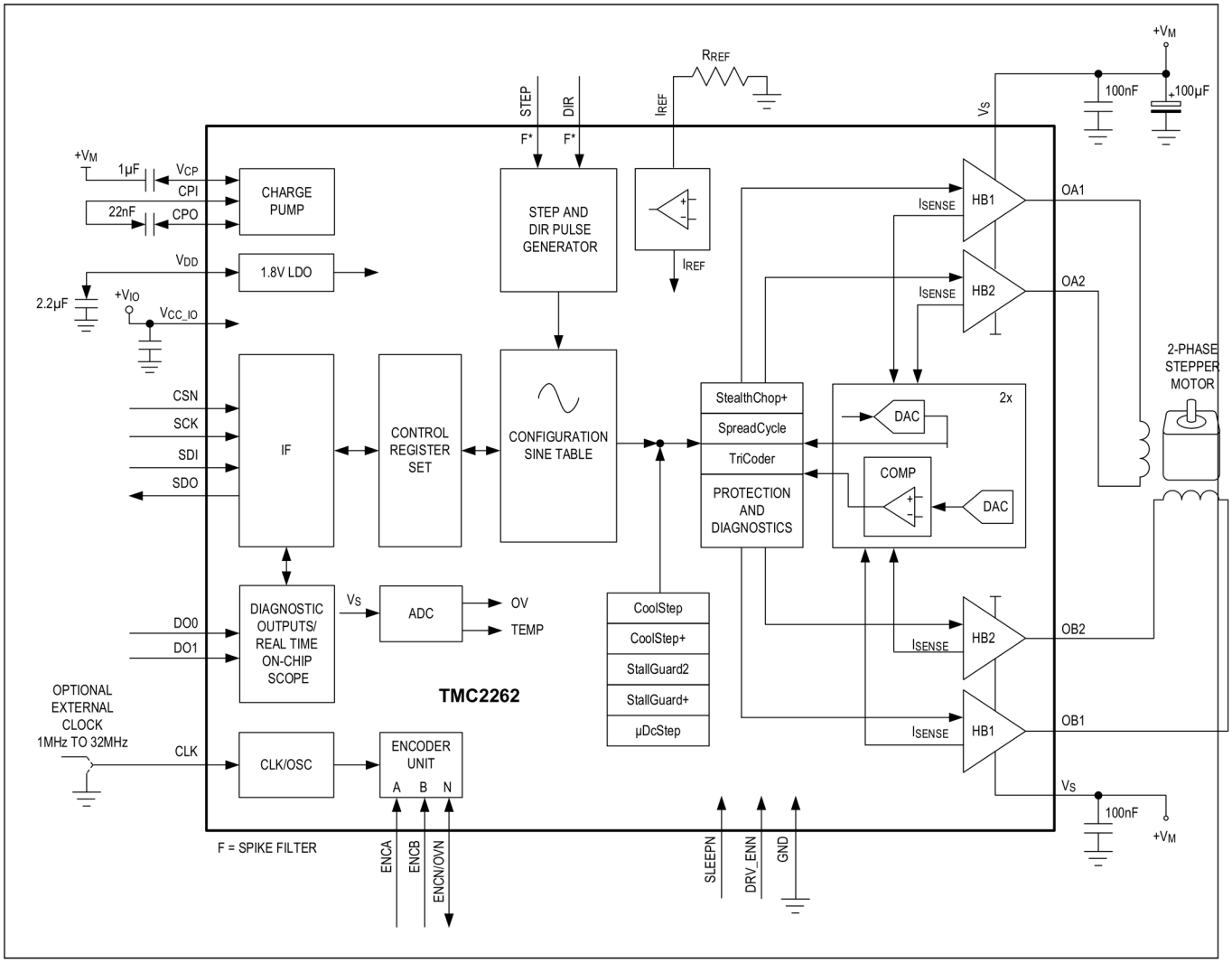


図 4. 代表的な外部コンポーネントを使用したブロック図

主要コンセプト

TMC2262 は、アナログ・デバイセズ製品独自の高度な機能を各種備えています。これらの機能は、多くのステッピング・モーター・アプリケーションにおいて、精度、エネルギー効率、信頼性、動作の滑らかさの向上や、動作による温度上昇の抑制に貢献しています。

表 2. TMC2262 の機能の概要

機能	説明
StealthChop+	円滑なモーター動作と共振減衰を実現する、静音電圧制御モーター・チョッパー。
StallGuard+	StealthChop+用の優れたセンサーレス・ストール検出および機械的負荷測定。
CoolStep+	StallGuard+を使用し、モーターやドライバの効率を最大にして温度上昇を最小限に抑えるよう、モーター電流を適応させます。
µDcStep	負荷に応じた自動速度適応機能により、可能な限り最速で動作している間にも過負荷を生じないステッピング・モーター動作を実現する、CoolStep+ベースのモード。
SpreadCycle	動的動作を最大限に高める、サイクルごとの高精度な電流制御。
StallGuard2	SpreadCycle 用のセンサーレス・ストール検出および機械的負荷測定。
CoolStep	StallGuard2 を使用し、モーターやドライバのエネルギー効率を向上して温度上昇を抑えるよう、モーター電流を適応させます。
TriCoder	モーターが給電されていないときにモーターのバック EMF を使用する、センサーレスの停止ステップロス検出機能。
リアルタイム・オン・チップ・スコープ・インターフェース	診断用出力に内蔵された DAC を用いる、ライブ・アナログ 2 チャンネル・デバッグ・オプション。

これらの性能向上機能に加え、アナログ・デバイセズのモーター・ドライバは、短絡出力、出力オープンサーキット、過熱、低電圧などの状態を検出して保護するための防護機能を備え、安全性および機器の誤動作からの回復をより確実なものにします。

制御インターフェース

TMC2262 は、シリアル・ペリフェラル・インターフェース（SPI）をサポートします。SPI は、バス・クロックに同期するビットシリアル・インターフェースです。バス・コントローラからバス・ペリフェラルに送られるビットごとに、同時に別のビットがペリフェラルからコントローラに送り返されます。SPI コントローラ（MCU など）とペリフェラルとの間の通信は、常に、1 つの 40 ビット・コマンド・ワードの送信と 1 つの 40 ビット・ステータス・ワードの受信で構成されます。

ステップおよび方向インターフェース

ステップおよび方向インターフェースは、モーターの動き（ポジション、速度、加速度、方向）を外部のモーション・コントローラから制御するために用いられます。STEP 入力のアクティブ・エッジは、立上がりエッジとするか、立上がりエッジと立下がりエッジ両方とすることができ、これはモード・ビット（dedge）により制御されます。両方のエッジを用いることで STEP 信号のトグル・レートを半分に短縮できます。これは、光学的に絶縁を行うカプラなど動作の遅いインターフェースで制御する場合に便利です。それぞれのアクティブ・エッジにおいては、DIR 入力からサンプリングされたステートが、ステップ動作を順方向にするか逆方向にするかを決定します。各ステップは、設定に応じてフルステップまたはマイクロステップとすることができます。DIR がロー状態のときのステップ・インパルスはマイクロステップ・カウンタ値を加算し、DIR がハイ状態のときは減算します。加減算の量はマイクロステップ分解能によって制御されます。内部テーブルはカウンタ値を、モーター電流を制御するターゲット電流値に換算します。

自動停止パワー・ダウン

自動的な電流低減機能により、アプリケーションの消費電力条件および冷却条件が大幅に軽減されます。動作電流を半分に低減するだけで既に、停止に要する消費電力がおよそ 25%に減少します。停止電流、遅延時間、減衰の各パラメータは、シリアル制御インターフェースを通じて設定できます。

自動フリーホイーリングおよび受動モーター・ブレーキングが、停止用オプションとして提供されています。受動ブレーキングにより、効果的な減衰およびブレーキングを維持したまま、モーター停止の消費電力をゼロに減少できます。

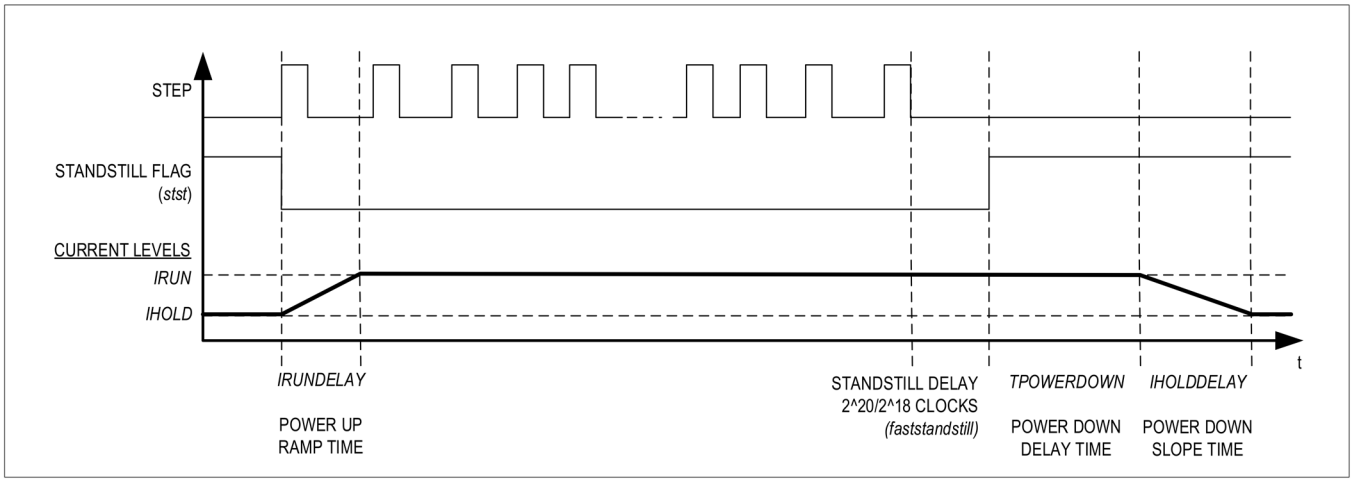


図 5. 停止時およびランプアップ時の自動モーター電流制御

StealthChop+および SpreadCycle ドライバ

StealthChop+は、電圧制御のモーター・チョッパー原理であり、高精度のセンサーレス負荷測定、共振減衰、優れたエネルギー効率という特長をそのコンセプトに取り込んでいると同時に、最高のポジション精度と極めて静かなモーター動作を実現しています。

モーターの弱め界磁範囲に深く入り込む極めて高速のアプリケーションでは、SpreadCycle が StealthChop+の代替となります。これは、サイクルごとの高度なチョッパー・モードです。広い速度範囲および負荷範囲にわたり、円滑な動作と良好な共振減衰が可能です。SpreadCycle のチョッパー方式では、高速減衰サイクルを自動的に統合・調整できるため、滑らかなゼロ交差性能を確実に実現します。

StealthChop+および SpreadCycle は、組み合わせて用いることで両者の利点をどちらも生かすことが可能です。StealthChop+は、無騒音の停止や遅い動作、およびフル・トルクが利用できるモーターの通常の速度範囲をカバーする静音で円滑な性能を実現するのに適しています。SpreadCycle は、高い運動性能と最高ピーク速度を低振動で実現するために、高速で用いるのに適しています。

利点

- 低コスト・モーターでのマイクロステッピング性能が大幅に向上。
- モーターは滑らかで静かに動作。
- 共振の抑制（特に中間領域での共振）。
- スタンバイ時のノイズが完全に無音。
- 機械的な共振が抑えられているためトルク出力が向上。

StallGuard2/StallGuard+を用いた機械的負荷のセンシング

StallGuard2 および StallGuard+では、モーターに加わる負荷を正確に測定できます。これは、ストール検出や、モーターをストールさせる下限未満の負荷の場合におけるその他の用途（CoolStep および CoolStep+の負荷適応型電流低減など）で用いることができます。

これにより駆動について得られる情報が多くなるため、センサーレス・ホーミングや駆動機能の診断などの機能が可能になります。StallGuard2 は SpreadCycle チョッパーと組み合わせられますが、StallGuard+を StealthChop+と組み合わせるには別の原理を用います。

CoolStep および CoolStep+を用いた負荷適応型電流制御

CoolStep および CoolStep+は、最適な電流でモーターを駆動します。これらは StallGuard2 または StallGuard+の負荷測定情報を用いて、実際の負荷状態に必要な最低限の量となるようモーター電流を調整します。

CoolStep および CoolStep+を用いることでエネルギーを節約でき、部品の発熱が抑えられます。CoolStep では最適な電流でモーターを駆動するので、トルク・リザーブが約 50%の標準的な動作と比較して、モーター効率が大幅に向上します。CoolStep+は、更に効率を向上します。

利点

- 最高のエネルギー効率、消費電力は CoolStep の場合最大 75%低減、CoolStep+の場合最大 90%低減。
- モーターの発熱が減少。
- 機械的精度が向上。
- 冷却が軽減または不要。
- 信頼性が向上。
- より小型のモーターを使用可能、必要なトルク・リザーブが軽減。
- モーターの過剰な励振エネルギーが低減するためモーター・ノイズが減少。

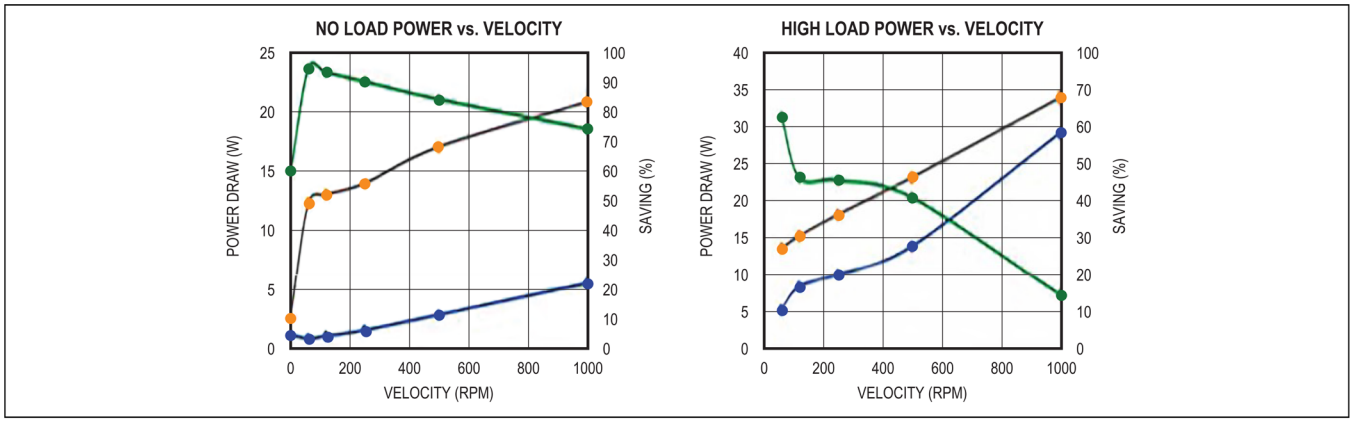


図 6. StealthChop+および CoolStep+を使用した場合と純粋な SpreadCycle を使用した場合の電力節約の比較
（無負荷および高負荷、24V 電源でモーターQSH4218-47-28-040 を使用）

µDcStep

µDcStep（マイクロ DcStep）は、DC モーターに類似した動作モードであり、ポジショニング能力を保護したまま、動的な速度応答と過負荷状態に対するモーターの調整が可能であることによりステップロスを防止しています。

リアルタイム・オン・チップ・スコープ・インターフェース

TMC2262 の斬新な機能の 1 つが、リアルタイム・オン・チップ・スコープ・インターフェース（RT-OCSI）です。2 つの内部 DAC を診断ピンに割り当て、チューニングのサポート、デバッグ、モニタなどのために、広い範囲の内部レジスタに出力できます。

TriCoder

TriCoder 機能は、モーターのバック EMF を利用する、TMC2262 のセンサーレスの停止ステップロス検出機能です。これは、モーターが給電されていないときのモーターの停止時のポジション・ロスを検出します。あるいは、モーターをエンコーダとして用いることにより、ティーチインを行うことができます。

エンコーダ・インターフェース

TMC2262 は、外部インクリメンタル・エンコーダ用にエンコーダ・インターフェースを備えています。エンコーダを使用することにより、接続しているマイクロコントローラがホーミングまたはポジションの検証を行うことができます。プログラマブル・プリスケーラを用いると、エンコーダの分解能をモーターの分解能に合わせることができます。32 ビットのエンコーダ・カウンタが備わっています。

シリアル・ペリフェラル・インターフェース（SPI）

SPI データグラム構造

TMC2262 は、40 ビットの SPI データグラムを用いてマイクロコントローラと通信します。ハードウェア SPI を備えたマイクロコントローラは、通常、8 ビットの整数倍を用いて通信を行うことができます。データグラムの伝送期間中常に、CSN ラインはアクティブ（ロー）状態を維持する必要があります。

各データグラムは、アドレス・バイトとそれに続く 4 データ・バイトで構成されます。これにより、レジスタ・セットとの間で直接、32 ビットのデータ・ワード通信を行うことができます。各レジスタは、使用するデータ・ビットが 32 ビット未満であっても、32 データ・ビットでアクセスされます。簡単化のため、各レジスタは以下のように 1 バイト・アドレスで指定されます。

- 読出しアクセスでは、アドレス・バイトの最上位ビットが 0 です。
- 書込みアクセスでは、アドレス・バイトの最上位ビットが 1 です。

全レジスタが読出し可能で、その大半は読書き可能、一部は読出し専用です。また、一部のレジスタをクリアするには 1 を書き込みます（例：GSTAT レジスタ）。

表 3. SPI データグラム構造

MSB (TRANSMITTED FIRST)				40-BIT				LSB (TRANSMITTED LAST)	
39 ... 0									
Write: 8-bit address Read: 8-bit SPI status		Read/write 32-bit data							
39 ... 32		31 ... 0							
Write to: RW + 7-bit address read from: 8-bit SPI status		8-bit data		8-bit data		8-bit data		8-bit data	
39 / 38 ... 32		31 ... 24		23 ... 16		15 ... 8		7 ... 0	
W	38...32	31...28	27...24	23...20	19...16	15...12	11...8	7...4	3...0

書込み／読出し選択（WRITE_notREAD）

読出しと書込みの選択は、アドレス・バイトの MSB（SPI データグラムのビット 39）で制御されます。このビットは読出しアクセスの場合は 0、書込みアクセスの場合は 1 です。そのため、W と記されたビットは WRITE_notREAD 制御ビットです。アクティブ・ハイの書込みビットはアドレス・バイトの MSB です。したがって、書込みアクセスではアドレスに 0x80 を追加する必要があります。SPI は、W ビットに関わらず、データを常にコントローラに送り返します。直前のアクセスが読出しアクセスであった場合、返されるデータは、直前のデータグラムで伝送されたアドレスで読み出されるデータです。直前のアクセスが書込みアクセスであった場合は、リード・バックされるデータはその前に受信した書込みデータを反映したものになります。したがって、読出しアクセスと書込みアクセスの違いは、読出しアクセスではアドレス指定されたレジスタにはデータを伝送せずにアドレスのみを伝送し、その 32 データ・ビットはダミーであるという点にあります。また、更に続けて読出しまたは書込みアクセスを行うと、その前の読出しサイクルで伝送されたアドレスからのデータ読出しが返されます。

読出しアクセス要求のデータグラムはダミーの書込みデータを使用します。読出しデータは、その後の読出しアクセスまたは書込みアクセスで再びコントローラに返されます。したがって、パイプライン方式で複数のレジスタを読み出せます。TMC5241 との間でデータの読出しや書込みが行われる場合は必ず、返される MSB に SPI のステータスが含まれます。SPI_STATUS は 8 個のうちから選択されたいくつかのステータス・ビットです。

例：アドレス 0x12 のレジスタ（*TSTEP*）への読出しアクセスの場合、読出しアクセスの直前のアクセス時にアドレス・バイトを 0x12 に設定する必要があります。レジスタ（*X_ENC*）への書込みアクセスの場合、アドレス・バイトは 0x80 + 0x39 = 0xB9 に設定する必要があります。読出しアクセスの場合、データ・ビットは任意の値（–）を持つことが可能です。そのため、0 に設定することもできます。

表 4. SPI 読出し／書込みフローの例

ACTION	DATA SENT TO *{Variable not found in the variables list}*	DATA RECEIVED FROM *{Variable not found in the variables list}*
Read *TSTEP*	0x1200000000	0xST and unused data*
Read *TSTEP*	0x1200000000	0xST and *TSTEP*
Write *X_ENC* = 0x00ABCDEF	0xB900ABCDEF	0xST and *TSTEP*
Write *X_ENC* = 0x00123456	0xB900123456	0xST00ABCDEF

*ST はステータス・ビット SPI_STATUS のためのプレースホルダです。

各データグラム・リード・バックで転送される SPI ステータス・ビット

新たなステータス情報は、各アクセスの最後にラッチされ、次の SPI 転送で使用できます。

表 5. SPI_STATUS – ビット 39～32 の各 SPI アクセスで転送されるステータス・フラグ

ビット	名称	コメント
7:4	*Don't care*	TMC2262 では使用しません
3	*standstill*	DRV_STATUS[31] – 1：モーターが停止状態であることを示す
2	*sg*	DRV_STATUS[24] – 1：StallGuard フラグがアクティブであることを示す
1	*driver_error*	GSTAT[1] – 1：ドライバ・エラーを示す （レジスタ *GSTAT* を用いてクリア）
0	*reset_flag*	GSTAT[0] – 1：リセットが発生したことを示す （レジスタ *GSTAT* を用いてクリア）

データ・アライメント

すべてのデータは右揃えです。一部のレジスタは符号なし（正）の値を表し、一部のレジスタは整数値（符号付き）を 2 の補数で表します。1 つのビットまたはビットのグループは、それぞれ単一ビットまたは整数グループで表されます。

SPI の信号

TMC2262 の SPI バスには次の 4 種類の信号があります。

- SCK – バス・クロック入力
- SDI – シリアル・データ入力
- SDO – シリアル・データ出力
- CSN – チップ・セレクト入力（アクティブ・ロー）

チップ・セレクト入力 CSN をローにすることで、SPI トランザクションの SPI ペリフェラルがイネーブルされます。ビット転送は、バス・クロック SCK に同期します。ペリフェラルは、SCK の立上がりエッジで SDI からのデータをラッチし、立下がりエッジの後 SDO にデータを送ります。最初に最上位ビットが送られます。TMC2262 でバス・トランザクションを行うには、最低でも 40 SCK クロック・サイクルが必要です。

40 より多くのクロックを駆動する場合、SDI にシフト・インされた追加ビットは、内部シフト・レジスタを介して 40 クロックの遅延後に SDO にシフト・アウトされます。これを用いると、複数チップをデイジーチェーン接続できます。

バス・トランザクション全体を通じて、CSN はローにする必要があります。CSN がハイになると、内部シフト・レジスタの内容は内部制御レジスタにラッチされ、SPI コントローラから SPI ペリフェラルへのコマンドとして認識されます。40 を超えるビットが送られた場合、CSN の立上がりエッジの前に受信された最後の 40 ビットのみが、コマンドとして認識されます。ドライバは、40 以上の 8 の倍数のビット数からなる SPI 転送のみを受け入れます。

SPI のタイミング

SPI の最大周波数は 8MHz です。SCK はシステムのクロック周波数とは無関係ですが、クロック周波数に依存する唯一のパラメータが最小 CSN ハイ時間です。すべての SPI 入力は内部でフィルタ処理が行われるため、10ns より短いパルスでトリガされることはありません。図 7 に SPI バス・トランザクションのタイミング・パラメータを示します。タイミング値は電気的特性の表に示されています。

SPI は SPI モード 3 を使用します。その他のモードを用いた場合、信頼できるデータ転送が行われません。

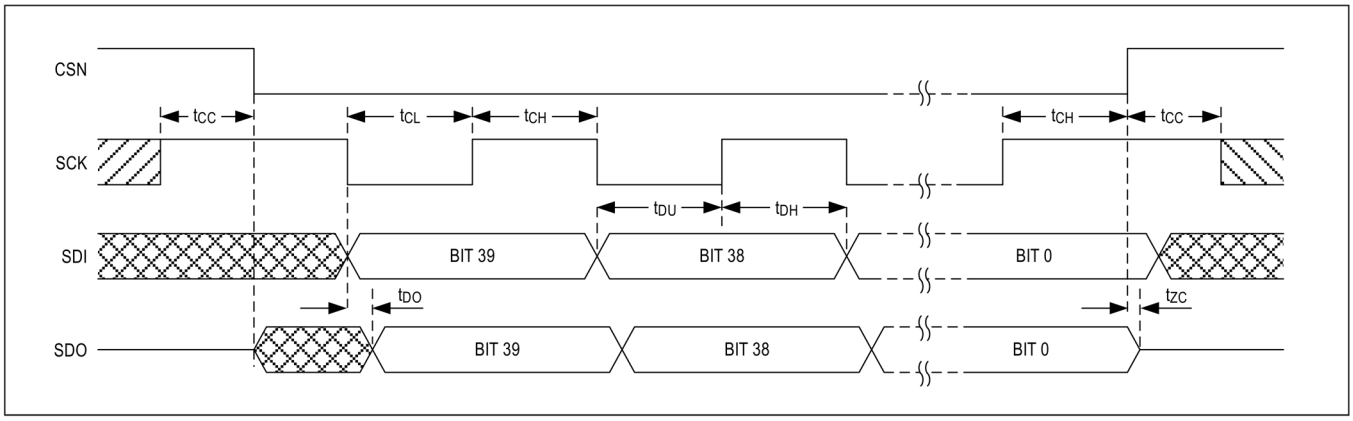


図 7. SPI のタイミング図

ステップおよび方向インターフェース

STEP および DIR 入力は、既存の多くのモーション・コントローラに対応できる、単純で標準的なインターフェースを備えています。MicroPlyer ステップ・パルス・インターポレータは、元来は粗いステッピング用に設計されたアプリケーションに対して、高分解能マイクロステッピングの滑らかなモーター動作を可能にします。

タイミング

図 8 は、STEP および DIR 信号のタイミング・パラメータを示しています。*CHOPCONF* レジスタの *dedge* モード・ビットをセットすると STEP 入力信号の両方のエッジがアクティブになります。*dedge* をクリアすると立上がりエッジだけがアクティブになります。STEP および DIR は、システム・クロックに対応してサンプリングと同期を行います。約 10ns の内蔵アナログ・フィルタが、長い PCB パターンによるグリッチなどの信号に存在するグリッチを除去します。信号源がチップから離れた場所にある場合、特に信号がケーブルを通じて伝送される場合は、信号を差動でフィルタリングまたは伝送する必要があります。

タイミング・パラメータの仕様については、電気的特性の表を参照してください。

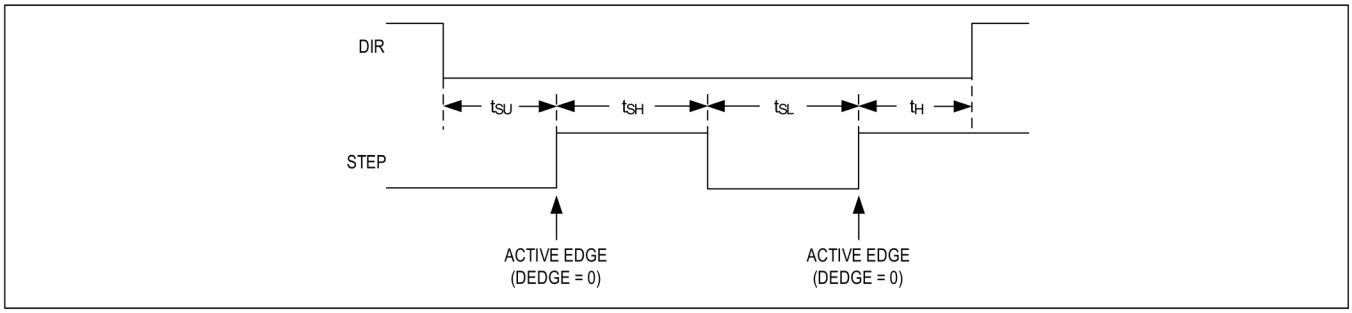


図 8. STEP および DIR 信号のタイミング

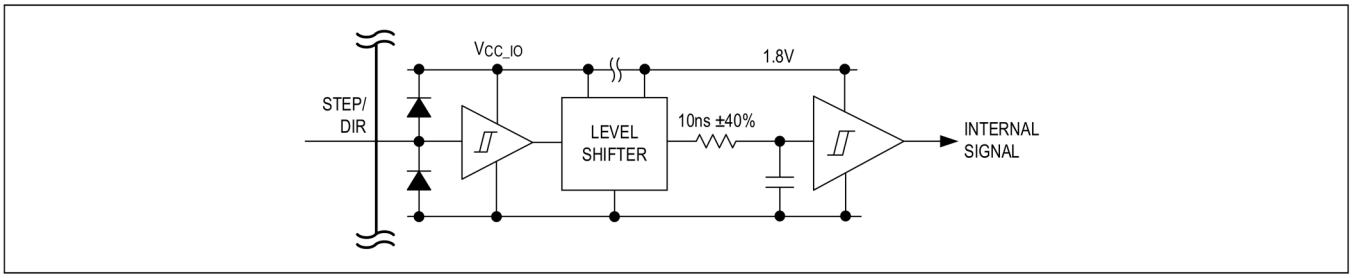


図 9. STEP および DIR 信号の入力フィルタ構造

分解能の変更

マイクロステップ分解能を低下させることで、STEP および DIR インターフェースのステップ周波数を制限したり、旧式で性能が劣るドライバに対応したりできます。1024 個のサイン波エントリを持つ内部マイクロステップ・テーブルは、サイン波のモーター・コイル電流を生成します。これら 1024 個のエントリは、1 回の電気的回転あるいは 4 つのフルステップに相当します。マイクロステップ分解能は、テーブル内で取られるステップ幅を決定します。DIR 入力に応じ、マイクロステップ・カウンタは STEP パルスごとにステップ幅だけ増加（DIR = 0）または減少（DIR = 1）します。マイクロステップ分解能はインクリメント量、およびデクリメント量を決定します。最大分解能の場合、シーケンサはステップ・パルスごとに 1 ステップ前進します。分解能が半分の場合、2 ステップ進みます。フルステップのインクリメントは最大 256 ステップです。シーケンサは、いつでも異なるマイクロステップ・レートにシームレスに切り替えることのできる、特別な機能を備えています。低マイクロステップ分解能に切り替わると、シーケンサは、目標分解能に最も近いステップを計算し、そのポジションでの電流ベクトルを読み出します。この動作は、フルステップやハーフステップなどの低分解能では特に重要です。その理由は、ステップ・シーケンス内での何らかの異常があると、時計方向と反時計方向に動作するモーターの比較時に非対称な動作の原因になるためです。

例：

フルステップ：テーブル・ポジション全体でのサイクル：128、384、640、896（45º、135º、225º、315º の電気的ポジション、両コイルが同一電流でオン）。各ポジションのコイル電流は、実効値（振幅の 0.71 倍）に対応します。ステップ・サイズは 256（電気的には 90º）です。

ハーフステップ：最初のテーブル・ポジションは 64（電気的には 22.5º）で、ステップ・サイズは 128（45º ステップ）です。

クォータ・ステップ：最初のテーブル・ポジションは 32（電気的には 90º/8＝11.25º）で、ステップ・サイズは 64（22.5º ステップ）です。

このようにして等距離のステップが生じ、これらはどちらの回転方向に対しても同じです。一部の旧式のドライバは、ステップ・テーブル内でゼロ電流（テーブル・エントリ 0、0º）とフル電流（90º）も使用します。この種のステップ動作はトルクが小さく、また、ドライバおよびモーターの消費電力も増加するため、回避されます。

表 6. 位相 A/B コイル電流のフルステップ／ハーフステップ・ルックアップ・テーブル値

STEP POSITION	TABLE POSITION	CURRENT COIL A	CURRENT COIL B
Half step 0	64	38.3%	92.4%
Fullstep 0	128	70.7%	70.7%
Half step 1	192	92.4%	38.3%
Half step 2	320	92.4%	-38.3%
Fullstep 1	384	70.7%	-70.7%
Half step 3	448	38.3%	-92.4%
Half step 4	576	-38.3%	-92.4%
Fullstep 2	640	-70.7%	-70.7%
Half step 5	704	-92.4%	-38.3%
Half step 6	832	-92.4%	38.3%
Fullstep 3	896	-70.7%	70.7%
Half step 7	960	-38.3%	92.4%

MicroPlyer ステップ・インターポレータおよび停止検出

STEP の各アクティブ・エッジにおいて、MicroPlyer は 256x の分解能でマイクロステップを生成します。これは、最後のステップ間隔に基づき、ステップ入力での 2 つのステップ・インパルス間の時間を補間します。したがって、2 マイクロステップ（256 マイクロステップに対して 128 マイクロステップの補間）から 256 マイクロステップ（256 マイクロステップに対するフルステップ入力）が単一ステップ・パルスに対して駆動されます。

MicroPlyer 機能は、*CHOPCONF* レジスタの *intpol* ビットによって有効にします。

補間された 2 マイクロステップ～256 マイクロステップに対するステップ・レートは、直前のステップ期間の時間間隔を測定し、それを最大 256 等分することで求められます。256 個のマイクロステップが均等に分布している場合、2 つのマイクロステップ間の最大時間は 2^{20}（およそ 100 万システム・クロック・サイクル）に相当します。その結果、システム・クロック周波数が 16MHz の場合、MicroPlyer 動作の最小ステップ入力周波数は 16Hz になります。ステップ・レートが低い場合は、静止イベントを示す *stst* ビットがセットされます。この周波数では、(システム・クロック周波数)/2^{16}（約 256Hz）のレートでマイクロステップが生じます。停止が検出されると、ドライバはモーター電流を自動的にホールド電流 IHOLD に切り替えます。

注：MicroPlyer は、安定した STEP 周波数でのみ完全に機能します。STEP 信号のデューティサイクルが 50%以外の場合は *dedge* オプションを使用しないでください。

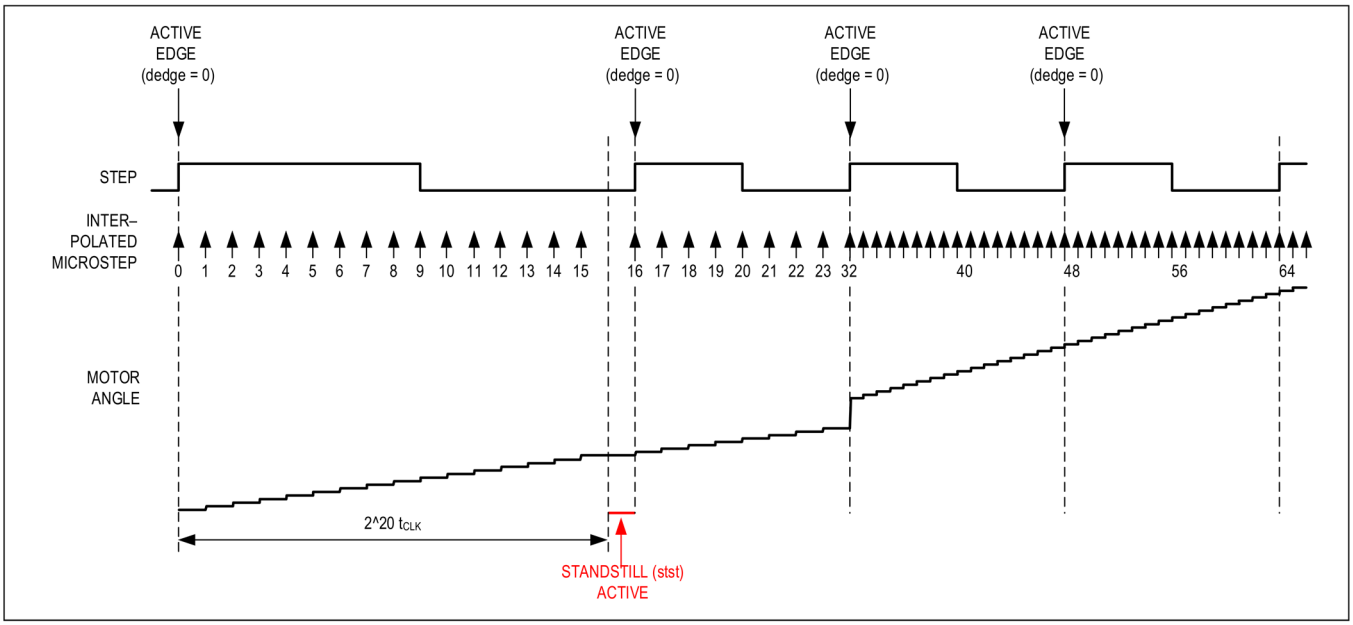


図 10. 立上がり STEP 周波数での MicroPlyer マイクロステップ補間（例：16～256）

図 10 では、最初の STEP サイクルは停止ビット *stst* を設定できるだけの長さになっています。このビットは、次の STEP アクティブ・エッジでクリアされます。次に、外部 STEP 周波数が増加します。高いレートで 1 サイクルが経過した後、MicroPlyer は補間されたマイクロステップ・レートをこのより高い周波数に適用します。より低いレートの最後のサイクルでは MicroPlyer が 16 個のマイクロステップすべてを生成するわけではありません。そのため、レートが高くなったときの最初のサイクルと 2 番目のサイクルの間でモーター角にわずかな跳びが生じます。

StealthChop+

StealthChop+は、静音の電圧制御モーター・チョッパー原理であり、高精度のセンサーレス負荷測定、共振減衰、優れたエネルギー効率という特長をそのコンセプトに取り込んでいると同時に、最高のポジション精度を実現しています。

StealthChop+は、ランプ・ジェネレータまたはStepおよびDir入力をモーター・ポジションの厳格なセット・ポイントとして用いることにより、センサーレスのフィードバックに基づく最適化手法と、従来の使いやすいフィードフォワード型のステッピング・モーター制御を組み合わせています。StealthChop+は、モーターの機械的な負荷を自動的に判定し、その機械的負荷にモーター電流のフィッティングを適応させ、モーター電流の電気的角度を強制的に設けることによって機械的な共振を処理します。

駆動システムが非常に効率的であるため、モーターを高温にならないようにすることにより、より小型で低コストのモーターを使用してピーク・トルク需要を満たすことができます。それにより、逆に、ピーク・パワーを上げることが必要な場合にモーターを短時間オーバードライブするためのヘッドルームが生まれます。また、小型のモーターを選択することで、モーターの相対的なコギング・トルクが減少し、より静かな動作が実現します。

減速オプションを用いると、モーターの運動エネルギーを電源に効率的に戻すことや、エネルギーのフィードバックを減少させて過電圧保護に対する要求を最小限に抑えることができます。

利点

高効率、低ノイズ、低振動：

- モーターは滑らかかつ静かに動作。
- 共振の減衰によりシステムの機械的なノイズを最小限に抑制。
- 高効率であるため、小型モーターの使用が可能。

システム・コストの低減：

- 平均モーター電流が少ないため、モーターをオーバードライブして高ピーク・トルクの供給が可能。
- モーターとドライバの動作温度が低いため、冷却および電源の必要性が低下。
- 共振が少ないため、機械的なシステム設計の負荷が軽減。

センサーレスの負荷測定：

- センサーレス負荷測定により、ホーミングおよびステップロス検出がセンサーレスで可能。
- 正確な負荷測定により、予防保全が可能。
- 低速のストール検出により、動作範囲が微小なアプリケーションでの使用が可能。

使いやすさおよび信頼性：

- 再パラメータ化を必要とせずに、1 つのパラメータ・セットを使って、モーター速度、電流、温度、電源電圧範囲をすべてカバーする、使いやすく安定した確定的なパラメータ化。
- コイル抵抗測定を用いて自動補償を行い、モーター温度の決定が可能。
- 通常はフル速度の数パーセントからフル・システム速度（電源電圧レベルに達するモーター駆動電圧によって決定）まで、更には弱め界磁領域にまで達する、高ダイナミック・レンジをセンサーレス機能が提供。

StealthChop+の動作原理

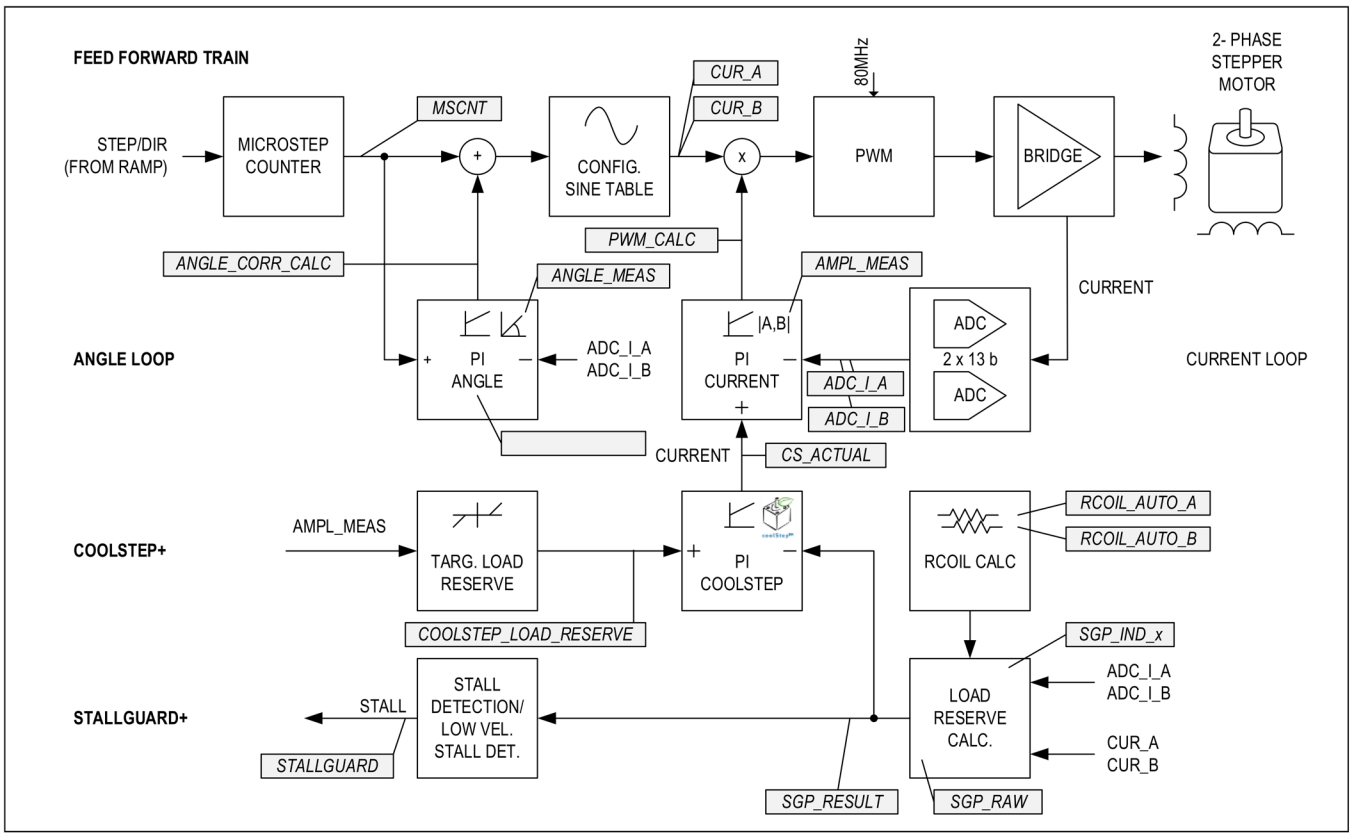


図 11. StealthChop+のレギュレータ方式

StealthChop+は、ステッピング・モーター制御に関連する、従来型のフィードフォワード列を用いて、コマンドで指定されたモーション・プロファイルをモーター・ポジションが直接追随するようにします。

高速の加速および減速プロファイル時やモーター・ストール時にも、デュアルのアナログ・デジタル・コンバータ（ADC）を備える電流制御ループが、完璧なモーター電流トラッキングを実現します。

角度制御ループは、モーターのステータ・フィールドがマイクロステップ・カウンタ・ポジションに揃った状態を保持します。それによって、モーターの発振が減衰され、コマンド指定された動作に対する、速度に依存した位相シフトによるモーターの遅延が回避されます。更には、角度制御ループにより、直接電流制御された SpreadCycle チョッパーに滑らかに切り替えることができます。

StallGuard+は、実際のモーター動作点での負荷余裕（使用可能な未使用トルクを表す指標）を計算することにより、機械的な負荷を追跡し続けます。StallGuard+は、負荷余裕をモニタし、センサーレスのストール検出を可能にします。

CoolStep+は、負荷余裕を追跡し続け、安全で安定した動作点にとどまったまま、モーター電流を実際の負荷要求に合うよう調整します。それにより、従来の純粋なフィードフォワード動作に比べ、著しくエネルギーの節約ができます。

µDcStep は CoolStep+と共に機能し、モーター速度を DC モーターのように動的に適応させステップロスを防止します。モーター・トルクが負荷を駆動するには不十分になった場合、バック EMF の減少により、高いトルク出力の動作点までモーター速度が自動的に減少します。

StealthChop+プロシージャのイネーブル

StealthChop+には一連のデフォルト・パラメータが付属しており、多くのモーターで開始点として使用できます。StealthChop+をイネーブルするのに必要なステップはわずかです。

表 7. StealthChop プロシージャのイネーブル

番号	タスク	詳細
1	モーター電流を設定	モーターの公称電流に一致するモーター電流設定値（*DRVCONF*、*IRUN*）を設定し、停止トルクに応じて *IHOLD* を設定します。256 のフル・マイクロステップ動作で動作します。または、分解能を下げ、マイクロステップ・インターポレーションをイネーブルして動作します。オプションで、モーターに一致するマイクロステップ・テーブルを最適化します（パラメータは、StealthChop+動作中にも決定できます）。
2	モーター・コイルのインダクタンスを設定	モーターのデータシートから取得したコイル・インダクタンスの値を *COIL_INDUCT* = <μH 単位のコイル・インダクタンス>に入力します（例えば、コイル・インダクタンスが 8mH のモーターの場合 8000）。
3	StealthChop+をイネーブル	*GCONF.en_stealthchop* を設定します。
4	速度スレッショルドを選択	CoolStep+には、約 0.25RPS～1RPS の低めの速度スレッショルドを選択します。1.8º のモーターの場合、0.3RPS はおよそ *TCOOLTHRS* = 1000 です（16MHz のクロック周波数時に 2 つの 1/256 ステップの間に 1000 クロック・サイクル）。
5	削減率を選択	電流低減係数 *COOL_CUR_DIV* を 2（1/2）～5（1/5）の範囲にプログラムすることにより、電力削減のレベルを選択します（詳細なチューニングを用いて最大 10 通りが可能）。*IRUN* および *COOL_CUR_DIV* を選択する場合、*IRUN* を *COOL_CUR_DIV* でスケール・ダウンすると動作時の電流が減少することを考慮してください。最善の結果を得るには、除算後の電流は 35 以上の値を保持していることが必要です。それより電流が小さい場合、デフォルトの角度レギュレータがディスエーブルされるためです。
6	ドライバをイネーブル	*CHOPCONF.TOFF* = 3（または 2～15 の範囲の任意の値）と設定することによりモーターをイネーブルします。
7	コイル抵抗を測定するための初期動作	最大 4000 マイクロステップ／秒（*T_RCOIL_MEAS* = 4096 の場合のデフォルト値に一致）の速度で少なくとも 1 フルステップ分、モーターを回転します。それによって、ドライバは、StealthChop+に必要なモーター・コイル抵抗を最初に測定できます。*DRV_CONF* の設定値が変更された場合はこのステップを繰り返します。
8	CoolStep の速度内でモーターを操作	モーターを 1RPS を超える速度または 50000 マイクロステップ／秒（以上）の速度に加速し、モーターを CoolStep+動作に切り替えます。

このプロシージャは、ここでは変更されないその他の設定に依存します。PI レギュレータを最も安定させるために、デフォルト設定では控え目の値を用いています。更に詳細な設定や調整を行うことで、モーターの性能を最大限に発揮できます。

注：SpreadCycle とは異なり、*IRUN* および *IHOLD* の値が 251～255 の場合、内部 AD コンバータを用いて安定化を行うための十分なヘッドルームを確保するために、これらの値は内部で 250 に切り詰められます。

StealthChop+の詳細な調整を行う理由

StealthChop+は、多くのパラメータに対し実用的なデフォルト値を設けていますが、特定の状況においては、パラメータを調整することによって以下の特性が改善できます。

- 負荷の急変時でのモーター電流および使用可能トルクの安定性
- モーターの効率
- 共振の減衰

単一の「優れた」設定というものはないことを知っておくのは重要ですが、パラメータが機能制限に近い場合を除き、StealthChop+レギュレータは、広い範囲の設定値で良好に動作します。そのため、各パラメータの有効な範囲を知り、設定値をその範囲の中央に設けるのは、理に適っています。

StealthChop+電流レギュレータの設定

StealthChop+は、PI レギュレータを用いてモーター電流を制御しています。通常、デフォルト値を用いることにより電流レギュレーションが良好に行われますが、詳細な調整を行うと電流の安定性を最適化できます。モーター適応型の設定を選択すると、高速のモーター加速時のアンダーシュートによるトルクの減少や、モーター電流のオーバーシュートによる減速時の電源電流の増加を防ぐことができます。

電流レギュレータは、実際の動作電流 *CS_ACTUAL*（*IRUN* または、*COOL_CUR_DIV* でスケーリングされた *IRUN* によって制御）に基づいて動作します。この値は 8 倍され、その値が、両方の相電流の ADC 測定値から計算した一時的な実効値電流の大きさ（*AMPL_MEAS*）と比較されます。2000 という大きさが測定された場合、StealtChop+の *IRUN* の上限値 250 に相当します。

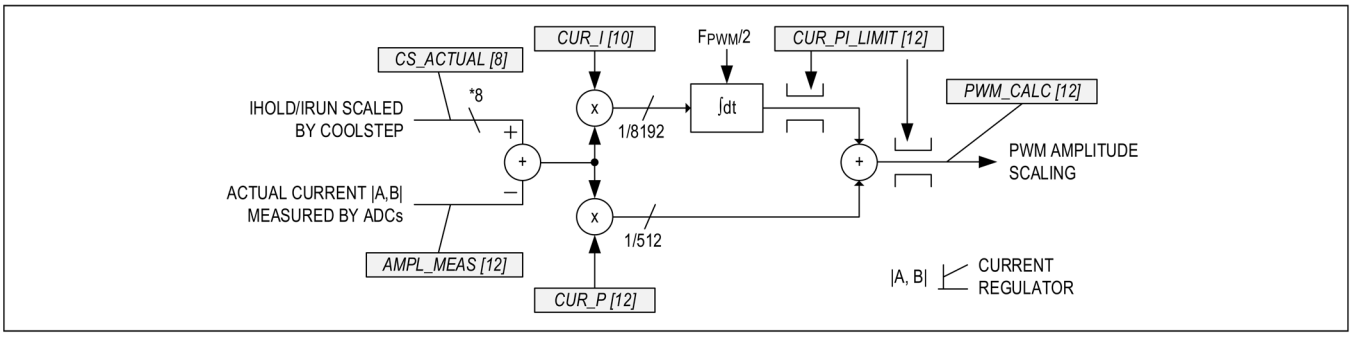


図 12. StealthChop+電流レギュレータ

電流レギュレータのパラメータを調整する目的は、特に、必要な最大加速および減速時に、モーター・ドライバの PWM 振幅がモーター速度の変化に十分高速に応答する、安定したレギュレーションを実現することです。

必要な動作電圧を使用して調整してください。電圧が大きく変化する可能性がある場合は、必要な最大動作電圧での安定性をチェックします。振幅の変化が同じでも、電源電圧が高くなるとモーター電流の変化も大きくなるためです。電流ステップ応答時間を加速するには、P 係数 *CUR_P* を増やします。発振が生じる場合には、P 係数を臨界値の半分以下に減少させます。P の部分は、電流レギュレーションの 50%～95%を補うことができますが、必ずしもそうする必要はありません。I 係数 *CUR_I* は、長い時間スケールで正確にセット・ポイントに安定化させる役割を担います。

CUR_P を 512 に設定した場合、これは、電流範囲と電源電圧範囲の比が 1.0 となることに相当します。つまり、モーター電流が 1%変化すると、これに対応して PWM のデューティサイクルが 1%変化します。*CUR_P* = 512 では、実際の測定電流が 0 の場合、ドライバは PWM 振幅を最大値（サイン波ピークの 96%のデューティサイクル）まで増加させます。96%という内部 PWM デューティサイクル制限値（デフォルトのサイン波を使用）は、モーターの実効値電圧として供給される電源電圧の 68%に相当します（サイン波の実効値振幅とピークは 1/SQRT(2)の係数分だけ異なるため）。

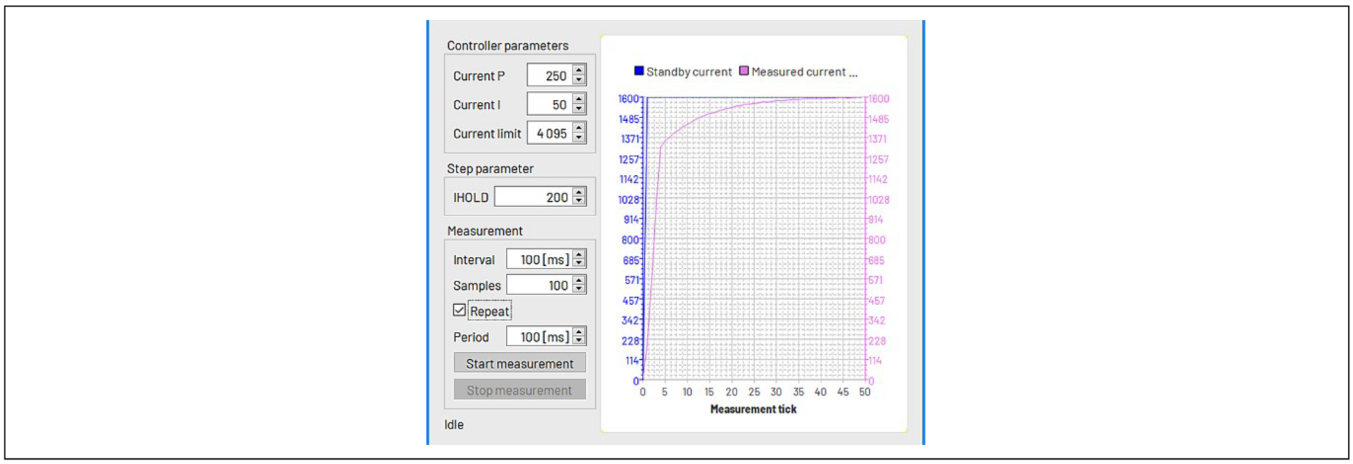


図 13. IHOLD 電流を 0 から増加させた場合の電流ステップ応答のモニタリングによる、電流レギュレータの調整

電流レギュレータの P 調整

CUR_P に対する最初の安定した設定値は、モーター・コイルの抵抗および選択した電流範囲に基づいて計算できます。その原理をここに示します。ただし、アプリケーションに対する最適設定値は、応答時間を改善するためにこれより大きな値にすることができます。あるいは、測定ノイズに起因するレギュレーション・リップルが低減されるためにこれより小さい値にすることもできます。

CUR_P を最適化するための開始値として、50%の P 部レギュレーションを計算します。

$$CUR_P_{50} = 512 \times \frac{R_{COIL}[\Omega]}{0.68\ VS[V]} \times CurrentRange[A]RMS$$

ここで、

- CurrentRange は、設定された、電流実効値フル・スケール設定値 ($\frac{peak\ current}{\sqrt{2}} \approx 0.71 \times peak\ current$) に相当し、レジスタ *DRV_CONF* の *CURRENT_RANGE* および *CURRENT_RANGE_SCALE* によって選択されます。
- V_Sは、モーター・ドライバの電源電圧（アプリケーションでの公称値）です。
- 係数 0.68 は、$\frac{248}{256} \times 0.71$ のマイクロステップ・サイン波ピークによるものです。
- R_{COIL} は、公称モーター・コイル抵抗で、モーターのデータシートから取得するか、R_COIL_AUTO によって測定して Ω に変換します。

非常に高速の加速および減速を伴うアプリケーションでは、*CUR_P*$_{50}$、または、この値の 5～20 倍までの値を使用し、P 値を用いる目標電流レギュレーションの 80%～95%を補うようにできます。設定値が高すぎると、不安定やレギュレーション・ノイズの原因になります。

モーターの加速や減速が低い場合または中程度の場合は、*CUR_P* の値を下げて *CUR_P*$_{50}$ の 20%～100%の範囲にすることを推奨します。そうすることにより、ADC の測定ノイズの影響を低減できるためです。レギュレーションの大半を補うよう、I 部分を調整してください。

電流レギュレータの I 調整

CUR_P を決定した後、積分部分である *CUR_I* を選択して、発振の可能性がない、安定した電流レギュレーションを実現する必要があります。小さな P 値を用いる場合、I 値を大きくすると、モーターの加速または減速が大きい場合の電流レギュレーションの応答を向上するのに有用です。

適切な開始値は、*CUR_P* 用に選択した数値の 25%です。電流差の大半（例えば 80%以上）をカバーするよう *CUR_P* が選択されている場合、*CUR_I* を *CUR_P* の 10%～100%の範囲で最適化します。*CUR_P* が低い設定値（例えば、*CUR_P*$_{50}$ の計算値の 2 倍未満）に選択されている場合、*CUR_I* を増加して、レギュレーション差の大半をカバーするようにします。この場合、*CUR_I* を、実際の *CUR_P* 設定値の最大 5 倍に設定します。*CUR_P* および *CUR_I* には、様々な数値範囲および上限値を考慮してください。

モーターの加速時や減速時、あるいは、*IRUN* や *IHOLD* を用いて目標電流設定値を変更する場合、電流レギュレーションの安定性をモニタしてください。加速フェーズの間に電流が数パーセント以上低下する場合は、応答が速くなるよう、*CUR_I* を増加します。発振が生じる場合は、その臨界値の 70%以下に減少します。

電流定格がモーターの定格を超える可能性のある実験時など、特殊な状況では、その結果生じるデューティサイクルを *CUR_PI_LIMIT* 設定を用いて制限できます。この場合、モーター・コイルの電源電圧に対する割合が最大目標値となるよう、*CUR_PI_LIMIT* を設定します。つまり、409 に設定するとデューティサイクルが 10%に制限され、それによって、実効的な電圧は、IC の電源電圧の最大 10%になります。通常動作の場合、制限値は 4095 のフル・スケール値のままにします。

過電流がトリガされるのを防ぐため、レギュレータの I 部分は、チップ・イネーブル時、StallGuard からトリガされるハード・ストップの実行時、または、ストップ・スイッチや仮想ストップ・イベントにより、自動的にクリアされます。

注：ほとんどの場合、電流レギュレータの PI 係数の選択は重要ではなく、広い範囲のパラメータが有効です。加速度が大きい（著しいバック EMF が生じる速度まで数ミリ秒以内でモーターが加速する）アプリケーションでは、PI 電流レギュレータ係数を増加させて、電流の一時的な低下に伴うトルクの低下を防ぐ必要があります。この場合は、係数の調整が必須です。

全体的に低いパラメータを設定すると、IC は、モーターのバック EMF の非サイン波形による電流歪みに対応しようとします。これは、特定の速度範囲において、高トルク・モーターや永久磁石モーターの滑らかさに影響することがあります。レギュレータのパラメータ設定には、小さい値のパラメータを試してみてください。

電流レギュレータの PI 調整のための大まかな指針

$$CUR_P_{50} = 512 \times \frac{R_{COIL}[\Omega]}{0.68\ VS[V]} \times CurrentRange[A\ RMS]$$

CUR_P を選択します：

低加速の場合、$CUR_P = CUR_P_{50}$ の 20%～100%。

高加速の場合、$CUR_P = 2 \ldots 20 \times CUR_P_{50}$。

CUR_I を選択します：

理に適った最適化範囲は、$CUR_I = 10\%\ CUR_P$～MIN。$(5 \times CUR_P_{50};\ 5 \times CUR_P)$：

CUR_P < 2 × CUR_P_{50} の場合、*CUR_P* の 10%～500%の範囲の *CUR_I* を選択します。

CUR_P > 2 × CUR_P_{50} の場合、*CUR_P* の 10%～100%の範囲の *CUR_I* を選択します。

電流レギュレータの PI 調整の例

コイル抵抗が 5Ω のモーターが、$1A_{RMS}$ の電流レンジにおいて 24V で動作します。

$$CUR_P_{50} = 512 \times \frac{5\Omega}{0.68 \times 24V} \times 1A = 157$$

この設定では、最初のステップの P 部分は任意の電流偏差を 1:1 で補正し、それによって、目標電流のおよそ 50%に安定化します。多くの場合、この設定で十分です。

このモーターの例では、以下の選択が可能です。

低速の低ノイズ電流レギュレーション：$CUR_P = ½\ CUR_P_{50} \approx 75$、$CUR_I = CUR_P = 75$

高速電流レギュレーション（I 部を使用）：$CUR_P = CUR_P_{50} = 157$、$CUR_I = 2 \times CUR_P_{50} = 314$

非常に高速の電流／速度ステップ応答：$CUR_P = 10 \times 157 = 1570$、$CUR_I = 0.25 \times 1570 = 392$

オシロスコープを用いた CUR_P および CUR_I の調整

電流プローブを使用したオシロスコープは、電流 PI レギュレータの動作を効果的に示します。反復的な手法を用いて、*CUR_P* と *CUR_I* を決定できます。この例では、700mA RMS/1A peak で NEMA 17 1A モーターを動作させた場合のコイル電流を示しています。初期設定値から始まって、パラメータが段階的に改善されています。モーターは停止状態（電流を低値に減少）から急速に 120RPM（AMAX = DMAX = 63000）まで加速し、15000 マイクロステップ（約 58 フルステップ）後に急速に減速します。

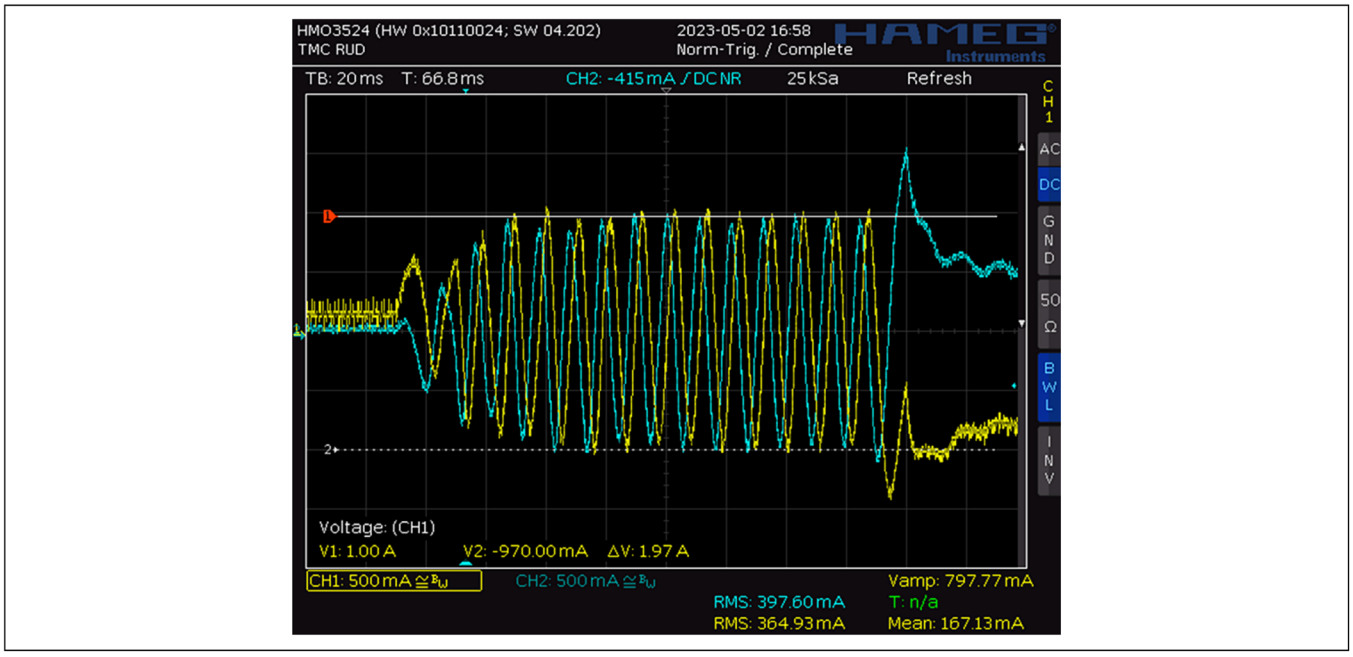


図 14. CUR_P（50% CUR_P_{50}）と CUR_I（25% CUR_P_{50}）が小さすぎる場合：緩やかな電流ランプアップ、減速時にオーバーシュート

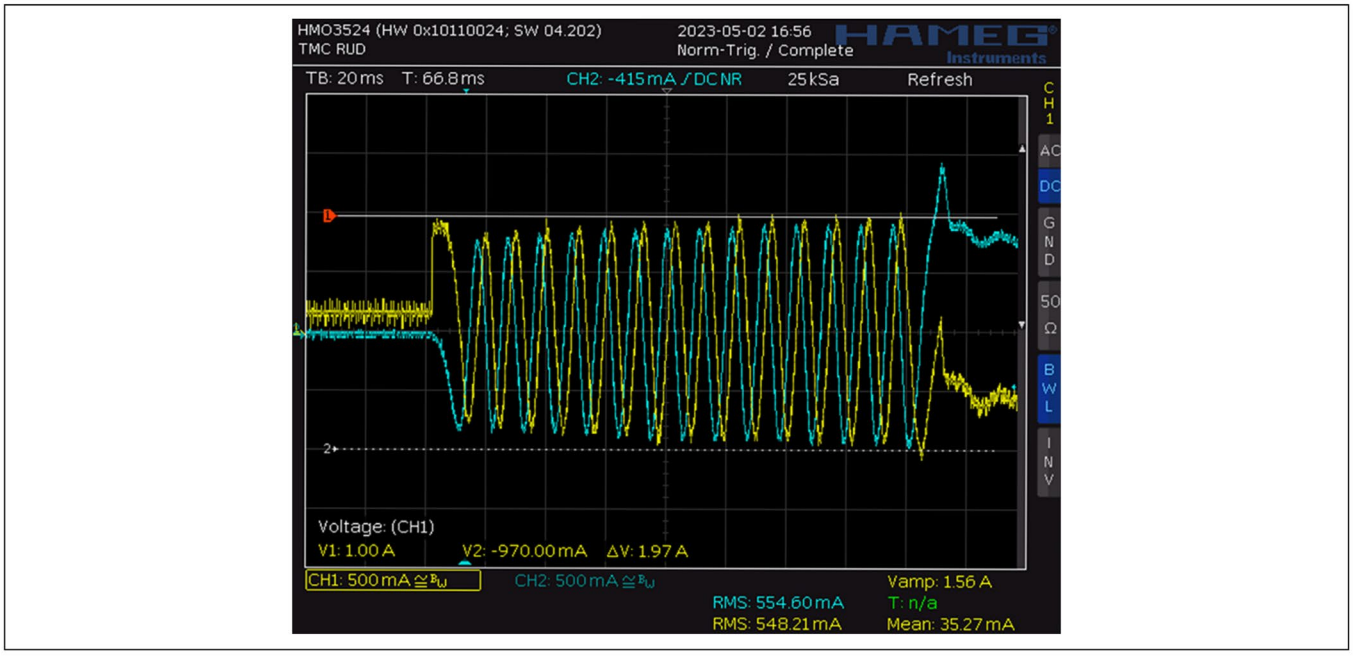


図 15. 大きな CUR_P（10 × CUR_P_{50}）設定により、起動時と停止時に高速の応答。
CUR_I が小さい（CUR_P の 10%）ため、残留している差異は緩やかにのみ補正。

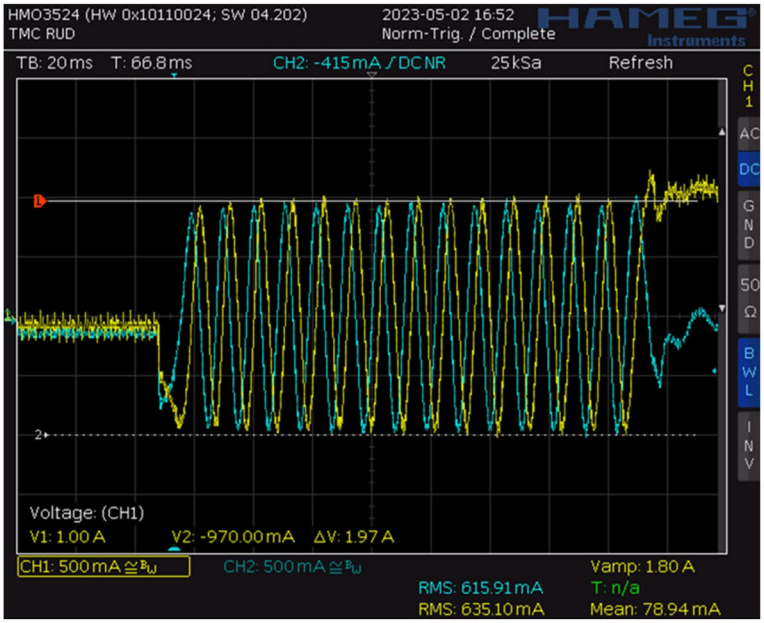


図 16. CUR_I（CUR_P の 25%）を増加させた大きな CUR_P（10 × CUR_P_{50}）設定により、セット・ポイントまで短時間で電流を安定化

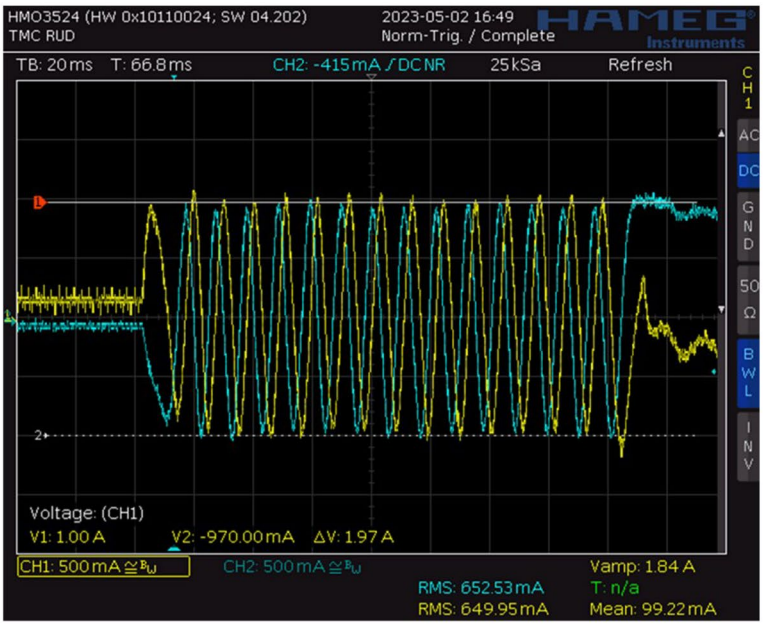


図 17. CUR_I（2 × CUR_P_{50}）を増加させた CUR_P = CUR_P_{50} 設定においても、
少ないオーバーシュートおよびアンダーシュートでセット・ポイントまで短時間で電流を安定化

StealthChop+角度レギュレータの設定

StealthChop+は、2 つ目の PI レギュレータを用いてモーター電流の角度を制御しています。角度のレギュレーションでは、電流の角度をマイクロステップ・カウンタからコマンド指定された角度に揃えます。角度制御により、電界の位相シフトによるローター遅延をなくし、ローターの磁界がコマンド指定されたセット・ポイントを中心に発振することを防止できます。これは内蔵されている共振減衰機構の一部です。デフォルトのパラメータは通常、電流角度を揃え、低周波数の発振を十分に抑えるのに役立ちます。

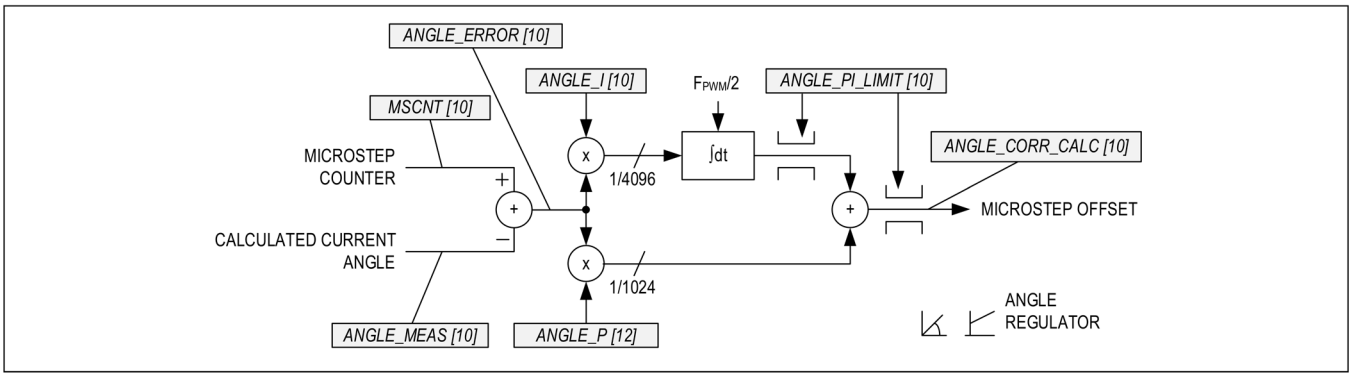


図 18. StealthChop+角度レギュレータ

角度レギュレータのパラメータを調整する目的は、速度、機械的負荷、モーターの共振などに起因するモーターのコイル電圧とコイル電流の間の位相シフトの変化に対して、レギュレータの十分に高速な応答が可能な安定的なレギュレーションを実現することです。

角度レギュレータの応用範囲

角度レギュレータは、*T_RCOIL_MEAS* で設定されているスレッショルドを速度が超過すると必ず、アクティブになります。このスレッショルドは、非常に低速（例えば数 RPM）に設定されます。この場合、モーターは、バック EMF の影響を無視でき、そのため、角度の偏差を無視できる、抵抗性の特性を主に示します。

角度レギュレーションの下限電流値は、*ANGLE_LOWER_I_LIMIT* によって規定されます。それにより、非常に小さなモーター電流での角度補正をディスエーブルできます。モーター電流が非常に小さく電気的角度を正確に決定できない場合は、角度計算が不正確になり、ジッタが生じる可能性があるためです。*AMPL_MEAS* が *ANGLE_LOWER_I_LIMIT* 未満に低下した場合、角度レギュレーションはディスエーブルされ、マイクロステップのオフセットは最後の有効な値に固定されます。*IRUN* が 250 の場合、*AMPL_MEAS* が 2000 の場合に相当するため、*ANGLE_LOWER_I_LIMIT* = 256 というデフォルト値は、*IRUN* が 33 の場合に相当します。*COOL_CUR_DIV* 設定値を非常に大きくする必要がある場合を除き、このデフォルト値を保持し、*IRUN* をおよそ 35 未満にスケール・ダウンしてください。このスケールは ADC 分解能の *PWM_MEAS* と同じです。

コイル・インダクタンスとバック EMF により、モーター電流はモーター電圧より最大 90º 遅延します。*ANGLE_PI_LIMIT* のデフォルト設定値は 256 なので、予想される最大偏差である 90º までの角度補正が可能です。*ANGLE_CORR_CALC* は、計算した実際の補正値を示す、符号付きの値です。

調整は、最適化の目標に応じて、アプリケーションの速度範囲における共振周波数（これは、無負荷のモーターの場合に通常 50Hz～150Hz のフルステップ周波数で発生する最初の共振周波数の倍数です）、または、目的の StealthChop+の高速または最大のモーター速度での共振周波数で実行できます。モーター電流設定とその機械的負荷によって共振周波数はシフトするため、最善の結果を得るには、現実のアプリケーション・セットアップにおいて調整を行う必要があります。

角度の P 調整

実際の角度偏差と角度ジッタを示す *ANGLE_ERROR* を追跡します。目的の機械的セットアップを用いて最善の効果（最低のモーター振動、最も静かな動作）および安定性が得られることをチェックします。負荷ステップ応答時間を加速するには、P 係数 *ANGLE_P* を増やします。発振が生じる場合には、P 係数を臨界値の半分以下に減少させます。I 係数 *ANGLE_I* は、長い時間スケールで正確にセット・ポイントに安定化させる役割を担います。全体的な応答時間は、機械的な発振周期よりも短い時間になるよう調整でき、通常は 10ms 程度です。

低い *ANGLE_P* 設定値（例えば 50）および低い *ANGLE_I* 設定値（例えば 10）で開始し、*ANGLE_P* を増加させ、*ANGLE_ERROR* 信号の振幅が低下し設定を上げても大きな改善がなくなるまで続けます。*ANGLE_P* に 1024 を設定すると、P 部分による角度誤差の補正が 50%であることに相当します。3000 まで設定を上げると、モーターの中間領域での共振を減衰するのに有効です。大まかな指針として、50～250 の範囲に設定するのが最も一般的です。これより低い値を使用すると、角度計算のノイズ（モーター電流の ADC 測定に起因）を防止できますが、低速度範囲でのモーターの振動が発生します。

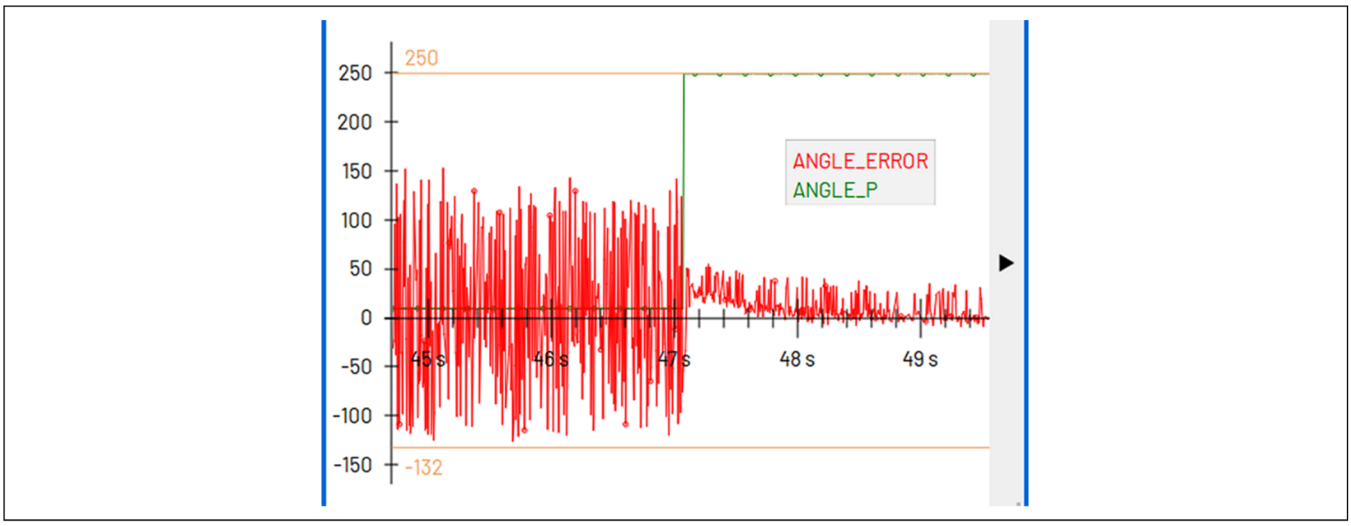


図 19. モーターの中間領域での共振（停止ポイント付近）が ANGLE_P を 250 に増加することにより解決

角度の I 調整

モーター負荷の変化による角度誤差に対し、10～100 の設定値で既に十分に応答します。設定をこれより高くすると、中間領域での共振に対処できますが、高い *ANGLE_P* を使用しないと、レギュレーションが不安定になる可能性があります。アプリケーションの速度が高い場合は安定性をチェックしてください。高加速で動作している場合、*ANGLE_I* の値を高くすると、応答時間が改善され、モーターの起動／停止時または反転時、あるいは SpreadCycle 動作への切り替え時の角度ジャークを防止できます。オプションで、高 *ANGLE_P* の代わりに、中程度の *ANGLE_P* と値を増加した *ANGLE_I* を組み合わせることで、中間領域での共振の減衰に役立つ可能性があります。

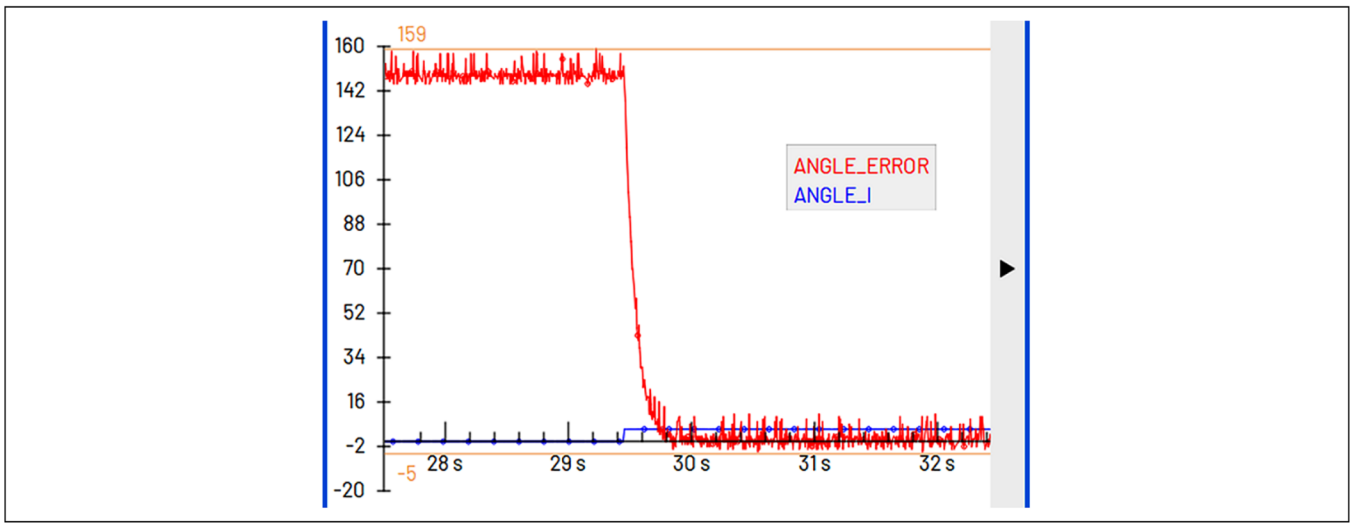


図 20. ANGLE_I を 0 から任意の値に切り替えることで、ANGLE_ERROR を±0 にする補正を開始

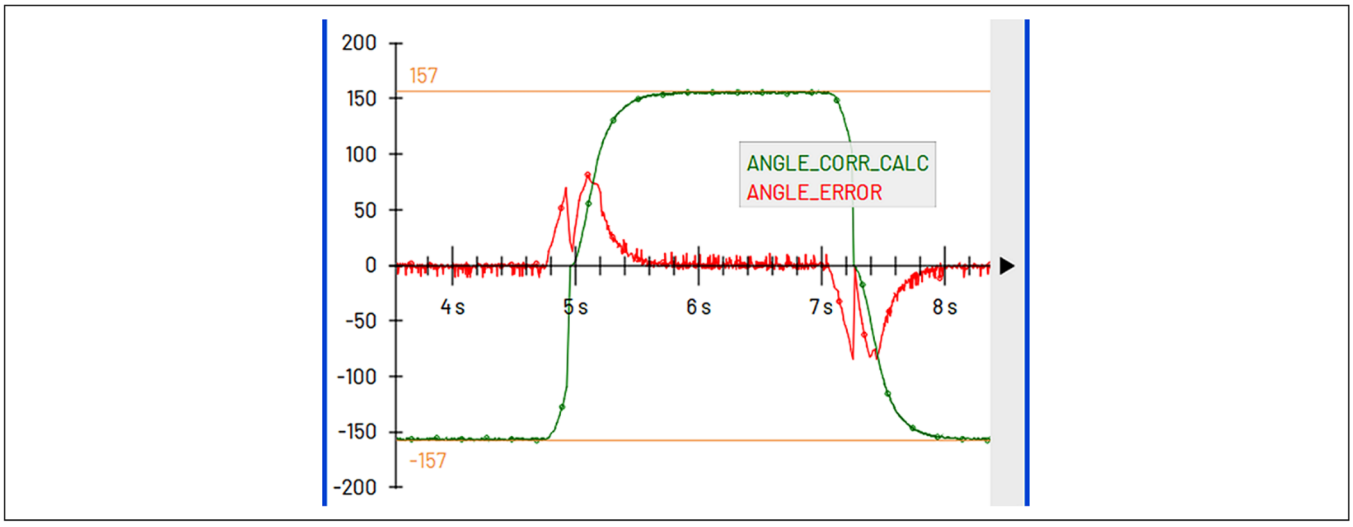


図 21. モーターの回転方向を 2 回反転。過度に低い値の ANGLE_I で反転させると適応が遅くなるため位相ジャークが発生します。
図におけるくぼみは、T_RCOIL_MEAS によって与えられた制限値未満の速度において ANGLE_ERROR が 0 にクリアされたことを示します。

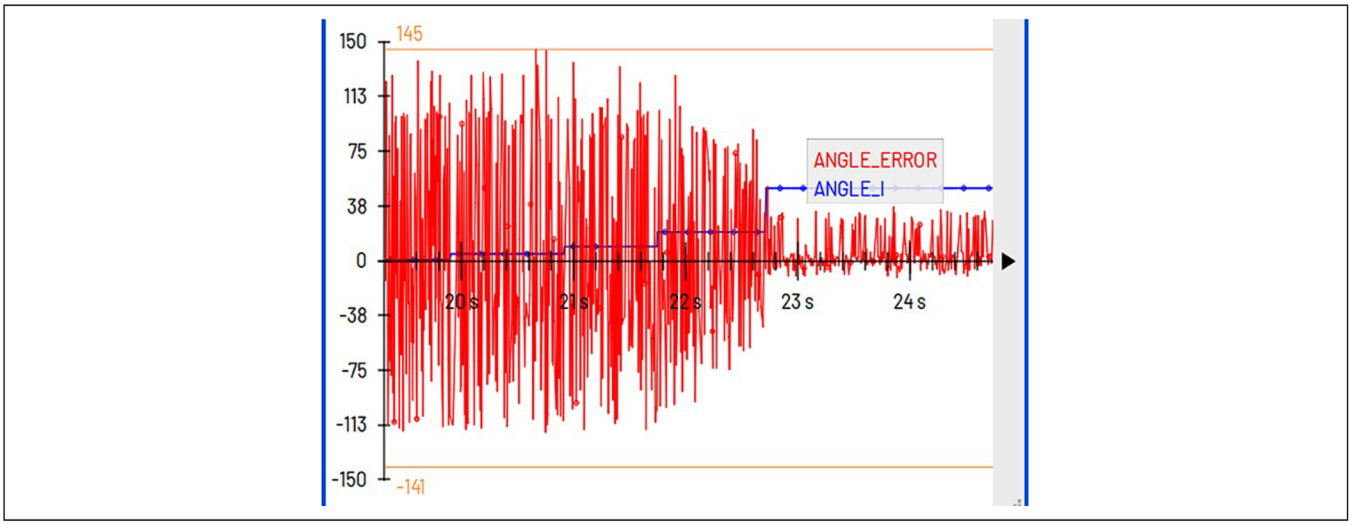


図 22. ANGLE_I の値を低値から 50 まで増加させることにより中間領域での共振が減衰

角度 PI の大まかな指針

ANGLE_P = 1024 に設定すると、角度誤差を 50%、直接補正します。それによって高速応答が可能ですが、場合によってはモーターの挙動が粗くなる可能性があります。

ANGLE_P を選択：

標準的な場合：50～250 の範囲の値を使用し、振動が最小となるよう最適化します。

中間領域での共振の最適化：250～3000 の設定を試します。安定性を慎重にチェックします。

ANGLE_I を選択：

標準的な場合：25～250 の範囲に設定し、高設定の場合は安定性を慎重にチェックします。中間領域での共振の最適化：100 から *ANGLE_P* と同じ値までの設定を試します。

StealthChop+チョッパーのオプション

StealthChop+チョッパーには、この他にも高度なオプションが多数あります。

チョッパー周波数

チョッパー周波数は、PWM_FREQ 設定を用いて、20kHz～60kHz の範囲を 8 つのステップで制御できます。ステップ幅はおよそ 5kHz です。20kHz は既に可聴範囲外であるため、通常はこの周波数で動作させれば十分です。周波数を上げることは、インダクタンスが非常に低い小型のモーターの場合には有益な場合があります。インダクタンスが低いことによる電流リップルを低減できるためです。反対に、高い周波数を選択すると、スイッチング・イベントが増加するため、ドライバの動的なスイッチング損失がわずかに増加します。スイッチング周波数を増加させる場合には、ドライバの勾配制御 *SLOPE_CONTROL* を最大値に設定する必要があります。

SD_ON_MEAS_LO および *SD_ON_MEAS_HI* は、チョッパーのデューティサイクルにスレッショルドを設定することによって、電流測定のタイミングを制御します。通常、電流は、両方のモーター・コイルの接続が GND レベルになっているとき、いわゆる低速減衰フェーズで測定されます。それによって、非常に低デューティサイクルまで安定した測定が確保されます。デューティサイクルが増加すると使用可能な測定時間が短くなるため、ひとたび使用可能な時間が電流測定のブランキングに必要な時間（SpreadCycle の *TBL* 設定に一致）未満に低下すると、ドライバは、モーター・コイルがオンの状態時に測定へ切り替わらなくてはなりません。この切り替えは、通常、PWM デューティサイクルがサイン波ピークにおける制限値に近づいているほどの高速でモーターが回転しているときに発生します。高いデューティサイクル・スレッショルド（デフォルト）が最善です。最高のマイクロステップ分解能が要求される、低速度動作時の時間の測定ポイントの切り替えを防止できるためです。更に、周波数が高くなると測定時間が短くなるため、より高いチョッパー周波数設定値に見合うようデューティサイクル・スレッショルドを適応させる必要があります。スイッチング・イベントのたびに、モーター電流が長く変動したり、大きなリップルを示したりする場合、モーターを 4095 付近の PWM デューティサイクル（*PWM_CALC*）で動作させると、角度リップルが増加する、つまり、StallGuard+信号が不安定になる可能性があります。この状況は、モーター接続の LC フィルタリングおよび長いモーター・ケーブルに付随して発生する可能性があります。これを改善するには、*SD_ON_MEAS_LO* および *SD_ON_MEAS_HI* の値を 1 または 2 だけ小さくします。

表 8. PWM 周波数の選択オプション

PWM_FREQ	FREQUENCY [kHz]	*SD_ON_MEAS_LO*	*SD_ON_MEAS_HI*
0	19.5	14	15
1	24.4	14	15
2	29.3	14	15
3	34.2	13	14
4	39.1	13	14
5	44.0	13	14
6	48.8	12	13
7	53.7	12	13
8	58.6	12	13

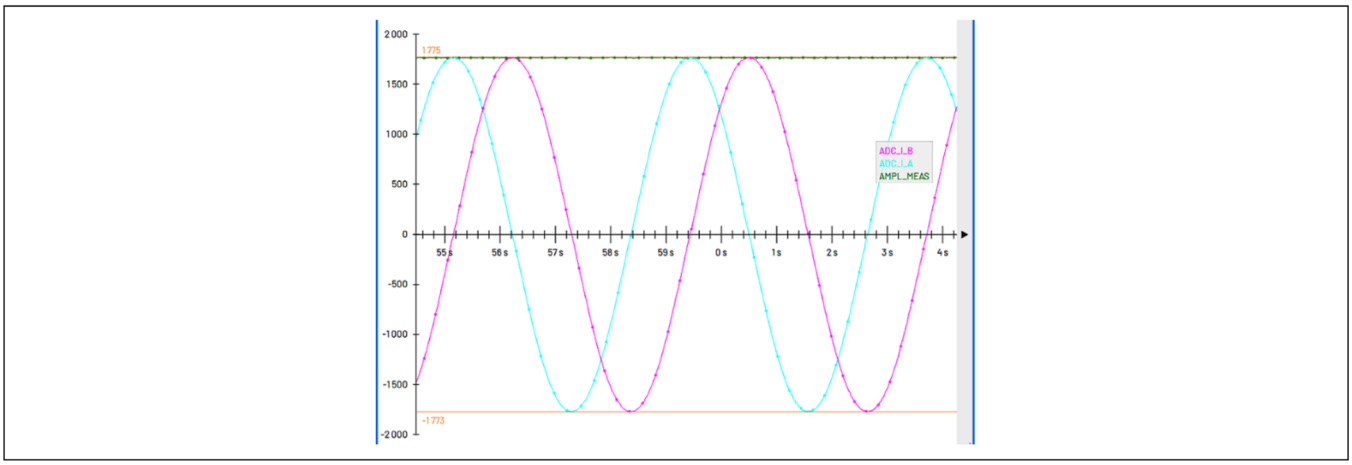


図 23. 乱れのない電流測定をチェックするための、StealthChop+において低速での ADC 結果を用いたモーター電流のパターン

フリーホイーリングおよび受動ブレーキング・モード

停止のためにモーター電流を 0 に設定する場合（*IHOLD* = 0）、モーターをスタンバイさせるために様々なモードの選択が可能です。デフォルトのモード（*FREEWHEEL* = 0、通常動作）は、電流コントローラをオンのままにし、最小限の電流をコイルに供給します。真のフリーホイーリングを実現するには、ドライバは、すべてのモーター・ドライバ・ブリッジを完全にオフにします（*FREEWHEEL* = 1、フリーホイーリング）。この代替方法は、受動ブレーキングです。受動ブレーキングでは、ドライバのローサイド FET またはハイサイド FET のどちらかを用いてそのバック EMF を短絡することにより、モーターにブレーキをかけます（*FREEWHEEL* = 2 または 3、受動ブレーキング・ローサイドまたはハイサイド）。フリーホイーリング・モード 1 または 2 は、モーターをホットプラグするために入力することもできます。

StallGuard+

StallGuard+は、実際のモーターの負荷を正確に測定しモニタする、センサーレス・システムです。そのメインの出力値 *SGP_RESULT* は、モーターの負荷余裕を表す尺度であり、どの程度のトルクが使用可能かを示します。*SGP_RESULT* は、モーター効率システム CoolStep+ の基礎となるものでもあります。そのため、すべての利点を活用するには、適切な設定が不可欠です。

StallGuard+は、動作時のモーター・パラメータを測定することにより得られる情報を主として使用します。モーターの負荷を正確に判定するには、コイルの抵抗やコイルの誘電率などモーターのいくつかの特性値を知る必要もあります。

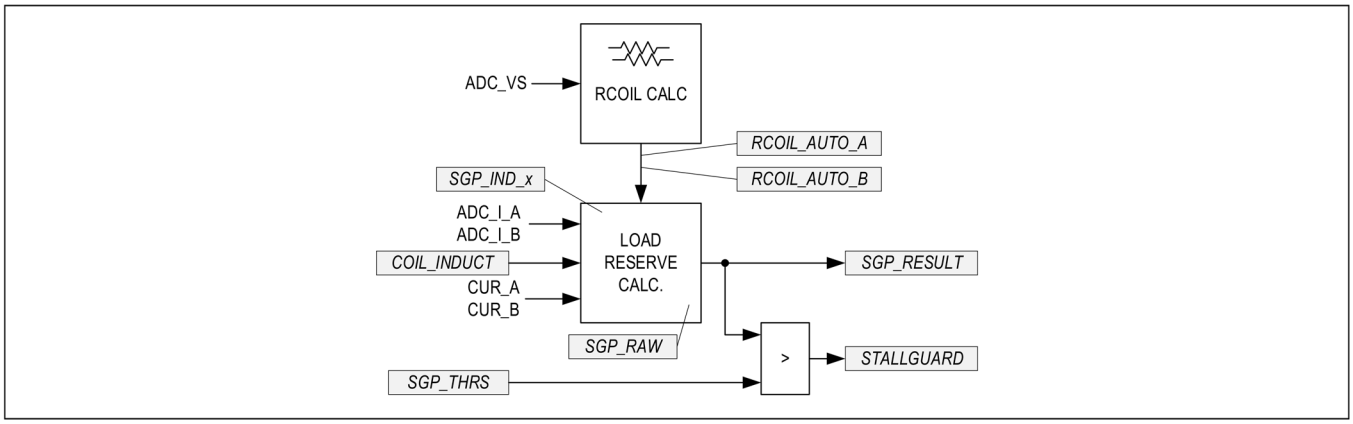


図 24. StallGuard+の負荷余裕の計算における構造および基本パラメータ

SGP_RESULT の解釈

StallGuard+は、負荷余裕値（*SGP_RESULT*）によってモーター負荷の実際の状況を説明する 4 象限の情報を提供します。また、アプリケーションで現在用いられている使用可能なモーター・トルクの量を判定します。安定動作領域では、モーターの負荷が（わずかに）増加すると、モーター・トルクがインクリメントして負荷角度ベクトルが右に回転し（図 25 参照）、*SGP_RESULT* が減少します。過負荷領域では、モーターの負荷が更に増加すると、使用可能なトルクが減少し、モーターのステップロスが生じる可能性が高くなります。負荷余裕値が 0 に近い領域では、負荷がわずかに変動した場合のモーター・トルクの増加はほんの少しです。各 4 フルステップを繰り返す負荷角度のサイン波関数に従ってトルク曲線が変化するためです（図 30 参照）。そのため、0 に近い領域（またはジェネレータ動作での 511 に近い領域）は重要であり、モーターは容易に不安定領域に入る可能性があります。更に、パラメータ化が不完全であったり、モーター・パラメータが浮遊したりすると、定められた負荷余裕値に一定の割合の不確実性が生じます。そのため、モーターの動作点は、不確定領域から十分離れるよう設定する必要があります。ただし、モーターのストールは、トルクのビットを犠牲にすることなく確実に検出できます。（フィルタ処理されていない）StallGuard+の値は、過負荷時の各ケースにおいて 0 のラインを横切るためです。

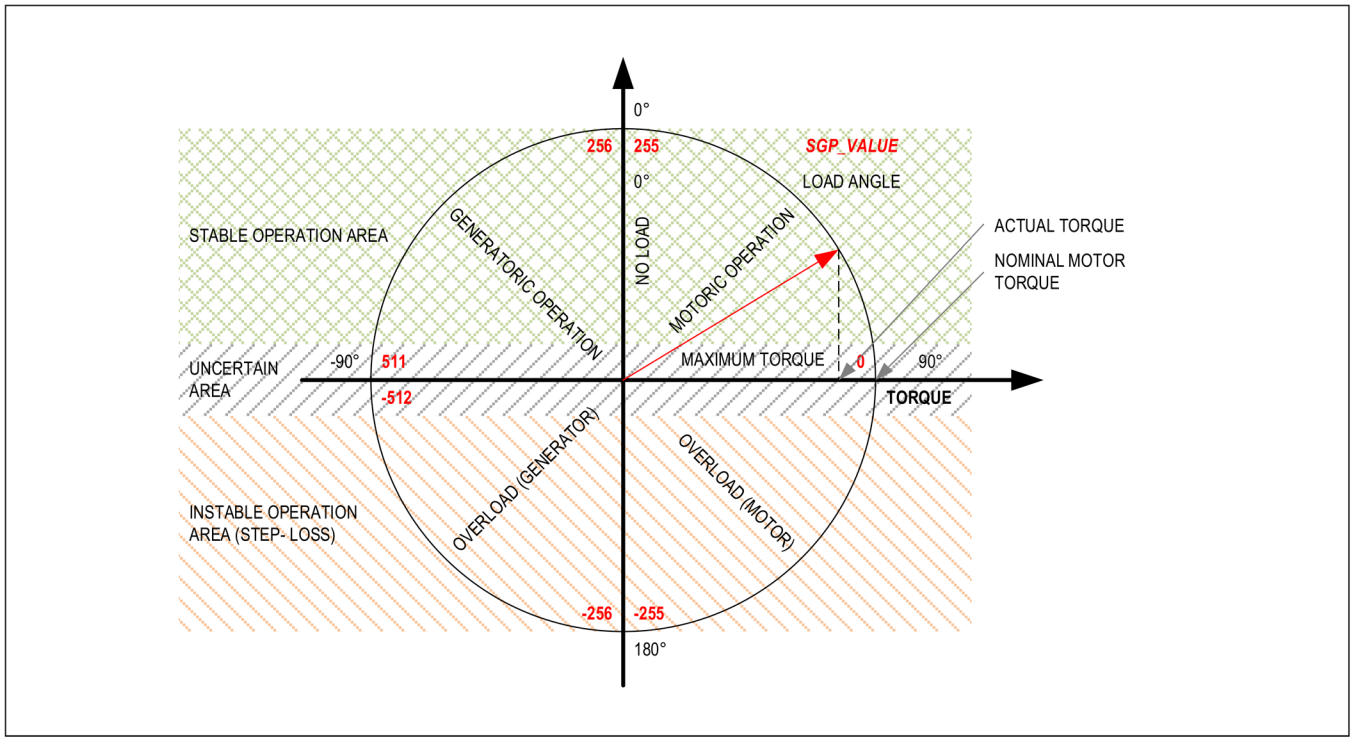


図 25. モーター動作の異なる象限での StallGuard+の値

図 25 に、モーター動作に関し、異なる象限での StallGuard+の値を示します。ベクトルの例は、円周上の特定の点を指し示す *SGP_VALUE* に応じてベクトルを TORQUE 軸に投影することによって、使用される相対トルクを示します。円の直径は、モーター電流に応じて変化します。不確定領域の割合は、適切な設定、モーター電流、モーター速度によって異なります。

ストール検出

ストール検出は、ストール・イベントを検出するため、または、過負荷を検出することによりモーターがストールするのを防止するために使用できます。*SGP_THRS* を用いて負荷の制限値を定義します。プログラムされた *SGP_THRS* を *SGP_RESULT* が下回ると、StallGuard+システムは stallguard 信号（*DRV_STATUS* レジスタ内）をアクティブ化します。*stallguard* 信号は、*DO_CONF.do0_stall* および *DO_CONF.do1_stall* を用いて出力ピン DO0 または DO1 のいずれかに割り当てられ、それによって外部コントローラに対しストール状態を示し、モーターを停止させることができます。*stallguard* 信号を用いるストール検出がイネーブルされるのは、CoolStep+の速度範囲が、スレッショルド *VCOOLTHRS* で定義される速度以上、*TPWMTHRS* で定義される上限速度以下の場合です。*THIGH* は、CoolStep+を使用できる上限速度範囲を規定しますが、それによってストール検出がディスエーブルされることはありません。

STEP ピンに入力されるステップ・パルスを停止せずにモーターをストールする場合、実際のストール・イベントによって、使用可能範囲内における（フィルタ処理されていない）StallGuard+のジッタの跳びが生じます。

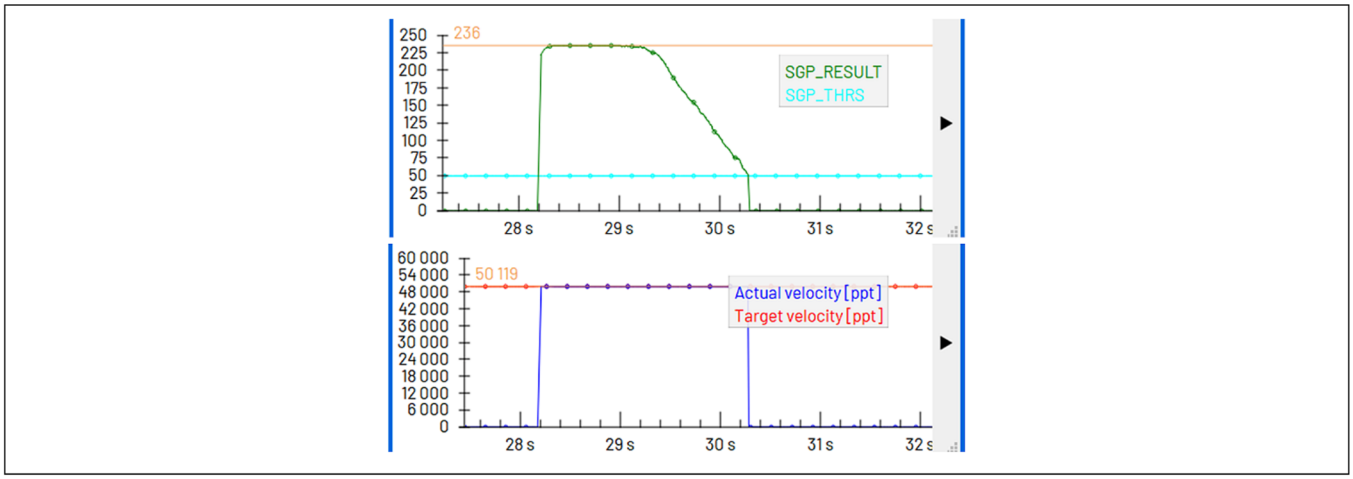


図 26. SGP_THRS を使用して高負荷時にモーターを停止（実際の速度と SGP_RESULT をプロット）

低速ストール検出

StallGuard+のストール検出は、絶対負荷値に依拠しています。ストール検出用に固定されたスレッショルドを用い、レベルと機械的負荷の間の関係は安定していなくてはなりません。しかし、モーター速度が非常に低い場合は、モーター温度の上昇や低下により、絶対レベルがフローティング状態になる可能性があり、モーターから出力されるバック EMF 振幅が比較的低いため、測定は不安定になります。

低速ストール検出は、低速範囲での機械的な停止端を確実に検出するための、二次的なシステムを追加するものです。このシステムの速度範囲は、*TSGP_LOW_VEL_THRS*（下限速度）および *TCOOLTHRS*（上限速度、これを基に、StallGuard+は、*SGP_THRS* によって設定された固定絶対スレッショルドを使用します）によって定義されます。この二次ストール検出システムは、*SGP_RESULT* の立下がり勾配をモニタします。その感度をプログラムするには、*SGP_LOW_VEL_SLOPE* を用いて 2 つの測定の間の最小変化レベルを設定すると共に、その最小変化レベルより上での *SG_RESULT* のそれぞれの減少を示すのに必要な連続フルステップ数 *SGP_LOW_VEL_CNTS*（0～3 の値が 1～4 イベントに相当）を設定します。このシステムは、ストール検出をトリガする測定ノイズを防止すると同時に、駆動中に物理的な障害物に衝突した場合によく発生する、モーター負荷の突然の増加を敏感に検出できます。ノイズ除去性能を向上するには、*SGP_CONF.sgp_filt_en* を用いて *SGP_RESULT* のオプション・フィルタをアクティブ化します。

COIL_INDUCT の設定

モーターの負荷を正確に計算するために、このドライバは、主としてコイル抵抗とインダクタンスに基づく、簡略化されたモーター・モデルを使用します。ドライバは、モーターの抵抗を自動的に判定できますが、インダクタンスについては、インターフェースを用いて設定する必要があります。コイル・インダクタンスは、モーターのデータシートから取得できます。インダクタの設定はいくつかの効果をもたらすため、試験的なセットアップを用いて、目的のモーター・タイプに合うよう最適設定値を調整します。

モーターのコイル・インダクタンス *COIL_INDUCT* は、公称モーター電流定格と同様、モーターのデータシートに基づいて設定できます。コイル・インダクタンスは µH 単位で入力します。つまり、8mH のコイルであれば 8000 を入力します。誘電率の高いフィルタ・インダクタをモーターに直列に追加して用いる場合には、そのインダクタンスの 71%をこの値に追加します。モーター・コイルのインダクタンスは、基本的には特定の鉄芯および特定の量の巻線数を使用する結果であるため、モーター・メーカーはモーター・タイプに応じた良好な公称値を提供できます。温度による変動は、コイル抵抗の変動に比べるとわずかです。StallGuard+が 0（全負荷）から 255（無負荷）（付近）までの全測定範囲に対応できるようにするには、*COIL_INDUCT* の適切にフィッティングされた設定値が必須です。これは、速度が増加している場合に特に該当します。

未知のモーターのインダクタンスの初期値を決定するオプションの手段として、*CUR_PI_LIMIT* を 0 に設定することによりモーター電流を公称値から 0 まで減少させたときの、停止状態のコイル電流を追跡するという方法があります。それによって、LR の動作を追跡し、*PWM_CALC* が元の値の 1/e = 37%まで減少するのに必要な時間からインダクタンスを計算できます。

$$L = t_{37} \times R_{COIL}$$

ただし、精度には限界があります。R_{COIL} には、コイル抵抗、ドライバ、相互接続抵抗を考慮する必要があるためです。特に、モーター温度が既に上昇している場合、インダクタンスの計算には、メーカーの公称値ではなく、測定したコイル抵抗を用いる方が理に適っています。

注：個々のモーターに応じて、StallGuard+は、データシートの値からわずかに異なるインダクタンス設定値、例えば、データシート値の 90%や 110%のインダクタンス設定値で最善の結果を提供する場合があります。モーターのタイプや個々のモーターに合わせて最適にフィッティングされたスケーリング値を求めるには、データシート値を開始点として用い、実験による最適化を行ってください。

COIL_INDUCT の実験による最適化

広い範囲のモーターおよび動作シナリオに対し、モーターのデータシートに記載されているコイル・インダクタンス値は、StallGuard+の信頼できる挙動を実現します。ただし、動作限界の方向に向けて動作させてインダクタンス値の調整を行うことにより、StallGuard+の結果を更に向上できます。

モーター・モデルにおけるモーターのコイル・インダクタンス設定を用いると、複数の効果に対応できます。そのため、最適な設定値が必ずしもモーターのインダクタンスに正確に対応しているわけではありませんが、特定のモーター・タイプに対しては、これが数パーセント高いか低い値になることがあります。速度を増加させて StallGuard+の応答を追跡することにより、最適な設定をインタラクティブに決定してください。試験の際に低速度を用いることは推奨しません。コイル抵抗の変化に対する熱の効果が結果に影響し、また、コイルの温度が数秒以内に大きく増加したり低下したりする可能性があるためです。最適化テストには、モーターが停止するまでモーターの機械的負荷を緩やかに増加することができる必要があります。このプロシージャによって、モーターが現実的な負荷条件で動作するため、最適なフィッティング結果が得られます。

COIL_INDUCT を、公称モーター・コイル・インダクタンス値に設定します。中速度域での目的の動作電流を用いてモーターを動作させます。適切にフィッティングされた速度を決定するには、*PWM_CALC* をモニタします。*PWM_CALC* が、停止時または低速時の同じ電流での読出し値より大幅に大きい値でありながら、まだ 4095 の制限値には達していないことを確認することにより、速度を選択します。

sgp_filt_enable をセットして StallGuard のフィルタ処理をイネーブルします。この段階で、機械的負荷を徐々に増加します。無負荷時には、*SGP_RESULT* の読出し値はおよそ 200～255 になりますが（モーター速度に応じ、上限値は速度の増加と共に減少）、0 に近い正の値まで低下してから、モーターがストールします。*SGP_RESULT* が 0 に近づく前にモーターがストールする場合は、実際の *COIL_INDUCT* が低すぎます。モーターがストールする前に 0 を下回る場合は、*COIL_INDUCT* が高すぎます。良好な値が定まるまで、設定値を最大で±20%調整します。

設定値があまりに高すぎると、不安定になったり、*SGP_RESULT* の範囲を完全に外れたりします。

設定値があまりに低すぎると、通常は、全負荷時に 128 のレンジでの応答になります。

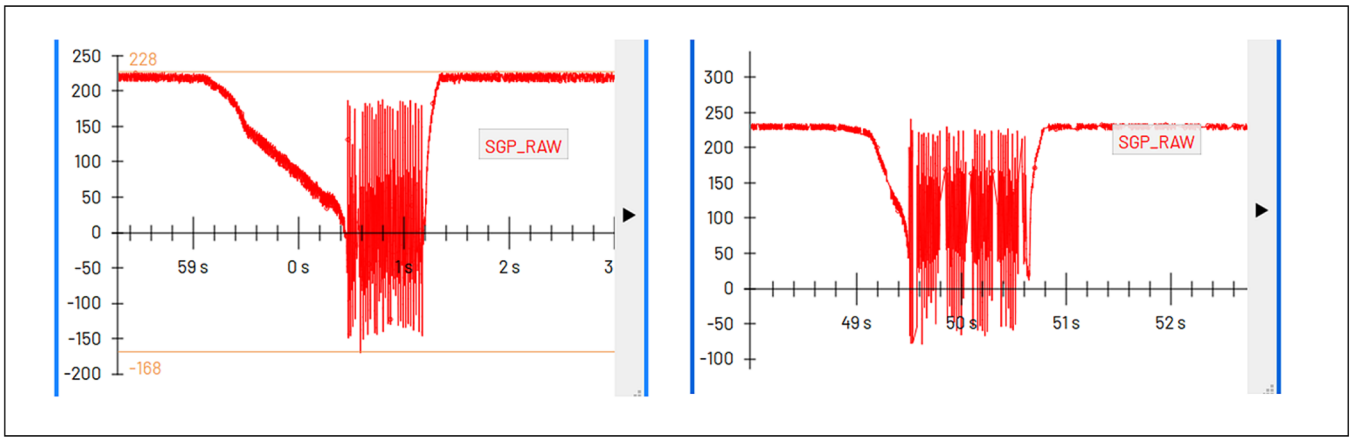


図 27. 良好な設定（SGP_RESULT が 0 近くの値になったときにモーターがストールを開始）と
低すぎる設定（0 に達する前にストールが開始）との比較

連続して生産する場合に、実験による最適化プロセスを繰り返さずに、増加するモーター浮遊を補正するには、測定したインダクタンスと *COIL_INDUCT* の設定値の間の係数に着目し、それを個々に測定したモーター・インダクタンスに適用します。

注：大きな共振や変形したバック EMF を示すモーターは、応答性に劣る可能性があり、無負荷時と全負荷時の間の *SGP_RESULT* の範囲が縮小します。実験による最適化で明確な最適値が得られない場合は、安定した結果を確保するために、元のモーター・インダクタンスに近い値を保持してください。

R_COIL の測定

低速度動作の場合は特に、StallGuard+が実際のモーター負荷を正確に判定するために、モーター・コイルの抵抗を正確に知ることが必要です。厳密に言うと、モーター・コイル（銅）の抵抗には、1 ケルビンあたり 0.39%の温度依存性があるため、コイル抵抗は完全に一定というわけではありません。そのため、モーターの温度が、環境温度からピークのコイル温度（大半の標準的なモーターの場合 130℃が限界値）まで上昇すると、抵抗は最大で 40%変化する可能性があります。実際の抵抗を追跡し続けるために、モーターが停止状態の場合、または *T_RCOIL_MEAS* で設定された速度スレッショルド未満で回転している場合は、ドライバは常に各コイルの実際のコイル抵抗を連続的に測定します。これは自動的に生じ、コイルごとの電流が最低でも *IRUN* の 50%の場合に完全な更新を行うために、50 ミリ秒から 100 ミリ秒以上モーターが停止したり非常に低速で動作したりする必要はありません。

コイルの温度は、数秒以内に変化する可能性があり（モーター金属の温度よりはるかに高速）、そのため、フル電流で動作しているモーターは、低速動作時の StallGuard+の結果の値から、数秒以内に大きく変化する可能性があります。そのため、一定の時間間隔で再キャリブレーションを行うことが必要になる場合もあります。

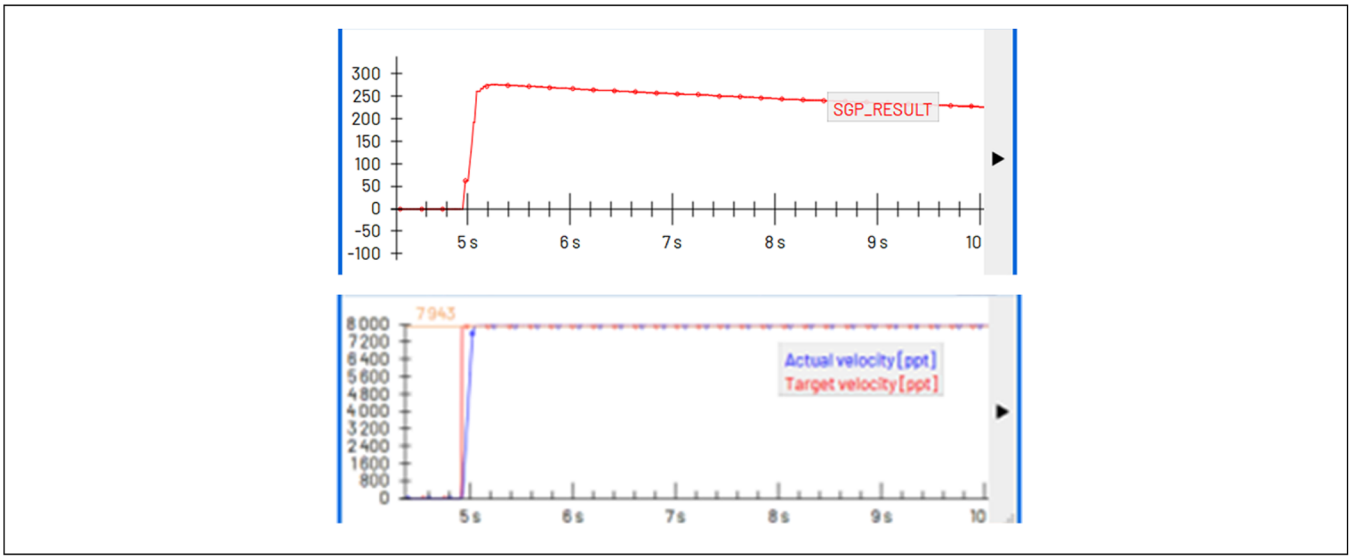


図 28. 非常に低速で動作している場合のコイルの温度上昇による、StallGuard+の結果（SGP_RESULT）の変化

オプションで、各コイルのレジスタ（*R_COIL_USER_A*、*R_COIL_USER_B*）によってコイルの抵抗を個別に定義します。このオプション（*rcoil_manual* ビットによってイネーブル）は、抵抗値がよくわかっている場合、例えば、モーター温度が直接測定されたり、ソフトウェアでモデル化されたりしている場合に用いてください。Ω 単位の抵抗値とレジスタ設定値との間の変換は、以下に示すように、用いる電流レンジによって異なります。

抵抗の内部表示：

$$R_COIL_x = 4 \times (R_{LSplusHS} + R_{COIL}[\Omega]) \times 7.1\ [1/V] \times \sqrt{2} \times CurrentRange[A\ RMS]$$

これは以下のように簡略化できます。

$$R_COIL_x \approx 40[\frac{1}{V}] \times (R_{LSplusHS} + R_{COIL}) \times CurrentRange[A\ RMS]$$

あるいは、コイル抵抗を測定値から求める場合は、次のようになります。

$$R_{COIL}[\Omega] \approx \frac{R_COIL_x}{40 \times CurrentRange[A\ RMS]} - R_{LSplusHS}[\Omega]$$

ここで、

- *R_COIL_x* は、R_COIL_AUTO、または、ユーザがプログラムした *R_COIL_USER* のいずれかを表します。
- $R_{LSplusHS}$ は、電力段の出力抵抗に巻線およびケーブルの抵抗を加えた値です。
- 係数 7.1 は、1 ボルトあたり 7.1 ADC カウントで電源電圧を測定するドライバに由来するものです。
- CurrentRange は、設定された、電流実効値フル・スケール設定値（1/SQRT(2) × ピーク電流）に相当し、レジスタ *DRV_CONF* の *CURRENT_RANGE* および *CURRENT_RANGE_SCALE* によって選択されます。

$$CurrentRange = 4.24A \times \frac{CURRENT_RANGE + 1}{4} \times \frac{CURRENT_RANGE_SCALE + 1}{4}$$

注：内部で測定されるコイル抵抗値は、*CURRENT_RANGE* および *CURRENT_RANGE_SCALE* の設定値で定義される実際の測定レンジによって異なります。モーターが回転しているときには、どちらも変更しないでください。

R_COIL_USER を用いてコイル抵抗を設定する場合は、必要な *CURRENT_RANGE* を考慮してください。

R_COIL_AUTO の解釈例

コイル抵抗が 5Ω のモーターが、1ARMS の電流レンジにおいて動作します。コイル抵抗の測定結果はどのようになると予測されるでしょうか。

ケーブルの寄与を無視すると、電力段は 1A のレンジでおよそ 100mΩ の出力抵抗を示します。この電流レンジの正確な実効値電流を電気に関する表から求めると 1.06A になります。

$$R_COIL_AUTO = 40[\frac{1}{V}] \times (5\Omega + 0.1\Omega) \times 1.06A = 216$$

これにより、どちらのコイルについても、このレンジでの *R_COIL_AUTO* の読出し値が 216 になると予想できます。現実には、ケーブルの影響やモーター・コイルの温度が上昇する可能性により、これよりわずかに大きな値になります。この読出し値は、2A の動作レンジでは 2 倍に、4A のレンジでは 4 倍になります。

R_COIL の測定フローチャート

内部のコイル抵抗計算ユニット RCOIL は、一方または両方のコイルの測定が可能な場合は常に、*R_COIL_AUTO* を測定し更新します。これは、*T_RCOIL_MEAS* で定義される低速（停止または停止に近い状態）になっていること、および、各コイルについて *IRUN* の 50%以上の電流が流れていることをチェックします。停止状態の場合、これは通常、モーター電流が *IHOLD* にランプダウンするまでの *TPOWERDOWN* の間、満たされます。スレッショルド速度 *T_RCOIL_MEAS* は、モーターのバック EMF がコイル抵抗での電圧降下に比べて無視できる値に設定する必要があります（速度スレッショルドを参照）。または、最低速度の 10%以下に設定する必要があります。この場合は、StallGuard+および CoolStep+が用いられます。両方の条件が満たされている場合、測定は、時定数が 2048 チョッパー・サイクルのフィルタを用いて更新されます。コイルは、サイン波とコサイン波で駆動されるため、一方のコイルが停止しているかハーフ・ステップ・ポジションに近い状態である場合など、コイルの電流が小さすぎるためにモーター停止時に他方のコイルのみが測定される可能性が大いにあります。それに対処するため、モーターがイネーブルされた後にコイルに対して行われる最初の測定は、両方のコイルに適用されます。更に、一方のコイルについて決定されたコイル抵抗の変化は、他方のコイルの値の変化を推定するために、自動的に適用されます（*rcoil_thermal_coupling* ビットをセット）。

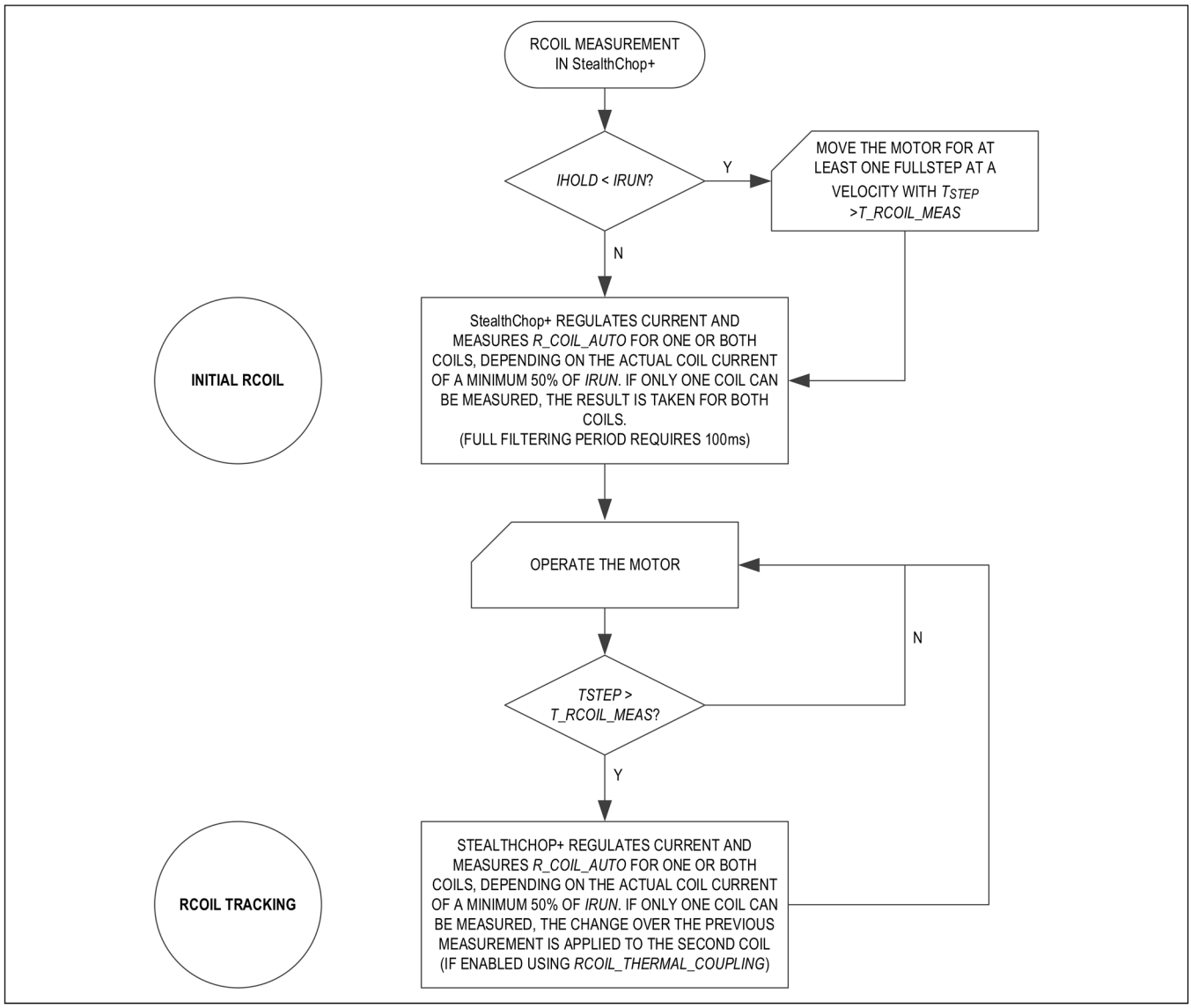


図 29. RCOIL 抵抗測定のフローチャート（モーター電流設定に続き StealthChop+をイネーブル）

モーター電流がパワー・ダウンする前に十分な時間を停止状態で確保するには、*TPOWERDOWN* を用います。それによって、コイル抵抗測定の定常の更新が可能になります。フィルタ処理を行うため、完全な測定には最大 100ms を要しますが、部分的なサイクルであっても漸進的な更新が行われます。モーター・コイルの温度が単調に上昇していない限り、両方のコイルで抵抗測定値を更新するには、*rcoil_thermal_coupling* をイネーブルします。各コイルの電流が *IRUN* の設定値の 50%以上であり、電流スケーリング *CS_ACTUAL* が 50 以上であれば、コイル抵抗は、停止時および低速動作時に測定されます。使用可能な測定時間およびこの測定の精度は、*TPOWERDOWN* の設定値により、または、*T_RCOIL_MEAS* で定義される非常に低速での動作時の測定を可能にすることにより、向上できます（適切な速度を選択するには速度スレッショルドを参照してください）。

注：モーター温度のモニタリング。

モーター・コイルの温度を計算するには、コイルの抵抗をモニタします。銅の熱係数はおよそ 0.0039/K であるため、モーター・コイルの温度は、室温時のコイル抵抗の初期測定からの抵抗変化に基づいて推定できます。20ºC から 120ºC に温度が上昇すると、コイル抵抗は初期値の 139%に大きく増加します。IC 自体の出力抵抗の温度依存性などの要因が寄与するため、低抵抗（数 Ω 未満）のモーターの絶対精度は低下します。

CoolStep+

電気モーターは、90º もしくはそれ付近の負荷角度で動作しているときに、最も高い効率を示します。負荷角度が 90º の場合、ステータの磁界はローターの磁界に比べ 90º オフセットされています。モーターのトルクは、サイン波の形状で、負荷角度と共に増加します（図 30 参照）。実際のトルクを発生するのに必要なステータ磁界の強度は、オフセットが 90º の場合に最小になります。

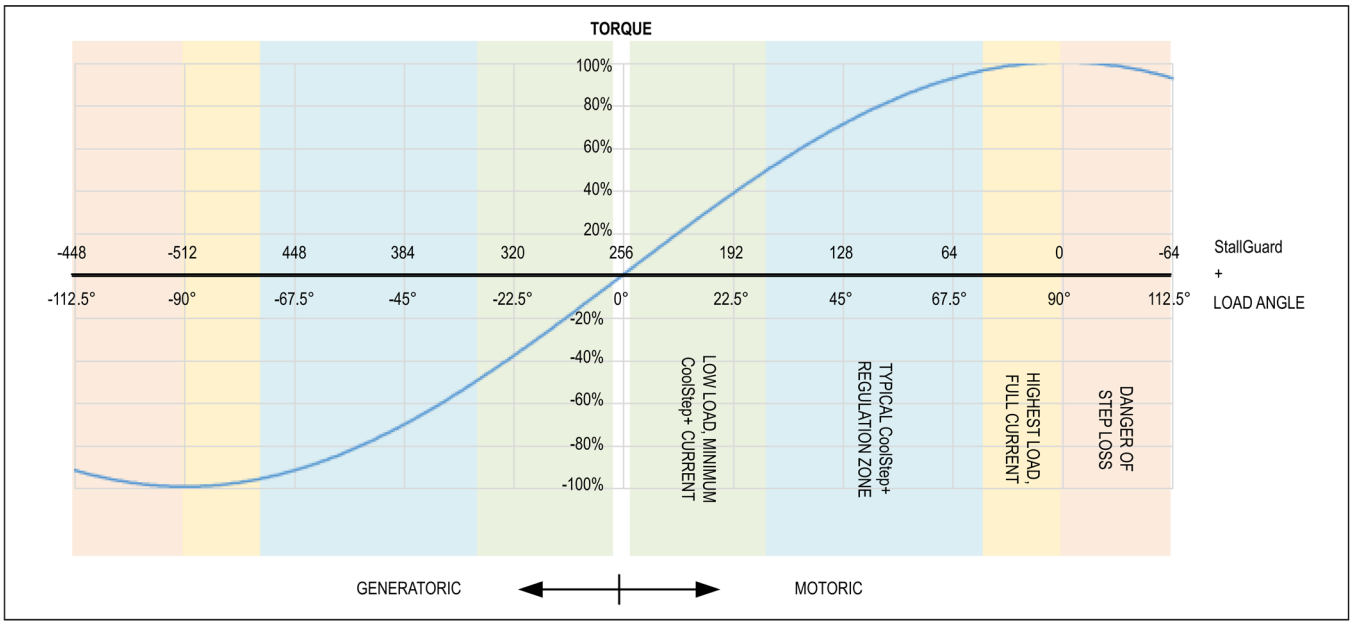


図 30. モーターのトルクと負荷角度の関係

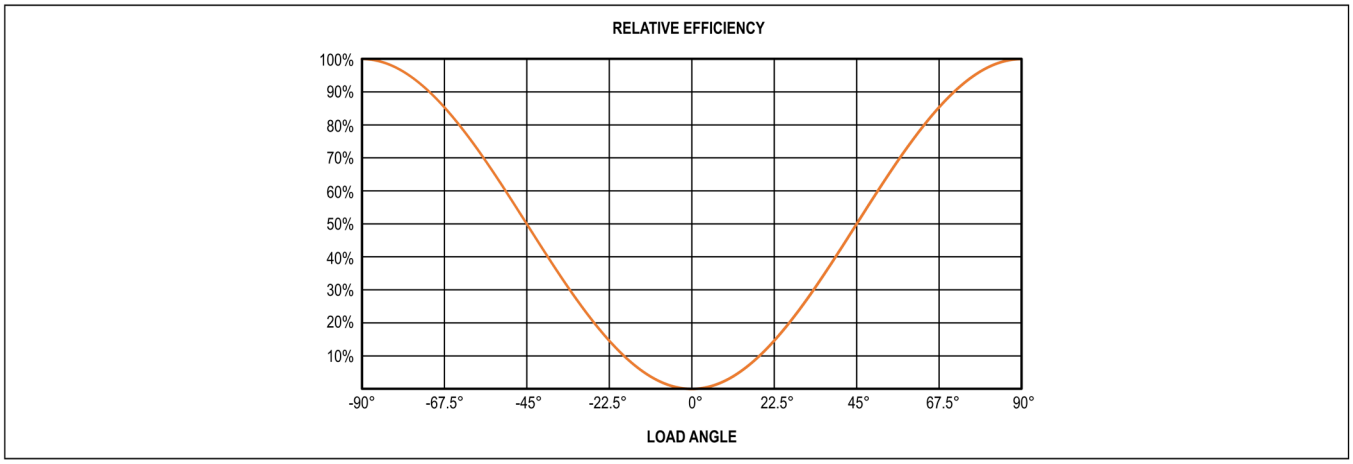


図 31. 相対的な効率と負荷角度の関係

ステータの磁界強度をこの必要最小限の値まで減少させることは、モーター電流を必要最小限の電流まで減少させることを意味します。ステッピング・モーターは一般的にオーバーサイズであり、アプリケーションで求められるピーク・トルクを確実に供給するためにフル電流で動作する、という点を念頭に置くと、電流を各時点で真に必要な値に適応させる場合に、最適化できる可能性は大いにあります。モーターの消費電力はモーター電流の二乗に比例して増加するのに対し、トルクは電流に対し単に比例して増加するため、適応型のモーター電流によりエネルギーを大幅に節約できます。効率のグラフ（図 31 参照）は、効率は負荷角度と共に変化するサイン波関数に従うことを示しています。CoolStep+は、負荷角度を 90º にわずかに満たないその最適値付近に設定しますが、数十度の安全領域があり、負荷の変化に対するステッピング・モーターの真の安定性を確保します。この方法は、最適化された効率と従来の動作の安定性を組み合わせるものです。

注：CoolStep+は、モーターの実際の負荷余裕（*SGP_RAW*）を提供する StallGuard+の結果に依拠して、負荷角度の 90º からの偏差を示します。そのため、CoolStep+を用いる前に、StallGuard+を正しく設定する必要があります。

CoolStep+の設定プロシージャ

CoolStep+は、ほとんどのパラメータにデフォルト設定値を備えています。しかし、詳細な調整を行うと、ドライバを機械的負荷プロファイルに適応させて最高の安全マージンを与えることができ、また、それと同時に最高レベルのエネルギー節約を実現できます。

表 9. CoolStep+の設定プロシージャ

番号	タスク	詳細
1	基本的なモーター設定	モーターのインダクタンスを設定し、コイル抵抗の自動判定を実行することによって、StallGuard+を設定します。
2	負荷余裕を設定	ガイドラインに従って負荷余裕値を決定するか、デフォルト値のままにします。
3	削減率を選択	COOL_CUR_DIV の設定を 2～10 にして、IRUN の 1/2～IRUN の 1/10 の間のより小さな電流を制御します。1 を設定すると、CoolStep+は無効になります。1/4 または 1/5 に減少させると最も良い安定性が得られ、大半のモーターで高レベルのエネルギー節約ができます。COOL_CUR_DIV = 4 から開始し、追加の設定を最適化した後、エネルギー節約と安定性の関係を再チェックします。IRUN/COOL_CUR_DIV が 35 未満の場合、ANGLE_LOWER_I_LIMIT をおよそ 150 まで低下させ、角度レギュレーションを小電流でアクティブ化し続けます（デフォルト値は 256）。
4	下限速度スレッショルドを決定	StallGuard+が安定な値を与え、モーター負荷の変化に正しく応答する、下限速度を決定します。この速度で TSTEP に等しい TCOOLTHRS を設定します。
5	CoolStep PI レギュレータの設定値を決定	COOLSTEP_P を 256 を超えないように増加させ、負荷ステップへの最初の高速応答（必要に応じ最初の数ミリ秒）を実現し、COOLSTEP_I を 10～128 の範囲に設定し、増加するモーター負荷への素早い応答を可能にします。予想される最大負荷上昇時に負荷余裕がアンダーシュートしないことを確認します。
6	電流低減速度を決定	COOL_PI_DOWN_LIMIT を増加させて、アプリケーションの負荷変化に最もよく対応しながらも不安定にならないような値になるまで、電流の低減を加速します。330 に設定すると、およそ 100ms～200ms の反応時間になります。電流の低減は、電流の増加に比べて大幅に低速です。
7	電流のランプアップ速度を決定	例えば速度範囲（下限は TCOOLTHRS で定義、上限は THIGH で定義）を離れることによって、CoolStep+をスイッチ・オン／オフします。COOL_PI_OFF_SPEED を、IRUN の電流が高速でジャークのないインクリメントを示す値に設定します。COOL_LOW_LOAD_RESERVE に設定されている値の 10%から開始し、この値の 50%まで増加します。このようにして、速度範囲を離れることにより、SGP_RAW が LOW_LOAD_RESERVE で定義された制限値未満になる状況と同じように、電流がインクリメントします。

CoolStep+の調整

CoolStep+は、CoolStep の高度な 2 象限バージョンです。これは、実際の StallGuard+値である *SGP_RAW* を評価することによって、モーター電流を実際のモーター負荷に適応させます。そして、その結果、この *SGP_RAW* が未使用の使用可能モーター・トルク量、つまり「負荷余裕」を表します。その目的は、モーターが各状況において、アプリケーションを確実に動作させるだけのトルクを備えるような形で、負荷余裕を最小限にすることです。

ターゲット負荷余裕ジェネレータ

ターゲット負荷余裕ジェネレータは、実際のモーター電流によって決定されるターゲット値を、負荷余裕に供給します。

小電流の状況では、使用可能なモーター・トルクは低く、機械的負荷がわずかに増加しても、モーターのトルク能力を容易に超えることがあります。更に、増加した負荷角度に反応するには、PI レギュレータが電流を円滑にランプアップするための一定の時間が必要です。そのため、小電流の状況では、大電流の状況と同じようなトルク制限付近でモーターを動作させてはなりません。大電流の状況では、モーターの使用可能トルクは大きく、負荷の増加量が同じである場合、モーター負荷角度への影響は比較的小さくなります。そのため、大電流状況でのモーターは、90º の最適負荷角度に近い負荷余裕のうち、より小さな割合で動作できます。モーターの消費電力はモーター電流の二乗に比例して増加することを考慮すると、これはよりフィッティングになります。より大きなモーター・トルクで動作した場合、最適な負荷角度に向かうことにより、エネルギーを節約できる可能性がはるかに高くなるためです。つまり、高トルク、大モーター電流の状況では、負荷余裕に対する割合と、この未使用の使用可能トルクを、最小限に抑える必要があります。

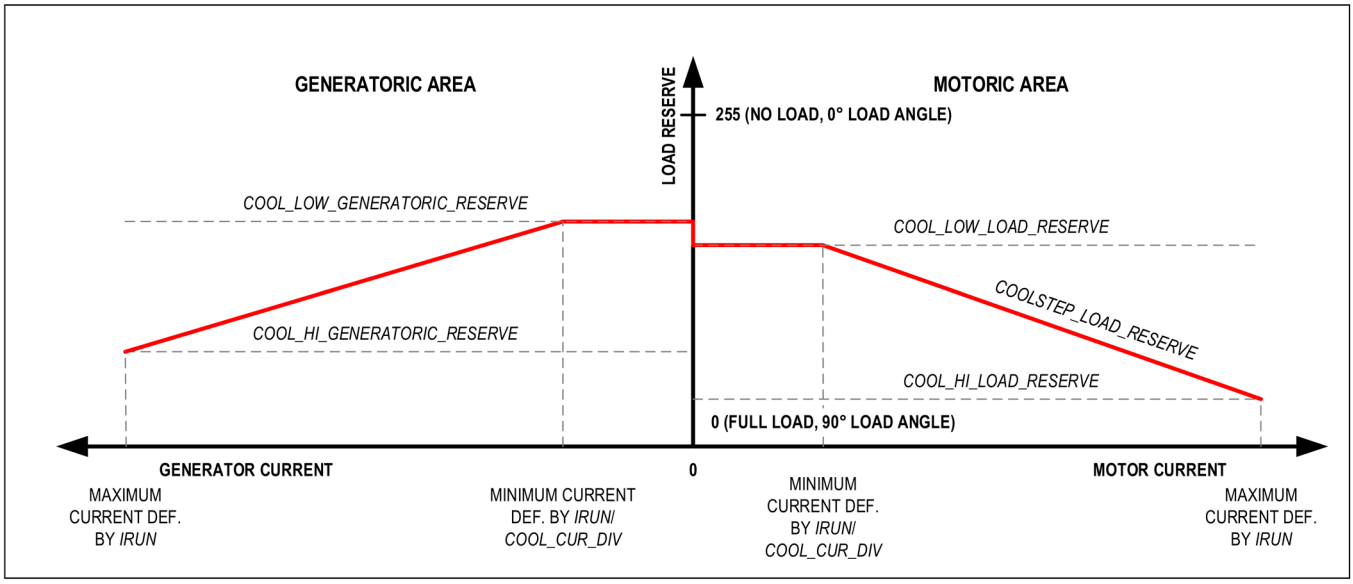


図 32. ジェネレータ動作で効率が低下する非対称設定を示す、ターゲット負荷余裕ジェネレータの機能

モーターは、特定の状況においては、ジェネレータとしても機能します。素早く減速するよう指令された場合や、負荷が回転と同じ方向に挿入された場合には、常にエネルギーがフィードバックされます。ジェネレータ動作の場合、CoolStep+は、通常動作と同じロジックを用いて負荷余裕を生成します。CoolStep の角度レギュレーションの場合、256～511 の負荷余裕値（*SGP_RAW*）は 255～0 にマッピングされ、ドライバは、ジェネレータ動作時の負荷余裕パラメータに応じて *COOLSTEP_LOAD_RESERVE* を計算します。

ジェネレータ動作では、機械的負荷からのエネルギーを電源にフィードバックするため、電圧を安全限界内に保持することは、通常 1 象限動作のみをサポートする電源システムにとって、難題になる可能性があります。フィードバックされたエネルギーを受け入れる能力は、ステッピング・モーター自体の電源コンデンサとエネルギー消費、およびその他の負荷によって制限されます。したがって、過電圧保護メカニズムを組み込むことが必要になる場合があります。この問題に対処するため、StealthChop+は、ジェネレータ動作時の負荷余裕機能を独立に規定できます。ジェネレータ動作の負荷余裕を 0º の負荷角度付近の値に選択すると、ジェネレータ動作時のステッピング・モーターの消費電力は増加します。そのため、電源にフィードバックされるエネルギーは、少なくなるかゼロになります。ただし、電流は規定された *IRUN* 値を超えて増加することはないため、この手段の効果は、ケースバイケースで証明する必要があります。

負荷余裕の大まかな指針

モーターの効率は、負荷余裕が 0 の場合に最高になります。しかし、ステッピング・モーターは、純粋なフィードフォワード制御システムであるため、最小限の正の値の負荷余裕による安全マージンが必要です。更に、このマージンによって、（例えば、インダクタンスとインダクタンス設定値との差異や、抵抗値の不一致による）測定の不正確さを補正できます。

COOL_HI_LOAD_RESERVE の選択：

これは、最大電流（*IRUN*）で動作している高負荷状態のモーターのための負荷余裕値であり、そのため、効率的な動作に最も重要です。必要な安全マージンを評価するには、モーター・ストールが発生する前に、最小の（正の）*SGP_RESULT* の帯域幅がアプリケーションの制限内にあることをチェックします。モーターの温度やインダクタンスの設定を変えることにより（例えば±10%）、コーナー・ケースの安定性をチェックします。アンダーシュート時に CoolStep レギュレータが反応できるよう、50%～100%の追加マージン、または 10 の最小オフセットを加えます。

その結果、*COOL_HI_LOAD_RESERVE* は 25～80 の範囲になるはずです。

COOL_LOW_LOAD_RESERVE の選択：

低負荷での負荷余裕設定値は、高負荷での負荷余裕よりも高くする必要があります。一定の負荷変動があった場合の影響が相対的に大きくなり、したがって、特に、低負荷の状況で 2 倍以上モーター電流を減少させることが想定されている場合に、ドライバは、反応するためにより多くのヘッドルームが必要になるからです。

- 減少が小さい場合：*COOL_CUR_DIV* = 2 とし、*COOL_HI_LOAD_RESERVE* と同じかそれよりわずかに大きい値を使用します。
- 標準的な場合：ヘッドルームを増加するために、*COOL_HI_LOAD_RESERVE* よりおよそ 64～128 だけ大きい値を用います。

COOL_HI_GENERATORIC_RESERVE および *COOL_LOW_GENERATORIC_RESERVE* の選択：

- 標準的な場合：*COOL_HI_LOAD_RESERVE* および *COOL_LOW_LOAD_RESERVE* と同じ値を用います。
- ジェネレータ動作による電源へのフィードバックが小さい場合には：モーター動作の場合より大きな値を用います。約 200 を超えないでください。

CoolStep+レギュレータ

CoolStep+電流レギュレーションは、最小限の電流を用いてモーターを確実に動作させることを目的にしています。負荷が変化したときの安定動作を確保するため、負荷の増加に対しては、CoolStep+は、電流を増加することによって素早く反応してステップロスを防ぎますが、負荷の減少時には、限定された勾配を示します。

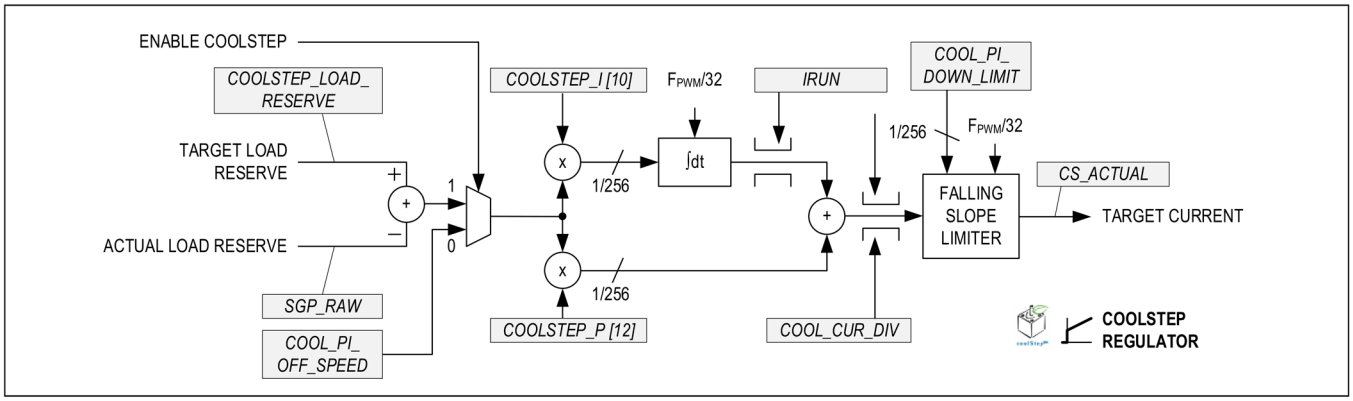


図 33. CoolStep+レギュレータの構造と基本的なパラメータ

CoolStep+は、電流および角度の PI レギュレータと同様、ターゲット・モーター電流を供給するために PI レギュレータを採用しています。レギュレータは、モーター負荷が変化した場合、特に、負荷が増加した場合に、素早く反応します。突然負荷が増加した場合にステップロスを防止するために、これは必須です。その結果、レギュレータは、*IRUN* で設定される上限値と、*COOL_CUR_DIV* で選択される *IRUN* の分数値（*IRUN* の 1/2、1/3、1/4～1/10）で定義される下限値の間でモーター・ターゲット電流（*CS_ACTUAL*）を安定化することにより、これを適応させます。

このレギュレータは、電流勾配の立下がり制限値（*COOL_PI_DOWN_LIMIT*）によって強化されています。この制限値は、ターゲット電流の最大変化を、各 32PWM 周期につき 1/256 ステップで設定します。それによって、モーターおよびレギュレータのカスケードは、CoolStep+がモーター電流をランプダウンする間に反応できます。

追加の勾配リミッタにより、CoolStep が無効化されている場合、例えば、モーターを減速することにより速度範囲を離れる場合や、上限速度スレッショルドを超える場合に、滑らかな遷移を確保します。*COOL_PI_OFF_SPEED* は、この場合に用いられる勾配を設定します。これは、レギュレーションの差と同じように作用しますが、それは、特定の値をレギュレータに供給し、その P 部分と特にその I 部分を用いてその差に反応することによって行います。しかし、あまりに遅い値には設定すべきではありません。*TPWMTHRS* によって設定される速度を超過したとき、または *standstill* フラグがアクティブになった場合に、電流が一時的に IRUN 設定値に設定されるためです。*COOL_PI_OFF_SPEED* は電流のインクリメントを扱い、*COOL_PI_DOWN_LIMIT* は電流のランプダウンを扱います。

注：CoolStep+ PI レギュレータを調整する前に、電流レギュレータを調整します。CoolStep+は、モーター電流レギュレータに依拠してモーターのターゲット電流の変化に十分高速に反応するためです。

CoolStep+の P 調整

CoolStep+の P 部分は、負荷余裕のアンダーシュートに即時に反応します。*COOLSTEP_P*=256 に設定すると、1:1 の応答になります。例えば、ターゲット負荷余裕を 1 だけアンダーシュートさせると、電流には 1 ステップのインクリメントが生じます。しかし、P レギュレータは、測定された信号のジッタを直接転送します。そのため、あまりに大きな係数は選択すべきではありません。アプリケーションの条件の範囲内で負荷測定のジッタを評価するには、*SGP_RAW* を追跡します。そのため、非常に速い負荷応答が必要な場合を除き、P 部分を大きな値に設定する必要がありません。*COOLSTEP_P* を数十～256 の範囲で選択します。機械的負荷が極端に突然でない限り、I 部分による応答は、ほとんどの反応をサポートします。大きな P 部分を用いている場合は特に、電流ジッタのフィードスルーを低減するために、フィルタ処理された生の値を常に選択します（*load_filt_en* = 1）。フィルタは、直前の 32 チョッパー・サイクル内で取得された直前の 16 個の負荷余裕測定値の平均値を計算します。そのため、著しい遅延は発生しません。

CoolStep+の I 調整

CoolStep+の I 部分は、P 部分と同じスケーリングを使用します。そのため、その設定値は P 部分の値を下回る必要があります。大まかな指針として、5～128 の範囲の *COOLSTEP_I* を選択します。

チョッパー周波数が 25kHz の場合、CoolStep+は毎秒約 750 回実行されます。これは、標準的な機械的時定数よりおよそ 10 倍、高速です。つまり、I レギュレータは通常、機械的負荷の変化（インクリメント）に対し、数サイクル以内に応答します。

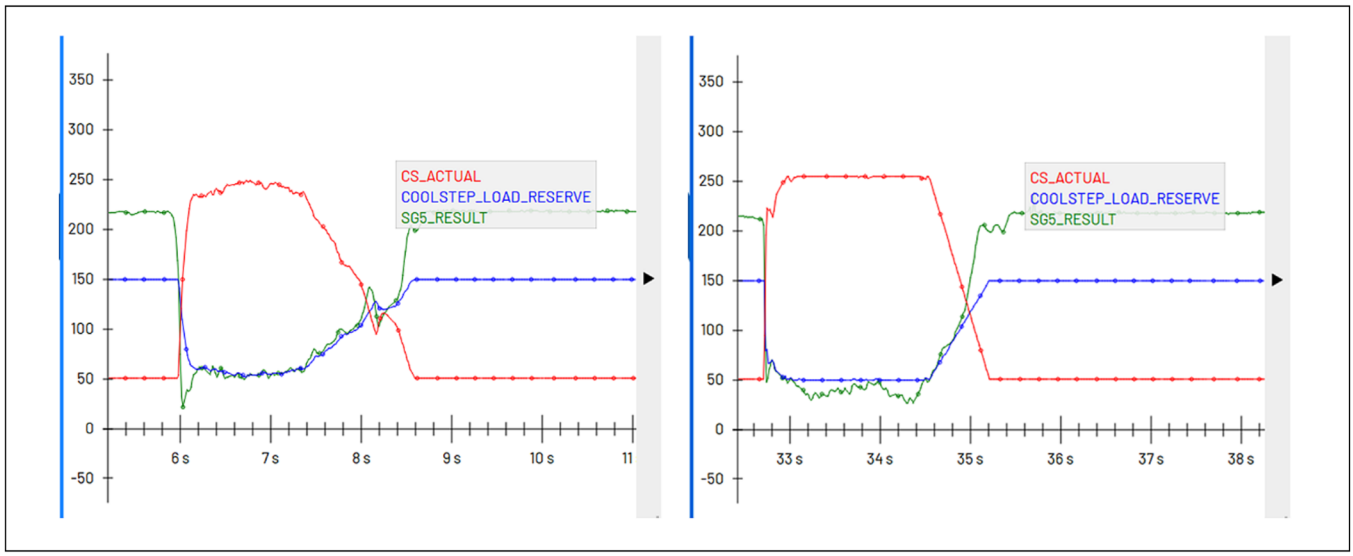


図 34. PI レギュレータの設定が低すぎる場合（負荷インクリメント時に SGP_RESULT（緑色）がアンダーシュート）と十分に高い場合との比較

COOL_PI_DOWN_LIMIT の調整

CoolStep PI レギュレータは、増加する負荷に対し素早く応答するようパラメータ化する必要があるため、減少する負荷への応答時間が同じであると、容易にレギュレーションが不安定になり、機械的な発振が励起される可能性があります。立下がり勾配リミッタは、レギュレーション・サイクル（32 チョッパー・サイクルごとに 1 回）ごとに最大電流減少を制限することによって、この問題に対処します。アプリケーションのモーター負荷が減少するのと同じ速さで電流がランプダウンするまで、*COOL_PI_DOWN_LIMIT* を増加し、エネルギー節約が最大になるようにします。負荷が突然変化した場合にアプリケーションのエネルギー効率を最適化するには、低負荷の時間の大半をカバーするよう、十分に電流低減できる値を使用します。

$$COOL_PI_DOWN_LIMIT = \frac{IRUN - \frac{IRUN}{COOL_CUR_DIV}}{t_{SLOPE} \times f_{CHOP}} \times 8192$$

ここで、

- t_{SLOPE} は、IRUN から *COOL_CUR_DIV* によってプログラムされた分数まで電流を減少させるための時間を表します。
- f_{CHOP} は、選択されているチョッパー周波数です。

例：チョッパー周波数が 25kHz の場合、*COOL_PI_DOWN_LIMIT* を 330 に設定すると、電流は 100ms 以内に 100 ステップ減少します。つまり、*IRUN* = 250 から *IRUN* = 50 まで減少するのに 200ms を要します。

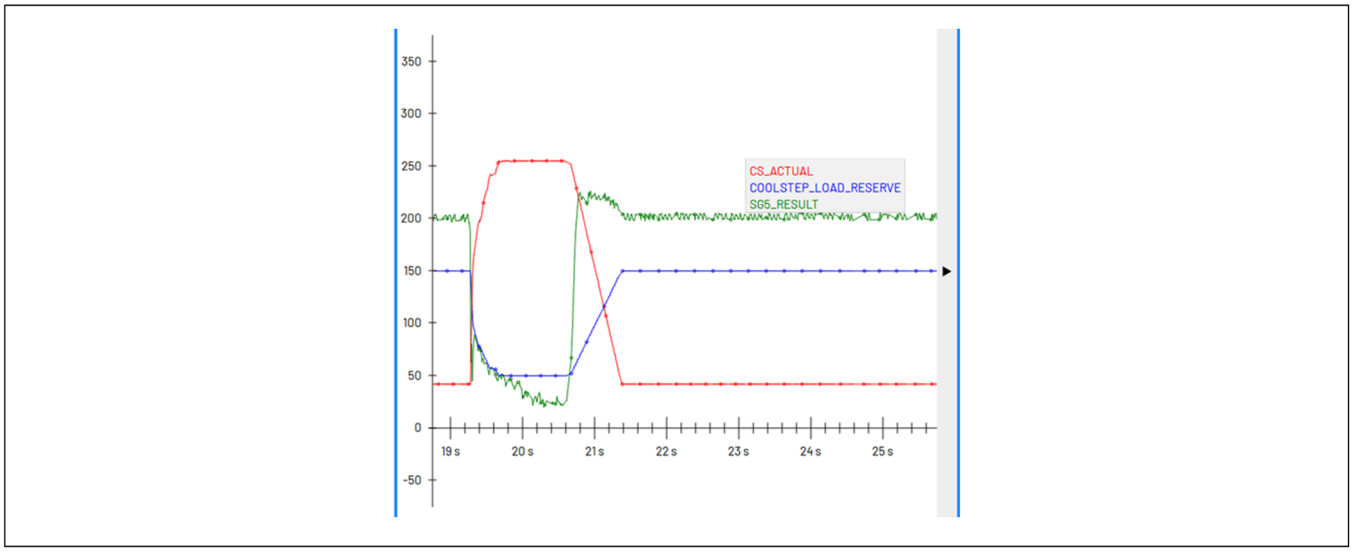


図 35. COOL_PI_DOWN_LIMIT で制御される立下がり勾配の制限

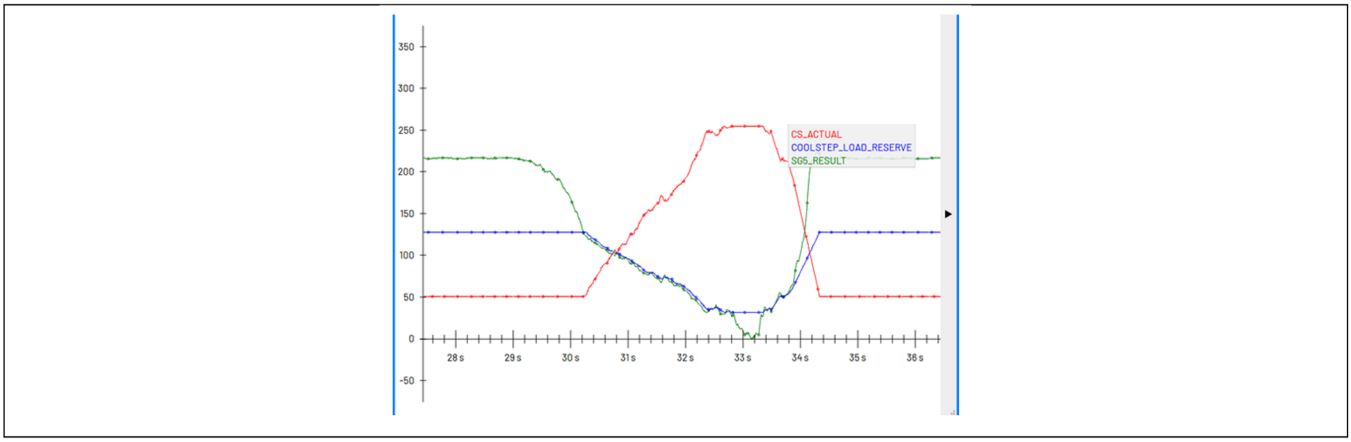


図 36. 最大電流（CS_ACTUAL = 255）に達するまでの、機械的負荷の増加に伴う SGP_RESULT の減少と COOLSTEP_LOAD_RESERVE の追跡。負荷の減少時の制限された電流減少勾配（COOL_PI_DOWN_LIMIT）

CoolStep+の PI および勾配制限についての大まかな指針

調整に必要な速度範囲の中央でモーターを動作させます。

COOLSTEP_P = 256 に設定すると、負荷余裕の不一致があるモーター電流に対し 1:1 の補正ができます。値を大きくすると電流リップルが増加するため、標準的な設定はこれより小さい値です。

COOLSTEP_P の選択：

- *COOLSTEP_P* を 25～256 の範囲で選択します。素早い反応が可能となるよう *COOLSTEP_I* を調整します。

COOLSTEP_I の選択：

- 5～128 の範囲に設定し、負荷の増加に十分高速に反応できることを慎重にチェックします。値が大きいほどより高速の反応が可能ですが、レギュレーションがオーバーシュートするリスクがあります。高い設定値でも初期反応が低速すぎる場合には、*COOLSTEP_P* を増加します。あるいは、オプションで、負荷余裕を増加します。最も安定性が高いのは、*COOLSTEP_I* < ½ *COOLSTEP_P* の場合です。
- 適用可能な最高速度と最低速度で安定性をチェックします。小電流時にレギュレーションが不安定でオーバーシュートする場合は、CoolStep+ の電流の分周係数を減らすことを検討します。例えば *COOL_CUR_DIV* = 4 とするか、または、*COOL_PI_DOWN_LIMIT* を減らします。より高い速度で十分に素早い反応をするのに必要なパラメータ・セットと同じセットを用いても、低速での安定性が得られない場合は、*TCOOLTHRS* を小さくすることを検討します。

COOL_PI_DOWN_LIMIT の選択：

- 反応時間が数 100ms の増加する低い設定値から開始し、低負荷周期での電流の節約が最適化されるまで続けます。機械的反応時間が数 10ms の範囲に近づくようになったとき、中程度の負荷での安定性を再チェックします。

COOL_PI_OFF_SPEED の選択：

- これは、負荷のインクリメントをシミュレートするものであるため、通常は、*COOL_LOW_LOAD_RESERVE* の 10%～50%の範囲に設定すれば十分です。ある速度範囲から、またはある速度範囲へ、必要な加速度でモーターを加速または減速します。この場合、CoolStep は、モーターを停止することなどにより、オフになっています。*COOL_PI_OFF_SPEED* を減らして、電流の突然の増加によるジャークを低減します。*TPWMTHRS* を用いて上限速度で SpreadCycle に切り替える場合、*THIGH*（CoolStep+がスイッチ・オフされる上限速度）と *TPWMTHRS*（ドライバが SpreadCycle に切り替わるときの速度）によって定義される速度差と合わせて調整します。アプリケーションの最大加速度で *TPWMTHRS* に達したときにジャークが発生する場合、*COOL_PI_OFF_SPEED* を増加するか、速度差を大きくします。

注意：CoolStep+の PI レギュレータの調整は、反応時間と安定性のトレードオフとなる場合があります。モーター速度が高いと多くの場合、短い反応時間が必要になりますが、これと同じパラメータで低速動作をさせると、安定性が低下する可能性があります。通常の（中程度および高い）アプリケーション速度範囲で動作させるには、*TCOOLTHRS* で制御される下限速度スレッショルドの値を大きくすることを検討します。また、低速度で長時間動作させる場合は第 2 のパラメータ・セットをロードすることを検討します。

StealthChop+に関連する速度スレッショルド

StealthChop+の高度な機能は、モーターの十分なバック EMF（BEMF）信号（ローターの回転によりモーターのステータ・コイルに誘起される電圧）の可用性に依存します。モーターの負荷および負荷角度を正しく判定するには、一定の BEMF が必要です。そのため、この機能をシームレスに有効化／無効化するために、いくつかの速度スレッショルドが設定されます。BEMF は、モーターの種類とその速度に依存します。

表 10. StealthChop+に関連する速度スレッショルド

パラメータ	説明	条件
T_RCOIL_MEAS	コイル抵抗を自動測定するための上限速度スレッショルド。 角度レギュレータの動作のため、および低速度 StallGuard+ストール検出システムのための、下限速度スレッショルド。	停止付近の、バックEMFが無視できる速度を決定します。代表的な範囲：1RPM～10RPM。この速度未満では、StealthChop+はモーターを純粋なオーミック負荷として取り扱い、そのコイル抵抗を判定します。 スレッショルド速度として提案される代表的な条件： $U_{BEMF} < 3\% \times U_{COIL}$ この条件では、バックEMFは抵抗による電圧降下の3%未満です。 $T_RCOIL_MEAS \geq 4 \times TSGP_LOW_VEL_THRS$
TSGP_LOW_VEL_THRS	*SGP_VALUE* をフリーズ化／ゼロ化するための上限速度スレッショルド。低速度 StallGuard の下限速度スレッショルド。	*SGP_VALUE*のフリーズを、*sgp_low_vel_freeze* = 1によって選択された速度未満に選択します。0の場合、*SGP_VALUE*が代わりにゼロに設定されます。 スレッショルド速度として提案される代表的な条件： $U_{BEMF} < 3\% \times U_R$ $TSGP_LOW_VEL_THRS \leq 1/4 \times T_RCOIL_MEAS$ 更に、十分な BEMF 電圧が必要です。 $BEMF_ABS \geq 30$

TCOOLTHRS	CoolStep+動作および StallGuard+動作の下限速度スレッショルド。低速度 StallGuard+ストール検出システムの上限スレッショルド。	十分にノイズの少ないStallGuard結果となる速度を決定します。また、動作サイクルの間にモーター・コイルの標準的な温度上昇に関して絶対的な値の安定性もチェックします。モーターが*T_RCOIL_MEAS*を超えて加速されると常に、新しい動作サイクルがモーター停止後にコイル抵抗測定の更新と共に開始されます。 *TSGP_LOW_VEL_THRS* > *TCOOLTHRS*であることを確認します。 スレッショルド速度として提案される代表的な条件： $U_{BEMF} > 30\% \times U_R$ 更に、十分なBEMF電圧が必要です。 $BEMF_ABS \geq 100$
TUDCSTEP	µDcStep をイネーブルするための下限速度スレッショルド、および、µDcStep が減速する際の下限速度。	CoolStep イネーブル速度から µDcStep イネーブル速度まで加速する間に CoolStep が電流をランプダウンできるための十分な速度差を維持して、*TUDCSTEP* < *TCOOLTHRS* となるよう選択します。
THIGH	CoolStep+用の上限速度スレッショルド。この速度以上の場合、電流は *IRUN* まで増加します。	StealthChop+動作をSpreadCycleと組み合わせている場合に、*TPWMTHRS*による決定に従い、この速度スレッショルドを用います。CoolStep+の速度領域を離れる場合に、*THIGH*と*TPWMTHRS*の速度差により、*COOL_PI_OFF_SPEED*による定義に従って、電流を*IRUN*まで滑らかにランプアップできるだけの時間が確保されます。それによって、電流差に起因して生じる可能性のあるジャークを防止できます。 スレッショルド速度として提案される代表的な条件： *TPWMTHRS* の 110%～120%
TPWMTHRS	StealthChop+動作用の上限速度スレッショルド。 IC はこの速度を超えると、デフォルトで SpreadCycle に戻り、ストール検出を StallGuard2 に切り替え、CoolStep2 をイネーブルします（そのように設定されている場合）。	StealthChop+は、停止状態から、*PWM_CALC*が4095の制限値に達していることによって特徴づけられる、モーター電圧が電源電圧レベルに達する速度の100%～150%までの速度範囲内で、モーターを動作できます。この速度を超えると、モーターは弱め界磁動作になり、安定したStealthChop+動作ができなくなる可能性があります。また、これは通常、使用可能なモーター・トルクが大幅に低下し始めるポイントでもあります。SpreadCycleは、弱め界磁速度に深く入り込んで動作できる能力を提供します。ただし、モーター・トルクは低減します。これは、選択したモーターがアプリケーションに対し最適でないことを示す兆候です。この領域での動作が必要になる場合、恐らくは、モーターが強すぎます。StealthChop+の動作範囲をより高速度に拡張する、または、電源電圧を増加するために、コイル抵抗が低く動作電流が大きいモーターを選択してください。 スレッショルド速度として提案される条件： 速度の100%～150%、ここで： • 必要なモーター駆動電圧がV_Sに達します。 • 必要な（最大）電流で動作し、機械的負荷が加わっている場合に、*PWM_CALC* が 4095 に達します。

モーターのバック EMF 定数の解釈

バック EMF 定数は、モーターが特定の速度で回転している場合に発生する電圧です。多くの場合、モーターのデータシートでは、この値は仕様規定されていません。モーターのトルクとコイル電流定格から計算できるためです。SI 単位系では、バック EMF 定数 C_{BEMF} は、トルク定数と同じ数値です。例えば、トルク定数が 1Nm/A のモーターの C_{BEMF} は 1V/rad/s です。そのようなモーターを 1rps（1rps = 毎秒 1 回転 = 6.28rad/s）で回転させると、発生するバック EMF の電圧は 6.28V となります。したがって、バック EMF 定数は次のように計算できます。

$$C_{BEMF}\left[\frac{V}{rad/s}\right] = \frac{HoldingTorque[Nm]}{2 \times I_{COILNOM}[A]}$$

$I_{COILNOM}$は、指定保持トルクに対するモーターの定格相電流です。

HoldingTorque はモーター固有の保持トルクであり、例えば、両方のコイルでのトルクは $I_{COILNOM}$ に達します。トルクの単位は[Nm]で、1Nm = 100Ncm = 1000mNm です。

$$U_{BEMF} = C_{BEMF}\left[\frac{V}{\frac{rad}{s}}\right] \times 2\pi \times \frac{f_{CLK} \times V_{ACTUAL}}{2^{24} \times MSPR}$$

BEMF 電圧 U_{BEMF}は、コイルごとの実効値電圧として有効です。そのため、この式において、公称電流には係数 2 が付きます。

MSPR は、1 回転あたりのマイクロステップ数です。例えば、1.8º のモーターの場合、256 マイクロステップに 200 フルステップを乗じて 51200 になります。

V_{ACTUAL}は、内蔵ランプ・ジェネレータによってコマンドされる実際の速度です。

StallGuard+は、U_{BEMF} を計算して、モーターの負荷を判定します。U_{BEMF} を直接測定することはできず、特定の駆動電圧を印加して得られるモーター電流測定値から導き出す必要があるので、これに重なる電圧 U_{COIL} の抵抗による降下が、この計算の正確性に関し重要な役割を果たします。そのため、この電圧降下をできる限り正確に知る必要があります。U_Rは、モーター・コイルでの電圧降下を表します。

$$U_R = R_{COIL} \times I_{COIL_IRUN}$$

R_{COIL}はコイルのオーミック抵抗です。

I_{COIL_IRUN}は、*IRUN* 設定値におけるコイルの実効値電流で、通常、*IRUN* が意図的に低い値または高い値に設定されている場合を除き、モーターに仕様規定された実効値電流です。

U_{BEMF}および U_R のソースを知れば、StallGuard+および CoolStep+が信頼できる動作をするのは、抵抗による電圧降下 U_R を計算する場合に生じる誤差の大きさを、U_{BEMF} が大きく超える電圧の範囲においてのみであることは明らかです。CoolStep+の大まかな指針として、U_{BEMF} >30% × U_R となる範囲で良好に動作できます。これは、動作中のモーターの温度変化による、決定されたコイル抵抗の不確定性を考慮しています。

速度に依存する機能の概要

上付き文字のオプション 1～4 を表 11 で説明します。

表 11. 速度に依存する StealthChop+の機能の概要

(Lower) Threshold Velocity (Ascending)	RCOIL Calibration	Low Velocity StallGuard+	StallGuard+	CoolStep+	µDcStep	Angle Regulator	StealthChop+	SpreadCycle	StallGuard2	CoolStep
Standstill	Y						Y			
T_RCOIL_MEAS						Y	Y			
TSGP_LOW_VEL_THRS		Y				Y	Y			
TCOOLTHRS1			Y	Y		Y	Y			
TUDCSTEP			Y	Y	Y	Y	Y			
THIGH3			Y			Y	Y			
TPWMTHRS								Y	Y1N2	Y1N3
TCOOLTHRS2								Y	Y	Y4N3
THIGH4								Y	Y	N

オプション 1：TCOOLTHRS(1)は、CoolStep+を制御するために使用でき（StealthChop+と組み合わせることを推奨）、また、StealthChop+の範囲の速度に設定する必要があります。あるいは、

オプション 2：TCOOLTHRS(2)は、StallGuard2 と組み合わせて CoolStep の活性化を制御するために使用できます。

オプション 3：THIGH(3)を使用して、CoolStep+をディスエーブルし、SpreadCycle に遷移する前に電流をランプアップできます（CoolStep を使用しない状態で SpreadCycle に切り替える場合に推奨）。あるいは、

オプション 4：THIGH(4)を使用して、高速度動作時に CoolStep をディスエーブルできます。

開放負荷の検出

StealthChop+によって、ADC 測定またはそれから計算された電流振幅（*AMPL_MEAS*）、および実際の電流を生じるのに必要な PWM 値（*PWM_CALC*）を用いて、モーターの実際の動作状態に対し正確なフィードバックが可能になります。また、計算したコイル抵抗（*RCOIL_AUTO_A* および *RCOIL_AUTO_B*）によって、各モーター・コイルおよびその接続の正確なチェックも可能になります。動作中にモーター接続が失われた場合、*DRV_STATUS* の開放負荷フラグ *OLA* および *OLB* を通じて、統合化されている単純な開放負荷テストが実行可能です。開放負荷検出は単に情報提供のためであり、検出時に IC はいずれのアクションもトリガしません。

モーター回転中の開放負荷検出：

モーターの回転時、要求されたピーク電流の少なくとも 75%/50%/25%/12.5%（*PWM_CONF* の *OL_THRSH* で選択）が、電気的な半回転（*MSCNT* の MSB の各変化によって指示）以内のある時点で最小値に達した場合、開放負荷検出の信号が発せられます。最初の更新は、単一方向に 1 回転した後です。そのため、速度が実際の電源電圧レベルを大きく上回る駆動電圧を必要とする状況において、モーターがそのターゲット電流に達することができなかった場合にも、開放負荷が示されます。こうした場合、通常、*PWM_CALC* はその最大値である 4095 に達しています。ただし、この方法は、電流レギュレータが加速設定に追従できない場合に、偽の開放負荷検出を示すこともあります。

注：モーターの回転が遅い場合、少なくとも 1 回の電気回転の間、開放負荷（*OLA* および *OLB*）をチェックしてください。

モーター停止中の開放負荷検出：

開放負荷検出は、要求されたコイル電流が少なくとも 80ADC カウントであれば、常に、各コイルの停止時にも更新されます。そのため、更新は、電流設定値およびマイクロステップ・ポジションに依存します。コイルがターゲット電流の半分未満になると、開放負荷の信号が発せられます。開放負荷検出では、大電流が流れているときには必ずコイルがモニタされているため、保持トルクが大きく失われた場合に、確実に通知されます。

注：最善を期すには、モーターの回転が遅い場合、少なくとも 1 回の電気回転の間、開放負荷（*OLA* および *OLB*）をチェックしてください。

µDcStep

µDcStep は、CoolStep+の上に付加されるものであり、過負荷の状況に対しモーター速度の動的な応答を可能にしてステップロスを防止することを目的とした、DC モーターに似た動作モードです。これは、高速時の機械的負荷が使用可能なモーター・トルク以上に増加する可能性のあるアプリケーションや、負荷に応じたモーター速度の変化が許容される可能性のあるアプリケーションを改善します。一方、このシステムは、できる限り高速での（与えられた状況でモーターが機械を回転させることができるだけの高速での）動作を可能にします。

外部モーション・コントローラを用いて µDcStep を効果的に使用するには、µDcStep の負荷検出機能をデジタル出力のいずれか（*do0_udcstep* フラグまたは *do1_udcstep* フラグ）に割り当てます。

µDcStep は、モーターを動作させるために CoolStep+が使用している電流（*CS_ACTUAL*）をモニタします。機械的負荷の増加に対応して CoolStep+がユーザ定義の電流上限値（*DECEL_THRS*、*IRUN* の *DECEL_THRS*/16 でプログラムされた制限値）に達すると、µDcStep は、モーター負荷が低減して CoolStep+が電流を再度制限値未満にランプダウンさせるようになるまでモーターを減速するよう、外部モーション・コントローラにフラグを発します。モーション・コントローラが µDcStep に対応して減速できる速度の下限値は、µDcStep アクティブ化スレッショルドによって定義されます（*TUDCSTEP* で定義）。これは、CoolStep+アクティブ化スレッショルドより大きい値であることが必要です。

µDcStep の効果は、モーターの慣性重量が短時間のトルク増加をもたらすことと、機械的負荷が通常は速度低下に伴って減少する一方、使用可能なモーター・トルクが（わずかに）増加することに依存します。モーターの慣性重量の効果を十分に発揮するには、高速の減速ランプをプログラムします。この減速の間の意図しないステップロスを避けるために、減速ランプは、外部負荷がない場合でもモーターがこれを維持できるように選択する必要があります。

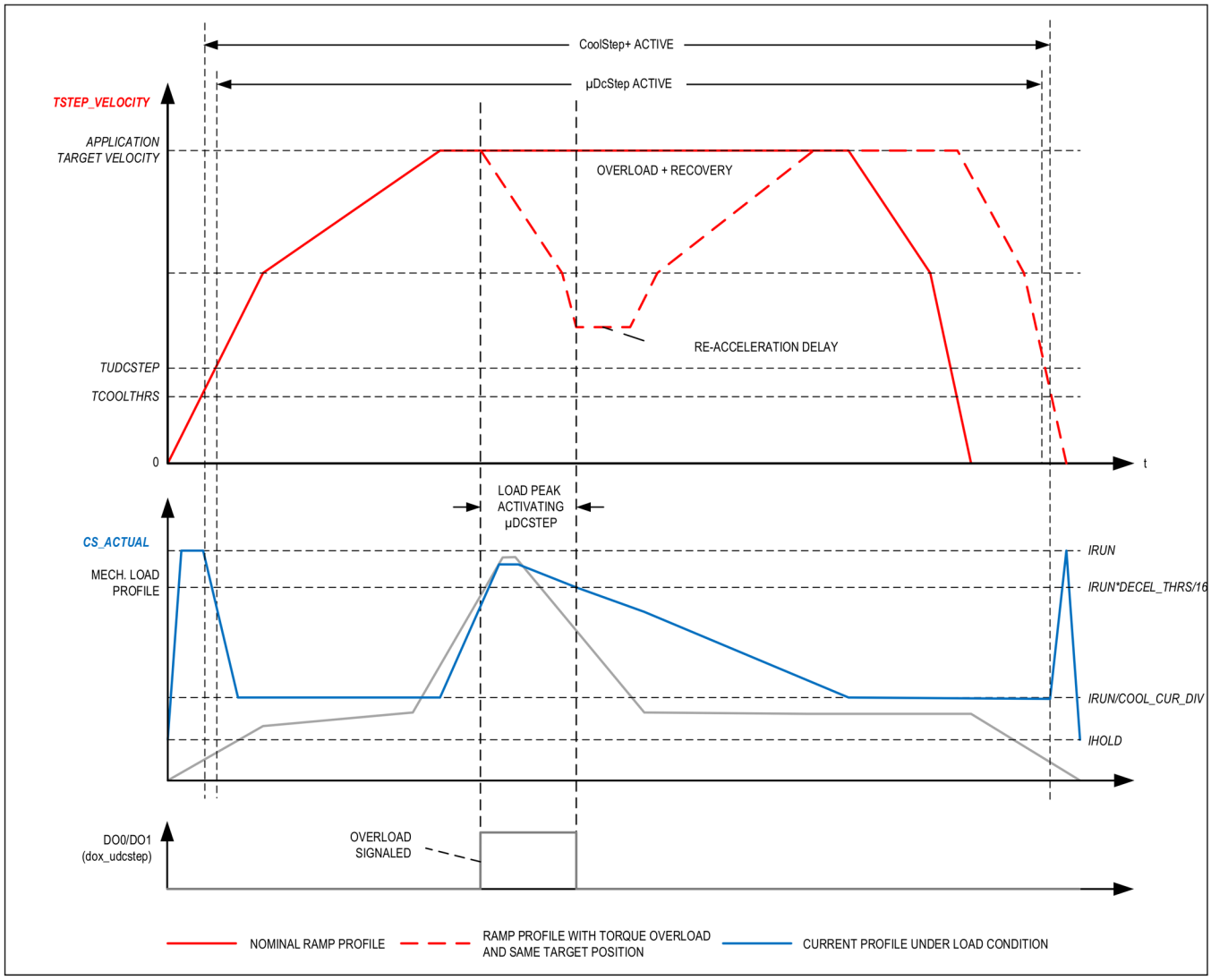


図 37. µDcStep を使用する速度プロファイルの例

このタイミング図は、DO0 または DO1 によってフラグされた機械的な過負荷状況の発生に対して取り得る、外部モーション・コントローラの反応を示しています。µDcStep は CoolStep+に依存しており、アクティブ化されている CoolStep+に結合しています。出力信号は、DECEL_THRS で設定されたスレッショルドを超えて電流をインクリメントするモーター負荷に対してアクティブになり、また、TCOOLTHRS および TUDCSTEP の両方の速度スレッショルドを超えていながらも THIGH 未満にとどまっている速度でのみ、使用可能です。外部モーション・コントローラは、特定の速度差によって、あるいは、安全と思われるより低い速度まで、あるいは、出力信号が再度非アクティブになるまで、モーターを急激に減速することによって反応する必要があります。ひとたび出力が解放されると、速度は再びランプアップできます。

所定の速度範囲において、DO0 と DO1 は以下の条件が成立するとフラグをセットします：*CS_ACTUAL* >= ((*DECEL_THRS* + 1) × *IRUN*)/16。

µDcStep のイネーブル

表 12. µDcStep のイネーブル・プロシージャ

番号	タスク	詳細
1	StallGuard+および CoolStep+を設定	StallGuard+および CoolStep+を設定。立上がり負荷と立下がり負荷の両方に素早く応答するよう、CoolStep+を調整します。
2	減速のための電流上限値を設定	µDcStep を 12～15（代表値）の範囲でアクティブ化するよう、電流上限値を設定します。そのためには、例えば、*DECEL_THRS* を 14 にプログラムします。したがって、電流制限値は、*DECEL_THRS*/16 × *IRUN* と示されるように、*IRUN* の分数にプログラムされます。
3	µDcStep をイネーブル	*DCU_CONF.dcu_enable* フラグをセットすることにより µDcStep をイネーブルします。*do0_udcstep* または *do1_udcstep* をセットすることにより、減速信号を出力するよう DO0 または DO1 を選択します。
4	下限速度スレッショルドを選択	*TUDCSTEP* は、アクティブ化できる下限速度スレッショルドを選択し、同時に、µDcStep が減速後に達する下限速度を選択します。CoolStep+がモーターを µDcStep のスレッショルド速度まで加速する間に、モーター電流のランプダウンを開始できるよう、*TCOOLTHRS* より高い速度を表す一定の速度差を加えて *TUDCSTEP* を選択します。
5	動作	選択された出力 DO0 または DO1 をモーターの動作中にモニタします。アクティブなレベルは、モーターが高負荷となっており過負荷の危険性があることを示します。応答として、出力がアクティブ化した場合にモーターをすぐに減速します。特定の速度差によって減速される、事前に選択された特定の最低速度まで減速される、出力がアクティブである限り減速される、のいずれかが生じます。減速の後、十分な時間（例えば数フルステップ）、出力が非アクティブのままであった場合、モーターを再加速します。

SpreadCycle および一般的なチョッパー

通常の動作モードは、StealthChop+チョッパーです。ただし、状況によっては、より一般的なチョッパー原理の方が有利な場合もあります。例えば、モーターのバック EMF が電源電圧レベルに達する点をはるかに超える速度でモーターが動作する場合です。

StealthChop+は電圧モード PWM 制御のチョッパーであるのに対して、SpreadCycle はサイクルごとに電流制御を行います。そのため、モーター速度やモーター負荷の変化に対して、極めて高速に反応します。両方のモーター・コイルを流れる電流はチョッパーを用いて制御されます。チョッパーは互いに独立に動作します。図 38 に、チョッパーの様々なフェーズを示します。

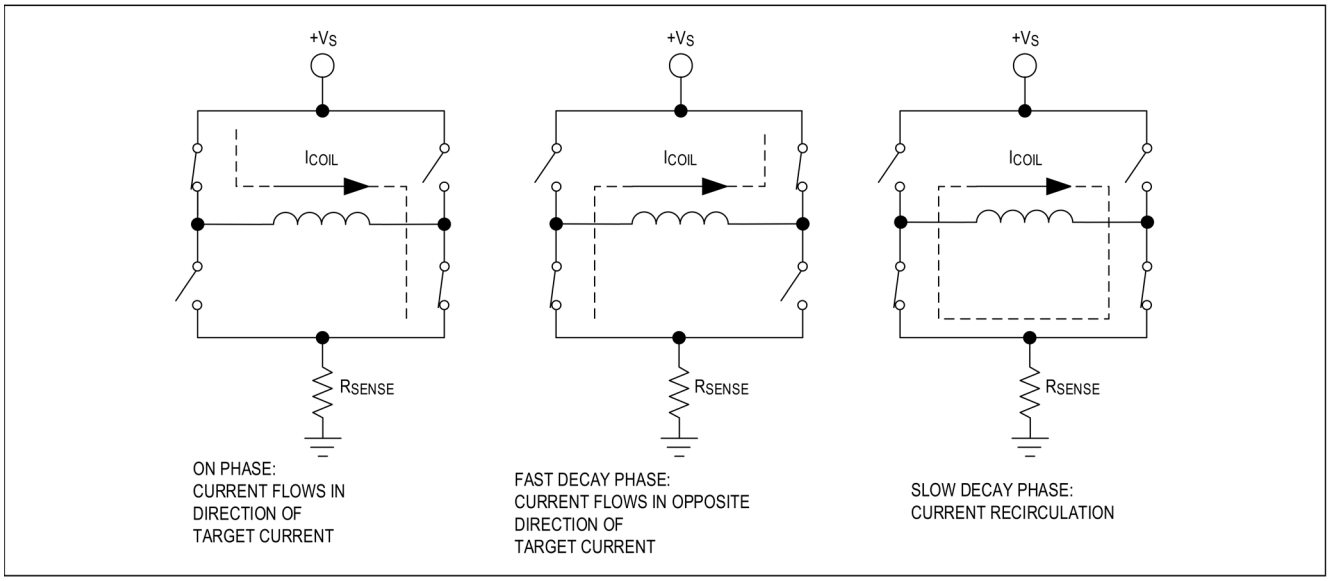


図 38. チョッパーの代表的な減衰フェーズ（センス抵抗は、下側経路の電流測定を表すための記号にすぎません）

電流はオン・フェーズと高速減衰フェーズのみを用いて安定化できますが、低速減衰フェーズを挿入することは、モーターの電気的な損失や電流リップルを低減する上で重要です。低速減衰フェーズに要する時間は制御パラメータで指定され、これによってチョッパー周波数の上限が定まります。電流コンパレータは、電流が厳密に 1 個のローサイド・トランジスタを流れるようなフェーズにあるときにはコイル電流を測定しますが、低速減衰フェーズ時には測定は行いません。低速減衰フェーズの終了はタイマーによって行われます。オン・フェーズの終了は、コイルを流れる電流が目標電流に達したときにコンパレータが行います。高速減衰フェーズの終了は、コンパレータまたは別のタイマーのどちらかによって行われます。

コイル電流の切り替わり時、寄生容量の充放電により $R_{DS(ON)}$ベースの電流測定でスパイクが生じます。この期間（通常 1～2 マイクロ秒）、電流は測定できず、マスクする必要があります。これを行うには、これらのスパイクをブロックするブランキング時間を設けます。

使用できるサイクルごとのチョッパー・モードは 2 つあります。SpreadCycle と呼ばれる高性能チョッパー・アルゴリズムと、定評のある定オフ時間チョッパー・モードです。定オフ時間モードは、オン、高速減衰、低速減衰の 3 つのフェーズを順番に繰り返します。SpreadCycle モードは、オン、低速減衰、高速減衰、第二低速減衰の 4 つのフェーズを順番に繰り返します。

チョッパー周波数は、チョッパー・モーター・ドライバにとって重要なパラメータです。周波数が低すぎると、可聴ノイズを発生する可能性があります。周波数を上げるとモーターの電流リップルは減りますが、上げすぎると磁気損失が生じる場合があります。また、ドライバの消費電力も、周波数の増加によって上昇します。動的消費電力の原因となるスイッチング勾配の影響が高まるためです。そのため、適切な妥協点を見つける必要があります。ほとんどのモーターは 20kHz～40kHz の周波数範囲で最適に動作します。チョッパー周波数は、多くのパラメータ設定値とモーターのインダクタンスおよび電源電圧の影響を受けます。

ヒント：20kHz～40kHz の範囲のチョッパー周波数は、SpreadCycle を用いた場合にほとんどのモーターに対して良好な結果をもたらします。周波数がこれより高いとスイッチング損失が増大します。

表 13. SpreadCycle および一般的な定オフ時間チョッパーを制御するパラメータ

パラメータ	説明	設定	コメント
TOFF	低速減衰時間（オフ時間）を設定します。この設定により、最大チョッパー周波数も制限されます。 StealthChop+での動作の場合、このパラメータは使用しませんが、モーターをイネーブルすることが必要です。StealthChop+で動作させる場合のみ、任意の設定が可能です。 このパラメータをゼロに設定すると、すべてのドライバ・トランジスタが完全にディスエーブルされ、モーターはフリーホイーリングが可能になります。 デフォルト = 0	0	チョッパー・オフ
		1…15	オフ時間の設定 N_{CLK} = 24 + 32 × TOFF （1 に設定すると 24 クロックの最小ブランキング時間で動作します）。
TBL	コンパレータのブランキング時間を選択します。この時間は、スイッチング・イベントとリンギングの時間を余裕を持ってカバーする必要があります。ほとんどのアプリケーションでは、1または2の設定で十分です。フィルタ・ネットワークを用いるなど、負荷の容量が大きい場合には、設定値を2または3にする必要があります。 デフォルト = 2	0	$20t_{CLK}$
		1	$28t_{CLK}$
		2	$40t_{CLK}$
		3	$58t_{CLK}$
chm	チョッパー・モードの選択。 デフォルト = 0	0	SpreadCycle
		1	一般的な定オフ時間

SpreadCycle チョッパー

SpreadCycle チョッパー・アルゴリズムは、高速減衰フェーズの最適な長さを自動的に決定する、高精度で使いやすいチョッパー・モードです。SpreadCycle は、デフォルト設定でも、優れたマイクロステッピング品質を提供します。チョッパーをアプリケーションに応じて最適化するために、いくつかのパラメータが使用できます。

各チョッパー・サイクルは、オン・フェーズ、低速減衰フェーズ、高速減衰フェーズ、第二低速減衰フェーズで構成されます。2 つの低速減衰フェーズおよびチョッパーあたり 2 つのブランキング時間によって、チョッパー周波数の上限が設定されます。低速減衰フェーズは、通常、チョッパー・サイクルの約 30%～70%を停止状態にし、モーターやドライバを低消費電力化するのに重要です。

TOFF の初期値を決定する例：
目標チョッパー周波数：25kHz

前提条件：2 つの低速減衰サイクルが全チョッパー・サイクル時間の 50%を使用。
→ tOFF = 1/25kHz × 50/100 × 1/2 = 10 µs

TOFF の設定に関しては、次のようになります：*TOFF* = (tOFF × f_{CLK} − 24)/32
クロック f_{CLK} が 16MHz の場合は *TOFF* = 4.25 という結果が得られ、*TOFF* は 4 に設定する必要があります。

ヒステリシスの開始設定値では、ドライバが最小限の量の電流リップルをモーター・コイルに導入します。最適なマイクロステッピングの結果を得るためには、この電流リップルが、モーターの抵抗性損失を原因とする電流リップルより大きいことが必要です。これによって、チョッパーは、立上がりおよび立下がりのどちらの目標電流に対しても正確に電流を安定化できます。モーター・コイルに電流リップルを導入するために必要な時間は、チョッパー周波数の減少ももたらします。そのため、ヒステリシスの設定値を高くすると、チョッパー周波数は低下します。モーターのインダクタンスは、チョッパーがモーター電流の変化に追随する能力を制限します。また、オン・フェーズの時間と高速減衰の時間は、ブランキング時間より長くなくてはなりません。これは、ブランキング時には電流コンパレータがディスエーブルされるためです。

最適な設定値を得るための最も簡単な方法は、低ヒステリシス設定（例えば、*HSTRT* = 0、*HEND* = 0）から始め、モーターが低速度設定で円滑に回転するようになるまで、*HSTRT* を増加させることです。これは、電流プローブを用いてモーター電流を測定すると、最もよく確認できます。ヒステリシスの設定値が小さすぎる場合、サイン波の波形をゼロ遷移付近でチェックすると、両半波の間に小さな棚形状が見られます。中速域（例えば1秒あたり100フルステップから400フルステップ程度）では、設定したヒステリシスが小さすぎるとモーターのハミング音や振動の増大を招きます。ヒステリシスの設定値が高すぎる場合は、チョッパー周波数が低下し、チョッパー・ノイズが増加しますが、波形には何ら好影響を及ぼしません。

ヒステリシスのデフォルト値として低い値から中程度の値では（例えば、実効ヒステリシス = 4）、ほとんどの場合で良好な結果が得られます。この設定は、モーターを用いて実際に試すことで最適化できます。設定が低すぎるとマイクロステップの精度が下がり、設定が高すぎるとチョッパー・ノイズとモーターの消費電力が増加します。高速減衰時間がブランキング時間よりわずかに長くなると、設定は最適なものとなります。これを実現するのが難しい場合は、オフ時間の設定値を減らすことができます。

電源電圧に比較してコイル抵抗が高い場合などのように、場合によっては、このヒステリシスの原理はチョッパー周波数が低くなりすぎる原因となります。これは、ヒステリシスの設定を開始設定（*HSTRT* + *HEND*）と終了設定（*HEND*）に分割することで回避できます。自動ヒステリシス・デクリメンタ（HDEC）は、ヒステリシス値を段階的に 16 システム・クロックごとにデクリメントすることで、両方の設定の間の補間を行います。ヒステリシスは、各チョッパー・サイクルの開始時に、開始値と終了値の和の値（*HSTRT* + *HEND*）で始まり、サイクルの間デクリメントを行い、チョッパー・サイクルの終了するまで、あるいはヒステリシスの最終値（*HEND*）に達するまで続きます。このようにして、チョッパー周波数は、周波数が低すぎるようになっても、大振幅かつ低電源電圧の状態で安定化します。これにより、周波数が可聴範囲に達するのが防止されます。

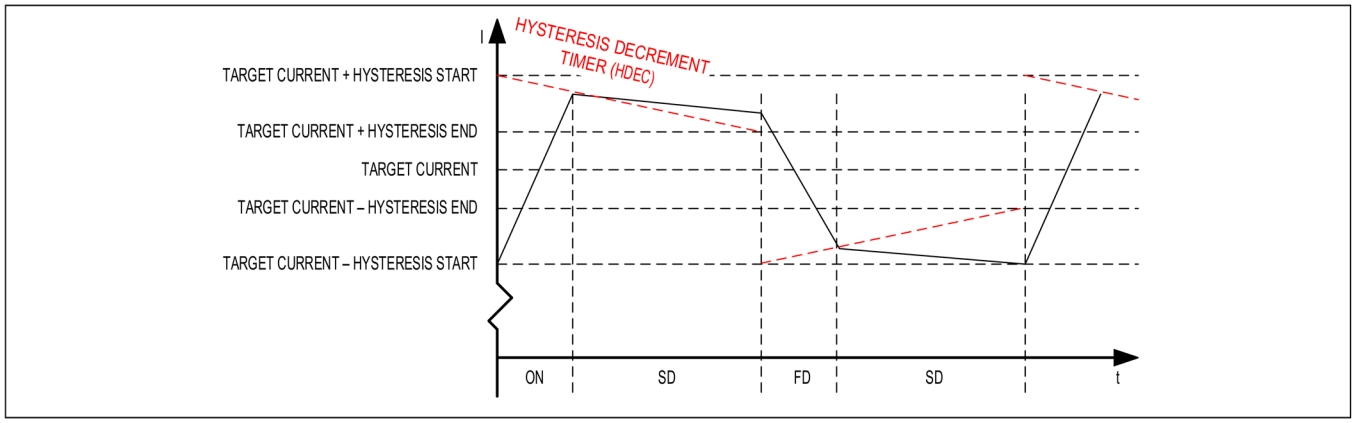


図 39. 1 チョッパー・サイクルの間のコイル電流を示す、SpreadCycle チョッパーの概略図

表 14. SpreadCycle モードのパラメータ

パラメータ	説明	設定	コメント
HSTRT	ヒステリシス開始設定値。この値は、ヒステリシス終了値 *HEND*からのオフセットです。 デフォルト = 5	0…7	*HSTRT* = 1…8 この値が *HEND* に加わります。
HEND	ヒステリシス終了設定値。いくつかのデクリメントを行った後のヒステリシス終了値を設定します。その和（*HSTRT* + *HEND*）は16以下にする必要があります。最大値251の電流設定の場合（振幅は245に低減）、この和には制限がありません。 変更されたサイン波を用いる場合、(*IRUN* + 1)/256によってスケーリングされたサイン波ピーク値が 256 – (*HSTRT* + *HEND*)を超えることのないようにしてください。これを確実に行うには、デフォルト波と同じようにサイン波のピークを248にスケール・ダウンするか、これに相当するよう*IRUN*および*IHOLD*を制限します。 デフォルト = 2	0…2	−3〜−1：負の *HEND*
		3	0：*HEND* はゼロ
		4…15	1〜12：正の *HEND*

例：

ヒステリシスとして 4 を選択します。最初のフィッティング設定：

HEND = 6（実効終了値を 6 - 3 = 3 に設定）

HSTRT = 0（最小ヒステリシスを設定。例えば 1 に設定すると：3 + 1 = 4）。

可変ヒステリシスを活用するには、値のほとんど（例えば 4）を *HSTRT* に設定し、残りの 1 をヒステリシス終了に設定します。その結果、設定レジスタ値は次のようになります。

HEND = 4（実効終了値を 1 に設定）。

HSTRT = 2（ヒステリシス終了に対する実効開始値を+3 に設定：1 + 3 = 4）。

直接モード

TMC2262 は、直接モードと呼ばれる、特別な直接モーター位相電流制御モードを備えています。このモードでは、SPI を用いることにより、モーターの位相ごとに専用の位相電流値を個別に指令できます。

GCONF レジスタ（0x0）のビット 6（*direct_mode*）をセットすることにより、直接モードに切り替わります。

レジスタ *DIRECT_MODE*（0x2D）は、符号付きのコイル A 電流（ビット 8～0）とコイル B 電流（ビット 24～16）を規定します。このモードでは、*IHOLD* 設定によって電流がスケーリングされます。

このモードは、SpreadCycle チョッパーとの併用のみが可能であり、StealthChop+の動作はサポートしていません。

一般的な定オフ時間チョッパー

一般的な定オフ時間チョッパーを、SpreadCycle の代替として用いることができます。定オフ時間チョッパーでは、各オン・フェーズの後に固定時間の高速減衰を使用します。オン・フェーズの時間はチョッパー・コンパレータによって決定される一方、高速減衰時間はドライバがサイン波の立下がり勾配に追随できるだけの長さであることが必要です。しかし、これは過大なモーター電流リップルおよび消費電力の原因となるため、長すぎてはいけません。これを調整するには、オシロスコープを用いるか、様々な速度でモーターの平滑さを評価します。良い開始値は、高速減衰時間を低速減衰時間と同じ設定にすることです。

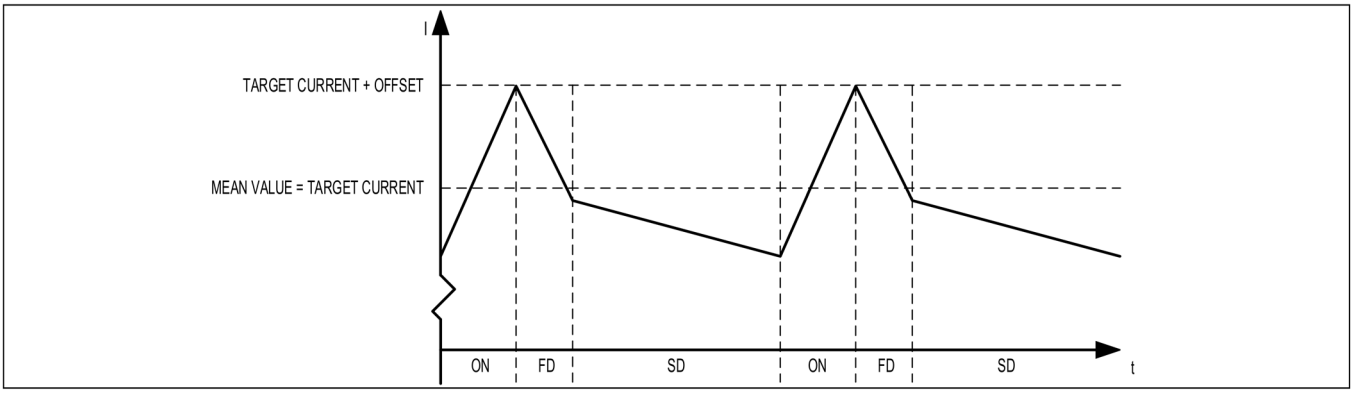


図 40. オフセットを持つ一般的な定オフ時間チョッパーが示すコイル電流

高速減衰時間を調整した後、ゼロ交差が滑らかになるようオフセットを調整する必要があります。これが必要なのは、高速減衰フェーズでは、モーター電流の絶対値が目標電流より小さくなるためです（図 41 参照）。ゼロ・オフセットが低すぎると、モーターは電流のゼロ交差時に短時間、停止します。ゼロ・オフセットの設定が高すぎると、マイクロステップが増大します。通常、最も滑らかな動作を実現するには、正のオフセット設定が必要です。

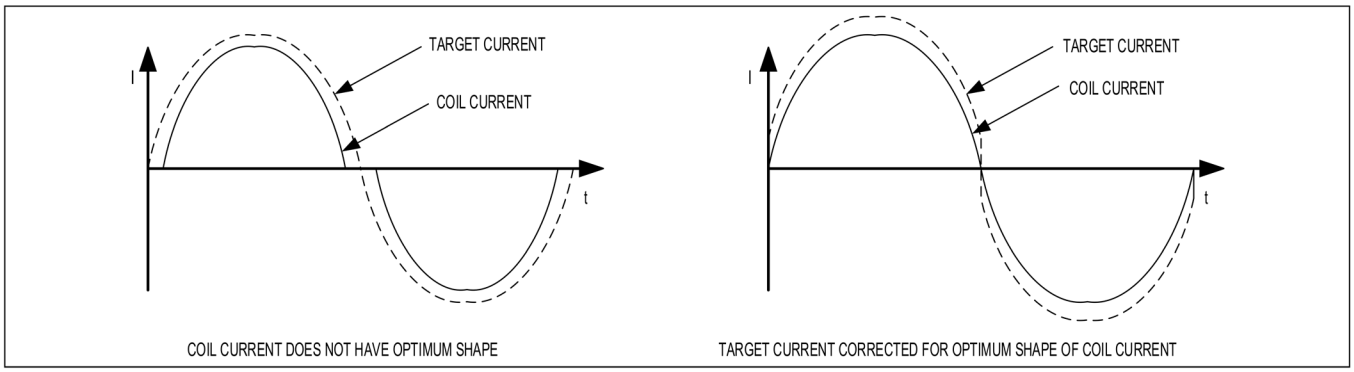


図 41. 一般的なチョッパーのゼロ交差とサイン波オフセットを使用した補正

表 15. 定オフ時間チョッパー・モードを制御するパラメータ

パラメータ	説明	設定	コメント
TFD (*fd3* and *HSTRT*)	高速減衰時間設定。*chm* = 1の場合、これらのビットが各チョッパー・サイクルの高速減衰の部分を制御します。 デフォルト = 5	0	低速減衰のみ
		1…15	高速減衰フェーズの時間
OFFSET (*HEND*)	サイン波オフセット。*chm* = 1の場合、これらのビットがサイン波のオフセットを制御します。オフセットを正にすると、ゼロ交差エラーを補正します。 デフォルト = 2	0…2	負のオフセット：−3～−1
		3	オフセットなし：0
		4…15	正のオフセット 1～12
disfdcc	高速減衰サイクルを終了させるための電流コンパレータの使用を選択します。電流コンパレータがイネーブルされていると、電流が実際の正の値より大きな負の値に達した場合に高速減衰サイクルが終了します。 デフォルト = 0	0	高速減衰サイクルのコンパレータによる終了をイネーブル
		1	時間でのみ終了

StallGuard2 の負荷測定

センサーレス・フィードバック・システムを用いて SpreadCycle 動作を補完するためにも、TMC2262 は StallGuard2 を備えています。StallGuard2 は SpreadCycle モードで動作します。これに対し、StallGuard+は StealthChop+動作と連携します。

StallGuard+と同様、StallGuard2 はモーターの負荷余裕を測定します。これは、ストール検出や、モーターをストールさせる下限未満の負荷の場合におけるその他の用途（CoolStep 負荷適応型電流低減など）で用いることができます。StallGuard2 の測定値は、広範な負荷、速度、電流の各設定値にわたり直線的に変化します。その絶対値（オフセット）は、動作条件や設定によって異なります。モーターの負荷が増加するにつれ、StallGuard の値（*SG_RESULT*）は減少します。ストールを正しく検出するには調整が必要です。モーターが過負荷／ストール状態となった場合に *SG_RESULT* が 0（あるいはほぼ 0）に達するように、StallGuard のスレッショルド（*SGTHRS*）を設定します。

ヒント：StallGuard2 と CoolStep を使用するには、最初に *SGT* 設定を使用して StallGuard2 の感度を調整してください。

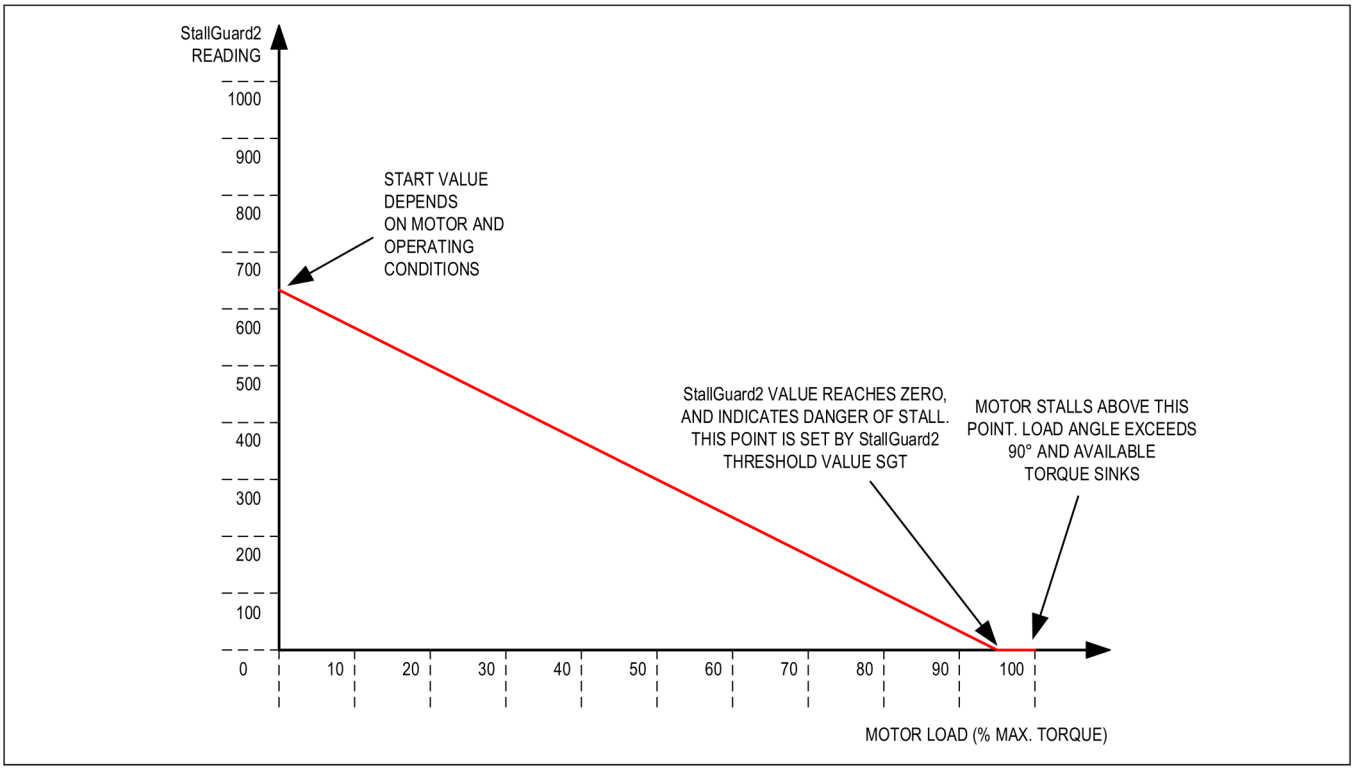


図 42. StallGuard2 の機能原理

表 16. StallGuard2 関連のパラメータ

パラメータ	説明	設定	コメント
SGT	この符号付きの値は、ストール検出用の StallGuard2 スレッショルド・レベルを制御し、読出し用に最適な測定範囲を設定します。値を小さくすると感度が高くなります。ゼロはほとんどのモーターで使用できる開始値です。値を大きくすると StallGuard2 の感度は低下し、ストールを示すために必要なトルクが大きくなります。	0	中立値
		+1…+63	低感度
		−1…−64	高感度
sfilt	測定精度を向上させるには StallGuard2 のフィルタをイネーブルします。これをセットすると、測定頻度がモーターの電気的周期（4 フルステップ）あたり 1 測定に減少します。	0	標準モード （ハード・ストール時のストール検出に最適）
		1	フィルタリング・モード
ステータス・ワード	**説明**	**範囲**	**コメント**
SG_RESULT	これは StallGuard2 の結果です。読出し値が大きいほど機械的負荷は小さいことを示します。読出し値が小さいほど負荷が高く、したがって負荷角度も大きいことを示します。モーター・ストール前の最大負荷時に *SG_RESULT* の読出し値が約 0～100 となるよう、*SGT* 設定値を調整してください。	0…1023	0：最大負荷 低値：高負荷 高値：低負荷

StallGuard2 のスレッショルド SGT の調整

StallGuard2 値 *SG_RESULT* は、モーター固有の特性と、アプリケーション固有の負荷、速度、電源電圧、電流レベルに対する要求による影響を受けます。そのため、StallGuard2 のスレッショルド *SGT* を特定のモーター・タイプおよび動作条件に合わせて調整するための最も簡単な方法は、実際のアプリケーションでインタラクティブに調整を行うことです。

表 17. StallGuard の調整プロシージャ

番号	タスク	詳細
1	モーターを動作	モーターをアプリケーションの通常の動作速度、電源電圧、電流設定で動作させ、*SG_RESULT* をモニタします。
2	負荷を印加し *SGT* を最適化	モーターに印加する機械的負荷を徐々に増加します。*SG_RESULT* がゼロになる前にモーターがストールした場合は、*SGT* を小さくします。モーターがストールする前に *SG_RESULT* がゼロに達した場合は、*SGT* を大きくします。*SGT* の良い開始値はゼロです。*SGT* は符号付きの値です。そのため、負または正の値が可能です。
3	モーション・コントロールをセットアップ	外部モーション・コントローラまたはマイクロコントローラを使用して、調整プロセスを自動化できます。そのため、stallguard 信号（*DRV_STATUS* レジスタのビット 24）を、診断出力ピン DO0 または DO1 のどちらかに割り当てることができます。
4	設定をテストし最適化	モーター・ストール直前の負荷増大時に *SG_RESULT* が 0 と約 100 の間にあり、負荷がない場合に *SG_RESULT* が 100 以上増加すれば、最適設定になっています。ほとんどの場合、*SGT* は特定の動作速度または動作範囲に合うよう調整できます。特定の範囲（例えば目的速度の 80%～120%）と極端なモーター状態下（適用される最低温度と最高温度）の両方で、その設定が確実に機能するようにしてください。

SGT の自動調整を可能にするオプション・プロシージャ：

SGT 設定の背景にある基本的な考え方は、StallGuard 測定をモーター内の抵抗性負荷に対して補償する、というものです。停止時あるいは極めて低い速度時には、抵抗性損失がモーターのエネルギー・バランスの主要因になります。機械的なパワーがゼロまたはほぼゼロであるためです。したがって、*SGT* は、速度がほぼゼロの場合の最適値に設定できます。このアルゴリズムは、環境条件やモーター浮遊容量などとは無関係に最高の結果を与えるよう、アプリケーション内で *SGT* を調整する場合に特に有用です。

表 18. 低速時に StallGuard を調整するプロシージャ

番号	タスク	詳細
1	モーターを動作	10RPM 未満（毎秒数フルステップ）の低速度かつ、目標動作電流および電源電圧でモーターを動作させます。この速度範囲では、*SG_RESULT* はモーター負荷にはそれほど依存しません。モーターは十分なバック EMF を発生しないためです。そのため、機械的負荷は結果に大きな違いを与えません。
2	高感度の *SGT* 設定を探索	*sfilt* をスイッチ・オンします。ここで、*SGT* を 0 から *SG_RESULT* が増加を始める値まで増加させます。*SGT* 値が一定の大きさになると、*SG_RESULT* は最大値に達します。再び、*SG_RESULT* がゼロにとどまる最大値まで減少させます。これで、SGT が可能な限り高感度の値に設定されます。速度が上がるにつれ *SG_RESULT* が増加するのを観察すると、有用なストール検出となります。
3	StallGuard を作動	モーターがストールすると *SG_RESULT* がゼロになります。モーターを停止するよう、外部モーション・コントローラまたはマイクロコントローラをプログラムできます。そのため、*stallguard* 信号（*DRV_STATUS* レジスタのビット 24）を、診断出力ピン DO0 または DO1 のどちらかに割り当てることができます。

この設定値を用いてストール検出を行うための上限速度は、モーターのバック EMF が電源電圧に近づき、速度を更に増加させた場合にモーター電流が低下し始める速度で決まります。

電源電圧は、*SG_RESULT* に影響します。そのため、レギュレーションを厳格に行うほど値の正確さが増加します。*SG_RESULT* の測定は高分解能であり、以降のセクションで説明するように精度を高めるための方法がいくつかあります。

可変速度制限値 *TCOOLTHRS* および *THIGH*

前述の *SGT* 調整に基づいて選択した *SGT* 設定は、ある特定の速度範囲で使用できます。この範囲を外れると、ストールが確実には検出されず、CoolStep が最適な結果を与えない場合があります。

多くのアプリケーションでは、ほとんどの時間で単一動作点あるいはそれに近い点での動作が用いられ、1 つの設定で十分です。ドライバは、これに合致する下限および上限速度スレッショルドを用います。決定した動作点の範囲外、例えば、*TCOOLTHRS* をマッチング値に設定する場合のセンサーレス・ホーミング・プロシージャに先立つ加速フェーズ時では、ストール検出はディスエーブルされます。上限値は *THIGH* で指定されます。

速度制限値 *VHIGH* および *VCOOLTHRS* は、*THIGH* および *TCOOLTHRS* の設定値で決まります。

アプリケーションによっては、少数のサポート点と線形補間を用いた *SGT* 値の速度依存調整が当を得ている場合があります。

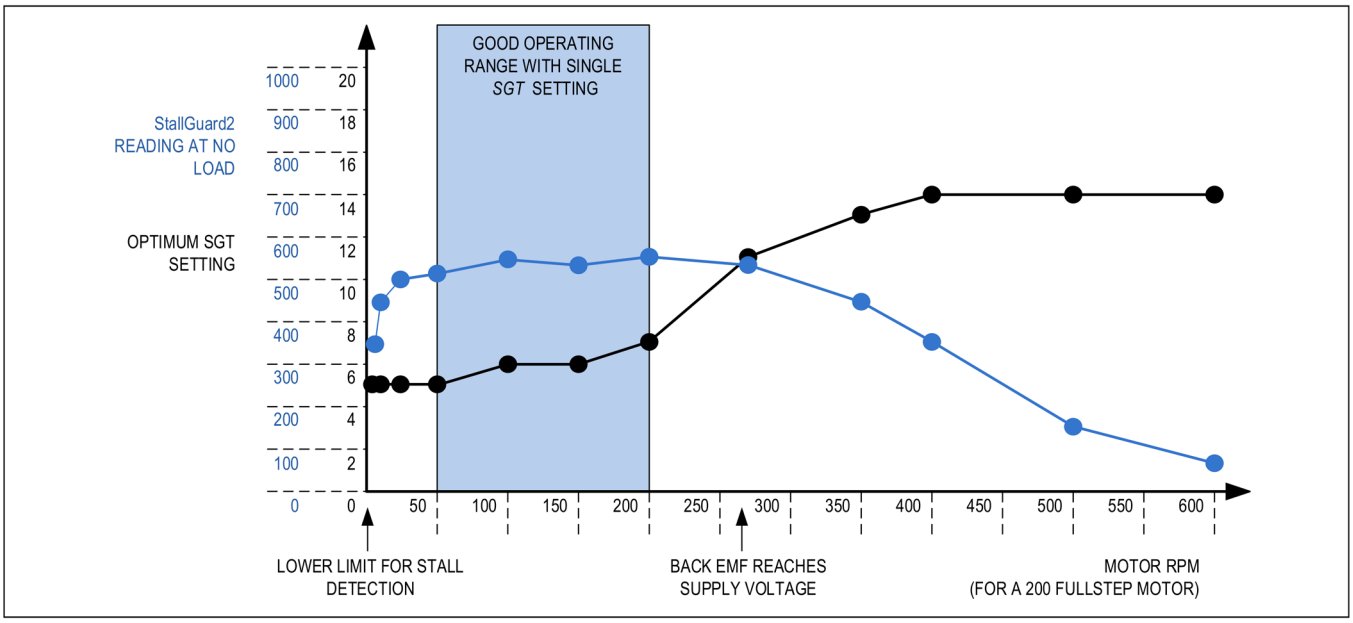


図 43. 例：サンプル・モーターを用いた SGT の最適設定値と StallGuard2 の読出し値

高トルク・リップルと共振を伴う小型モーター

保持トルクの高いモーターでは、モーター電流の変化に伴う StallGuard2 の測定値 *SG_RESULT* の変動が、特に小電流時に大きくなります。こうしたモーターでは、最善の結果が得られるよう、電流依存性を確認する必要があります。

モーター・コイル抵抗の温度依存性

広い温度範囲で動作するモーターでは、温度上昇に伴いモーターのコイル抵抗が増加するため、温度補正が必要となる場合があります。この補正は、モーター効率の低下と同様に、温度上昇に比例して *SG_RESULT* を直線的に減少させることで行うことができます。

StallGuard2 測定の正確さと再現性

量産環境においては、1 つのモーター・タイプ用のアプリケーション内で、固定 *SGT* 値を用いることが望ましい場合があります。StallGuard2 測定におけるユニットごとのばらつきのほとんどは、モーター構築時の製造許容誤差に起因します。StallGuard2 の測定誤差は、他のパラメータはすべて安定していると仮定すると、以下に示す程度まで低くできます。

$$\text{StallGuard2 measurement error} = +/- \max(1, |SGT|)$$

StallGuard2 の更新レートおよびフィルタ

StallGuard2 の測定値 *SG_RESULT* は、モーターのフルステップごとに更新されます。ストールでは必ず 4 つのフルステップが失われるので、この方法によりストールを十分確実に検出できます。実際のアプリケーション、特に、CoolStep を用いる場合では、より高精度の測定を行うことの方がフルステップごとに更新を行うより重要となる場合があります。機械的な負荷は決してステップごとに瞬時に変化することはないためです。こうしたアプリケーションでは、*sfilt* ビットによって、4 回の負荷測定にわたるフィルタリング機能を有効化できます。高精度測定が必要な場合は、このフィルタを常にイネーブルしてください。このフィルタは、例えば A 相と B 相の磁石のミスアライメントによる、モーター構造のばらつきを補償します。また、負荷増大に対する迅速な応答が求められる場合や、StallGuard を使用するセンサーレス・ホーミングにおいて最良の結果が求められる場合は、フィルタをディスエーブルする必要があります。

モーター・ストールの検出

最適なストール検出を行うには、StallGuard2 のフィルタリング機能を無効化します（*sfilt* = 0）。モーターのストールを確実に検出するには、特定の *SGT* 設定値を用いてストール・スレッショルドを決定する必要があります。そのため、モーターをストールさせずに駆動できる最大負荷を決定する必要があります。同時に、この負荷での *SG_RESULT* の値（例えば 0～100 の範囲のいずれかの値）をモニタします。ストール・スレッショルドは、パラメータの浮遊を許容できるよう、動作制限範囲に十分収まる値とする必要があります。*SGT* 設定値が 0 または 0 に近い場合の応答を調べると、信号品質に関する何らかの手がかりが得られます。無負荷時と最大負荷時の *SG_RESULT* の値をチェックしてください。これらは、少なくとも 100 または数百の差を示すはずで、これは主としてオフセットと比較されます。最大モーター負荷時に 0 の読出し値が得られるように *SGT* を設定している場合は、モーター・ストールは自動的に検出されてモーター停止を指示します。ステップロスの原因となるステップの瞬間には、最小の読出し値が示されます。ステップロスの後、モーターは振動し、*SG_RESULT* の読出し値は増加します。

StallGuard2 を使用したホーミング

線形駆動のホーミングでは、モーターをハード・ストップの方向に移動させることが必要です。StallGuard2 は、動作するには一定の速度が必要なので（*TCOOLTHRS* で設定）、加速フェーズに必要な距離が得られるよう、開始点がハード・ストップから十分離れているようにしてください。*SGT* のセットアップ後、stallguard 信号を DO0 または DO1 に設定し、ハード・ストップの方向にモーションを開始します。ストールが検出され、DO0 または DO1 で外部モーション・コントローラに出力されたら、外部モーション・コントローラはステップ信号のモーション生成を止め、モーターを停止します。ストップ条件は、*DRV_STATUS* のフラグ StallGuard によっても示されます。

StallGuard2 動作の制限

StallGuard2 は、モーター速度が極端な場合は信頼できる動作ができません。モーター速度が非常に低い（多くのモーターでは 1RPS 未満）場合、生成されるバック EMF は低く、測定は不安定で環境条件（温度など）に依存するようになります。前述の自動調整プロシージャによってこれらの効果の一部を補償できます。その他の条件でも、*SGT* の設定が極端な値になったり、測定値 *SG_RESULT* のモーター負荷に対する応答が低下したりする原因になります。広い動作範囲にわたって安定した StallGuard の結果が必要な場合は、StealthChop+での動作を検討してください。

サイン波電流の必ずしもすべてをモーター・コイルに供給することができないような、非常に高いモーター速度での動作の場合も、低応答の原因となります。これらの速度は、通常、モーターのバック EMF が電源電圧に達することで特徴づけられます。

CoolStep の負荷適応型電流スケーリング

CoolStep は、モーターの機械的な負荷に基づきステッピング・モーターに対しスマートなエネルギー最適化を自動的に行います。負荷が変動するアプリケーションでは、モーターの効率を飛躍的に増加させる可能性があります。実際のチョッパー・モードに応じて、CoolStep または CoolStep+は、StealthChop+では StallGuard+の負荷測定結果、SpreadCycle では StallGuard2 の負荷測定結果を自動的に使用します。CoolStep では、使用前に StallGuard2 を調整しておく必要があります。StallGuard2 と同様、CoolStep は 1 つの目標速度での動作に限定されます。1 つの調整がカバーする速度動作点は 1 つであるためです。

CoolStep 用の設定

CoolStep はいくつかのパラメータで制御されますが、その作用を理解するのに重要なのは 2 つです。

表 19. CoolStep の重要パラメータ

パラメータ	説明	範囲	コメント
SEMIN	下限スレッショルドを設定する 4 ビットの符号なし整数。*SG_RESULT* がこのスレッショルドを下回る（負荷が大きいことを示す）場合、CoolStep は両方のコイルへの電流を増加します。4 ビットの *SEMIN* の値が 32 倍され、10 ビットの *SG_RESULT* の値の範囲の下半分を占めます。（このパラメータの名前は、CoolStep の以前の名称である smartEnergy に由来するものです）。	0	CoolStep をディスエーブル
		1…15	スレッショルドは *SEMIN* × 32。1 が動作のための良い開始値です。
SEMAX	上限スレッショルドを制御する 4 ビットの符号なし整数。このスレッショルド以上の値の *SG_RESULT* が十分な回数サンプリングされる（軽負荷であることを示す）場合、CoolStep は両コイルへの電流を減少します。上限スレッショルドは、(*SEMIN* + *SEMAX* + 1) × 32 です。	0…15	スレッショルドは (*SEMIN* + *SEMAX* + 1) × 32。多くの場合、アプリケーションは、0 または 1（わずかなヒステリシス）の設定で最善の機能を示します。

図 44 に、CoolStep の動作領域を示します。

- 黒線は *SG_RESULT* の StallGuard 測定値を示します。
- 青線はモーターに加えられた機械的負荷を示します。
- 赤線は CoolStep によって定められた電流スケーリング *CS_ACTUAL* を示します。

機械的負荷が増加すると、*SG_RESULT* が *SEMIN* × 32 未満になり、CoolStep が電流を増加します。負荷が減少すると、*SG_RESULT* が (*SEMIN* + *SEMAX* + 1) × 32 を上回り、電流は減少します。

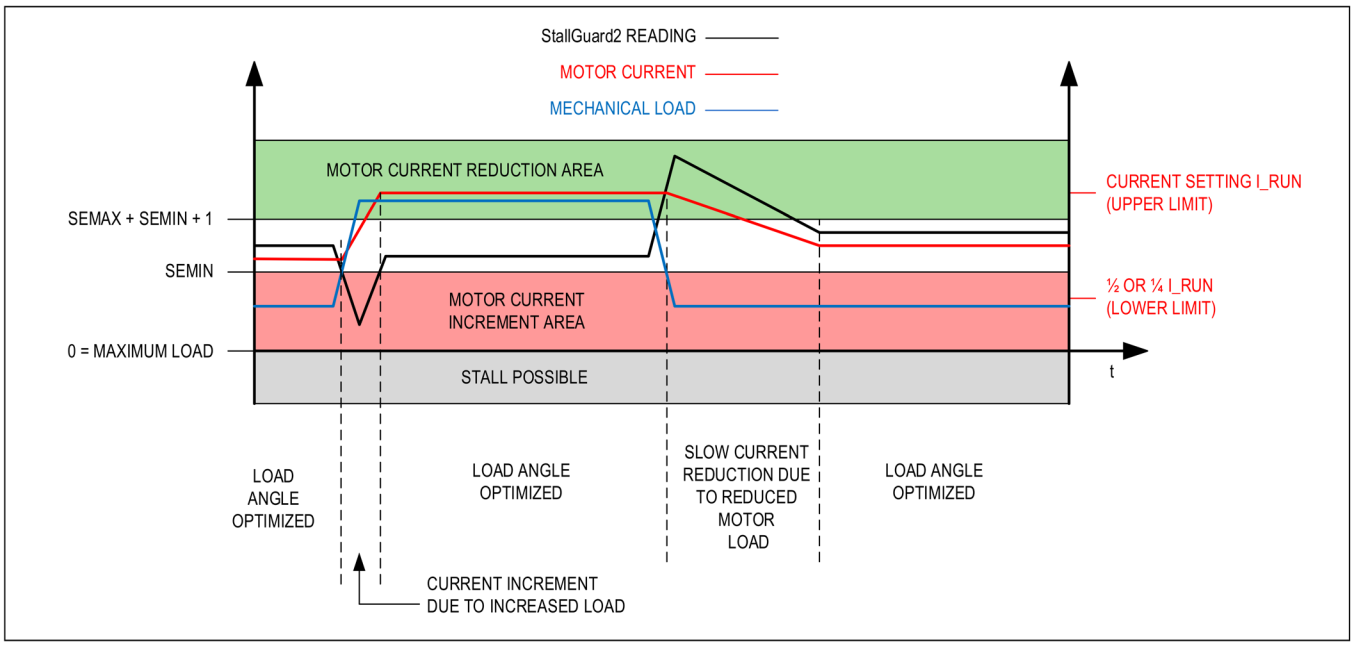


図 44. CoolStep によるモーター電流の負荷への適合

表 20. CoolStep のその他のパラメータとステータス情報

パラメータ	説明	範囲	コメント
SEUP	電流のインクリメント・ステップを設定します。*SEMIN* で設定された下限スレッショルド未満の新しい StallGuard2 値が測定されると、モーター電流は必ずこの設定値だけインクリメントされます。	0…3	*CS_ACTUAL* のステップ・アップ幅は 8、16、32、64 です。
SEDN	上限スレッショルドを超える StallGuard2 の読出し値の電流減少勾配を制御します。	0…7	StallGuard2の読出し値による *CS_ACTUAL*の低減量： 1/8、¼、½、1、2、4、8、16
SEIMIN	*IRUN* 電流設定値を½または¼にスケール・ダウンして、CoolStep 動作の下限モーター電流制限を設定します。	0	0：*IRUN*の1/2 （標準設定）
		1	1：*IRUN*の1/4 （限定された場合にのみ使用可能）
TCOOLTHRS	CoolStep に切り替えるための下限速度スレッショルド。この速度を下回ると CoolStep はディスエーブルされます。StallGuard2 が安定した結果を与える速度範囲の下限に合わせてください。 **ヒント**：スレッショルド速度を *VMAX* よりわずかに低い値に設定することで、加速フェーズおよび減速フェーズ時に CoolStep をディスエーブルするよう修正できます。	1… $2^{20} - 1$	スレッショルド値を *TSTEP* と比較することで、下限 CoolStep 速度を指定します。
THIGH	CoolStep用の上限速度スレッショルド値。この速度を超えると CoolStepはディスエーブルされます。StallGuard2または CoolStepが良好な結果を与える速度範囲に合わせてください。	1… $2^{20} - 1$	また、フルステップへの切り替えなど、その他の機能も制御します。

ステータス・ワード	説明	範囲	コメント
CS_ACTUAL	このステータス値は、CoolStep によって制御される実際のモーター電流スケールを示します。値の上限は I_{RUN} の値で、下限は *SEIMIN* で指定される *IRUN* の一部です。	0…255	CoolStep の動作をモニタするためにチェックします。

CoolStep の調整

CoolStep を SpreadCycle と併用して調整する前に、まず StallGuard2 のスレッショルド・レベル *SGT* を調整します。これは、負荷測定値 *SG_RESULT* の範囲に影響します。CoolStep では *SG_RESULT* を使用して、モーターを+90º の最適負荷角度付近で動作させます。

電流のインクリメント速度は *SEUP* で指定され、電流のデクリメント速度は *SEDN* で指定されます。これらは、異なる応答を必要とする可能性のある異なるイベントでトリガされるため、個別に調整できます。これらのパラメータのエンコーディングは、コイル電流の増加を減少よりもはるかに速く行うことができます。下限スレッショルドに達することは、高速応答を必要とする可能性のある、より厳しいイベントであるためです。応答が遅すぎる場合、モーターはストールする可能性があります。反対に、上限スレッショルドに達するまでの応答が遅い場合は、電力を節約する機会を失う以外に重大なリスクを生じることはありません。

CoolStep は、電流スケール・パラメータ *IRUN* と *seimin* ビットによって制御される制限値の間で動作します。

応答時間

モーター負荷の増加に対して高速に反応するには、電流インクリメント・ステップ *SEUP* に大きな値を用います。モーター負荷の変化が緩やかな場合は、電流インクリメント・ステップを下げることでモーターの発振を防止できます。*sfilt* で制御されたフィルタがイネーブルされている場合は、測定レートおよびレギュレーション速度は 1/4 になります。**アドバイス：**最も一般的で最も利点の多い使用法は、CoolStep を標準的なシステムの目標動作速度で動作するように調整し、それに応じた速度スレッショルドを設定することです。加速および減速は通常、短時間で行われるものなので、フル・モーター電流が必要ですが、時間が短いため全体の消費電力への影響はわずかなものにすぎません。

低速度およびスタンバイ動作

CoolStep は、停止時および非常に低 RPM 時のモーター負荷は測定できません。そのため、*TCOOLTHRS* によって、下限速度スレッショルドが設定されています。これには、アプリケーション固有のデフォルト値を設定する必要があります。このスレッショルド未満では、*IRUN* または *IHOLD* を通じた通常の電流設定が有効です。上限スレッショルドは *VHIGH* の設定により利用できます。速度制限値 *VHIGH* および *VCOOLTHRS* は、*THIGH* および *TCOOLTHRS* の設定値で決まります。

どちらのスレッショルドも StallGuard2 の調整プロセスの結果として設定できます。

内蔵電流検出機能

TMC2262 には、電力散逸のない電流検出機能が内蔵されています（ICS）。この機能により、外部電流検出機能を用いる場合に通常必要になる大きな外付け電力抵抗が不要になります。そのため、外付け検出抵抗に基づく主流のアプリケーションと比較して、ICS は大幅な省スペースと省電力を実現できます。最適性能を実現するために、ICS では各パワーMOSFET の $R_{DS(ON)}$を個別に測定し、最良の結果が得られるよう個々の MOSFET の温度を考慮します。

フル・スケール電流レンジの設定

最大フル・スケール電流 I_{FS} は、外部リファレンス抵抗 R_{REF} によって定義されます。精度 1%の標準的な低電力抵抗で十分です。通常、R_{REF} の値は 12kΩ です。抵抗がないと、フラグ *IOIN.ext_ref_det* が 0 のままになることによって通知されます。

一連の抵抗により、特定のアプリケーションおよびモーターに対しフル・スケール電流のスケーリングが可能になります。*CURRENT_RANGE* および *CURRENT_RANGE_SCALE* のグローバル設定を用いて、最初に目的のモーター電流範囲を調整することが必要です。StealthChop+動作では、*CURRENT_RANGE* が 0 より大きい（1A RMS より大きい）状態で動作している場合、*CURRENT_RANGE_SCALE* は、3（100%）にセットされます。これらの設定を選択すると、モーター動作に必要な最大電流に対応できます。通常、これは、モーターのデータシートで仕様規定された公称モーター電流値または最大でその 150%の値であり、これによって、短時間のモーター・トルクの増加が可能です。選択した範囲内で、個々のモーター、アプリケーション、動作状況に合わせて *IRUN* および *IHOLD* を用い、モーター電流を微調整します。CoolStep および CoolStep+による、負荷に応じた自動電流適応機能は、*IRUN* を最大値として用い、実際の機械的負荷に一致するよう、その一部まで自動的にスケール・ダウンします。モーター電流もリファレンス電流に合わせてスケーリングされる点に注意してください。そのため、リファレンス抵抗は厳格にマッチングするよう選択する必要があります。

I_{FS} は次式を用いて計算されます。

$$I_{FS} = CRS \times \frac{KIFS}{R_{REF}}$$

表 21 に、R_{REF} が 12kΩ（代表値）の場合に実現できる最大フル・スケール・レンジを示します。異なる値の R_{REF} を用いることは推奨しません。

表 21. R_{REF} = 12kΩ でのフル・スケール電流レジスタ設定

DRV_CONF REGISTER BITS 1...0 (CURRENT_RANGE)	KIFS [A x kΩ]	DRV_CONF REGISTER BITS 3…2 (CURRENT_RANGE_ SCALE)	CURRENT RANGE SCALE (CRS)	MAX. FULL-SCALE SETTING I_{FS}		TYPICAL $R_{DS(ON)}$ (LS) (Ω)
				[A] PEAK	[A] RMS	
11	72	11	100%	6.000	4.24	0.036
10	54	11	100%	4.500	3.18	0.047
1	36	11	100%	3.000	2.12	0.070
00 (default)	18	11	100%	1.500	1.06	0.138
		10	75%	1.125	0.795	
		01	50%	0.750	0.530	
		00 (default)	25%	0.375	0.265	

これらの値は StealthChop+に対して有効です。SpreadCycle 電流は、これと同じ設定の場合、わずかに低くなります。デフォルトのサイン波振幅が 248/256 = 97%に制限されており、また、低速の減衰時間の間の電流低下によりヒステリシス・チョッパーの平均電流が目標電流をわずかに下回るためです。SpreadCycle 動作に対するより正確な計算式は以下の式で与えられます。これはサイン波振幅を考慮していますが、両方のチョッパー原理を組み合わせるアプリケーションに対しては、一般に StealthChop+の式を用いれば十分です。

$$RMSCurrent_{Stealth} = 4.24A \times \frac{CURRENT_RANGE + 1}{4} \times \frac{CURRENT_RANGE_SCALE + 1}{4} \times \frac{IRUN}{250}$$

$$RMSCurrent_{spread} = 4.24A \times \frac{248}{256} \times \frac{CURRENT_RANGE + 1}{4} \times \frac{CURRENT_RANGE_SCALE + 1}{4} \times \frac{IRUN + 1}{250}$$

ここで、248/256 は、サイン波テーブルの振幅です。

注意：*IRUN* の制限：*IRUN* を選択すると、内部 ADC の範囲の使用に直接影響します。フル・レンジは、*IRUN* ≥ 250（StealthChop+）または *IRUN* = 255（SpreadCycle）で動作している場合に用いられます。低い電流スケーリング（*IRUN* または、*COOL_CUR_DIV* によってスケール・ダウンされた *IRUN*）は、ADC の読出し値に比例して減少し、それによって、StallGuard 測定に使用可能な電流および角度レギュレーションの範囲もそれぞれ減少します。最高性能を得るには、最低の *CURRENT_RANGE* を選択し、それと共に、必要なモーター電流を供給する最大設定の *IRUN* を選択します。レンジの 10%未満に電流をスケール・ダウンすることは避けてください。

表 22 に示すパラメータは、電流スケーリングや電流のランプアップおよびランプダウンに適応させることができます。

表 22. モーター電流を制御するパラメータ

パラメータ	説明	設定	コメント
CURRENT_RANGE	モーターの動作前にモーターの最大電流に見合うよう選択し、アプリケーション内でこの値を固定したままにします。このように設定されたピーク電流は、下側 MOSFET のオン抵抗を低い値に変更することによって、より大きなモーター電流に適応します。	0…3	ドライバの電流レンジを 1.5A、3A、4.5A、6A peak に選択します。
CURRENT_RANGE_SCALE	モーターの動作前にモーターの最大電流に見合うよう選択し、アプリケーション内でこの値を固定したままにします。StealthChop+の場合、100%の設定を *CURRENT_RANGE* > 0 と組み合わせて保持する場合に最高の結果が得られます。	0…3	実際の電流レンジを 25%、50%、75%、100%にスケーリングします。小型モーターには、0～2 の値を適用し *CURRENT_RANGE* を 0 にスケール・ダウンします。
IRUN	モーター動作中の電流スケーリング。内蔵のサイン波テーブルから取得したコイル電流値をスケーリングします。高精度のモーター動作を実現しCoolStep(+)の最高性能を得るためには、電流スケーリング係数を128～250（255）の範囲にします。電流値をスケーリング・ダウンするとマイクロステップが粗くなり、実効的なマイクロステップ分解能が低下するためです。この設定は、CoolStep(+)で設定される最大電流値も制御します。 デフォルト = 240	0…250 (0…255)	スケーリング・ファクタ 0/256、1/256、…250/256、または SpreadCycle の場合、1/256…256/256。
IHOLD	*IRUN*と同じですが、停止時のモーター用です。*IHOLD*の下限値はアプリケーション固有であり、保持トルクが不要な場合は*IHOLD*を0に設定することもできます。 デフォルト = 64		
IHOLDDELAY	動作電流からホールド電流への滑らかな電流減少を可能にします。*IHOLDDELAY*は、モーターの*TZEROWAIT*後のパワー・ダウンのためのクロック・サイクル数を、2^{11}クロックのインクリメントで制御します。0 = 即時パワー・ダウン、1～255：2^{11}クロックの倍数で表した電流ステップあたりの電流低減遅延。 **例**：IRUN = 240およびIHOLD = 40の場合、ホールド電流を低減するには、200電流ステップが必要です。したがって、*IHOLDDELAY*を4に設定すると、パワー・ダウン時間は 4 × 200 × 2^{11}クロック・サイクルとなります。例えば、16MHzの場合100ms秒になります。 デフォルト = 7	0	I_{HOLD} への即時パワー・ダウン。
		1…255	電流デクリメントあたり、1 × 2^{11}～15 × 2^{11} クロック。
IRUNDELAY	開始が検出されてからモーターが起動するまでのクロック・サイクル数を制御します。 動作の開始時に、ホールド電流（*IHOLD*）から動作電流（*IRUN*）への滑らかな電流インクリメントを可能にします。フル・モーター・トルクを確立するには、短時間でのパワーアップが重要ですが、わずかなランピング時間を設けることで、可聴ノ	0	IRUN への即時パワーアップ。
		1...15	電流インクリメント・ステップあたりの遅延は *IRUNDELAY* × 64 クロックの倍数です。

	イズを低減し、また、電源電流のオーバーシュートを防止できます。 **例**：IRUN = 240 および IHOLD = 40 の場合、動作電流をランプアップするには、200 電流ステップが必要です。したがって、*IRUNDELAY* を 4 に設定すると、パワー・ダウン時間は 4 × 200 × 64 クロック・サイクルとなります。例えば、16MHz の場合 3.2ms になります。 デフォルト = 4		
TPOWERDOWN	停止（stst）検出後からモーター電流がパワー・ダウンするまでの遅延時間を設定します。時間範囲は、約0秒～4.2秒です。 大幅な温度上昇が予想されるアプリケーションでは、一定の最小値を設定するとモーターのコイル抵抗を測定するのに役立ちます。 デフォルト = 10	0	最速のパワー・ダウン
		1…255	2^{18}クロックの倍数の遅延。
fast_standstill	停止状態が検出されるまでステップ・パルス・タイムアウトを制御します。デフォルト設定の 0 は、最小速度設定 16 に相当します。最速設定は、迅速な *TPOWERDOWN* のタイミングと併用することを推奨します。これは速度 64 に相当します。	0	2^{20} クロック
		1	2^{18} クロック

ADC 値の解釈

StealthChop+の内部電流測定では、両モーター・コイルの電流を評価するのに ADC を用います。実際の電流は、*ADC_I_A* レジスタおよび *ADC_I_B* レジスタから読み出され、コイルあたりの実効値電流は、両方から計算された *AMPL_MEAS* で与えられます。ADC のスケーリング、および、それに加えて ADC 値の変換結果は、*CURRENT_RANGE* および *CURRENT_RANGE_SCALE* の各電流設定値によって異なります。これらの値は、SpreadCycle 動作の間は使用できません。*ADC_I_A* および *ADC_I_B* 用に評価される電流は、ピーク値が実効値の SQRT(2)倍の値を超える一時的な電流です。

$$RMSCurrent = 4.24A \times \frac{CURRENT_RANGE + 1}{4} \times \frac{CURRENT_RANGE_SCALE + 1}{4} \times \frac{AMPL_MEAS}{2000}$$

$$ADCCurrent = 6.0A \times \frac{CURRENT_{RANGE} + 1}{4} \times \frac{CURRENT_{RANGE_{SCALE}} + 1}{4} \times \frac{ADC_I_A \mid ADC_I_B}{2000}$$

勾配制御

TMC2262 は、その電力段出力用にプログラマブルな勾配制御機能をサポートしています。それによって、*SLOPE_CONTROL* 設定値を用いることで出力勾配を 4 段階に形成できます。ドライバの消費電力を最低にするには最速の勾配が最善ですが、勾配を下げることで電磁妨害波が減少し、電源電圧のリンギングおよびモーター・ケーブルに隣接するケーブルへのカップリングが低減します。400V/µs 未満の最も遅い設定を用いるのは、低動作速度の状況に限定する必要があります。24V を超える電圧や 2A を超える電流設定で動作させる場合は特に、最速の勾配設定を選択することを推奨します。

一般に、最速の勾配で開始し勾配を下げるという制御は、勾配に関連する問題を PCB レイアウトやケーブル配線の最適化といった別の手段では解決できない場合にのみ行います。

速度ベースのモード制御

TMC2262 では、最適なモーター制御を実現するために、様々なチョッパー・モードと動作モードが設定可能です。モーター負荷に応じて、最小ノイズおよび高精度、最高動的性能、あるいは最高速時の最大トルクを実現できるよう、様々なモードを最適化できます。限定された速度範囲では、CoolStep、CoolStep+、StallGuard2、StallGuard+などの一部の機能が有用です。いくつかの速度スレッショルドでは、広い速度範囲を必要とするアプリケーション内で異なる動作モードを組み合わせることができます。

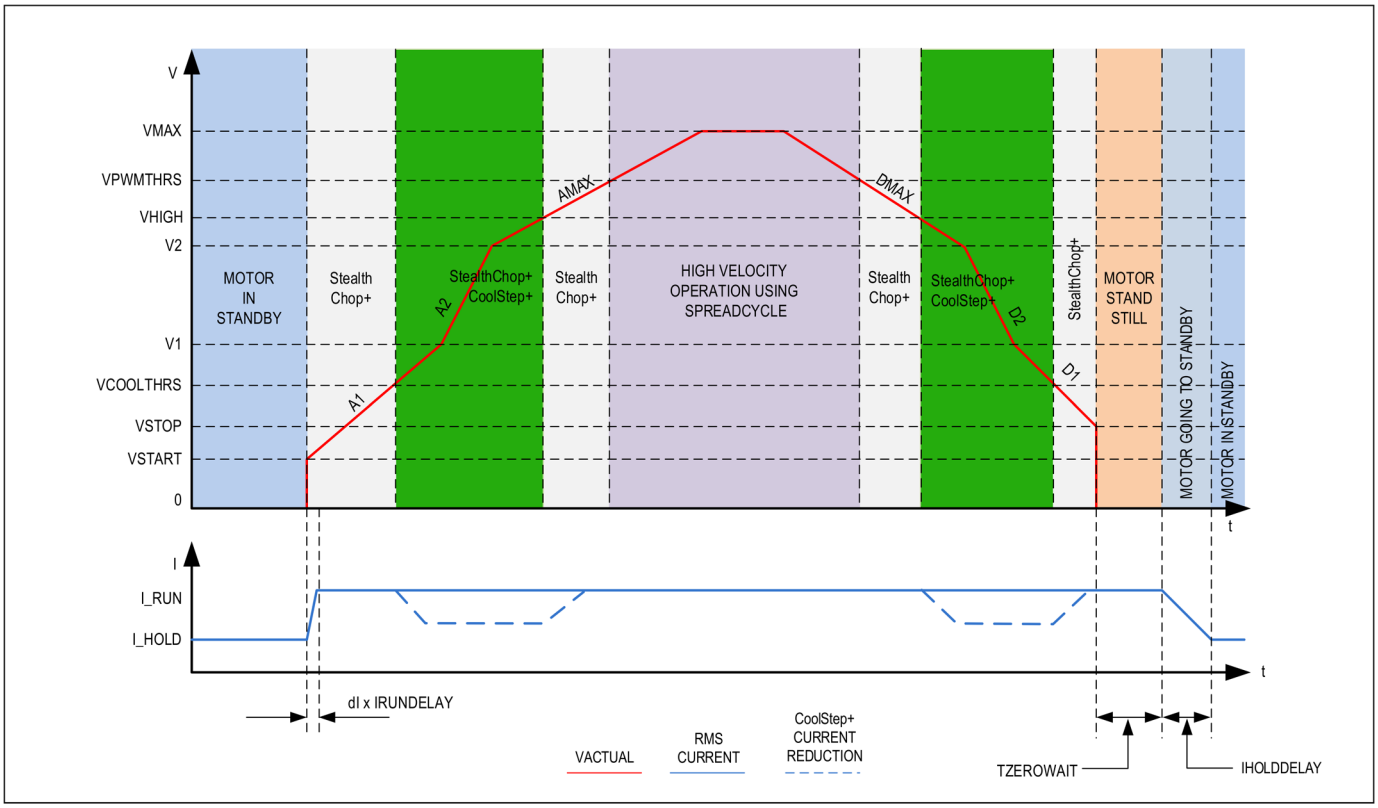


図 45. 速度依存モードの選択

図 45 に、最も重要なスレッショルドと必要な順序を示します。VPWMTHRS、VHIGH、VCOOLTHRS は、TPWMTHRS、THIGH、TCOOLTHRS の設定によって決まります。速度は、2 つのステップ・パルスの時間間隔 TSTEP で記述されます。それにより、外部ステップ源を用いた場合でも速度を決定できます。TSTEP は、常に 256 マイクロステップに正規化されます。したがって、マイクロステップの分解能が変化した場合でも、スレッショルドをそれに合わせて変化させる必要はありません。スレッショルドは、マイクロステップの設定値とは無関係に、同じモーター速度を表します。TSTEP は、これらのスレッショルド値と比較されます。TSTEP の測定でジッタが生じた場合に、比較した結果が連続的に切り替わるのを避けるために、1/32TSTEP または 1/16TSTEP のヒステリシスが自動的に適用されます。上限のスイッチング速度は、スレッショルドとして設定した 1/32 または 1/16 の値より高くなります（*GCONF* レジスタの設定ビット *small_hysteresis* で選択します）。モーター電流は、停止フラグ *stst* に応じて、動作レベルまたはホールド・レベルにスケーリングします。

自動速度スレッショルドを用いるとアプリケーションを様々な速度レンジに調整できます。CoolStep のような機能は、セットアップにおいてトランスペアレントに統合できます。したがって、一度パラメータを設定すれば、ソフトウェアを使ってアクティブにしたり非アクティブにしたりする必要はありません。

表 23. 速度ベースのモード制御パラメータ

パラメータ	説明	設定	コメント
stst	各動作モードでモーターが停止していることを示します。時間は最後のステップ・パルスの、2^{20} クロック後または 2^{18} クロック後です（*fast_standstill* で選択）。 デフォルト／リセット：0	0/1	ステータス・ビット、読出し専用
TPOWERDOWN	これは、モーターの停止検出（stst）後からモーター電流がパワー・ダウンするまでの遅延時間です。時間範囲は、約0～4秒（f_{CLK} = 16MHzの場合）です。 0に設定すると遅延なし、1に設定すると1クロック・サイクルの遅延です。 それ以上は、2^{18}クロック・サイクルの離散的なステップでインクリメントします。 デフォルト：10	0…255	$2^{18} \times t_{CLK}$ の倍数で表した時間。

TSTEP	ステップ入力周波数から引き出される2つの1/256マイクロステップ間の実際の測定時間（1/fCLK単位）。オーバーフローまたは停止時の測定値は、$(2^{20})-1$です。 デフォルト／リセット：0	0… 1048575	ステータス・レジスタ、読出し専用。t_{CLK}の倍数で表した実際の測定ステップ時間。
TPWMTHRS	*TSTEP* < *TPWMTHRS* • StealthChop+はディスエーブルされ、ドライバはSpreadCycleに切り替わります。 デフォルト：0	0	StealthChop+を全速度範囲で使用可能。
		1… 1048575	StealthChop2 の動作用上限速度スレッショルドを制御するための設定値。
TCOOLTHRS	*TCOOLTHRS* ≥ *TSTEP* ≥ *THIGH*: • StallGuard2とCoolStepがイネーブルされます（設定されている場合）。またはStallGuard+とCoolStep+がイネーブルされます。 *TCOOLTHRS* ≥ *TSTEP* • ストール時に停止する機能が有効化され、StallGuard(+)のストール出力信号がイネーブルされます（設定されている場合）。 デフォルト：0	0	CoolStep/CoolStep+オフ。
		1… 1048575	CoolStep と StallGuard の動作用下限速度スレッショルドを制御するための設定値。
THIGH	*TSTEP* ≤ *THIGH*: • 基準速度を超えるとCoolStepやCoolStep+はディスエーブルされます（モーターは通常の電流スケールで動作します）。 デフォルト：0	0	上限速度スレッショルドなし。
		1… 1048575	CoolStep(+)動作用上限スレッショルドを制御するための設定値。
small_hysteresis	*TSTEP*（下限速度スレッショルド）と(*TSTEP* × 15/16) − 1または(*TSTEP* × 31/32) − 1（上限速度スレッショルド）に基づくステップ周波数比較のヒステリシス。 デフォルト：1	0	ヒステリシスは 1/16（ジッタの増加した STEP/DIR ソースでのみ推奨）。
		1	ヒステリシスは 1/32。
en_stealthchop	StealthChop+イネーブル・フラグ（速度スレッショルドに依存）。停止時にのみオフ状態からオン状態に切り替わります。 デフォルト：0	0	StealthChop+不使用。
		1	StealthChop+アクティブ（設定されている場合）、および、*TSTEP* > *TPWMTHRS*。

TriCoder - バック EMF センサーレス停止ステップロス検出

TriCoder 機能は、モーターのバック EMF（BEMF）を利用する、センサーレスの停止ステップロス検出機能です。

BEMF デコーダは、モーターがディスエーブル時（モーター・コイルに有効な電流が加えられていないとき）のモーター動作を検出できます。この機能は、消費電力を節約する要求があるために保持電流を印加できないながらも、通常一定量のコギング・トルクまたは摩擦がモーターを定位置に保持するようなデバイスに対して、特に有用です。TriCoder は、モーターが外力によって回転している場合のモーターの動きを追跡します。その結果を利用して、モーターが動いた場合の新しいホーミング・シーケンスをトリガしたり、ステップ数の追跡を行い続けその後にそれを補正したりできます。

それ以外に、モーターを入力デバイスとして用いて、手動のティーチインを行う、またはユーザとのインタラクションを行う、という使い方があります。

感度と精度に関する考慮事項：

- この原理では、モーターによって生成される一定の最小 BEMF 電圧が動作を検知する必要があります。モーターによっては、この必要な BEMF レベルは、数 RPM の範囲で容易に到達でき、したがって動作の発生は容易に検知できます。
- 動作開始時または終了時には、1 ステップまたは数ステップが失われたり、正しくカウントされなかったりする可能性があります。これと実際のモーターおよびアプリケーションの機構との関連性をチェックする必要があります。
- システムは、1 フルステップの倍数でカウントを行います。エンコーダのスケーリング係数をマイクロステップの分解能に合うよう設定することにより、変更されたモーター・ポジションを直接追跡できます。

TriCoder BEMF デコーダの動作原理

TriCoder BEMF デコーダは、モーターがディスエーブルの間、または、ローサイド・ドライバを使用した受動ブレーキングでモーターが停止させられた場合、モーターの動作を追跡できます。モーターの動作を検知するために、ドライバ IC は、モーター・コイルの電圧をチェックし、それをプログラマブルなヒステリシス・スレッショルド一式と比較します。図 46 に動作の原理を示します。

ディテクタ回路は、プログラマブルなヒステリシス・スレッショルドと帯域幅制限の組み合わせを使用し、モーター・ラインに結合した電圧スパイクなどによる誤ったトリガを回避します。

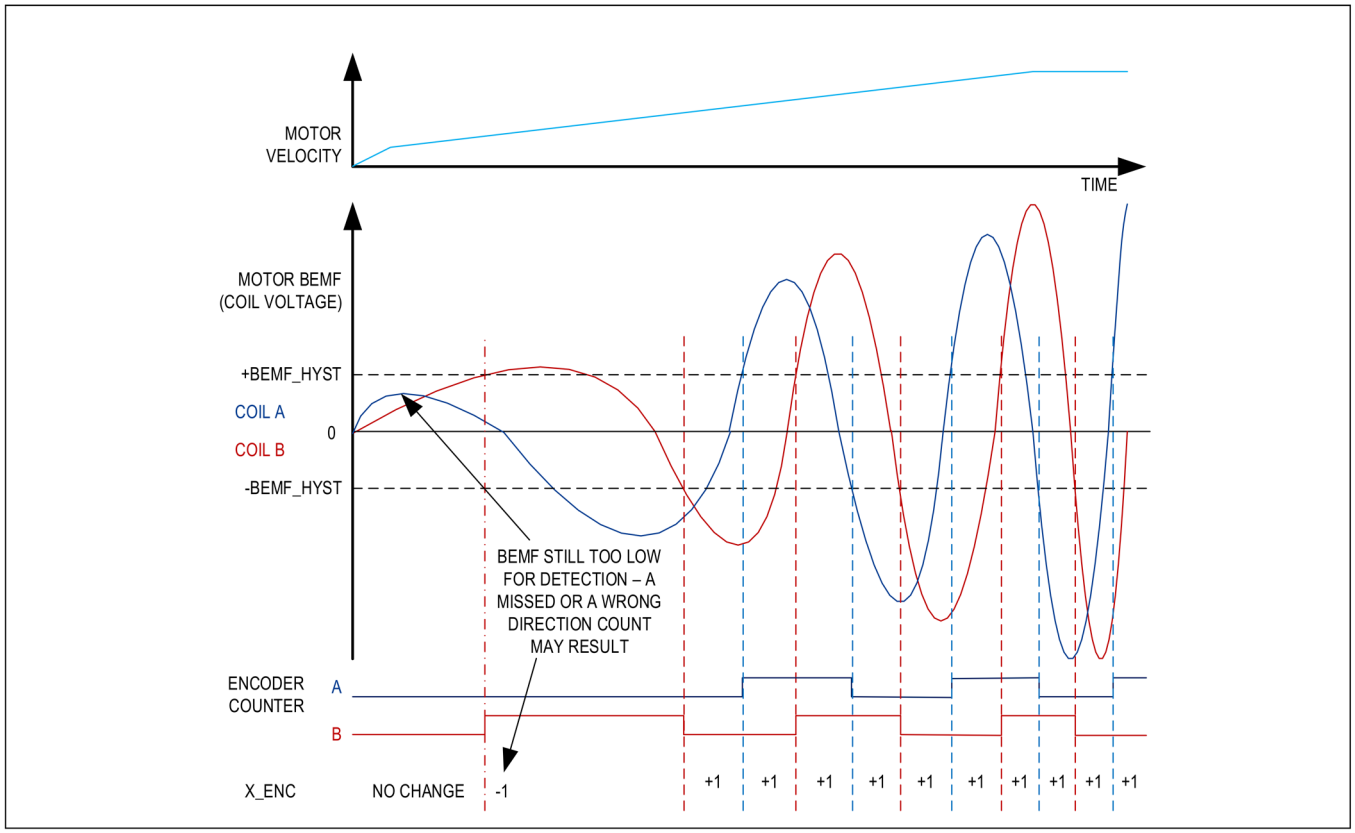


図 46. モーター動作の検知

TriCoder 動作に対するモーターおよび速度の条件

特定のヒステリシス設定を用いて検知できる最低速度を決定するには、モーターのバック EMF の電圧をチェックします。

$$C_{BEMF}\left[\frac{V}{rad/s}\right] = \frac{HoldingTorque[Nm]}{2 \times I_{COILNOM}[A]}$$

$$U_{BEMFopen} = \frac{HoldingTorque[Nm]}{2 \times I_{COILNOM}} \times \frac{2\pi \times Velocity[RPM]}{60}$$

受動ブレーキング・モードでは、モーターはコイルを短絡することで電気的に制動されます。したがって、ある特定のモーター速度に対するエンコーダの感度が低下します。最初に生成されるバック EMF 電圧はモーター・コイルの抵抗から生じるモーター電流に変換される必要があるためです。この電流が再び、ドライバ内のローサイド MOSFET スイッチの $R_{DS(ON)}$の電圧に転送されます。このモードで最高の結果を得るには、電力段の電流設定 *CURRENT_RANGE* として最低の値を選択し、コイル抵抗に対するローサイド $R_{DS(ON)}$の比率を増加させます。

$$U_{BEMFbrake} = \frac{HoldingTorque[Nm]}{2 \times I_{COILNOM}} \times \frac{2\pi \times Velocity[RPM]}{60}) \times \frac{RDSon}{RDSon + R_{COIL}}$$

ここで、

$R_{DS(ON)}$は、ローサイド MOSFET の抵抗であり、電気的特性の表の $R_{DS(ON)}$ (LS)から取得できます。

TriCoder のイネーブルまでの時間

TriCoder 機能を動作させるには、モーターがフリーホイーリング状態になっているか、残留コイル電流のない受動ブレーキング・モードになっていることが必要です。モーターの残留コイル電流が確実になくなるよう、それ以前の動作に応じて一定の遅延時間が必要となることがあります。モーター停止の検出に続き（停止、下限動作速度の要求に従い、*fast_standstill*を用いて停止検出時間を 16ms または 65ms に設定します）、*IHOLDDELAY* の設定値による制御に従って、モーター電流がゼロ電流までランプダウンします。高速の *IHOLDDELAY* 設定（0 は即時パワー・ダウン）でインダクタンスが高いモーターを用いる場合、残留モーター電流を電源に戻すことでこれを減衰させるために、数百マイクロ秒から数ミリ秒の追加時間（*BEMF_BLANK_TIME*）が必要となることがあります。残留電流はモーターの BEMF のように見えることがあり、そうでない場合は、ステップ誤検出をもたらすこともあります。

TriCoder の設定

TriCoder のイネーブル

モーターがフリーホイーリング・モードになっている場合は常に、TriCoder 機能を自動的にイネーブルするために以下の設定が必要です。

- *nBEMF_ABN_sel*：外部エンコーダ入力をディスエーブルするには 0 に設定します。モーターは停止状態（*stst* = 1）であることが必要です。
- *PWM_CONF.FREEWHEEL*：1（フリーホイール）または 2（LS ドライバを使用する受動ブレーキング）に設定します。

StealthChop+において：

- *IHOLD*：停止時にフリーホイーリング・オプションをイネーブルするには 0 に設定します。モーターが停止状態で *CS_ACTUAL* が 0 に達している場合にのみ、フリーホイーリングがイネーブルされます。

SpreadCycle において：

- モーターをディスエーブルしフリーホイーリングを可能にするには、*TOFF* = 0 に設定します。

動作設定

以下に説明する一連のパラメータを用いて TriCoder をイネーブルします。更に、*X_ENC* カウンタがモーターのマイクロステップ分解能に一致するよう *ENC_CONST* を使用します。例えば、256 マイクロステップ設定の場合は 256 です。この設定により、モーターのフルステップあたり *X_ENC* が 256 だけ増加／減少します。エンコーダがアクティブになる前に、*X_ENC* を 0 に設定して、元のポジションからの偏差をカウントしたり、TriCoder の動作時の実際のポジションを追跡したりできます。エンコーダの動作中にステップ差または更新されたモーター・ポジションを読み出します。N チャンネルは TriCoder 機能と共に使用することはできません。

表 24. TriCoder の制御パラメータ

パラメータ	説明	設定	コメント
nBEMF_ABN_sel	エンコーダ入力の選択 デフォルト／リセット：0	0	TriCoder：エンコーダはモーターのバック EMF を使用します。
		1	ABN 入力を使用する外部インクリメンタル・エンコーダ。
IHOLD	モーターの停止電流。モーターが停止している間に TriCoder をイネーブルするには 0 にセットします（StealthChop+と組み合わせた場合のみ有効）。表 22 を参照してください。停止検出までの時間および電流が 0 に減少するまでの時間を短縮するには、*IHOLDDELAY* および *fast_standstill* もチェックします。	0	モーター停止中に TriCoder がイネーブル状態。
		1…255	TriCoder は使用不可。
FREEWHEELING	*CS_ACTUAL* = 0 の場合に、フリーホイーリングまたは受動ブレーキング・モードのどちらを StealthChop+と組み合わせるかの選択。（*IHOLD* が 0 に減少後）	0	チョッパーはオンのまま。TriCoder は使用不可。
		1	モーターはフリーホイーリングになります。高感度の TriCoder。
		2	モーターはローサイドによる受動ブレーキング。TriCoder は使用可能。ただし、LS ドライバ FET を使用してモーターの BEMF が短絡されるため感度が低減します。
		3	モーターはハイサイドによる受動ブレーキング。TriCoder は使用不可。
BEMF_HYST	BEMFエンコーダ用のヒステリシス設定。ヒステリシスを大きくすると、ノイズ抑制が低下します。ただし、モーション検出のための下限速度は増加します。モーター出力のリップルがモーションの誤検出の原因になる場合は、これを増加させます。 デフォルト：0（最高感度）	0…7	10mV/25mV/50mV/75mV/100mV/150mV/200mV/250mV
BEMF_BLANK_TIME	電流を 0 まで減少させた後 TriCoder をイネーブルするまでの追加待機時間。残留するモーター電流や発振により生じるスパイクの影響を回避するために使用します。TriCoder に切り替えると誤動作が検出される場合は、この値を増加してください。	0…255	ブランキング時間は 2^14 × 1/F_{CLK} の倍数です。
BEMF_FILTER_SEL	両方のモーター・コイル電圧に適用し擾乱をフィルタにより除去するための BEMF フィルタの帯域幅を制限します。直交エンコーディングにより、理論上のカウント・レートは最大で帯域幅の 4 倍になります。予測される最大カウント・レートを十分に上回るよう設定してください。	0	帯域幅：500Hz
		1	帯域幅：1kHz
		2	帯域幅：2kHz
		3	帯域幅：4.3kHz

ABN インクリメンタル・エンコーダ・インターフェース

TMC2262 は、ABN エンコーダに対応したインクリメンタル・エンコーダ・インターフェースを備えています。エンコーダは、デジタル・インクリメンタル直交信号（通常 A および B と名付けます）およびインデックス信号（通常ヌルには N、ゼロには Z、インデックスには I が用いられます）を通じてポジションを示します。

N 信号

N 信号を使用すると、エンコーダ・ポジション・カウンタ *X_ENC* をクリアするか、その受け渡し時にスナップショットを *ENC_LATCH* に取ることができます。N チャンネルを継続的にモニタし、N チャンネル・イベントが検出されたエンコーダの位置をクリア、あるいはラッチするトリガを発するには、フラグ *clr_cont* をセットします。あるいは、エンコーダの次の N チャンネル・イベントにのみ応答し、最初の N 信号イベント後のエンコーダ・ポジションのクリアまたはラッチを自動的にディスエーブルすることもできます（フラグ *clr_once*）。エンコーダがこの信号を供給するのは 1 回転ごとに一度であるため、この機能が必要となる可能性があります。

n_event をチェックするか、このイベントを示す割込み出力をマッピングします。その後、*n_event* をクリアします。

エンコーダによっては、A および B の極性の特定の設定によって N 信号を検証することが必要となります。これは、*ENCMODE* レジスタの *pol_A* と *pol_B* のフラグで制御できます。例えば、*pol_A* と *pol_B* の両方がセットされている場合、アクティブな N イベントが受け入れられるのは、A と B の両方のチャンネルがハイ極性になっている間のみです。

エンコーダ・ポジション *X_ENC* を次のアクティブな N イベントでクリアするには、*clr_enc_x* = 1 および *clr_once* = 1 または *clr_cont* = 1 に設定します。

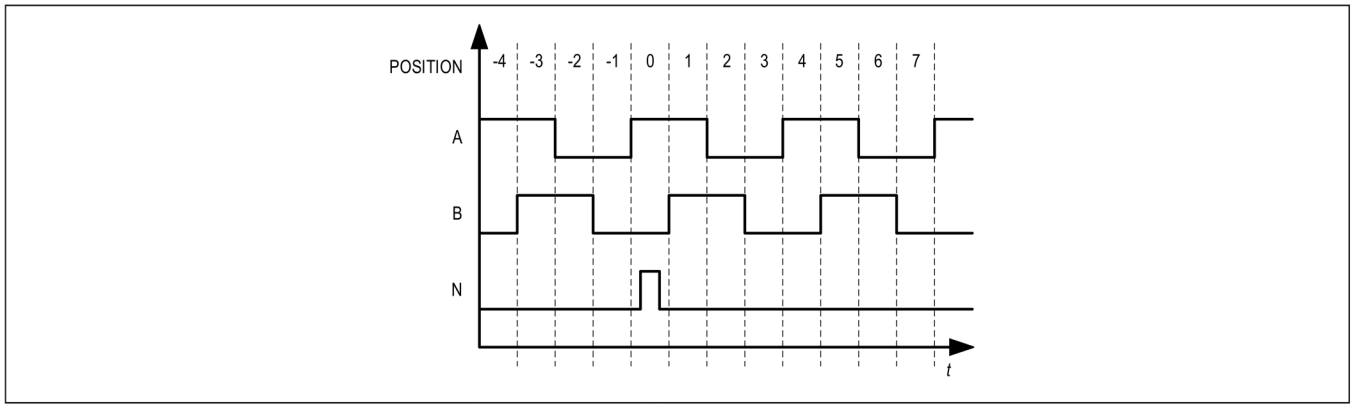


図 47. インクリメンタル・エンコーダの ABN 信号の概要

エンコーダ・カウンタ X_ENC

エンコーダ・カウンタ *X_ENC* は、電流エンコーダを読出し可能なポジションに保持します。信号 A、B、N の処理に関する様々なモードは、様々なタイプのエンコーダに見られるアクティブ・ローおよびアクティブ・ハイの信号を考慮します。

レジスタ ENC_STATUS

レジスタ *ENC_STATUS* は、N チャンネル信号でのエンコーダ・クリアのイベントに関するステータスを保持します。レジスタ *ENC_LATCH* は、N 信号イベント時の実際のエンコーダ・ポジションを常時保存します。

エンコーダ定数 ENC_CONST

エンコーダ定数（またはエンコーダ・ファクタ）*ENC_CONST* は、インクリメンタル・エンコーダの直交信号 AB の極性がそれぞれ変化したときに、エンコーダ・カウンタに加算、あるいはエンコーダ・カウンタから減算されます。エンコーダ定数 *ENC_CONST* は、符号付きの固定小数点数（16.16）を表し、モーターとエンコーダ間の一般的な調整を容易なものにします。2 進数モードの場合、下位 16 ビットは 0～65535 の数を表し、エンコーダ・イベントごとに 1/2^16 = 1/65536 のインクリメントを表現します。10 進数モードの場合、下位 16 ビットは 0～9999 の数を表し、エンコーダ・イベントごとに 1/10000 のインクリメントを表現します。インクリメンタル・エンコーダを備えたステッピング・モーターでは、固定小数点表示により、パラメータ化が非常に行いやすくなります。また、機械的なギアのかみ合わせを容易に考慮できます。*ENC_CONST* の符号を反転すると、モーターとエンコーダの方向に一致するようカウント方向を逆転できます。

エンコーダの方向を反転するには負のエンコーダ係数を用います。

- 2 進数モードでは、負のエンコーダ定数はエンコーダ定数の 2 の補数です。
  - (FACTOR.FRACTION) = (2^{16} − (FACTOR + 1)) × (2^{16} − FRACTION)
- 10 進数モードでは、負のエンコーダ定数は、次式を用いて同様に計算されます。
  - (FACTOR.DECIMALS) = (2^{16} − (FACTOR + 1)) × (10000 − DECIMALS)

エンコーダ定数の設定例：

Encoder factor 1.0	*ENC_CONST* = 0x0001.0x0000 = FACTOR.FRACTION
Encoder factor -1.0	*ENC_CONST* = 0xFFFF.0x0000. This is the two's complement of 0x00010000. It equals (2^{16} - (FACTOR + 1)) × (2^{16} - FRACTION)
Decimal mode encoder factor 25.6	00025.6000 = 0x0019.0x1770 = FACTOR.DECIMALS (DECIMALS = first 4 digits of fraction)
Decimal mode encoder factor -25.6	(2^{16} – (25 + 1)) × (10000 - 6000) = (2^{16} – 26) × (4000) = 0xFFE6.0x0FA0.

モーターの分解能に合うエンコーダ設定

モーター・パラメータに対するエンコーダの設定例：

- USC = 256 マイクロステップ
- FSC = 200 フルステップ・モーター
- 係数 = FSC × USC/エンコーダ分解能

表 25. 200 フルステップのモーターに対して 256 マイクロステップを使用するエンコーダの設定例

ENCODER RESOLUTION	REQUIRED ENCODER FACTOR	COMMENT
200	256	
360	142.2222 = 9320675.5555/2^{16} (binary) = 1422222.2222/10000 (decimal)	No exact match possible!
500	102.4 = 6710886.4/2^{16}(binary) = 1024000/10000 (decimal)	Exact match with decimal setting only.
1000	51.2	Exact match with decimal setting only.
1024	50	
4000	12.8	Exact match with decimal setting only.
4096	12.5	
16384	3.125	

計算例：

エンコーダ定数レジスタは、10 進数モードで 51.2 にプログラムされます。そのため次のように設定します。

ENC_CONST = 51 × 2^{16} + 0.2 × 10000 = 51.2000 = 0x0033.0x07D0

サイン波ルックアップ・テーブル

TMC2262 は、マイクロステップの電流波形を保存するためにプログラマブルなルックアップ・テーブルを備えています。デフォルトでは、このテーブルはサイン波で事前プログラムされており、ほとんどのステッピング・モーターにとって良い開始点となっています。テーブルをモーター固有の波形に再プログラミングすることで、特に低コスト・モーターでのマイクロステッピングを大幅に改善できます。ユーザに対する利点は次のとおりです。

- マイクロステッピング - 低コスト・モーターでステッピングの等距離性が改善。
- モーター - 滑らかかつ静かに動作。
- トルク - 機械的な共振が低減することでトルクが増加。
- 低周波数のモーター・ノイズ - モーターの製造公差にサイン波およびコサイン波のシフトを加えることで低減。

マイクロステップ・テーブル

プログラムするのに必要なメモリとデータ量を最小限に抑えるために、保存される波形は 1/4 のみです。内蔵のマイクロステップ・テーブルはマイクロステップ波形を 0º～90º に割り当てます。これが 360º まで対称的に拡張されます。テーブルの読出し時には、10 ビットのマイクロステップ・カウンタ *MSCNT* が、完全に拡張された波形テーブルをアドレス指定します。このテーブルは、エントリごとに 1 ビットを使用して、昇順に保存されています。そのため、波形の 1/4 を保存するのに必要なのは、わずか 256 ビット（*ofs00*～*ofs255*）のみです。これらのビットが 8 個の 32 ビット・レジスタに割り当てられます。各 *ofs* ビットは、テーブルで+1 ステップ進行する際に、勾配 Wx を加算するかどうか、すなわち Wx + 1 とするかどうかを制御します。Wx が 0 の場合、実際のマイクロステップ・ポジションでのテーブルのビットが 1 であれば、これは次のマイクロステップに進む際に「1 を追加」することを意味します。波形は 1 よりも大きな勾配を持つ場合もあるので、1/4 波形内で最大 4 つの柔軟なプログラマブル・セグメントを用いて、基本勾配 Wx を−1、0、1、または 2 にプログラムできます。したがって、負の勾配でも実現できます。4 つの勾配セグメントは、位置レジスタ *X1*～*X3* で制御されます。勾配セグメント 0 は、マイクロステップ・ポジション 0 から *X1*−1 までの範囲で、その基本勾配は *W0* で制御されます。また、セグメント 1 は *X1* から *X2*−1 の範囲で、基本勾配は *W1* で制御されます。他のセグメントも同様です。

波形を変更する場合、波形の 1/4 を全波形に拡張するときに滑らかで対称的なゼロ遷移が確保されるよう注意してください。実現可能な最大分解能を実現すると同時にヒステリシスベースのチョッパーがオフセットを追加できるヘッドルームを残しておくために、変更の結果得られる波形の最大振幅が−248～248 の範囲となるよう調整する必要があります。

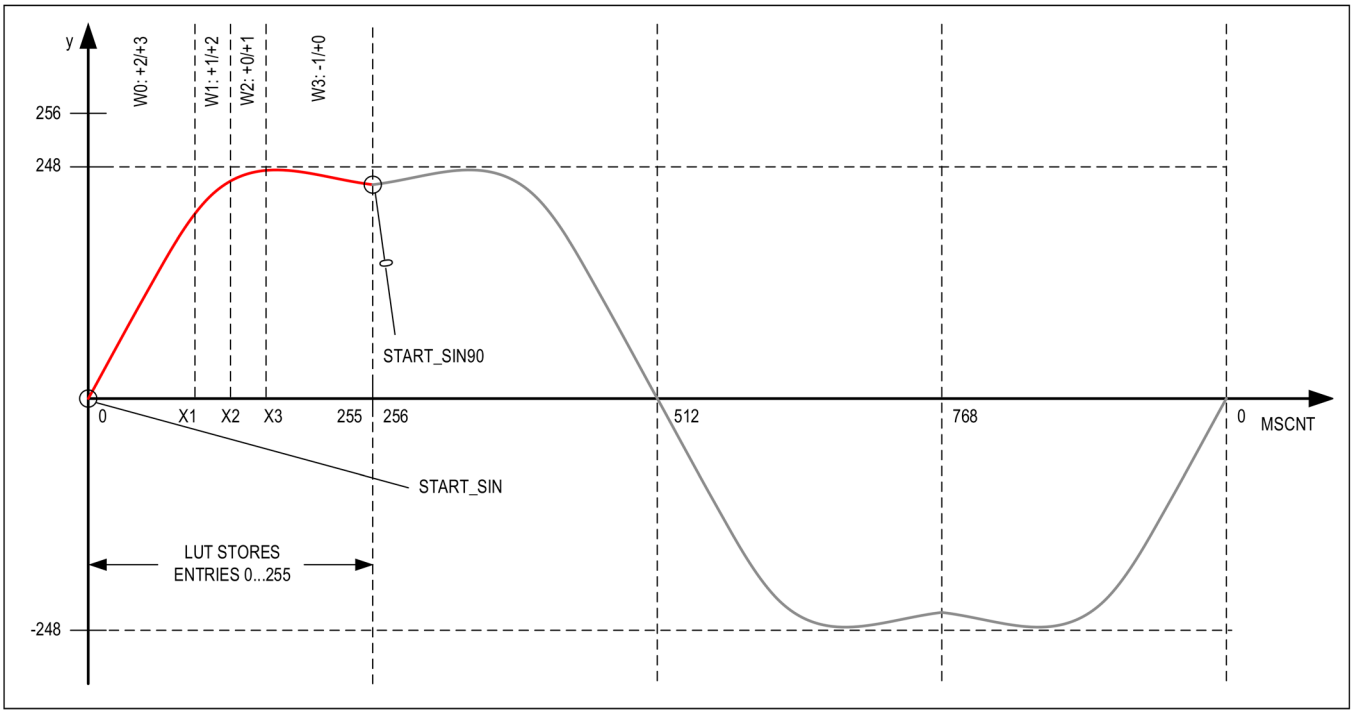


図 48. LUT プログラミングの例

マイクロステップ・シーケンサはテーブル内を進行する際に、モーター・コイルの実際の電流値をマイクロステップごとに計算し、それらをレジスタ *CUR_A* および *CUR_B* に保存します。ただし、インクリメント・コーディングは、特にマイクロステップ・テーブルが変更される場合には、絶対的な初期化を必要とします。そのため、*CUR_A* と *CUR_B* は、*MSCNT* がゼロと交差するときは必ず、*START_SIN90* および *START_SIN* から初期化されます。

注： StealthChop+での波形

StealthChop+を用いる場合、サイン波振幅は常に実効値に安定化されます。これは、振幅レギュレーションが、強制的に *CUR_A* と *CUR_B* の一定の二乗和を取ろうとするためですそのため、適応された波形は、三角波などのまったく異なる波形を用いるのではなく、サイン波を変調する（対称に引き伸ばしや圧縮を行う）ことによってのみ生成する必要があります。

位相シフトとモーターのマッチング

2 つのレジスタがサイン波とコサイン波の開始値を制御します。

- マイクロステップ 0 でのサイン波の開始値は 0 である必要がありますが、波形を変形する特別な場合には 1 または 2 にオフセットさせることもできます。これは開始点レジスタ *START_SIN* を用いて定めることができます。
- 2 つ目のモーター・コイル用のコサイン波の開始点は *START_SIN90* にプログラムする必要があります。単調性の波形の場合、これは波の振幅であり、1/4 周期内のインクリメント量の和に相当します。それと共に、レジスタは、2 相モーターの 90º の位相シフト用に得られたテーブル・エントリを保存します。
- ただし、モーターの許容誤差に合わせるため、コサイン波の 0 点は左または右にシフトできます。特殊な場合では、−127～+127 の範囲のマイクロステップ・オフセット（レジスタ *OFFSET_SIN90*）を追加することで、位相シフトを 90º（256 マイクロステップ）から 45º～135º の範囲の任意の値に変更できます。しかし、モーターの許容誤差に対する補正は、ハイブリッド・ステッピング・モーターの場合は数ステップ、低品質モーターの場合は最大数十ステップの、中程度の適応にとどめることが必要です。必要な補正オフセットは、高分解能エンコーダを用いて個々のモーターを注意深く測定することにより、見出すことができます。この場合、新たなゼロ遷移点に合うよう *START_SIN90* を調整する必要があります。

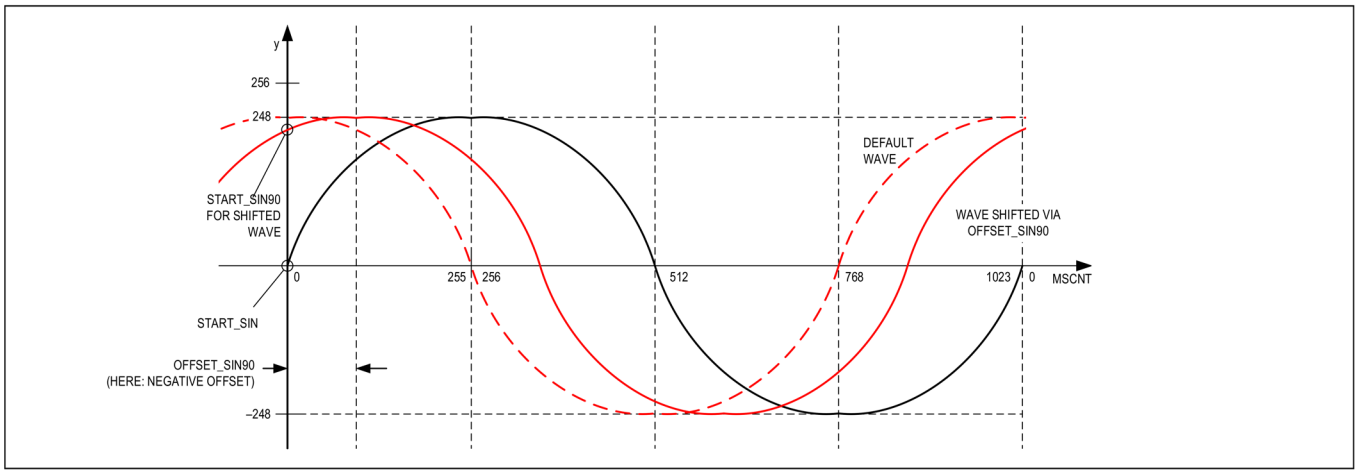


図 49. OFFSET_SIN90 を使用したコサイン波のシフト

デフォルト・テーブルは、一般的なハイブリッド・ステッピング・モーターのほとんどに対し最もよく適合します。これは、独自のテーブルを実現するための良い基礎となります。リセット後のデフォルト・マイクロステップ・テーブルの初期化例を以下に示します。

MSLUT[0] = %10101010101010101011010101010100 = 0xAAAAB554

MSLUT[1] = %01001010100101010101010010101010 = 0x4A9554AA

MSLUT[2] = %00100100010010010010100100101001 = 0x24492929

MSLUT[3] = %00010000000100000100001000100010 = 0x10104222

MSLUT[4] = %11111011111111111111111111111111 = 0xFBFFFFFF

MSLUT[5] = %10110101101110110111011101111101 = 0xB5BB777D

MSLUT[6] = %01001001001010010101010101010110 = 0x49295556

MSLUT[7] = %00000000010000000100001000100010 = 0x00404222

MSLUTSEL = 0xFFFF8056:

X1 = 128, X2 = 255, X3 = 255

W3 = %01, W2 = %01, W1 = %01, W0 = %10

MSLUTSTART = 0x00F70000: START_SIN_0 = 0, START_SIN90 = 247

モーターの位相シフトを最適化するには、高分解能エンコーダを用いてモーターの回転をチェックし、*OFFSET_SIN90* をインクリメントまたはデクリメントして非対称な動作を 360º の周期以内に縮小します。最適な一致が得られるまでこれを繰り返し、逆方向の動作時にダブル・チェックを行うか、平均値を取得します。

波形に不連続点が生じるのを防ぐため、コサイン波の開始値 *START_SIN90* を確実に正しく入力してください。デフォルト波形を用いる場合、オフセットが−10～+9 のときは *START_SIN90* = 247 が必要です。−17 または+17 までは *START_SIN90* = 246 が必要です。*START_SIN* は常に 0 です。異なる波形の場合、これらの値は、*OFFSET_SIN90* を 0 にとどめたまま、オフセット・ポジションに移動し *CUR_A* をチェックすることにより、決定できます。

リセット、ディスエーブル／停止、パワー・ダウン

緊急停止

ドライバは、負のアクティブ・イネーブル・ピン DRV_ENN を備えており、すべてのパワーMOSFET を確実にスイッチ・オフします。これにより、モーターがフリーホイール状態になります。更に、これは、ソフトウェアと連携しない緊急停止が必要な場合には常に、安全なハードウェア機能となります。アプリケーションによっては、ドライバがアクティブなホールド電流の状態または受動ブレーキング・モードの状態になることが要求されます。これを行うには、ピン ENCA がステップ・ディスエーブル機能として作用するようプログラミングします。*GCONF.stop_enable* をセットするとこのオプションをアクティブにできます。ENCA がハイにプルアップされ、それがハイにとどまる限り、I_{HOLD}、*IHOLD_DELAY*、および StealthChop+の停止オプションを通じた設定に従い、モーターは突然に停止し、パワー・ダウン状態になります。

外部リセットおよびスリープ・モード

リセットおよびスリープ・モードは SLEEPN ピンで制御されます。

持続時間が 30µs を超える短パルスを SLEEPN に印加すると、チップ・リセットが生じます（*GSTAT.register_reset* フラグでも認識可能）。

30µs 未満の非常に短いパルスはフィルタ除去され、動作には影響しません。

SLEEPN が GND に維持されている場合は、IC は低消費電力のスタンバイ状態（スリープ・モード）になります。内部電源はすべてスイッチ・オフされます

リセットとスタンバイのどちらの場合も、すべての内部レジスタの値と設定値はクリアされてそのデフォルト値に設定され、電源ブリッジはオフになります。

パワーアップ後、あるいはスリープ・モードやリセット状態からの解放後、レジスタは再設定する必要があります。

誤った設定でモーターが動作するのを防ぐため、再設定の最後のステップとしてドライバを再イネーブルする必要があります。

モーターが大電流で駆動されたり、高速で回転したりしている間にドライバをスリープ・モードにしたりリセットしたりしないでください。モーターからフィードバックされるエネルギーによりチップが損傷する可能性があります。

使用しない場合は、V_{CC_IO} に接続します。

診断と保護

TMC2262 では、地絡保護、Vs への短絡（過電流）保護、低電圧検出など、フルセットの診断機能および保護機能が利用できます。開放負荷状態の検出では、モーター・コイルの接続が切断されているかどうかをテストできます。詳細については *DRV_STATUS* レジスタの表を参照してください。

ステータス・フラグの他、TMC2262 では、チップ温度の測定および読出しや、コイル抵抗の変化を評価することによるモーター位相用の巻線温度へのフィードバックが可能です（*R_COIL* の測定を参照）。ドライバの温度測定により、負荷を低減するためにモーターを減速したり、モーターおよびドライバを冷却できるよう動作を中断したりといった、極端な温度条件に対する予防動作が可能になります。

システムの信頼性と回路全体の保護能力を向上するため、TMC2262 には過電圧コンパレータとトリガ出力 OV（DO0 ピンおよび DO1 ピンにマッピングするためのオプション機能）が内蔵され、電源電圧の過剰な上昇があった場合に外部スイッチを制御します。

更に、内部パラメータをアナログ信号としてオシロスコープ上に可視化するために、DO0 および DO1 を、内蔵の *リアルタイム・オン・チップ・スコープ・インターフェース* の出力として設定できます。

診断出力

TMC2262 は、DO0 と DO1 の 2 つの設定可能診断出力を備えています。

どちらのピンも、CPU 割込み、加熱や過電圧などの緊急事態の検出、外部イベントのトリガなどに利用できる、様々なステータス情報ビットをマッピングできます。これらのオプションは、レジスタ *DO_CONF* を用いて設定できます。

オープン・ドレイン（アクティブ・ロー）出力信号（デフォルトは *doosen0x_nOE_PP* = 0）またはアクティブ・ハイのプッシュプル出力信号（*do0x_nOE_PP* = 1）を選択できます。負のアクティブ・オープン・ドレイン出力を使用する場合は、複数のドライバ出力信号を論理的に OR 接続できます。4.7kΩ～100kΩ の範囲の外部プルアップ抵抗が必要です。リセット条件を確実に決定するには、SPI により *reset* フラグをモニタするか、いずれかのレジスタを読み出してチップがパワーアップされていることを確認します。

オプション・モードでは、DO0 および DO1 の出力アナログ信号は、デバッグやパラメータ調整に役立ちます。

- DO0 ピンの診断出力は、RT-OCSI の DAC チャンネル 0 と機能的に共有するピンです。
- DO1 ピンの診断出力は、RT-OCSI の DAC チャンネル 1 と機能的に共有するピンです。

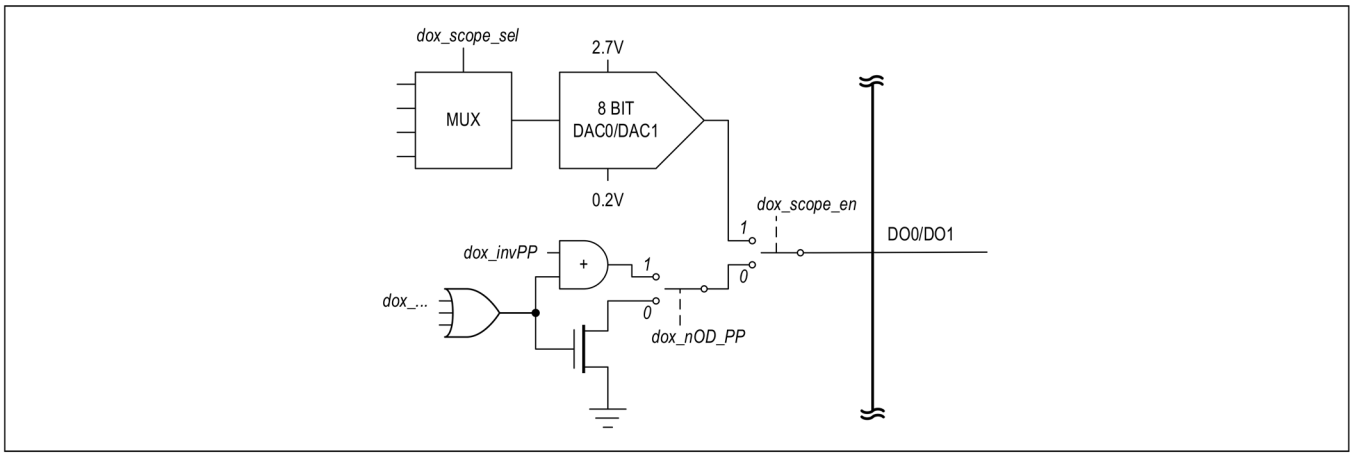


図 50. DO0 および DO1 の出力の回路図

リアルタイム・オン・チップ・スコープ・インターフェース（RT-OCSI）

リアルタイム・オン・チップ・スコープ・インターフェース（RT-OCSI）は、2 つのオシロスコープ・プローブ用のテスト・ポイントを提供するため、アプリケーションの調整やデバッグを容易に行うことができ、SPI にアクセスして PC 上でリアルタイム・データを可視化する必要がありません。TMC2262 は、診断出力 DO0 および DO1 にマッピングできる 2 つの独立した内蔵 DAC 出力を用いて、オシロスコープ上にリアルタイム・データを可視化できます。

それと共に、RT-OCSI は、モーター電流のスケーリングと StallGuard の結果など、2 つの動作パラメータを一度に直接モニタリングすることによって、StealthChop+、StallGuard+、CoolStep+の調整や、StallGuard2 および CoolStep の調整もサポートします。これらのパラメータは、4 チャンネル・オシロスコープを用い、物理的にアクセス可能な電圧やモーター電流などの電流と並行して、モニタできます。

どちらの DAC チャンネルも、様々な内部パラメータを柔軟に割り当てることができます。オシロスコープおよび DAC の分解能や、ステッピング・モーター・アプリケーションでの電気的なノイズ・フロアによって加わる制限を克服するため、12 ビットを超えるビット範囲の選択がレジスタに対して可能です。

診断出力 DO0 と DO1 の一方または両方で DAC 出力を可能にするには、*DO_SCOPE_CONF* レジスタ（0x03）の *do0_scope_en* および *do1_scope_en* をセットします。

デジタル・ゼロ・スケールおよびフル・スケールへの高速アナログ・セトリングをサポートするため、DAC 出力は、0.2V の内部ゼロ・オフセットを付加することによって、0.2V～2.7V の電圧範囲において 2.5V の振幅を供給します。したがって、符号付きの値の場合、0 の値は 128 のオフセットを付加することにより、(2.7V + 0.2V)/2 = 1.45V にマッピングされます。

DAC 出力をテストしたり、外部アナログ電圧を供給するために出力を用いたりするのは、USER_VALUE のインデックス 0x1C を選択することで可能です。

各 DAC 出力では、*DO_SCOPE_CONF* レジスタ（0x03）の *DO0_SCOPE_SEL* および *DO1_SCOPE_SEL* 設定パラメータを用い、そのインデックス番号を通じて以下の内部パラメータの 1 つを選択できます。

表 26. リアルタイム・オン・チップ・スコープ・インターフェース（RT-OCSI）の DAC 設定オプション

インデックス	信号	説明	使用事例
0x0	*ADC_I_A*[11..5] + 128	電流コイル A（符号とオフセット付き）	コイル A/B の電流をモニタ。電流 PI レギュレータをチェック。
0x1	*ADC_I_B*[11..5] + 128		
0x2	*RCOIL_A*[11..4]	コイル A の抵抗がドライバによって自動的に決定されます。	更新時刻、低速度動作、コイル抵抗測定 A/B の概略値をチェック。
0x3	*RCOIL_B*[11..4]		
0x4	*UL_A*[11..5] + 128	コイル A/B（符号とオフセット付き）について計算したインダクタンス電圧（電流は 90° 遅れ）。	-
0x5	*UL_B*[11..5] + 128		
0x6	*COOLSTEP_LOAD_RESERVE*	目標負荷余裕の実際の値	電流に依存する負荷余裕をモニタ。CoolStep を追跡するために実際の *SGP_RAW* の値と比較。
0x7	*ADC_TEMPERATURE*	温度 ADC の実際の値	IC 温度をモニタ。
0x8	*ANGLE_MEAS*[9..2]	コイル電流から計算した電気的角度	計算した電流角度をモニタし、*MSCNT* と比較して角度レギュレータをチェック。
0x9	*SGP_RAW*[7..0]	0～255 の範囲にクリップされた StallGuard+の生の値（SGP_FILT_EN は不適用）。	モーターの負荷および負荷測定の安定性をモニタ。*COOLSTEP_LOAD_RESERVE* と比較。
0xA	*ANGLE_CORR_CALC*	角度レギュレータの結果	角度レギュレータの是正措置をモニタ。
0xB	*AMPL_MEAS*[11..4]	ADC_I_A および ADC_I_B から計算された実効的なモーター電流。100%の電流が、目標値の½フル・スケール（2048）に相当します。	電流レギュレータおよび電流振幅の応答をチェック。V/div の 1/2 のスケーリングを用いて *CS_ACTUAL* と共に可視化。
0xC	*PWM_CALC*[11..4]	電流レギュレータの結果	PWM のデューティサイクルをチェックし、駆動電圧のヘッドルームをチェック。
0xD	*CS_ACTUAL*	*IRUN*、*IHOLD*、CoolStep から生じる実際の電流スケール。	負荷に対する CoolStep の電流プロファイルをチェック。トルクのヘッドルームをチェック。
0xE	*UBEMF_ABS*[11..4]	バック EMF 電圧の大きさ	バック EMF ベースの StallGuard+信号の予想品質を評価するために大きさをチェック。
0xF	*SG_RESULT*[9..2]	StallGuard2/StallGuard+の実際の値（符号なし、0～1023 の範囲）	StallGuard の設定を調整するためにモニタ。

0x10	*SGP_RESULT*[9..2]+128	StallGuard+の実際の値（符号付き、−512～+511の範囲）	フィルタ処理した信号とフィルタ処理しない信号の関係をチェックし、ジェネレータ領域の範囲をモニタ。
0x11 0x12	*CUR_A*[8..1]+128 *CUR_B*[8..1]+128	全振幅での実際のマイクロステップ波形テーブル（コイル電圧）。モーターのインダクタンスおよびBEMFによるCUR_xによって与えられる、コイル電流（ADC_I_xの値）の電圧に対する遅延。	マイクロステップ・テーブルの波形をチェック。
0x13 0x14	DAC_A DAC_B	SpreadCycleでの電流レギュレーション用の内部DACの入力（チョッパー動作を示す）。DACは正の値（オン・フェーズ）と負の値（高速減衰フェーズ）を受け取ります。	SpreadCycle の目標電流をモニタ（ヒステリシス・チョッパーを示す）。
0x15	*ANGLE_ERROR*	*MSCNT_SNPSHT* と *ANGLE_MEAS* の角度差	角度レギュレータのレギュレーション誤差をモニタ。（**注：***MSCNT_SNPSHT* の更新ごとに約 12µs の *ANGLE_MEAS* の時間遅延による信号ジッタが発生します）
0x16	*MSCNT*[9..2]	マイクロステップ・カウンタ（*MRES* の定義に従いステップごとにインクリメント）。	マイクロステップ・カウンタのポジションをモニタ。
0x17	*MSCNT_SNPSHT*	角度計算用に 32 チョッパー・サイクルごとにサンプリングされた *MSCNT_OFFSET*。	パラメータと波形をサンプリングしたポジションとの関係をモニタ。ポジションは鋸歯波として与えられます。
0x18	*MSCNT_OFFSET*	*MSCNT* に適合するよう補間されたマイクロステップ・カウンタ。ただしマイクロステップ補間を使用（角度レギュレータによって計算されたオフセットは不使用）。	低いマイクロステップ分解能を用いた場合の電流波形におけるパラメータとポジションの関係をモニタ。ポジションは鋸歯波として与えられます。（*ANGLE_MEAS* に対するリファレンス）
0x19 0x1A 0x1B	*TSTEP_VELOCITY*[20...13] *TSTEP_VELOCITY*[18...11] *TSTEP_VELOCITY*[16...9]	低いレンジや高いレンジをモニタするために様々なスケーリングで *TSTEP* から計算したモーター速度（符号なし）。	パラメータと速度の関係をモニタ。選択されたビット範囲を超えると、値はフル・スケールから 0 にオーバーフローします。
0x1C	USER_VALUE	DO0 には *DO_CONF*[7..0]、DO1 には *DO_CONF*[20..13] で定義された、出力 DAC 電圧（符号なしの値）。	DAC インターフェースのテスト、ユーザ定義の値の出力。
0x1D 0x1E 0x1F	BEMF voltage A/B[13…7] + 128 BEMF voltage A/B[11…5] + 128 BEMF voltage A/B[9…3] + 128	様々なスケーリングでの、内部BEMFの計算結果の符号付きベクトル。範囲の限界にクリップされることはありません。 DO0 ではコイル A の BEMF、DO1 ではコイル B の BEMF	計算した SGP BEMF 電圧をモニタ。電流波形と比較。最低感度のレンジ（0x1D）から開始し、モニタリングに十分な信号を与えるフィッティング・レンジを選択。

RT-OCSI の使用例

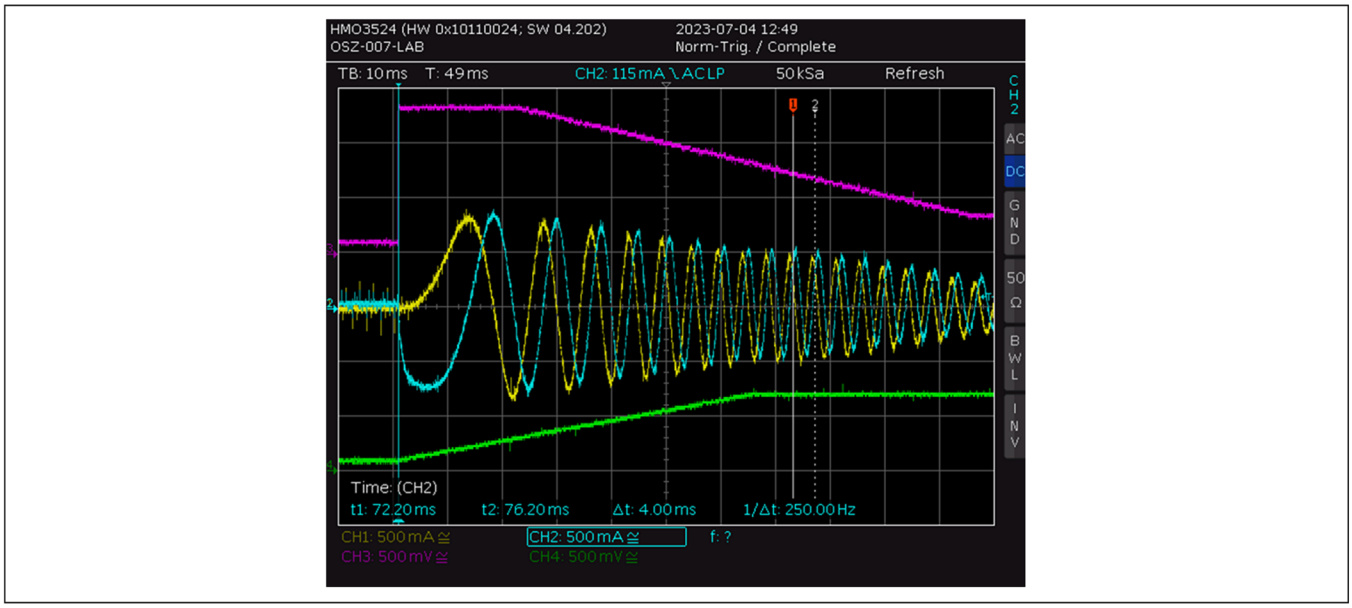


図 51. RT-OCSI を使用した CoolStep+のモニタリング

図 51 の例は、RT-OCSI を用いて CoolStep+をモニタする方法を示しています。

黄色と青色のトレース：電流をクランプして測定したモーター電流。CoolStep+をイネーブルしてモーターを停止状態から始動（全 *IRUN* 電流を使用）し、モーションの開始後約 22ms で CoolStep+の設定による制御に従って電流を減少させる様子を示しています。

緑色のトレース：加速を示す、0x1A の *TSTEP_VELOCITY* モーター速度。

紫色のトレース：CoolStep+により制御される電流低減を示す、0xD の *CS_ACTUAL* 電流スケール。

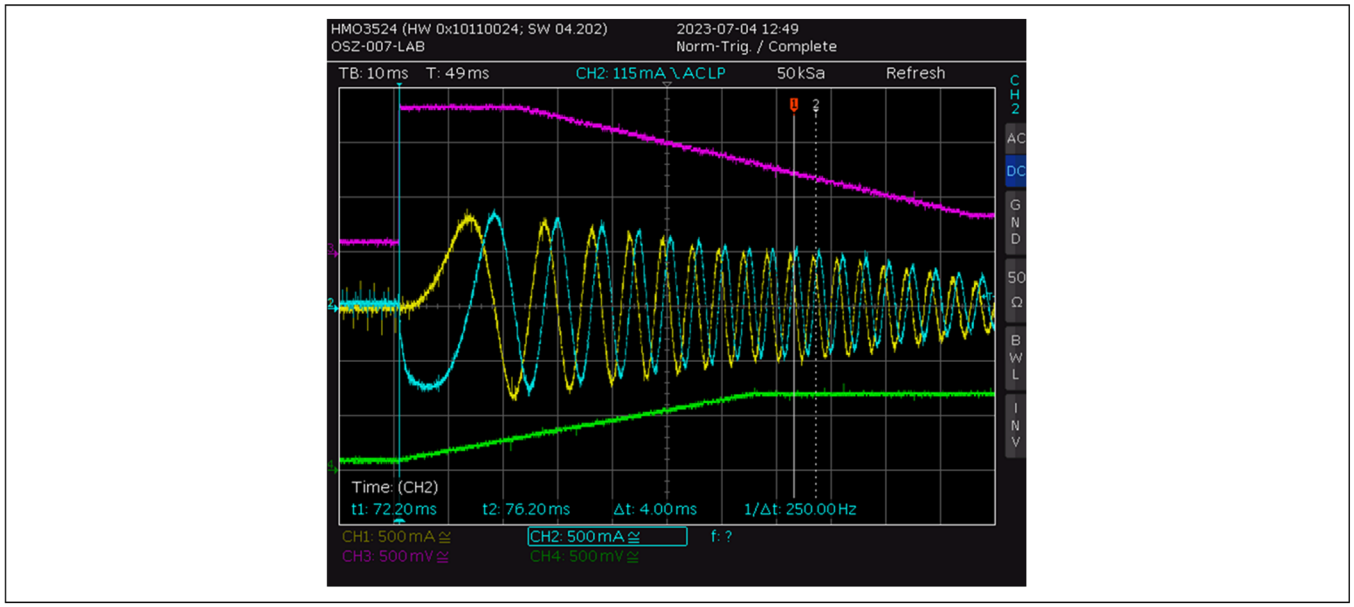


図 52. RT-OCSI を使用した電流 PI レギュレータのモニタリング

図 52 の例は、RT-OCSI を用いて電流 PI レギュレータをモニタする方法を示しています。

黄色と青色のトレース：電流をクランプして測定したモーター電流。モーターを停止状態から始動させる様子を示しています。

緑色のトレース：電流をランプアップするようデューティサイクルを調整したことを示す、0xC の *PWM_CALC* 電流レギュレータ出力。

紫色のトレース：実効モーター電流を 90%まで 4ms でランプアップしたことを示す、両コイルで測定し計算された 0xB の *AMPL_MEAS* モーター電流振幅。

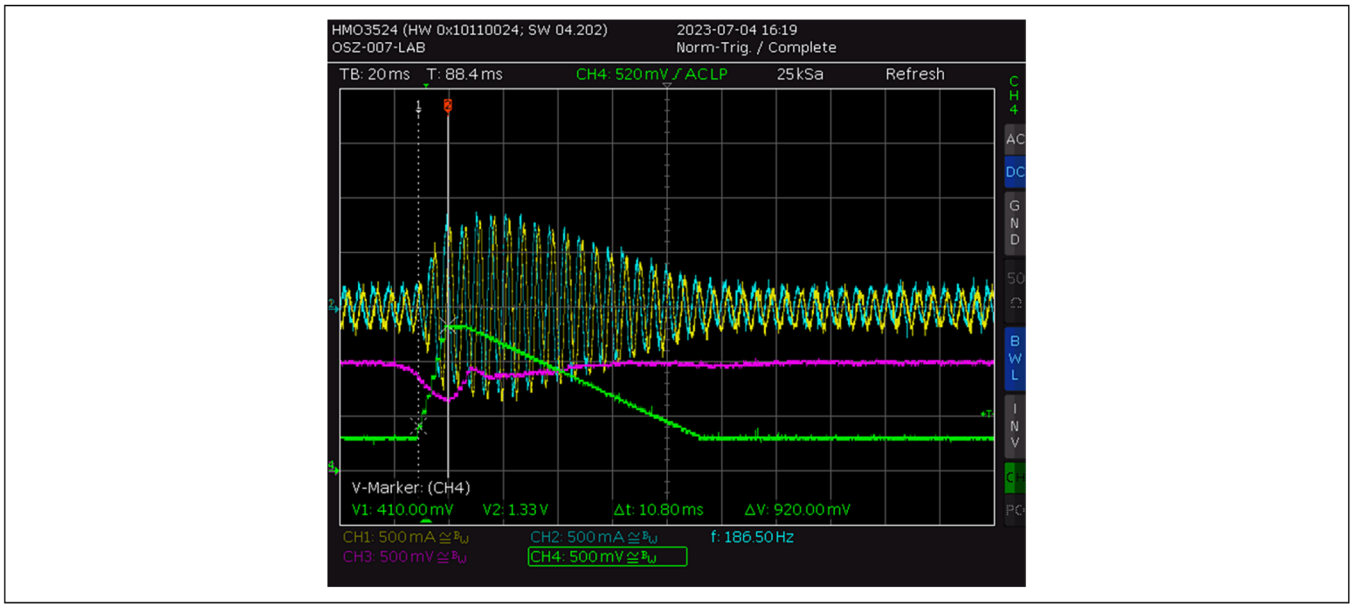


図 53. RT-OCSI を使用した CoolStep+ PI レギュレータのモニタリング

図 53 の例は、RT-OCSI を用いて CoolStep+ PI レギュレータをモニタする方法を示しています。

黄色と青色のトレース：電流をクランプして測定したモーター電流。t = 20ms で始まる軸に負荷ピークがある、電流減少状態でのモーター動作の様子を示しています。

緑色のトレース：負荷衝撃と電流デクリメント勾配に応答して電流レギュレータに加えられる電流スケーリングを示す、0xD の *CS_ACTUAL*。

紫色のトレース：無負荷状態または無負荷に近い状態から負荷衝撃のピーク時に 0（DAC 範囲の 1/2）に変化する、StallGuard+の負荷測定を示す 0x10 の *SGP_RESULT*。

過電流保護

過電流保護（OCP）は、レール（電源電圧とグラウンド）への短絡と出力（OA1、OA2、OB1、OB2）間の短絡からデバイスを保護します。

OCP スレッショルドは、選択したフル・スケール電流範囲に依存します。また、それぞれのスレッショルド値については電気的特性の表を参照してください。

ローサイド MOSFET での短絡検出は電流レンジ（*CURRENT_RANGE*）を併用して自動的に制御されます。したがって、選択したモーター・タイプやピーク電流に合わせてより高感度に反応します。この点を考慮して、モーターに適する最低の *CURRENT_RANGE* を選択すると、例えば、誤配線や絶縁不良の場合に、最適な保護が提供されます。これを用いると、モーターが大きなコイル電流に駆動されている間に *CURRENT_RANGE* を減少した場合、過電流を検出できます。そのため、モーターがイネーブルされ目標の *CURRENT_RANGE* のピーク電流より大きな電流で駆動されている間も、*CURRENT_RANGE* を変更すべきではありません。いずれにせよ、*CURRENT_RANGE* の変更には、StealthChop+コイル抵抗の再測定が必要です。

デグリッチ時間（ブランキング時間）より長い間にわたって出力電流が OCP スレッショルドを上回る場合、OCP イベントが検出されます。

OCP イベントが検出されると、影響を受ける H ブリッジが直ちにディスエーブルされます。

ESD イベント時の誤トリガを防ぐために、連続するチョッパー・サイクルで 3 回、短絡保護の再試行を行ってから、フォルト・フラグ（*DRV_STATUS* レジスタの *s2ga*、*s2gb*、*s2vsa*、*s2vsb*）がセットされ、ブリッジが連続的にディスエーブルされます。

デバイスはアクティブのままであり、設定やステータスの読出しが可能です。ブリッジにフォルトがあっても、ランプ・ジェネレータまたはシーケンサのアクションを引き起こすことはありません。また、結果として生じるすべてのアクションはソフトウェアが対応します。

電源ブリッジを再イネーブルするには、DRV_ENN ピンを用いてこれをディスエーブルして再イネーブルするか、*CHOPCONF* の *TOFF* を 0 にセットし、次いで *TOFF* > 0 にセットしてブリッジを再イネーブルします。

過熱保護およびサーマル・シャットダウン

TMC2262 は過熱保護機能を内蔵しています。

過度な加熱の場合にチップに熱的損傷が生じないようにするには、各パワーMOSFET の近くでダイ温度をモニタします。いずれかのセンサーが測定する温度が 165ºC（代表値）を超過した場合は常に、フォルト指示フラグ（*DRV_STATUS* の *ot*）がセットされ、ドライバはジャンクション温度が約 145ºC（代表値）未満になるまでスリーステートとなります。その後、ドライバは再度イネーブルされます。

過熱の前に予防措置ができるよう、TMC2262 は、ADC ベースで設定可能な過熱事前警告レベルをサポートしています。これは、パラメータ *OVERTEMPPREWARNING_VTH* を用いてレジスタ *OTW_OV_VTH* で設定できます。ADC は、ダイの中心でドライバ段から一定の距離を保ってチップ温度を検出し、各ドライバ段の個々の加熱は検知しません。そのため、事前警告スレッショルドは、シャットダウン温度から少なくとも 20ºC～40ºC 異なる、アプリケーション固有の温度に設定する必要があります。

したがって、事前警告制限値は、必ずしも TMC2262 がこの制限に達するまではシャットダウンしないことを意味するわけではありません。制限がこれより低い温度に設定されていたとしても、熱は IC に不均等に分布する可能性があるためです。主として、熱はモーター・ドライバ段で発生し、個々のコイル電流および極性に応じて 8 個のオン・チップ・パワーMOSFET に分布します。ドライバ MOSFET が過熱するような重大な状況のほとんどは、短絡保護機能によって回避できます。多くのアプリケーションでは、過熱事前警告は異常な動作状況にあることを示すもので、これを用いて、ユーザ警告を発したり、モーター電流低減などの電力低減措置を開始したりできます。サーマル・シャットダウンは単なる緊急措置であり、シャットダウン・レベルまで温度が上昇することは、設計によって回避しなくてはなりません。

温度測定

TMC2262 は内部チップ温度およびモーター温度を測定する機能を備えています。

これらの診断機能は、アプリケーションにおいて、チップまたは PCB の温度およびモーター温度の時間変化をモニタし、システムの堅牢性を向上したり、予防的メンテナンスのための追加情報を取得したりするのに有用です。

チップ温度の測定

過熱の事前警告や過熱フラグの他、チップ温度自体が、*ADC_TEMP* レジスタの *ADC_TEMP* パラメータを用いて決定できます。

ADC_TEMP と *OVERTEMPPREWARNING_VTH* の ºC 単位の最終温度は、次の式で計算できます。

$$ADC_TEMP = \frac{(\mathrm{TEMP} + 264.6)}{1.042}$$

$$TEMP[^\circ C] = ADC_TEMP \times 1.042 - 264.6$$

注：温度測定値は、IC の中央での温度を示します。モーター電流や PCB サイズに応じて、電力段の温度はこれより大幅に高くなる可能性があります。実際の温度差を決定するには、サーマル・イメージング・カメラを用いて確認してください。

モーター温度の測定

StealthChop+は、その機能を果たすために、モーター・コイルの抵抗を必要とするため、モーター・コイル抵抗の比較的正確な測定が可能です。コイル抵抗のその 25º の初期値からの変化をモニタすることにより、モーター・コイルの温度を判定できます。銅製コイルの場合、熱係数は 1 ケルビンあたり 0.39%です。しかし、コイル抵抗の測定には、ドライバの出力電力段の抵抗も含まれ、これもドライバの温度と共に増加します。モーターのコイル抵抗がドライバ MOSFET の抵抗よりも高い限り、この部品から生じるのは小さい誤差です。非常に低抵抗のモーターでは、補正係数を測定によって決める必要があります。

$$T_{Motor} = T_{REF} + \frac{R_{COIL} - R_{REF}}{R_{REF} \times \alpha}$$

α：コイルの材料の温度係数で、銅の場合は 0.39%/K。

T_{REF}、R_{REF}：リファレンス温度とリファレンス温度におけるモーター・コイルの抵抗。

過電圧保護と OV 出力

ステッピング・モーター・アプリケーションでは、特にモーターが高速状態から短時間で減速する場合、モーターがストールする場合、あるいは、外力によって急速に回転させられる場合に、大きな過電圧が発生します。こうした状況では、機械的回転によるエネルギーや、モーターの磁界に蓄えられたエネルギーが、ドライバの出力段を通じて電源レールにフィードバックされます。

代表的な NEMA17 やそれより大型のモーター、および十分な慣性重量を持つ小型のモーターでは、フィードバックされるエネルギーは相当な量となるため、電源コンデンサや回路でのエネルギー消費では、電源をその制限値以内に保持するには不十分です。低エネルギー・フィードバックのシステムではツェナー・ダイオードを過電圧保護に用いることができますが、その内部抵抗は比較的高く、消費電力能力は限られています。

エネルギー・フィードバックの高いシステムでドライバや接続回路を保護するために、TMC2262 では、過電圧の検出および保護のための機構を備えています。TMC2262 は、ADC を用いて電源電圧を永続的に測定し、その結果をプログラマブルなスレッショルドと比較します。ADC の結果がスレッショルドに達するかこれを超えると、常に、内部過電圧フラグ DRV_STAT.ov がセットされます。過電圧信号は、一方または両方の DO 出力にマッピングできる他、エンコーダの N ピン（ENCN/OVN）にマッピングして、エンコーダの N チャンネルを使用しない環境において専用の出力を供給できます。ADC は約 2.45kHz で更新を行うので、OV 信号の場合 1ms 未満の更新レートになります。したがって、過電圧ダンピング抵抗を駆動する外部トランジスタは、約 1kHz でチョッピングを行い電源を制限値以内に保ちます。

所定のアプリケーションに対する電源電圧の上限レベルは、レジスタ OTW_OV_VTH のパラメータ OVERVOLTAGE_VTH を用いて設定できます。すべての許容誤差を考慮に入れて、通常の電源電圧に対し十分なヘッドルームを確保し、過電圧ディテクタが通常の状況でトリガすることのないようにしてください。ダンピング回路を破壊する可能性があるためです。

電源電圧の実際の ADC 値は、レジスタ ADC_VSUPPLY_AIN のパラメータ ADC_VSUPPLY を使って読み出せます。

ADC の値を V_S に変換したり、あるいはその逆を行ったりするには、次の式を用います。

$$V_S = ADC_VSUPPLY \times 140.9\text{mV}$$

$$OVERVOLTAGE_VTH = V_{S(MAX)}/140.9\text{mV} + 1$$

OV 出力ピンは、過電圧モニタの実際の状態を示します。

ADC_VSUPPLY が OVERVOLTAGE_VTH 以上になるとすぐに、またその状態である限り、OV 出力信号はアクティブになります（ENCN/OVN がローに駆動）。

エネルギーを電力抵抗にダンプする過電圧保護トランジスタを駆動するには、外部ゲート・ドライバを用いる必要があります。最初にこの機能を、選択した出力にマッピングする必要があるため、過電圧保護機構は、設定完了後に外部信号によって明示的にイネーブルする必要があります。スレッショルド電圧でモーターからフィードバックされる最大電流に適合するよう、電力抵抗を選択してください。代表的なモーター・フィードバック電流は、定格のモーター・コイル電流よりはるかに小さい値です。これを用いて、ダンプ抵抗のリファレンス値を次のように計算できます。

$$R_{DUMP} = V_{S(MAX)}/I_{COILNOM}$$

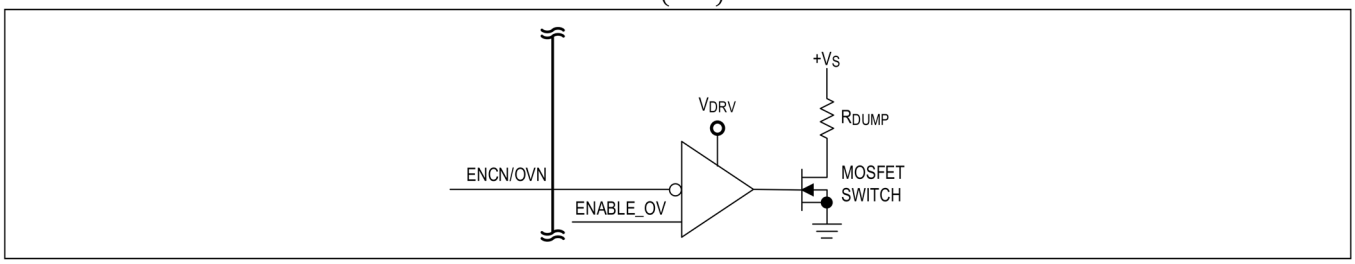


図 54. ブレーキ・チョッパー回路の例

短絡保護（地絡および V_S への短絡）

TMC2262 の電力段は、ハイサイド MOSFET を流れる電流を追加測定することで、回路の短絡状態に対し保護されます。大半の短絡状態は、システムのグラウンドに接続された導電素子に接触するというような、モーター・ケーブルの絶縁不良が原因です。短絡検出は、3 度再試行してからモーターをスイッチ・オフすることで、ESD 放電などによる誤トリガから保護されます。

短絡状態が確実に検出されると、対応するドライバ・ブリッジがスイッチ・オフされ、*s2ga* フラグまたは *s2gb* フラグがセットされます。モーターを再始動するには、ドライバをディスエーブルした後再度イネーブルするというユーザの介入が必要です。短絡イベントは様々であり、また、外部コンポーネントの複合的なネットワークが影響を受ける可能性があるため、地絡保護では、必ずしも起こり得るすべての短絡イベントからシステムと電力段を保護できるとは限らない点に注意してください。そのため、基本的に短絡を回避する必要があります。

フル・スケール電流設定に応じて、様々な過電流保護スレッショルドでローサイドの短絡保護がトリガします。詳細な値については電気的特性の表を参照してください。

開放負荷診断

断線したケーブルは、コネクタが確実に差し込まれていない場合などのように、よく見られるシステム故障の原因です。TMC2262 は、目的のモーター・コイル電流に達したかどうかをチェックすることで、開放負荷条件を検出します。したがって、低電圧条件、高モーター速度設定、あるいは短絡および過熱条件も、開放負荷フラグをトリガする可能性があります。モーター停止時には、結果的にコイルの電流がゼロであるため、開放負荷を測定できません。

断線したコイル接続を確実に検出するには、SpreadCyle で動作させ、低速あるいは通常のモーター速度動作のみを用い、選択したマイクロステップ分解能の最低 4 倍（4 フルステップ）の単一方向動作の後に開放負荷フラグをチェックします。ただし、*ola* フラグと *olb* フラグは、単に情報を伝えるだけの性質しかなく、ドライバの何らかの動作の原因になるわけではありません。

低電圧ロックアウト保護

TMC2262 は、V_S、V_{CC_IO}、チャージ・ポンプに対する UVLO 保護機能を備えています。

V_S に対する UVLO 条件は、4.15V 未満でトリガされます。

V_{CC_IO} に対する UVLO 条件は 2.2V 未満でトリガされます。

チャージ・ポンプに対する UVLO 条件は、コンデンサ値が不適切であることなどによるチャージ・ポンプのエラー条件が生じた場合にトリガされます。

V_S に UVLO 条件が発生したことは、レジスタ GSTAT のフラグ *vm_uvlo* で読み出せます。これは、電源電圧のブラウンアウトに続いてセットされます。このフラグは「書き込んでクリア」フラグです。これをクリアするにはアクティブに 1 を書き込む必要があります。

V_S または V_{CC_IO} の UVLO 時には、IC との通信は不可能で、すべてのレジスタはリセットされ、ドライバはディスエーブルされます。DO0 ピンは、アクティブ・ロー（オープン・ドレイン）で外部 CPU にこの条件を通知します。

ESD 保護

チップには、ピンごとに ESD 保護機能が内蔵されています。

TMC2262 のモーター位相出力ピンは、正電圧電源（V_S ピン）に 1µF 以上のバイパス・コンデンサを用いたアプリケーションで、最大 8kV の人体モデル（HBM）に対して保護されています。これはモーターのホットプラグに対する保護ではありません。

クロック発振器とクロック入力

内部 PLL ブロック

TMC2262 は、PLL 周波数逓倍器を内蔵しており、内部調整された発振器を基にした動作、またはより正確な発振器から引き出される外部クロック・ソースを基にした動作が可能です。外部クロック・ソースは、1MHz～32MHz の範囲における 1MHz 単位の任意の値が可能です。ユーザ定義の分周係数（*CLOCK_DIVIDER*）を用いた周波数分周器は、選択されたクロック・ソースから 1MHz の内部リファレンス周波数を引き出します。この 1MHz のクロックを基に、PLL は、固定された 16MHz を内部コントローラのロジックおよび計算用に供給し、より高い 80MHz の周波数を StealthChop+の PWM の動作用に供給します。

PLL は、IC の動作前に正しく設定し稼働させる必要があります。PLL がディスエーブルされている場合やフォルト・モードになっている場合、設定レジスタにはアクセスできますが、ドライバおよびフル制御ロジック・ブロックの動作は停止し、ドライバはディスエーブルされます。クロック信号に何らかの異常がある場合や、PLL 設定が不適切な場合は、IC がこのディスエーブル・モードになり、エラー・フラグ *clk_is_stuck* および *clk_loss* がセットされます。

PLL を設定するには、常に 2 つのレジスタ書込みアクセスが必要です。つまり、最初にパラメータの設定と確定を行い、次に 2 回目のアクセスでエラー・フラグをクリアします。PLL のスタートアップを可能にするには、2 つの書込みの間に遅延が必要です。遅延は、待機命令（最低でも $t_{PLLSTRT}$）によって、または *commit* をポーリングして 0 にリセットすることによって、時間設定されます。この遅延後の 2 回目のアクセスは、エラー・フラグ *clk_is_stuck* および *clk_loss* をリセットすることであり、それによって、今度は制御ロジックとドライバ・ブロックがイネーブルされます。

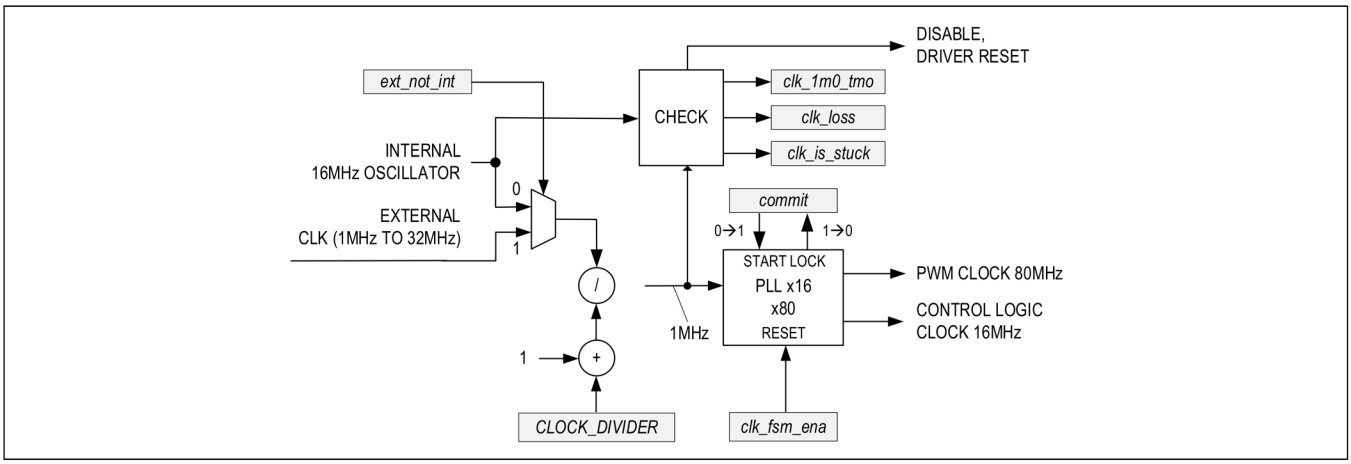


図 55. PLL のブロック図

注意：モーターを動作させる前に PLL を設定し起動してください。動作させるには、PLL のエラー・フラグをクリアする必要があります。

例

16MHz の内部発振器を用いて動作するよう PLL を設定します。

1. クロックが 16MHz の場合、1MHz のリファレンス・クロックを発生するために 1/16 の分周器が必要です。

   CLOCK_DIVIDER を 15（= 16 − 1）に設定します。内部クロックを用いる場合は、*ext_not_int* を 0 にままにします。
   clk_sys_sel および *clk_fsm_ena* は常に 1 に設定する必要があります。*clk_1M0_tmo* を 1 に設定してこのビットをクリアします。

   commit フラグをセットして PLL レジスタに以下を書き込みます。
   PLL = 0x01ED。
2. PLL レジスタをポーリングして、*commit* が再度クリアされるまで待機します。
3. これで、PLL の動作準備ができました。*clk_is_stuck* および *clk_loss* をクリアすることにより、デジタル部分のリセットを解放します。これを行うには、両方のビットをセットし、同じワードを PLL レジスタに書き込みます。ただし、*commit* はクリアされたままにします。
   PLL = 0x61EC
4. PLL 設定エラーが生じた場合は、まず PLL をディスエーブルしてこれをリセットし（*clk_fsm_ena* = 0 にセット）、ステップ 1 から再開します。

表 27. PLL 制御レジスタ PLL のパラメータ

パラメータ	説明	設定	コメント
commit	**書込み**：このフラグは、パラメータ（*ext_not_int*、*clk_sys_sel* *CLOCK_DIVIDER*）をPLLブロックに転送するかどうかを決定します。新しい値を確定すると、*clk_loss*および*clk_is_stuck*が1にセットされます。ICを動作させるには： - フラグ*commit*が0でリード・バックされるまで待機します。 - *clk_loss*および*clk_is_stuck*をクリアする場合、*commit*に0を書き込みます。 **読出し**：PLL設定の変更を確定した後、PLLの準備が整うとフラグは0にリセットされます。	0	PLL は影響を受けません。フラグがクリアされるだけです。
		1	設定を PLL に転送し PLL を再始動 - PLL の準備が整うとフラグは自動的に 0 にリセットされます。
ext_not_int	PLL クロック信号用ソースの選択。より高い精度の速度およびランプ・タイミングが必要な場合、あるいは使用可能な場合は常に、外部クロックを用いることを推奨します。正確な局部発振器が入手できない場合には、内部クロックの使用が有益です。PLL を始動する前に、適切なクロック・ソースを選択する必要があります。	0	内部発振器（16MHz）を選択します。*CLOCK_DIVIDER* を 15 に設定します。
		1	クロック入力をクロック・ソースとして選択します。
clk_sys_sel	通常動作では、このビットを 1 に設定します。	0	PLL クロックは不使用、ドライバはオフ。
		1	PLL をクロック・ソースとして選択します。
clk_fsm_ena	PLL をイネーブルします。*ext_not_int* と *CLOCK_DIVIDER* によって選択された適切なクロック・ソースと組み合わせた場合にのみ、このビットを 1 に設定します。 不適切なクロック周波数状態（*clk_1M0_tmo set*）から回復するには、0 にセットしてから 1 にセットし直す必要があります。	0	PLL ブロックがリセット状態。
		1	PLL の通常動作。
CLOCK_DIVIDER	PLL クロック分周器。外部クロックまたは内部クロックの周波数に一致するようにこのレジスタを設定します（MHz 単位の周波数-1）。内部発振器の場合は常に 15 に設定します。誤った設定や不正確なクロック入力で動作させないでください。余裕のない設定を行うと *clk_1M0_tmo* がセットされる可能性があるためです。	0…	1MHz の外部クロック～
		14	15MHz の外部クロック
		15	内部クロックまたは 16MHz の外部クロック
		16…	17MHz～
		31	32MHz の外部クロック
clk_1M0_tmo	ステータス・ビット。クリアするには「1」を書き込みます。PLLの起動フェーズの間に誤った周波数が1MHzのクロックから引き出されています（このクロックが内部クロックと比較されます）。動作時にはチェックされません。動作時には、*clk_loss*が外部クロックからの逸脱をチェックします。 誤った値がPLL内にある間、このエラー・フラグはクリアできません。このフラグをクリアする前に以下を実行します。 - *clk_fsm_ena*を0に設定してPLLをディスエーブルします。 - （外部）クロックおよびCLOCK_DIVIDER設定が正しいことを確認し、更に、 - *commit* および *clk_fsm_ena* を 1 に設定し直して値を確定します。そうなって初めて、すべてのエラー・フラグをクリアできます。	0	PLL クロックが存在し、内部クロックと比較しておおむね適正な値になっています。
		1	外部クロック周波数が範囲外になっているか、*CLOCK_DIVIDER* が誤っています。
clk_loss	ステータス・ビット。クリアするには「1」を書き込みます（*commit* が 0 でなくてはなりません）。 外部クロック信号が失われているか、誤ったクロック周波数になっています。このビットは、PLL 設定に何らかの変更が行われた場合にもセットされます。	0	通常動作
		1	PLL がエラー状態になっています。クロック信号が失われています。
clk_is_stuck	ステータス・ビット。クリアするには「1」を書き込みます（*commit*が0でなくてはなりません）。 外部クロック信号がスタックしています。また、*clk_loss* と組み合わせてセットされます。このビットは、PLL 設定に何らかの変更が行われた場合にもセットされます。	0	通常動作
		1	PLL がエラー状態になっています。クロック信号の極性が固定されています。

内部クロックの使用

内部クロック発振器を使用するには、CLK 入力ピンを GND または V_{CC_IO} に接続します。外部クロックに切り替えるには、PLL レジスタを用いてクロック・ソースを動的に設定する必要があります。

内部クロックは、16MHz（代表値）の周波数で動作し、PWM の生成やチョッパーの時間制御を行う場合にも、また、複数のモーター間の同期動作が不要な、1 軸を使用する多くのアプリケーションに対しても、十分安定しています。

外部クロックの使用

周波数が既知で確定している場合には、水晶で安定化させたクロックを用いることを推奨します。このクロックは、水晶発振器から直接供給することも、マイクロコントローラによって複数のドライバに分配することもできます。複数の IC にクロック信号を分配するには低めの周波数を用いると有益です。ボード上で高い周波数を分配するには、クロック・ツリーにインピーダンス・マッチしたライン・ドライバを組み合わせた、注意深いレイアウトが必要になるためです。

外部クロック周波数の範囲は 1MHz～32MHz とすることができますが、内部 PLL のパラメータ化を行うために既知であることが必要です。内部 PLL を用いると、この周波数は、ランプ生成用には 16MHz のクロック、StealthChop+の PWM ユニット用には 80MHz のクロックに変換されます。

クロック信号の必要な最小および最大のデューティサイクルは、電気的特性の表に定義されています。特に、32MHz に近いクロック周波数では、クロックのデューティサイクル条件を満たす必要があり、また、エッジがクリーンでなくてはなりません。

どの場合においても、クロック・ソースが完全な CMOS 出力ロジック・レベルを供給しており、反射が発生していないことを確認してください。反射があるとロジック・スレッショルドが不安定になったり、複数の遷移を発生したりします。高いクロック周波数を用いる場合は急峻な勾配が必要です。勾配が急峻であることで、隣接する配線パターンとのカップリングによる擾乱のリスクが低減します。

外部クロックを印加した後は、PLL レジスタへのレジスタ・アクセスにより、外部クロック入力 CLK がクロック・ソースとしてアクティブに選択されなくてはなりません（*ext_not_int* をセット）。また、それに対応する PLL 係数を設定する必要があります。クロックに異常が生じた場合、IC はエラー・モードに移行してリセットされるため、PLL 設定ステップを繰り返す必要があります。*reset* フラグおよび PLL レジスタ・フラグを読み出して、エラーをチェックします。他のファミリ製品とは異なり、内部クロックへの自動フォールバックはできません。そのため、モーターの動作中にクロック入力を停止することは推奨されません。モーターが制御不能になると、電源への大きな電力フィードバックあるいは制御されない電力フィードバックが生じる可能性があるためです。

レジスタ IOIN のビット *ext_clk* を読み出すと、現在どのクロック・ソースを使用しているかがフィードバックされます（1 = 外部クロック）。

クイック設定ガイド

このガイドは、レジスタの初期設定を行い、更にドライバ調整に関する最小限の測定と決定を行うための、実用的なツールを提供することを意図したものです。ここではすべての先進的機能とオプションをカバーしているわけではなく、モーターをスムーズに動作させるための基本的な機能セットに重点を置いています。モーターが動作するようになった後で、その他の機能やより高度な機能の詳細を確認してください。1 つのモーター・コイルに電流プローブを配置することは、最適な設定を見出す上で良い助けとなります。

PLL の初期化ガイド

制御ロジックおよびドライバ・ブロックの動作より前に、PLL を設定し起動する必要があります。

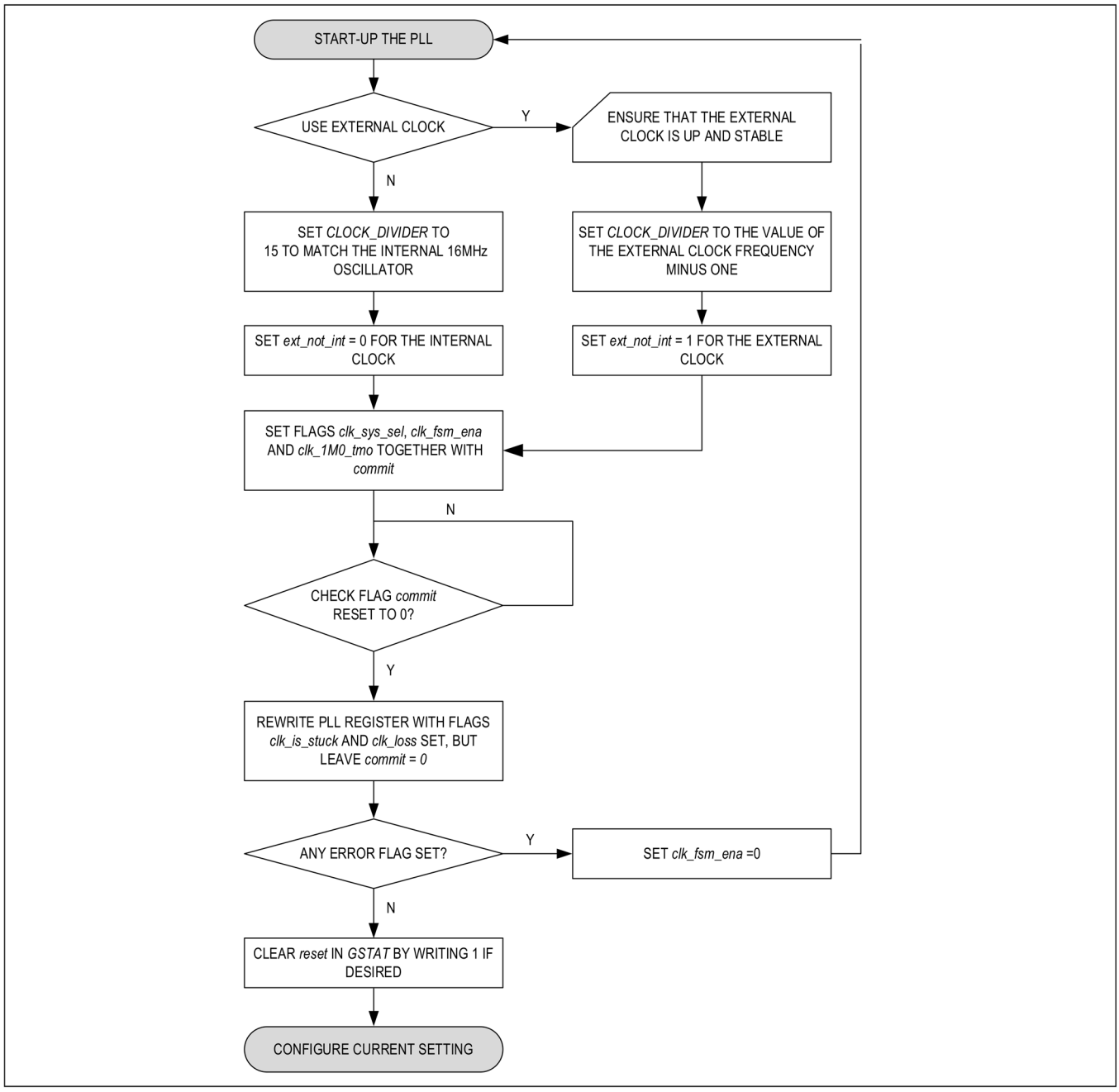


図 56. PLL を起動するためのクイック設定ガイド

電流設定ガイド

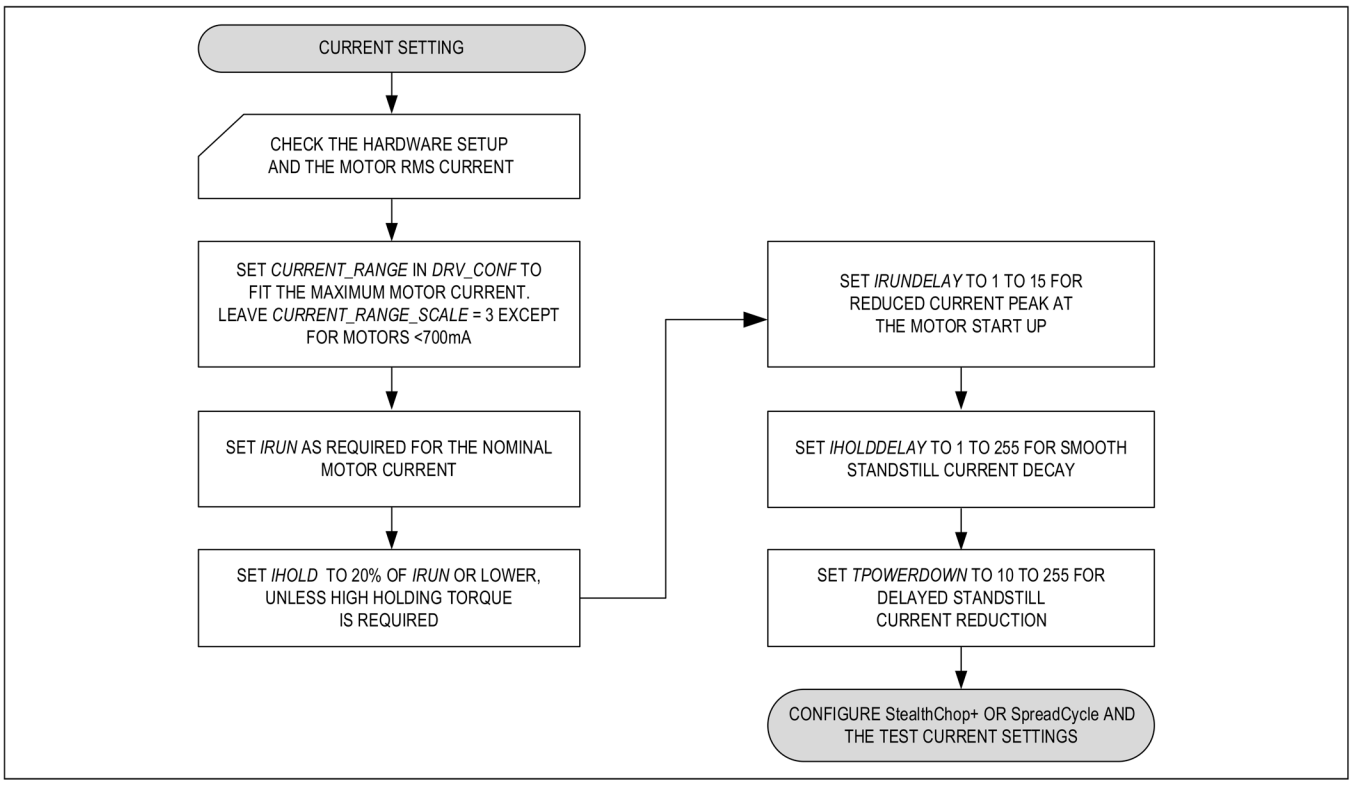


図 57. 電流設定のためのクイック設定ガイド

StealthChop+の設定

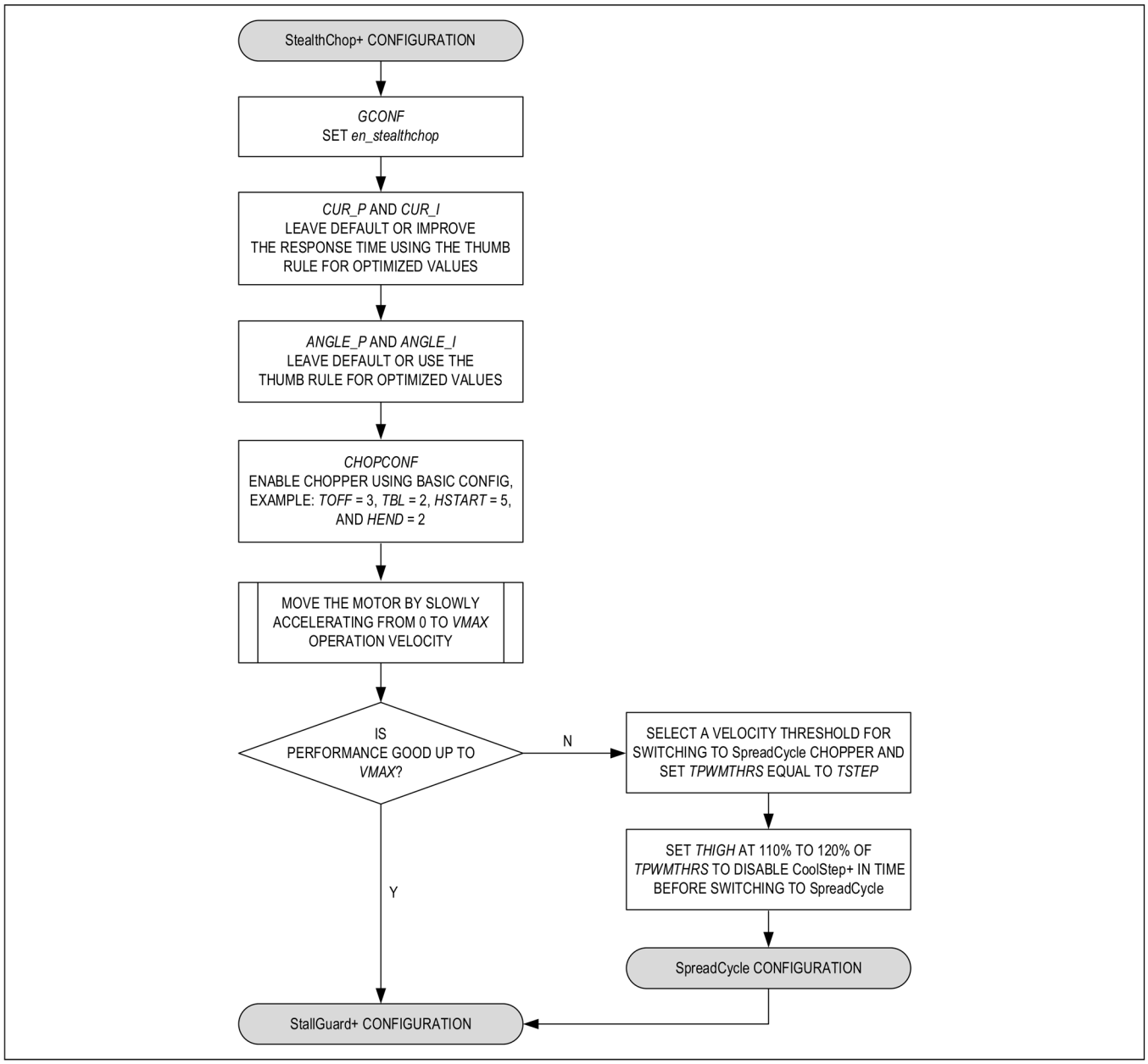


図 58. StealthChop+を動作させるためのクイック設定ガイド

StealthChop+と組み合わせた StallGuard+

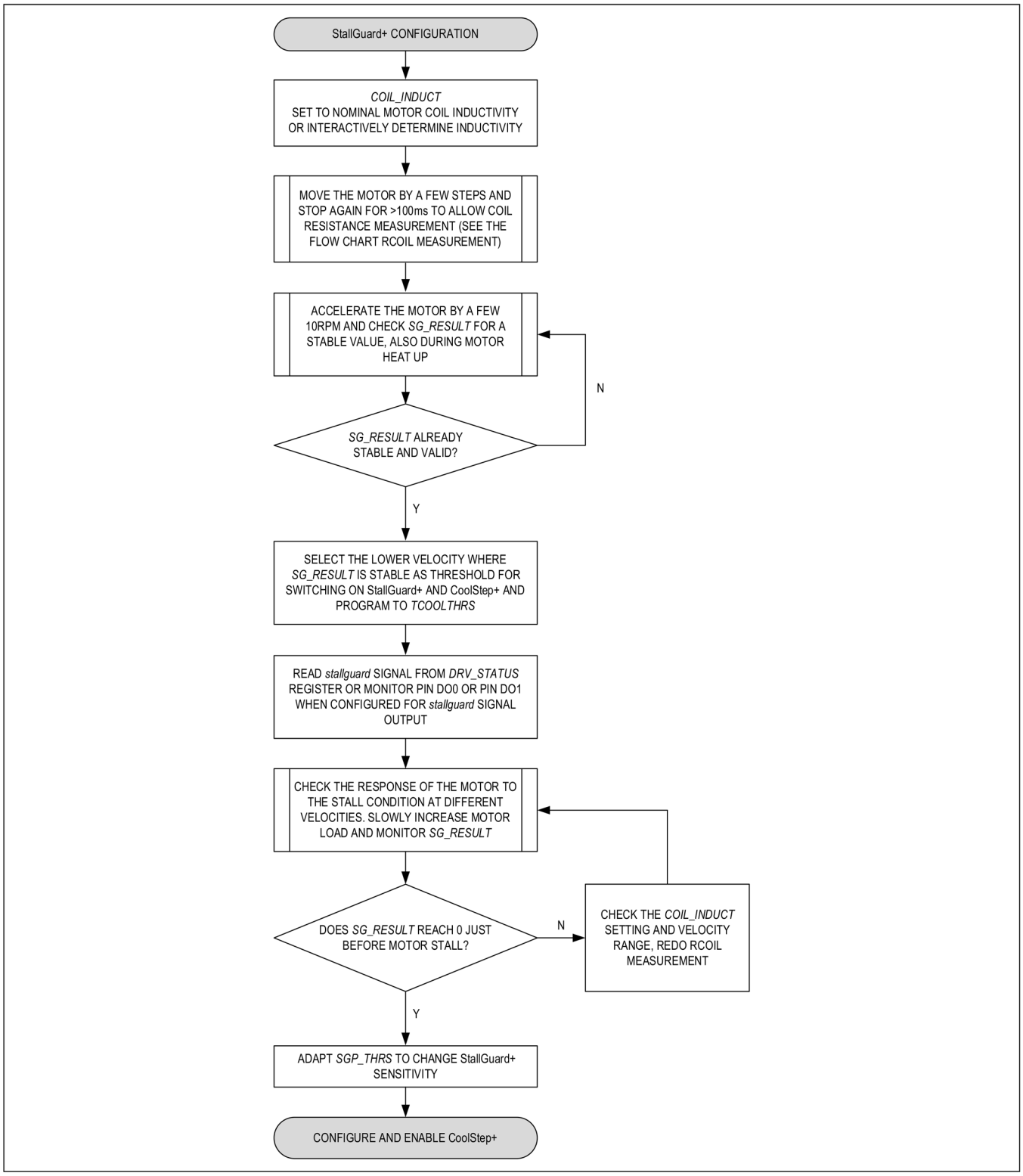


図 59. StallGuard+の（StealthChop+の設定に従う）基本的な設定のためのクイック設定ガイド

StealthChop+と組み合わせた CoolStep+

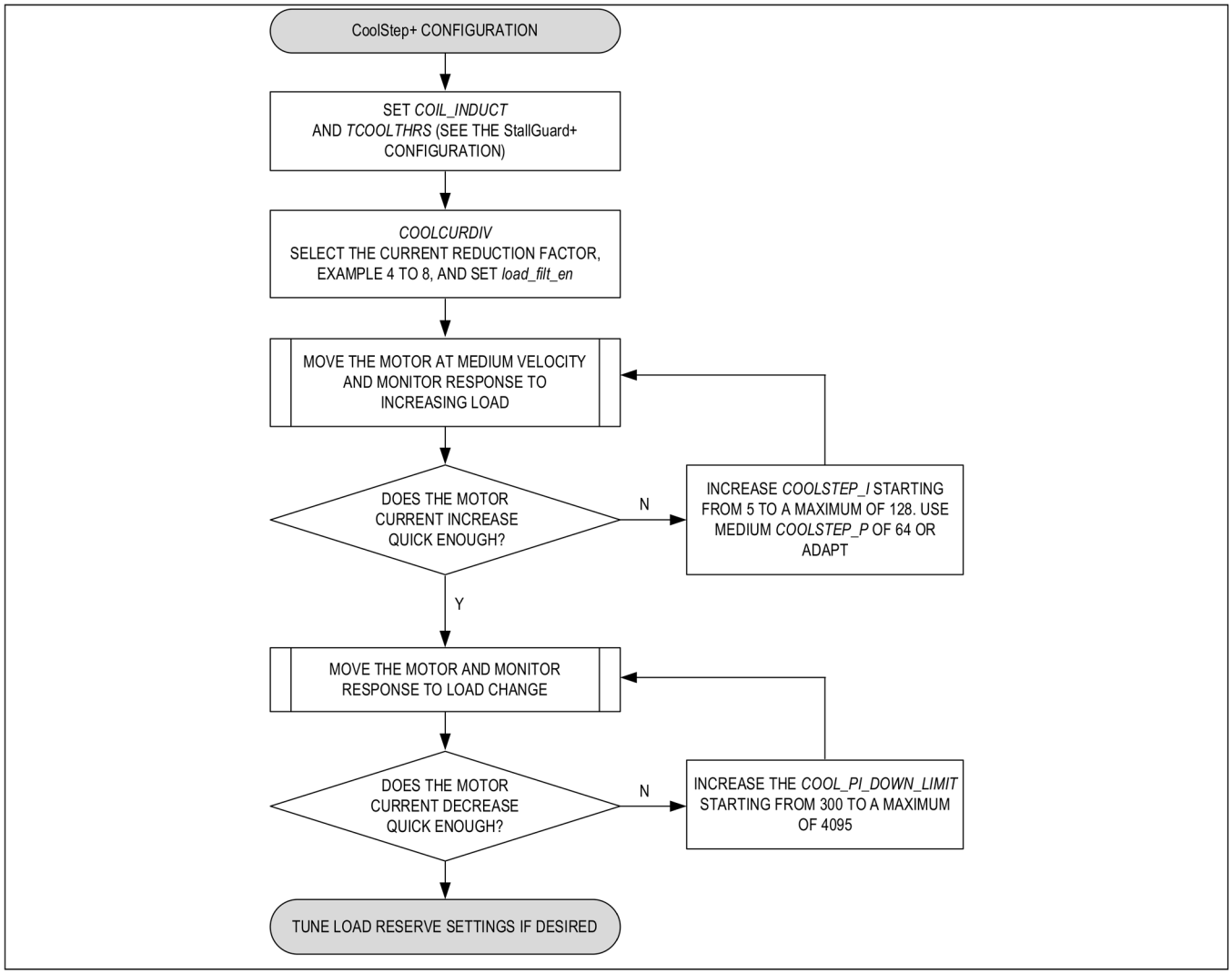


図 60. CoolStep+の（StealthChop+の設定に従う）設定のためのクイック設定ガイド

STEALTHCHOP+と組み合わせた µDCSTEP 動作

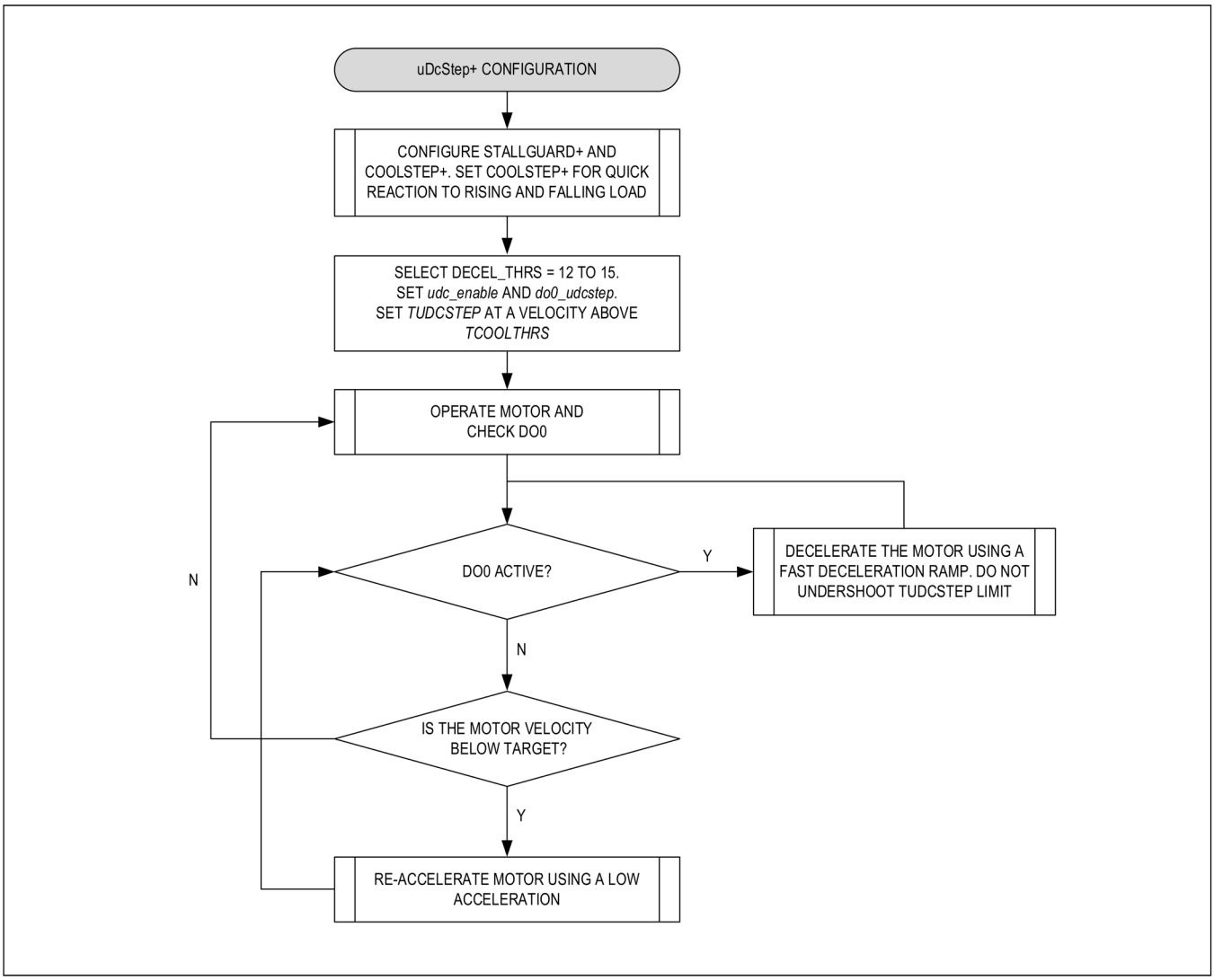


図 61. µDcStep の（StealthChop+および CoolStep+の設定に従う）クイック設定および動作ガイド

SpreadCycle の設定

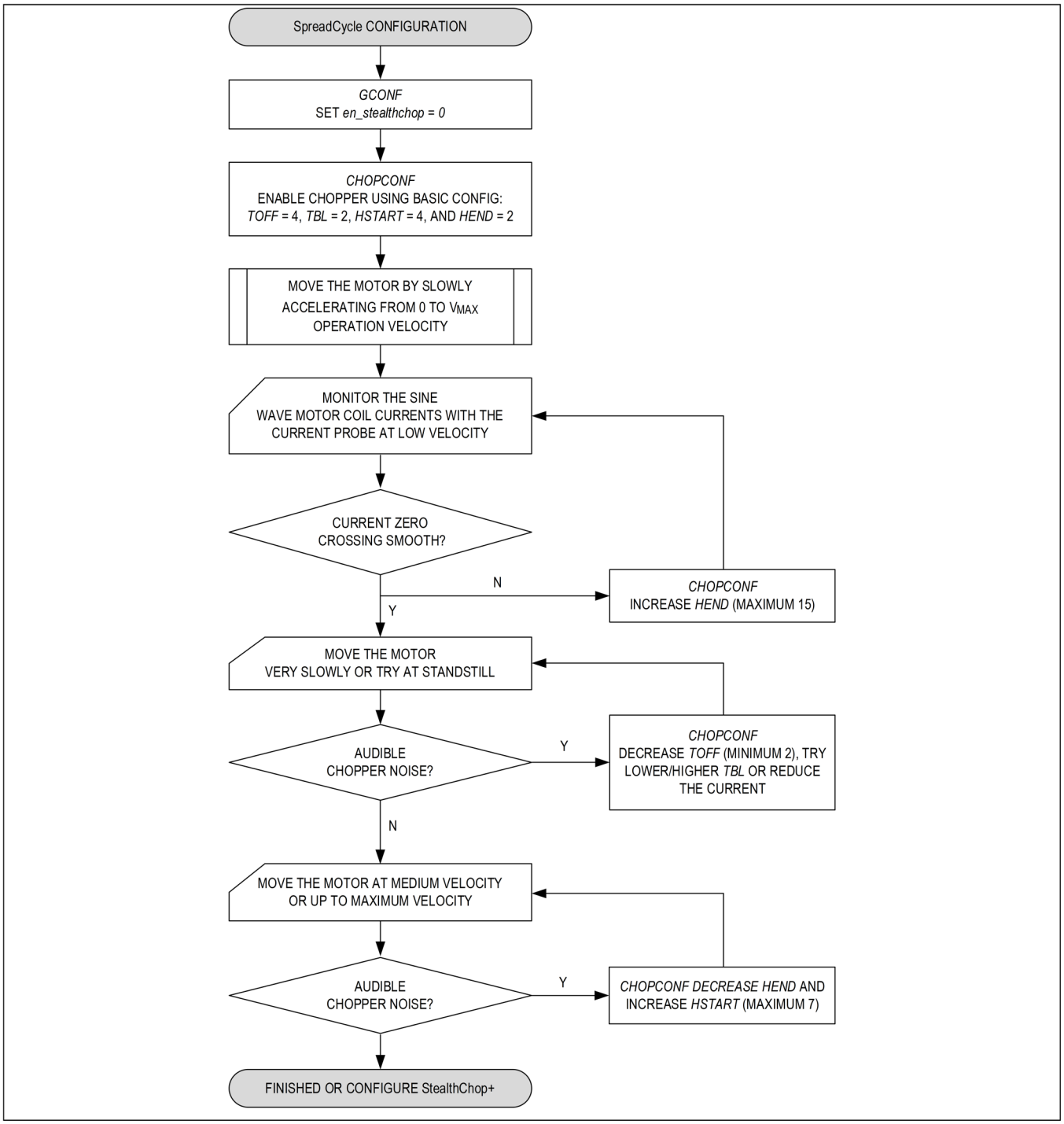


図 62. SpreadCycle のクイック設定ガイド

SpreadCycle と組み合わせた CoolStep

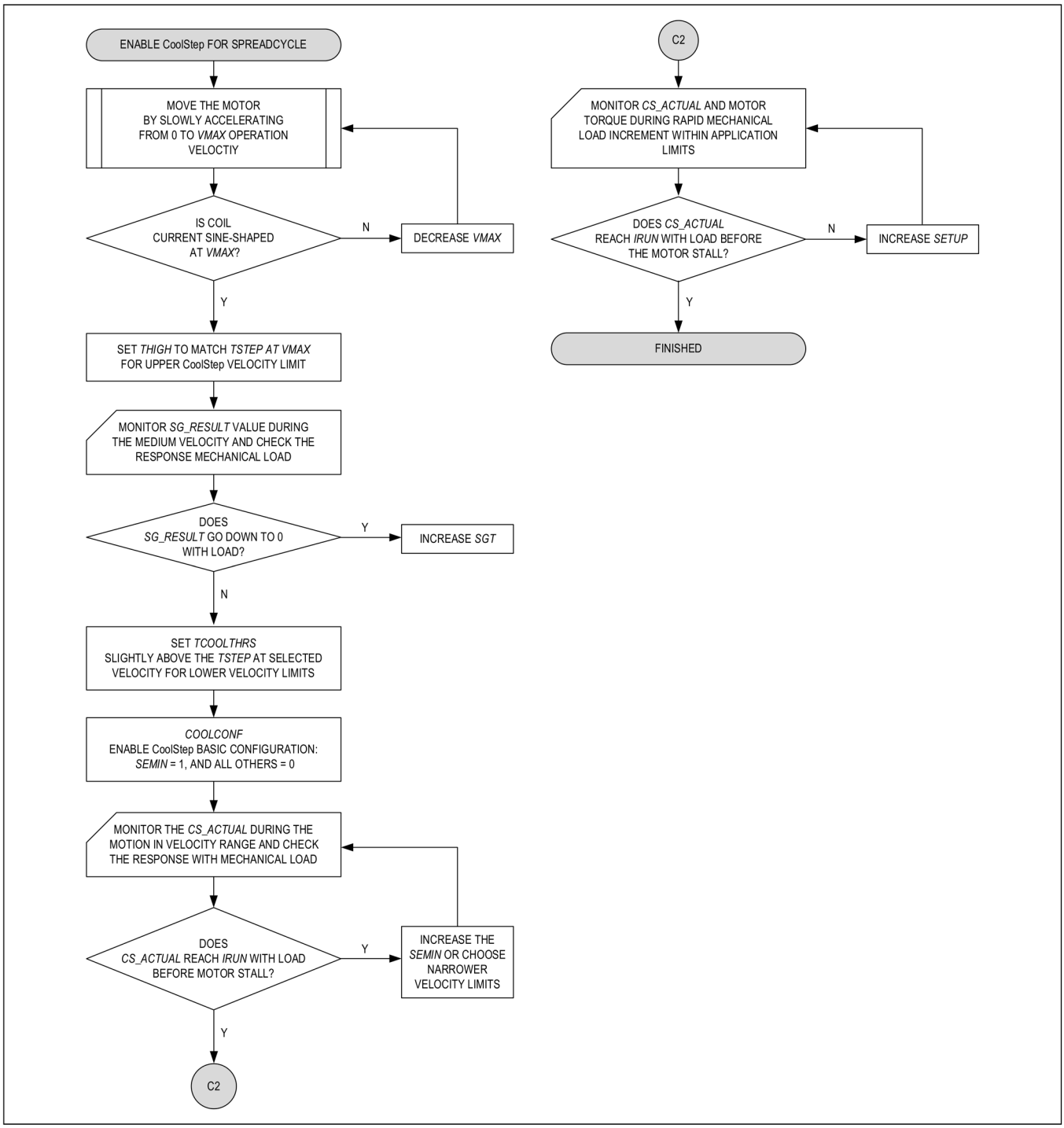


図 63. SpreadCycle と組み合わせた CoolStep 設定のためのクイック設定ガイド

初期化および設定の例

以下に、TMC2262 のパワーアップ後またはリセット後のレジスタの基本的な初期化例を示します。Write()関数および Read()関数は、この例のダミー関数です。特定のマイクロコントローラや SPI ペリフェラルを実装して TMC2262 と共に使用するためには、これらの関数を、実際の読出し／書込み機能で置き換える必要があります。

```
//PLL Initialization
Write(PLL, 0x05FF);                 //Configure for 16MHz external clock; commit set
Wait(1);                            //Wait 1ms
Write(PLL, 0x65FE);                 //Re-Write with commit 0 and reset error flags
Read(PLL);                          //PLL-Register read to poll error flags

//Most relevant configuration
Write(CHOPCONF, 0x00410255);        //MRES=0, TPDFD=4, TBL=3, CHM=0, DISFDCC=0, FD3=0
                                    //HEND_OFFSET=4, HSTRT_TFD210=5, TOFF=5
Write(DRV_CONF, 0x0000003f);        //full current range (4A),CURRENT_RANGE=3,
                                    //CURRENT_RANGE_SCALE=3, SLOPE_CONTROL=3
Write(IHOLD_IRUN, 0x00077f01);      //IRUN=127(=50%), IHOLD=1, IHOLDDELAY=7
Write(TCOOLTHRS, 1048575);          //Set lower threshold velocity for switching on
                                    //CoolStep(+) and StallGuard(+)
```

サステナビリティのための設計

持続可能な成長は、現在最も重要かつ急を要する課題の 1 つです。アナログ・デバイセズは、消費エネルギーを最小限に抑える高効率のモーター・ドライバ製品を設計することでこれに貢献します。それにより、円滑で静音の動作によって最高のカスタマ・エクスペリエンスと長期にわたる満足感を確保すると同時に、電源および冷却用インフラなどの外部リソースに対する要求を最小限に抑え、インテリジェント制御と高度なアルゴリズムによりモーターのサイズや磁石材料の使用量を削減します。

レジスタ・マップ

TMC2262

ADDRESS	NAME	MSB							LSB
General Configuration Registers									
0x00	GCONF[31:24]	–	–	–	–	–	–	–	–
	GCONF[23:16]	–	–	–	–	–	–	–	–
	GCONF[15:8]	–	–	–	OV_nN	–	–	–	–
	GCONF[7:0]	–	direct_mode	stop_enable	small_hysteresis	shaft	multistep_filt	en_stealthchop	fast_standstill
0x01	GSTAT[31:24]	–	–	–	–	–	–	–	–
	GSTAT[23:16]	–	–	–	–	–	–	–	–
	GSTAT[15:8]	–	–	–	–	–	–	–	–
	GSTAT[7:0]	–	–	vccio_uv	vm_uvlo	register_reset	uv_cp	drv_err	reset
0x02	DO_CONF[31:24]	do1_invPP	do1_nOD_PP	do0_invPP	do0_nOD_PP	–	–	do1_ev_n_deviation	–
	DO_CONF[23:16]	–	–	do1_udcstep	do1_ov	–	–	–	do1_index
	DO_CONF[15:8]	do1_stall	do1_otpw	do1_error	do0_ev_n_deviation	–	–	–	do0_udcstep
	DO_CONF[7:0]	do0_ov	–	–	–	do0_index	do0_stall	do0_otpw	do0_error
0x03	DO_SCOPE_CONF[31:24]	–	–	–	–	–	–	–	–
	DO_SCOPE_CONF[23:16]	–	–	–	–	–	–	–	–
	DO_SCOPE_CONF[15:8]	–	–	–	–	–	–	–	–
	DO_SCOPE_CONF[7:0]	–	–	–	–	–	–	–	–
0x04	IOIN[31:24]	–	–	–	–	–	–	–	–
	IOIN[23:16]	–	–	–	–	–	SILICON_RV[2:0]		
	IOIN[15:8]	–	ext_clk	ext_res_det	–	–	–	–	–

ADDRESS	NAME	MSB							LSB
	IOIN[7:0]	reserved	–	ENCN	DRV_ENN	ENCA	ENCB	DIR	STEP
0x0A	DRV_CONF[31:24]	–	–	–	–	–	–	–	–
	DRV_CONF[23:16]	–	–	–	–	–	–	–	–
	DRV_CONF[15:8]	–	–	–	–	–	–	–	–
	DRV_CONF[7:0]	–	–	–	–	–	–	–	–
0x0B	PLL[31:24]	–	–	–	–	–	–	–	–
	PLL[23:16]	–	–	–	–	–	–	–	–
	PLL[15:8]	–	clk_is_stuck	clk_loss	clk_1m0_tmo	–	–	–	–
	PLL[7:0]	–	–	–	–	clk_fsm_ena	clk_sys_sel	ext_not_int	commit
Velocity Dependent Configuration Registers									
0x10	IHOLD_IRUN[31:24]	–	–	–	–	IRUNDELAY[3:0]			
	IHOLD_IRUN[23:16]	IHOLDDELAY[7:0]							
	IHOLD_IRUN[15:8]	IRUN[7:0]							
	IHOLD_IRUN[7:0]	IHOLD[7:0]							
0x11	TPOWERDOWN[31:24]	–	–	–	–	–	–	–	–
	TPOWERDOWN[23:16]	–	–	–	–	–	–	–	–
	TPOWERDOWN[15:8]	–	–	–	–	–	–	–	–
	TPOWERDOWN[7:0]	TPOWERDOWN[7:0]							
0x12	TSTEP[31:24]	–	–	–	–	–	–	–	–
	TSTEP[23:16]	–	–	–	–	TSTEP[19:16]			
	TSTEP[15:8]	TSTEP[15:8]							
	TSTEP[7:0]	TSTEP[7:0]							
0x13	TPWMTHRS[31:24]	–	–	–	–	–	–	–	–
	TPWMTHRS[23:16]	–	–	–	–	TPWMTHRS[19:16]			
	TPWMTHRS[15:8]	TPWMTHRS[15:8]							

ADDRESS	NAME	MSB							LSB
	TPWMTHRS[7:0]	TPWMTHRS[7:0]							
0x14	TCOOLTHRS[31:24]	–	–	–	–	–	–	–	–
	TCOOLTHRS[23:16]	–	–	–	–	TCOOLTHRS[19:16]			
	TCOOLTHRS[15:8]	TCOOLTHRS[15:8]							
	TCOOLTHRS[7:0]	TCOOLTHRS[7:0]							
0x15	THIGH[31:24]	–	–	–	–	–	–	–	–
	THIGH[23:16]	–	–	–	–	THIGH[19:16]			
	THIGH[15:8]	THIGH[15:8]							
	THIGH[7:0]	THIGH[7:0]							
0x16	TSGP_LOW_VEL_THRS[31:24]	–	–	–	–	–	–	–	–
	TSGP_LOW_VEL_THRS[23:16]	–	–	–	–	TSGP_LOW_VEL_THRS[19:16]			
	TSGP_LOW_VEL_THRS[15:8]	TSGP_LOW_VEL_THRS[15:8]							
	TSGP_LOW_VEL_THRS[7:0]	TSGP_LOW_VEL_THRS[7:0]							
0x17	T_RCOIL_MEAS[31:24]	–	–	–	–	–	–	–	–
	T_RCOIL_MEAS[23:16]	–	–	–	–	T_RCOIL_MEAS[19:16]			
	T_RCOIL_MEAS[15:8]	T_RCOIL_MEAS[15:8]							
	T_RCOIL_MEAS[7:0]	T_RCOIL_MEAS[7:0]							
0x18	TUDCSTEP[31:24]	–	–	–	–	–	–	–	–
	TUDCSTEP[23:16]	–	–	–	–	TUDCSTEP[19:16]			
	TUDCSTEP[15:8]	TUDCSTEP[15:8]							
	TUDCSTEP[7:0]	TUDCSTEP[7:0]							
0x19	UDC_CONF[31:24]	–	–	–	–	–	–	–	–
	UDC_CONF[23:16]	–	–	–	–	–	–	–	–
	UDC_CONF[15:8]	–	–	–	–	–	–	–	udc_enable

ADDRESS	NAME	MSB							LSB
	UDC_CONF[7:0]	–	–	–	–	DECEL_THRS[3:0]			
0x1A	STEPS_LOST[31:24]	–	–	–	–	–	–	–	–
	STEPS_LOST[23:16]	–	–	–	–	STEPS_LOST[19:16]			
	STEPS_LOST[15:8]	STEPS_LOST[15:8]							
	STEPS_LOST[7:0]	STEPS_LOST[7:0]							
Encoder Registers									
0x38	ENCMODE[31:24]	–	–	–	–	–	–	–	–
	ENCMODE[23:16]	BEMF_BLANK_TIME[7:0]							
	ENCMODE[15:8]	–	–	–	–	–	–	–	–
	ENCMODE[7:0]	–	–	–	–	–	–	–	–
0x39	X_ENC[31:24]	X_ENC[31:24]							
	X_ENC[23:16]	X_ENC[23:16]							
	X_ENC[15:8]	X_ENC[15:8]							
	X_ENC[7:0]	X_ENC[7:0]							
0x3A	ENC_CONST[31:24]	ENC_CONST[31:24]							
	ENC_CONST[23:16]	ENC_CONST[23:16]							
	ENC_CONST[15:8]	ENC_CONST[15:8]							
	ENC_CONST[7:0]	ENC_CONST[7:0]							
0x3B	ENC_STATUS[31:24]	–	–	–	–	–	–	–	–
	ENC_STATUS[23:16]	–	–	–	–	–	–	–	–
	ENC_STATUS[15:8]	–	–	–	–	–	–	–	–
	ENC_STATUS[7:0]	–	–	–	–	–	–	deviation_warn	n_event
0x3C	ENC_LATCH[31:24]	ENC_LATCH[31:24]							
	ENC_LATCH[23:16]	ENC_LATCH[23:16]							
	ENC_LATCH[15:8]	ENC_LATCH[15:8]							

ADDRESS	NAME	MSB							LSB
	ENC_LATCH[7:0]	ENC_LATCH[7:0]							
0x3D	ENC_DEVIATION[31:24]	–	–	–	–	–	–	–	–
	ENC_DEVIATION[23:16]	–	–	–	–	ENC_DEVIATION[19:16]			
	ENC_DEVIATION[15:8]	ENC_DEVIATION[15:8]							
	ENC_DEVIATION[7:0]	ENC_DEVIATION[7:0]							
StealthChopPlus									
0x40	CURRENT_PI_REG[31:24]	–	–	–	–	–	–	CUR_I[9:8]	
	CURRENT_PI_REG[23:16]	CUR_I[7:0]							
	CURRENT_PI_REG[15:8]	–	–	–	–	CUR_P[11:8]			
	CURRENT_PI_REG[7:0]	CUR_P[7:0]							
0x41	ANGLE_PI_REG[31:24]	–	–	–	–	–	–	ANGLE_I[9:8]	
	ANGLE_PI_REG[23:16]	ANGLE_I[7:0]							
	ANGLE_PI_REG[15:8]	–	–	–	–	ANGLE_P[11:8]			
	ANGLE_PI_REG[7:0]	ANGLE_P[7:0]							
0x42	CUR_ANGLE_LIMIT[31:24]	cur_pi_neg_clip	cur_pi_pos_clip	cur_pi_int_neg_clip	cur_pi_int_pos_clip	CUR_PI_LIMIT[11:8]			
	CUR_ANGLE_LIMIT[23:16]	CUR_PI_LIMIT[7:0]							
	CUR_ANGLE_LIMIT[15:8]	angle_pi_neg_clip	angle_pi_pos_clip	angle_pi_int_neg_clip	angle_pi_int_pos_clip	–	–	ANGLE_PI_LIMIT[9:8]	
	CUR_ANGLE_LIMIT[7:0]	ANGLE_PI_LIMIT[7:0]							
0x43	ANGLE_LOWER_LIMIT[31:24]	–	–	–	–	–	–	ANGLE_ERROR[9:8]	
	ANGLE_LOWER_LIMIT[23:16]	ANGLE_ERROR[7:0]							
	ANGLE_LOWER_LIMIT[15:8]	–	–	–	–	–	–	ANGLE_LOWER_I_LIMIT[9:8]	

ADDRESS	NAME	MSB							LSB
	ANGLE_LOWER_LIMIT[7:0]	ANGLE_LOWER_I_LIMIT[7:0]							
0x44	CUR_ANGLE_MEAS[31:24]	–	–	–	–	–	–	ANGLE_MEAS[9:8]	
	CUR_ANGLE_MEAS[23:16]	ANGLE_MEAS[7:0]							
	CUR_ANGLE_MEAS[15:8]	–	–	–	–	AMPL_MEAS[11:8]			
	CUR_ANGLE_MEAS[7:0]	AMPL_MEAS[7:0]							
0x45	PI_RESULTS[31:24]	–	–	–	–	–	–	ANGLE_CORR_CALC[9:8]	
	PI_RESULTS[23:16]	ANGLE_CORR_CALC[7:0]							
	PI_RESULTS[15:8]	–	–	–	PWM_CALC[12:8]				
	PI_RESULTS[7:0]	PWM_CALC[7:0]							
0x46	COIL_INDUCT[31:24]	–	–	–	–	–	–	–	–
	COIL_INDUCT[23:16]	–	–	–	–	–	–	rcoil_thermal_coupling	rcoil_manual
	COIL_INDUCT[15:8]	–	COIL_INDUCT[14:8]						
	COIL_INDUCT[7:0]	COIL_INDUCT[7:0]							
0x47	R_COIL[31:24]	–	–	–	–	R_COIL_AUTO_A[11:8]			
	R_COIL[23:16]	R_COIL_AUTO_A[7:0]							
	R_COIL[15:8]	–	–	–	–	R_COIL_AUTO_B[11:8]			
	R_COIL[7:0]	R_COIL_AUTO_B[7:0]							
0x48	R_COIL_USER[31:24]	–	–	–	–	R_COIL_USER_A[11:8]			
	R_COIL_USER[23:16]	R_COIL_USER_A[7:0]							
	R_COIL_USER[15:8]	–	–	–	–	R_COIL_USER_B[11:8]			
	R_COIL_USER[7:0]	R_COIL_USER_B[7:0]							
0x49	SGP_CONF[31:24]	–	–	SGP_LOW_VEL_CNTS[1:0]		–	–	–	–

ADDRESS	NAME	MSB							LSB
	SGP_CONF[23:16]	SGP_LOW_VEL_SLOPE[7:0]							
	SGP_CONF[15:8]	–	–	–	–	–	–	–	SGP_THRS[8]
	SGP_CONF[7:0]	SGP_THRS[7:0]							
0x4A	SGP_IND_2_3[31:24]	–	–	–	–	–	–	SGP_IND_3[9:8]	
	SGP_IND_2_3[23:16]	SGP_IND_3[7:0]							
	SGP_IND_2_3[15:8]	–	–	–	–	–	–	SGP_IND_2[9:8]	
	SGP_IND_2_3[7:0]	SGP_IND_2[7:0]							
0x4B	SGP_IND_0_1[31:24]	–	–	–	–	–	–	SGP_IND_1[9:8]	
	SGP_IND_0_1[23:16]	SGP_IND_1[7:0]							
	SGP_IND_0_1[15:8]	–	–	–	–	–	–	SGP_IND_0[9:8]	
	SGP_IND_0_1[7:0]	SGP_IND_0[7:0]							
0x4C	INDUCTANCE_VOLTAGE[31:24]	–	–	–	–	UL_A[11:8]			
	INDUCTANCE_VOLTAGE[23:16]	UL_A[7:0]							
	INDUCTANCE_VOLTAGE[15:8]	–	–	–	–	UL_B[11:8]			
	INDUCTANCE_VOLTAGE[7:0]	UL_B[7:0]							
0x4D	SGP_BEMF[31:24]	–	–	–	–	UBEMF_ABS[11:8]			
	SGP_BEMF[23:16]	UBEMF_ABS[7:0]							
	SGP_BEMF[15:8]	–	–	–	–	–	–	SGP_RAW[9:8]	
	SGP_BEMF[7:0]	SGP_RAW[7:0]							
0x4E	COOLSTEPPLUS_CONF[31:24]	–	–	–	–	–	–	–	–
	COOLSTEPPLUS_CONF[23:16]	–	–	–	–	–	–	–	–
	COOLSTEPPLUS_CONF[15:8]	–	–	–	–	–	–	–	–

ADDRESS	NAME	MSB							LSB
	COOLSTEPPLUS_CONF[7:0]	–	–	–	load_filt_en	–	–	–	–
0x4F	COOLSTEPPLUS_PI_REG[31:24]	–	–	–	–	–	–	COOLSTEP_I[9:8]	
	COOLSTEPPLUS_PI_REG[23:16]	COOLSTEP_I[7:0]							
	COOLSTEPPLUS_PI_REG[15:8]	–	–	–	–	COOLSTEP_P[11:8]			
	COOLSTEPPLUS_PI_REG[7:0]	COOLSTEP_P[7:0]							
0x50	COOLSTEPPLUS_PI_DOWN[31:24]	–	–	–	–	COOL_PI_OFF_SPEED[11:8]			
	COOLSTEPPLUS_PI_DOWN[23:16]	COOL_PI_OFF_SPEED[7:0]							
	COOLSTEPPLUS_PI_DOWN[15:8]	–	–	–	–	COOL_PI_DOWN_LIMIT[11:8]			
	COOLSTEPPLUS_PI_DOWN[7:0]	COOL_PI_DOWN_LIMIT[7:0]							
0x51	COOLSTEPPLUS_RESERVE_CONF[31:24]	COOL_HI_GENERATORIC_RESERVE[7:0]							
	COOLSTEPPLUS_RESERVE_CONF[23:16]	COOL_LOW_GENERATORIC_RESERVE[7:0]							
	COOLSTEPPLUS_RESERVE_CONF[15:8]	COOL_HI_LOAD_RESERVE[7:0]							
	COOLSTEPPLUS_RESERVE_CONF[7:0]	COOL_LOW_LOAD_RESERVE[7:0]							
0x52	COOLSTEPPLUS_LOAD_RESERVE[31:24]	–	–	–	–	–	–	–	COOLSTEP_LOAD_RESERVE[8]
	COOLSTEPPLUS_LOAD_RESERVE[23:16]	COOLSTEP_LOAD_RESERVE[7:0]							
	COOLSTEPPLUS_LOAD_RESERVE[15:8]	–	–	–	–	–	–	SGP_RESULT[9:8]	
	COOLSTEPPLUS_LOAD_RESERVE[7:0]	SGP_RESULT[7:0]							

ADDRESS	NAME	MSB							LSB
0x53	TSTEP Velocity[31:24]	–	–	–	–	–	–	–	–
	TSTEP Velocity[23:16]	–	TSTEP_VELOCITY[22:16]						
	TSTEP Velocity[15:8]	TSTEP_VELOCITY[15:8]							
	TSTEP Velocity[7:0]	TSTEP_VELOCITY[7:0]							
ADC Registers									
0x58	ADC_VSUPPLY_TEMP[31:24]	–	–	–	–	–	–	–	ADC_TEMP[8]
	ADC_VSUPPLY_TEMP[23:16]	ADC_TEMP[7:0]							
	ADC_VSUPPLY_TEMP[15:8]	–	–	–	–	–	–	–	ADC_VSUPPLY[8]
	ADC_VSUPPLY_TEMP[7:0]	ADC_VSUPPLY[7:0]							
0x59	ADC_I[31:24]	–	–	–	–	ADC_I_B[11:8]			
	ADC_I[23:16]	ADC_I_B[7:0]							
	ADC_I[15:8]	–	–	–	–	ADC_I_A[11:8]			
	ADC_I[7:0]	ADC_I_A[7:0]							
0x5A	OTW_OV_VTH[31:24]	–	–	–	–	–	–	–	OVERTEMPPREWARNING_VTH[8]
	OTW_OV_VTH[23:16]	OVERTEMPPREWARNING_VTH[7:0]							
	OTW_OV_VTH[15:8]	–	–	–	–	–	–	–	OVERVOLTAGE_VTH[8]
	OTW_OV_VTH[7:0]	OVERVOLTAGE_VTH[7:0]							
Motor Driver Registers									
0x60	MSLUT_0[31:24]	MSLUT_0[31:24]							
	MSLUT_0[23:16]	MSLUT_0[23:16]							
	MSLUT_0[15:8]	MSLUT_0[15:8]							
	MSLUT_0[7:0]	MSLUT_0[7:0]							
0x61	MSLUT_1[31:24]	MSLUT_1[31:24]							

ADDRESS	NAME	MSB							LSB
	MSLUT_1[23:16]	MSLUT_1[23:16]							
	MSLUT_1[15:8]	MSLUT_1[15:8]							
	MSLUT_1[7:0]	MSLUT_1[7:0]							
0x62	MSLUT_2[31:24]	MSLUT_2[31:24]							
	MSLUT_2[23:16]	MSLUT_2[23:16]							
	MSLUT_2[15:8]	MSLUT_2[15:8]							
	MSLUT_2[7:0]	MSLUT_2[7:0]							
0x63	MSLUT_3[31:24]	MSLUT_3[31:24]							
	MSLUT_3[23:16]	MSLUT_3[23:16]							
	MSLUT_3[15:8]	MSLUT_3[15:8]							
	MSLUT_3[7:0]	MSLUT_3[7:0]							
0x64	MSLUT_4[31:24]	MSLUT_4[31:24]							
	MSLUT_4[23:16]	MSLUT_4[23:16]							
	MSLUT_4[15:8]	MSLUT_4[15:8]							
	MSLUT_4[7:0]	MSLUT_4[7:0]							
0x65	MSLUT_5[31:24]	MSLUT_5[31:24]							
	MSLUT_5[23:16]	MSLUT_5[23:16]							
	MSLUT_5[15:8]	MSLUT_5[15:8]							
	MSLUT_5[7:0]	MSLUT_5[7:0]							
0x66	MSLUT_6[31:24]	MSLUT_6[31:24]							
	MSLUT_6[23:16]	MSLUT_6[23:16]							
	MSLUT_6[15:8]	MSLUT_6[15:8]							
	MSLUT_6[7:0]	MSLUT_6[7:0]							
0x67	MSLUT_7[31:24]	MSLUT_7[31:24]							
	MSLUT_7[23:16]	MSLUT_7[23:16]							
	MSLUT_7[15:8]	MSLUT_7[15:8]							

ADDRESS	NAME	MSB							LSB
	MSLUT_7[7:0]	MSLUT_7[7:0]							
0x68	MSLUTSEL[31:24]	X3[7:0]							
	MSLUTSEL[23:16]	X2[7:0]							
	MSLUTSEL[15:8]	X1[7:0]							
	MSLUTSEL[7:0]	–	–	–	–	–	–	–	–
0x69	MSLUTSTART[31:24]	OFFSET_SIN90[7:0]							
	MSLUTSTART[23:16]	START_SIN90[7:0]							
	MSLUTSTART[15:8]	–	–	–	–	–	–	–	–
	MSLUTSTART[7:0]	START_SIN[7:0]							
0x6A	MSCNT[31:24]	–	–	–	–	–	–	–	–
	MSCNT[23:16]	–	–	–	–	–	–	–	–
	MSCNT[15:8]	–	–	–	–	–	–	MSCNT[9:8]	
	MSCNT[7:0]	MSCNT[7:0]							
0x6B	MSCURACT[31:24]	–	–	–	–	–	–	–	CUR_A[8]
	MSCURACT[23:16]	CUR_A[7:0]							
	MSCURACT[15:8]	–	–	–	–	–	–	–	CUR_B[8]
	MSCURACT[7:0]	CUR_B[7:0]							
0x6C	CHOPCONF[31:24]	–	–	–	–	–	–	–	–
	CHOPCONF[23:16]	TPFD[3:0]				–	–	–	–
	CHOPCONF[15:8]	–	chm	–	–	fd3	HEND_OFFSET[3:1]		
	CHOPCONF[7:0]	HEND_OFFSET[0]	HSTRT_TFD210[2:0]			–	–	–	–
0x6D	COOLCONF[31:24]	–	–	–	–	–	–	–	–
	COOLCONF[23:16]	–	sgt[6:0]						
	COOLCONF[15:8]	–	–	–	–	–	–	–	–
	COOLCONF[7:0]	–	–	–	–	–	–	–	–

ADDRESS	NAME	MSB							LSB
0x6F	DRV_STATUS[31:24]	–	olb	ola	s2gb	s2ga	–	ot	stallguard
	DRV_STATUS[23:16]	CS_ACTUAL[7:0]							
	DRV_STATUS[15:8]	–	stealth	s2vsb	s2vsa	–	–	SG_RESULT[9:8]	
	DRV_STATUS[7:0]	SG_RESULT[7:0]							
0x70	PWMCONF[31:24]	–	–	–	–	–	–	–	–
	PWMCONF[23:16]	–	–	–	–	SD_ON_MEAS_HI[3:0]			
	PWMCONF[15:8]	SD_ON_MEAS_LO[3:0]				–	–	–	–
	PWMCONF[7:0]	–	–	–	–	–	–	–	–

レジスタ・ブロック

全般的な設定レジスタ

レジスタの詳細

GCONF（0x0）

グローバル設定フラグ

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8

Field	–	–	–	OV_nN	–	–	–	–
Reset	–	–	–	0x0	–	–	–	–
Access Type	–	–	–	Write, Read	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	direct_mode	stop_enable	small_hysteresis	shaft	multistep_filt	en_stealthchop	fast_standstill
Reset	–	0b0	0b0	0b1	0b0	0b1	0b0	0b0
Access Type	–	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read	Write, Read

ビットフィールド	ビット	説明	デコード
OV_nN	12	ENCN の過電圧出力（ハイ・アクティブのオープン・ドレイン）としての使用をイネーブル。	0x0: Use ENCN as N-channel of encoder input (default). 0x1: Use ENCN as overvoltage output (high active open-drain). Hint: in this case, the N signal seen by the IC is always 0.
direct_mode	6	シリアル・インターフェースを介して直接モーター位相電流制御をイネーブル。	0x0: Normal operation 0x1: Motor coil currents and polarity directly programmed through the serial interface: register XTARGET (0x2D) specifies signed coil A current (bits 8..0) and coil B current (bits 24..16). In this mode, the current is scaled by IHOLD setting. Velocity-based current regulation of StealthChop+ is not available in this mode. The automatic StealthChop+ current regulation works only for low stepper motor velocities.
stop_enable	5	モーター・ハード・ストップ機能をイネーブル。	0x0: Normal operation 0x1: Emergency stop: ENCA stops the sequencer when tied high (no steps are executed by the sequencer, motor goes to standstill state).
small_hysteresis	4	速度スレッショルドの TSTEP 比較のためのヒステリシス制御。	0x0: Hysteresis for step frequency comparison is 1/16 (higher tolerance to frequency jitter with external STEP and DIR source). 0x1: Hysteresis for step frequency comparison is 1/32 (recommended with hardware-based ramp generator).
shaft	3	モーター方向反転 DIR 信号を変更。	0x0: Default motor direction 0x1: Inverse motor direction
multistep_filt	2	StealthChop+用のステップ入力フィルタ処理をイネーブル。	0x0: Step input filtering disabled. 0x1: Enable step input filtering for StealthChop+ optimization with external step source (default = 1).
en_stealthchop	1	StealthChop+モードを有効化	0x0: No StealthChop+ 0x1: StealthChop+ voltage PWM mode enabled (depending on velocity thresholds). Switch from off to on state while in standstill.
fast_standstill	0	停止検出までのステップ・パルス・タイムアウト。	0x0: Normal time: 2^{20} clocks (required if very low step frequencies are used, for example, with low microstep resolution). 0x1: Short time: 2^{18} clocks (recommended for fast current reduction).

GSTAT（0x1）

グローバル・ステータス・フラグ

（各フラグをクリアするには「1」のビットを再書込みします）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	vccio_uv	vm_uvlo	register_reset	uv_cp	drv_err	reset
Reset	–	–	0b0	0b0	0b1	0b0	0b0	0b1
Access Type	–	–	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read

ビットフィールド	ビット	説明	デコード
vccio_uv	5	1：最後のリセット後に VCC_IO 低電圧が発生。最初のパワーアップ後にクリアされます。	0x0: Normal operation 0x1: VCC_IO dropped below the lower threshold after last access to this register. Interface commands may have been lost.
vm_uvlo	4	1：最後のリセット後に VM 低電圧が発生。最初のパワーアップ後にクリアされます。	0x0: Normal operation 0x1: VM had undervoltage, potential loss of motor position.
register_reset	3	設定レジスタがリセットされたことを示します。	0x0: Normal operation 0x1: Indicates the register map is reset. All registers are cleared to reset values.
uv_cp	2	チャージ・ポンプ低電圧条件フラグ。パワーアップ後にクリアされます。	0x0: Normal operation 0x1: Indicates an undervoltage on the charge pump. The driver is disabled during undervoltage. This flag is latched for information.

ビットフィールド	ビット	説明	デコード
drv_err	1	ドライバ・エラー・フラグ	0x0: Normal operation 0x1: Indicates the driver is shut down due to overtemperature or short circuit detection. Read DRV_STATUS for details. The flag can only be cleared when the temperature is below the limit again.
reset	0	リセット・フラグ	0x0: Normal operation 0x1: Indicates the IC is reset since the last clear of this bit.

DO_CONF（0x2）

DO0 と DO1 の各出力ピンの設定。

BIT	31	30	29	28	27	26	25	24
Field	do1_invPP	do1_nOD_PP	do0_invPP	do0_nOD_PP	–	–	do1_ev_n_deviation	–
Reset	0x1	0b0	0x1	0b0	–	–		–
Access Type	Write, Read	Write, Read	Write, Read	Write, Read	–	–	Write, Read	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	do1_udcstep	do1_ov	–	–	–	do1_index
Reset	–	–			–	–	–	
Access Type	–	–	Write, Read	Write, Read	–	–	–	Write, Read
BIT	15	14	13	12	11	10	9	8
Field	do1_stall	do1_otpw	do1_error	do0_ev_n_deviation	–	–	–	do0_udcstep
Reset	0b0	0b0	0b0		–	–	–	
Access Type	Write, Read	Write, Read	Write, Read	Write, Read	–	–	–	Write, Read
BIT	7	6	5	4	3	2	1	0
Field	do0_ov	–	–	–	do0_index	do0_stall	do0_otpw	do0_error
Reset		–	–	–		0b0	0b0	0b0
Access Type	Write, Read	–	–	–	Write, Read	Write, Read	Write, Read	Write, Read

ビットフィールド	ビット	説明	デコード
do1_invPP	31	プッシュプル・モードでの DO1 の極性を選択。	0x0: High active output 0x1: Low active output
do1_nOD_PP	30	DO1 の出力タイプ設定。	0x0: DO1 is open collector output (active low). 0x1: Enable DO1 push-pull output.
do0_invPP	29	プッシュプル・モードでの DO0 の極性を選択。	0x0: High active output 0x1: Low active output
do0_nOD_PP	28	DO0 の出力タイプ設定。	0x0: DO0 is open collector output (active low). 0x1: Enable DO0 push-pull output.
do1_ev_n_deviation	25	N イベントおよび偏差警告を DO1 に割り当てます。	0x0: Disable 0x1: Enable
do1_udcstep	21	外部ランプ・ジェネレータが減速できるようにする µDcStep 高負荷検出を、DO1 に割り当てます。	0x0: Disable 0x1: Enable
do1_ov	20	過電圧の検出を DO1 に割り当てます。	0x0: Disable 0x1: Enable
do1_index	16	インデックス信号を DO1 に割り当てます。	0x0: Disable 0x1: Enable
do1_stall	15	StallGuard 信号 stallguard を DO1 に割り当てます。	0x0: Disable 0x1: Enable
do1_otpw	14	OTPW（過熱事前警告）を DO1 に割り当てます。	0x0: Disable 0x1: Enable
do1_error	13	ドライバのエラー状態を DO1 に割り当てます。	0x0: Disable 0x1: Enable
do0_ev_n_deviation	12	N イベントおよび偏差警告を DO0 に割り当てます。	0x0: Disable 0x1: Enable
do0_udcstep	8	外部ランプ・ジェネレータが減速できるようにする µDcStep 高負荷検出を、DO0 に割り当てます。	0x0: Disable 0x1: Enable
do0_ov	7	過電圧の検出を DO0 に割り当てます。	0x0: Disable 0x1: Enable
do0_index	3	インデックス信号を DO0 に割り当てます。	0x0: Disable 0x1: Enable
do0_stall	2	StallGuard 信号 stallguard を DO0 に割り当てます。	0x0: Disable 0x1: Enable
do0_otpw	1	OTPW（過熱事前警告）を DO0 に割り当てます。	0x0: Disable 0x1: Enable
do0_error	0	ドライバのエラー状態をDO0に割り当てます。 DO0 は常にリセット状態を示します。これは、リセット状態の間、アクティブ・ローです。	0x0: Disable 0x1: Enable

IOIN（0x4）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–

Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	–	SILICON_RV[2:0]		
Reset	–	–	–	–	–	0b000		
Access Type	–	–	–	–	–	Read Only		
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	ext_clk	ext_res_det	–	–	–	–	–
Reset	–	0b0	0b0	–	–	–	–	–
Access Type	–	Read Only	Read Only	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	reserved	–	ENCN	DRV_ENN	ENCA	ENCB	DIR	STEP
Reset	0b0	–	0b0	0b0	0b0	0b0	0b0	0b0
Access Type	Read Only	–	Read Only	Read Only	Read Only	Read Only	Read Only	Read Only

ビットフィールド	ビット	説明	デコード
SILICON_RV	18:16	シリコン・リビジョン番号	
ext_clk	14	有効な外部クロック CLK の検出。	0x0: The internal oscillator is used for generating the system clock. 0x1: An external clock signal is used for generating the system clock.
ext_res_det	13	外部リファレンス抵抗のチェック。	0x0: Resistor is missing, driver cannot operate. 0x1: Normal operation: External resistor between REF and GND.
reserved	7		
ENCN	5	ピン ENCN の状態。	0x0: Low logic level 0x1: High logic level
DRV_ENN	4	ドライバのディスエーブル／イネーブル状態。	0x0: Enable 0x1: Disable
ENCA	3	ピン ENCA の状態。	0x0: Low logic level 0x1: High logic level
ENCB	2	ピン ENCB の状態。	0x0: Low logic level 0x1: High logic level
DIR	1	DIR ピンの状態。	0x0: Low logic level 0x1: High logic level
STEP	0	STEP ピンの状態。	0x0: Low logic level 0x1: High logic level

DRV_CONF（0xA）

ドライバの電流レンジおよび勾配の設定。

PLL（0xB）

PLL の設定と動作。

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	clk_is_stuck	clk_loss	clk_1m0_tmo	–	–	–	–
Reset	–				–	–	–	–
Access Type	–	Write 1 to Clear, Read	Write 1 to Clear, Read	Write 1 to Clear, Read	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	clk_fsm_ena	clk_sys_sel	ext_not_int	commit
Reset	–	–	–	–	0x0	0x0	0x0	0x0
Access Type	–	–	–	–	Write, Read	Write, Read	Write, Read	Write, Read

ビットフィールド	ビット	説明	デコード
clk_is_stuck	14	外部クロック信号の喪失。また、CLK 喪失の場合と新しい値が PLL に提供された場合にもセットされます。	0x0: Clock OK 0x1: Clock missing
clk_loss	13	外部クロック用クロック・フォルト検出。クロック周波数または分周器が範囲外の場合にセットされます。また、新しい値が PLL に提供された場合にもセットされます。	0x0: Clock OK 0x1: Clock is faulty or lost.
clk_1m0_tmo	12	PLL の起動中、分周されたクロック周波数を内部発振器と比較することにより、その周波数が範囲内に収まっていることをチェックします。範囲外の場合、このフラグがセットされます。リセットの前に、clk_fsm_ena をディスエーブルし、正しいパラメータを設定する間に再イネーブルします。	0x0: Clock OK 0x1: Clock timeout
clk_fsm_ena	3	PLL FSMをイネーブル。	0x0: PLL disabled, chip goes to reset but registers bank is kept. 0x1: Chip in operation.
clk_sys_sel	2	マスタ・クロックのセレクタ。	0x0: Internal 16MHz clock with PLL disabled. The chip remains in reset during this time. 0x1: PLL clock
ext_not_int	1		0x0: Select internal clock oscillator. 0x1: Select external clock signal applied to pin CLK.
commit	0	変更を PLL ロジックに適用（自動クリア・ビット）。このビットは、PLL が安定するとクリアされます。このビットがクリアされるまで待ってから、エラー・フラグ clk_loss および clk_is_stuck をリセットしてください。	0x0: Write 0 to allow resetting clk_loss and clk_is_stuck, and release the driver from reset state. 0x1: Set to 1 after new valid values have been progammed to ext_not_int, clk_sys_sel, and CLOCK_DIVIDER. The chip goes to a reset state (except for register space) during this time.

速度依存設定レジスタ

動作モードを選択するための電流設定値および速度スレッショルド（TSTEP に基づく）。

レジスタの詳細

IHOLD_IRUN（0x10）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	IRUNDELAY[3:0]			
Reset	–	–	–	–	0x4			
Access Type	–	–	–	–	Write, Read			
BIT	23	22	21	20	19	18	17	16
Field	IHOLDDELAY[7:0]							
Reset	0x7							

Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	IRUN[7:0]							
Reset	0xF0							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	IHOLD[7:0]							
Reset	0d64							
Access Type	Write, Read							

ビットフィールド	ビット	説明	デコード
IRUNDELAY	27:24	開始が検出されてからモーターが起動するまでのクロック・サイクル数を制御します。モーターの始動時の突入電流を低減するには、高めの値を設定します。	0x0: Instant power-up 0x1.. 0xF: Delay per current increment step in multiples of *IRUNDELAY* × 512 clocks.
IHOLDDELAY	23:16	停止が検出されて（stst = 1）*TPOWERDOWN* の時間が経過すると直ちにモーション後のモーター・パワー・ダウンを行うための、クロック・サイクル数を制御します。滑らかな遷移にすることで、パワー・ダウン時のモーター・ジャークを回避できます。	0x0: Instant power down Delay per current reduction step in multiples of 2^{11} clocks.
IRUN	15:8	SpreadCycleのモーター稼働時電流（0 = 1/256、…、255 = 256/256）。 StealthChop+のモーター稼働時電流（0 = 0/256、...、250 = 250/250）。 詳細：最高のマイクロステップ性能を実現するには、通常時の IRUN が 128～255 となるように CURRENT_RANGE を選択します。StealthChop+電流を 251～255 に設定すると、内部で 250 にクリップされます。	
IHOLD	7:0	停止時電流。 詳細：StealthChop+と組み合わせた場合、*IHOLD* = 0 と設定することで、フリーホイールまたはコイル短絡回路を選択してモーターを停止できます。	

TPOWERDOWN（0x11）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	TPOWERDOWN[7:0]							
Reset	0x0A							
Access Type	Write, Read							

ビットフィールド	ビット	説明
TPOWERDOWN	7:0	*TPOWERDOWN*は、モーターの停止（stst）後からモーター電流がパワー・ダウンするまでの遅延時間を設定します。時間範囲は、約0秒～4秒です。 大幅な温度上昇が予想されるアプリケーションでは、一定の最小値を設定することによりモーターのコイル抵抗を再測定できます。 リセット後のデフォルト = 10 0…$((2^8)-1) \times 2^{18}\ t_{CLK}$

TSTEP（0x12）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	TSTEP[19:16]			
Reset	–	–	–	–	0x00000			
Access Type	–	–	–	–	Read Only			
BIT	15	14	13	12	11	10	9	8
Field	TSTEP[15:8]							
Reset	0x00000							
Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	TSTEP[7:0]							
Reset	0x00000							
Access Type	Read Only							

ビットフィールド	ビット	説明
TSTEP	19:0	ステップ入力周波数から引き出される2つの1/256マイクロステップ間の実際の測定時間（$1/f_{CLK}$単位）。オーバーフローまたは停止時の測定値は、$(2^{20})-1$です。 *TSTEP*に関連するスレッショルドはすべて、比較値の1/16のヒステリシスを使用してクロックまたはステップ周波数のジッタを補償します。フラグsmall_hysteresisは、このヒステリシスをより小さな値である1/32に変更します。 (Txxx × 15/16) − 1、または、 (Txxx × 31/32) − 1は、各比較値の2番目の比較値として使用されます。つまり、下限スイッチング速度は計算された設定値に等しくなりますが、上限スイッチング速度はヒステリシス設定で定義された値よりも高くなります。

TPWMTHRS（0x13）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	TPWMTHRS[19:16]			
Reset	–	–	–	–	0x00000			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	TPWMTHRS[15:8]							
Reset	0x00000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	TPWMTHRS[7:0]							
Reset	0x00000							
Access Type	Write, Read							

ビットフィールド	ビット	説明
TPWMTHRS	19:0	StealthChop+動作用の上限速度。 TSTEP ≥ TPWMTHRS StealthChop+がイネーブルされます（設定されている場合）。

TCOOLTHRS（0x14）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–

Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	TCOOLTHRS[19:16]			
Reset	–	–	–	–	0x00000			
Access Type	–	–	–	–	Write, Read			
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	TCOOLTHRS[15:8]							
Reset	0x00000							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	TCOOLTHRS[7:0]							
Reset	0x00000							
Access Type	Write, Read							

ビットフィールド	ビット	説明
TCOOLTHRS	19:0	CoolStep/CoolStep+およびStallGuard/StallGuard+をスイッチ・オンするための下限スレッショルド速度（符号なし）。 このパラメータを設定すると、信頼できる動作を行うことのできない低速時にCoolStep(+)がディスエーブルされます。この速度を超えるとストール出力信号がイネーブルされます。この速度を超えるとイネーブルされます。速度がこのスレッショルド未満になると再度ディスエーブルされます。 *TCOOLTHRS* ≥ *TSTEP* ≥ *THIGH* : CoolStep(+)がイネーブルされます（設定されている場合）。 *TCOOLTHRS* ≥ *TSTEP* : ピン DO0 または DO1 のストール出力信号が有効化されます（設定されている場合）。

THIGH（0x15）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	THIGH[19:16]			
Reset	–	–	–	–	0x00000			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	THIGH[15:8]							
Reset	0x00000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	THIGH[7:0]							
Reset	0x00000							
Access Type	Write, Read							

ビットフィールド	ビット	説明
THIGH	19:0	CoolStep/CoolStep+用の上限速度制限値。 *TSTEP* < *THIGH*の場合、CoolStepおよびCoolStep+はディスエーブルされます。 フル電流時に CoolStep+を使用した StealthChop+から SpreadCycle に滑らかに切り替えるには、*TPWMTRHS* と共に使用します。 *THIGH* > *TSTEP* > *TPWMTHRS* : CoolStep+がディスエーブルされ、電流は *COOL_PI_OFF_SPEED* の定義に従い *IRUN* までランプダウンします。

TSGP_LOW_VEL_THRS（0x16）

BIT	31	30	29	28	27	26	25	24

Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	TSGP_LOW_VEL_THRS[19:16]			
Reset	–	–	–	–	0x1000			
Access Type	–	–	–	–	Write, Read			
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	TSGP_LOW_VEL_THRS[15:8]							
Reset	0x1000							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	TSGP_LOW_VEL_THRS[7:0]							
Reset	0x1000							
Access Type	Write, Read							

ビットフィールド	ビット	説明
TSGP_LOW_VEL_THRS	19:0	低速度StallGuard+ベースのストール検出を行うための*TSTEP*をベースとする速度スレッショルド。 *TCOOLTHRS*より低速／高速の値に設定します。 *TSGP_LOW_VEL_THRS* > *TSTEP* > *TCOOLTHRS* : 低速度 StallGuard+がイネーブルされます。

T_RCOIL_MEAS（0x17）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	T_RCOIL_MEAS[19:16]			
Reset	–	–	–	–	0x1000			
Access Type	–	–	–	–	Write, Read			

BIT	15	14	13	12	11	10	9	8
Field	T_RCOIL_MEAS[15:8]							
Reset	0x1000							
Access Type	Write, Read							

BIT	7	6	5	4	3	2	1	0
Field	T_RCOIL_MEAS[7:0]							
Reset	0x1000							
Access Type	Write, Read							

ビットフィールド	ビット	説明
T_RCOIL_MEAS	19:0	コイル抵抗を自動測定するための上限速度。 *TSTEP* > *T_RCOIL_MEAS* : 自動抵抗測定は低速動作時または停止時にイネーブルされます。 *TSTEP* <= *T_RCOIL_MEAS* : 自動抵抗測定はディスエーブルされます。

TUDCSTEP（0x18）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	TUDCSTEP[19:16]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	TUDCSTEP[15:8]							
Reset								
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	TUDCSTEP[7:0]							
Reset								
Access Type	Write, Read							

ビットフィールド	ビット	説明
TUDCSTEP	19:0	µDcStepのイネーブル・スレッショルド。 *TUDCSTEP* < *TSTEP* : µDcStepがイネーブルされます。 （CoolStep+がアクティブになっている場合のみ）。

UDC_CONF（0x19）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–

Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	–	udc_enable
Reset	–	–	–	–	–	–	–	
Access Type	–	–	–	–	–	–	–	Write, Read
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	–	–	DECEL_THRS[3:0]			
Reset	–	–	–	–	0xE			
Access Type	–	–	–	–	Write, Read			

ビットフィールド	ビット	説明	デコード
udc_enable	8	イネーブル機能。 StealthChop+モードになっていなくてはならず、Softstopがアクティブ化されている必要があります。 *TSTEP* < *TCOOL_THRS*、*TSTEP* > *THIGH*。	0x0: Disable udcstep 0x1: Enable udcstep
DECEL_THRS	3:0	(*CS_ACTUAL* >= ((*DECEL_THRS* + 1) × *IRUN*)/16) の場合、diag を設定してください。	

STEPS_LOST（0x1A）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	–	–	–	–	STEPS_LOST[19:16]			

Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	STEPS_LOST[15:8]							
Reset								
Access Type	Read Only							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	STEPS_LOST[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
STEPS_LOST	19:0	ストール後または ENC_A 停止の実行後に消失したステップ・パルスの数をカウントします。

エンコーダ・レジスタ

インクリメンタル・エンコーダ・インターフェースに関連するレジスタ。

レジスタの詳細

ENCMODE（0x38）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	BEMF_BLANK_TIME[7:0]							
Reset	0d16							

Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

ビットフィールド	ビット	説明
BEMF_BLANK_TIME	23:16	ブランク時間は *BEMF_BLANK_TIME* × 62.5ns ×((2^{14}) − 1)です。

X_ENC（0x39）

BIT	31	30	29	28	27	26	25	24
Field	X_ENC[31:24]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	X_ENC[23:16]							
Reset	0x00000000							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	X_ENC[15:8]							

Reset	0x00000000							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	X_ENC[7:0]							
Reset	0x00000000							
Access Type	Write, Read							

ビットフィールド	ビット	説明
X_ENC	31:0	実際のエンコーダ・ポジション（符号付き）。

ENC_CONST（0x3A）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	ENC_CONST[31:24]							
Reset	0x00010000							
Access Type	Write, Read							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	ENC_CONST[23:16]							
Reset	0x00010000							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	ENC_CONST[15:8]							
Reset	0x00010000							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**

Field	ENC_CONST[7:0]
Reset	0x00010000
Access Type	Write, Read

ビットフィールド	ビット	説明
ENC_CONST	31:0	加算定数（符号付き） 16ビットの整数部分、16ビットの小数部分。 *X_ENC*で加算される量は以下のとおりです ± *ENC_CONST*/(2^{16} × *X_ENC*)（2進数） または ± *ENC_CONST*/(10^4 × *X_ENC*)（10進数） *ENCMODE*の*enc_sel_decimal*ビットにより10進数設定と2進数設定の間の切り替えが行われます。 回転方向を合わせるには符号を用いてください。 2進数： ± [µsteps/2^{16}] ±(0…32767.999847) 10進数： ±(0.0…32767.9999) リセット後のデフォルト= 1.0（= 65536）

ENC_STATUS（0x3B）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–

Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	–	–	–	–	–	–	deviation_warn	n_event
Reset	–	–	–	–	–	–	0b0	0b0
Access Type	–	–	–	–	–	–	Write 1 to Clear, Read	Write 1 to Clear, Read

ビットフィールド	ビット	説明	デコード
deviation_warn	1		0x0: No warning 0x1: Deviation_warn cannot be cleared while a warning still persists. Set *ENC_DEVIATION* to zero to disable.
n_event	0		0x0: No event 0x1: Event detected. To clear the status bit, write with a 1 bit at the corresponding position.

ENC_LATCH（0x3C）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	ENC_LATCH[31:24]							
Reset	0x00000000							
Access Type	Read Only							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	ENC_LATCH[23:16]							
Reset	0x00000000							
Access Type	Read Only							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	ENC_LATCH[15:8]							
Reset	0x00000000							

Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	ENC_LATCH[7:0]							
Reset	0x00000000							
Access Type	Read Only							

ビットフィールド	ビット	説明
ENC_LATCH	31:0	N イベント時にラッチされたエンコーダ・ポジション *X_ENC*。

ENC_DEVIATION（0x3D）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	ENC_DEVIATION[19:16]			
Reset	–	–	–	–	0x00000			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	ENC_DEVIATION[15:8]							
Reset	0x00000							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	ENC_DEVIATION[7:0]							

Reset	0x00000
Access Type	Write, Read

ビットフィールド	ビット	説明	デコード
ENC_DEVIATION	19:0	これは、偏差警告モニタリングのためのBEMFエンコーダ・ステップの最大カウント数です。この機能を有効化するには0より大きい値を設定します。 モニタリング期間を開始するときにX_ENCが0にセットされます。 結果は、フラグ*ENC_STATUS.deviation_warn*（外部ピンDO0またはDO1に割り当てることができます）で表されます。 0 = この機能をオフ。	0x0: Deviation check = off

StealthChopPlus

すべてのレジスタは、StealthChop+チョッパー・モードに関連します。

レジスタの詳細

CURRENT_PI_REG（0x40）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	CUR_I[9:8]	
Reset	–	–	–	–	–	–	0d10	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	CUR_I[7:0]							
Reset	0d10							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	CUR_P[11:8]			
Reset	–	–	–	–	0d64			

Access Type	–	–	–	–	Write, Read			
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	CUR_P[7:0]							
Reset	0d64							
Access Type	Write, Read							

ビットフィールド	ビット	説明	デコード
CUR_I	25:16	StealthChop+電流レギュレータの I パラメータ。 **ヒント**：この値を 0 に設定すると、積分器の内容もクリアされます。	0x0: Clear current integrator.
CUR_P	11:0	StealthChop+電流レギュレータの P パラメータ。	

ANGLE_PI_REG（0x41）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	–	–	–	–	–	–	ANGLE_I[9:8]	
Reset	–	–	–	–	–	–	0d20	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	ANGLE_I[7:0]							
Reset	0d20							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	ANGLE_P[11:8]			
Reset	–	–	–	–	0d50			
Access Type	–	–	–	–	Write, Read			

BIT	7	6	5	4	3	2	1	0
Field	ANGLE_P[7:0]							
Reset	0d50							
Access Type	Write, Read							

ビットフィールド	ビット	説明	デコード
ANGLE_I	25:16	StealthChop+角度レギュレータの I パラメータ。 I = 0 に設定すると積分器がクリアされます。	0x0: Clear angle PI integrator.
ANGLE_P	11:0	StealthChop+角度レギュレータの P パラメータ。	

CUR_ANGLE_LIMIT（0x42）

BIT	31	30	29	28	27	26	25	24
Field	cur_pi_neg_clip	cur_pi_pos_clip	cur_pi_int_neg_clip	cur_pi_int_pos_clip	CUR_PI_LIMIT[11:8]			
Reset					0xFFF			
Access Type	Read Only	Read Only	Read Only	Read Only	Write, Read			
BIT	23	22	21	20	19	18	17	16
Field	CUR_PI_LIMIT[7:0]							
Reset	0xFFF							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	angle_pi_neg_clip	angle_pi_pos_clip	angle_pi_int_neg_clip	angle_pi_int_pos_clip	–	–	ANGLE_PI_LIMIT[9:8]	
Reset					–	–	0d256	
Access Type	Read Only	Read Only	Read Only	Read Only	–	–	Write, Read	
BIT	7	6	5	4	3	2	1	0

Field	ANGLE_PI_LIMIT[7:0]
Reset	0d256
Access Type	Write, Read

ビットフィールド	ビット	説明	デコード
cur_pi_neg_clip	31	StealthChop+電流 PI レギュレータを 0 まで負方向にクリッピング（電流が過大）。	0x0: No clipping 0x1: Clipping
cur_pi_pos_clip	30	StealthChop+電流 PI レギュレータを *CUR_PI_LIMIT* まで正方向にクリッピング（電流が未達）。	0x0: No clipping 0x1: Clipping
cur_pi_int_neg_clip	29	StealthChop+電流 PI レギュレータの I 部を 0 まで負方向にクリッピング（電流が過大）。	0x0: No clipping 0x1: Clipping
cur_pi_int_pos_clip	28	StealthChop+電流 PI レギュレータの I 部を *CUR_PI_LIMIT* まで正方向にクリッピング（電流が未達）。	0x0: No clipping 0x1: Clipping
CUR_PI_LIMIT	27:16	電流レギュレータの出力の符号なし制限値（*PWM_CALC*を制限することによりPWMのデューティサイクルを制限）。 制限しない場合は 4095 に設定します（標準設定）。	
angle_pi_neg_clip	15	角度 PI レギュレータを*-ANGLE_PI_LIMIT* まで負方向にクリッピング。	0x0: No clipping 0x1: Clipping
angle_pi_pos_clip	14	角度 PI レギュレータを *ANGLE_PI_LIMIT* まで正方向にクリッピング。	0x0: No clipping 0x1: Clipping
angle_pi_int_neg_clip	13	角度 PI レギュレータの I 部を*-ANGLE_PI_LIMIT* まで負方向にクリッピング。	0x0: No clipping 0x1: Clipping
angle_pi_int_pos_clip	12	角度 PI レギュレータの I 部を *ANGLE_PI_LIMIT* まで正方向にクリッピング。	0x0: No clipping 0x1: Clipping
ANGLE_PI_LIMIT	9:0	角度コントローラの出力を±ANGLE_PI_LIMIT に符号なしで制限（256 のデフォルト設定では±90°にクリップ。これは、通常動作には十分であり、次の電流周期へオーバーフロー／アンダーフローする原因にはなりません）。	

ANGLE_LOWER_LIMIT（0x43）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	ANGLE_ERROR[9:8]	
Reset	–	–	–	–	–	–		
Access Type	–	–	–	–	–	–	Read Only	

BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	ANGLE_ERROR[7:0]							
Reset								
Access Type	Read Only							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	ANGLE_LOWER_I_LIMIT[9:8]	
Reset	–	–	–	–	–	–	0x100	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	ANGLE_LOWER_I_LIMIT[7:0]							
Reset	0x100							
Access Type	Write, Read							

ビットフィールド	ビット	説明
ANGLE_ERROR	25:16	*MSCNT − ANGLE_MEAS* によって計算される電流角度の実際の誤差。
ANGLE_LOWER_I_LIMIT	9:0	角度レギュレータの動作のための下限電流値。*AMPL_MEAS <= ANGLE_LOWER_I_LIMIT*の場合、電流測定ノイズによってレギュレーション・ノイズが生じるのを防ぐため、角度レギュレータはディスエーブルされます。 **注意**：制限値を設定する場合、*COOL_CUR_DIV* の選択に従い、*IRUN* を低減することを検討してください。CoolStep+の電流低減によって電流がこの制限値未満に低下すると、角度レギュレータをディスエーブルする原因になることがあります。それによって、モーターの安定性が損なわれる可能性があります。

CUR_ANGLE_MEAS（0x44）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	ANGLE_MEAS[9:8]	
Reset	–	–	–	–	–	–		
Access Type	–	–	–	–	–	–	Read Only	

BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	ANGLE_MEAS[7:0]							
Reset								
Access Type	Read Only							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	AMPL_MEAS[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	AMPL_MEAS[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
ANGLE_MEAS	25:16	ADC の値から計算した（符号付きの）電流角度（±256 = ±90°）。
AMPL_MEAS	11:0	*ADC_I_A*と*ADC_I_B*から計算した実際の電流振幅。この値は振幅レギュレータに与えられます。 電流スケールが 250 の場合、振幅は 2000 の値に安定化されます。*AMPL_MEAS* は、これより高い範囲をカバーし、PI レギュレータのより高速の応答を可能にします。

PI_RESULTS（0x45）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	ANGLE_CORR_CALC[9:8]	
Reset	–	–	–	–	–	–		
Access Type	–	–	–	–	–	–	Read Only	

BIT	23	22	21	20	19	18	17	16
Field	ANGLE_CORR_CALC[7:0]							
Reset								
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	PWM_CALC[12:8]				
Reset	–	–	–					
Access Type	–	–	–	Read Only				
BIT	7	6	5	4	3	2	1	0
Field	PWM_CALC[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
ANGLE_CORR_CALC	25:16	角度レギュレータの出力。
PWM_CALC	12:0	PWM 振幅 0～4095 の変調に対する電流 PI レギュレータの出力。

COIL_INDUCT（0x46）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	rcoil_thermal_coupling	rcoil_manual
Reset	–	–	–	–	–	–		

Access Type	–	–	–	–	–	–	Write, Read	Write, Read
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	COIL_INDUCT[14:8]						
Reset	–							
Access Type	–	Write, Read						
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	COIL_INDUCT[7:0]							
Reset								
Access Type	Write, Read							

ビットフィールド	ビット	説明	デコード
rcoil_thermal_coupling	17	停止状態時のコイル抵抗測定用に熱的な結合をイネーブルします。一方のコイル抵抗が測定可能な場合（目標電流値が電流設定値の 50%を超え、実際に測定された電流が電流設定値の 25%を超えている）、実際の電流が *IRUN* の 50%未満であれば、測定可能なコイル抵抗の変化に対応して、他方のコイル抵抗値が調整されます。	0x0: No thermal coupling 0x1: Thermal coupling
rcoil_manual	16	StallGuard+および CoolStep+でのコイル抵抗のソースの選択。	0x0: Automatic resistance measurement is used. 0x1: Resistance from user value is used.
COIL_INDUCT	14:0	StallGuard+および CoolStep+を使用する場合にモーターのインダクタンスを µH 単位で入力します。	

R_COIL（0x47）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	R_COIL_AUTO_A[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	R_COIL_AUTO_A[7:0]							

Reset								
Access Type	Read Only							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	R_COIL_AUTO_B[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	R_COIL_AUTO_B[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
R_COIL_AUTO_A	27:16	コイル A について測定したコイル抵抗。*TSTEP* > *T_RCOIL_MEAS* の場合に測定されます。測定は、フラグ *rcoil_manual* とは無関係です。コイル抵抗は、実際の目標コイル電流が *IRUN* の 50%を超え、測定された電流が *IRUN* の 25%を超えている場合にのみ、更新されます。
R_COIL_AUTO_B	11:0	コイル B について測定したコイル抵抗。*TSTEP* > *T_RCOIL_MEAS* の場合に測定されます。測定は、フラグ *rcoil_manual* とは無関係です。コイル抵抗は、実際の目標コイル電流が *IRUN* の 50%を超え、測定された電流が *IRUN* の 25%を超えている場合にのみ、更新されます。

R_COIL_USER（0x48）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	–	–	–	–	R_COIL_USER_A[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Write, Read			
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	R_COIL_USER_A[7:0]							

Reset								
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	R_COIL_USER_B[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Write, Read			
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	R_COIL_USER_B[7:0]							
Reset								
Access Type	Write, Read							

ビットフィールド	ビット	説明
R_COIL_USER_A	27:16	コイル抵抗のオプションのユーザ値。温度に依存したコイル抵抗に対しソフトウェア・モデルを使用できます。使用するには *rcoil_manual* をセットします。
R_COIL_USER_B	11:0	コイル抵抗のオプションのユーザ値。温度に依存したコイル抵抗に対しソフトウェア・モデルを使用できます。使用するには *rcoil_manual* をセットします。

SGP_CONF（0x49）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	–	–	SGP_LOW_VEL_CNTS[1:0]		–	–	–	–
Reset	–	–			–	–	–	–
Access Type	–	–	Write, Read		–	–	–	–
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	SGP_LOW_VEL_SLOPE[7:0]							
Reset								
Access Type	Write, Read							

BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	–	–	–	SGP_THRS[8]
Reset	–	–	–	–	–	–	–	
Access Type	–	–	–	–	–	–	–	Write, Read
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	SGP_THRS[7:0]							
Reset								
Access Type	Write, Read							

ビットフィールド	ビット	説明	デコード
SGP_LOW_VEL_CNTS	29:28	負の勾配が低速ストール検出に必要な *SGP_LOW_VEL_SLOPE* より大きい、連続する *SGP_RESULT* 測定の数。	0x0: 1 event 0x1: 2 events 0x2: 3 events 0x3: 4 events
SGP_LOW_VEL_SLOPE	23:16	ストール検出のための SGP_RESULT の速度勾配の下限。	
SGP_THRS	8:0	*SGP_RESULT* に基づくストール検出を行う場合の StallGuard のストール・スレッショルド。	

SGP_IND_2_3（0x4A）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	–	–	–	–	–	–	SGP_IND_3[9:8]	
Reset	–	–	–	–	–	–		
Access Type	–	–	–	–	–	–	Read Only	
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	SGP_IND_3[7:0]							
Reset								

Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	SGP_IND_2[9:8]	
Reset	–	–	–	–	–	–	0b0000000000	
Access Type	–	–	–	–	–	–	Read Only	
BIT	7	6	5	4	3	2	1	0
Field	SGP_IND_2[7:0]							
Reset	0b0000000000							
Access Type	Read Only							

ビットフィールド	ビット	説明
SGP_IND_3	25:16	MSCNT のインデックスの範囲が 768～1023 での各 StallGuard+測定値。
SGP_IND_2	9:0	MSCNT のインデックスの範囲が 512～767 での各 StallGuard+測定値。

SGP_IND_0_1（0x4B）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	SGP_IND_1[9:8]	
Reset	–	–	–	–	–	–		
Access Type	–	–	–	–	–	–	Read Only	
BIT	23	22	21	20	19	18	17	16
Field	SGP_IND_1[7:0]							
Reset								
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8

Field	–	–	–	–	–	–	SGP_IND_0[9:8]	
Reset	–	–	–	–	–	–	0x00	
Access Type	–	–	–	–	–	–	Read Only	
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	SGP_IND_0[7:0]							
Reset	0x00							
Access Type	Read Only							

ビットフィールド	ビット	説明
SGP_IND_1	25:16	MSCNT のインデックスの範囲が 256～511 での各 StallGuard+測定値。
SGP_IND_0	9:0	MSCNT のインデックスの範囲が 0～255 での各 StallGuard+測定値。

INDUCTANCE_VOLTAGE（0x4C）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	–	–	–	–	UL_A[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	UL_A[7:0]							
Reset								
Access Type	Read Only							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	UL_B[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			

BIT	7	6	5	4	3	2	1	0
Field	UL_B[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
UL_A	27:16	情報提供用の内部計算結果：コイル・インダクタンスおよびモーター電圧から計算し、VS レンジにスケーリングした（符号付き 12 ビットにクリップした符号付き 14 ビット、つまり、フル・スケール = 電圧の 25%）、コイル・インダクタンスでの電圧降下。
UL_B	11:0	情報提供用の内部計算結果：コイル・インダクタンスおよびモーター電圧から計算し、VS レンジにスケーリングした（符号付き 12 ビットにクリップした符号付き 14 ビット、つまり、フル・スケール = 電圧の 25%）、コイル・インダクタンスでの電圧降下。

SGP_BEMF（0x4D）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	UBEMF_ABS[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			

BIT	23	22	21	20	19	18	17	16
Field	UBEMF_ABS[7:0]							
Reset								
Access Type	Read Only							

BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	SGP_RAW[9:8]	
Reset	–	–	–	–	–	–		
Access Type	–	–	–	–	–	–	Read Only	

BIT	7	6	5	4	3	2	1	0
Field	SGP_RAW[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
UBEMF_ABS	27:16	情報提供用の内部計算結果：計算したバック EMF の振幅。
SGP_RAW	9:0	情報提供用の内部計算結果：StallGuard+のフィルタ処理していない負荷余裕。

COOLSTEPPLUS_CONF（0x4E）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	load_filt_en	–	–	–	–
Reset	–	–	–	0x1	–	–	–	–

Access Type	–	–	–	Write, Read	–	–	–	–

ビットフィールド	ビット	説明	デコード
load_filt_en	4	coolstep 入力をフィルタ処理するために SG5_RAW の値の 16 個の測定値に対する RC フィルタをイネーブル。	0x0: Disable 0x1: Enable (recommended)

COOLSTEPPLUS_PI_REG（0x4F）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	–	–	–	–	–	–	COOLSTEP_I[9:8]	
Reset	–	–	–	–	–	–	0d16	
Access Type	–	–	–	–	–	–	Write, Read	
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	COOLSTEP_I[7:0]							
Reset	0d16							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	–	–	–	–	COOLSTEP_P[11:8]			
Reset	–	–	–	–	0d128			
Access Type	–	–	–	–	Write, Read			
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	COOLSTEP_P[7:0]							
Reset	0d128							
Access Type	Write, Read							

ビットフィールド	ビット	説明
COOLSTEP_I	25:16	CoolStepPlus 目標電流レギュレータ用の I パラメータ。
COOLSTEP_P	11:0	CoolStepPlus 目標電流レギュレータ用の P パラメータ。

COOLSTEPPLUS_PI_DOWN（0x50）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	COOL_PI_OFF_SPEED[11:8]			
Reset	–	–	–	–	0d64			
Access Type	–	–	–	–	Write, Read			
BIT	23	22	21	20	19	18	17	16
Field	COOL_PI_OFF_SPEED[7:0]							
Reset	0d64							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	COOL_PI_DOWN_LIMIT[11:8]			
Reset	–	–	–	–	0d128			
Access Type	–	–	–	–	Write, Read			
BIT	7	6	5	4	3	2	1	0
Field	COOL_PI_DOWN_LIMIT[7:0]							
Reset	0d128							
Access Type	Write, Read							

ビットフィールド	ビット	説明
COOL_PI_OFF_SPEED	27:16	CoolStepPlus がオフになっている場合の電流勾配。*COOL_PI_OFF_SPEED* はレギュレーションの誤差をシミュレーションし、それによって電流が低減します。
COOL_PI_DOWN_LIMIT	11:0	CoolStepPlus の目標電流の立下がり勾配を制限します。最大の低減量は、チョッパーの 32 周期あたり、1/256 *COOL_PI_DOWN_LIMIT* です。

COOLSTEPPLUS_RESERVE_CONF（0x51）

BIT	31	30	29	28	27	26	25	24
Field	COOL_HI_GENERATORIC_RESERVE[7:0]							
Reset	0d100							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	COOL_LOW_GENERATORIC_RESERVE[7:0]							
Reset	0d220							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	COOL_HI_LOAD_RESERVE[7:0]							
Reset	0d50							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	COOL_LOW_LOAD_RESERVE[7:0]							
Reset	0d150							
Access Type	Write, Read							

ビットフィールド	ビット	説明
COOL_HI_GENERATORIC_RESERVE	31:24	CoolStep 電流が *IRUN* に等しい場合にジェネレータ・モードで用いられる負荷余裕。レギュレーションを最大限に安定化するには、*COOL_HI_GENERATORIC_RESERVE* < *COOL_LOW_GENERATORIC_RESERVE* となるようにします。
COOL_LOW_GENERATORIC_RESERVE	23:16	CoolStep 電流が最小レベルの場合にジェネレータ・モードで用いられる負荷余裕。レギュレーションを最大限に安定化するには、*COOL_HI_GENERATORIC_RESERVE* < *COOL_LOW_GENERATORIC_RESERVE* となるようにします。
COOL_HI_LOAD_RESERVE	15:8	CoolStep 電流が *IRUN* に等しい場合に用いられる負荷余裕。レギュレーションを最大限に安定化するには、*COOL_HI_LOAD_RESERVE* < *COOL_LOW_LOAD_RESERVE* となるようにします。
COOL_LOW_LOAD_RESERVE	7:0	最小 CoolStep 電流で用いられる負荷余裕。レギュレーションを最大限に安定化するには、*COOL_HI_LOAD_RESERVE* < *COOL_LOW_LOAD_RESERVE* となるようにします。

COOLSTEPPLUS_LOAD_RESERVE（0x52）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	COOLSTEP_LOAD_RESERVE[8]
Reset	–	–	–	–	–	–	–	
Access Type	–	–	–	–	–	–	–	Read Only
BIT	23	22	21	20	19	18	17	16
Field	COOLSTEP_LOAD_RESERVE[7:0]							
Reset								
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	SGP_RESULT[9:8]	
Reset	–	–	–	–	–	–		
Access Type	–	–	–	–	–	–	Read Only	
BIT	7	6	5	4	3	2	1	0
Field	SGP_RESULT[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
COOLSTEP_LOAD_RESERVE	24:16	CoolStepPlus が現在用いられている負荷余裕。
SGP_RESULT	9:0	最後または最後の4つのフルステップ・セグメントでフィルタ処理された*SGP_RAW*の値。 *sgp_filt_en* = 0 : 最後のフルステップ・セグメントにわたって平均化されたSGP_RAWの値。 *sgp_filt_en* = 1 : 最後の 4 つのフルステップ・セグメントにわたって平均化された *SGP_RAW* の値。

TSTEP Velocity（0x53）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	TSTEP_VELOCITY[22:16]						
Reset	–							
Access Type	–	Read Only						
BIT	15	14	13	12	11	10	9	8
Field	TSTEP_VELOCITY[15:8]							
Reset								
Access Type	Read Only							
BIT	7	6	5	4	3	2	1	0
Field	TSTEP_VELOCITY[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
TSTEP_VELOCITY	22:0	TSTEP の値から導かれた実際の速度（STEP および DIR 動作の StallGuardPlus に使用）。

ADC レジスタ

内部 ADC に関連するレジスタ。

レジスタの詳細

ADC_VSUPPLY_TEMP（0x58）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	ADC_TEMP[8]
Reset	–	–	–	–	–	–	–	0b0000000000000
Access Type	–	–	–	–	–	–	–	Read Only
BIT	23	22	21	20	19	18	17	16
Field	ADC_TEMP[7:0]							
Reset	0b0000000000000							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	ADC_VSUPPLY[8]
Reset	–	–	–	–	–	–	–	0b0000000000000
Access Type	–	–	–	–	–	–	–	Read Only
BIT	7	6	5	4	3	2	1	0
Field	ADC_VSUPPLY[7:0]							
Reset	0b0000000000000							
Access Type	Read Only							

ビットフィールド	ビット	説明
ADC_TEMP	24:16	温度[°C] = (1.042 × ADC_TEMP − 264.6)°C
ADC_VSUPPLY	8:0	V_S [V] = 0,1409V × ADC_VSUPPLY

ADC_I（0x59）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	ADC_I_B[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			
BIT	23	22	21	20	19	18	17	16
Field	ADC_I_B[7:0]							
Reset								
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	ADC_I_A[11:8]			
Reset	–	–	–	–				
Access Type	–	–	–	–	Read Only			
BIT	7	6	5	4	3	2	1	0
Field	ADC_I_A[7:0]							
Reset								
Access Type	Read Only							

ビットフィールド	ビット	説明
ADC_I_B	27:16	StealthChopPlus 動作で測定されたコイル B の一時的な電流。
ADC_I_A	11:0	StealthChopPlus 動作で測定されたコイル A の一時的な電流。

OTW_OV_VTH（0x5A）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	OVERTEMPPREWARNING_VTH[8]
Reset	–	–	–	–	–	–	–	0x1FF
Access Type	–	–	–	–	–	–	–	Write, Read
BIT	23	22	21	20	19	18	17	16
Field	OVERTEMPPREWARNING_VTH[7:0]							
Reset	0x1FF							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	OVERVOLTAGE_VTH[8]
Reset	–	–	–	–	–	–	–	0x1FF
Access Type	–	–	–	–	–	–	–	Write, Read
BIT	7	6	5	4	3	2	1	0
Field	OVERVOLTAGE_VTH[7:0]							
Reset	0x1FF							
Access Type	Write, Read							

ビットフィールド	ビット	説明
OVERTEMPPREWARNING_VTH	24:16	温度事前警告制限値。過熱時に警告する必要性に応じて設定します。*ADC_TEMP* が *OVERTEMPPREWARNING_VTH* 以上の場合、*DRV_STATUS* の *otpw* が 1 に設定されます。
OVERVOLTAGE_VTH	8:0	システム環境を一致させるための過電圧スレッショルドの選択。*ADC_VSUPPLY* が *OVERVOLTAGE_VTH* 以上の場合、*DRV_STATUS* の *ov* が 1 に設定されます。必要に応じ、このフラグを診断出力ピン（DO0 または DO1）に出力します。

モーター・ドライバ・レジスタ

モーター・ドライバ・ユニットに関連するレジスタ。

レジスタの詳細

MSLUT_0（0x60）

マイクロステップ・テーブルの項目 0～31

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_0[31:24]							
Reset	0xAAAAB554							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_0[23:16]							
Reset	0xAAAAB554							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_0[15:8]							
Reset	0xAAAAB554							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_0[7:0]							
Reset	0xAAAAB554							
Access Type	Write, Read							

ビットフィールド	ビット	説明
MSLUT_0	31:0	各ビットは、対応する*MSLUTSEL*の*W*ビットと組み合わせた場合のエントリxとエントリx + 1の差を示します。 0: *W* = %00: −1 %01: +0 %10: +1 %11: +2

ビットフィールド	ビット	説明
		1: *W* = %00: +0 %01: +1 %10: +2 %11: +3 これは、波形の最初の1/4について差分をコード化したものです。*CUR_A*と*CUR_B*の開始値は、*START_SIN*と*START_SIN90*の*MSCNT*のポジション0に対して保存されています。 *ofs31, ofs30, …, ofs01, ofs00* … *ofs255, ofs254, …, ofs225, ofs224.* リセット後のデフォルト = サイン波テーブル

MSLUT_1（0x61）

マイクロステップ・テーブルの項目 32～63

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_1[31:24]							
Reset	0x4A9554AA							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_1[23:16]							
Reset	0x4A9554AA							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_1[15:8]							
Reset	0x4A9554AA							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_1[7:0]							

Reset	0x4A9554AA
Access Type	Write, Read

ビットフィールド	ビット	説明
MSLUT_1	31:0	各ビットは、対応する*MSLUTSEL*の*W*ビットと組み合わせた場合の項目xと項目x+1の差を示します。 0: *W* = %00: −1 %01: +0 %10: +1 %11: +2 1: *W* = %00: +0 %01: +1 %10: +2 %11: +3 これは、波形の最初の1/4について差分をコード化したものです。*CUR_A*と*CUR_B*の開始値は、*START_SIN*と*START_SIN90*の*MSCNT*のポジション0に対して保存されています。 *ofs31, ofs30, …, ofs01, ofs00* … *ofs255, ofs254, …, ofs225, ofs224.* リセット後のデフォルト = サイン波テーブル

MSLUT_2（0x62）

マイクロステップ・テーブルの項目 64～95

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_2[31:24]							
Reset	0x24492929							
Access Type	Write, Read							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	MSLUT_2[23:16]							
Reset	0x24492929							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	MSLUT_2[15:8]							

Reset	0x24492929							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	MSLUT_2[7:0]							
Reset	0x24492929							
Access Type	Write, Read							

ビットフィールド	ビット	説明
MSLUT_2	31:0	各ビットは、対応するMSLUTSELのWビットと組み合わせた場合の項目xと項目x+1の差を示します。 0: *W* = %00: −1 %01: +0 %10: +1 %11: +2 1: *W* = %00: +0 %01: +1 %10: +2 %11: +3 これは、波形の最初の1/4について差分をコード化したものです。*CUR_A*と*CUR_B*の開始値は、*START_SIN*と*START_SIN90*の*MSCNT*のポジション0に対して保存されています。 *ofs31, ofs30, …, ofs01, ofs00* … *ofs255, ofs254, …, ofs225, ofs224.* リセット後のデフォルト = サイン波テーブル

MSLUT_3（0x63）

マイクロステップ・テーブルの項目 96～127

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_3[31:24]							
Reset	0x10104222							
Access Type	Write, Read							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	MSLUT_3[23:16]							

Reset	0x10104222							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	MSLUT_3[15:8]							
Reset	0x10104222							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	MSLUT_3[7:0]							
Reset	0x10104222							
Access Type	Write, Read							

ビットフィールド	ビット	説明
MSLUT_3	31:0	各ビットは、対応する*MSLUTSEL*の*W*ビットと組み合わせた場合の項目xと項目x+1の差を示します。 0: *W* = %00: −1 %01: +0 %10: +1 %11: +2 1: *W* = %00: +0 %01: +1 %10: +2 %11: +3 これは、波形の最初の 1/4 について差分をコード化したものです。*CUR_A* と *CUR_B* の開始値は、*START_SIN* と *START_SIN90* の *MSCNT* のポジション 0 に対して保存されています。 *ofs31, ofs30, …, ofs01, ofs00* … *ofs255, ofs254, …, ofs225, ofs224.* リセット後のデフォルト = サイン波テーブル

MSLUT_4（0x64）

マイクロステップ・テーブルの項目 128～159

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_4[31:24]							

Reset	0xFBFFFFFF							
Access Type	Write, Read							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	MSLUT_4[23:16]							
Reset	0xFBFFFFFF							
Access Type	Write, Read							
BIT	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Field	MSLUT_4[15:8]							
Reset	0xFBFFFFFF							
Access Type	Write, Read							
BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	MSLUT_4[7:0]							
Reset	0xFBFFFFFF							
Access Type	Write, Read							

ビットフィールド	ビット	説明
MSLUT_4	31:0	各ビットは、対応する*MSLUTSEL*の*W*ビットと組み合わせた場合の項目xと項目x+1の差を示します。 0: *W* = %00: −1 %01: +0 %10: +1 %11: +2 1: *W* = %00: +0 %01: +1 %10: +2 %11: +3 これは、波形の最初の1/4について差分をコード化したものです。*CUR_A*と*CUR_B*の開始値は、*START_SIN*と*START_SIN90*の*MSCNT*のポジション0に対して保存されています。 *ofs31, ofs30, …, ofs01, ofs00* … *ofs255, ofs254, …, ofs225, ofs224.* リセット後のデフォルト = サイン波テーブル

MSLUT_5（0x65）

マイクロステップ・テーブルの項目 160～191

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_5[31:24]							
Reset	0xB5BB777D							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_5[23:16]							
Reset	0xB5BB777D							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_5[15:8]							
Reset	0xB5BB777D							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_5[7:0]							
Reset	0xB5BB777D							
Access Type	Write, Read							

ビットフィールド	ビット	説明
MSLUT_5	31:0	各ビットは、対応する*MSLUTSEL*の*W*ビットと組み合わせた場合の項目xと項目x+1の差を示します。 0: *W* = %00: -1 %01: +0 %10: +1 %11: +2 1: *W* = %00: +0 %01: +1 %10: +2 %11: +3

ビットフィールド	ビット	説明
		これは、波形の最初の1/4について差分をコード化したものです。*CUR_A*と*CUR_B*の開始値は、*START_SIN*と*START_SIN90*の*MSCNT*のポジション0に対して保存されています。 *ofs31, ofs30, …, ofs01, ofs00* … *ofs255, ofs254, …, ofs225, ofs224.* リセット後のデフォルト = サイン波テーブル

MSLUT_6（0x66）

マイクロステップ・テーブルの項目 192～223

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_6[31:24]							
Reset	0x49295556							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_6[23:16]							
Reset	0x49295556							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_6[15:8]							
Reset	0x49295556							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	MSLUT_6[7:0]							
Reset	0x49295556							
Access Type	Write, Read							

ビットフィールド	ビット	説明
MSLUT_6	31:0	各ビットは、対応するMSLUTSELのWビットと組み合わせた場合の項目xと項目x+1の差を示します。 0: *W* = %00: −1 %01: +0 %10: +1 %11: +2 1: *W* = %00: +0 %01: +1 %10: +2 %11: +3 これは、波形の最初の1/4について差分をコード化したものです。*CUR_A*と*CUR_B*の開始値は、*START_SIN*と*START_SIN90*のMSCNTのポジション0に対して保存されています。 *ofs31, ofs30, …, ofs01, ofs00* … *ofs255, ofs254, …, ofs225, ofs224.* リセット後のデフォルト = サイン波テーブル

MSLUT_7（0x67）

マイクロステップ・テーブルの項目 224～255

BIT	31	30	29	28	27	26	25	24
Field	MSLUT_7[31:24]							
Reset	0x00404222							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	MSLUT_7[23:16]							
Reset	0x00404222							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	MSLUT_7[15:8]							
Reset	0x00404222							
Access Type	Write, Read							

BIT	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Field	MSLUT_7[7:0]							
Reset	0x00404222							
Access Type	Write, Read							

ビットフィールド	ビット	説明
MSLUT_7	31:0	各ビットは、対応する*MSLUTSEL*の*W*ビットと組み合わせた場合の項目xと項目x+1の差を示します。 0: *W* = %00: −1 %01: +0 %10: +1 %11: +2 1: *W* = %00: +0 %01: +1 %10: +2 %11: +3 これは、波形の最初の1/4について差分をコード化したものです。*CUR_A*と*CUR_B*の開始値は、*START_SIN*と*START_SIN90*の*MSCNT*のポジション0に対して保存されています。 *ofs31, ofs30, …, ofs01, ofs00* … *ofs255, ofs254, …, ofs225, ofs224.* リセット後のデフォルト = サイン波テーブル

MSLUTSEL（0x68）

BIT	**31**	**30**	**29**	**28**	**27**	**26**	**25**	**24**
Field	X3[7:0]							
Reset	0xFF							
Access Type	Write, Read							
BIT	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**
Field	X2[7:0]							
Reset	0xFF							
Access Type	Write, Read							

BIT	15	14	13	12	11	10	9	8
Field	X1[7:0]							
Reset	0x80							
Access Type	Write, Read							
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

ビットフィールド	ビット	説明
X3	31:24	LUTのセグメント3の開始。 サイン波のルックアップ・テーブルは、個別のステップ幅制御エントリ*Wx*を用いて、最大4つのセグメントに分割できます。セグメントの境界は、*X1*、*X2*、*X3*で選択できます。 セグメント0は0～*X1*−1の範囲です。 セグメント1は*X1*～*X2*−1の範囲です。 セグメント2は*X2*～*X3*−1の範囲です。 セグメント3は*X3*～255の範囲です。 定義された応答をするには、各値が以下を満たすことが必要です。 0<*X1*<*X2*<*X3*
X2	23:16	LUTのセグメント2の開始。 サイン波のルックアップ・テーブルは、個別のステップ幅制御エントリ*Wx*を用いて、最大4つのセグメントに分割できます。セグメントの境界は、*X1*、*X2*、*X3*で選択できます。 セグメント0は0～*X1*−1の範囲です。 セグメント1は*X1*～*X2*−1の範囲です。 セグメント2は*X2*～*X3*−1の範囲です。 セグメント3は*X3*～255の範囲です。 定義された応答をするには、各値が以下を満たすことが必要です。 0<*X1*<*X2*<*X3*
X1	15:8	LUTのセグメント1の開始。 サイン波のルックアップ・テーブルは、個別のステップ幅制御エントリ*Wx*を用いて、最大4つのセグメントに分割できます。セグメントの境界は、*X1*、*X2*、*X3*で選択できます。 セグメント0は0～*X1*−1の範囲です。 セグメント1は*X1*～*X2*−1の範囲です。 セグメント2は*X2*～*X3*−1の範囲です。 セグメント3は*X3*～255の範囲です。 定義された応答をするには、各値が以下を満たすことが必要です。 0<*X1*<*X2*<*X3*

MSLUTSTART（0x69）

リファレンス・ポジション MSCNT = 0 を通過した場合、常に開始値がマイクロステップ・レジスタ CUR_A および CUR_B に転送されます。

BIT	31	30	29	28	27	26	25	24
Field	OFFSET_SIN90[7:0]							
Reset	0x00							
Access Type	Write, Read							
BIT	23	22	21	20	19	18	17	16
Field	START_SIN90[7:0]							
Reset	0xF7							
Access Type	Write, Read							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	START_SIN[7:0]							
Reset	0x00							
Access Type	Write, Read							

ビットフィールド	ビット	説明
OFFSET_SIN90	31:24	コサイン波に対する符号付きオフセット（±127 マイクロステップ）。*MSCNT* = 0 のポジションでマイクロステップ波形テーブルに適合するよう、*START_SIN90* を修正します。
START_SIN90	23:16	*START_SIN90* は、*MSCNT* = 0（テーブル・ポジション 256 + *OFFSET_SIN90*）のポジションでのコサイン波マイクロステップ・テーブル項目に絶対値を与えます。
START_SIN	7:0	*START_SIN* は、マイクロステップ・テーブル項目 0 に絶対値を与えます。

MSCNT（0x6A）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	MSCNT[9:8]	
Reset	–	–	–	–	–	–	0b0000000000	
Access Type	–	–	–	–	–	–	Read Only	
BIT	7	6	5	4	3	2	1	0
Field	MSCNT[7:0]							
Reset	0b0000000000							
Access Type	Read Only							

ビットフィールド	ビット	説明
MSCNT	9:0	マイクロステップ・カウンタ。実際のポジションを*CUR_A*のマイクロステップ・テーブルに示します。*CUR_B*は256のオフセットを使用します（2相モーター）。 **注：***MSLUTSTART* または *MSLUT* および *MSLUTSEL* を再初期化する前に、*MSCNT* がゼロとなるポジションに移動させてください。

MSCURACT（0x6B）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	CUR_A[8]
Reset	–	–	–	–	–	–	–	0b011110111
Access Type	–	–	–	–	–	–	–	Read Only
BIT	23	22	21	20	19	18	17	16
Field	CUR_A[7:0]							
Reset	0b011110111							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	CUR_B[8]
Reset	–	–	–	–	–	–	–	0b000000000
Access Type	–	–	–	–	–	–	–	Read Only
BIT	7	6	5	4	3	2	1	0
Field	CUR_B[7:0]							
Reset	0b000000000							
Access Type	Read Only							

ビットフィールド	ビット	説明
CUR_A	24:16	MSLUT から読み出した、モーター位相 A（コサイン波）の実際のマイクロステップ電流（電流によるスケーリングなし）。
CUR_B	8:0	MSLUT から読み出した、モーター位相 B（サイン波）の実際のマイクロステップ電流（電流によるスケーリングなし）。

CHOPCONF（0x6C）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	TPFD[3:0]				–	–	–	–
Reset	0x4				–	–	–	–
Access Type	Write, Read				–	–	–	–
BIT	15	14	13	12	11	10	9	8
Field	–	chm	–	–	fd3	HEND_OFFSET[3:1]		
Reset	–	0b0	–	–	0b0	0x2		
Access Type	–	Write, Read	–	–	Write, Read	Write, Read		
BIT	7	6	5	4	3	2	1	0
Field	HEND_OFFSET[0]	HSTRT_TFD210[2:0]			–	–	–	–
Reset	0x2	0b101			–	–	–	–
Access Type	Write, Read	Write, Read			–	–	–	–

ビットフィールド	ビット	説明	デコード
TPFD	23:20	受動高速減衰時間。	

ビットフィールド	ビット	説明	デコード
		*TPFD*を用いることで、モーターの中間領域での共振の緩和が可能です。 受動高速減衰時間の設定により、ブリッジの極性変更後に挿入される高速減衰フェーズの時間が制御されます。 N_{CLK} = 8µs × *TPFD* %0000：ディスエーブル %0001 … %1111: 1 … 15	
chm	14	チョッパー・モード	0x0: Standard mode (SpreadCycle) 0x1: Constant off-time with fast decay time. Fast decay time is also terminated when the negative nominal current is reached. Fast decay is after on-time.
fd3	11	TFD[3] *chm* = 1の場合： 高速減衰時間設定 *TFD* の MSB	
HEND_OFFSET	10:7	*chm* = 0の場合：*HEND*（ヒステリシスの低値）。 %0000 … %1111： ヒステリシスは−3、−2、−1、0、1、…、12。 （この設定値の1/512が電流設定値に加算されます） これは、ヒステリシス・チョッパーで用いられるヒステリシス値です。 *chm* = 1の場合：*OFFSET*（サイン波オフセット）。 %0000 … %1111: オフセットは−3、−2、−1、0、1、…、12。 これはサイン波のオフセットで、この値の 1/512 が各サイン波エントリの絶対値に加算されます。	
HSTRT_TFD210	6:4	*chm* = 0の場合：*HSTRT*（*HEND*に加算されるヒステリシス開始値）。 %000 … %111： ヒステリシスの低値HENDに1、2、…、8が加算されます。（この設定値の1/512が電流設定値に加算されます） **注意：**実効*HEND* + *HSTRT* ≤ 16 **注：**ヒステリシスのデクリメントは16クロックごとに行われます。 *chm* = 1の場合：*TFD*[2..0]（高速減衰時間設定）。 高速減衰時間設定（MSB：*fd3*）： %0000 … %1111： 下記条件での高速減衰時間設定TFD N_{CLK} = 32 × *TFD* の高速減衰時間設定値 *TFD* （%0000：低速減衰のみ）	

COOLCONF（0x6D）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	sgt[6:0]						
Reset	–	0b0000000						
Access Type	–	Write, Read						
BIT	15	14	13	12	11	10	9	8
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

ビットフィールド	ビット	説明
sgt	22:16	StallGuard2のスレッショルド値。 この符号付きの値は、ストール信号出力の場合はStallGuard2レベルを制御し、読出しの場合は最適な測定範囲を設定します。値を小さくすると感度が高くなります。ゼロはほとんどのモーターで使用できる開始値です。 −64～+63：値を大きくすると StallGuard2 の感度は低下し、ストールを示すために必要なトルクが大きくなります。

DRV_STATUS（0x6F）

BIT	31	30	29	28	27	26	25	24
Field	–	olb	ola	s2gb	s2ga	–	ot	stallguard
Reset	–	0b0	0b0	0b0	0b0	–	0b0	0b0
Access Type	–	Read Only	Read Only	Read Only	Read Only	–	Read Only	Read Only
BIT	23	22	21	20	19	18	17	16
Field	CS_ACTUAL[7:0]							
Reset	0b00000							
Access Type	Read Only							
BIT	15	14	13	12	11	10	9	8
Field	–	stealth	s2vsb	s2vsa	–	–	SG_RESULT[9:8]	
Reset	–	0b0	0b0	0b0	–	–	0b0000000000	
Access Type	–	Read Only	Read Only	Read Only	–	–	Read Only	
BIT	7	6	5	4	3	2	1	0
Field	SG_RESULT[7:0]							
Reset	0b0000000000							
Access Type	Read Only							

ビットフィールド	ビット	説明	デコード
olb	30	位相 B の開放負荷インジケータ。	0x0: Normal operation 0x1: Open load detected on phase B. **Note:** This is just an informative flag. The driver takes no action upon it. False detection may occur in fast motion and standstill. Check during slow motion only.
ola	29	位相 A の開放負荷インジケータ。	0x0: Normal operation 0x1: Open load detected on phase A. **Hint:** This is just an informative flag. The driver takes no action upon it. False detection may occur in fast motion and standstill. Check during slow motion only.
s2gb	28	位相 B の地絡インジケータ。	0x0: Normal operation 0x1: Short-to-GND detected on phase B. The driver is disabled. The flags stay active until the driver is disabled by software (*TOFF* = 0) or by the ENN input.

ビットフィールド	ビット	説明	デコード
s2ga	27	位相Aの地絡インジケータ。	0x0: Normal operation 0x1: Short-to-GND detected on phase A. The driver is disabled. The flags stay active until the driver is disabled by software (*TOFF* = 0) or by the ENN input.
ot	25	過熱フラグ。	0x0: Normal operation 0x1: Overtemperature limit is reached. Drivers are disabled until otpw is also cleared due to cooling down of the IC. The overtemperature flag is common for both bridges.
stallguard	24	StallGuard2/StallGuard+のステータス。	0x0: Normal operation 0x1: Motor stall detected by StallGuard2 (in SpreadCycle operation), respectively, by StallGuard+ (in StealthChop+ operation).
CS_ACTUAL	23:16	実際のモーター目標電流（CoolStep機能の実際のCoolStep/CoolStep+電流がイネーブル）。 CoolStep のモニタリングおよび *IRUN* と *IHOLD* の電流スケーリングに使用。	
stealth	14	StealthChop+インジケータ。	0x0: SpreadCycle mode 0x1: StealthChop+ mode
s2vsb	13	位相 B の電源への短絡インジケータ。	0x0: No error 0x1: Short-to-supply detected on phase B. The driver is disabled. The flags stay active until the driver is disabled by software (*TOFF* = 0) or by the ENN input.
s2vsa	12	位相 A の電源への短絡インジケータ。	0x0: No error 0x1: Short-to-supply detected on phase A. The driver is disabled. The flags stay active until the driver is disabled by software (*TOFF* = 0) or by the ENN input.
SG_RESULT	9:0	StallGuard2の結果、StallGuard+の結果（チョッパー・モードに依存）、またはSpreadCycleチョッパーを使用しモーター温度推定のために停止しているコイルAのPWMオン時間。 機械的負荷測定： StallGuard2/StallGuard+の結果は、機械的なモーター負荷を測定する手段を与えます。値が大きいと機械的負荷が小さいことを意味します。StallGuard2では、値が0の場合負荷が最大であることを示します。最適な*SGT*設定値を用いた場合、これはモーター・ストールのインジケータとなります。ストール検出では、*SG_RESULT*と0を比較してストールの検出を行います。*SG_RESULT*は、これをプログラマブルな上限値および下限値と比較することで、CoolStep動作の基盤として使用されます。StealthChop+モードでは使用できません。 StallGuard2 は、微細なマイクロステップ動作で最もよく機能します。 SpreadCycle モード時の温度測定： 停止時には、StallGuard2 の結果は取得できません。代わりに、*SG_RESULT* は、モーター・コイル A のチョッパー・オン時間を示します。この測定を十分に高抵抗のモーターを組み合わせて用いることにより、モーター・コイルの温度を推定できます：モーターを定められたマイクロステップ・ポジションに向けて特定の電流設定値で動作させると、チョッパー・オン時間を読み出すことでモーター温度を大まかに推定できます。モーター温度が上昇するにつれてコイル抵抗が増し、チョッパー・オン時間も増加します。StallGuard4 の詳細については、*SG4_RESULT* を参照してください。	

PWMCONF（0x70）

BIT	31	30	29	28	27	26	25	24
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–
BIT	23	22	21	20	19	18	17	16
Field	–	–	–	–	SD_ON_MEAS_HI[3:0]			
Reset	–	–	–	–	0x0F			
Access Type	–	–	–	–	Write, Read			
BIT	15	14	13	12	11	10	9	8
Field	SD_ON_MEAS_LO[3:0]				–	–	–	–
Reset	0x0E				–	–	–	–
Access Type	Write, Read				–	–	–	–
BIT	7	6	5	4	3	2	1	0
Field	–	–	–	–	–	–	–	–
Reset	–	–	–	–	–	–	–	–
Access Type	–	–	–	–	–	–	–	–

ビットフィールド	ビット	説明
SD_ON_MEAS_HI	19:16	ADCのサンプル・ポイントを低速減衰時間中の測定からオン時間中の測定に切り替えるための、ヒステリシスのデューティサイクルの上限スレッショルド。 *SD_ON_MEAS_HI* > *SD_ON_MEAS_LO*

ビットフィールド	ビット	説明
SD_ON_MEAS_LO	15:12	ADCのサンプル・ポイントをオン時間中の測定から低速減衰時間中の測定に切り替えるための、ヒステリシスのデューティサイクルの下限スレッショルド。 *SD_ON_MEAS_HI* > *SD_ON_MEAS_LO*

型番

PART NUMBER	TEMPERATURE RANGE	PIN-PACKAGE
TMC2262AFV+	-40°C to +125°C	30L FC2QFN - 6mm x 6mm
TMC2262AFV+T	-40°C to +125°C	30L FC2QFN - 6mm x 6mm

+は鉛（Pb）フリー／R_OHS 適合のパッケージであることを示します。

T はテープのリール巻きを示します。

改訂履歴

版数	改訂日	説明	改訂ページ
0	02/26	初版発行	–

注記

アナログ・デバイセズ社は、提供する情報が正確で信頼できるものであることを期していますが、その情報の利用に関して、あるいは利用によって生じる第三者の特許やその他の権利の侵害に関して一切の責任を負いません。また、アナログ・デバイセズ社の特許または特許の権利の使用を明示的または暗示的に許諾するものでもありません。仕様は、予告なく変更される場合があります。本紙記載の商標および登録商標は、各社の所有に属します。※日本語版資料は REVISION が古い場合があります。最新の内容については、英語版をご参照ください。

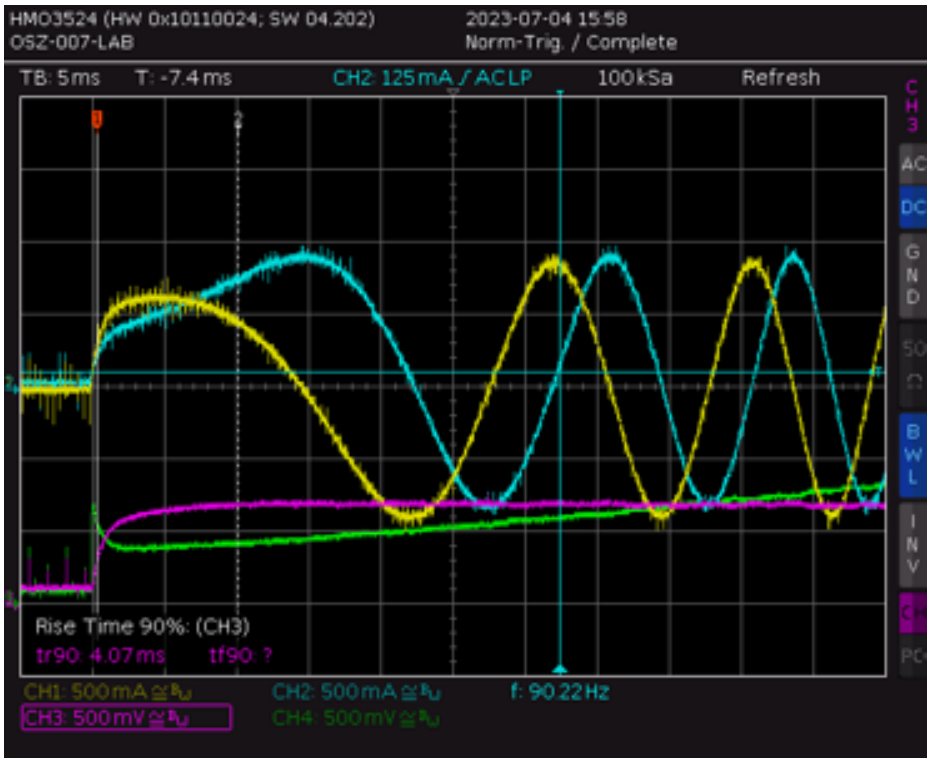
正誤表

この製品のデータシートに間違いがありましたので、お詫びして訂正いたします。
この正誤表は、2026 年 10 月 14 日現在、アナログ・デバイセズ株式会社で確認した誤りを記したものです。
なお、英語のデータシート改版時に、これらの誤りが訂正される場合があります。

正誤表作成年月日： 2026 年 10 月 14 日
製品名：　TMC2262
対象となるデータシートのリビジョン(Rev)：Rev.0
訂正箇所：90 頁、図 52

【誤】
図 51 と図 52 が同じ図となっています。

【正】
正しい図 52 は以下のものです。



アナログ・デバイセズ株式会社

本　　社／〒105-7323　東京都港区東新橋 1-9-1
東京汐留ビルディング 23F
大　阪営業所／〒532-0003　大阪府大阪市淀川区宮原 3-5-36
新大阪トラストタワー 10F
名古屋営業所／〒451-6038　愛知県名古屋市西区牛島 6-1
名古屋ルーセントタワー 40F

正誤表

この製品のデータシートに間違いがありましたので、お詫びして訂正いたします。
この正誤表は、2026 年 10 月 14 日現在、アナログ・デバイセズ株式会社で確認した誤りを記したものです。
なお、英語のデータシート改版時に、これらの誤りが訂正される場合があります。

正誤表作成年月日： 2026 年 10 月 14 日
製品名：　TMC2262
対象となるデータシートのリビジョン(Rev)：Rev.0
訂正箇所：136 頁、2 番目の表、ビットフィールド DCEL_THRS の説明欄

【誤】
「(*CS_ACTUAL* >= ((*DECEL_THRS* + 1) × *IRUN*)/16)の場合、**diag** を設定してください。

【正】
「(*CS_ACTUAL* >= ((*DECEL_THRS* + 1) × *IRUN*)/16)の場合、**flag** を設定してください。

アナログ・デバイセズ株式会社

本　　社／〒105-7323　東京都港区東新橋 1-9-1
東京汐留ビルディング 23F
大　阪営業所／〒532-0003　大阪府大阪市淀川区宮原 3-5-36
新大阪トラストタワー 10F
名古屋営業所／〒451-6038　愛知県名古屋市西区牛島 6-1
名古屋ルーセントタワー 40F

正誤表

この製品のデータシートに間違いがありましたので、お詫びして訂正いたします。
この正誤表は、2026 年 10 月 14 日現在、アナログ・デバイセズ株式会社で確認した誤りを記したものです。
なお、英語のデータシート改版時に、これらの誤りが訂正される場合があります。

正誤表作成年月日： 2026 年 10 月 14 日
製品名：　TMC2262
対象となるデータシートのリビジョン(Rev)：Rev.0
訂正箇所：184 頁、表内、ビットフィールド SG_RESULT の説明欄の一番下の文

【誤】
「StallGuard4 の詳細については、*SG4_RESULT* を参照してください。」

【正】
「StallGuard5 の詳細については、*SG_RESULT* を参照してください。」

アナログ・デバイセズ株式会社

本　　社／〒105-7323　東京都港区東新橋 1-9-1
東京汐留ビルディング 23F
大　阪営業所／〒532-0003　大阪府大阪市淀川区宮原 3-5-36
新大阪トラストタワー 10F
名古屋営業所／〒451-6038　愛知県名古屋市西区牛島 6-1
名古屋ルーセントタワー 40F