日本語参考資料
最新版英語データシートはこちら

全機能内蔵ベクトル制御ブラシレス DC ドライバ

TMC6460

製品のハイライト

- 超小型サーボ駆動 SoC
  - 36V の BLDC モーターおよび DC モーターを駆動
  - 最大 $3A_{RMS}$ の連続電流能力
  - 最大 $5A_{FS}$ の電流検出能力
  - −40ºC～125ºC の周囲温度
  - 容易に導入可能
- 高性能モーション・コントロール
  - ハードウェアベースのベクトル制御（FOC）
  - 最大 200kHz の制御ループおよび PWM エンジン
  - 8 点ランプ・コントローラを内蔵
  - 柔軟なフィードバック・エンジン（ABN、SinCos、デジタル／アナログ・ホール、SPI エンコーダ）
- 電力段およびシグナル・コンディショニング機能を内蔵
  - 3 個の 36V ハーフ・ブリッジ（55mΩ/FET）
  - スケーラブルで無損失の電流検出機能を内蔵
  - 正確な CSA および ADC
  - 短絡保護および過熱保護

詳細については機能説明のセクションを参照してください。

主なアプリケーション

- ロボティクスおよびオートメーション
- CCTV および PTZ カメラ
- SMT ピック・アンド・プレース機
- 写真用ジンバル
- 輸液ポンプおよびシリンジ・ポンプ
- BLDC/PMSM および DC サーボ・モーター

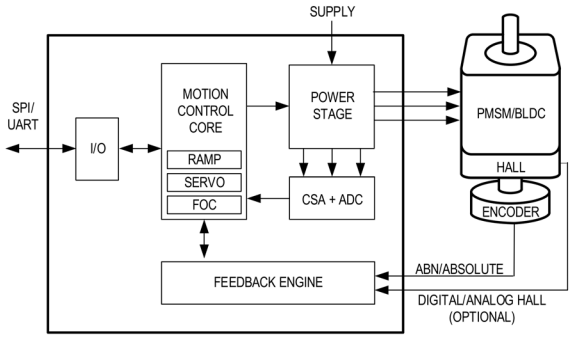
TMC6460 は、FOC コントローラ、電力段、電流検出機能、フィードバック・エンジンなどをすべて 1 つのチップに集積した、業界初の全機能内蔵サーボ駆動 IC です。その高度に集積化された設計により、比類のない小型化を実現しつつ、トルク、速度、ポジションの動作制御が可能です。

TMC6460 の内部には、FOC アルゴリズムおよびモーション・コントロールがハードウェアで実装されており、ソフトウェア実装のソリューションと比較して、モーターの優れた動的制御を実現しています。

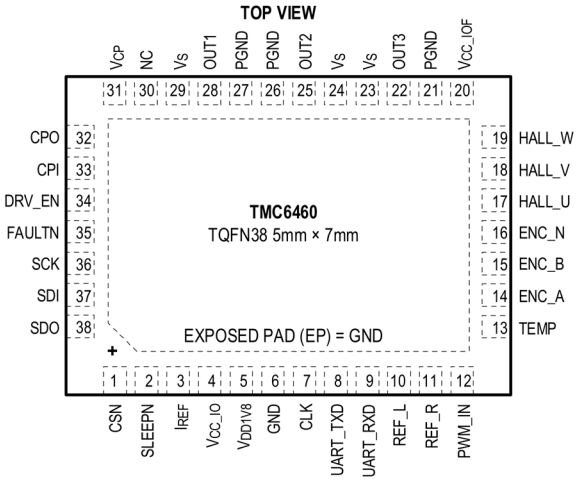
TMC6460 は高い PWM 周波数対応能力を備えているため、低インダクタンス・モーターやコアレス・モーターを効率的かつ高精度に制御できます。

TMC6460 は、柔軟性が高く容易に導入可能なソリューションであり、高精度の位置決めと高速の加速が求められるアプリケーションにおいて、理想的な選択肢となります。

簡略アプリケーション回路図



ピン配置



型番は、データシートの末尾に記載しています。

※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。

19-102107; Rev. 0; 03/26

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

製品のハイライト 1
主なアプリケーション 1
簡略アプリケーション回路図 1
ピン配置 1
絶対最大定格 15
パッケージ 16
38 ピン TQFN 16
電気的特性 16
タイミング図 21
ピン配置 22
端子説明 22
機能図 25
機能説明 25
通信インターフェースおよび制御インターフェース 26
フィードバック・エンジン 26
モーター制御コア 27
電力出力段 27
電流アクイジション 27
診断、フォルト処理、保護 28
詳細説明 29
PLL およびクロックの設定 29
パワー・マネージメント 29
フォルト管理 29
ADC の再初期化 30
ステータス・フラグとイベント 30
FAULTN ピン 31
シリアル・ペリフェラル・インターフェース（SPI） 32
SPI 読出し要求データグラム 32
SPI 読出し応答データグラム 32
SPI 書込み要求データグラム 33
SPI 書込み応答データグラム 33
SPI のデイジーチェーン・サポート 34
UART インターフェース 35
UART のボー・レート設定 35
UART 読出し要求データグラム 36
UART 読出し応答データグラム 37
UART 書込み要求データグラム 38

UART 書込み応答データグラム 39
UART リアルタイム・モニタリング・インターフェース（RTMI） 40
UART RTMI のチャンネル応答データグラム 41
UART RTMI ストリーム書込み要求データグラム 42
UART タイムアウト 43
UART 故障モニタリング 43
UART 受信データのフィルタ処理 44
UART CRC の設定および計算 44
ピン設定 45
デジタル入力の反転 45
デジタル入力のフィルタ処理 45
アナログ入力分圧器 46
プルアップ／プルダウン設定 46
ピンの再割り当て 46
フィードバック・エンジン・ピンの再割り当て 47
リファレンス・スイッチ・ピンの再割り当て 49
通信インターフェース・ピンの再割り当て 49
制御信号ピンの再割り当て 50
入力設定の説明図 51
ADC の設定と測定 54
角度フィードバック ADC 測定 55
その他の ADC 測定 55
モーター角度フィードバックの設定 56
インクリメンタル ABN エンコーダ 56
デジタル・ホール・センサー 58
アナログ・ホール・センサー 58
アナログ SinCos エンコーダ 60
SPI/SSI エンコーダ 62
フィードバック・エンジン 62
設定プロシージャ 63
16 ビット変換 65
補正ルックアップ・テーブル 66
Phi 外挿機能 68
Phi E の計算 69
速度のアクイジション 72
ポジションのアクイジション 74
現実の単位への変換 74
モーター制御コア 76

電流アクイジション 76
PWM エンジン 78
電流制御ループ 80
弱め界磁コントローラ 81
速度制御ループ 82
ポジション制御ループ 83
バイクワッド・フィルタ 85
直接モーション・モード 85
オープン・ループ・モーションのモード 86
標準的なモーション・モード 87
PRBS のモーション・モード 87
DC モーター・モード 88
電力段の設定 89
過電流保護 89
過熱保護 90
過電圧故障およびブレーキ・チョッパーの出力 91
出力周波数制限 91
ゼロ電流限界 91
ランプ・ジェネレータ 91
ランプ・ポジショニング・モード 93
ランプ速度モード 96
オープン・ループでの現実の単位への変換 97
リファレンス・スイッチ 98
ポジションのラッチ 99
停止モード 99
ホーミング・ルーチンの実行 100
スイッチ・モード・レジスタの概要 101
ランプ・ジェネレータのステータス・フラグおよびイベント 102
プログラマブル I/O コントローラ 102
ROM コード機能 103
GET_VER 105
I2C_INIT_PP 105
I2C_INIT_OD 105
I2C_READ 105
I2C_WRITE 106
I2C_EEPROM 106
RAM_READ 106
RAM_WRITE 106

RAM_EXE 107
FILTER_SETUP 107
CALC_FILT 107
DEBUG 108
PWM_INPUT 109
AIN_INPUT 112
SPI_ENC 114
アプリケーション情報 119
標準的なアプリケーション回路 120
レジスタ・マップ 121
レジスタの概要 121
レジスタの詳細な説明 138
0x000：CHIP.ID 138
0x001：CHIP.VARIANT 138
0x002：CHIP.REVISION 138
0x004：CHIP.INPUTS_RAW 138
0x005：CHIP.OUTPUTS_RAW 140
0x006：CHIP.IO_MATRIX 140
0x007：CHIP.IO_PU_PD 142
0x008：CHIP.IO_CONFIG 143
0x009：CHIP.STATUS_FLAGS 145
0x00A：CHIP.EVENTS 146
0x00B：CHIP.FAULTN_INT_MASK 148
0x00C：CHIP.SPI_STATUS_MASK 148
0x040：CLK_CTRL.CONFIG 148
0x041：CLK_CTRL.STATUS 149
0x080：ADC.CONFIG 149
0x081：ADC.ADC_VERSION 150
0x082：ADC.I2_I1_RAW 150
0x083：ADC.VM_I3_RAW 150
0x084：ADC.TEMP_RAW 150
0x085：ADC.AIN_V_AIN_U_RAW 150
0x086：ADC.AIN_AIN_W_RAW 151
0x08A：ADC.STATUS 151
0x08C：ADC.I123 152
0x0C0：MCC_ADC.I_GEN_CONFIG 152
0x0C1：MCC_ADC.IW_IU 153
0x0C2：MCC_ADC.IV 153

0x0C3：MCC_ADC.CSA_GAIN 153
0x0C4：MCC_ADC.EVENTS 153
0x0C5：MCC_ADC.DYN_GAIN_LIMITS_4X_3X 154
0x0C6：MCC_ADC.DYN_GAIN_LIMIT_2X 154
0x0C7：MCC_ADC.TEMP_LIMITS 154
0x0C8：MCC_ADC.CURRENT_OVERLOAD 154
0x100：MCC_CONFIG.MOTOR_MOTION 155
0x101：MCC_CONFIG.GDRV 156
0x102：MCC_CONFIG.PWM 157
0x103：MCC_CONFIG.PWM_PERIOD 157
0x104：MCC_CONFIG.BRAKE_CHOPPER_LIMITS 157
0x105：MCC_CONFIG.MCC_STATUS 158
0x106：MCC_CONFIG.TORQUE_FF_ACC_CONFIG 159
0x107：MCC_CONFIG.TORQUE_FF_VISC_FRIC_CONFIG 159
0x108：MCC_CONFIG.TORQUE_FF_COULOMB_FRIC 159
0x109：MCC_CONFIG.TORQUE_FEEDFORWARD 160
0x140：FOC.PID_CONFIG 160
0x141：FOC.PID_U_S_MAX 161
0x142：FOC.PID_FLUX_COEFF 161
0x143：FOC.PID_TORQUE_COEFF 162
0x144：FOC.PID_FIELDWEAK_COEFF 162
0x145：FOC.PID_VELOCITY_COEFF 162
0x146：FOC.PID_POSITION_COEFF 162
0x147：FOC.PID_POSITION_TOLERANCE 162
0x148：FOC.PID_POSITION_TOLERANCE_DELAY 163
0x149：FOC.PID_UQ_UD_LIMITS 163
0x14A：FOC.PID_TORQUE_FLUX_LIMITS 163
0x14B：FOC.PID_VELOCITY_LIMIT 163
0x14C：FOC.PID_POSITION_LIMIT_LOW 163
0x14D：FOC.PID_POSITION_LIMIT_HIGH 163
0x14E：FOC.PID_TORQUE_FLUX_TARGET 164
0x14F：FOC.PID_TORQUE_FLUX_OFFSET 164
0x150：FOC.PID_VELOCITY_TARGET 164
0x151：FOC.PID_VELOCITY_OFFSET 164
0x152：FOC.PID_POSITION_TARGET 164
0x153：FOC.PID_TORQUE_FLUX_ACTUAL 164
0x154：FOC.PID_VELOCITY_ACTUAL 165
0x155：FOC.PID_POSITION_ACTUAL 165

0x156：FOC.PID_POSITION_ACTUAL_OFFSET 165
0x157：FOC.PID_TORQUE_ERROR 165
0x158：FOC.PID_FLUX_ERROR 165
0x159：FOC.PID_VELOCITY_ERROR 165
0x15A：FOC.PID_VELOCITY_ERROR_MAX 166
0x15B：FOC.PID_POSITION_ERROR 166
0x15C：FOC.PID_POSITION_ERROR_MAX 166
0x15D：FOC.PID_TORQUE_INTEGRATOR 166
0x15E：FOC.PID_FLUX_INTEGRATOR 166
0x15F：FOC.PID_VELOCITY_INTEGRATOR 166
0x160：FOC.PID_POSITION_INTEGRATOR 167
0x161：FOC.PIDIN_TORQUE_FLUX_TARGET 167
0x162：FOC.PIDIN_VELOCITY_TARGET 167
0x163：FOC.PIDIN_POSITION_TARGET 167
0x164：FOC.PIDIN_TORQUE_FLUX_TARGET_LIMITED 167
0x165：FOC.PIDIN_VELOCITY_TARGET_LIMITED 167
0x166：FOC.PIDIN_POSITION_TARGET_LIMITED 168
0x167：FOC.FOC_IBETA_IALPHA 168
0x168：FOC.FOC_IQ_ID 168
0x169：FOC.FOC_UQ_UD 168
0x16A：FOC.FOC_UQ_UD_LIMITED 168
0x16B：FOC.FOC_UBETA_UALPHA 169
0x16C：FOC.FOC_UW_UU 169
0x16D：FOC.FOC_UV 169
0x16E：FOC.PWM_V_U 169
0x16F：FOC.PWM_W 169
0x170：FOC.FOC_STATUS 169
0x171：FOC.U_S_ACTUAL_I_S_ACTUAL 170
0x172：FOC.P_MOTOR 170
0x173：FOC.I_T_ACTUAL 170
0x174：FOC.PRBS_AMPLITUDE 170
0x175：FOC.PRBS_DOWN_SAMPLING_RATIO 170
0x180：BIQUAD.BIQUAD_EN 171
0x181：BIQUAD.VELOCITY_A1 171
0x182：BIQUAD.VELOCITY_A2 171
0x183：BIQUAD.VELOCITY_B0 171
0x184：BIQUAD.VELOCITY_B1 171
0x185：BIQUAD.VELOCITY_B2 172

0x186：BIQUAD.TORQUE_A1 172
0x187：BIQUAD.TORQUE_A2 172
0x188：BIQUAD.TORQUE_B0 172
0x189：BIQUAD.TORQUE_B1 172
0x18A：BIQUAD.TORQUE_B2 172
0x1C0：RAMPER.TIME_CONFIG 173
0x1C1：RAMPER.SWITCH_MODE 173
0x1C3：RAMPER.PHI_E 174
0x1C4：RAMPER.A1 175
0x1C5：RAMPER.A2 175
0x1C6：RAMPER.A_MAX 175
0x1C7：RAMPER.D1 175
0x1C8：RAMPER.D2 175
0x1C9：RAMPER.D_MAX 175
0x1CA：RAMPER.V_START 176
0x1CB：RAMPER.V1 176
0x1CC：RAMPER.V2 176
0x1CD：RAMPER.V_STOP 176
0x1CE：RAMPER.V_MAX 176
0x1CF：RAMPER.ACCELERATION 176
0x1D0：RAMPER.V_ACTUAL 177
0x1D1：RAMPER.POSITION 177
0x1D2：RAMPER.POSITION_LATCH 177
0x1D3：RAMPER.POSITION_ACTUAL_LATCH 177
0x1D4：RAMPER.STATUS 177
0x1D5：RAMPER.EVENTS 178
0x200：EXT_CTRL.VOLTAGE 179
0x202：EXT_CTRL.PWM_V_U 179
0x203：EXT_CTRL.PWM_W 179
0x240：FEEDBACK.CONF_CH_A 180
0x241：FEEDBACK.CONF_CH_B 181
0x242：FEEDBACK.PHI_E_OFFSET 181
0x243：FEEDBACK.LUT 182
0x244：FEEDBACK.VELOCITY_FRQ_CONF 182
0x245：FEEDBACK.VELOCITY_PER_CONF 183
0x246：FEEDBACK.VELOCITY_PER_FILTER 183
0x247：FEEDBACK.PHI_CONVERTED 183
0x248：FEEDBACK.CH_A 183

0x249：FEEDBACK.CH_B 184
0x24A：FEEDBACK.VELOCITY_FRQ 184
0x24B：FEEDBACK.VELOCITY_PER 184
0x24C：FEEDBACK.LUT_WDATA 184
0x24D：FEEDBACK.PHI_EXT_A 184
0x24E：FEEDBACK.PHI_EXT_B 185
0x24F：FEEDBACK.VELOCITY_EXT 185
0x250：FEEDBACK.OUTPUT_CONF 185
0x251：FEEDBACK.PHI_E 186
0x252：FEEDBACK.PHI_DIFF_LIMIT 186
0x280：ABN.CONFIG 186
0x281：ABN.COUNT 186
0x282：ABN.COUNT_N_CAPTURE 187
0x283：ABN.COUNT_N_WRITE 187
0x284：ABN.EVENTS 187
0x2C0：ABN2.CONFIG 187
0x2C1：ABN2.COUNT 188
0x2C2：ABN2.COUNT_N_CAPTURE 188
0x2C3：ABN2.COUNT_N_WRITE 188
0x2C4：ABN2.EVENTS 188
0x300：HALL.MAP_CONFIG 189
0x301：HALL.DIG_COUNT 189
0x302：HALL.ANA_CONFIG 190
0x303：HALL.ANA_UX_CONFIG 190
0x304：HALL.ANA_VN_CONFIG 190
0x305：HALL.ANA_WY_CONFIG 191
0x306：HALL.ANA_UX_OUT 191
0x307：HALL.ANA_VN_OUT 191
0x308：HALL.ANA_WY_OUT 191
0x309：HALL.ANA_OUT 191
0x30A：HALL.DIG_EVENTS 191
0x340：UART.CONTROL 192
0x341：UART.TIMEOUT 192
0x342：UART.STATUS 192
0x343：UART.EVENTS 193
0x344：UART.RTMI_CH_0 193
0x345：UART.RTMI_CH_1 193
0x346：UART.RTMI_CH_2 193

0x347：UART.RTMI_CH_3 194
0x348：UART.RTMI_CH_4 194
0x349：UART.RTMI_CH_5 194
0x34A：UART.RTMI_CH_6 194
0x34B：UART.RTMI_CH_7 194
0x380：IO_CONTROLLER.CONTROL 195
0x381：IO_CONTROLLER.COMMAND 195
0x382：IO_CONTROLLER.RESPONSE_0 195
0x383：IO_CONTROLLER.RESPONSE_1 195
0x384：IO_CONTROLLER.RESPONSE_2 196
0x385：IO_CONTROLLER.RESPONSE_3 196
型番 197
改訂履歴 198

図一覧

図 1. SPI のタイミング図（SPI モード 1）.....21
図 2. 機能図.....25
図 3. FOC およびモーション・コントロール・コア.....27
図 4. SPI 読出し要求データグラムの構造.....32
図 5. SPI 読出し応答データグラムの構造.....33
図 6. 直前の読出し要求データグラムからデータを受信するための SPI ダミー読出し要求.....33
図 7. SPI 書込み要求データグラムの構造.....33
図 8. SPI 書込み応答データグラムの構造.....34
図 9. 3 個の SPI ペリフェラルを一列に接続した SPI デイジーチェーンの例.....34
図 10. 3 個の SPI ペリフェラルを一列に接続した SPI デイジーチェーン・データグラムの例.....35
図 11. UART 読出し要求データグラムの構造（NORMAL_CRC_EN = 0_b）.....37
図 12. UART 読出し要求データグラムの構造（NORMAL_CRC_EN = 1_b）.....37
図 13. UART 読出し応答データグラムの構造（NORMAL_CRC_EN = 0_b）.....37
図 14. UART 読出し応答データグラムの構造（NORMAL_CRC_EN = 1_b）.....37
図 15. UART 読出しアクセスの例.....38
図 16. UART 書込み要求データグラムの構造（NORMAL_CRC_EN = 0_b）.....38
図 17. UART 書込み要求データグラムの構造（NORMAL_CRC_EN = 1_b）.....38
図 18. UART 書込み応答データグラムの構造（RTMI_CRC_EN = 0_b）.....39
図 19. UART 書込み応答データグラムの構造（RTMI_CRC_EN = 1_b）.....39
図 20. UART 書込みアクセスの例.....40
図 21. RTMI ストリーム・チャンネル応答データグラムの構造（RTMI_CRC_EN = 0_b）.....41
図 22. UART ストリーム・チャンネル応答データグラムの例.....42
図 23. RTMI ストリーム・チャンネル応答データグラムの構造（RTMI_CRC_EN = 1_b）.....42
図 24. RTMI ストリーム書込み要求データグラムの構造（RTMI_CRC_EN = 0_b）.....42
図 25. RTMI ストリーム書込み要求データグラムの構造（RTMI_CRC_EN = 1_b）.....43
図 26. デジタル差動 ABN エンコーダ入力、ENC_A ピンと HALL_U ピンの例.....48
図 27. デジタル差動 ABN エンコーダ入力、ENC_B ピンと HALL_V ピンの例。DIFF_DELAY は、ENC_B が切り替わった後にアクティブになります。.....49
図 28. エンコーダ・フィードバックの設定オプション.....51
図 29. デジタル・ホール・フィードバックの設定オプション.....52
図 30. アナログ・ホール・フィードバックおよびアナログ入力の設定オプション.....53
図 31. リファレンス・スイッチ入力の設定オプション.....54
図 32. 代表的な ABN エンコーダ信号と結合 N パルス.....57
図 33. アナログ三相エンコーダのスケーリングとオフセットのセットアップ例.....60
図 34. フィードバック・エンジンの概要（レジスタ・フィールドを含む）.....63
図 35. 補正 LUT の機能図.....68
図 36. モーター速度の増加に伴う速度メータの測定ノイズ.....74
図 37. PWM 変調モード.....79
図 38. トルクおよび磁束の制御ループ.....80
図 39. 弱め界磁の働き.....82
図 40. 弱め界磁の制御ループ.....82
図 41. 速度制御ループ.....82
図 42. ポジション制御ループ.....84
図 43. バイクワッド・フィルタの構造.....85
図 44. PRBS シフト・レジスタ.....88
図 45. ランプ・コントローラの目標値の生成.....92
図 46. ポジショニング・モーション・プロファイル用の代表的な 8 点速度ランプ.....95
図 47. 修正されたランプ・ポジショニング・プロファイルの例.....95
図 48. 目標ポジションの変更を原因としてランプ・ポジショニング・プロファイルに生じる可能性のある修正.....95
図 49. 代表的なランプ速度プロファイル.....97
図 50. 目標速度の変更を原因としてランプ速度プロファイルに生じる可能性のある修正.....97
図 51. リファレンス・スイッチの使用例.....99
図 52. PWM_INPUT ROM コード機能の例.....112

図 53. AIN_INPUT ROM コード機能の例 113
図 54. 連続 SPI エンコーダ読出しの例。目標レジスタは PHI_EXT_A。 117
図 55. 同一のデータグラムを 2 回送信して RESPONSE_3 レジスタに応答を得ることによる、ワンタイム SPI エンコーダ読出しの例 . 117
図 56. 連続 SSI エンコーダ読出しの例。目標レジスタは PHI_EXT_B。 118
図 57. ギアボックス後段にインクリメンタル光エンコーダ、モーターにデジタル・ホール・センサーを備えた BLDC 119
図 58. アナログ・ホール・センサーまたは SinCos エンコーダを備えた BLDC 119
図 59. 1 つはインクリメンタル A/B/N エンコーダ、もう 1 つはインクリメンタルまたは絶対（SSI など）エンコーダの、
2 つのエンコーダを備えた BLDC。1 つのエンコーダはモーターに取り付けられ、もう 1 つはギアボックスの後段にあります。 119
図 60. モーターに光エンコーダとデジタル・ホールを備え、更にブレーキ・チョッパーを使用する BLDC 119
図 61. 標準的なアプリケーション回路のブロック図 120

表一覧

表 1. 概要：ステータス・フラグとイベント 30
表 2. FAULTN ピンの代替機能の割り当て 31
表 3. ボー・レートの設定 36
表 4. UART の速度制限 36
表 5. フィードバック・エンジン・ピンの再割り当て 47
表 6. ホール入力ピンのその他の再割り当て 48
表 7. リファレンス・スイッチ・ピンの再割り当て 49
表 8. SPI ピンの再割り当て 49
表 9. UART ピンの再割り当て 50
表 10. 制御ピンの再割り当て 50
表 11. アナログ入力分圧器オプション 55
表 12. アナログ・エンコーダ入力値の計算オプション 55
表 13. ADC のクリップ・イベントおよびフラグの概要 56
表 14. ホール・センサー区画のオフセットおよびカウント方向の設定 58
表 15. アナログ・ホール・センサーの設定レジスタ・フィールド 59
表 16. アナログ・ホール・センサー区画のオフセットおよびカウント方向の設定 60
表 17. アナログ SinCos エンコーダの設定レジスタ・フィールド 61
表 18. ポジションの現実の単位への変換式 75
表 19. 速度の現実の単位へのスケーリング・ファクタ 75
表 20. 電流検出の計算 76
表 21. 様々なゲイン設定に対して測定可能な最大電流 77
表 22. ADC の電流値と MCC_ADC の生の値の関係 78
表 23. 空間ベクトル PWM モード 78
表 24. CSA セトリング・タイム T_{SET} 79
表 25. PRBS のモーション・モード 88
表 26. SLEW_RATE レジスタ・フィールド 89
表 27. StealthChop2 の PWM 周波数の選択 89
表 28. 過電流保護のスレッショルド 89
表 29. 過電流保護のデグリッチおよび自動再試行時間 90
表 30. 様々な MOTION_MODE の選択に対するランプ・ジェネレータの使用法 92
表 31. ポジション・モードのランプ・パラメータ 94
表 32. 速度モードのランプ・パラメータ 96
表 33. RAMPER.SWITCH_MODE のビットフィールド 101
表 34. RAMPER.STATUS のビットフィールド 102
表 35. RAMPER.EVENTS のビットフィールド 102
表 36. ROM コード機能の概要 104
表 37. GET_VER のコマンドおよび応答 105
表 38. I2C_INIT_PP のコマンドおよび応答 105
表 39. I2C_INIT_OD のコマンドおよび応答 105
表 40. I2C_READ のコマンドおよび応答 105
表 41. I2C_WRITE のコマンドおよび応答 106
表 42. I2C_EEPROM のコマンドおよび応答 106
表 43. RAM_READ のコマンドおよび応答 106
表 44. RAM_WRITE のコマンドおよび応答 106
表 45. RAM_EXE のコマンドおよび応答 107
表 46. FILTER_SETUP のコマンドおよび応答 107
表 47. CALC_FILT のコマンドおよび応答 107
表 48. DEBUG のコマンドおよび応答 108
表 49. ROM コードの DEBUG：トリガ・オプション 109
表 50. PWM_INPUT のコマンドおよび応答 109
表 51. AIN_INPUT のコマンドおよび応答 112
表 52. SPI/ENC のコマンドおよび応答 114
表 53. SPI/SSI エンコーダの機能ピンの割り当て 114

表 54. I/O コントローラの入力および出力に対するピンの可用性 114
表 55. SPI/SSI エンコーダのマスク／シフト例 116

絶対最大定格

V_S～GND ..... −0.3V～41V
V_{DD1V8}～GND ..... −0.3V～MIN (2.2, V_{CC_IO} + 0.3)V
V_{CC_IO}～GND ..... −0.3V～6V
V_{CC_IOF}～GND ..... −0.3V～6V
OUT1、OUT2、OUT3～GND ..... −0.6V～V_S + 0.3V
V_{CP}～GND .....V_S − 0.3V～Min. (44, V_S + 6)V
CPO～GND ..... V_S − 0.3V～Min. (44, V_S + 6)V
CPI～GND ..... −0.3V～Min. (41, V_S + 0.3)V
I_{REF}～GND ..... −0.3V～Min. (2.2, V_{DD1V8} + 0.3)V
IO ピン～GND
（V_{CC_IO} でリファレンス供給） ..... −0.3V～V_{CC_IO} + 0.3V
IO ピン～GND
（V_{CC_IOF} でリファレンス供給） ..... −0.3V～V_{CC_IOF} + 0.3V
動作ジャンクション温度範囲 .....−40°C～125°C
ジャンクション温度 ..... 160°C
保管温度範囲 ..... −65°C～150°C
ハンダ付け処理温度（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。

パッケージ

38 ピン TQFN

Package Code	T3857+1C
Outline Number	21-0172
Land Pattern Number	90-0076
Thermal Resistance, Single Layer Board:	
Junction-to-Ambient (θ_{JA})	38°C/W
Junction-to-Case Thermal Resistance (θ_{JC})	1°C/W
Thermal Resistance, Four Layer Board:	
Junction-to-Ambient (θ_{JA})	28°C/W
Junction-to-Case Thermal Resistance (θ_{JC})	1°C/W

電気的特性

（V_S = 2.4V～36V、R_{REF} = 60.4kΩ 0.1%、代表値は T_A = 25°Cおよび V_S = 24V を前提としています。動作温度範囲およびこれに伴う電源電圧範囲にわたる制限値は、設計および特性評価によって裏付けられています。「GBD」と記載された仕様は設計により確保されていますが、製品テストは行っていません。）

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
POWER SUPPLY							
Supply Voltage on V_S	V_S			2.4		36	V
Sleep Mode Current Consumption on V_S	I_{VS_SLEEP}	SLEEPN = 0V	Output leakage not included.		0	2	µA
Standby Mode Current Consumption on V_S	I_{VS_STOP}	SLEEPN = V_{CC_IO} LP_MODE = 1_b	Output leakage not included.		0	2	µA
Quiescent Mode Current Consumption on V_S	I_{VS_DRVOFF}	DRV_EN = 0V SLEEPN = V_{CC_IO} LP_MODE = 0_b			730	1100	µA
Supply Voltage on V_{CC_IO}	V_{CC_IO}			2.2		5.5	V
Sleep Mode Current Consumption on V_{CC_IO}	I_{VCCIO_SLEEP}	SLEEPN = 0V			0	2	µA
Standby Mode Current Consumption on V_{CC_IO}	I_{VCCIO_STOP}	SLEEPN = V_{CC_IO} LP_MODE = 1_b			3.1	5	mA
Quiescent Mode Current Consumption on V_{CC_IO}	I_{VCCIO_DRVOFF}	DRV_EN = 0V SLEEPN = V_{CC_IO} LP_MODE = 0_b			16	25	mA
Supply Voltage on V_{CC_IOF}	V_{CC_IOF}			2.2		5.5	V
Sleep Mode Current Consumption on V_{CC_IOF}	I_{VCCIOF_SLEEP}	SLEEPN = 0V			0	1	µA
Standby Mode Current Consumption on V_{CC_IOF}	I_{VCCIOF_STOP}	SLEEPN = V_{CC_IO} LP_MODE = 1_b			8	13	µA
Quiescent Mode Current Consumption on V_{CC_IOF}	I_{VCCIOF_DRVOFF}	DRV_EN = 0V SLEEPN = V_{CC_IO} LP_MODE = 0_b			70	100	µA

（V_S = 2.4V～36V、R_{REF} = 60.4kΩ 0.1%、代表値は T_A = 25°Cおよび V_S = 24V を前提としています。動作温度範囲およびこれに伴う電源電圧範囲にわたる制限値は、設計および特性評価によって裏付けられています。「GBD」と記載された仕様は設計により確保されていますが、製品テストは行っていません。）

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
1.8V Regulator Output Voltage	V_{DD1V8}	V_{CC_IO} = 2.2V I_{LOAD} = 40mA		1.8		V
V_{DD1V8} Short Circuit Current	I_{VDD1V8_SHT}	V_{DD1V8} = 0V			20	mA
V_{DD1V8} Current Limit	I_{VDD1V8_LIM}	Normal operation	40			mA
Charge Pump Voltage	V_{CP}	V_S > 3.3V I_{LOAD} = 1mA		V_S + 2.7		V
		V_S = 2.4V I_{LOAD} = 1mA	V_S + 1.8			
LOGIC INPUTS AND OUTPUTS						
V_{CC_IO} Input Voltage Level High	V_{VCCIO_IH}		0.7 × V_{CC_IO}			V
V_{CC_IO} Input Voltage Level Low	V_{VCCIO_IL}				0.3 × V_{CC_IO}	V
V_{CC_IO} Input Hysteresis	V_{VCCIO_HYS}			0.15 × V_{CC_IO}		V
V_{CC_IOF} Input Voltage Level High	V_{VCCIOF_IH}		0.7 × V_{CC_IOF}			V
V_{CC_IOF} Input Voltage Level Low	V_{VCCIOF_IL}				0.3 × V_{CC_IOF}	V
V_{CC_IOF} Input Hysteresis	V_{VCCIOF_HYS}			0.15 × V_{CC_IOF}		V
Pull-Up Resistance I/O Pins	R_{PU}	To V_{CC_IO}/V_{CC_IOF}	60	100	140	kΩ
Pull-Down Resistance I/O Pins	R_{PD}	To GND	60	100	140	kΩ
Pull-Down Resistance SLEEPN	R_{PD_SLEEPN}		0.6	1	1.4	MΩ
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} - 0.4			V
Open-Drain Output Logic High Leakage Current	I_{OH}	V_{PIN} = 5.5V	-1		+1	µA
POWER STAGE						
Max. Full-Scale Current	I_{FS}				5	A
Full-Scale Current		LS_RES_ON = 11_b			5	A
		LS_RES_ON = 10_b			3.75	
		LS_RES_ON = 01_b			2.5	
		LS_RES_ON = 00_b			1.25	

（V_S = 2.4V～36V、R_{REF} = 60.4kΩ 0.1%、代表値は T_A = 25°Cおよび V_S = 24V を前提としています。動作温度範囲およびこれに伴う電源電圧範囲にわたる制限値は、設計および特性評価によって裏付けられています。「GBD」と記載された仕様は設計により確保されていますが、製品テストは行っていません。）

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Output ON-Resistance Low-Side	$R_{ON,LS}$	LS_RES_ON = 11_b		55	110	mΩ
		LS_RES_ON = 10_b		78	150	
		LS_RES_ON = 01_b		112	220	
		LS_RES_ON = 00_b		200	400	
Output ON-Resistance High-Side	$R_{ON,HS}$			56	110	mΩ
Output Leakage	I_{LEAK}	DRV_EN = 0V	-20	0	+20	µA
Output Slew Rate	SR	SLEW_RATE = 00_b		100		V/µs
		SLEW_RATE = 01_b		200		
		SLEW_RATE = 10_b		400		
		SLEW_RATE = 11_b		800		
Output Frequency Limit		See *Output Frequency Limit.*			599	Hz
PROTECTIONS						
Overcurrent Protection Threshold	OCP	LS_RES_ON = 11_b OCP_DETECTION_MODE = 0_b	10			A
		LS_RES_ON = 10_p OCP_DETECTION_MODE = 0_b	7.5			
		LS_RES_ON = 01_b OCP_DETECTION_MODE = 0_b	5			
		LS_RES_ON = 00_b OCP_DETECTION_MODE = 0_b	2.5			
Overcurrent Protection Threshold in V_{MODE}	OCP_VMODE	OCP_DETECTION_MODE = 1_b	1.6	2.2	2.8	V
Overcurrent Protection Blanking Time	t_{OCP}	OCP_DEGLITCH = 00_b	0.25	0.5	0.63	µs
		OCP_DEGLITCH = 01_b	0.3	0.8	1.06	
		OCP_DEGLITCH = 10_b	0.65	1.4	1.92	
		OCP_DEGLITCH = 11_b	1.3	2.6	3.65	
UVLO Threshold on V_S	$UVLO_{VS}$	V_S rising	2.1	2.2	2.3	V
UVLO Threshold on V_S Hysteresis	$UVLO_{VS_HYS}$			0.1		V
UVLO Threshold on V_{CC_IO}	$UVLO_{VCCIO}$	V_{CC_IO} rising	1.6	1.9	2.1	V
UVLO Threshold on V_{CC_IO} Hysteresis	$UVLO_{VCCIO_HYS}$			0.11		V
UVLO Threshold on V_{CC_IOF}	$UVLO_{VCCIOF}$	V_{CC_IOF} rising	1.6	1.9	2.1	V
UVLO Threshold on V_{CC_IOF} Hysteresis	$UVLO_{VCCIOF_HYS}$			0.11		V
Thermal Protection Shutdown Threshold	TSD			165		°C

（V_S = 2.4V～36V、R_{REF} = 60.4kΩ 0.1%、代表値は T_A = 25°Cおよび V_S = 24V を前提としています。動作温度範囲およびこれに伴う電源電圧範囲にわたる制限値は、設計および特性評価によって裏付けられています。「GBD」と記載された仕様は設計により確保されていますが、製品テストは行っていません。）

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Thermal Protection Threshold Hysteresis				20		°C
CURRENT SENSE AND SIGNAL CONDITIONING						
IREF Pin Resistor	R_{REF}			60.4		kΩ
IREF Output Voltage	V_{REF}		0.882	0.9	0.918	V
IREF Monitor Window	R_{REF_OK}		57		63	kΩ
Current Trip Regulation Accuracy	DITRIP1	ITRIG from 430mA to I_{FS}	-5		+5	%
	DITRIP1_INT	INT_IREF_SEL = 1_b ITRIG from 430mA to I_{FS}	-8		+8	
SYSTEM TIMINGS						
Sleep Time	t_{SLEEP}	SLEEPN falling edge to OUT1,2,3 in tristate.			50	µs
Wakeup Time from Sleep	t_{WAKE}	SLEEPN rising edge to normal operation.			4	ms
Wakeup Time from Standby Mode		LP_MODE changes to 0_b to normal operation.			3	ms
Enable Time	t_{EN}	DRV_EN_BIT changes to 1_b to driver on.			1	µs
Disable Time	t_{OFF}	DRV_EN falling edge to driver off.			650	ns
Internal Oscillator Frequency	f_{OSC}		14.25	15	15.75	MHz
PLL Frequency	f_{PLL}			120		MHz
System Clock Frequency	f_{SYS}			60		MHz
External Clock Frequency	f_{EXT_CLK}		1		32	MHz
PWM Frequency	f_{PWM}		1.831		200	kHz
SPI						
SCK Valid Time before or after Change of CSN	t_{CC}	See also *Figure 1* for all timings.	t_{SCK}			ns
CSN High Time	t_{CSH}		4 × t_{SCK}			ns
SCK Low Time	t_{CL}		20			ns
SCK High Time	t_{CH}		20			ns
SCK Frequency	f_{SCK} = 1/t_{SCK}	V_{CC_IO} = 3.0 V			8	MHz
SDI Setup Time before SCK Falling Edge	t_{DS}		10			ns
SDI Hold Time after SCK Falling Edge	t_{DH}		10			ns
Data Out Valid Time after SCK Rising Edge	t_{DO}	V_{CC_IO} = 2.2 V		55	75	ns
		V_{CC_IO} = 3.0 V		36	55	

（V_S = 2.4V～36V、R_{REF} = 60.4kΩ 0.1%、代表値は T_A = 25°Cおよび V_S = 24V を前提としています。動作温度範囲およびこれに伴う電源電圧範囲にわたる制限値は、設計および特性評価によって裏付けられています。「GBD」と記載された仕様は設計により確保されていますが、製品テストは行っていません。）

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
UART						
Baud Rate (No Autobaud, External Clock)		AUTOBAUD_EN = 0_b PLL_SRC = 1_b V_{CC_IO} = 3V	9600		12M	bit/s
Baud Rate (No Autobaud, Internal Oscillator)		AUTOBAUD_EN = 0_b PLL_SRC = 0_b V_{CC_IO} = 3V	9600		12M	bit/s
Baud Rate (Autobaud, External Clock)		AUTOBAUD_EN = 1_b PLL_SRC = 1_b V_{CC_IO} = 3V	9600		7.5M	bit/s
Baud Rate (Autobaud, Internal Oscillator)		AUTOBAUD_EN = 1_b PLL_SRC = 0_b V_{CC_IO} = 3V	9600		6M	bit/s
ENCODER INTERFACE AND ANALOG INPUTS						
A/B/N Input Minimum Low Time	$t_{ABNL,MIN}$			50		ns
A/B/N Input Minimum High Time	$t_{ABNH,MIN}$			50		ns
A/B/N Maximum Input Frequency	$f_{ABN,MAX}$	ENC_FLT = 00_b		10		MHz
A/B/N Input Filter Frequency[1]	$f_{FILTABN}$	ENC_FLT = 01_b		1.0		MHz
		ENC_FLT = 10_b		0.5		
		ENC_FLT = 11_b		0.1		
ADC Resolution				12		Bit
Analog Input Voltage Range	V_{AIN}	ANA_DIV_VCCIO = 11_b ANA_DIV_VCCIOF = 11_b	75m		1.1	V
		ANA_DIV_VCCIO = 10_b ANA_DIV_VCCIOF = 10_b	165m		2.4	
		ANA_DIV_VCCIO = 01_b ANA_DIV_VCCIOF = 01_b	225m		3.3	
		ANA_DIV_VCCIO = 00_b ANA_DIV_VCCIOF = 00_b	338m		5	
Analog Input Pull-Down Resistance	R_{PD_AIN}	ANA_DIV_VCCIO = 11_b ANA_DIV_VCCIOF = 11_b	40	55	75	kΩ
		ANA_DIV_VCCIO = 10_b ANA_DIV_VCCIOF = 10_b	70	100	138	
		ANA_DIV_VCCIO = 01_b ANA_DIV_VCCIOF = 01_b	57	79	113	
		ANA_DIV_VCCIO = 00_b ANA_DIV_VCCIOF = 00_b	50	70	95	
AIN Analog Input Sampling Frequency	f_{AIN}			f_{PWM}		Hz

（V_S = 2.4V～36V、R_{REF} = 60.4kΩ 0.1%、代表値は T_A = 25°Cおよび V_S = 24V を前提としています。動作温度範囲およびこれに伴う電源電圧範囲にわたる制限値は、設計および特性評価によって裏付けられています。「GBD」と記載された仕様は設計により確保されていますが、製品テストは行っていません。）

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Internal Temperature Measurement Accuracy				±10		°C
V_S Supply Voltage Measurement Accuracy			-5		+5	%

Note 1: ENC_FLT = 00_b の場合、A/B/N 入力フィルタはディスエーブルされます。

タイミング図

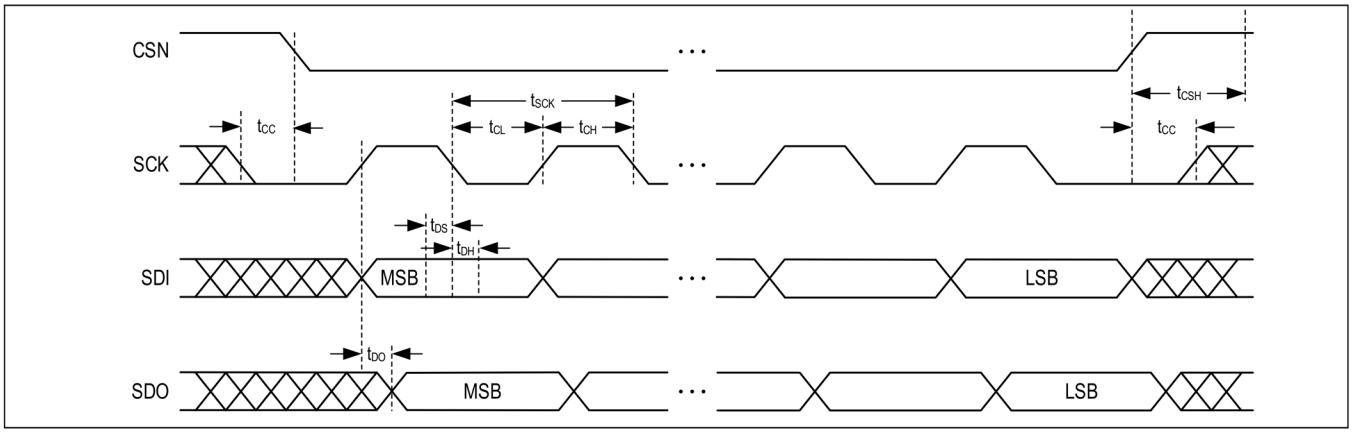
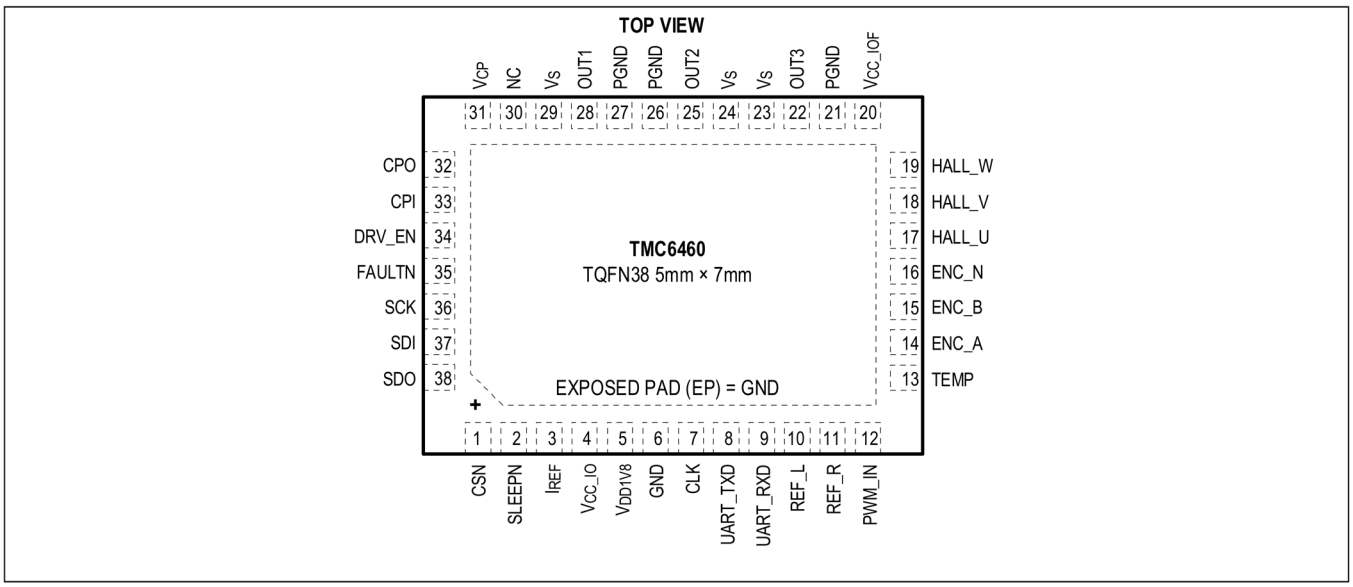


図 1. SPI のタイミング図（SPI モード 1）

ピン配置

TMC6460 TQFN のピン配置



端子説明

ピン	名称	機能	リファレンス電源	タイプ
1	CSN	SPI チップ・セレクト入力（アクティブ・ロー）。	VCC_IO	Digital Input (Pull-Up)
2	SLEEPN	スリープ／リセット入力（アクティブ・ロー）。このピンがアサートされるとデバイスはスリープ・モードになります。スリープ・モードの間、レジスタの内容は保持されないため、このピンはリセット入力としても作用します。このピンにはプルダウンが内蔵されています。使用しない場合は、V_{CC_IO} に接続します。モーターの動作中には、このピンをアサートしないでください。	VCC_IO	Digital Input (Pull-Down)
3	I_{REF}	電流リファレンス入力。GND との間に 60.4kΩ の 0.1%抵抗を接続します。デバイスが内部リファレンス抵抗を用いるよう設定されている場合には、このピンを GND に直接接続できます。		Analog Input
4	V_{CC_IO}	内部ロジック、I/O ピン、内部 1.8V LDO 用の電源入力。最高性能を発揮するには、1µF の低 ESR セラミック・コンデンサを短い配線パターンを用いてピンの近くで GND に接続します。	VCC_IO	Supply
5	V_{DD1V8}	内部 1.8V LDO の出力。最高性能を発揮するには、2.2µF の低 ESR セラミック・コンデンサを短い配線パターンを用いてピンの近くで GND に接続します。		Supply
6	GND	共通グラウンド。		GND
7	CLK	外部クロック入力。内部発振器の精度がアプリケーションに対し不十分な場合は、クロック信号を供給します。詳細については、PLL およびクロックの設定のセクションを参照してください。	VCC_IO	Digital Input (Pull-Down)
8	UART_TXD	デフォルト：UART_TXD – UART 送信出力 ALT1：BRAKE – ブレーキ・チョッパーPWM 出力 ALT2：DIRECT_OUT[0] – I/O コントローラ直接出力 0 ALT3：DIRECT_OUT[1] – I/O コントローラ直接出力 1	VCC_IO	Digital Output
9	UART_RXD	デフォルト：UART_RXD – UART 受信入力 ALT1：DIR_IN – 方向制御入力（I/O コントローラ使用） ALT3：DIRECT_IN[0] – I/O コントローラ直接入力 0	VCC_IO	Digital Input (Pull-Up/Pull-Down)
10	REF_L	デフォルト：REF_L – 左リファレンス・スイッチ入力 ALT1：REF_H – ホーム・リファレンス・スイッチ入力 ALT3：DIRECT_IN[0] – I/O コントローラ直接入力 0	VCC_IOF	Digital Input (Pull- Up/Pull-Down)

11	REF_R	デフォルト：REF_R – 右リファレンス・スイッチ入力 ALT1：REF_H – ホーム・リファレンス・スイッチ入力 ALT2：BRAKE – ブレーキ・チョッパーPWM出力 ALT3：DIRECT_OUT[1] – I/O コントローラ直接出力 1	VCC_IOF	Digital Input (Pull-Up/Pull-Down)/Digital Output
12	PWM_IN	デフォルト：PWM_IN – PWM制御入力（I/Oコントローラ使用） ALT1：AIN – アナログ制御入力（I/Oコントローラ使用） ALT3：DIRECT_IN[1] – I/O コントローラ直接入力 1	VCC_IO	Digital Input (Pull-Up/Pull-Down)/Analog Input
13	TEMP	モーター温度アナログ入力。	VCC_IO	Analog Input
14	ENC_A	デフォルト：ENC_A – 第一ABNエンコーダのA入力（デジタル差動入力の非反転信号としても使用）。 ALT1：AINN_U – 差動入力と共に使用するための反転アナログ・ホールUまたはSinCos X信号。 ALT2：HALL_U – デジタル・ホール・センサーのU入力 ALT3：DIRECT_OUT[0] – I/O コントローラ直接出力 0	VCC_IOF	Digital Input (Pull-Up/Pull-Down)/Digital Output/Analog Input
15	ENC_B	デフォルト：ENC_B – 第一ABNエンコーダのB入力（デジタル差動入力の非反転信号としても使用）。 ALT1：AINN_V – 差動入力と共に使用するための反転アナログ・ホールVまたはSinCos N信号。 ALT2：HALL_V – デジタル・ホール・センサーのV入力 ALT3：DIRECT_IN[0] – I/O コントローラ直接入力 0	VCC_IOF	Digital Input (Pull-Up/Pull-Down)/Analog Input
16	ENC_N	デフォルト：ENC_N – 第一ABNエンコーダのN入力（デジタル差動入力の非反転信号としても使用）。 ALT1：AINN_W – 差動入力と共に使用するための反転アナログ・ホールWまたはSinCos Y信号。 ALT2：HALL_W – デジタル・ホール・センサーのW入力 ALT3：DIRECT_IN[1]/DIRECT_OUT[1] – I/O コントローラの直接入力 1 および直接出力 1	VCC_IOF	Digital Input (Pull-Up/Pull-Down)/Digital Output/Analog Input
17	HALL_U	デフォルト：HALL_U – デジタル・ホール・センサーのU入力 ALT1：AINP_U – アナログ・ホールU／SinCos X入力（アナログ差動入力の非反転信号としても使用） ALT2：ENC2_A – 第二ABNエンコーダのA入力 ALT3：I2C_SDA – I/OコントローラのI²Cデータ入出力 ALT6：DIG_DIFF_ENC_A_MONITOR – デジタル差動入力用の第一ABNエンコーダの反転A信号のモニタリング。 ALT7：DIG_DIFF_ENC_A_N – デジタル差動入力と共に使用するための第一 ABN エンコーダの反転 A 信号。	VCC_IOF	Digital Input (Pull-Up/Pull-Down)/Digital Output/Analog Input
18	HALL_V	デフォルト：HALL_V – デジタル・ホール・センサーの V 入力 ALT1：AINP_V – アナログ・ホール V／SinCos N 入力（アナログ差動入力の非反転信号としても使用） ALT2：ENC2_B – 第二 ABN エンコーダの B 入力 ALT3：I2C_SCL – プッシュプル出力としての I/O コントローラ I2C クロック ALT4：BRAKE – ブレーキ・チョッパーPWM 出力 ALT5：I2C_SCL_OD – オープン・ドレイン出力としての I/O コントローラ I2C クロック ALT6：DIG_DIFF_ENC_B_MONITOR – デジタル差動入力用の第一 ABN エンコーダの反転 B 信号のモニタリング。 ALT7：DIG_DIFF_ENC_B_N – デジタル差動入力と共に使用するための第一 ABN エンコーダの反転 B 信号。	VCC_IOF	Digital Input (Pull-Up/Pull-Down)/Digital Output/Analog Input
19	HALL_W	デフォルト：HALL_W – デジタル・ホール・センサーの W 入力 ALT1：AINP_W – アナログ・ホール W／SinCos Y 入力（アナログ差動入力の非反転信号としても使用）	VCC_IOF	Digital Input (Pull-Up/Pull-Down)/Digital Output/Analog Input

		ALT2：ENC2_N – 第二 ABN エンコーダの N 入力 ALT3：DIRECT_OUT[2] – I/O コントローラ直接出力 2 ALT4：REF_H – ホーム・リファレンス・スイッチ入力 ALT6：DIG_DIFF_ENC_N_MONITOR – デジタル差動入力用の第一 ABN エンコーダの反転 N 信号のモニタリング。 ALT7：DIG_DIFF_ENC_N_N – デジタル差動入力と共に使用するための第一 ABN エンコーダの反転 N 信号。		
20	V_{CC_IOF}	モーター・ポジション・フィードバックおよびリファレンス・スイッチ I/O ピン用の電源入力。最高性能を発揮するには、100nF の低 ESR セラミック・コンデンサを短い配線パターンを用いてピンの近くで GND に接続します。		Supply
21, 26, 27	PGND	電源グラウンド。		GND
22	OUT3	ハーフ・ブリッジ出力 3 – BLDC モーター相 W に接続するか、DC モーターの場合は無接続にします。	VS	Analog Output
23, 24, 29	V_S	モーター電源電圧入力。最高性能を発揮するには、100nF の低 ESR セラミック・コンデンサを短い配線パターンを用いてピンの近くで GND に接続します。		Supply
25	OUT2	ハーフ・ブリッジ出力 2 – BLDC モーター相 V または DC モーターに接続します。	VS	Analog Output
28	OUT1	ハーフ・ブリッジ出力 1 – BLDC モーター相 W または DC モーターに接続します。	VS	Analog Output
30	NC	内部接続されていません。		
31	V_{CP}	チャージ・ポンプ出力。V_{CP} と V_S の間に 1µF のコンデンサを接続します。		Analog Output
32	CPO	チャージ・ポンプ・フライング・コンデンサのハイ端子。CPO と CPI の間に 47nF のコンデンサを接続します。		Analog Output
33	CPI	チャージ・ポンプ・フライング・コンデンサのロー端子。CPO と CPI の間に 47nF のコンデンサを接続します。		Analog Output
34	DRV_EN	ドライバ・イネーブル入力（アクティブ・ハイ）。	VCC_IO	Digital Input (Pull-Down)
35	FAULTN	デフォルト：FAULTN_OD – 設定可能なフォルト出力（オープン・ドレイン、アクティブ・ロー） ALT1：INT_OD – 設定可能な割込み／イベント出力（オープン・ドレイン） ALT2：FAULTN_PP – 設定可能なフォルト出力（プッシュプル、アクティブ・ロー） ALT3：INT_PP – 設定可能な割込み／イベント出力（プッシュプル）	VCC_IO	Digital Output (Pull-Up)
36	SCK	デフォルト：SCK – SPIクロック入力 ALT3：DIRECT_IN[1] – I/O コントローラ直接入力 1	VCC_IO	Digital Input (Pull-Up/Pull-Down)
37	SDI	デフォルト：SDI – SPIシリアル・データ入力 ALT1：DIR_IN – 方向制御入力（I/Oコントローラ使用） ALT3：DIRECT_IN[0] – I/O コントローラ直接入力 0	VCC_IO	Digital Input (Pull-Up/Pull-Down)
38	SDO	デフォルト：SDO – SPIシリアル・データ出力 ALT1：BRAKE – ブレーキ・チョッパーPWM出力 ALT3：DIRECT_OUT[0] – I/O コントローラ直接出力 0	VCC_IO	Digital Output (Pull-Up)
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用の GND ピンとして機能します。		GND

機能図

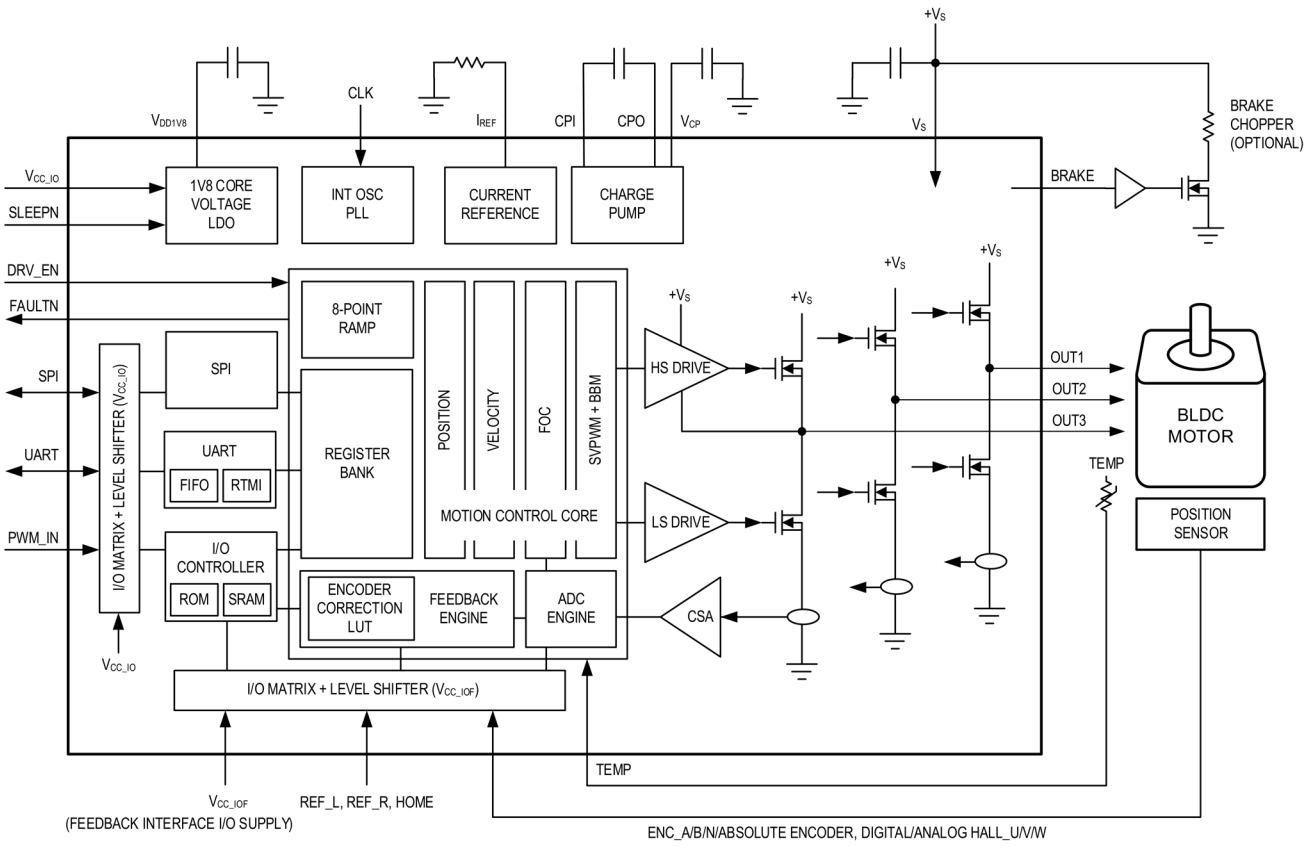


図 2. 機能図

機能説明

TMC6460 は、ブラシレス DC（BLDC）モーターや三相永久磁石同期モーター（PMSM）向けの、36V 高性能ベクトル制御（FOC）サーボ駆動用 IC です。また、DC モーターも駆動できます。高集積かつ容易に導入可能な TMC6460 を用いることで、センサーベースのモーション・コントロール向け FOC の設計を簡素化できます。このフル機能ソリューションは、学習曲線を低く抑えながら、クラス最高の性能を発揮します。

TMC6460 には、ホスト・コントローラとの通信に必要なペリフェラル・インターフェースや、柔軟な IO インターフェースおよび汎用モーター角度フィードバック・エンジンが内蔵されており、ほとんどのアプリケーション・ニーズに適応できます。

TMC6460 は、FOC の変換を実行するための強力なモーション・コントロール・コア（MCC）を備えています。MCC には、トルク、速度、ポジションを制御するための、フル機能制御ループ・カスケードが内蔵されています。リアルタイムの重要なタスクはすべてハードウェアに実装されているため、高速の制御ループが可能です。MCC の PWM エンジンは、低インダクタンス・モーターやコアレス・モーターを使用する高精度サーボ・アプリケーション向けに、高速のチョッピング周波数をサポートしています。統合化されている 8 点ランプ・ジェネレータが MCC に給電しているため、極めて動的なポジショニングや、滑らかな遷移と最小限のジャークでの速度ランピング・モーションを実現します。

TMC6460 は、低インピーダンス FET で構成された 3 個の 36V 対応ハーフ・ブリッジを備えているため、駆動効率が高く、熱生成を低く抑えることができます。TMC6460 の合計の R_{ON} は 110mΩ（ハイサイド＋ローサイド）であり、最大 $3A_{RMS}$ のモーターを駆動できます。

TMC6460 では、ローサイド FET 測定に基づく正確な電流アクイジション機能が統合化されています。内部電流検出機能を備えていることから、大きな外付け電力抵抗が不要となり、外部センス抵抗を用いるアプリケーションと比べ、スペースと消費電力を大幅に節減できます。

TMC6460 は、包括的な測定および診断機能の他、堅牢な故障検出や保護機構も備えています。それによって、連続的なシステム・モニタリングが可能になり、エンド・アプリケーションにおける確実なモーター動作を実現できます。

TMC6460 は、5mm × 7mm の 38 ピン TQFN パッケージを採用しています。

通信インターフェースおよび制御インターフェース

TMC6460 は、ホスト・コントローラとの通信を行うための SPI インターフェースおよび UART インターフェースを備えています。更に、単線制御モードにも対応しています。

シリアル・ペリフェラル・インターフェース（SPI）

TMC6460 は、ホスト・マイクロコントローラ・ユニット（MCU）と通信を行うシリアル・ペリフェラル・インターフェース（SPI）を 1 つ備えています。これは SPI モード 1（CPOL＝0 および CPHA＝1）を使用し、入力（SDI）は SPI クロック（SCK）の立下がりエッジでサンプリングされ、出力（SDO）は、SCK の立上がりエッジで生成されます。この SPI は最大 8MHz のクロック周波数に対応します。

SPI は、複数の TMC6460 デバイスのデイジーチェーン接続をサポートします。

UART インターフェース

TMC6460 の UART は、業界標準の NRZ 非同期シリアル・データ・フォーマットを使用する外部デバイスとの全二重データ交換をサポートしています。このインターフェースは、1 つのスタート・ビット、8 つのデータ・ビット、1 つのストップ・ビットを使用し、極性ビットはありません。

UART インターフェースは、多様なボー・レートの他、自動ボー・レート検出（オートボー）もサポートしています。高速の周期的データ・ストリーミングを可能にするリアルタイム・モニタリング・インターフェース（RTMI）も組み込まれています。

UART インターフェースは、巡回冗長検査（CRC）エラー検出に対応しています。

また、SPI が不要な場合には、UART インターフェースをホスト MCU との間のメインの通信チャンネルとして用いることができます。

あるいは、UART インターフェースを SPI と並列に使用することもできます。これは、RTMI を介したリアルタイムの調整やモニタリングを行う場合などに有用となります。この場合、UART インターフェースの TXD ピンのみを接続する必要があります。

単線式インターフェース

TMC6460 は、PWM_IN 入力ピンを用いて、単線式インターフェースとして動作させることができます。単線式入力信号は、可変アナログ電圧と可変デューティサイクル PWM 信号のどちらかです。したがって、入力信号は、例えばコントローラの速度目標またはポジション目標を指定するなど、レジスタ・バンクの任意のフィールドに対する制御変数として使用できます。

単線式入力値は、内蔵の I/O コントローラを通じて処理されます。入力信号のスケール、制御目標のスケール、書込み先のレジスタ・フィールドは、すべて、個別に設定できます。

オプションで、DIR_IN 入力を第二の信号として用いて、制御変数の符号を定義することもできます。この機能の初期設定をするために、UART と SPI のどちらかのインターフェースが依然として必要である点に注意してください。

フィードバック・エンジン

TMC6460 は、汎用のフィードバック・エンジンをインターフェースと統合化し、下記の様々なタイプのセンサーに対応します。

- デジタル・ホール・センサー
- ABN インクリメンタル・エンコーダ
- アナログ・ホール・センサー
- アナログ SinCos 絶対エンコーダ
- SPI/SSI 絶対エンコーダ（内蔵 I/O コントローラを使用）

また、フィードバック・エンジンは、以下に例を示すように、デュアル・フィードバック・セットアップ用に設定することもできます。

- 通信用デジタル・ホール＋速度およびポジショニング用 ABN。
- モーターでの第一 ABN＋ギアボックス後段の第二 ABN。
- モーターでの ABN＋ギアボックス後段の SSI エンコーダ。

またフィードバック・エンジンは以下も内蔵しています。

- 速度測定モジュール
- マルチターン・ポジション・カウンタ（32 ビット）

内蔵 I/O コントローラにカスタム・プログラムをアップロードして、BiSS-C や多摩川精機 T フォーマットなどのその他のエンコーダ・プロトコルを TMC6460 と統合することもできます。

更に、一般的なレジスタ・アクセスで角度データを書き込んで、任意の外部角度フィードバック・ソースにインターフェースすることもできます。例えば、ホスト MCU 自体が角度センサーに接続されている場合などが該当します。

モーター制御コア

TMC6460 の MCC は、フル機能の制御ループ・アーキテクチャ、つまりトルク、速度、ポジションの各制御ループを内蔵しています。MCC は、FOC のすべての重要なリアルタイム・タスクをユーザ・アプリケーション層から切り離します。図 3 に、MCC の FOC アーキテクチャを簡略化して示します。

FOC トルク PI コントローラは、MCC の基盤コンポーネントを形成します。これには、次の変換が含まれます：クラーク、パーク、逆パーク、逆クラーク、空間ベクトル PWM（SVPWM）。MCC は電流測定値およびモーター角度情報を受信し、相電圧信号を PWM エンジンに出力します。FOC アルゴリズムは、磁束（I_D）とトルク（I_Q）の電流成分を個別に制御するため、高効率で静音のモーター動作が実現できます。

PWM エンジンは、1.83kHz～200kHz の範囲で設定できる、ユーザ・プログラマブルなスイッチング周波数を特徴としています。PWM 変調は、中央揃えのチョッピング手法、最高電圧を利用するための設定可能 SVPWM モード、および組み込み電力段のためのブレーク・ビフォア・メーク（BBM）デッドタイム・ジェネレータに基づいています。

内部の FOC トルク・ループに加え、外部の速度およびポジション PI コントローラが実装されます。

最後に、TMC6460 には 8 点ランプ・ジェネレータが統合されています。1 つの目標ポジションまたは目標速度に基づき、ランプ・ジェネレータは速度と加速度のパラメータ・セットに基づくモーション・プロファイルを自動的に計算します。ランプ・ジェネレータは、最大 6 個の個別の加減速フェーズと一定の速度セグメント 1 つからなる速度モーション・プロファイルの計算を行います。この方法により、遷移が滑らかでジャークが最小限に抑えられた、極めて動的なポジションまたは速度モーションを実現できます。ランプ・ジェネレータの出力は、直接制御目標やフィードフォワード入力など、制御カスケード内の様々なループに転送されます。

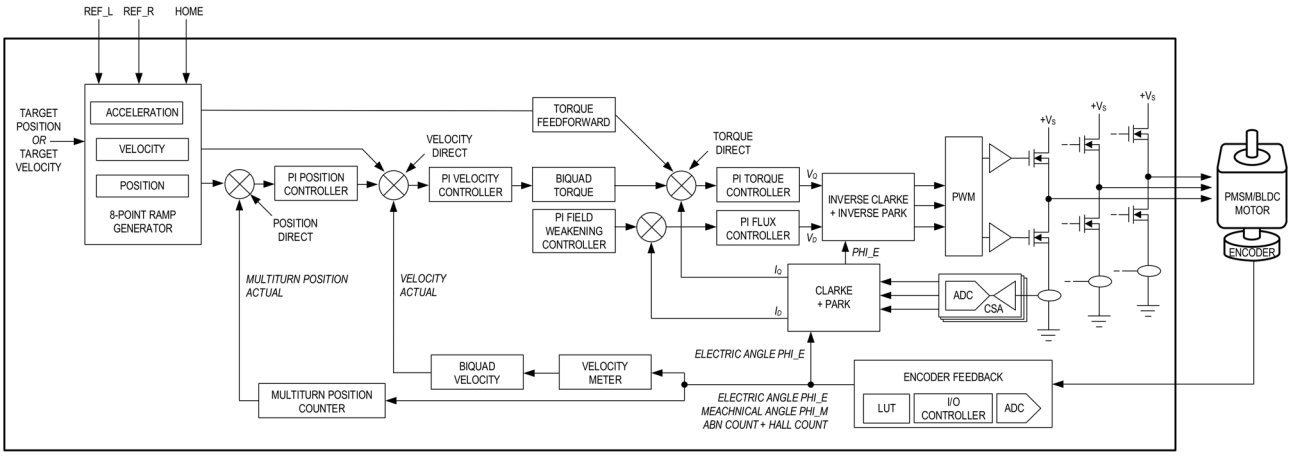


図 3. FOC およびモーション・コントロール・コア

電力出力段

TMC6460 は、低インピーダンス FET を用いる 3 個の 36V 対応ハーフ・ブリッジを内蔵しています。合計の R_{ON} は 110mΩ（ハイサイド + ローサイド）であり、電力段は、高い駆動効率を実現し発熱を最小限に抑えることができます。

TMC6460 は、相あたり最大 $3A_{RMS}$ の電流でモーターを駆動できます。最大実効値電流は主として熱に関する考慮事項によって制限されます。この考慮事項は、PCB のグラウンド・プレーン、ヒートシンク、換気など、アプリケーションの PCB レイアウトと熱特性に大きく依存します。$3A_{RMS}$ という定格値は、室温（T_A = 25ºC）で動作する 4 層 2s2p PCB に対する JEDEC 規格に基づいています。

ハーフ・ブリッジには 4 通りの出力スルー・レートを設定できるため、電磁妨害波と駆動効率の間の最適な妥協点を見出せます。

電流アクイジション

TMC6460 は電流検出機能を内蔵しているため、外付け電力抵抗が不要になり、スペースと消費電力の大幅な節減ができます。代わりにパワーFET を電流検出部品として用いており、それによってこの機構が消費電力を増加させることはありません。

TMC6460 は可変電流検出スケーリングに対応しています。検出部品の等価抵抗（$R_{ON,LS}$）は、精度および分解能を最適化するよう、様々なモーター・サイズおよびアプリケーションに合わせて 4 通りの値の中から選択できます。

更に、電流検出アンプのゲイン係数を設定して、アナログ／デジタル変換の間の等価範囲を拡張できます。4 通りのゲイン値が選択できます。この設定は、動作中に変更できます。例えば、モーター電流がピークに達するような強い加速の場合には低ゲイン設定を用い、モーターが一定速度で動作するために電流要求が減少する場合には高ゲイン設定を用いることができます。

診断、フォルト処理、保護

TMC6460 は、包括的な測定および診断機能を備えています。モーターの実際のポジションや速度、およびモーターの電圧、電流、電力といったパラメータはすべて、モニタリングによって容易に入手できます。制御カスケードや FOC 変換に関係するほとんどの変数も、クエリによって入手できます。

豊富な種類の状態フラグやイベント・ビットは、システムの状態の概要を示します。また、これらによって、効率的なエラー検出やグローバル診断も可能になります。

TMC6460 には過電流保護（OCP）機構が備わっており、外部の短絡回路や大電流ピークを検出した場合に内部 FET を保護できます。OCP がトリガされるスレッショルドは、設定可能です。

その他、電源の過電圧、IC の過熱、取り付けた温度センサーの過熱などのイベントによりトリガされる保護機構も、TMC6460 に備わっています。

これらの全機能により、連続的なシステム・モニタリングが可能になり、エンド・アプリケーションでの確実なモーター動作を実現できます。

詳細説明

PLL およびクロックの設定

TMC6460 はデフォルトで、f_{OSC} = 15MHz の内部発振器をメインのクロック・ソースとして用います。内蔵のフェーズ・ロック・ループ（PLL）は、この内部発振器の信号を用いて f_{PLL} = 120MHz のクロックを生成し、それを基にその他のクロック信号が生成されます。モーター出力を駆動する PWM 信号を生成するために用いられる PWM クロックは、PLL から直接生成されるため、120MHz で稼働します。システム・クロックは PLL 周波数の半分の値に生成され、f_{SYS} = 60MHz で稼働します。デフォルトのクロック設定は、パワーアップ時およびリセット後に自動的に実行されます。それ以上の設定は不要です。

システム・クロックは、ランプ・ジェネレータの動作など、一部のタイミング動作のために用いられます。したがって、内部発振器の精度（電気的特性の表を参照）が特定の用途に対して不十分な場合は、より正確な外部クロック信号を CLK ピンに供給できます。外部クロック信号を PLL のソースとして用いるには、CLK_CTRL.CONFIG レジスタの PLL_SRC ビットフィールドに 1_b の値を設定します。更に、PLL に 1MHz のクロックが入力されるよう、クロック分周器を設定する必要があります。これを行うには、同じレジスタの CLOCK_DIVIDER フィールドに、f_{EXT_CLK}/1MHz − 1 の値を設定します。例えば、外部クロック周波数が f_{EXT_CLK} = 32MHz の場合、31 の値を使用します。通常動作の場合、CLK_CTRL.CONFIG レジスタの CLK_FSM_EN、PWM_CLK_EN、ADC_CLK_EN、PLL_EN の各ビットをすべてセットする必要があります。新しいクロック設定を適用するには、同じレジスタの COMMIT ビットに 1_b を書き込みます。CLK_CTRL.CONFIG レジスタへの 1 回の書込みアクセスで、すべての設定を完了できます。

例えば、設定の誤りや外部クロック信号の欠損などにより、クロック・エラーまたは PLL エラーが発生した場合、CHIP.STATUS_FLAGS レジスタの CLK_PLL_FAIL_STATUS ビット、およびこれに対応する CHIP.EVENTS レジスタの CLK_PLL_FAIL_EVENT ビットによって、エラーの発生が示されます。フォルトが検出されると、PLL は自動的にディスエーブルされ、システム・クロック・ソースは、15MHz の内部発振器に戻されます。エラー原因が解消したら、CLK_PLL_FAIL_EVENT ビットを手動でクリアする必要があります。CLK_CTRL.CONFIG レジスタの設定をチェックし、COMMIT ビットを再度 1_b に設定します。これにより、所定の設定でクロックと PLL が再始動します。

クロックの有限ステート・マシン（FSM）の最新のステータスは、CLK_CTRL.STATUS レジスタからクエリできます。

パワー・マネージメント

LP_MODE_EN を 1_b に設定すると、TMC6460 は低電力／スタンバイ・モードになります。このモードでは内部 PLL がディスエーブルされます。したがって、内部動作はすべて、内部発振器のクロック周波数 f_{OSC} = 15MHz で実行されます。更に、PWM、ADC、I/O コントローラの動作は停止し、したがって電力の消費はありません。

デジタル・エンコーダとホール・センサーの各インターフェースは、低電力モードでも使用可能な状態であるため、外部で実行されるモーターの動作を追跡できます。同様に、UART や SPI のインターフェースを用いる通信も、このモードでは使用可能な状態です。

低電力モードを終了するには、LP_MODE_EN を 0_b にセットし、DRV_EN ピンをアサートするか、レジスタ・アクセスにより I/O コントローラをリセットします。

更に消費電流を低減するには、SLEEPN ピンをアサートして TMC6460 をスリープ・モードに設定します。スリープ・モードの間、UART や SPI による通信は使用できず、レジスタの内容は保持されない点に注意してください。

スタンバイ・モード時およびスリープ・モード時の消費電流については、電気的特性の表を参照してください。

フォルト管理

通常のパワーアップ・シーケンスの後、TMC6460 はオフ状態になります。この状態では、すべての機能が正常に動作しますが、出力段のみディスエーブルのままになります。この状態では、CHIP.STATUS_FLAGS レジスタの SYS_READY_STATE ビットがセットされます。出力段をイネーブルし、ドライバ・オン状態にするには、MCC_CONFIG.GDRV レジスタの DRV_EN_BIT ビットフィールドをセットし、DRV_EN ピンをアサートします。すべての条件が満たされると、IC はドライバ・オン状態になり、CHIP.STATUS_FLAGS レジスタの GDRV_ON_STATE ビットがセットされ、SYS_READY_STATE ビットが自動的にクリアされます。ドライバ・オン状態では、TMC6460 のすべての機能を正常に使用できます。

様々なフォルト条件によって、TMC6460 はドライバ・オン状態を終了し出力段をディスエーブルする可能性があります。CHIP.STATUS_FLAGS レジスタおよび CHIP.EVENTS レジスタは、エラーの原因を探るためのリファレンス・ポイントです。何らかのフォルト条件を検出すると、DRV_EN_BIT ビットフィールドが自動的にクリアされます。エラーが解消した後、このビットを再度セットして、出力段をイネーブルしてください。エラーを解消する操作には、いくつかのフォルト・フラグをクリアして、その他の要素を再イネーブルすることを意味する場合がある点に注意してください。

ADC の再初期化

CHIP.STATUS_FLAGS レジスタの ADC_FAIL_STATUS ビットによって示される ADC フォルトが発生すると、ADC エンジンを再初期化することが必要になる可能性があります。この再初期化を行うには、IO_CONTROLLER.CONTROL レジスタの RESET ビットを通じて I/O コントローラをリセットします。

ステータス・フラグとイベント

ステータス・フラグは、特定のエラー状態の最新の状態を表します。これらは、エラー状態が解消するとクリアされます。イベントが発生すると、その対応するステータス・フラグの値がラッチされます。これらは手動でクリアする必要があります。それを行うのは、対応するステータス・フラグも非アクティブになった場合にのみ可能です。

イベントは、最初のパワーアップが正常に実行されるまでは生成されませんが、ステータス・フラグは常に読み出せます。

表 1 に CHIP.STATUS_FLAGS レジスタのステータス・フラグと CHIP.EVENTS レジスタのイベントを示します。CHIP.STATUS_FLAGS の PWRUP_FAIL_STATUS ビットには対応するイベント・ビットがない点に注意してください。代わりに、CHIP.EVENTS のビット 0 は RST_EVENT ビットに対応します。それ以外のすべての場合に対し、機能／コメントの欄はステータス・フラグについての記述です。

表 1. 概要：ステータス・フラグとイベント

ステータス・フラグ	ビット	対応イベント	機能／コメント
	0	RST_EVENT	リセットが発生。（EVENTSのみ。）
PWRUP_FAIL_STATUS	0		パワーアップ時にエラーが発生。電源再投入が必要です。（STATUS_FLAGSのみ。）
SPI_FAIL_STATUS	1	SPI_FAIL_EVENT	SPIタイムアウト・エラーまたは割込みエラーがアクティブ。CSNをハイ、SCKをローにセットすることによりSPIがリセットされ、次のデータグラムに対する準備が整います。
UART_FAIL_STATUS	2	UART_FAIL_EVENT	UARTエラー・ビットがアクティブ。UART_RXDをハイにセットし70ms以上待機することによりUARTインターフェースがリセットされ、次のデータグラムに対する準備が整います。
VCC_IOF_FAIL_STATUS	3	VCCIOF_FAIL_EVENT	V_{CC_IOF}電源電圧が1.85V未満。
CP_FAIL_STATUS	4	CP_FAIL_EVENT	チャージ・ポンプが非アクティブ。
UV_VM_STATUS	5	UV_VM_EVENT	V_S電源電圧が2.2V未満。
POWER_FAIL_STATUS	6	POWER_FAIL_EVENT	ドライバ段の電源電圧が予想最小値未満。
CLK_PLL_FAIL_STATUS	7	CLK_PLL_FAIL_EVENT	クロック発生故障がアクティブ。システム・クロックは内部発振器f_{SYS} = 15MHzを使用します。 デフォルトのクロック設定に戻すには、CLK_CTRL.CONTROLレジスタを手動で再設定する、I/Oコントローラをリセットする、ICの電源を再投入する、のいずれかを行います。
EXT_RES_FAIL_STATUS	8	EXT_RES_FAIL_EVENT	外部リファレンス抵抗は57kΩ～63kΩの予想範囲外。内部抵抗を用いる場合はアクティブではありません。
ADC_FAIL_STATUS	9	ADC_FAIL_EVENT	いずれかのADCが非アクティブ。ADCを再設定するには、I/Oコントローラをリセットするか、ICの電源を再投入します。
TEMP_LIMIT_FAIL_STATUS	10	TEMP_LIMIT_FAIL_EVENT	内部チップ温度が165ºCを超過。
VEL_FAIL_STATUS	11	VEL_FAIL_EVENT	電力段の出力周波数が599Hz（1秒あたりの電気的な回転数）の制限値を超過。ドライバ段はディスエーブルされます。
SHRT_FAIL_STATUS	12	SHRT_FAIL_EVENT	いずれかのドライバ出力において、地絡またはV_Sへの短絡が3連続PWMサイクル以上にわたり検出。ドライバ段はディスエーブルされます。
OT_FAIL_STATUS	13	OT_FAIL_EVENT	いずれかのドライバ段チャンネルの内部温度が165ºCを超過。
DIFF_ENC_FAIL_STATUS	14	DIFF_ENC_FAIL_EVENT	不適切なデジタル差動エンコーダ入力。

IO_CONTROLLER_STATUS	15	IO_CONTROLLER_EVENT	I/Oコントローラが非アクティブ、または、手動で割り当てられたI/Oコントローラ・フラグがアクティブ。
RAMP_STALL_FAIL_STATUS	16	RAMP_STALL_FAIL_EVENT	ランプ・ジェネレータのポジションまたは速度の最大偏差がトリガ。STALL_IN_POSITION_ERRおよびSTALL_IN_V_ERRを参照してください。
RAMP_TARGET_ REACHED_STATUS	17	RAMP_TARGET_ REACHED_EVENT	ランプ・ジェネレータの目標ポジションに到達。POSITION_REACHED_STATUSを参照してください。
RAMP_V_REACHED_STATUS	18	RAMP_V_REACHED_EVENT	ランプ・ジェネレータの目標／最大速度値に到達。V_REACHED_STATUSを参照してください。
RAMP_V_ZERO_STATUS	19	RAMP_V_ZERO_EVENT	ランプ・ジェネレータの速度値がゼロに等しい。V_ZEROを参照してください。
RAMP_REF_LR_STATUS	20	RAMP_REF_LR_EVENT	ランプ・ジェネレータの左／右リファレンス・スイッチ信号がアクティブ。RAMP_REF_L_STATUSおよびRAMP_REF_R_STATUSを参照してください。
RAMP_REF_H_STATUS	21	RAMP_REF_H_EVENT	ランプ・ジェネレータのホーム・リファレンス・スイッチ信号がアクティブ。RAMP_REF_H_STATUSを参照してください。
ABN_FAIL_STATUS	22	ABN_FAIL_EVENT	ABNとABN2の複合エラー（これらのエンコーダ・インターフェースがイネーブルの場合）。
HALL_FAIL_STATUS	23	HALL_FAIL_EVENT	ホール・センサー・デコーダの無効な組み合わせが検出。
CURRENT_OVERLOAD_STATUS	24	CURRENT_OVERLOAD_ EVENT	絶対相電流値が手動定義された制限値を超過。
I_CLIPPED_STATUS	25	I_CLIPPED_EVENT	測定された相電流値がクリップされています。
AIN_CLIPPED_STATUS	26	AIN_CLIPPED_EVENT	測定されたアナログ値がクリップされています。
VM_TEMP_CLIPPED_STATUS	27	VM_TEMP_CLIPPED_EVENT	測定された温度またはV_Sの電圧がクリップされています。
OV_VM_LIMIT_FAIL_STATUS	28	OV_VM_LIMIT_FAIL_EVENT	測定されたV_S電圧値が手動定義された制限値を超過。
OVERTEMP_WARN_STATUS	29	OVERTEMP_WARN_EVENT	測定された外部または内部温度値が手動定義された制限値を超過。
SYS_READY_STATE	30	SYS_READY_EVENT	システム・レディ状態。ドライバ段はディスエーブルされます。
GDRV_ON_STATE	31	GDRV_ON_EVENT	ドライバ・オン状態。ドライバ段はイネーブルされます。

この表の一部の項目には、本マニュアルの関連セクションへの参照を伴うリンクがあります。

FAULTN ピン

TMC6460 には、CHIP.STATUS_FLAGS レジスタまたは CHIP.EVENTS レジスタのビットに対応する FAULTN ピンを駆動する機能があります。FAULTN ピンを駆動するリファレンスとしてステータス・フラグを用いるかイベントを用いるかは、そのピンに対する代替機能の割り当てによって決まります。これは、表 2 に示すように、CHIP.IOMATRIX レジスタの PIN_FAULTN フィールドを通じて設定されます。

表 2. FAULTN ピンの代替機能の割り当て

PIN_FAULTNの選択	FAULTNピンの機能
FAULTN_OD	CHIP.STATUS_FLAGSを参照するオープン・ドレイン出力設定。
INT_OD	CHIP.EVENTSを参照するオープン・ドレイン出力設定。
FAULTN_PP	CHIP.STATUS_FLAGSを参照するプッシュプル出力設定。
INT_PP	CHIP.EVENTSを参照するプッシュプル出力設定。

ピンの代替機能の割り当ての他、ビット選択に対する適切なマスクを設定する必要があります。これは、CHIP.FAULTN_INT_MASK レジスタの FAULTN_INT_MASK フィールドを用いて行います。マスク・フィールドのビットを 1_b にセットすると、ピン機能の割り当てに応じて CHIP.STATUS_FLAGS レジスタまたは CHIP.EVENTS レジスタの対応するビットが選択されます。マスクにおいて 2 個以上のビットが設定されている場合、対応するビットすべてが FAULTN ピンに OR 接続されます。

例えば、SHRT_FAIL_STATUS ビットのみを FAULTN 出力ピンに出力する必要がある場合、FAULTN_INT_MASK の値を 0x1000 に設定し、PIN_FAULTN を FAULT_OD または FAULT_PP に設定します。同様に、FAULTN 出力ピンが CP_FAIL_EVENT ビットまたは VCCIOF_FAIL_EVENT ビットのいずれかを出力する場合、FAULTN_INT_MASK の値を 0x18 に設定し、PIN_FAULTN を INT_OD または INT_PP に設定します。

FAULTN_INT_MASK フィールドには 0_b のリセット値があります。これは、どのステータス・フラグまたはイベントもデフォルトでは FAULTN ピンに反映されないことを意味します。

パワーアップ時には、FAULTN ピンは、パワーアップ・シーケンスが進行中であることを示すようアサートされます。パワーアップ処理が正常に終了すると、FAULTN ピンはデアサートされ、上述の設定に置き換えられます。

シリアル・ペリフェラル・インターフェース（SPI）

TMC6460 の SPI は、シンプルなコマンドおよび制御構造を採用しています。これは SPI モード 1（CPOL = 0 および CPHA = 1）を使用し、入力（SDI）は SPI クロック（SCK）の立下がりエッジでサンプリングされ、出力（SDO）は、SCK の立上がりエッジで生成されます。有効な SPI コマンドは、少なくとも 48 ビット長であり、8 の倍数（48、56、64 など）でなくてはなりません。SPI コマンドのビット数が 8 の倍数でない場合、または 48 ビットより短い場合、コマンドは無視され CHIP.STATUS_FLAGS レジスタの SPI_FAIL_STATUS ビットがセットされます。

SPI はタイムアウト機構に対応しています。チップ・セレクト・ノット（CSN）の立下がりエッジ後 t_{CCR_MAX} = 16µs 以内に SCK の立上がりエッジが生じないとタイムアウトが発生し、この場合も SPI_FAIL_STATUS ビットがセットされます。低電力モードでは、t_{CCR_MAX} は 67µs に増加します。

SPI を介した TMC6460 との通常の通信は、48 ビットのデータグラムで行われます。各データグラムでは、1 つの要求が SDI を通じて送られ、同時に、その前の要求の返答が SDO を通じて送り返されます。SPI データグラムの要求は、繰り返し要求されるのがわずか 8 ビットであっても、受信 SCK サイクルごとに定期的に応答されます。そのため、SPI メイン・デバイスがステータス・ビットをポーリングできます。このビットは応答データグラムで常に最初に送信されます。

SPI 読出し要求データグラム

読出し要求コマンドは、レジスタの内容を読み出すために用いられます。読出し要求に対する応答は、次に発行されるコマンドの応答データグラムとして提供されます。図 4 に SPI 読出し要求データグラムの構造を示します。

- MSB（ビット 47）が最初に送信され、LSB（ビット 0）が最後に送信されます。
- MSB（ビット 47）は WRITE_notREAD（WnR）ビットであり、読出し要求の場合は 0 にする必要があります。
- ビット 46～42 は予約済みビットであり、常に 0 でなくてはなりません。
- ビット 41～32 はアドレス・ビット（10 ビット・アドレス）です。
- ビット 31～0 は読出しデータグラムとは無関係です（32 ビットのダミー・データ）。

SPI READ REQUEST DATAGRAM			
each byte is MSB to LSB			
47 0			
WnR bit	**Reserve bits**	**10-bit address**	**Dummy (32-bit data)**
0	00000	address [9...0]	xxxxx
47	46 42	41 32	31 0

図 4. SPI 読出し要求データグラムの構造

SPI 読出し要求データグラムの一例は 0x014000000000 です。これは、アドレス 0x140（FOC.PID_CONFIG）を読み出すために送信されます。

SPI 読出し応答データグラム

読出し応答には、リード・バックされるレジスタの内容が含まれます。これは、読出し要求コマンド後の次に発行されたコマンドの応答データグラムとして送信されます。例えば、書込みコマンドを発行せずにレジスタの内容を読み出すには、2 つの読出しコマンドを連続して送信します。それにより応答は、2 番目の読出しコマンドの応答データグラムに含まれます。図 5 に SPI 読出し応答データグラムの構造を示します。

- MSB（ビット 47）が最初に送信され、LSB（ビット 0）が最後に送信されます。
- ビット 47～42 は 6 個のステータス・ビットであり、CHIP.SPI_STATUS_MASK レジスタを用いることにより CHIP.EVENTS レジスタから選択したビットを含むことができます。
- ビット 41～32 は、読出し要求データグラムからの 10 ビットのアドレスです。
- ビット 31～0 は、指定されたアドレスからの 32 ビットの応答データです。

SPI READ RESPONSE DATAGRAM		
each byte is MSB to LSB		
47 0		
Status data[5:0]	**Address[9:0]**	**Reply data[31:0]**
xxxxxx	xxxxxx	xxxxx
47 42	41 32	31 0

図 5. SPI 読出し応答データグラムの構造

図 6 に SPI データグラムの例を示します。メイン・デバイスは、従属デバイスを選択するよう CSN をアサートし、データ転送を同期するよう SCK にクロック・パルスを発生します。各クロック・エッジは、バイトごとに番号付きのポジション（7～0）として示される 1 ビットを、ハイ状態またはロー状態として表されるロジック・レベルでシフトします。SDI では、メイン・デバイスがダミーの読出し要求をレジスタ 0x00 に送信し、SDO では、従属デバイスが、直前の要求に対し、ステータス・ビット、アドレス（0x140）、32 ビットの応答データからなる応答を返します。

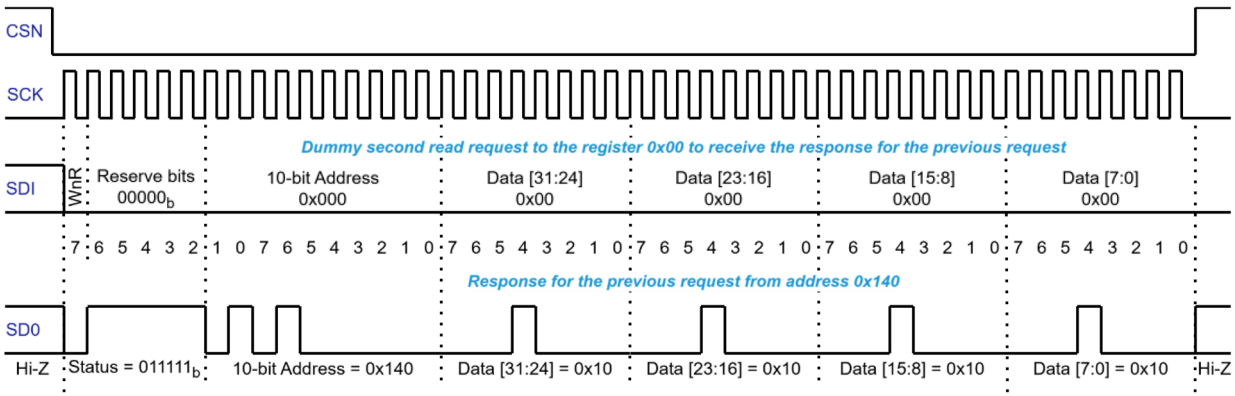


図 6. 直前の読出し要求データグラムからデータを受信するための SPI ダミー読出し要求

SPI 書込み要求データグラム

書込み要求コマンドは、レジスタの内容を変更するために用いられます。書込み要求に対する応答は、次に発行されるコマンドの応答データグラムとして提供されます。図 7 に SPI 書込み要求データグラムの構造を示します。

- MSB（ビット 47）が最初に送信され、LSB（ビット 0）が最後に送信されます。
- MSB（ビット 47）は WRITE_notREAD（WnR）ビットであり、1 にする必要があります。
- ビット 46～42 は予約済みビットであり、0 でなくてはなりません。
- ビット 41～32 はアドレス・ビット（10 ビット・アドレス）です。
- ビット 31～0 は、定義されたアドレスでレジスタに書き込まれるデータ・ビット（32 ビット・データ）です。

SPI WRITE REQUEST DATAGRAM			
each byte is MSB to LSB			
47 0			
WnR bit	**Reserve bits**	**10-bit address**	**Write data (32-bit data)**
1	00000	address [9...0]	xxxxx
47	46 42	41 32	31 0

図 7. SPI 書込み要求データグラムの構造

SPI 書込み要求データグラムの一例は 0x80C300000002 です。これは、アドレス 0x0C3（MCC_ADC.CSA_GAIN）に 0x00000002 というデータを書き込みます。

SPI 書込み応答データグラム

書込み応答には、書き込まれるレジスタのリードアフターライト・データが含まれます。これは、書込み要求コマンド後の次に発行されたコマンドの応答データグラムとして送信されます。例えば、2 番目の書込みコマンドを発行せずに書込み要求の応答を受信するには、読出し要求コマンドを送信します。それにより応答は、読出しコマンドの応答データグラムに含まれます。図 8 に SPI 読出し応答データグラムの構造を示します。

- MSB（ビット 47）が最初に送信され、LSB（ビット 0）が最後に送信されます。
- ビット 46～42 はステータス・ビットであり、CHIP.SPI_STATUS_MASK の設定に基づき CHIP.EVENTS レジスタから選択したビットを含むことができます。
- ビット 41～32 は、書込み要求データグラムからのアドレスです。
- ビット 31～0 は、指定されたアドレスからのリードアフターライト・データです。

SPI WRITE RESPONSE DATAGRAM		
each byte is MSB to LSB		
47 0		
Status data[5:0]	**Address[9:0]**	**Write data[31:0]**
xxxxxx	xxxxxx	xxxxx
47 42	41 32	31 0

図 8. SPI 書込み応答データグラムの構造

SPI のデイジーチェーン・サポート

TMC6460 の SPI は、複数のペリフェラルのデイジーチェーン接続をサポートします。この構成では、デイジーチェーン接続されたすべてのペリフェラルが CSN 信号および SCK 信号を共有します。ホスト・コントローラの SDO が最初の TMC6460 デバイスの SDI に接続され、次いで各デバイスの SDO がチェーン内の次のデバイスの SDI に接続されます。このシーケンスは、最後のペリフェラルの SDO がホスト・コントローラの SDI に接続されるまで続きます。図 9 に、3 個の SPI ペリフェラルと 1 個のホスト・マイクロコントローラ（MCU）を持つデイジーチェーン・セットアップの例を示します。

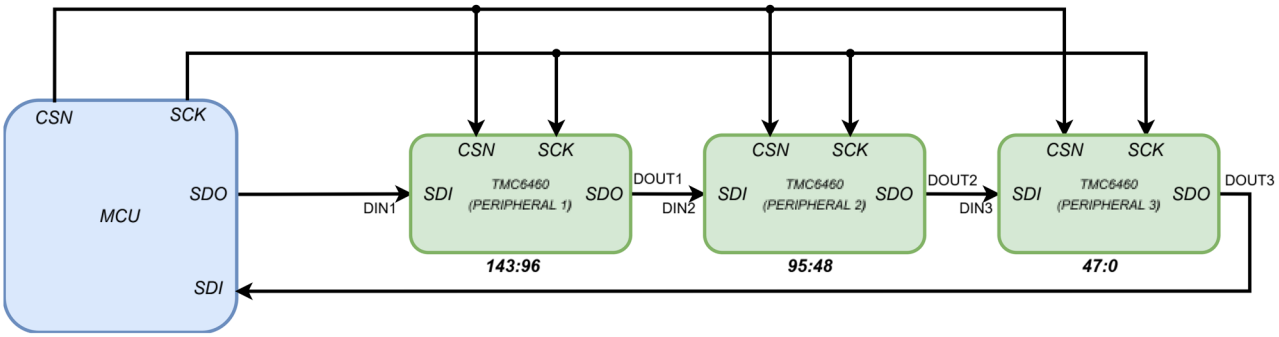


図 9. 3 個の SPI ペリフェラルを一列に接続した SPI デイジーチェーンの例

TMC6460 でデイジーチェーン動作を有効にするために追加の設定を行う必要はありません。CSN 信号をアサートしたまま複数のデータグラムをデバイス・チェーンに送信することによって、デイジーチェーン動作が行われます。TMC6460 の SPI が SDI で 48 ビットを超えるビット数を受信した場合、古いビットが SDO にシフト・アウトされ、次のペリフェラルに転送されます。CSN がデアサートされると、現在ペリフェラルに保持されているビットのみが考慮されます。トランザクションの最後でペリフェラルに保持されているデータグラムを SPI が正しく処理できるためには、データグラムが 48 ビット長を維持していなくてはなりません。図 9 には、1 つのデイジーチェーン・トランザクションの最後に各ペリフェラルで処理されるデータ・ビットも示されています。

図 10 は、3 つのペリフェラルのそれぞれから異なるレジスタを読み出すための、SPI トランザクションを表しています。実際のトランザクションでは、SCK およびデータ・ラインには、データグラムとデータグラムの間に割込みがない点に注意してください。これは見やすくするために示されているものです。以下の説明を参照してください。

- ペリフェラル 1 を通じて MCU からすべてのペリフェラルに要求データグラムが送信されます。
- SCK の最初の 48 サイクルで、ペリフェラル 1 が 47:0 からのデータグラム（0x004000000000）を受信します（DIN1）。
- SCK の 2 番目の 48 サイクルでは、ペリフェラル 2 がペリフェラル 1 の出力を受信し（DOUT1 から DIN2）、ペリフェラル 1 は 95:48 からのデータグラム（0x8080000CC003）を受信します（DIN1）。
- SCK の 3 番目の 48 サイクルでは、ペリフェラル 3 がペリフェラル 2 の出力を受信し（DOUT2 から DIN3）、ペリフェラル 2 はペリフェラル 1 の出力を受信し（DOUT1 から DIN2）、ペリフェラル 1 は 143:96 からのデータグラム（0x80C07FE20000）を受信します（DIN1）。

- SCK の 144 サイクルの最後では、ペリフェラル 1 に読出し要求データグラム 143:96、ペリフェラル 2 に読出し要求データグラム 95:48、ペリフェラル 3 に読出し要求データグラム 47:0 があります。
- CSN がデアサートされると、各ペリフェラルは、現在保持しているデータグラムを処理します。
- 次に応答データグラムが実行され、CSN がアサートされている間に 144 サイクルの SCK で同じ処理が行われます。今度は、各ペリフェラルの応答データグラムのみが関心の対象なので、ダミー・データグラム（0x000000000000）が送信されます。
- SCK の最初の 48 サイクルで、ペリフェラル 3 はその応答（0x004000000E3C）を MCU に送信し（DOUT3）、ペリフェラル 2 はその応答（0x0080000CC003）をペリフェラル 3 に送信し（DOUT2 から DIN3）、ペリフェラル 1 もその応答（0x00C07FE20000）をペリフェラル 2 に送信します（DOUT1 から DIN2）。
- SCK の 2 番目の 48 サイクルで、ペリフェラル 3 はペリフェラル 2 からの元の応答（0x0080000CC003）を MCU に送信し（DOUT3）、ペリフェラル 2 はペリフェラル 1 の元の応答（0x00C07FE20000）をペリフェラル 3 に送信します（DOUT2 から DIN3）。
- SCK の 3 番目の 48 サイクルで、ペリフェラル 3 はペリフェラル 1 の元の応答（0x00C07FE20000）を MCU に送信します（DOUT3）。その後、CSN がデアサートされます。

この例に示したダミー・データグラムの代わりとして、次の一連の読出しまたは書込み要求データグラムを送信して、通信プロセスをより効率的にすることもできます。

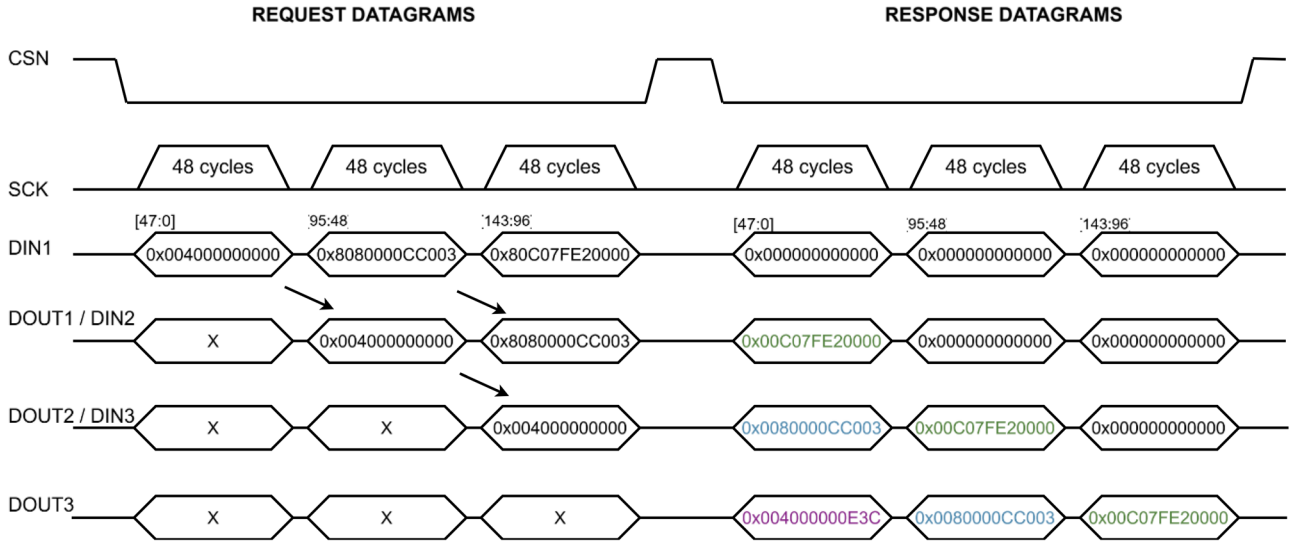


図 10. 3 個の SPI ペリフェラルを一列に接続した SPI デイジーチェーン・データグラムの例

UART インターフェース

TMC6460 の UART インターフェースは、1 つのスタート・ビット、8 つのデータ・ビット、1 つのストップ・ビットで構成されたパリティ・ビットのない信号を最大 12Mbit/s の転送レートでサポートする、3 ピンのインターフェース（GND、RXD、TXD）です。

TMC6460 の UART は標準的なボー・レートをサポートしています。表 3 にこれらの例を示します。更に、自動ボー・レート検出用のオートボー・モードも備わっています。このモードでは、ボー・レートは受信データグラムから検出され、適切なボー・レート値が自動的に設定されます。オートボー機能はデフォルトで有効化されています。

更に、UART インターフェースにはオプションの 8 ビット巡回冗長検査（CRC）機能があり、データグラムでのビット反転を検出できます。データグラムに 1 つ以上のビット反転がある場合、そのデータグラムはインターフェースによって無視されます。CRC 機能はデフォルトで無効化されています。

また、UART インターフェースは、リアルタイム・モニタリング・インターフェース（RTMI）を内蔵しており、周期的データを高速でストリーミングする目的に使用できます。

UART のボー・レート設定

オートボーはデフォルトで有効化されており、TMC6460 は、UART_RXD での各 UART 転送の開始時に同期ビットからボー・レートを計算します。オートボーを無効化するには、UART.CONTROL レジスタの AUTOBAUD_EN ビットをクリアします。必要な値にボー・レートを設定するには、以下の式を用いて同じレジスタの MANTISSA_LIMIT ビットフィールドに書き込みます。

$$MANTISSA_LIMIT = \frac{f_{CLK}}{baud_rate}$$

ここで、PLL がイネーブルの場合は、クロック周波数 $f_{CLK} = f_{SYS}$ = 60MHz、PLL がディスエーブルの場合は、$f_{CLK} = f_{OSC}$ = 15MHz です。MANTISSA_LIMIT は整数値であり、端数が生じるボー発生はサポートされていないため、特定のボー・レートのみが実現できます。例えば、7Mbit/s、8Mbit/s、9Mbit/s、11Mbit/s などのボー・レートは生成できません。表 3 に、PLL がイネーブルの場合の、選択された仮数制限値に対してサポートされるボー・レートを示します。

表 3. ボー・レートの設定

Baud Rate [bit/s]	9600	19200	38400	57600	115200	230400	460800	921600	
MANTISSA_LIMIT	6250	3125	1563	1042	521	260	130	65	
Baud Rate [Mbit/s]	1	2	3	4	5	6	7.5	10	12
MANTISSA_LIMIT	60	30	20	15	12	10	8	6	5

12Mbit/s を超えるボー・レート、または 9600bit/s 未満のボー・レートは、クロック設定によらずサポートされていません。

ボー・レートの動的な変更はサポートされていません。UART_TXD 伝送がアクティブなときにボー・レートを変更しようとしても拒否されます。このルールはすべての場合に適用されます。ボー・レートを変更できるのは、UART_TXD がアクティブでない場合のみです。

MANTISSA_LIMIT の通常の書込み要求コマンドを通じてボー・レートを変更できます。書込み要求に対する応答が送信されるのは、データグラムが UART インターフェースによって完全に処理された後のみであるためです。この場合、書込み要求は、新たに設定されたボー・レートで送り返されます。

転送レートの制限

UART の伝送レートの制限は、クロック設定、UART データグラムの種類、オートボー設定によって決まります。表 4 に様々な制限を示します。

表 4. UART の速度制限

UART CONFIGURATION		**CLOCK SETUP**		
			PLL ENABLED	
DATAGRAM TYPE	**AUTOBAUD**	**PLL DISABLED**	**PLL SOURCE: INTERNAL OSCILLATOR**	**PLL SOURCE: EXTERNAL CLOCK**
Regular Read/Write Datagrams	Enabled	Up to 1Mbit/s	Up to 6Mbit/s	Up to 7.5Mbit/s
	Disabled	Up to 3Mbit/s	Up to 12Mbit/s	Up to 12Mbit/s
RTMI Stream Write Datagram	Enabled	-	Up to 5Mbit/s	Up to 6Mbit/s
	Disabled	-	Up to 12Mbit/s	Up to 12Mbit/s

ボー・レートが非常に高い場合、特に、RTMI ストリームの場合は、オートボーをディスエーブルすることを推奨します。

PLL が無効化されている場合、システム・クロック周波数は 15MHz に減少します。このクロック周波数では、UART インターフェースは 3Mbit/s 以下の伝送レートにのみ対応できます。更に、端数のボー・レート生成はサポートされていないため、PLL がディスエーブルの場合、2Mbit/s というような標準的な一部の伝送レートはサポートされません。

UART 読出し要求データグラム

UART 読出し要求データグラムは、16 ビット長（CRC ディスエーブル）または 24 ビット長（CRC イネーブル）です。図 11 に、CRC がディスエーブルの場合の UART 読出し要求データグラムの構造を示します。これは、オートボーの設定によらず適用されます。

- LSB（ビット 0）が最初に送信され、MSB（ビット 15 またはビット 23）が最後に送信されます。
- ビット 0～2、6、7 は、オートボーの計算用に用いられる同期ビットです。この同期ビットは、ビット 0～2 に対しては 010_b、ビット 6 および 7 に対しては 10_b にセットする必要があります。
- ビット 3 は Write_notRead ビットであり、読出し要求の場合 0_b にセットする必要があります。
- ビット 8～15 は、10 ビットのアドレスの最初の部分（アドレス[7:0]）を表し、ビット 4 および 5 は次の部分（アドレス[9:8]）を表します。
- （CRC を使用する場合のみ）バイト 2 は 8 ビット CRC です。

UART READ REQUEST DATAGRAM (WITHOUT CRC)				
byte 0				byte 1
sync bits	WnR	address	sync bits	8-bit address
010	0	address [8...9]	10	address [0....7]
0...2	3	4 5	6, 7	8 15

図 11. UART 読出し要求データグラムの構造（NORMAL_CRC_EN = 0_b）

読出し要求データグラムの一例は｛0x52 0x40｝です。これは、アドレス 0x140 にある FOC.PID_CONFIG レジスタを読み出すために用いれらます。

CRC がイネーブルの場合（UART.CONTROL レジスタの NORMAL_CRC_EN ビット）、図 12 に示すように、UART インターフェースは、読出し要求が 3 バイトであると想定します。

UART READ REQUEST DATAGRAM (WITH CRC)					
byte 0				byte 1	byte 2
sync bits	WnR	address	sync bits	8-bit address	8-bit CRC
010	0	address [8...9]	10	address [0....7]	checksum
0 ...2	3	4 5	6, 7	8 15	16 23

図 12. UART 読出し要求データグラムの構造（NORMAL_CRC_EN = 1_b）

CRC がイネーブルの場合の読出し要求データグラムの一例は｛0x52 0x40 0x12｝で、0x12 は計算された CRC 値です。また、このコマンドは、アドレス 0x140 の FOC.PID_CONFIG レジスタを読み出すためにも用いられます。

UART 読出し応答データグラム

UART 読出し応答データグラムは、48 ビット長（CRC ディスエーブル）または 56 ビット長（CRC イネーブル）です。図 13 に CRC がディスエーブルの場合の UART 読出し応答データグラムの構造を示します。

- LSB（ビット 0）が最初に送信され、MSB（ビット 47 またはビット 55）が最後に送信されます。
- 最初の 2 バイト（バイト 0 およびバイト 1）は、読出し要求の場合と同じで、同期ビット、Write_notRead ビット、アドレス・ビットで構成されています。
- ビット 16～47（バイト 2～5）には、要求されたアドレスからの 32 ビットの読出しデータが含まれます。
- （CRC を使用する場合のみ）バイト 6 は 8 ビット CRC です。

UART READ RESPONSE DATAGRAM (WITHOUT CRC)							
byte 0				byte 1	byte 2	byte 3	byte 4	byte 5
sync bits	WnR	address	sync bits	8-bit address	reply data (32-bit)			
010	0	address [8...9]	10	address [0....7]	data byte 3	data byte 2	data byte 1	data byte 0
0 ...2	3	4 5	6, 7	8 15	16 23	24 31	32 39	40 47

図 13. UART 読出し応答データグラムの構造（NORMAL_CRC_EN = 0_b）

CRC を使用しない読出し応答データグラムの一例は｛0x52 0x40 0x00 0x00 0x85 0x5C｝で、これはアドレス 0x140 の FOC.PID_CONFIG レジスタからの 0x0000855C の読出しデータに対応しています。

CRC がイネーブルの場合（UART.CONTROL レジスタの NORMAL_CRC_EN ビット）、読出し応答には内部で計算された CRC バイトが付加されます。図 14 に、それを示します。

UART READ RESPONSE DATAGRAM (WITH CRC)									
byte 0				byte 1	byte 2	byte 3	byte 4	byte 5	byte 6
sync bits	WnR	address	sync bits	8-bit address	reply data (32-bit)				8-bit CRC
010	0	address [8...9]	10	address [0....7]	data byte 3	data byte 2	data byte 1	data byte 0	checksum
0 ...2	3	4 5	6, 7	8 15	16 23	24 31	32 39	40 47	48 55

図 14. UART 読出し応答データグラムの構造（NORMAL_CRC_EN = 1_b）

CRC を使用する読出し応答データグラムの一例は｛0x52 0x40 0x00 0x00 0x85 0x5C 0x6A｝で、バイト 6（0x6A）は UART インターフェースによって計算された CRC です。

図 15 に、読出し要求および読出し応答のデータグラムの詳細な例を示します。CRC がイネーブルで、最初の 2 バイトのビットのいずれかが破損したか UART インターフェースによって不適切に受信された場合、要求は無視され、応答は送信されません。

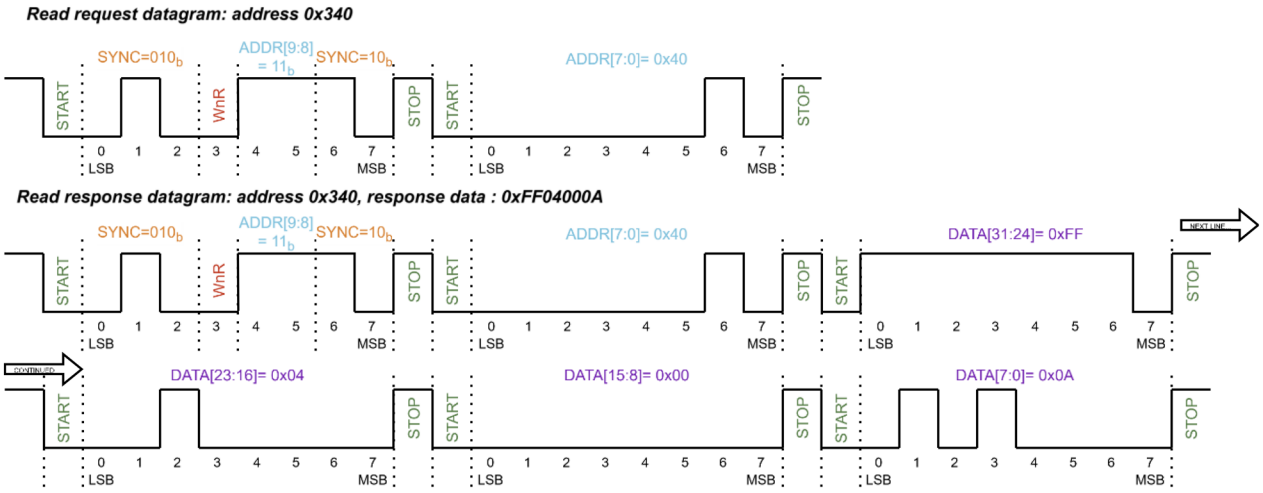


図 15. UART 読出しアクセスの例

UART 書込み要求データグラム

UART 書込み要求データグラムは、48 ビット長（CRC ディスエーブル）または 56 ビット長（CRC イネーブル）です。図 16 に、CRC がディスエーブルの場合の UART 書込み要求データグラムの構造を示します。これは、オートボーの設定によらず適用されます。

- LSB（ビット 0）が最初に送信され、MSB（ビット 47 またはビット 55）が最後に送信されます。
- ビット 0～2、6、7 は、オートボーの計算用に用いられる同期ビットです。この同期ビットは、ビット 0～2 に対しては 010b、ビット 6 および 7 に対しては 10_b にセットする必要があります。
- ビット 3 は Write_notRead ビットであり、書込み要求の場合 1_b にセットする必要があります。
- ビット 8～15 は、10 ビットのアドレスの最初の部分（アドレス[7:0]）を表し、ビット 4 および 5 は次の部分（アドレス[9:8]）を表します。
- ビット 16～47（バイト 2～5）は、指定されたアドレスのレジスタに書き込まれる 32 ビットのデータです。
- （CRC を使用する場合のみ）バイト 6 は 8 ビット CRC です。

UART WRITE REQUEST DATAGRAM (WITHOUT CRC)								
byte 0				byte 1	byte 2	byte 3	byte 4	byte 5
sync bits	WnR	address	sync bits	8-bit address	write data (32-bit)			
010	1	address [8...9]	10	address [0....7]	data byte 3	data byte 2	data byte 1	data byte 0
0 ...2	3	4 5	6, 7	8 15	16 23	24 31	32 39	40 47

図 16. UART 書込み要求データグラムの構造（NORMAL_CRC_EN = 0_b）

書込み要求データグラムの一例は｛0x5A 0x4E 0x01 0x00 0x01 0x00｝で、アドレス 0x14E のレジスタ FOC.PID_TORQUE_FLUX_TARGET に値 0x01000100 を書き込みます。

CRC がイネーブルの場合（UART.CONTROL レジスタの NORMAL_CRC_EN ビット）、図 17 に示すように、UART インターフェースは、書込み要求に CRC バイトが付加されるものと想定します。

UART WRITE REQUEST DATAGRAM (WITH CRC)									
byte 0				byte 1	byte 2	byte 3	byte 4	byte 5	byte 6
sync bits	WnR	address	sync bits	8-bit address	write data (32-bit)				8-bit CRC
010	1	address [8...9]	10	address [0....7]	data byte 3	data byte 2	data byte 1	data byte 0	checksum
0 ...2	3	4 5	6, 7	8 15	16 23	24 31	32 39	40 47	48 55

図 17. UART 書込み要求データグラムの構造（NORMAL_CRC_EN = 1_b）

CRC がイネーブルの場合の書込み要求データグラムの一例は、｛0x5A 0x4E 0x01 0x00 0x01 0x00 0x5A｝で、アドレス 0x14E の FOC.PID_TORQUE_FLUX_TARGET レジスタに書き込まれます。この場合、最後のバイト 0x5A は計算された CRC 値です。

UART 書込み応答データグラム

UART 書込み応答データグラムは、40 ビット長（CRC ディスエーブル）または 48 ビット長（CRC イネーブル）です。書込み要求に対する応答が送信されるのは、UART.RTMI_CH_7 レジスタの CH_7_EN ビットフィールドがセットされた場合のみです。セットされていない場合、書込み要求は処理されますが、応答は送信されません。詳細については、UART RTMI のセクションを参照してください。図 18 に CRC がない場合の書込み応答データグラムの構造を示します。

- LSB（ビット 0）が最初に送信され、MSB（ビット 39 またはビット 47）が最後に送信されます。
- ビット 0 は 1_b、ビット 7 は 0_b の同期ビットです。
- ビット 1～3 は、チャンネル ID を表します。これは通常の書込み応答データグラムでは常に 111_b です。
- ビット 4～6 には、実行された書込み要求の数を示す書込みカウントが含まれます。書込みカウントは、ストリームと通常の両方の UART 書込み要求と SPI 書込み要求で共有されます。これらのいずれかのインターフェースから実行される書込み要求によって、同じ書込みカウントが増加します。
- ビット 8～39（バイト 1～4）は 32 ビットの応答データです。
- （CRC を使用する場合のみ）バイト 5 は 8 ビット CRC です。

UART WRITE RESPONSE DATAGRAM (WITHOUT CRC)							
byte 0				byte 1	byte 2	byte 3	byte 4
sync bit	reg_id	wrt_cnt	sync bit	response data (32-bit)			
1	111	xxx	0	data byte 3	data byte 2	data byte 1	data byte 0
0	1...3	4...6	7	8 15	16 23	24 31	32 39

図 18. UART 書込み応答データグラムの構造（RTMI_CRC_EN = 0_b）

書込み応答データグラムの一例は｛0x1F 0x01 0x00 0x01 0x00｝で、アドレス 0x14E の FOC.PID_TORQUE_FLUX_TARGET レジスタに値 0x01000100 を書き込む応答です。最初のバイト 0x1F は、それまでに 1 回の書込みアクセスが行われたことを示します。

CRC がイネーブルの場合（UART.CONTROL レジスタの RTMI_CRC_EN ビット）、書込み応答には内部で計算された CRC バイトが付加されます。図 19 に、それを示します。NORMAL_CRC_EN ビットは、このデータグラムでは考慮**されない**点に注意してください。

UART WRITE RESPONSE DATAGRAM (WITH CRC)								
byte 0				byte 1	byte 2	byte 3	byte 4	byte 5
sync bit	reg_id	wrt_cnt	sync bit	response data (32-bit)				8-bit CRC
1	111	xxx	0	data byte 3	data byte 2	data byte 1	data byte 0	checksum
0	1...3	4...6	7	8 15	16 23	24 31	32 39	40 47

図 19. UART 書込み応答データグラムの構造（RTMI_CRC_EN = 1_b）

図 20 に、UART 書込み要求とそれに対応する応答の構造を示します。上部の波形は、アドレス 0x240 およびデータ 0x80808091 を指定して書込みコマンドを送信するホスト・ペリフェラルを表し、下部の波形は同じアドレスおよびデータの書込みを確認する応答を示します。

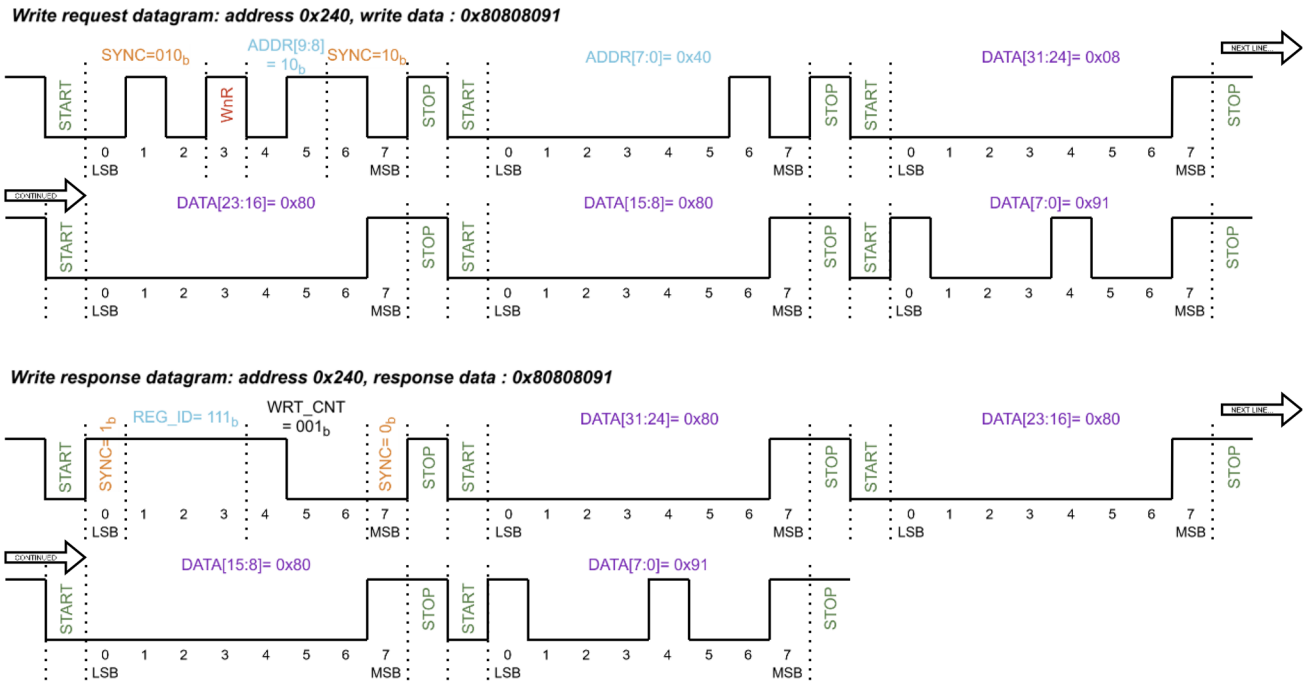


図 20. UART 書込みアクセスの例

UART リアルタイム・モニタリング・インターフェース（RTMI）

UART リアルタイム・モニタリング・インターフェース（RTMI）は、TMC6460 との間でリアルタイムのデータ・ストリーミングを可能にする、付加的なインターフェースです。それによって、モーターやシステムのパラメータを特定するというような、アプリケーションレベルのソフトウェア機能が促進されます。モニタリングのために最大 8 チャンネルを IC 内の任意のレジスタに割り当てることができます。

RTMI の設定

UART RTMI 機能を用いるには、UART.CONTROL レジスタの RTMI_EN ビットをセットします。RTMI のトリガ周波数および使用する各 RTMI チャンネルは、事前に設定しておく必要があります。

使用可能な各 RTMI チャンネルは UART.RTMI_CH_x レジスタ（x = 0～7）を通じて個別に設定とイネーブルができます。

各チャンネルには以下の 2 つの設定オプションがあります。

- レジスタ・フィールド CH_x_ADDR（x = 0～7）は、この RTMI チャンネルに割り当てられるレジスタのアドレスを定義します。
- レジスタ・フィールド CH_x_EN（x = 0～7）は、RTMI チャンネルを個別にイネーブルします。

イネーブルされたすべてのチャンネルのデータは、設定された周波数のトリガに基づき、周期的に UART_TXD ピンに送り出されます。トリガは、PWM サイクルの開始と共に生成され、UART.CONTROL レジスタの RTMI_SAMPLING フィールドのダウン・サンプリング・レートの値に従います。トリガ周波数を計算するには以下の式を用います。

$$f_{TRIGGER} = \frac{f_{PWM}}{RTMI_SAMPING + 1}$$

例えば、現在の PWM 周波数が 50kHz で、RTMI_SAMPLING = 1 の場合、新しい RTMI データ・ストリームは、25kHz の周波数でトリガされます。

RTMI トリガ周波数を選択する場合、イネーブルされたチャンネル数と設定されたボー・レートに従って、データ・ストリームの長さを考慮する必要があります。新しい RTMI トリガがデータ・ストリームの進行中に発行された場合、新しいストリームはそれ以前のストリームを中断し、UART.EVENTS レジスタの RTMI_INTERRUPT_EVENT フィールドがセットされます。

次式の関係が満たされていれば、イネーブルされたチャンネル応答は RTMI トリガの間で完全に受信できます。

$$T_{TRIGGER} > T_{RTMI}$$

- $T_{TRIGGER} = \frac{1}{f_{TRIGGER}}$ であり、これは、2 つの連続する RTMI トリガの間の時間です。

- T_{RTMI}は、送信する RTMI ストリーム・データ（イネーブルされたチャンネル）に要する時間であり、次式で表せます。

$$T_{RTMI} = n_{bytes} \times n_{channels} \times 10 \times T_{BIT}$$

  - n_{bytes} = 5 RTMI_CRC_EN = 0 の場合）またはn_{bytes} = 6（RTMI_CRC_EN = 1 の場合）
  - $n_{channels}$は、イネーブルされたチャンネルの数です。
  - $T_{BIT} = \frac{1}{f_{UART}}$ は、各 UART ビットを送信するのに要する時間です。

例えば、PWM 周波数が 50kHz、RTMI_SAMPLING が 4、イネーブルされたチャンネルの数が 5、UART のボー・レートが 1MBaud（CRC ディスエーブル）の RTMI データ・ストリームを仮定します。

- $f_{TRIGGER} = \frac{f_{PWM}}{RTMI_SAMPING+1} = \frac{50kHz}{4+1} = 10kHz$
- $T_{TRIGGER} = \frac{1}{f_{TRIGGER}} = \frac{1}{10KHz} = 100\mu s$
- $T_{RTMI} = n_{bytes} \times n_{channels} \times 10 \times T_{BIT} = 5 \times 5 \times 10 \times \frac{1}{1\,MBaud} = 250 \times 1\mu s = 250\mu s$

この場合、$T_{TRIGGER}$はT_{RTMI}より小さいため、RTMI トリガの間ですべてのチャンネル応答が送信されるわけではありません。この状況では、すべての応答を受信するには、RTMI_SAMPLING、PWM 周波数、または UART のボー・レートを変更する必要があります。UART のボー・レートを 4MBaud に増加すると、T_{RTMI}は 62.5µs になります。それによって、$T_{TRIGGER}$がT_{RTMI}より大きくなり、チャンネル応答は中断なく受信されます。

RTMI チャンネル 0 を設定してイネーブルするための UART 書込み要求コマンドの例を以下に示します。

- ADC.I2_I1_RAW レジスタに以下を書込み、RTMI チャンネル 0 をイネーブルして割り当てます：｛0x7A 0x44 0x00 0x01 0x00 0x82｝。
  - 最初の 2 バイト（0x7A 0x44）には、同期および目標アドレス（RTMI_CH_0 レジスタ）データが含まれます。
  - 最後の 4 バイト（0x00 0x01 0x00 0x82）には、RTMI_CH_0 レジスタに書き込まれるデータが含まれます。
    - バイト（0x01）は、CH_0_EN ビットフィールドの設定に対応します。
    - バイト（0x0082）は、割り当てられた ADC.I2_I1_RAW レジスタのアドレスに対応します。

同様のコマンドを、使用するその他すべての RTMI チャンネルに発行できます。

UART RTMI のチャンネル応答データグラム

UART RTMI チャンネル応答データグラムは、40 ビット長（CRC ディスエーブル）または 48 ビット長（CRC イネーブル）です。上述のように、RTMI がイネーブルの場合、UART インターフェースは、設定されたトリガ周波数で周期的にチャンネル応答を発行します。図 21 に CRC がない場合の RTMI チャンネル応答データグラムの構造を示します。

- LSB（ビット 0）が最初に送信され、MSB（ビット 39 またはビット 47）が最後に送信されます。
- ビット 0 は 1_b、ビット 7 は 0_bの同期ビットです。
- ビット 1～3 はチャンネル ID ビットです。これらは、000_b（チャンネル 0）と 111_b（チャンネル 7）の間の値に設定され、データ送信先の RTMI チャンネルを示します。
- ビット 4～6 には、実行された書込み要求の数を示す書込みカウントが含まれます。書込みカウントは、ストリームと通常の両方の UART 書込み要求と SPI 書込み要求で共有されます。これらのいずれかのインターフェースから実行される書込み要求によって、同じ書込みカウントが増加します。
- ビット 8～39（バイト 1～4）は、チャンネル ID に対応する RTMI チャンネルの 32 ビットの応答データです。
- （CRC を使用する場合のみ）バイト 5 は 8 ビット CRC です。

CHANNEL RESPONSE DATAGRAM (WITHOUT CRC)							
byte 0				byte 1	byte 2	byte 3	byte 4
sync bit	reg_id	wrt_count	sync bit	response data (32-bit)			
1	xxx	xxx	0	data byte 3	data byte 2	data byte 1	data byte 0
0	1...3	4...6	7	8 15	16 23	24 31	32 39

図 21. RTMI ストリーム・チャンネル応答データグラムの構造（RTMI_CRC_EN = 0_b）

図 22 は、RTMI チャンネルの応答の一例を示すもので、RTMI チャンネル 2、4、6、7 はイネーブルされていません。そのため、これらのチャンネルのデータは、ストリーム応答には含まれません。イネーブルされたすべてのチャンネルのチャンネル応答は、伝送が行われている間に RTMI_EN ビットがクリアされたとしても、RTMI トリガが受信されたときに送信されます。伝送が終了した後、新しい RTMI トリガが生成されるまで、TXD ピンはアイドル状態のままです。

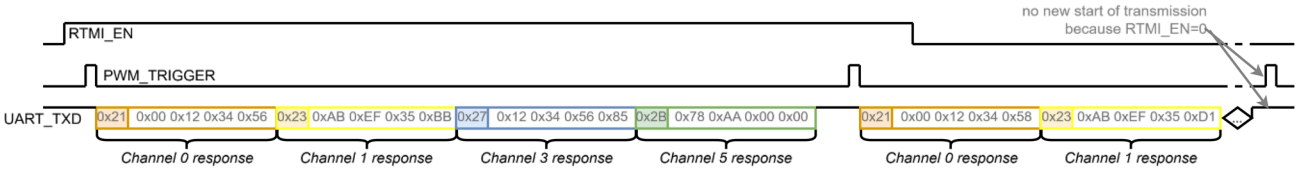


図 22. UART ストリーム・チャンネル応答データグラムの例

これらのタイプの応答データグラムには、UART.CONTROL レジスタの RTMI_CRC_EN ビットフィールドをイネーブルすることにより、8 ビットの CRC も含まれます。図 23 にこのデータグラム構造を示します。

CHANNEL RESPONSE DATAGRAM (WITH CRC)							
byte 0				byte 1	byte 2	byte 3	byte 4	byte 5
sync bit	reg_id	wrt_count	sync bit	response data (32-bit)				8-bit CRC
1	xxx	xxx	0	data byte 3	data byte 2	data byte 1	data byte 0	checksum
0	1...3	4...6	7	8 15	16 23	24 31	32 39	40 47

図 23. RTMI ストリーム・チャンネル応答データグラムの構造（RTMI_CRC_EN = 1_b）

UART RTMI ストリーム書込み要求データグラム

ストリーム書込み要求は、通常の書込み要求の代替策として行われます。通常の書込み要求データグラムより 1 バイト短いという利点があり、より高速のデータ転送レートが可能になります。ただし、この方法で書き込めるレジスタは 8 種類のみです。この動作は、UART RTMI チャンネルに依拠しています。そのため、それぞれの RTMI チャンネルと RTMI 機能が、ストリーム書込み要求が機能するように有効化されている必要があります。

ストリーム書込み要求コマンドを通じて書き込めるレジスタは、レジスタ UART.RTMI_CH_x（x＝0～7）の設定によって決まります。例えば、RTMI チャンネル 0 がアドレス 0x155 のレジスタ FOC.PID_POSITION_ACTUAL に割り当てられた場合、このレジスタはストリーム書込みコマンドで使用できるようになります。したがって、このレジスタの値は、RTMI チャンネル 0 にストリーム書込み要求を発行することによって更新できます。

ストリーム書込み要求データグラムは、40 ビット長（CRC ディスエーブル）または 48 ビット長（CRC イネーブル）です。図 24 に書込み要求データグラムの構造を示します。なお、このデータグラムのタイプは、RTMI のトリガとしては使用されません。

- LSB（ビット 0）が最初に送信され、MSB（ビット 39 またはビット 47）が最後に送信されます。
- ビット 0 およびビット 7 は同期ビットです。ビット 0 は 1_b、ビット 7 は 0_b にセットする必要があります。これらは、オートボーの計算に用いられます。
- ビット 1～3 はチャンネル ID ビットです。これらのビットは、書込み動作に用いるよう選択された RTMI チャンネルを表します。000_b（チャンネル 0）～111_b（チャンネル 7）の値を設定する必要があります。
- ビット 4～6 は予約ビットであり、001_b にセットする必要があります。これらのビットは、オートボーの計算にも用いられます。
- ビット 8～39（バイト 1～4）は、指定された RTMI チャンネルに割り当てられたレジスタに書き込まれる 32 ビットのデータです。
- （CRC を使用する場合のみ）バイト 5 は 8 ビット CRC です。

STREAM WRITE REQUEST DATAGRAM (WITHOUT CRC)							
byte 0				byte 1	byte 2	byte 3	byte 4
sync bit	reg_id	resv_bits	sync bit	write data (32-bit)			
1	3-bit	001	0	data byte 3	data byte 2	data byte 1	data byte 0
0	1...3	4...6	7	8 15	16 23	24 31	32 39

図 24. RTMI ストリーム書込み要求データグラムの構造（RTMI_CRC_EN = 0_b）

CRC がイネーブルの場合（UART.CONTROL レジスタの RTMI_CRC_EN ビット）、図 25 に示すように、UART インターフェースは、書込み要求に CRC バイトが付加されるものと想定します。

STREAM WRITE REQUEST DATAGRAM (WITH CRC)							
byte 0				byte 1	byte 2	byte 3	byte 4	byte 5
sync bit	reg_id	resv_bits	sync bit	write data (32-bit)				8-bit CRC
1	3-bit	001	0	data byte 3	data byte 2	data byte 1	data byte 0	checksum
0	1...3	4...6	7	8 15	16 23	24 31	32 39	40 47

図 25. RTMI ストリーム書込み要求データグラムの構造（RTMI_CRC_EN = 1_b）

UART タイムアウト

TMC6460 の UART インターフェースは、データグラムの連続するバイト間での UART_RXD 信号のアイドル時間を測定する、タイムアウト機構も備えています。アイドル時間が設定された制限値より長い場合は、UART.EVENTS レジスタの RX_IDLE_TIMEOUT_EVENT ビットがセットされます。タイムアウト機構はデフォルトでイネーブルされています。

タイムアウトは、システム・クロックの f_{SYS} に基づき、カウンタによって定義されます。カウンタの制限値は、UART.TIMEOUT レジスタの LIMIT フィールドを通じて定めることができます。タイムアウトは次式で計算できます。

$$timeout = \frac{LIMIT}{f_{SYS}} \; with \; f_{SYS} = 60 \; MHz$$

このように、リセット値を LIMIT = 65535 にすると、デフォルトのタイムアウト時間はおよそ 1.1ms になります。

システム・クロックの前置分周器として 4096 を追加するのは、UART.CONTROL レジスタの TIMEOUT_PRE_DIVIDER_EN ビットをセットすることにより可能です。前置分周器がイネーブルされている場合、タイムアウトは以下のように計算できます。

$$timeout = \frac{LIMIT \times 4096}{f_{SYS}} \; with \; f_{SYS} = 60 \; MHz$$

TIMEOUT_PRE_DIVIDER_EN ビットがセットされ、LIMIT = 1_b の場合、最低タイムアウトは 68.26µs になります。LIMIT を 0 にセットすると、タイムアウト機能は無効化されます。

UART 故障モニタリング

このセクションでは、UART.EVENTS レジスタに現れる可能性のある故障イベントについて説明します。

- UART_SYNC_FAIL_EVENT
  - オートボー・モードが有効化されている場合：
    - このビットは、同期ビットが正しく受信されたと仮定すると、データグラムの破損が検出された場合にセットされます。
    - 受信したデータグラムの同期ビットが正しくない場合にもセットされます。上述のデータグラム構造を参照してください。
  - オートボー・モードが無効化されている場合：
    - 受信したデータグラムの同期ビットが正しくない場合にもセットされます。上述のデータグラム構造を参照してください。
- RX_IDLE_TIMEOUT_EVENT
  - このビットがセットされるのは、データグラムの連続するバイトの間で、指定されたタイムアウト時間（デフォルトでは約 1.1ms）を超えて UART_RXD ピンがアイドルになった場合です。
- INVALID_START_BIT_EVENT
  - いずれかのバイトのスタート・ビットが破損した場合にこのビットがセットされます。
- INVALID_STOP_BIT_EVENT
  - いずれかのバイトのストップ・ビットが破損した場合にこのビットがセットされます。
- INVALID_CRC_BIT_EVENT
  - CRC がイネーブルの場合、外部で計算された CRC 値を含む追加バイトが必要になります。
  - このビットがセットされるのは、提供された CRC 値と内部で計算された CRC 値が同一でない場合です。

これらのイベントはすべて、対応するイベント・ビットに 1_b を書き込むとクリアできます。これらのイベントがトリガされる原因になるデータグラムは破棄されるため、処理されません。

次のセクションで説明する NOISY_DATA_EVENT を含む、これらの全イベントに対する信号は、CHIP.STATUS_FLAGS レジスタの UART_FAIL_STATUS ビットと、それに対応する CHIP.EVENTS レジスタの UART_FAIL_EVENT ビットに組み込まれます。UART_FAIL_STATUS ビットは、各イベント・ビットのいずれかがセットされている限り、アクティブのままです。同様に、UART_FAIL_EVENT ビットをクリアする前に、各イベント・ビットをすべてクリアする必要があります。

UART が応答しない場合、UART_RXD ピンを 70ms 以上ハイにセットし、UART モジュールをリセットする必要があります。

UART 受信データのフィルタ処理

TMC6460 の UART インターフェースには、ノイズやタイミング変動の影響を軽減するためのフィルタ処理機能があります。これを有効化するには、UART.CONTROL レジスタの RX_FILTER_EN ビットフィールドをセットします。この機能が有効化されていると、UART_RXD で受信するデータは、ビット周期あたり 3 回、サンプリングされます。最終的なビット値は、多数決方式で決まります。つまり、3 つの読出し値のうち 2 つで最終ビット値が決まります。

サンプリングされた値が 1 ビット周期以内に変化する場合（例えば、001_b、101_b 等）、UART.EVENTS レジスタの NOISY_DATA_EVENT ビットがセットされます。このイベントは警告の役割のみを果たします。このイベントの原因となるデータグラムはそのまま処理されます。

この機能はデフォルトで無効化されています。無効化されている場合、RXD 信号は各ビット周期の中央で 1 回のみサンプリングされます。

UART CRC の設定および計算

UART データグラムの要求および応答における意図せぬ変化を検出するために、CRC-8 計算が用いられます。デフォルトでは、UART インターフェースの CRC はディスエーブルされています。UART.CONTROL レジスタには、UART データグラムの CRC の評価と生成を行うために、2 つのビットフィールドがあります。

- NORMAL_CRC_EN ビットをセットすると、以下が可能になります。
  - TMC6460 に送信される読出し要求データグラムの CRC 評価をイネーブル。
  - TMC6460 から送信される読出し応答データグラムの CRC 生成。
  - TMC6460 に送信される通常の書込み要求データグラムの CRC 評価をイネーブル。
- RTMI_CRC_EN ビットをセットすると、以下が可能になります。
  - TMC6460 に送信される RTMI ストリーム書込み要求データグラムの CRC 評価をイネーブル。
  - TMC6460 から送信される書込み応答データグラムの CRC 生成。
  - TMC6460 から送信される RTMI チャンネル応答データグラムの CRC 生成。

UART データグラムの最後のバイトには、それまでの全バイトから計算された CRC 値が格納されます。受信 CRC が内部で計算された CRC と一致しない場合、要求は無視され、応答は送信されません。

CRC-8 の計算のチェックサムでは、多項式$P(x) = x^8 + x^4 + x^3 + x^2 + 1$ を使用します。これは、2 進数パターン 100011101_b に対応します。8 ビットの CRC 多項式には 9 つの項があるため、9 ビットで表されます。2 進数表示を形成するために、多項式の各項がビット位置に割り当てられます。つまり、その項が存在するとそのビットが 1 にセットされます。この多項式の場合、x^8、x^4、x^3、x^2、x^0 が存在します。そのため、8、4、3、2、0 の各ビットが 1 にセットされます。

UART データグラムの各バイトは最下位ビット・ファーストで送信されるため、CRC 計算では、最上位ビットが先頭に配置されるよう、ビットを逆順にしたバイトを用いなくてはなりません。

1. 各バイトを逆順にする必要があります。例えば、0x4A（01001010_b）は 01010010_b になります。すべてのデータグラムのバイトを逆順にした後、それらを連結し、CRC 計算用に 8 個のゼロ・ビットを追加します。

```
Sync/Address[9:8]   Address[7:0]   Data
01010010            00000010       00000000 00000000 00000000 10000000 00000000
```

2. 多項式を入力データの間でビットごとに XOR（^）操作を行います。まず、多項式の最も左にある 1 のビットを入力の最も左にある 1 のビットに揃えます。多項式と重なる全ビットに XOR を適用し、右側のビットは変更せずにおきます。元のすべての入力ビット（ゼロを付加する前の入力ビット）がゼロに変わるまで、必要に応じて多項式を右にシフトしながらこのプロセスを繰り返します。このステップを以下のシーケンスで示します。

XOR Steps 1…10

```
 0101001010000000000000000000000000000000010000000000000000
^ 100011101
=000101011100000000000000000000000000000010000000000000000
^   100011101
=00000100000100000000000000000000000000010000000000000000
^     100011101
=00000000011001000000000000000000000000010000000000000000
^         100011101
=00000000001000110100000000000000000000010000000000000000
^          100011101
=00000000000000011100000000000000000000010000000000000000
^                100011101
=00000000000000001101110100000000000000010000000000000000
^                 100011101
=00000000000000000101001110000000000000010000000000000000
^                  100011101
=00000000000000000010100110000000000000010000000000000000
^                   100011101
=00000000000000000001010001000000000000010000000000000000
^                    100011101
=00000000000000000000101100100000000000010000000000000000
```

(*Resume on top of right side.*)

XOR Steps 11…20

```
 00000000000000000000001011001000000000010000000000000000
^                      100011101
=00000000000000000000001111001000000000010000000000000000
^                       100011101
=00000000000000000000001111110010000000010000000000000000
^                        100011101
=00000000000000000000001111011110000100000000000000000000
^                         100011101
=00000000000000000000001110000110001000000000000000000000
^                          100011101
=00000000000000000000001001101100100000000000000000000000
^                           100011101
=00000000000000000000001010110100000000000000000000000000
^                            100011101
=00000000000000000000001000111000000000000000000000000000
^                             100011101
=00000000000000000000001000011100000000000000000000000000
^                              100011101
=00000000000000000000000111010000000000000000000000000000
^                               100011101
=00000000000000000000000011001101000000000000000000000000
```

3. 最上位ビットを最初にして、残りの 8 ビットを取ります。これが CRC 値を表します。この例では、CRC 値の結果は 11001101（0xCD）です。

ピン設定

TMC6460 では、システムやレイアウトの様々な制約に適応できる、柔軟なピン設定方式が可能です。様々なピン設定の概要については、端子説明の表を参照してください。このピン設定方式では、いくつかのピンについて、入力／出力のプルアップ／プルダウン抵抗やアナログ入力用の分圧器の選択ができる他、入力レベルの反転や入力ノイズのフィルタ処理などの前処理オプションもあります。

デジタル入力の反転

前処理ブロックでは、一部のデジタル入力信号を反転できます。入力信号の反転は、例えば、リファレンス・スイッチから必要なロジックを取得する場合や、デジタル・フィードバック・システムから反転ロジックを取得する場合などに役立つことがあります。CHIP.IO_CONFIG レジスタの対応する INV_PIN_x ビット（x = ENC_A など）を用いると、以下の入力ピンの信号を反転できます。

- ABN エンコーダ・ピン ENC_A、ENC_B、ENC_N
- ホール・センサー・ピン HALL_U、HALL_V、HALL_W
- リファレンス・スイッチ・ピン REF_L および REF_R
- SCK ピンおよび SDI ピン（代替機能を用い、デフォルトの SPI 機能は用いない場合にのみ適用可）
- UART ピン RXD
- 制御入力ピン PWM_IN

入力信号反転を用いる場合、そのピンがデジタル入力として用いられる限り、対応するピン信号に直接適用されます。例えば、INV_PIN_ENC_A がセットされると、ENC_A ピンがそのメイン機能であるインクリメンタル・エンコーダ入力 A で用いられるか、代替機能（デジタル・ホール入力 U など）で用いられるかに関係なく、入力信号は反転します。

この機能を再設定する際には注意が必要です。稼働中に UART RXD ピンのロジックを変更すると、リセットされるまでチップとの通信に不要な中断が発生する可能性があるためです。

デジタル入力のフィルタ処理

前処理ブロックは、デジタル入力信号用のフィルタを用いる機能も備えています。このフィルタは、以下の入力に適用できます。

- ABN エンコーダ・ピン ENC_A、ENC_B、ENC_N
- ホール・センサー・ピン HALL_U、HALL_V、HALL_W
- リファレンス・スイッチ・ピン REF_L および REF_R
- 制御入力ピン PWM_IN

フィルタはブロックごとに設定されます。つまり、ENC_x ピン（x＝A、B、N）全部に対し 1 つの設定、HALL_x ピン（x＝U、V、W）全部に対し 1 つの設定、という具合です。入力フィルタを用いる場合、そのピンがデジタル入力として用いられる限り、対応する IC ピンに直接適用されます。唯一の例外は HALL_W ピンです。このピンがホーム・リファレンス・スイッチに再割り当てされた場合、リファレンス・スイッチ・ブロックからフィルタ設定を取得します。

設定は、CHIP.IO_CONFIG レジスタ内の ENC_FLT、HALL_FLT、REF_FLT、PWM_IN_FLT の各フィールドを用いて行われます。各ブロックは、フィルタ処理を行わないように設定するか、次のフィルタ・カットオフ周波数となるように設定できます。つまり、設定値が 00_b、01_b、10_b、11_b に対し、それぞれ 1MHz、500kHz、100kHz です。

フィルタは、ウィンドウが可変の多数決フィルタのように機能します。フィルタは、電気的ノイズからスプリアス・スイッチング・イベントを除去するために用いられます。デフォルトでは、上述のすべてのインターフェースに対し 1MHz のフィルタがイネーブルされています。フィルタは、そのカットオフ周波数から始まるスプリアス・イベントの一部を除去し、その周波数の約 15%高い周波数から始まるすべてのイベントを除去します。

アナログ入力分圧器

TMC6460 のアナログ入力には、調整可能な分圧器を設定できます。V_{CC_IO} と V_{CC_IOF} のリファレンス電圧の各々のピンに、分圧器の値を個別に設定できます。分圧器を設定するには、CHIP.IO_CONFIG レジスタの対応する ANA_DIV_VCCIO フィールドおよび ANA_DIV_VCCIOF フィールドを使用します。使用可能な分圧オプションは、4.5（設定値 00_b）、3.0（設定値 01_b）、2.2（設定値 10_b）、1（分圧なし、設定値 11_b）です。ANA_DIV_VCCIO と ANA_DIV_VCCIOF のどちらのフィールドに対しても、4.5 がデフォルト・オプションです。

ADC の設定に関するその他の詳細については、ADC の設定と測定のセクションを参照してください。

プルアップ／プルダウン設定

TMC6460 のいくつかのピンには、プルアップ抵抗またはプルダウン抵抗を個別に設定できます。なお、すべてのピンに両方のプル・オプションがあるわけではない点に注意してください。抵抗の特性については、電気的特性の表を参照してください。

プルアップ抵抗は以下のピンで使用できます。

- ABN エンコーダ・ピン ENC_A、ENC_B、ENC_N
- ホール・センサー・ピン HALL_U、HALL_V、HALL_W
- リファレンス・スイッチ・ピン REF_L および REF_R
- SPI ピン SDI、SCK、SDO
- UART ピン UART_RXD
- 制御入力ピン PWM_IN
- 制御出力ピン FAULTN（オープン・ドレインとして設定されている場合のみ有効）

プルダウン抵抗は以下のピンで使用できます。

- ABN エンコーダ・ピン ENC_A、ENC_B、ENC_N
- ホール・センサー・ピン HALL_U、HALL_V、HALL_W
- リファレンス・スイッチ・ピン REF_L および REF_R
- SPI ピン SDI および SCK
- UART ピン UART_RXD
- 制御入力ピン PWM_IN

プルアップ／プルダウン抵抗は入力パッドに直接付加されます。そのため、これまでのセクションで示したすべてのデジタル適応の前に、パッド・レベルが操作されます。この機能を有効化するには、CHIP.IO_PU_PD レジスタの対応する PIN_x_PU ビットまたは PIN_x_PD ビット（例えば x = ENC_A）をセットします。1 本のピンに対し、プルアップとプルダウンの両方がイネーブルされる場合は、内部プルアップ／プルダウン抵抗のどちらも接続されません。

ピンの再割り当て

TMC6460 のピンの多くは、様々な機能を果たすために再割り当てができます。ピン名は、単にデフォルトのピン割り当てを示すにすぎません。例えば、ピン ENC_A（14 番ピン）には、第一の ABN エンコーダ・ブロックの A 入力としてデフォルトの割り当てがあります。しかし、その他のオプションの中から、これをデジタル・ホール・センサー・ブロックの U 入力として再度割り当てることもできます。

すべてのピン再割り当ては、CHIP.IO_MATRIX レジスタの PIN_x フィールド（例えば x = ENC_A）を通じて行われます。特定のピンに代替機能を設定するには、単に目的の機能に対応する値を設定するだけです。例えば、AIN の代替機能を設定するには、PIN_PWM_IN を 01_b に設定します。ただし、すべての代替機能の値が有効とは限らない点に注意してください。使用できないオプションや予約済みオプションに対する値は設定できません。

以下のピンは代替機能に再割り当てできます。

- ABN エンコーダ・ピン ENC_A、ENC_B、ENC_N
- ホール・センサー・ピン HALL_U、HALL_V、HALL_W
- リファレンス・スイッチ・ピン REF_L および REF_R
- SPI ピン SDI、SDO、SCK（CSN ピンには再割り当てできません）
- UART ピン UART_RXD および UART_TXD
- 制御入力ピン PWM_IN
- 制御出力ピン FAULTN

稼働中に SPI または UART ピンを再割り当てする際には注意が必要です。IC との通信に不要な中断が発生する原因になる可能性があるためです。

CHIP.INPUTS_RAW レジスタには、生のピン入力の状態が含まれる他、様々な機能ブロックに応じ反転され前処理される可能性のある信号が含まれます。それにより、割り当ておよび前処理の設定の変更が可能になります。例えば、PIN_HALL_U_RAW ビットは、HALL_U ピン（17 番ピン）の実際の状態を報告します。対照的に、HALL_U ビットは、デジタル・ホール・ブロックの U 信号の状態を表します。これは、適切な代替機能が設定されている場合に、HALL_U ピンまたは ENC_A ピン（14 番ピン）のどちらかから発生する信号です。

フィードバック・エンジン・ピンの再割り当て

フィードバック・エンジン・ブロックには次のインターフェースがあります。

- 第一および第二 ABN インクリメンタル・エンコーダ。2 つのインクリメンタル・エンコーダを同時に使用できます。
- デジタル・ホール・センサー
- 内蔵 ADC を用いるアナログ・ホール・センサー／SinCos エンコーダ

これらのインターフェースは、ENC_x ピンおよび HALL_x ピン（14 番ピン～19 番ピン）を通じて使用できます。更に、これらのピンは、I/O コントローラの入出力、ブレーキ・チョッパー出力、ホーム・リファレンス・スイッチ入力の一部にアクセスできます。これを表 5 および表 6 に示します（これらの表では、機能グループが同じ色で表現されています）。ここから、使用するフィードバック・インターフェースの様々な組み合わせを引き出せます。フィードバック・エンジン・ピンはすべて、V_{CC_IOF} 電圧源を基準としています。

表 5. フィードバック・エンジン・ピンの再割り当て

PIN NO.	NAME	MAIN FUNCTION	ALT. FUNC 1	ALT. FUNC 2	ALT. FUNC 3 (I/O Controller)	TYPE
14	ENC_A	Incremental Encoder Input A	Inverted Analog Sensor Input U/X*	Digital Hall Input U	DIRECT_OUT[0]	Input/Output + Analog Input
15	ENC_B	Incremental Encoder Input B	Inverted Analog Sensor Input V/N*	Digital Hall Input V	DIRECT_IN[0]	Input + Analog Input
16	ENC_N	Incremental Encoder Input N	Inverted Analog Sensor Input W/Y*	Digital Hall Input W	DIRECT_IN[1] / DIRECT_OUT[1]	Input/Output + Analog Input
17	HALL_U	Digital Hall Input U	Analog Sensor Input U/X	2nd Incremental Encoder Input A	I^2C SDA	Input/Output + Analog Input
18	HALL_V	Digital Hall Input V	Analog Sensor Input V/N	2nd Incremental Encoder Input B	I^2C SCL as Push-Pull	Input/Output + Analog Input
19	HALL_W	Digital Hall Input W	Analog Sensor Input W/Y	2nd Incremental Encoder Input N	DIRECT_OUT[2]	Input + Analog Input

（*差動入力での使用）

表 6. ホール入力ピンのその他の再割り当て

PIN NO.	NAME	MAIN FUNCTION	ALT. FUNC 4	ALT. FUNC 5	ALT. FUNC 6	ALT. FUNC 7	TYPE
17	HALL_U	Digital Hall Input U			Inverted Incremental Encoder Input A, Monitoring Only*	Inverted Incremental Encoder Input A*	Input/Output + Analog Input
18	HALL_V	Digital Hall Input V	Brake Chopper Output	I^2C SCL as Open-Drain	Inverted Incremental Encoder Input B, Monitoring Only*	Inverted Incremental Encoder Input B*	Input/Output + Analog Input
19	HALL_W	Digital Hall Input W	Home Ref. Switch		Inverted Incremental Encoder Input N, Monitoring Only*	Inverted Incremental Encoder Input N*	Input + Analog Input

（*差動入力での使用）

フィードバック・エンジンの使用に関する詳細については、フィードバック・エンジンのセクションを参照してください。

SPI/SSI エンコーダ入力

SPI/SSI エンコーダには、I/O コントローラの直接的な入出力を通じてインターフェースできます。そのため、フィードバック・エンジン・ピンおよびほとんどの再割り当てピンを、このタイプのエンコーダに使用できます。詳細については、プログラマブル I/O コントローラのセクションおよびその SPI_ENC コマンドを参照してください。

デジタル差動 ABN エンコーダ入力

TMC6460 は、デジタル差動 ABN エンコーダ入力もサポートしています。この場合、差動信号ペアは以下のように供給する必要があります。

- 入力 A に対しては、ENC_A ピンおよび HALL_U ピンが非反転および反転差動ペアとして用いられます。
- 入力 B に対しては、ENC_B ピンおよび HALL_V ピンが非反転および反転差動ペアとして用いられます。
- 入力 N に対しては、ENC_N ピンおよび HALL_W ピンが非反転および反転差動ペアとして用いられます。

ピンに対し選択した再割り当てオプションに応じて、2 種類のモードのデジタル差動入力が使用できます。

- 図 26 に示す、デジタル差動 ABN エンコーダ入力。
  - アクティブになるのは、CHIP.IO_MATRIX レジスタの PIN_HALL_U、PIN_HALL_V、PIN_HALL_W の各ビットフィールドが 111_b にセットされ、PIN_ENC_A、PIN_ENC_B、PIN_ENC_N が 00_b（それぞれのデフォルト機能）にセットされている場合です。
  - このモードでは、ABN エンコーダ入力は、それぞれの対応する差動ペア信号によって定義されます。
  - エンコーダ入力の値は、差動入力ペアが有効にならない限り、更新されません。

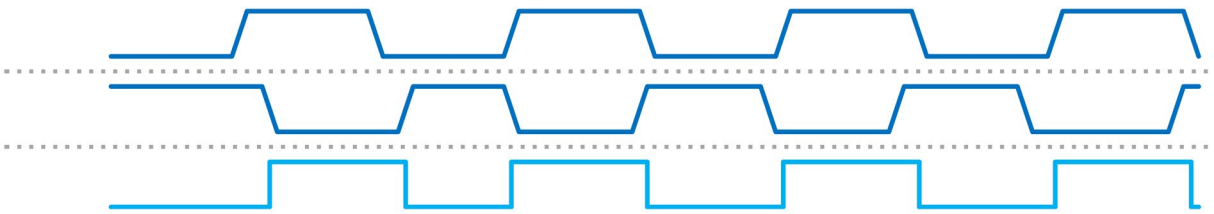


図 26. デジタル差動 ABN エンコーダ入力、ENC_A ピンと HALL_U ピンの例

- 図 27 に示す、デジタル差動 ABN エンコーダのモニタリングのみ。
  - アクティブになるのは、CHIP.IO_MATRIX レジスタの PIN_HALL_U、PIN_HALL_V、PIN_HALL_W の各ビットフィールドが 110_b にセットされ、PIN_ENC_A、PIN_ENC_B、PIN_ENC_N が 00_b（それぞれのデフォルト値）にセットされている場合です。
  - このモードでは、ABN エンコーダ入力は、差動ペアの非反転信号によってのみ、定義されます。
  - 差動ペアの反転信号は、モニタリング目的にのみ用いられます。
  - 差動信号ペアのいずれかが無効な場合、CHIP.STATUS_FLAGS レジスタの DIFF_ENC_FAIL_STATUS ビットがセットされます。更に、CHIP.OUTPUTS_RAW レジスタの INVALID_DIFF_ENC_x（x = A、B、N）ビットが、どの差動入力が無効であるかを正確に示します。
  - 無効な差動信号は、2 つの信号ペアの値が同じである場合に生じます。CHIP.IO_CONFIG レジスタの DIFF_DELAY フィールドの値は、非反転信号が変化してから無効な信号がレジスタに登録されるまでに待機する f_{SYS} クロック・サイクル数を定めます。これによって、信号の値が安定化するための時間が確保されます。例えば、デフォルト値である 7 の場合、最初の信号が変化してからの待機時間は $7/f_{SYS} = 117ns$ になります。

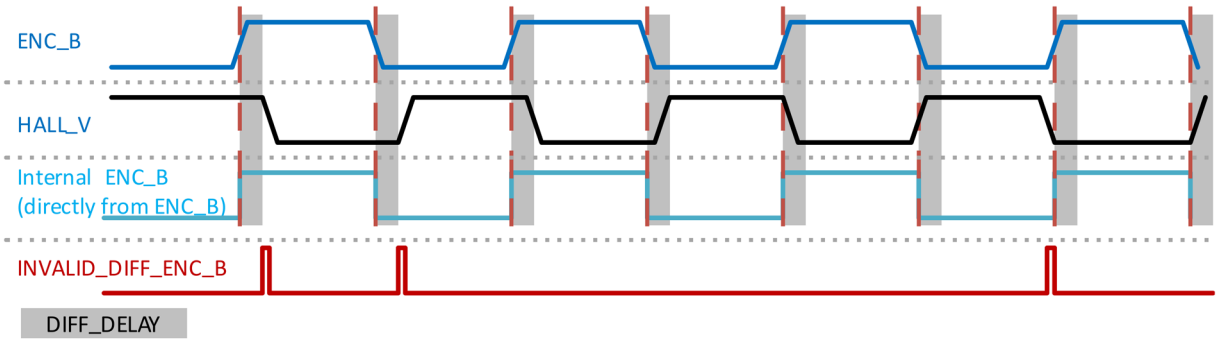


図 27. デジタル差動 ABN エンコーダ入力、ENC_B ピンと HALL_V ピンの例。
DIFF_DELAY は、ENC_B が切り替わった後にアクティブになります。

リファレンス・スイッチ・ピンの再割り当て

デフォルトでは、左および右のリファレンス・スイッチ入力が、REF_L ピンと REF_R ピンで使用できます。追加のホーム・リファレンス・スイッチ入力も使用可能で、これらは、REF_L ピン、REF_R ピン、または HALL_W ピンに割り当てることができます。リファレンス・スイッチ入力は、V_{CC_IOF} 電圧源を基準としています。

表 7 に示すように、これらのピンでは、その代替機能を通じてブレーキ・チョッパー出力および I/O コントローラ入出力にもアクセスできます。

表 7. リファレンス・スイッチ・ピンの再割り当て

PIN NO.	NAME	MAIN FUNCTION	ALT. FUNC 1	ALT. FUNC 2	ALT. FUNC 3 (I/O Controller)	TYPE	VOLTAGE
10	REF_L	Left Reference Switch	Home Reference Switch		DIRECT_IN[0]	Input	V_{CC_IOF}
11	REF_R	Right Reference Switch	Home Reference Switch	Brake Chopper Output	DIRECT_OUT[1]	Input/Output	V_{CC_IOF}

リファレンス・スイッチ機能は、ランプ・ジェネレータ・ブロックの一部です。詳細については、ランプ・ジェネレータのセクションを参照してください。

通信インターフェース・ピンの再割り当て

通信インターフェース・ピンは、その他のペリフェラルに再割り当てすることもできます。これは特に、UART または SPI のどちらか一方のみを用いて TMC6460 と通信を行う場合に有用です。通信ピンは、V_{CC_IO} 電圧源を基準としています。

表 8 に示すように、SPI ピンは、ブレーキ・チョッパー出力および I/O コントローラ入出力にもアクセスできる代替機能を備えています。これらのピンのいずれかが代替機能で使用されている場合、SPI を通じた通信はできません。SPI 通信を使用しない場合は、CSN ピンをアサートしないでください。

表 8. SPI ピンの再割り当て

PIN NO.	NAME	MAIN FUNCTION	ALT. FUNC 1	ALT. FUNC 2	ALT. FUNC 3 (I/O Controller)	TYPE	VOLTAGE
1	CSN	Chip Select	(No alternate functions, keep at high level if SPI is not used.)			Input	V_{CC_IO}
36	SCK	Serial Clock			DIRECT_IN[1]	Input	V_{CC_IO}
37	SDI	Serial Data In	DIR_IN Input		DIRECT_IN[0]	Input	V_{CC_IO}
38	SDO	Serial Data Out	Brake Chopper Output		DIRECT_OUT[0]	Output	V_{CC_IO}

表 9 に示すように、UART ピンにも同様の代替機能があります。

表 9. UART ピンの再割り当て

PIN NO.	NAME	MAIN FUNCTION	ALT. FUNC 1	ALT. FUNC 2 (I/O Controller)	ALT. FUNC 3 (I/O Controller)	TYPE	VOLTAGE
8	UART_TXD	UART Data Output	Brake Chopper Output	DIRECT_OUT[0]	DIRECT_OUT[1]	Output	V_{CC_IO}
9	UART_RXD	UART Data Input	DIR_IN Input		DIRECT_IN[0]	Input	V_{CC_IO}

稼働中に通信ピンを再割り当てする際には注意が必要です。IC との通信に不要な中断が発生する原因になる可能性があるためです。

制御信号ピンの再割り当て

制御信号には、PWM 入力、ドライバ・イネーブル入力、フォルト出力があります。制御信号は、V_{CC_IO} 電圧源を基準としています。表 10 に示すように、制御信号ピンの一部の機能を再設定することもできます。

表 10. 制御ピンの再割り当て

PIN NO.	NAME	MAIN FUNCTION	ALT. FUNC 1	ALT. FUNC 2	ALT. FUNC 3 (I/O Controller)	TYPE	VOLTAGE
12	PWM_IN	PWM Control Input	AIN Analog Control Input		DIRECT_IN[1]	Digital input + Analog Input	V_{CC_IO}
34	DRV_EN	Driver Enable	(No alternate functions available.)			Input	V_{CC_IO}
35	FAULTN	Open-Drain Fault Output	Open-Drain Interrupt/Event Output	Push-Pull Fault Output	Push-Pull Interrupt/Event Output	Output	V_{CC_IO}

FAULTN ピンでは、フォルト出力と割込み出力の 2 つのケースを明確に区別できます。大きな違いは、フォルト出力は選択したフラグの実際の状態（CHIP.STATUS_FLAGS レジスタ）を反映するのに対し、割込み出力は、選択したフラグのラッチされた値（CHIP.EVENTS レジスタ）を反映し、それによって、フォルト状態が発生した場合に、その条件が現在アクティブでなくても検証できる点です。どちらの場合も、出力はオープン・ドレインまたはプッシュプルに設定できます。FAULTN ピンのフラグ選択マスクの詳細については、FAULTN ピンのセクションを参照してください。

入力設定の説明図

このセクションでは、前記ピン設定のセクションおよびピンの再割り当てのセクションのまとめとして、特定のアプリケーションで実現できる様々な入力設定を示します。

図 28 では、ABN と ABN2 の両方のフィードバック信号に適用できる様々なオプションを示しています。

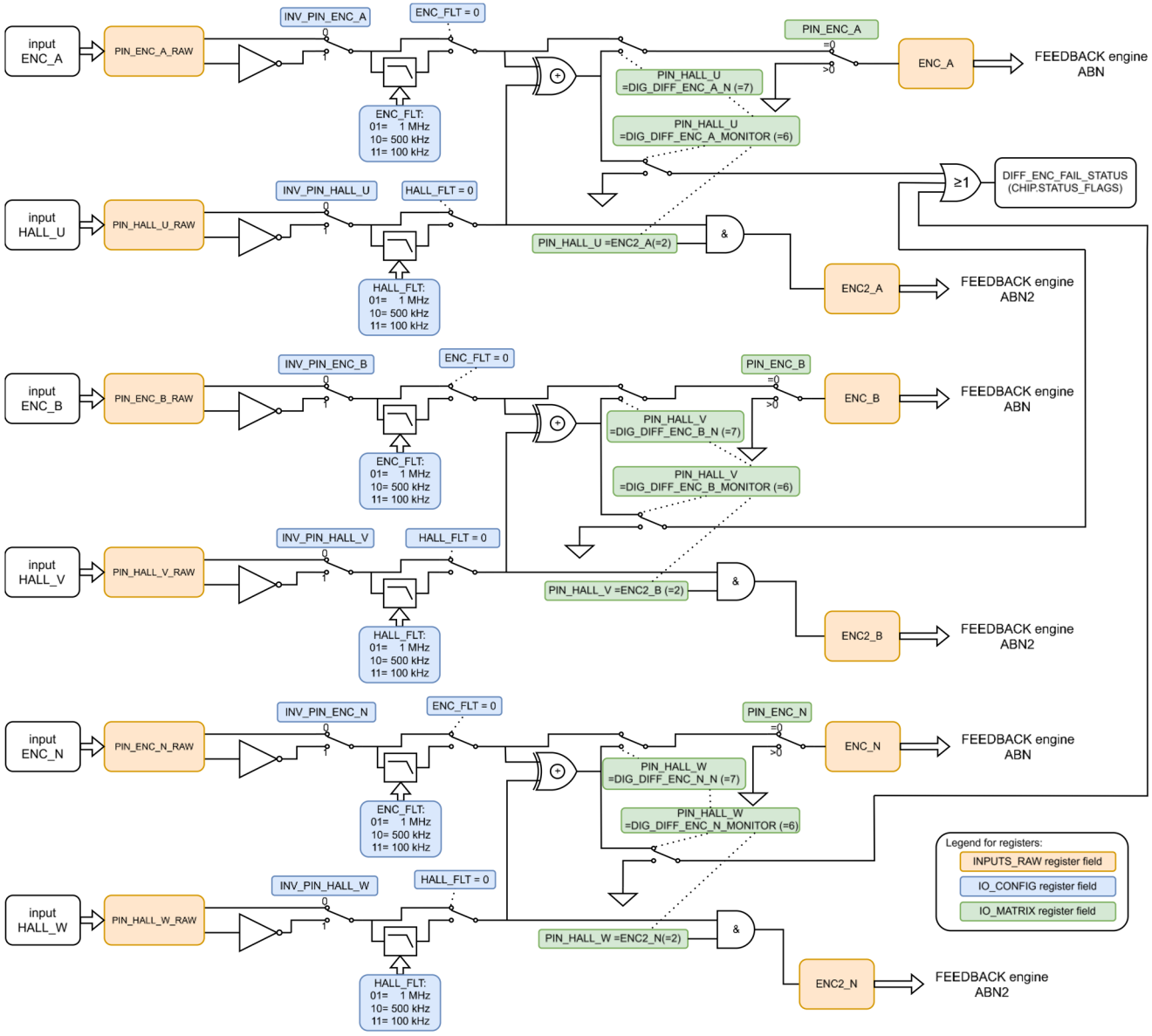


図 28. エンコーダ・フィードバックの設定オプション

図 29 では、デジタル・ホールのフィードバック信号に適用できる様々なオプションを示しています。その他の HALL_MAP フィールドは HALL.MAP_CONFIG レジスタにあります。

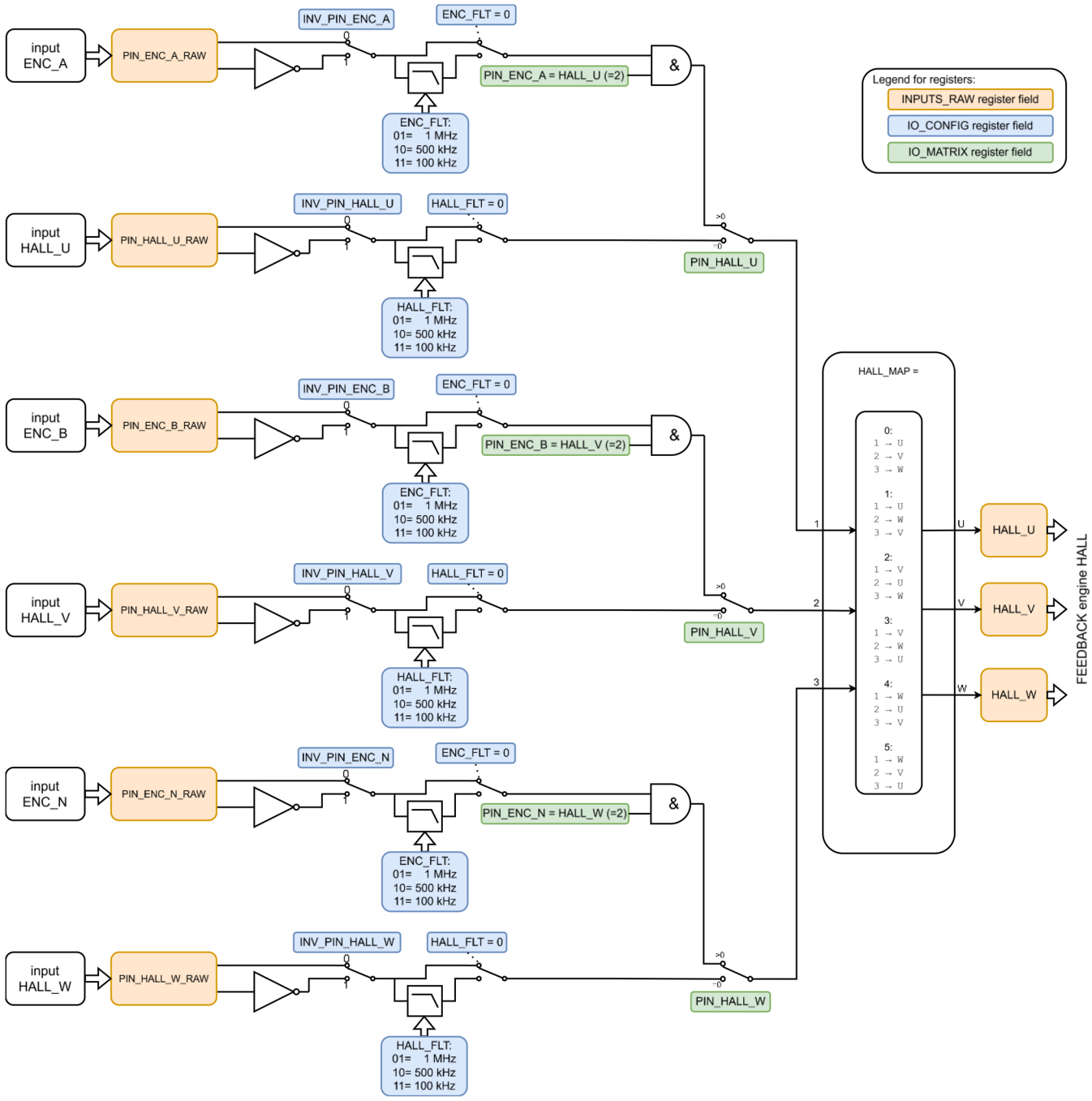


図 29. デジタル・ホール・フィードバックの設定オプション

図 30 では、アナログ・センサーの様々なオプションを示しています。更に、この図には、PWM_IN 入力ピンからの追加アナログ入力オプションも示されています。ANA_DIV_VCCIO フィールドと ANA_DIV_VCCIOF フィールドは、CHIP.IO_CONFIG レジスタの一部です。

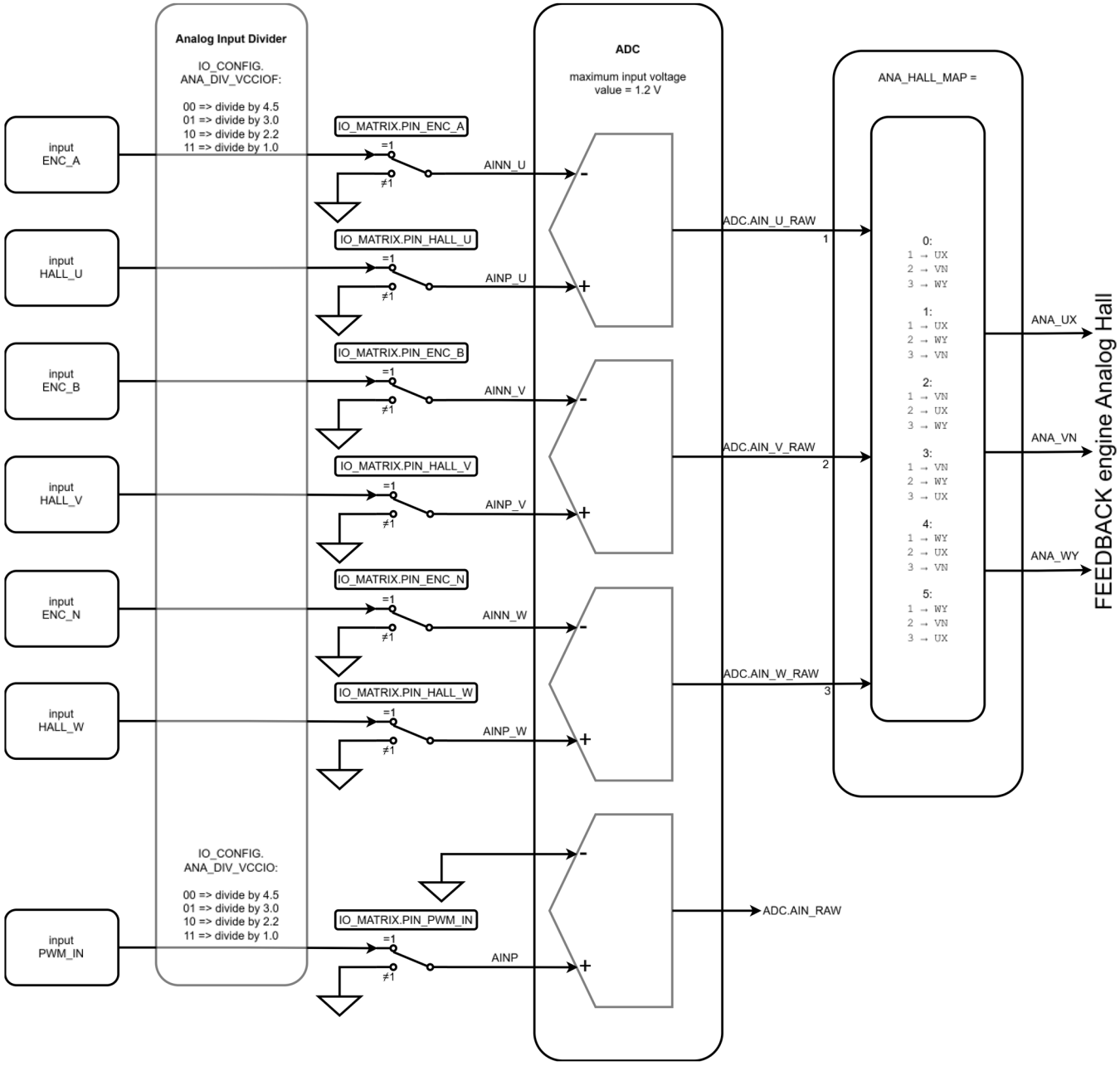


図 30. アナログ・ホール・フィードバックおよびアナログ入力の設定オプション

図 31 では、リファレンス・スイッチ入力ピンの設定オプションを示しています。CHIP.IO_CONFIG の USE_PIN_REF_L_AS_REF_R ビットまたは USE_PIN_REF_R_AS_REF_L ビットのどちらかをセットすると、1 つの入力ピンを両方の内部リファレンス信号に割り当てることができます。両方のフィールドがアクティブな場合は、どちらのオプションも非アクティブになります。

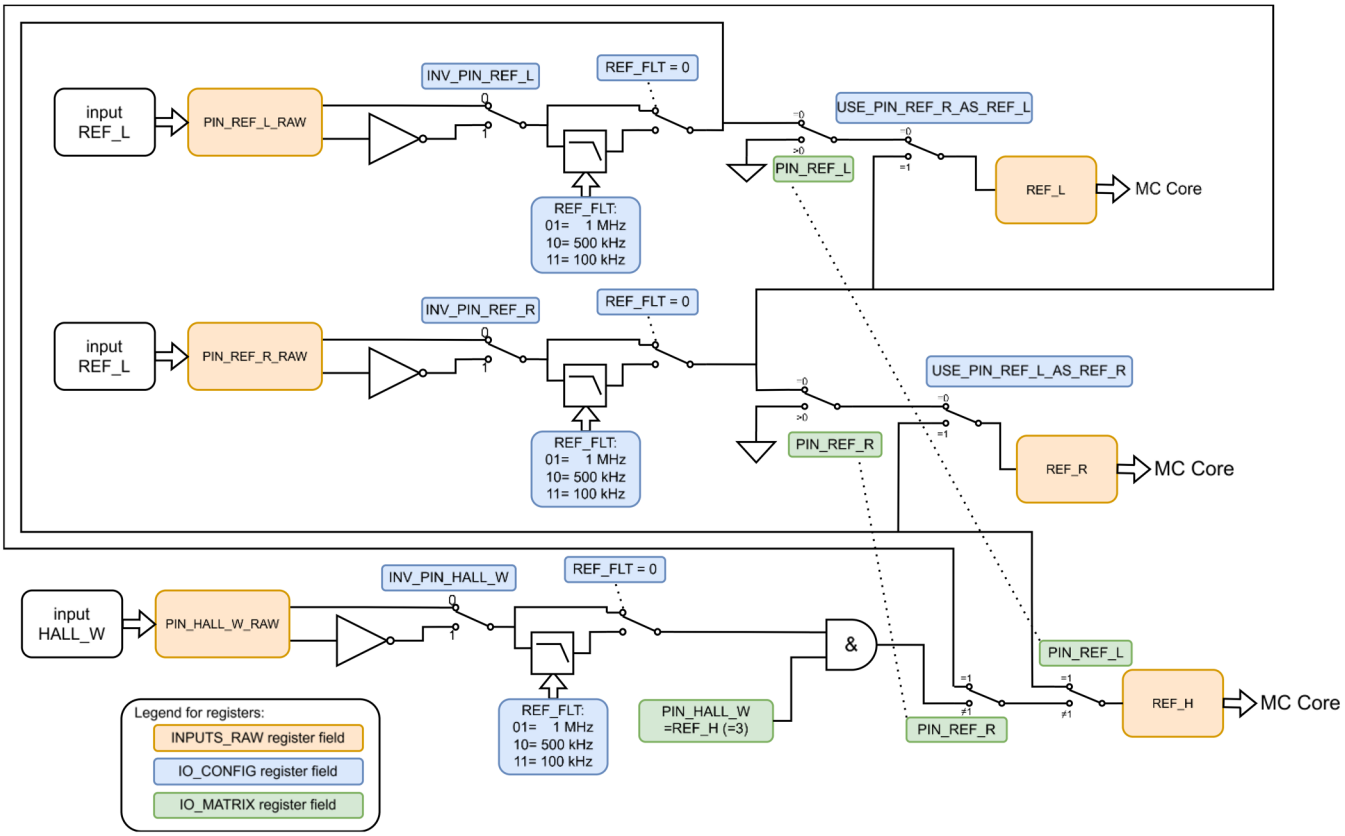


図 31. リファレンス・スイッチ入力の設定オプション

ADC の設定と測定

ADC エンジンは、f_{SYS} = 60MHz のシステム・クロックで動作します。その設定は、パワーアップ中に自動的に実行されます。あるいは、PLL が再設定された場合や、I/O コントローラがリセットされた場合にも、自動的に実行されます。これらの場合、追加の操作は不要です。ADC の設定が実行された後、MCC_CONFIG.GDRV レジスタの USE_INTERNAL_R_REF ビットが変更された場合、I/O コントローラを手動でリセットし（IO_CONTROLLER.CONTROL レジスタの RESET ビット）、ADC の自動設定を再度行うことを推奨します。

ADC エンジンは、3 つのモーター位相のすべてについて、その電流を自動的に測定します。ローサイドの抵抗および電流検出アンプのゲイン設定のみ、アプリケーションに応じて選択する必要があります。適切な設定を見出す方法に関するガイダンスについては、電流アクイジションのセクションを参照してください。

アナログ入力分圧器のセクションで言及したように、IO 電圧源領域はそれぞれ、そのアナログ入力に対し独立した分圧器を有しています。表 11 に、CHIP.IO_CONFIG レジスタの ANA_DIV_VCCIO フィールドおよび ANA_DIV_VCCIOF フィールドの設定値に応じた、様々な DIV_VCCIO および DIV_VCCIOF の分圧比を示します。

ADC の入力電圧（分圧後の電圧）が 1.25V を超えないよう、分圧比は入力ピンで予想される最高電圧に応じて選択する必要があります。表 11 には、異なる分圧オプションに対応した最高入力電圧も示されています。

表 11. アナログ入力分圧器オプション

ANA_DIV_VCCIO/ANA_DIV_VCCIOF	DIV_VCCIO/DIV_VCCIOF	MAXIMUM VOLTAGE VALUE FOR V_{CC_IO}/V_{CC_IOF} ANALOG INPUTS
00 (default)	4.5	5.5
01	3.0	3.6
10	2.2	2.7
11	1.0	1.2

ANA_DIV_VCCIO と ANA_DIV_VCCIOF のどちらのフィールドに対しても、00_bがデフォルト・オプションです。

角度フィードバック ADC 測定

ADC 測定を更に使用することにより、モーター角度のフィードバックを行うアナログ・センサーとインターフェースできます。これらのセンサーは、HALL_U、HALL_V、HALL_W の各フィードバック・エンジン・ピン、および、差動入力が必要な場合にはオプションで ENC_A、ENC_B、ENC_N の各ピンに接続できます。これらはすべて一定の代替機能に設定します。そのため、これらの ADC 測定は、V_{CC_IOF} 電源を基準とします。表 12 に、CHIP.IO_MATRIX レジスタに設定されたピン代替機能に従い、角度フィードバック ADC 測定のソースを示します。

表 12. アナログ・エンコーダ入力値の計算オプション

PIN_ENC_A / PIN_HALL_U	= AINN_U	≠ AINN_U
= AINP_U	AINU = V(HALL_U vs. ENC_A)	AINU = V(HALL_U vs. GND)
≠ AINP_U	AINU = 0	AINU = 0
PIN_ENC_B / PIN_HALL_V	**= AINN_V**	**≠ AINN_V**
= AINP_V	AINV = V(HALL_V vs. ENC_B)	AINV = V(HALL_V vs. GND)
≠ AINP_V	AINV = 0	AINV = 0
PIN_ENC_N / PIN_HALL_W	**= AINN_W**	**≠ AINN_W**
= AINP_W	AINW = V(HALL_W vs. ENC_N)	AINW = V(HALL_W vs. GND)
≠ AINP_W	AINW = 0	AINW = 0

フィードバック・エンジン・ピンからは、3 つの ADC 測定値が供給されます。それらの生の値は、AIN_V_AIN_U_RAW レジスタの AIN_U フィールドおよび AIN_V フィールドと、AIN_AIN_W_RAW レジスタの AIN_W フィールドから読み出せます。すべての AIN_x フィールド（x = U、V、W）には 16 ビットの符号付きの値が保持されており、以下の式を通じて ADC の測定入力電圧と関係付けられます。

$$AIN_x = \frac{V_{AnalogSensor} \times 2^{15}}{DIV_VCCIOF \times 1.25\ [V]} \qquad V_{AnalogSensor} = AIN_x \times \frac{1.25\ [V]}{2^{15}} \times DIV_VCCIOF$$

分圧比 DIV_VCCIOF は、表 11 から得られます。

その他の ADC 測定

更に 2 つの ADC 測定値が、V_{CC_IO} 電源を基準としています。

- AIN は、PWM_IN ピンで使用できる外部アナログ測定値です。CHIP.IO_MATRIX レジスタで PIN_PWM_IN = AIN が選択されている場合、このピンのアナログ値が自動的に計算されます。計算された値は、ADC.AIN_AIN_W_RAW レジスタの AIN ビットフィールドから読み出せます。このフィールドは、次式を用いる 16 ビットの符号付きの値でもあります。

$$AIN = \frac{V_{AnalogPWM_IN} \times 2^{15}}{DIV_VCCIO \times 1.25\ [V]} \qquad V_{AnalogPWM_IN} = AIN \times \frac{1.25\ [V]}{2^{15}} \times DIV_VCCIO$$

- TEMP_EXT は、TEMP ピンで使用できる外部アナログ測定値です。外部温度センサーまたはサーミスタ-抵抗配置とインターフェースすることで、モーター温度などのモニタリングが可能になります。このピンのアナログ値は常に計算されます。この値は、ADC.TEMP_RAW レジスタの TEMP_EXT ビットフィールドから読み出せます。このフィールドは、次式を用いる 16 ビットの符号付きの値でもあります。

$$TEMP_EXT = \frac{V_{AnalogTEMP} \times 2^{15}}{DIV_VCCIO \times 1.25\,[V]} \qquad V_{AnalogTEMP} = TEMP_EXT \times \frac{1.25\,[V]}{2^{15}} \times DIV_VCCIO$$

分圧比 DIV_VCCIO は、表 11 から得られます。

2 つの追加アナログ値も常に計算されます。

- ADC.TEMP_RAW レジスタの TEMP_INT フィールドは内部温度 T_J を表しています。これは、次式を用いる 16 ビットの符号付きの値です。

$$T_J = \left(TEMP_INT \times \frac{1.25\,[V]}{2^{15}} - 0.622[V]\right) \times \frac{1250}{3}\left[\frac{°C}{V}\right]$$

- 最後に、ADC.VM_I3_RAW レジスタの V_M フィールドは、V_S ピンで内部測定された電圧値を表します。これは、次式を用いる 16 ビットの符号付きの値です。

$$V_{VM} = VM \times \frac{1.25\,[V]}{2^{15}} \times \frac{287}{9}$$

モーターの相電流および AIN アナログ入力は、PWM サイクルごとに測定されます。その他のアナログ値（TEMP_EXT、TEMP_INT、V_M）はすべて、$f_{PWM} \leq 50$kHz の場合は PWM サイクルごとに、$f_{PWM} > 50$kHz の場合は、3PWM サイクルごとに計算されます。

入力での電圧が ADC の有効範囲を超えたため AIN_U、AIN_V、AIN_W、または AIN のいずれかの生の値がクリップされた場合、ADC.STATUS レジスタの対応するイベント・ビット AIN_U_CLIPPED、AIN_V_CLIPPED、AIN_W_CLIPPED、AIN_CLIPPED がセットされます。これらのビットは、CHIP.STATUS_FLAGS レジスタの AIN_CLIPPED_STATUS ビットおよびそれに対応する CHIP.EVENTS レジスタの AIN_CLIPPED_EVENT ビットにマージされます。

同様に、V_M、TEMP_EXT、または TEMP_INT の生の値がクリップされた場合、ADC.STATUS レジスタの対応するイベント・ビット VM_CLIPPED、TEMP_EXT_CLIPPED、TEMP_INT_CLIPPED がセットされます。これら 3 つのイベントは、CHIP.STATUS_FLAGS レジスタの VM_TEMP_CLIPPED_STATUS ビットにマージされます。

表 13 に、上述のマージされたステータス・ビットとイベント・ビットの関係を示します。

表 13. ADC のクリップ・イベントおよびフラグの概要

RAW ADC INPUT REGISTER FIELD	CORRESPONDING EVENT BIT IN ADC.STATUS REGISTER	MERGED STATUS FLAG IN CHIP.STATUS_FLAGS	MERGED EVENT BIT IN CHIP.EVENTS
AIN_U	AIN_U_CLIPPED	AIN_CLIPPED_STATUS	AIN_CLIPPED_EVENT
AIN_V	AIN_V_CLIPPED		
AIN_W	AIN_W_CLIPPED		
AIN	AIN_CLIPPED		
VM	VM_CLIPPED	VM_TEMP_CLIPPED_STATUS	VM_TEMP_CLIPPED_EVENT
TEMP_INT	TEMP_INT_CLIPPED		
TEMP_EXT	TEMP_EXT_CLIPPED		

ADC.STATUS の特定のイベント・ビットをクリアしてからでないと、CHIP.EVENTS の対応するマージされたイベント・ビットをクリアできない点に注意してください。

モーター角度フィードバックの設定

TMC6460 は、数種類の角度フィードバックをサポートしています。つまり、ABN エンコーダ、デジタルおよびアナログ・ホール・センサー、SinCos 絶対エンコーダ、SPI/SSI 絶対エンコーダです。これらの各フィードバック・タイプに対し適切な設定を行ってからでないと、フィードバック・エンジンの角度ソースとして使用できません。

このセクションでは、これらの各フィードバック・タイプに必要な設定手順を説明します。

インクリメンタル ABN エンコーダ

インクリメンタル・エンコーダには 2 つの位相シフトされたパルス信号 A および B があり、それらを基に移動の量と方向が決定できます。インクリメンタル・エンコーダによっては、ヌル・ポジション信号 N（ゼロ・パルス信号 Z とも呼ばれます）が追加されており、これを基に絶対ポジション・リファレンスが求められます。インクリメンタル・エンコーダ（以降、ABN エンコードと呼びます）は、1 回転あたりのラインまたはパルスの数（PPR）で特徴づけられます。各パルスには 2 つのエッジ（立上がりおよび立下がり）があり、パルスを供給する 2 つの信号（A および B）があります。したがって、TMC6460 は、ABN エンコーダの 1 ラインあたり 4 カウントを登録できます。1 回転あたりのカウント数は CPR と略記され、PPR × 4 に相当します。

TMC6460 には 2 つの ABN デコーダ・モジュールがあります。つまり、2 つの ABN エンコーダを同時に使用できます。第一のエンコーダ・モジュールは ABN レジスタ・ブロックに対応し、その入力を ENC_A、ENC_B、ENC_N の各ピンから取得します。逆に言うと、第二のエンコーダ・モジュールは、ABN2 レジスタ・ブロックに対応し、その入力を HALL_U、HALL_V、HALL_W の各ピン（010_b にセットされた代替機能）から取得します。2 つのデコーダ・ブロックは独立である点に注意してください。そのため、デコーダの入力を混合すること、例えば、1 つのエンコーダを ENC_A ピンと HALL_V ピンに接続することはできません。

A および B の両方の入力信号が同じ内部クロック・サイクル内で変化した場合、ABN.EVENTS（あるいは、第二 ABN エンコーダを使用している場合は、ABN2.EVENTS）レジスタの INVALID_SIGNAL_EVENT がセットされます。これは、ABN エンコーダのハードウェアまたは接続の問題を検出する機構、または、TMC6460 の更新レート能力を超える入力信号を検出する機構としての役割を果たします。この故障が発生した場合、デコーダ・カウントは、その特定のスイッチングに対してインクリメントもデクリメントもせず、その後は通常の動作が維持されます。無効入力に対する 2 つの信号は、CHIP.STATUS_FLAGS の ABN_FAIL_STATUS ビットに OR 接続され、CHIP.EVENTS レジスタの対応する ABN_FAIL_EVENT ビットにラッチされます。ABN_FAIL_EVENT をクリアするには、まず両方のイベント・ビットをクリアする必要があります。

ABN デコーダの設定

ABN デコーダは、検出された移動方向に応じて、A および B 信号の検出エッジごとに ABN.COUNT レジスタの値をインクリメントまたはデクリメントします。このカウント方向は、ABN.CONFIG レジスタの INV_DIR ビットをセットすることで変更できます。更に、N パルスが検出された場合、ABN.EVENTS レジスタの N_EVENT ビットが自動的にセットされ、最新の COUNT 値が ABN.COUNT_N_CAPTURE レジスタにラッチされます。

いくつかのエンコーダに対して N パルスの検出精度を向上するために、結合信号モードが備わっています。このモードでは、N パルスがカウントされるのは、N 信号のみに基づく場合とは逆に、A、B、N 信号がすべて同時にハイになる場合です。これを行うために、ピン設定のセクションで説明したように、A、B、または N の信号が反転される場合があります。このモードを使用するには、ABN.CONFIG レジスタの COMBINED_N ビットをセットします。代表的な ABN エンコーダの信号とこの機能を用いた場合に生じる結合 N パルス信号を、図 32 に示します。この機能を用いる場合、3 つの信号はすべて、使用エンコーダにおいて実際に同時にハイ・レベルを示すよう注意することが必要です。これはエンコーダがどちらの方向に回転する場合にもあてはまらなくてはなりません。この条件が満たされない場合、N パルスは検出されません。

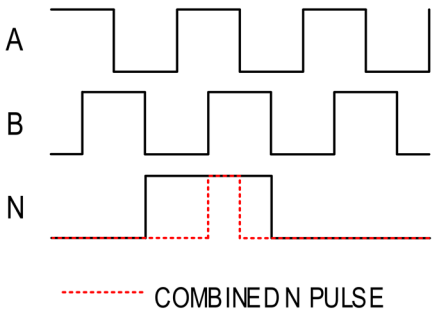


図 32. 代表的な ABN エンコーダ信号と結合 N パルス

デコーダが所与の機能を発揮するには、1 回転が完了したことを検出する方法が必要です。それには 2 つの方法があります。エンコーダの CPR に基づいてリセット値を設定する方法とエンコーダの N パルスをリセット信号として用いる方法です。

エンコーダの CPR をカウント・リセット機構として用いるには、ABN.CONFIG レジスタの CPR フィールドを、デコーダがゼロにリセットされるまでにカウントする最大値に設定します。そのため、CPR フィールドには cpr_encoder − 1 の値を書き込みます。例えば、接続されている ABN エンコーダの分解能が 1024 ライン（PPR = 1024）である場合、CPR = (1024 × 4) − 1 = 4095 の値を書き込みます。この機構を用いない場合は、CPR フィールドをそのリセット値である 16777215 のままにします。

代替方法として、エンコーダの N 信号を用いて ABN.COUNT のデコーダのカウント値をリセットできます。この機構を用いるには、ABN.CONFIG レジスタの CLN ビットをセットします。N パルスを用いて、COUNT レジスタをリセットする代わりに任意の値を書き込むこともできます。これを行うには、ABN.COUNT_N_WRITE に必要な値を書き込みます（デフォルトでは 0）。

第二 ABN エンコーダの設定は、代わりに ABN2 ブロックのレジスタ、つまり、ABN2.CONFIG、ABN2.COUNT というようなレジスタを用いて、同じ方法で行うことができます。

デジタル・ホール・センサー

デジタル・ホール・センサーは、モーターの 3 つの位相に対応する 3 つのデジタル信号を供給します。その信号からは、それぞれが電気的な 1 回転のうちの 60º ずつを表す 6 つの区画において、現在モーターがどの区画にあるかを判定できます。

ホール入力信号が測定時に同じ値になる場合は、HALL.DIG_EVENTS レジスタの INVALID_SIGNAL_EVENT ビットが自動的にセットされ、それに対応する状態変化は処理されません。これは、ホール・センサーのハードウェアまたは接続の問題を検出する機構として機能します。無効状態に対する信号は、CHIP.STATUS_FLAGS レジスタの HALL_FAIL_STATUS ビットに転送され、CHIP.EVENTS レジスタの対応する HALL_FAIL_EVENT ビットにラッチされます。HALL_FAIL_EVENT をクリアするには、まず INVALID_SIGNAL_EVENT ビットをクリアする必要があります。

デジタル・ホール・センサー・デコーダの設定

デジタル・ホール・デコーダは 3 つの信号を受け取り、それらの値をルックアップ・テーブルと比較して対応するホール区画を取得します。現在のホール区画とは、電気的な 1 回転のうちの 60º からなる 6 区画におけるモーターの現在位置を指します。ホール区画は、0～5 の値で表され、HALL.DIG_COUNT レジスタの COUNT フィールドから読み出せます。

ピンがホール・センサーを使用できるよう正しく設定されていると仮定すると、ホール・デコーダに必要な唯一の設定は、ホール信号の適切な割り当てです。これは、ピン設定のセクションで説明したように、ピン入力反転と組み合わせた、HALL.MAP_CONFIG レジスタの HALL_MAP フィールドを通じて変更できます。信号の割り当てを変更することで、モーター位相とホール・センサー入力の接続の間の不整合を補償できます。言い換えると、信号の再割り当てを行うことで、現在のホール区画のオフセットを効果的に適用したり、カウントの方向を効果的に逆転したりできます。表 14 に、割り当てとピン入力反転の設定に応じて可能な、ホール区画オフセットおよびカウント方向の反転の概要を示します。

表 14. ホール・センサー区画のオフセットおよびカウント方向の設定

HALL_MAP[2:0]	PIN INPUT INVERSION	HALL.DIGITAL_COUNT RESULTING SECTOR OFFSET	HALL.DIGITAL_COUNT RESULTING COUNTING DIRECTION
000	Disabled	0°	Non-inverted
011	Enabled	60°	Non-inverted
100	Disabled	120°	Non-inverted
000	Enabled	180°	Non-inverted
011	Disabled	240°	Non-inverted
100	Enabled	300°	Non-inverted
001	Disabled	0°	Inverted
010	Enabled	60°	Inverted
101	Disabled	120°	Inverted
001	Enabled	180°	Inverted
010	Disabled	240°	Inverted
101	Enabled	300°	Inverted

この表では、ピン入力反転（PIN INPUT INVERSION）の欄は、ホール・センサーに使用されるピンに対応する CHIP.IO_CONFIG レジスタの 3 つのピン反転ビットを指しています。つまり、ホール・センサーに HALL_x ピン（x = U、V、W）を用いる場合は INV_PIN_HALL_x ビット、ENC_x ピン（x = A、B、N）を用いる場合は INV_PIN_ENC_x ビットです。

なお、ホール・デコーダのカウント方向を正しく設定することは、適切なモーター整流や速度およびポジションのトラッキングを行う上で必須です。ただし、上述のホール区画オフセットは、モーター整流にのみ関係します。つまり、ホール・センサー角度の内部の phi_e 角に対するアライメントにのみ関係します。phi_e の計算のセクションでこれについて詳述します。

アナログ・ホール・センサー

アナログ・センサー・デコーダは三相（アナログ・ホール）入力を受け入れます。その入力には、相互に 120º のオフセットのある 3 つのサイン波信号が見込まれます。

アナログ・センサー・デコーダには、それぞれのアナログ入力をオフセットとスケーリング値を用いて補償する機能があります。各アナログ入力値は、補償が適用される前および後のどちらでも読み出せます。補償されたアナログ入力値を用いた逆正接計算が行われ、アナログ・ホール角が求められます。

アナログ・デコーダをアナログ・ホール・センサー用に設定する方法

アナログ・デコーダを三相（アナログ・ホール）センサー用に設定するには、HALL.ANA_CONFIG レジスタの ANA_MODE ビットフィールドを 1_b（いわゆる UVW モード）に設定します。次いで、表 15 をリファレンスとして、以下の手順に従います。

1. 各アナログ入力のオフセット値は、モーターの回転時の補償されたサイン波がゼロを中心となるように設定します。オフセットは符号付きの値で、スケーリング前の生の入力値に付加されます。
2. 各アナログ入力のスケーリング値は、補償された 3 つのサイン波がすべて同じ振幅になるように調整する必要があります。スケーリング値が 1024 の場合、生の ADC 値からのスケーリング比が 1:1 であることを意味します。負のスケーリング値は生のアナログ入力を反転します。スケーリング値を選択する場合、スケーリングされた ADC 値が 21845 の絶対値（符号付き 16 ビット値の 3 分の 2）を超えないようにする必要があります。
3. スケーリングを変更した後、再度オフセットを調整する必要が生じる場合もあります。

表 15. アナログ・ホール・センサーの設定レジスタ・フィールド

	U	V	W
Raw ADC Value Field	ADC.AIN_V_AIN_U_RAW:: AIN_U	ADC.AIN_V_AIN_U_RAW:: AIN_V	ADC.AIN_AIN_W_RAW:: AIN_W
Offset Value Field	HALL.ANA_UX_CONFIG:: UX_OFFSET	HALL.ANA_VN_CONFIG:: VN_OFFSET	HALL.ANA_WY_CONFIG:: WY_OFFSET
Scale Value Field	HALL.ANA_UX_CONFIG:: UX_SCALE	HALL.ANA_VN_CONFIG:: VN_SCALE	HALL.ANA_WY_CONFIG:: WY_SCALE
Compensated Value Field	HALL.ANA_UX_OUT:: UX_OUT	HALL.ANA_VN_OUT:: VN_OUT	HALL.ANA_WY_OUT:: WY_OUT

UX_OUT、VN_OUT、WY_OUT の各レジスタ・フィールドは、オフセットおよびスケーリングを適用した後の ADC 値を表します。

生の ADC 値と補償値を比較するプロット例を図 33 に示します。ここでは、以下のオフセット値およびスケーリング値が用いられています：UX_OFFSET = 15143、UX_SCALE = 2549、VN_OFFSET = 15108、VN_SCALE = 2501、WY_OFFSET = 15141、WY_SCALE = 2647。

アナログ・デコーダから求められる角度は、HALL.ANA_OUT レジスタの ANA_PHI フィールドから読み出せます。

表 16 に示すように、区画のオフセットおよび ANA_PHI フィールドのアナログ・ホール角のカウント方向の反転は、アナログ入力反転と併せて、HALL.MAP_CONFIG レジスタの ANA_HALL_MAP フィールドに適切な値を設定することにより実現できます。なお、生の ADC 値 AIN_x（x = U、V、W）を、表 15 に示すように、アナログ・センサーの設定値および出力値 x_OFFSET、x_SCALE、x_OUT（x = UX、VN、WY）に割り当てることは、ANA_HALL_MAP = 000_b のデフォルト値を用いている場合にのみ有効である点に注意してください。レジスタ・マップのフィールドの説明および図 30 の右側に、様々な割り当てオプションが記載されています。

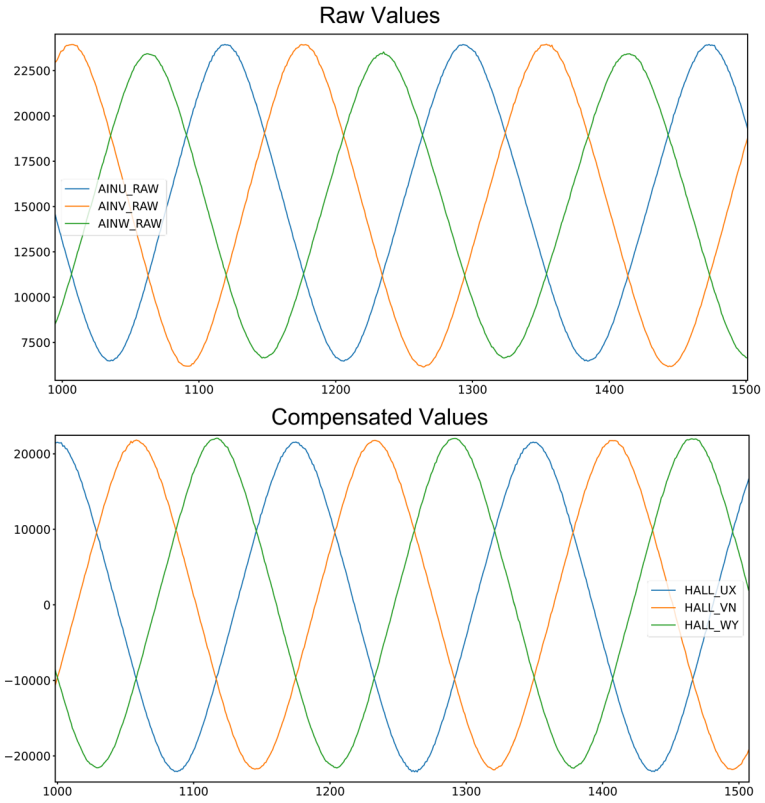


図 33. アナログ三相エンコーダのスケーリングとオフセットのセットアップ例

表 16. アナログ・ホール・センサー区画のオフセットおよびカウント方向の設定

ANA_HALL_MAP[2:0]	ANALOG INPUT INVERSION	HALL.ANA_OUT::ANA_PHI RESULTING ANGLE OFFSET	HALL.ANA_OUT::ANA_PHI RESULTING COUNTING DIRECTION
000	Disabled	0°	Non-inverted
011	Enabled	60°	Non-inverted
100	Disabled	120°	Non-inverted
000	Enabled	180°	Non-inverted
011	Disabled	240°	Non-inverted
100	Enabled	300°	Non-inverted
001	Disabled	0°	Inverted
010	Enabled	60°	Inverted
101	Disabled	120°	Inverted
001	Enabled	180°	Inverted
010	Disabled	240°	Inverted
101	Enabled	300°	Inverted

表 16 において、アナログ入力反転（ANALOG INPUT INVERSION）の欄は、表 15 およびその上の対応する手順で説明したのと同様、UX_SCALE、VN_SCALE、WY_SCALE の 3 つのフィールドにおいて正または負の値を用いることを意味します。

なお、アナログ・センサー・デコーダのカウント方向を正しく設定することは、適切なモーター整流や速度およびポジションのトラッキングを行う上で必須です。ただし、上述の角度オフセットは、モーター整流にのみ関係します。つまり、アナログ・ホール・センサー角度の内部の phi_e 角に対するアライメントにのみ関係します。phi_e の計算のセクションでこれについて詳述します。

アナログ SinCos エンコーダ

アナログ・センサー・デコーダはアナログ SinCos エンコーダ入力も受け入れます。その入力には、相互に 90º のオフセットのある 2 つのサイン波信号が見込まれます。

アナログ・センサー・デコーダには、それぞれのアナログ入力をオフセットとスケーリング値を用いて補償する機能があります。両方のアナログ入力値、つまり補償前と後のアナログ入力値は、各入力のレジスタ・マップから読み出せます。補償されたアナログ入力値を用いた逆正接計算が行われ、エンコーダ角が求められます。

アナログ・デコーダを SinCos エンコーダ用に設定する方法

アナログ・デコーダを SinCos エンコーダ用に設定するには、HALL.ANA_CONFIG レジスタの ANA_MODE ビットフィールドを 0_b（いわゆる XY モード）に設定します。

SinCos エンコーダによっては、モーターが 1 回転するごとに 2 つのサイン波サイクルを発生するものもあります。この種のエンコーダを用いる場合は、HALL.ANA_CONFIG レジスタの TWO_CYCLE_MODE_EN ビットをセットします。

三相センサーの場合と同様、サイン波がゼロを中心とするように、また、モーターの回転中に同じ振幅になるように、オフセット値とスケーリング値を定める必要があります。このタイプのエンコーダが用いるアナログ信号は 2 つのみであるため、UX 入力と WY 入力に対してのみこのように設定する必要があります。設定のリファレンスとして表 17 を使用してください。

表 17. アナログ SinCos エンコーダの設定レジスタ・フィールド

	X	Y
Raw ADC Value Field	ADC.AIN_V_AIN_U_RAW::AIN_U	ADC.AIN_AIN_W_RAW::AIN_W
Offset Value Field	HALL.ANA_UX_CONFIG::UX_OFFSET	HALL.ANA_WY_CONFIG::WY_OFFSET
Scale Value Field	HALL.ANA_UX_CONFIG::UX_SCALE	HALL.ANA_WY_CONFIG::WY_SCALE
Compensated Value Field	HALL.ANA_UX_OUT::UX_OUT	HALL.ANA_WY_OUT::WY_OUT

UX_OUT および WY_OUT の各レジスタ・フィールドは、オフセットおよびスケーリングを適用した後の ADC 値を表します。

アナログ・デコーダから求められる角度は、HALL.ANA_OUT レジスタの ANA_PHI フィールドから読み出せます。

HALL.MAP_CONFIG レジスタの ANA_HALL_MAP フィールドまたはアナログ入力反転は、ANA_PHI フィールドの角度のカウント方向を反転するために用いることができます。これを行うには 2 つの方法があり、UX_SCALE フィールドと WY_SCALE フィールドのどちらかに符号が逆のスケーリング値を設定するか、ANA_HALL_MAP に値 5 を設定します。なお、生の ADC 値 AIN_x（x＝U、W）を、表 17 に示すように、アナログ・センサーの設定値および出力値 x_OFFSET、x_SCALE、x_OUT（x＝UX、WY）に割り当てることは、ANA_HALL_MAP＝000_b のデフォルト値を用いている場合にのみ有効である点に注意してください。レジスタ・マップのフィールドの説明および図 30 の右側に、様々な ANA_HALL_MAP の割り当てオプションが記載されています。

標準的な SinCos エンコーダに使用されるアナログ信号は 2 つのみであるため、別の目的のために 3 つ目のアナログ入力ピンを使用することが可能です。例えば、HALL_V ピン（通常は AIN_V の生のアナログ入力を供給）をブレーキ・チョッパー出力として用いたり、HALL_W ピン（AIN_W の生のアナログ入力）をホーム・リファレンス・スイッチ入力や I/O コントローラ直接出力 2 として用いたりできます。このようなケースでは、UX および WY のアナログ・センサー・チャンネルが使用可能なアナログ入力ピンを用いるよう、ANA_HALL_MAP の値を調整することが必要になる場合もあります。

オプションの N パルスを SinCos エンコーダ用に設定する方法

SinCos エンコーダによっては、ヌル・ポジション信号として機能するアナログ信号またはデジタル信号を追加できます。これを使用する場合、追加の信号は VN アナログ入力に接続する必要があります。

アナログ・センサー・デコーダの N パルス機能を用いるには、HALL.ANA_CONFIG レジスタの USE_N_PULSE ビットをセットします。この機能を有効化した場合、生成された N パルスは、サイクルの終了を検出する機構として用いられます（導出されたアナログ・デコーダの角度値を用いるのとは逆）。これは、1 回転あたり 2 サイクルを発生するエンコーダにあてはまります。生成された N パルスは、アナログ・デコーダの最新の角度値（ANA_PHI）を HALL.ANA_OUT レジスタの ANA_PHI_N_CAPTURE フィールドにラッチするためのトリガ信号としても使用されます。

生成された N パルス信号の値は、HALL.ANA_CONFIG レジスタの N_PULSE_OUT ビットで読み出せます。

N パルスを生成するには、まず適切なスレッショルドおよび極性の値を設定する必要があります。これを行うには、HALL.ANA_CONFIG レジスタの N_PULSE_THRESHOLD フィールドと N_PULSE_POL フィールドを用います。N_PULSE_POL＝0_b（非反転）の場合、生成された N パルス信号は、設定されたスレッショルドより最新の VN 入力値が高いとハイに設定されます。逆に、N_PULSE_POL＝1_b（反転）の場合は、生成された N パルス信号は、設定されたスレッショルドより最新の VN 入力値が低いとハイに設定されます。なお、この比較は、生の VN アナログ入力値（ADC.AIN_V_AIN_U_RAW.AIN_V）に対し行われ、オフセットやスケーリングで補償された値ではない点に注意してください。

最後に、HALL.ANA_CONFIG レジスタの N_PULSE_EDGE ビットは、生成された N パルスのどちらのエッジを用いて、サイクルの終了を検出し、導出された角度のキャプチャをトリガするかを選択します（立上がりエッジは 0_b、立下がりエッジは 1_b）。

SPI/SSI エンコーダ

SPI/SSI エンコーダは、I/O コントローラを通じて設定およびインターフェースができます。この機能は、FEEDBACK.PHI_EXT_x（x = A、B）レジスタの外部書込み可能角度を利用します。このタイプのエンコーダを設定する方法の手順については、プログラマブル I/O コントローラのセクションおよび SPI_ENC コマンドを参照してください。

フィードバック・エンジン

TMC6460 のフィードバック・エンジンには、すべての角度フィードバック・オプションに共通のロジックが含まれています。これには、16 ビット変換、補正ルックアップ・テーブル（LUT）、外挿、phi_e 計算、速度測定、ポジション・トラッキングがあります。図 34 に、フィードバック・エンジンのブロック図を示します。

チャンネル A およびチャンネル B の 2 つのフィードバック・チャンネルが使用できます。それによって、2 つの角度フィードバック・ソース（1 つは電気的整流用、もう 1 つは速度およびポジション測定用）を同時に使用できます。あるいは、1 つの角度フィードバック・ソースを両方のチャンネルに対して用いることもできます。この場合、チャンネルの出力が電気的整流用または速度およびポジション測定用により適したものになるよう設定されます。アプリケーションが許容すれば、整流と速度およびポジション測定の両方に対し、使用可能な 2 つのチャンネルの一方のみを用いることも可能です。

2 つの 16 ビット変換モジュールが利用できます。その各々に、使用可能な 9 つのフィードバック・ソースのいずれかを利用するよう設定できます。16 ビットの変換によって、どのフィードバック・ソースを選択しても正しくスケーリングされるようにできます。このモジュールの詳細については、16 ビット変換のセクションを参照してください。

補正 LUT の 1 つのインスタンスのみが使用可能である点に注意してください。テーブル全体は、チャンネル A またはチャンネル B のどちらか一方のみに対して使用できます。あるいは、分割モードが可能な場合は、各チャンネルがテーブルの半分ずつを用いることもできます。特定のチャンネルに対して LUT がイネーブルされていない場合、そのチャンネルからの値は、変更されずに通過するだけです。補正 LUT の使用方法および設定方法の詳細については、補正ルックアップ・テーブルのセクションを参照してください。

外挿機能は、phi_e 計算用に使用できます。LUT と同様、外挿の 1 つのインスタンスが提供されます。ただし、外挿のソースとして一度に使用できるのは 1 つのチャンネルのみです。この機能の詳細については、phi 外挿機能のセクションおよび phi_e の計算のセクションを参照してください。

なお、速度計算用には、周期速度メータおよび周波数速度メータの 2 つのモジュールがある点に注意してください。この 2 つのメータは入力を共有し、その入力は、チャンネル A または B のどちらかを使用するよう設定できます。両方のメータが同時にアクティブになっている場合でも、速度コントローラへの入力として選択できる出力は 1 つのみです。この機能の詳細については、速度のアクイジションのセクションを参照してください。

ポジション測定は、チャンネル A または B のどちらか一方を用いるよう選択できます。アプリケーションにランプ・ジェネレータを用いる場合は特に、速度測定とポジション測定の両方で同じフィードバック・チャンネルを用いることを推奨します。詳細については、ポジションのアクイジションのセクションを参照してください。

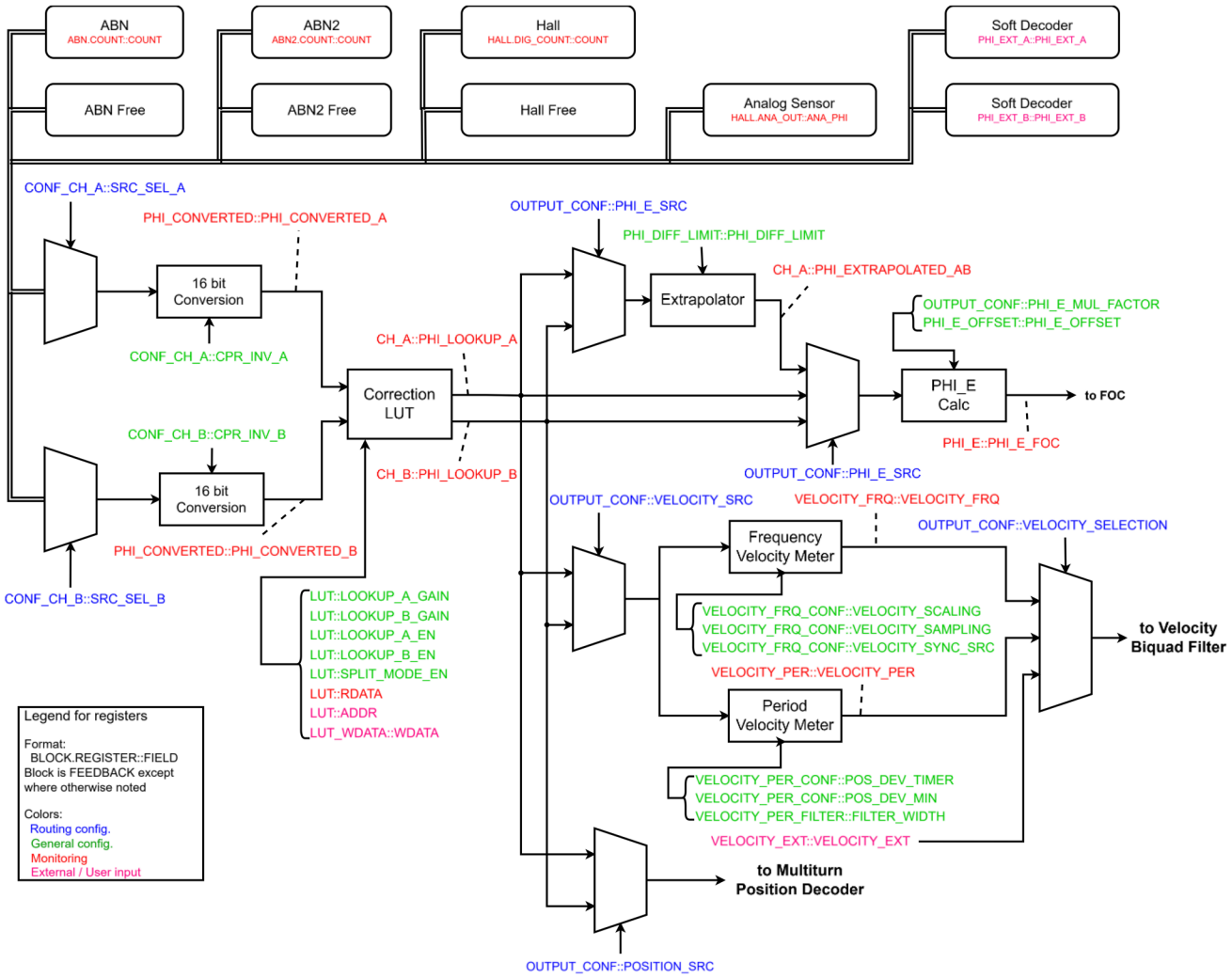


図 34. フィードバック・エンジンの概要（レジスタ・フィールドを含む）

設定プロシージャ

以下のセクションでは、フィードバック・エンジンのための設定プロシージャの概要を示します。アプリケーションで用いられるフィードバック・ソースに応じて、3 つのシナリオに分けられます。この章に含まれる各フィードバック・エンジン・モジュールについての詳細な説明を参照しながら、プロシージャに従ってください。

1 つのフィードバック・ソースを 1 チャンネルで使用

チャンネル A と B の両方が同じ機能を持つため、1 つのフィードバック・ソースを用いる場合、どちらも選択できます。以下のプロシージャ例ではチャンネル A を用います。

1 つのフィードバック・ソースをモーター用に設定するには、以下のプロシージャを使用します。

1. FEEDBACK.CONF_CH_A レジスタの SRC_SEL_A フィールドに必要な値を設定して、チャンネル A のフィードバック・ソースを選択します。
2. チャンネル A の 16 ビット変換モジュールを設定します。これを行うには、FEEDBACK.CONF_CH_A レジスタの CPR_INV_A フィールドを 2^{24}/CPR に設定します。ここで、CPR は、フィードバックの 1 回転あたりにインクリメントする値の数を表します。詳細については以降の各セクションを参照してください。
3. 補正 LUT を設定します。これはオプションのステップであり、LUT が不要な場合は、スキップできます。FEEDBACK.LUT レジスタの LOOKUP_EN_A ビットをセットし、同じレジスタの SPLIT_MODE_EN ビットをクリアします。LUT の項目には、前もって補正データを入力しておく必要があります。詳細については以降の各セクションを参照してください。

4. phi 外挿機能を設定します。このステップもオプションですが、デジタル・ホール・センサーのフィードバックにはほとんどの場合必要です。これを行うには、FEEDBACK.PHI_DIFF_LIMIT レジスタに適切な値を設定します。詳細については以降の各セクションを参照してください。
5. フィードバック・エンジンの出力を設定します。これは、FEEDBACK.OUTPUT_CONF レジスタの PHI_E_SRC、VELOCITY_SRC、POSITION_SRC の各フィールドを通じて行います。
   - PHI_E_SRC の場合：phi 外挿機能を用いるかどうかに応じて、LOOKUP_A または EXTRAPOLATOR_A を選択します。
   - VELOCITY_SRC の場合：LOOKUP_A を選択します。
   - POSITION_SRC の場合：LOOKUP_A を選択します。

補正 LUT がディスエーブルの場合、角度データは、変更されないまま通過します。

この後、phi_e の計算およびアライメントと、速度およびポジションのアクイジション・ステップに続きます。これはここに示すすべてのプロシージャ例に共通です。

1 つのフィードバック・ソースを 2 チャンネルで使用

ABN エンコーダまたはホール・センサーが単一のフィードバック・ソースとして用いられている場合に、チャンネル A と B の両方を使用できます。この場合、一方のチャンネルがモーター整流用に設定され、他方が速度およびポジションの測定用に設定されます。両方のチャンネルに同じ機能が備わっているため、どちらを整流用に用いてどちらを速度およびポジション・トラッキング用に用いるかは任意です。以下のプロシージャ例ではチャンネル A を整流用に用います。

ABN エンコーダまたはホール・センサーを単一のフィードバック・ソースに設定するには、以下の手順を使用します。

1. FEEDBACK.CONF_CH_A レジスタの SRC_SEL_A フィールドに必要な値を設定して、チャンネル A のフィードバック・ソースを選択します。ABN_1、ABN_2、または HALL のいずれかのオプションを選択します。
2. チャンネル A の 16 ビット変換モジュールを設定します。これを行うには、FEEDBACK.CONF_CH_A レジスタの CPR_INV_A フィールドを 224/CPR に設定します。ここで、CPR は、フィードバックの 1 回転あたりにインクリメントする値の数を表します。詳細については以降の各セクションを参照してください。
3. FEEDBACK.CONF_CH_B レジスタの SRC_SEL_B フィールドに必要な値を設定して、チャンネル B のフィードバック・ソースを選択します。ABN_1_FREE、ABN_2_FREE、または HALL_FREE のいずれかのオプションを選択します。これらのオプションは、特に速度およびポジション・トラッキングに適しています。
4. チャンネル B の 16 ビット変換モジュールを設定します。これを行うには、FEEDBACK.CONF_CH_B レジスタの CPR_INV_B フィールドを設定します。ABN エンコーダのフィードバックの場合は 256 の値、ホール・センサーのフィードバックの場合は 4096 の値を用いることを推奨します。それにより 16 ビット変換を「バイパス」でき、スケーリングをフィードバックの真の分解能に近付けることができます。詳細については以降の各セクションを参照してください。
5. 補正 LUT を設定します。これはオプションのステップであり、LUT が不要な場合は、スキップできます。一般には、ABN エンコーダやホール・センサーのフィードバックに補正 LUT は不要です。
6. phi 外挿機能を設定します。このステップもオプションですが、デジタル・ホール・センサーのフィードバックにはほとんどの場合必要です。これを行うには、FEEDBACK.PHI_DIFF_LIMIT レジスタに適切な値を設定します。詳細については以降の各セクションを参照してください。
7. フィードバック・エンジンの出力を設定します。これは、FEEDBACK.OUTPUT_CONF レジスタの PHI_E_SRC、VELOCITY_SRC、POSITION_SRC の各フィールドを通じて行います。
   - PHI_E_SRC の場合：phi 外挿機能を用いるかどうかに応じて、LOOKUP_A または EXTRAPOLATOR_A を選択します。
   - VELOCITY_SRC の場合：LOOKUP_B を選択します。ここには速度トラッキングに適したフィードバック角度が含まれています。
   - POSITION_SRC の場合：LOOKUP_B を選択します。ここにはポジション・トラッキングに適したフィードバック角度が含まれています。

補正 LUT がディスエーブルの場合、角度データは、変更されないまま通過します。

この後、phi_e の計算およびアライメントと、速度およびポジションのアクイジション・ステップに続きます。これはここに示すすべてのプロシージャ例に共通です。

デュアル・フィードバック・ソースで使用

アプリケーションにおいて 2 つのフィードバック・ソースが使用可能な場合、チャンネル A とチャンネル B の両方を用いる必要があります。この場合、一方のチャンネルがモーター整流用に設定され、他方が速度およびポジションの測定用に設定されます。両方のチャンネルに同じ機能が備わっているため、どちらを整流用に用いてどちらを速度およびポジション・トラッキング用に用いるかは任意です。以下のプロシージャ例ではチャンネル A を整流用に用います。

デュアル・フィードバック・ソースの設定は次の手順で行います。

1. FEEDBACK.CONF_CH_A レジスタの SRC_SEL_A フィールドに必要な値を設定して、チャンネル A のフィードバック・ソースを選択します。
   - ABN エンコーダまたはホール・センサーをこのチャンネルのフィードバック・ソースとして用いる場合、対応する ABN_1、ABN_2、または HALL のオプションを選択します。
   - 別のフィードバック・タイプを用いる場合には、それに対応するオプションを選択します。
2. チャンネル A の 16 ビット変換モジュールを設定します。これを行うには、FEEDBACK.CONF_CH_A レジスタの CPR_INV_A フィールドを 2^{24}/CPR に設定します。ここで、CPR は、フィードバックの 1 回転あたりにインクリメントする値の数を表します。詳細については以降の各セクションを参照してください。
3. FEEDBACK.CONF_CH_B レジスタの SRC_SEL_B フィールドに必要な値を設定して、チャンネル B のフィードバック・ソースを選択します。
   - ABN エンコーダまたはホール・センサーをこのチャンネルのフィードバック・ソースとして用いる場合、対応する ABN_1_FREE、ABN_2_FREE、または HALL_FREE のオプションを選択します。
   - 別のフィードバック・タイプを用いる場合には、それに対応するオプションを選択します。
4. チャンネル B の 16 ビット変換モジュールを設定します。これを行うには、FEEDBACK.CONF_CH_B レジスタの CPR_INV_B フィールドを設定します。詳細については以降の各セクションを参照してください。
   - ABN エンコーダをこのチャンネルのフィードバック・ソースとして用いる場合、256 の値を使用することを推奨します。
   - ホール・センサーをフィードバック・ソースとして用いる場合は、4096 の値を使用することを推奨します。
   - 別のフィードバック・タイプを用いる場合、2^{24}/CPR の値を設定します。ここで、CPR は、フィードバックの 1 回転あたりにインクリメントする値の数を表します。
5. 補正 LUT を設定します。これはオプションのステップであり、LUT が不要な場合は、スキップできます。LUT の項目には、前もって補正データを入力しておく必要があります。詳細については以降の各セクションを参照してください。
   - 1 つのチャンネルを使用する場合に LUT をイネーブルする場合は、FEEDBACK.LUT レジスタの SPLIT_MODE_EN ビットをクリアし、LOOKUP_EN_x ビット（x = A または B）を補正する必要のあるチャンネルに応じて設定し、更に、その他のチャンネルの LOOKUP_EN_x ビットをクリアします。
   - 両チャンネルに対し LUT をイネーブルする場合は、FEEDBACK.LUT レジスタの SPLIT_MODE_EN ビットをセットし、LOOKUP_EN_A および LOOKUP_EN_A の両方のビットをセットします。
6. phi 外挿機能を設定します。このステップもオプションですが、デジタル・ホール・センサーのフィードバックにはほとんどの場合必要です。これを行うには、FEEDBACK.PHI_DIFF_LIMIT レジスタに適切な値を設定します。詳細については以降の各セクションを参照してください。
7. フィードバック・エンジンの出力を設定します。これは、FEEDBACK.OUTPUT_CONF レジスタの PHI_E_SRC、VELOCITY_SRC、POSITION_SRC の各フィールドを通じて行います。
   - PHI_E_SRC の場合：phi 外挿機能を用いるかどうかに応じて、LOOKUP_A または EXTRAPOLATOR_A を選択します。
   - VELOCITY_SRC の場合：LOOKUP_B を選択します。ここには速度トラッキングに適したフィードバック角度が含まれています。
   - POSITION_SRC の場合：LOOKUP_B を選択します。ここにはポジション・トラッキングに適したフィードバック角度が含まれています。

補正 LUT がディスエーブルの場合、角度データは、変更されないまま通過します。

この後、phi_e の計算およびアライメントと、速度およびポジションのアクイジション・ステップに続きます。これはここに示すすべてのプロシージャ例に共通です。

16 ビット変換

16 ビット変換モジュールは、様々なモーター角度フィードバック・ソースをサポートしているため、どのフィードバック・ソースを選択しても適切なスケーリングが確実に用いられるようにする手段としての役割を果たしています。その目的を達するために、すべてのソースが、このモジュールで処理されるときに、16 ビットの分解能までスケール・アップされます。

チャンネル A 用とチャンネル B 用の 2 つの変換モジュールが使用できます。これらは、それぞれ、FEEDBACK.CONF_CH_A レジスタおよび FEEDBACK.CONF_CH_B レジスタを用いて設定できます。SRC_SEL_x（x = A、B）のフィールドは、対応するチャンネル用に用いられるフィードバック・ソースを選択します。CPR_INV_x（x = A、B）のフィールドは、変換動作の出力を定めます。

16 ビットのスケーリングは、フィードバック・ソースの 1 回転（機械的か電気的かは不問）が 0～65535 の値に割り当てられるように行われます。変換モジュールの出力は次式で計算されます。

$$phi_{CONVERTED} = \frac{phi_{SOURCE} \times CPR_INV_x}{256}$$

そのため、16 ビットのスケーリングを実現するには、cpr_inv の適切な値を次式に従って計算する必要があります。

$$CPR_INV_x = \frac{2^{24}}{counts_per_rev}$$

例えば、SRC_SEL_x フィールドでの選択に応じて、以下の counts_per_rev の値を使用します。

- ABN_1、ABN_2：使用するエンコーダの CPR に等しい counts_per_rev を用います。例えば、
  - 1024 ラインのエンコーダの場合、CPR 値は 1024 × 4 = 4096 です。そのため、CPR_INV_x = 2^{24}/4096 = 4096 となります。
  - 10000 ラインのエンコーダの場合、CPR 値は 40000 です。そのため、CPR_INV_x = 2^{24}/40000 = 419 となります。
- ANALOG_SENSOR：2^{16} = 65536 に等しい counts_per_rev を用います。これが ANA_PHI フィールドのフル・スケール範囲であるためです。そのため、常に、CPR_INV_x = 2^{24}/2^{16} = 256 を使用します。
- HALL：6 に等しい counts_per_rev を用います。ホール・デコーダの COUNT フィールドは常に 0～5 の範囲で巡回するためです。そのため、常に CPR_INV_x = 2^{24}/6 = 2796202 を使用します。
- PHI_EXT_A、PHI_EXT_B：この場合、counts_per_rev は、外部から書き込まれた角度値のフル・スケール範囲によって決まります。この角度値は、マイクロコントローラまたは内蔵の I/O コントローラから供給されます。例えば、
  - 14 ビットの分解能の SPI エンコーダの場合、2^{14} に等しい counts_per_rev を用います。そのため、CPR_INV_x = 2^{24}/2^{14} = 1024 となります。

FREE オプションと呼ばれる一部の特殊な選択では、特別な配慮が必要です。これらのオプションの目的は、16 ビット変換を「バイパス」することであり、それによって、変換された値をフィードバックの真の分解能とスケーリングに近付けることができます。CPR_INV_x は、28 以上の 2 のべき乗（256、512、1024、2048 など）に設定する必要があります。2 の指数が大きいほど、その結果生じる速度とポジションの値は大きくなります。256 という値に設定すると、フィードバックの元のスケーリングになります。これは、現実の単位への変換のセクションで説明するように、現実の単位への変換にも影響します。

- ABN_1_FREE、ABN_2_FREE：256 という値の CPR_INV_x を用いることを推奨します。それによって、エンコーダの元の分解能とスケーリングを再現できます。
- HALL_FREE：4096 という値の CPR_INV_x を用いることを推奨します。これより小さい 2 の指数も有効ですが、推奨しません。変換された値が小さくなりすぎるためです。これは、ランプ・ジェネレータを使用する場合に問題になる可能性があります。

16 ビット変換の出力は補正 LUT の入力です。変換された値は、FEEDBACK.PHI_CONVERTED レジスタの PHI_CONVERTED_A フィールドおよび PHI_CONVERTED_B フィールドで読み出せます。

補正ルックアップ・テーブル

TMC6460 の補正ルックアップ・テーブル（LUT）は、例えば、モーター・シャフトに取り付けられた回転磁石を基準とする絶対磁気エンコーダのミスアライメントによって生じる誤差など、連続的かつ巡回的な角度のフィードバック誤差または歪みを補正する機能を備えた、256 項目からなる補間テーブルです。

LUT は、各フィードバック・チャンネルに対し、FEEDBACK.LUT レジスタの LOOKUP_A_EN ビットまたは LOOKUP_B_EN ビットのどちらかをセットすることにより、独立にイネーブルできます。LUT は、両方のフィードバック・チャンネルを同時に操作できる、スプリット・モードも備えています。スプリット・モードが有効な場合、256 項目のテーブルは、2 つの 128 項目のテーブルに分割されます。最初の 128 項目はフィードバック・チャンネル A 用に用いられ、残りの 128 項目はチャンネル B 用に用いられます。LUT のスプリット・モードは、FEEDBACK.LUT レジスタの SPLIT_MODE ビットをセットすることにより有効化できます。両方の LUT チャンネルがイネーブルされている場合は、スプリット・モードを有効化しなくてはなりません。ただし、スプリット・モードが有効な場合でも 1 つのチャンネルのみをイネーブルすることはできます。この場合、イネーブルされたチャンネルに対応する 128 項目のみが用いられます。

LUT の各項目は、−128～127 の範囲の 8 ビット値で構成されます。ただし、LUT には、1、2、4、8、16 に設定可能なゲイン係数があり、補正範囲を拡張するのに有用です。このように、LUT は、最大で-2048～2032 の範囲で角度誤差値を補正できます。FEEDBACK.LUT レジスタの LOOKUP_A_GAIN フィールドおよび LOOKUP_B_GAIN フィールドを通じて、各 LUT チャンネルに対し独立にゲイン係数を設定できます。

LUT の出力は、外挿モジュール、phi_e 計算モジュール、速度メータ、マルチターン・ポジション・デコーダの入力になります。LUT チャンネルの出力は、FEEDBACK.CH_A レジスタの PHI_LOOKUP_A フィールドおよび、FEEDBACK.CH_B レジスタの PHI_LOOKUP_B フィールドから読み出せます。

エンコーダ誤差の特性評価

補正 LUT を正しく用いるには、まずエンコーダ誤差の特性評価を行わなくてはなりません。それを行うには、リファレンスとして第二の角度ソースを用いる必要があります。リファレンス・ソースは、モーター・シャフトの実際の角度に正確に従う角度を提供する必要があります。リファレンス角度ソースとしては、高分解能 ABN エンコーダが良い選択肢です。外部からの第二の角度フィードバックが利用できない場合は、その代わりとしてランプ・ジェネレータからの phi_e を用いることもできます。

モーターが回転している間、使用しているフィードバックおよびリファレンスの 2 つのソースからの角度値を、同時に追跡する必要があります。そのためには、オープン・ループ・モーションのモードのセクションで説明するように、モーターをオープン・ループ・モードで回転する必要があります。モーターが角度フィードバック・ソースの 1 回転を完了する間に、256 以上のサンプルを等間隔に取得しなくてはなりません。LUT のスプリット・モードを用いる場合には、この最低サンプル数を 128 に減らせます。取得データのノイズをフィルタ処理するためにより多数のサンプルを取得できますが、フィードバック・ソースの 1 回転あたり 256（スプリット・モードを用いる場合は 128）の倍数のサンプルを取得するのが理想的です。以下に 2 つの例を示します。

- 絶対エンコーダを用い、かつ、第二のフィードバック・ソースが利用できない場合、ランプ・ジェネレータによる内部 phi_e をリファレンスとして使用できます。この場合、データ・サンプルは、FEEDBACK.PHI_CONVERTED レジスタの PHI_CONVERTED_A フィールドおよび RAMP.PHI_E レジスタの PHI_E フィールドから取得できます。これは、使用するフィードバック・ソースがフィードバック・チャンネル A で正しく設定されていることが前提です。ただし、内部 phi_e を有効なリファレンス角度として用いるには、モーターのコギング・トルクの影響を考慮することが必須です。それを補償するには、モーター回転時にモーター・シャフトが内部 phi_e に円滑かつ緊密に従うよう、モーターが回転しデータ・サンプリングを行っている間には十分に大きなオープン・ループ電流を用いる必要があります。更に、2 つの角度値が同程度の大きさとなるよう、内部 phi_e の取得データを調整する必要があります。つまり、この 2 つの値は同じスケーリングでなくてはなりません。それは、phi_e は電気的な 1 回転の間に 1 サイクル（0～65535）を完了するのに対し、絶対エンコーダは機械的な 1 回転後にのみ同じサイクルを完了するためです。
- 絶対エンコーダを用い、かつ、ABN エンコーダが利用できる場合、ABN エンコーダによる角度をリファレンスとして使用できます。この場合、モーターが回転している間のデータ・サンプルは、FEEDBACK.PHI_CONVERTED レジスタの PHI_CONVERTED_A フィールドおよび PHI_CONVERTED_B フィールドから取得できます。これは、そのようなフィードバック・ソースが、フィードバック・チャンネル A および B で正しく設定されていることが前提です。この場合、ABN エンコーダからの角度は既にモーター・シャフトの角度に従っているため、コギング・トルクの影響を考慮する必要はありません。そのため、オープン・ループ電流は前述の例ほど大きくする必要がありません。更に、ABN 用に取得したデータは、調整の必要がありません。この角度と絶対エンコーダの角度は既に同程度の大きさであるためです。

その後、エンコーダ誤差関数を、使用するフィードバックで報告された角度とリファレンス・ソースから報告された角度の差として取得できます。取得した誤差に由来する「DC オフセット」は、例えば、すべての誤差データ・ポイントの平均値を計算し、各エラー・ポイントからその値を差し引くことなどの方法によって、除去しなくてはなりません。

より多くのサンプル数が使用できる場合、移動平均フィルタなどの円滑化機能を誤差関数に適用できます。その結果円滑化された誤差関数は、LUT に書き込めるようになるには、256（スプリット・モードを用いる場合は 128）データ・ポイントまでサンプリングする必要があります。

得られた誤差データ・ポイントが-128～127 の LUT 項目の有効範囲を超える場合には、LUT に入力する前に、更なる変更が必要です。誤差データ・ポイントの最大絶対値に応じ、データを利用可能な LUT ゲイン係数に対応して 2、4、8、または 16 で除算する必要があります。例えば、得られたエラー関数の最小値が−180 で最大値が 190 であれば、誤差関数の値を 2 分の 1 にする必要があり、誤差関数の範囲が−510～500 の範囲であれば、誤差関数の値を 4 分の 1 にする必要があります。

動作原理

LUT の項目は、そのアドレスで特定され、アドレスは 0～255 の値で表されます。テーブルの各項目は 8 ビットの値であり、-128～127 の範囲の符号付き整数として解釈されます。上述のように、エンコーダ誤差関数$fERR\ (phi)$は、フィードバック・ソースから報告される角度$phi_{FB}(phi)$と実際の機械的角度（またはリファレンスとして取得された角度）phiの差を表し、次式で計算されます。

$$f_{ERR}(phi) = phi_{FB}(phi) - phi$$

与えられた 16 ビットの入力角度phi_{FB}に対し、2 つの LUT 項目が読み出されます。最初の項目のアドレスは、phi_{FB}の上位 8 ビットで定義され、後続のアドレスは、2 つ目の項目として取得されます。つまり、$addr0 = phi_{FB} \gg 8$、および$addr1 = (phi_{FB} \gg 8) + 1$ です。

$addr0 = 256$ の場合は、$addr1$ は自動的に 0 に設定されます。

$lut(addr0)$と$lut(addr1)$の 2 つの LUT 項目の値の重み付け平均が計算されます。その場合、角度phi_{FB}の下位 8 ビットを使用して、各値の重みが決定されます。この量は、補間結果$res_interpolation$と呼ばれ、次式で表されます。

$$res_interpolation = lut(addr0) \times \frac{phi_{FB}\ \%\ 256}{256} + lut(addr1) \times \frac{256 - (phi_{FB}\ \%\ 256)}{256};\ \% - modulo\ operator$$

例えば、$phi_{FB} = 20000 = 0x4E20$ の場合、アドレスは、$addr0 = 0x4E$ および$addr0 = 0x4F$ となります。

LUT の項目の値がlut(0x4E) = 10 およびlut(0x4F) = −3 であれば、補間結果は次式のように計算されます。

$$res_interpolation = 10 \times \frac{20000\ \%\ 256}{256} - 3 \times \frac{256 - (20000\ \%\ 256)}{256} = 1.25 - 2.625 = -1.375$$

その後、この補間結果が設定済みのゲイン係数（FEEDBACK.LUT レジスタの LOOKUP_GAIN_A フィールドまたは LOOKUP_GAIN_B フィールド）によってスケーリングされ、それによって、補正値$res_correction$が求まります。この補正係数が入力角度に付加され、補正された出力角度が得られます。図 35 に、この全体のプロシージャを示します。

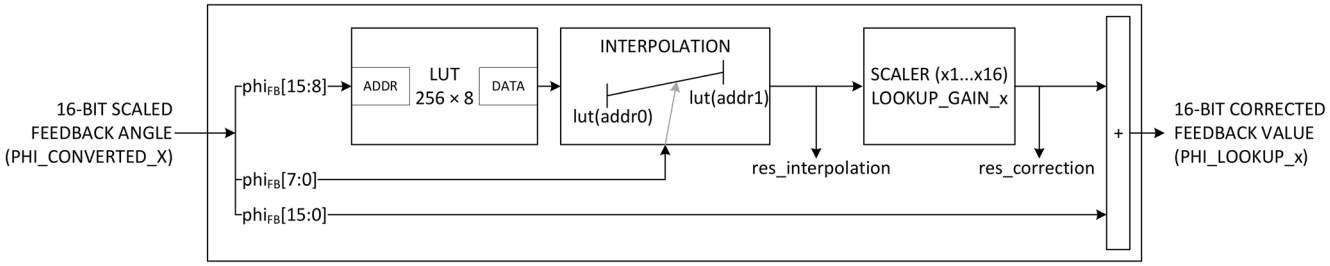


図 35. 補正 LUT の機能図

LUT 項目の読出しおよび書込み

LUT の項目の値を読み出すには、まず、FEEDBACK.LUT レジスタの ADDR フィールドに目的の項目のアドレスを書き込みます。次に、FEEDBACK.LUT レジスタの RDATA フィールドを読み出して、所定アドレスの LUT 値を取得します。

LUT の書込みは、一度に 4 項目のグループでのみ行うことができます。LUT の 4 項目を書き込むには、FEEDBACK.LUT レジスタの ADDR フィールド（読出しと同じアドレス・フィールド）に 4 項目の最初のアドレスを書き込みます。LUT に入力する場合、渡されるアドレスは 4 の倍数、つまり、0、4、8、12 等でなくてはなりません。その後、4 つの LUT 項目のデータを含む 32 ビットの値を FEEDBACK.LUT_WDATA レジスタに書き込みます。LUT は、WDATA[7:0]ビットを ADDR フィールドで指定されるテーブル項目、WDATA[15:8]ビットを ADDR+1 のテーブル項目、WDATA[23:16]ビットを ADDR+2 のテーブル項目、WDATA[31:24]ビットを ADDR+3 のテーブル項目に、自動的に書き込みます。その後、ADDR フィールドの値を 4 だけ増加し、FEEDBACK.LUT_WDATA レジスタに次の 4 つの LUT 項目の値を書き込みます。最後のアドレス 252 を使用した後、テーブル全体が入力されるまでこの処理を繰り返します。

phi 外挿機能

phi 外挿機能モジュールは、角度フィードバック更新の間の時間とそれに対応するステップ・サイズを使用し、次の更新時間および値を予測します。外挿機能はこの予測値を用いて、中間の多数の角度更新を行います。それにより、フィードバック角度のステップ・サイズが効果的に低減されます。モジュールは、デジタル・ホール・センサーなどの低分解能フィードバックで使用することを目的としています。ただし、どのフィードバック・タイプで使用しても構いません。

phi 外挿機能は、FEEDBACK.OUTPUT_CONF レジスタの PHI_E_SRC フィールドにおいて、EXTRAPOLATOR_A（10_b）または EXTRAPOLATOR_B（11_b）のオプションを選択することにより、有効化されます。外挿機能モジュールに対しソース・チャンネル（チャンネル A またはチャンネル B）を手動で選択することはできません。代わりに、選択は、PHI_E_SRC フィールドで設定されたオプションに基づき、自動的に行われます。

外挿機能の予測は、実際のフィードバック角度に先立って角度を発生します。この外挿機能が予測する最大距離は、FEEDBACK.PHI_DIFF_LIMIT レジスタの PHI_DIFF_LIMIT フィールドによって設定されます。このフィールドは 16 ビットの角度差を保持します。フィードバック・システムの角度分解能より大きな値を選択できますが、必ず 4 分の 1 の分解能（16384）よりは小さくなくてはなりません。例えば、ホール・センサーの場合、最小の PHI_DIFF_LIMIT は $2^{16}/6 = 10923$ です。PHI_DIFF_LIMIT の値を大きくすると、定常状態の速度での動作が滑らかになりますが、速度が低くなって外挿機能が自動的に無効化されるとジッタが増加します。

なお、ホール・センサーを用いるアプリケーションの場合、PHI_DIFF_LIMIT の値は通常、電気的回転の 65º、つまり、$2^{16} \times 65º/360º = 11833$ の値に設定されます。これは、ホール・センサーが磁界とアライメントできるよう、ホール区画の 60º に加え一定のマージンをカバーしています。

phi 外挿機能が正常に働くためには、最小限の更新レートが必要です。この最小限の更新レートが満たされない場合、外挿機能は自動的に無効化され、フィードバック値は変更されないまま通過します。外挿機能の最小限の更新レートは次式で定められます。

$$f_{EXTRAPOLATION,min} = \frac{f_{SYS}}{65536 \times 16} = 57.22\ Hz \quad with\ f_{SYS} = 60MHz$$

これを拡張し、ホール・センサーを使用している場合に発生する外挿機能の最低速度は、次のように計算できます。

$$v_{EXTRAPOLATION,min} = \frac{f_{SYS} \times 60}{65536 \times 16 \times 6 \times N_{polepairs}} [rpm] \quad with\ f_{SYS} = 60MHz$$

外挿された角度は、常に、FEEDBACK.CH_A レジスタの PHI_EXTRAPOLATED_AB フィールドから読み出せます。その元のレジスタ FEEDBACK.CH_A に関わらず、チャンネル B が整流用に選択されている場合、このビットフィールドは、チャンネル B の外挿角度を示すこともできます。

Phi E の計算

phi_e 計算モジュールは、機械的な角度を電気的な角度に変換するために用いられる他、フィードバックの角度を FOC に用いられる内部 phi_e とアライメントするために用いることもでき、そのため電流制御にも使用できます。

この phi_e 計算モジュールは、その入力を LUT 出力または外挿機能出力から取得します。これは、FEEDBACK.OUTPUT_CONF レジスタの PHI_E_SRC フィールドを通じて設定されます。phi_e 計算モジュールの出力は、FEEDBACK.PHI_E レジスタの PHI_E_FOC フィールドから読み出せます。

更に、このモジュールは、FEEDBACK.OUTPUT_CONF レジスタの PHI_E_MUL_FACTOR フィールドを通じて設定されます。フィードバック角度の 1 回転あたりのモーターの電気的回転量に等しい値を書き込みます。例えば ABN エンコーダや SinCos エンコーダなどの、機械的角度を生成するフィードバック・ソースの場合、モーターの極ペアの数を PHI_E_MUL_FACTOR フィールドに書き込みます。デジタル・ホール・センサーやアナログ・ホール・センサーなどの、電気的角度を直接測定するフィードバック・ソースの場合は、PHI_E_MUL_FACTOR フィールドに 1 の値を書き込みます。

Phi E のアライメント

フィードバックの計算した電気的角度が内部 phi_e と完全には一致しないことも、たびたびあり得ます。このアライメントは、正しい FOC を実現するために重要であり、2 つの角度を揃えるための適切なプロシージャを実行する必要があります。このプロシージャは、使用されるフィードバック・ソースに応じて選択する必要があります。いくつかの異なるケースを区別できます。

- インクリメンタル ABN エンコーダ：
  - ABN エンコーダの角度は、「初期化」によって内部 phi_e の角度に揃えることができます。
  - ABN エンコーダを初期化するには、次の手順に従います。
    - オープン・ループ・ポジション・モードで動作するよう IC を設定します。詳細については、オープン・ループ・モーションのモードのセクションを参照してください。
    - 必ず、FOC.PID_POSITION_TARGET をゼロにセットします。
    - MOTION_MODE が VOLTAGE_EXT と TORQUE のどちらに設定されているかに応じて、EXT_CTRL.VOLTAGE レジスタの UD フィールドに電圧値を設定するか、あるいは、PID_TORQUE_FLUX_TARGET レジスタの PID_FLUX_TARGET フィールドに目標磁束値を設定します。
    - ローターは、電気的角度 0º に見合う特定のポジションに位置する必要があります。電圧／磁束の値が大きいほど、アライメントの精度は高まりますが、より多くの電流がモーターのコイルに流されます。
    - 電圧／磁束の値は、最初から最終値を設定する代わりに、手動で小さなインクリメント量を用いてランプ・アップすることもできます。それによって、過電流状態が誘導される可能性を低減できます。
    - モーターが確実に停止状態になるよう、短時間待機します。モーターの特性や使用する電圧／磁束の値に応じて、0.5 秒～1 秒の時間で十分です。
    - 初期化を可能な限り最短の時間で行う必要がある場合は、電圧／磁束の値を設定した後に、ABN.COUNT レジスタの COUNT フィールドをモニタできます。モニタ値が変化しなくなったら、つまり、モーターが停止状態になったら、直ちにこのプロシージャを続行できます。
    - ABN.COUNT レジスタの COUNT フィールドの値をゼロにセットします。
  - これで、PHI_E フィールドと PHI_E_FOC フィールドは、モーターの回転時に非常に近い値を示すはずです。また、ABN エンコーダの phi_e 角度はクローズド・ループ動作に使用できます。
  - このプロシージャは、TMC6460 とエンコーダがアクティブな状態になっているときにのみ、実行する必要があります。IC の電源再投入またはリセット後、あるいは、エンコーダ信号または物理的接続が破損した後には、このプロシージャを実行する必要があります。
  - なお、このプロシージャは絶対に確実というわけではない点に注意してください。例えば、高負荷状態のモーターは、初期ポジションが電気的角度の 180º に近い場合、0º の目標角度には揃いません。改善された初期化方法は、ここでは詳述しません。ただしいくつかの例を示します。
    - 0º の目標で終了する前に、目標角度を動的に振動させたり、スイングしたりします。
    - 初期目標角度を 90º に設定し、停止状態になるのを待ってから 0º の最終目標を設定します。

- デジタル・ホール・センサー – phi 外挿を使用：
  - ホール・センサーの角度は、オフセット値を用いて内部 phi_e の角度に揃えることができます。
  - オフセット値は、FEEDBACK.PHI_E_OFFSET レジスタの PHI_E_OFFSET フィールドを通じて設定できます。
  - オフセット値を求めるには以下の手順を使用します。
    - モーターをオープン・ループ・モードで回転させながら（オープン・ループ・モーションのモードのセクションを参照）、RAMPER.PHI_E レジスタの PHI_E フィールド（オープン・ループの角度）と FEEDBACK.PHI_E レジスタの PHI_E_FOC フィールド（フィードバックの角度）の両方を取得します。データの取得中、モーターは phi 外挿が行えるだけの十分な高速度で回転する必要があります（最低速度の詳細については、phi 外挿機能のセクションを参照）。
    - 取得されたデータから 2 つの角度の差の平均値を計算します。
    - 計算した平均値を PHI_E_OFFSET フィールドのオフセット値として使用します。
    - より良い結果を得るために、モーターを反対方向に回転させながらこのプロセスを繰り返し、両方向での動作のデータを用いて平均値を計算する必要が生じる場合もあります。
  - これで、PHI_E フィールドと PHI_E_FOC フィールドは、モーターの回転時に非常に近い値を示すはずです。また、その結果生じるホール・センサーの phi_e 角度はクローズド・ループ動作に使用できます。
  - ホール・センサーは、絶対（電気的）角度を示すので、この特定のモーター・センサー・セットアップの場合には、計算されたオフセット値は変化しません。つまり、上記の手順を実行する必要があるのは 1 回のみで、その結果得られるオフセット値は、IC のリセット後または電源再投入後、毎回使用できます。
- デジタル・ホール・センサー – phi 外挿を不使用：
  - ホール・センサーの角度は、デジタル・ホール・センサーのセクションで説明したように、デジタル・ホール・デコーダ区画のオフセットを用いて、内部 phi_e の角度に揃えることができます。
  - 区画のオフセット設定を求めるには、以下の手順を使用します。
    - モーターをオープン・ループ・モードで低速で回転させながら（オープン・ループ・モーションのモードのセクションを参照）、RAMPER.PHI_E レジスタの PHI_E フィールド（オープン・ループの角度）と FEEDBACK.PHI_E レジスタの PHI_E_FOC フィールド（ホール・センサーの角度）の両方を取得します。
    - 取得されたデータから 2 つの角度の差の平均値を計算します。
    - 計算された平均値から、$average_diff/65535 \times 360^{o}$ の式を用いて、それと等価な角度（º）単位の値を計算します。
    - 表 14 をリファレンスとして用い、計算された平均差に最も近い値の区画オフセットに従って、割り当ておよびピン入力反転値を設定します。
  - これで、PHI_E_FOC から求まる角度区画はモーター回転時のオープン・ループ角度に従うはずです。また、その結果生じるホール・センサーの phi_e 角度はクローズド・ループ動作に使用できます。
  - ホール・センサーは、絶対（電気的）角度を示すので、この特定のモーター・センサー・セットアップの場合には、取得されたホール区画オフセット値は変化しません。つまり、上記の手順を実行する必要があるのは一回のみで、その結果得られる割り当ておよびピン入力反転設定値は、IC のリセット後または電源再投入後、毎回使用できます。
- アナログ・ホール・センサー、SinCos エンコーダ、絶対 SPI/SSI エンコーダ（（電気的角度のみであっても）絶対角度を生成するフィードバック・ソース）：
  - フィードバックの電気的角度は、FEEDBACK.PHI_E_OFFSET レジスタの PHI_E_OFFSET フィールドを用いて、内部 phi_e の角度に揃えることができます。
  - オフセット値を求めるために phi 外挿機能を用いる場合のデジタル・ホール・センサーと同じ手順を使用します。
  - これらのフィードバック・ソースでは一般的に phi 外挿機能が不要であるため、phi_e データの取得時に上記の場合のような高速でモーターを回転する必要はありません。
  - オフセット値を求める手順を実行する必要があるのは一回のみで、その結果得られるオフセット値は、IC のリセット後または電源再投入後、毎回使用できます。

Phi E の丸め誤差補正

phi_e 計算のソースとして 2 の指数でない分解能のフィードバック・ソースを用いる場合、16 ビット変換モジュールの CPR_INV_x に対する式からは、整数以外の値が生じます。そのような場合、CPR_INV_x フィールドの実際の値を、計算値の床関数値に設定する必要があります。それによって、フィードバックの各ステップは本来の値より小さくなります。ステップが小さくなることにより積算誤差が生じます。これを補償するために、回転の最後のステップが、それまでのすべてのステップより大きくなります。

補正 LUT は、この誤差を完全に除去するよう設定できます。不連続な誤差関数を補正するよう LUT を正しく使用するための重要ポイントは、テーブルの最初と最後の項目の間での補間を避けることです。これを行うためには以下の手順に従います。

1. CPR_INV を減少させて、フィードバック角度が、0～0xFFFF の範囲でなく 0～0xFEFF の範囲にスケーリングされるようにします（$CPR_INV = (2^{24} - 2^{16})/CPR$ の式を使用）。

2. 調整された CPR_INV 値を用いて、最初のステップ（0～1）と最後のステップ（CPR − 1～0）の間の差（16 ビット変換後）を計算します。
3. 最後のステップのサイズから最初のステップのサイズを差し引きます。それにより得られた量を「補正係数」と呼びます。
4. この補正係数は、127（LUT 項目の最大値）を超える場合もあります。そのような場合、補正係数を 127 以下の数に除算できる最小の数となるようスケーリング・ファクタを選択する必要があります。
5. 選択したスケーリング・ファクタで LUT を設定し、すべての項目（最後の項目を含む）を、ゼロから始まりスケーリング・ファクタで除算した補正係数で終了する、直線的な増加率でプログラムします。このようにして、補正 LUT は、16 ビットのスケーリングによって生じる誤差を完全に除去できます。

丸め誤差補正の例

CPR = 30000 のエンコーダの場合、CPR_INV 設定値は、$2^{24}/30000 = 559.24053$ と計算されます。この値は 559 に丸める必要があります。CPR_INV は、常に、最も近い整数値に切り捨てる必要がある点に注意してください。

未補償の誤差による影響の計算

16 ビットの変換モジュールは、単にエンコーダの角度に CPR_INV を乗算し、次に 256 で除算するだけです。CPR_INV が丸められているため、各エンコーダ・ステップの変換後の角度の変化は、本来の値よりわずかに小さくなります。このわずかな誤差は、エンコーダの角度がその最高値に達するまで積算され、この時点で誤差が最大になります。

エンコーダの角度がゼロに戻る回転の終了時、変換される角度のステップは大きくなり、それよりわずかに小さいすべてのステップを補償します。そのため、複数回の回転があっても誤差が累積されることはありません。

特定のエンコーダに対する最大誤差を判定するために、変換される角度が、デコーダがその最大値を示しているときに丸めを行う場合と行わない場合で比較されます。

- この例では、最大エンコーダ値は 29999（CPR − 1）です。
- この最大値において、丸めを行わない変換後の角度は、$(2^{24}/30000) \times 29999 / 256 = 65533.81547$ です。
- 丸めた CPR_INV を用いた変換後の角度は、$559 \times 29999/256 = 65505.62891$ となります。
- 16 ビットの変換モジュールは常に切り捨てを行います。
  - したがって、この例では、65505 に切り捨てる必要があります。
- 最大誤差は、65533.81547 − 65505 = 28.81547 になります。

丸め誤差の除去

計算された最大誤差をアプリケーションで許容できない場合は、補正 LUT 機能を用いてこれを補正できます。この手順が機能するには、CPR_INV 計算の変更が必要です。つまり、$CPR_INV = (2^{24} - 2^{16})/CPR$ です。

この構成において、補正 LUT を用いてエンコーダ誤差を補償することは、依然として可能ですが、LUT を設定する手順がより複雑になります。

変換誤差を除去するよう LUT を設定するには、CPR_INV の変更された式が必要です。CPR_INV 式の変更が必要になるのは、補正 LUT は隣接する項目間で補間を行い、最初と最後の項目が隣接しているものとしてカウントされるためです。誤差関数は回転の最後では不連続になるため、最初と最後の項目の間で補間を行うのは望ましくありません。

変更された CPR_INV 値は、角度をわずかに縮小した範囲にスケーリングし、それによって、これらの項目の間での補間を防ぎます。縮小した範囲にスケーリングすることで、実際には誤差が更に大きくなりますが、LUT がこれを完全に補正できるようにもなります。

1. LUT を設定するには、まず、変更された CPR_INV 値を計算します。このエンコーダ例（CPR = 30000）の場合、次のようになります。

$$CPR_INV = \frac{2^{24}-2^{16}}{CPR} = \frac{2^{24}-2^{16}}{30000} \approx 557 \ (rounded\ down)$$

2. この値を用いて、最初に変換されるステップのサイズを計算します。これは、エンコーダ角度が 1 の場合の変換された角度です。そのため、最初のステップ・サイズは $1 \times 557 / 256 \approx 2$（小数点以下切り捨て）となります。
3. 最大エンコーダ値で変換後の角度を計算すると、$29999 \times 557/256 \approx 65271$（小数点以下切り捨て）となります。
4. この値を用いて、最後に変換されるステップ（0 にステップ・バック）のサイズを計算します。最後のステップ・サイズは $2^{16} - 65271 = 265$ になります。（変換された値は、2^{16} にラップ・アラウンドされる点に注意してください。これは、16 ビットの範囲の最大値に 1 を加えた値です。）

5. 最初と最後のステップのサイズを計算したので、次のステップは、これらの値の差を計算することです。これは correction_factor = 265 − 2 = 263 です。
6. この場合、計算した補正係数は、8 ビットの LUT 項目の最大値である 127 より大きくなっています。そのため、ゲイン係数を計算する必要があります。ゲイン係数のオプションは、1、2、4、8、16 です。
7. ゲイン係数で除算した補正係数が 127 以下になるように、LUT のゲイン係数を最小値に設定します。例えば、最小の lookup_gain_factor は 4 です。このゲイン係数は、FEEDBACK.LUT レジスタでも定義する必要があります。
8. 次に、各インデックス（ADDR ϵ {0…255}）に対し LUT をプログラムします。その際、0 から始まり、ゲイン係数で除算した補正係数（直近の整数値に切り捨て）で終了する、直線的な増加率で行います。なお、255 という値は、最後の項目のインデックスである点に注意してください（これはスプリット・モードが無効になっていることが前提です）。したがって、各項目の値は次のようになります。
$LUT[ADDR] \frac{correction_factor}{lookup_gain_factor} \times \frac{ADDR}{255}\ with\ ADDR \in \{0 \dots 255\}$
9. 以上でゲイン係数と LUT の項目が設定されたので、LUT によって 16 ビットの変換誤差を完全に除去できます。

LUT をスプリット・モードで使用する場合は、各項目の値を次のように設定します（直近の整数値に切り捨て）。

$$LUT[ADDR] = \frac{correction_factor}{lookup_gain_factor} \times \frac{ADDR}{127}\ with\ ADDR \in \{0 \dots 127\}$$

更に、CPR_INV の計算方法が以下のようにわずかに異なります（小数点以下切り捨て）。

$$CPR_INV = \frac{2^{24} - 2^{17}}{CPR}$$

丸め誤差を補正するよう LUT を設定する前に、コントローラの最新の応答が、丸め誤差補正をしなくて十分かどうかを評価してください。多くの場合、丸め誤差は、最新のコントローラ動作にそれほど大きな変化を加えません。

速度のアクイジション

速度制御を可能にするには、フィードバック経路を速度アクイジション用に設定する必要があります。FEEDBACK.OUTPUT_CONF レジスタの VELOCITY_SRC フィールドを用いて、2 つのフィードバック・チャンネルのいずれかからソースを選択します。

ABN エンコーダまたはデジタル・ホール・センサーを速度測定用に用いる場合、使用されるフィードバック・チャンネルの入力ソースは、それぞれ、ABN_1_FREE、ABN_2_FREE、または HALL_FREE のいずれかのオプション（FEEDBACK.CONF_CH_x レジスタの SRC_SEL_x フィールド、x = A、B）の 1 つにする必要があります。これは、ホール・センサーの場合、および CPR が 2 の指数でない ABN エンコーダの場合に、特にあてはまります。phi_e の計算に用いられるフィードバック・チャンネルは、これまでと同様、通常の ABN_1、ABN_2、または HALL オプションを使用しなくてはならない点に注意してください。そのため、1 つのフィードバック・ソース（ABN エンコーダまたはホール・センサー）のみを用いる場合でも、フィードバック・チャンネルの A と B の両方が必要となります。

上述のように、速度計算用には、周期速度メータおよび周波数速度メータの 2 つのモジュールがあります。これらは同時に動作しますが、そのうちの 1 つの出力のみを速度コントローラの入力として選択できます。速度コントローラの入力は、FEEDBACK.OUTPUT_CONF レジスタの VELOCITY_SELECTION フィールドで定義されます。オプションには、VELOCITY_FRQ、VELOCITY_PER、VELOCITY_EXT の 3 つがあります。これらはそれぞれ、周波数速度メータ、周期速度メータ、外部書込み可能速度値に対応しています。

周期速度メータ

周期速度メータは、連続する角度更新の間の経過時間をその入力で測定することにより機能します。このメータは、モーターが非常に低速の場合でも測定に適しています。

周期速度メータの設定は、FEEDBACK.VELOCITY_PER_CONF レジスタで行います。周期速度測定は、選択したフィードバックの角度が、POS_DEV_MIN フィールドで定められた値を超えて変化した場合にのみ、更新されます。定義された最小変化は、POS_DEV_TIMER フィールドで定められた時間内に生じる必要もあります。タイムアウトは、POS_DEV_TIMER/f_{SYS} で計算され、f_{SYS} の値は、60MHz です。

更に、この速度メータの測定には、移動平均フィルタを使用できます。移動平均ウィンドウの幅は、FEEDBACK.VELOCITY_PER_FILTER レジスタの FILTER_WIDTH フィールドを通じて設定されます。フィルタをディスエーブルするには、FILTER_WIDTH の値を 0_b にセットします。

周期速度メータの計算値は、常に FEEDBACK.VELOCITY_PER レジスタから読み出せます。

周波数速度メータ

周波数速度メータは、一定の設定可能レートで角度値の変化を測定することにより機能します。このメータは、モーター速度が高い方が適しており、非常に低速の場合には推奨されません。周波数速度メータの設定は、FEEDBACK.VELOCITY_FRQ_CONF レジスタで行います。

周波数速度メータの更新レート f_{vel} は、アクティブな PWM 周波数および FEEDBACK.VELOCITY_FRQ_CONF レジスタの VELOCITY_SAMPLING フィールドのダウン・サンプリング係数を用いて、以下の式で定められます。

$$f_{vel} = \frac{f_{PWM}}{VELOCITY_SAMPLING + 1}$$

VELOCITY_SAMPLING フィールドは、速度コントローラ全体の更新レートも定義する点に注意してください。そのため、この値は、周期速度メータのみを使用する場合にも考慮する必要があります。

周波数速度メータには、FEEDBACK.VELOCITY_FRQ_CONF レジスタの VELOCITY_SCALING フィールドを通じて設定される、付加的なスケーリング・ファクタもあります。周波数速度メータにこのスケーリング・ファクタを使用して、周波数速度メータの出力と周期速度メータの出力を一致させることを推奨します。その場合のみ、動作時に両方のメータを動的に切り替えることができます。適切なスケーリング・ファクタを選択するには、以下の式を用います。

$$VELOCITY_SCALING = \frac{2^{24} \times f_{vel}}{f_{SYS}} \; with \; f_{SYS} = 60MHz$$

周波数速度メータの計算値は、常に、FEEDBACK.VELOCITY_FRQ レジスタから読み出せます。

速度メータの比較

ほとんどの状況において、周期速度メータの方が周波数速度メータより正確です。フィードバックの更新周波数が非常に高い場合には、周期速度メータには一定の制限があります。通常、これが発生するのは、高分解能 ABN エンコーダを高いモーター速度で使用する場合のみです。

以下の式において、counts_per_mrev は、16 ビット変換モジュール（上述の式を参照）で計算した場合のように、機械的な 1 回転あたりの等価カウント数を示しています。この値は、IC の設定に用いられるのではなく、速度メータの比較のためだけに用いられる点に注意してください。この値は、対応するチャンネルの、FEEDBACK.CONF_CH_x レジスタの SRC_SEL_x フィールドおよび CPR_INV_x フィールド（x = A、B）の設定に依存します。また、フィードバック・ソースが機械的角度または電気的角度かによっても異なります。以下に値のいくつかの例を示します。

- SRC_SEL_x = ABN_1_FREE、かつ CPR_INV_x = 256（4096 ラインの ABN エンコーダの場合）の場合：
  counts_per_mrev = 16384
- SRC_SEL_x = PHI_EXT_A、かつ CPR_INV_x = 1024（14 ビットの SPI エンコーダの場合）の場合：
  counts_per_mrev = 65536
- SRC_SEL_x = HALL_FREE、かつ CPR_INV_x = 256 の場合：
  counts_per_mrev = $6 \times N_{polepairs}$
- SRC_SEL_x = HALL_FREE、かつ CPR_INV_x = 4096 の場合：
  counts_per_mrev = $96 \times N_{polepairs}$
- SRC_SEL_x = ANALOG_SENSOR、かつ CPR_INV_x = 256 の場合：
  counts_per_mrev = $2^{16} \times N_{polepairs}$

次式を用いると、周期速度メータの移動平均フィルタがディスエーブルの場合、つまり、FEEDBACK.VELOCITY_PER_FILTER レジスタの FILTER_WIDTH フィールドの値が 0_b に設定されている場合に、このメータが歪みを生じずに測定できる最大モーター速度（RPM 単位）を計算できます。

$$v_{PER,max} = \frac{f_{SYS} \times POS_DEV_MIN \times 60}{53 \times \text{counts_per_mrev}} [rpm] \; with \; f_{SYS} = 60MHz$$

次式を用いると、周期速度メータの移動平均フィルタがイネーブルの場合、つまり、FEEDBACK.VELOCITY_PER_FILTER レジスタの FILTER_WIDTH フィールドの値が 0_b 以外の値に設定されている場合に、このメータが歪みを生じずに測定できる最大モーター速度（RPM 単位）を計算できます。

$$v_{PER,max} = \frac{f_{SYS} \times POS_DEV_MIN \times 60}{105 \times \text{counts_per_mrev}} [rpm] \; with \; f_{SYS} = 60MHz$$

与えられたアプリケーションの速度が、計算した$v_{PER,max}$に近い値に達すると予測される場合、POS_DEV_MIN フィールドの値を増加するか、速度コントローラの入力を周波数速度メータへ切り替えることを推奨します。$v_{PER,max}$の実際の値は、現実のセットアップおよび設定によって計算値とは異なる可能性があります。

周期速度メータの性能は速度が増加すると共に低下しますが、周波数速度メータの性能は一定です。次式を用いると、周期速度メータの性能が周波数速度メータの性能より低下する交差ポイント（v_{COP}）を計算できます。

$$v_{COP} = 60 \times \frac{f_{vel} + \sqrt{f_{vel}^2 + f_{vel} \times f_{SYS} \times POS_DEV_MIN \times 4}}{2 \times \text{counts_per_mrev}} [rpm] \quad with\ f_{SYS} = 60MHz$$

なお、v_{COP}の式は、フィードバックからの理想的なポジション・フィードバックが前提となっている点に注意してください。そのため、結果の値は、目安にすぎません。周期速度メータには、ノイズの多い角度フィードバックや不正確な角度フィードバックの影響を受けやすい性質があります。つまり、実際のアプリケーションでは、現実の交差ポイントがこれより低い可能性があります。アプリケーションで目的とする速度範囲に、計算したv_{COP}に近い速度が含まれる場合、両方のメータの実際の性能を実験的に測定し、その実際の測定に基づいて交差ポイントを決定することを推奨します。

図 36 に、両方の速度メータについて、モーター速度が増加した場合の測定ノイズを表す曲線を示します。この場合、測定ノイズが低いことが求められます。

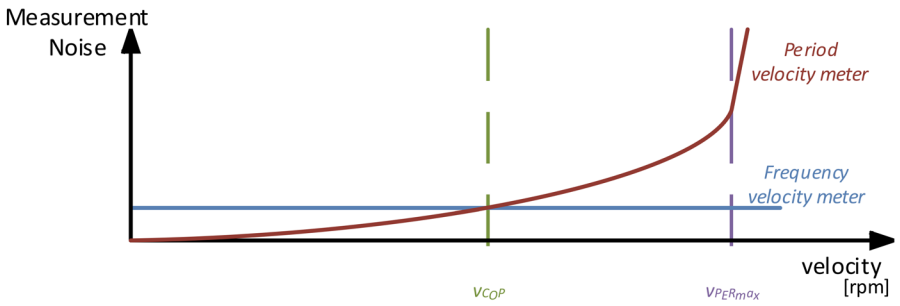


図 36. モーター速度の増加に伴う速度メータの測定ノイズ

設定に応じて、v_{COP}は、$v_{PER,max}$よりも高くなる場合もあれば、低くなる場合もあります。つまり、使用する速度メータを評価する際には、両方のスレッショルドを考慮する必要があります。

ポジションのアクイジション

ポジション制御を可能にするには、フィードバック経路をポジション・アクイジション用に設定する必要があります。FEEDBACK.OUTPUT_CONF レジスタの POSITION_SRC フィールドを用いて、2 つのフィードバック・チャンネルのいずれかからソースを選択します。

ABN エンコーダまたはデジタル・ホール・センサーをポジション・トラッキング用に用いる場合、使用されるフィードバック・チャンネルの入力ソースは、それぞれ、ABN_1_FREE、ABN_2_FREE、または HALL_FREE のいずれかのオプション（FEEDBACK.CONF_CH_x レジスタの SRC_SEL_x フィールド、x＝A、B）の 1 つにする必要があります。これは、ホール・センサーの場合、および CPR が 2 の指数でない ABN エンコーダの場合に、特にあてはまります。phi_e の計算に用いられるフィードバック・チャンネルは、これまでと同様、通常の ABN_1、ABN_2、または HALL オプションを使用しなくてはならない点に注意してください。そのため、1 つのフィードバック・ソース（ABN エンコーダまたはホール・センサー）のみを用いる場合でも、フィードバック・チャンネルの A と B の両方が必要となります。

現実の単位への変換

TMC6460 は、システム・クロック f_{SYS} のリファレンスとして、内部発振器または外部クロック信号を使用できます。速度測定は、ランプ・ジェネレータのモーション・プロファイルと同様、システム・クロックを基準としています。そのため、測定およびランプ・モーション・プロファイルにおいて最高の安定性と再現性を確保するために、外付けの水晶発振器を使用するか、マイクロコントローラから外部クロック信号を供給することを推奨します。

TMC6460 は、様々な PI コントローラやランプ・ジェネレータに対し、内部スケーリングを使用します。このセクションでは、内部値をクローズド・ループ動作中（MCC_CONFIG.MOTOR_MOTION レジスタの RAMP_USE_PHI_E ビットがセットされていないとき）の現実の単位に変換する式を示します。

オープン・ループ・モードでの現実の単位への変換については、オープン・ループでの現実の単位への変換のセクションを参照してください。

ポジションの変換

ポジションのスケーリングは、対応するチャンネルのFEEDBACK.CONF_CH_xレジスタのSRC_SEL_xフィールドおよびCPR_INV_xフィールド（x = A、B）の設定に依存します。表 18 に、内部ローター・ポジションP_{int}から機械的ローター回転数P_{mech}に変換する式を、様々なSRC_SEL_xオプションについて示します。

表 18. ポジションの現実の単位への変換式

SRC_SEL_x	P_{mech}[rev]
ABN_1_FREE/ABN_2_FREE	$P_{int} \times \frac{256}{CPR_INV_x \times CPR_{ABN}}$
ANALOG_SENSOR, ANA_MODE = UVW	$P_{int} \times \frac{256}{CPR_INV_x \times 2^{16} \times N_{polepairs}}$
ANALOG_SENSOR, ANA_MODE = XY	$P_{int} \times \frac{256}{CPR_INV_x \times 2^{16}}$
HALL_FREE	$P_{int} \times \frac{256}{CPR_INV_x \times 6 \times N_{polepairs}}$
PHI_EXT_A/PHI_EXT_B	$P_{int} \times \frac{256}{CPR_INV_x \times CPR_{EXT}}$

この表において、CPR_{ABN}は、ABNエンコーダの1回転あたりのカウント数、CPR_{EXT}は、外部書込み角度値の1回転あたりのカウント数、$N_{polepairs}$は、モーターの極ペアの数です。P_{int}は、内部ポジション・スケーリングを使用するポジション・レジスタの値です。これには、次のレジスタが含まれますが、これに限定されるわけではありません：FOC.PID_POSITION_ACTUAL、FOC. PID_POSITION_TARGET、RAMPER.POSITION。

速度の変換

内部速度値をRPM単位のローター速度に変換するには、次式とスケーリング・ファクタkを使用します。

$$v\,[rpm] = \frac{v_{int}}{k}$$

この場合、$vint$は、内部速度スケーリングを使用する速度レジスタの値です。これには、次のレジスタが含まれますが、これに限定されるわけではありません：FEEDBACK.VELOCITY_FRQ、FEEDBACK.VELOCITY_PER、FOC.PID_VELOCITY_ACTUAL、FOC.PID_VELOCITY_TARGET、およびランプ・ジェネレータ・レジスタからの速度パラメータ。

スケーリング・ファクタkは、対応するチャンネルのFEEDBACK.CONF_CH_xレジスタのSRC_SEL_xフィールドおよびCPR_INV_xフィールド（x = A、B）の設定に依存します。更に、SRC_SEL_x = ANALOG_SENSORを用いる場合は、HALL.ANA_CONFIGのANA_MODEフィールドの設定も影響します。

表 19 に、各バリエーションに対するスケーリング・ファクタを計算する式を示します。ここで、k_{PER}およびk_{FRQ}は、それぞれ、周波数速度メータおよび周期速度メータのスケーリング・ファクタです。なお、上述のようにVELOCITY_SCALINGフィールドに推奨値を用いる場合、k_{PER}とk_{FRQ}の速度スケーリング・ファクタはどちらも同じです。

表 19. 速度の現実の単位へのスケーリング・ファクタ

SRC_SEL_x	k_{PER}	k_{FRQ}
ABN_1_FREE/ABN_2_FREE	$\frac{CPR_{ABN} \times CPR_INV_x \times 2^{24}}{256 \times f_{SYS} \times 60}$	$\frac{CPR_{ABN} \times CPR_INV_x \times VELOCITY_SCALING}{256 \times f_{vel} \times 60}$
ANALOG_SENSOR, ANA_MODE = UVW	$\frac{2^{16} \times N_{polepairs} \times CPR_INV_x \times 2^{24}}{256 \times f_{SYS} \times 60}$	$\frac{2^{16} \times N_{polepairs} \times CPR_INV_x \times VELOCITY_SCALING}{256 \times f_{vel} \times 60}$
ANALOG_SENSOR, ANA_MODE = XY	$\frac{2^{16} \times CPR_INV \times 2^{24}}{256 \times f_{SYS} \times 60}$	$\frac{2^{16} \times N_{polepairs} \times CPR_INV_x \times VELOCITY_SCALING}{256 \times f_{vel} \times 60}$
HALL_FREE	$\frac{6 \times N_{polepairs} \times CPR_INV_x \times 2^{24}}{256 \times f_{SYS} \times 60}$	$\frac{6 \times N_{polepairs} \times CPR_INV_x \times VELOCITY_SCALING}{256 \times f_{vel} \times 60}$
PHI_EXT_A/PHI_EXT_B	$\frac{CPR_{EXT} \times CPR_INV_x \times 2^{24}}{256 \times f_{SYS} \times 60}$	$\frac{CPR_{EXT} \times CPR_INV_x \times VELOCITY_SCALING}{256 \times f_{vel} \times 60}$

ここで、f_{vel}は、上述の周波数速度メータの更新レートです。CPR_{ABN}は、ABN エンコーダの一回転あたりのカウント数、CPR_{EXT}は、外部書込み角度値の一回転あたりのカウント数、$N_{polepairs}$は、モーターの極ペアの数、$f_{SYS} = 60MHz$です。

加速度の変換

速度およびポジションとは対照的に、加速度値はランプ・ジェネレータ・ブロックを通じてのみ用いられます。

加速度の変換は、システム・クロック周波数 f_{SYS}および速度スケーリング・ファクタkにのみ依存します。速度のスケーリング・ファクタは、上述のように、使用する速度メータおよびフィードバック・チャンネルの設定に応じて選択する必要があります。内部加速度値a_{int}をローターの加速度（rev/s² 単位）に変換するには、次式を使用します。

$$a\left[\frac{rev}{s^2}\right] = a_{int} \times \frac{f_{SYS}}{k \times 60 \times 2^{17}} \quad with\ f_{SYS} = 60MHz$$

この場合、a_{int}は、ランプ・ジェネレータ・レジスタからの加速度パラメータの値です。代表的な例は以下のとおりです。RAMPER.A1、RAMPER.D_MAX、RAMPER.ACCELERATION。

モーター制御コア

TMC6460 は、トルクおよび磁束の制御ループや速度およびポジションの制御ループなどの FOC 手法、およびパルス幅変調（PWM）ジェネレータを採用する、高度なモーター制御コア（MCC）を内蔵しています。

電流アクイジション

電流は、3 個のローサイド MOSFET すべてにおいて無損失の電流検出を用い、TMC6460 内部で測定されます。ADC の測定は、新しい PWM 周期の開始時にトリガされます。通常、すべてのローサイド・スイッチはこの時点でオンになるためです。

生の電流値である、ADC.I2_I1_RAW レジスタおよび ADC.VM_I3_RAW レジスタの I1、I2、I3 の各ビットフィールドは、電流検出アンプ（CSA）のゲインを自動的に考慮して計算されます。つまり、測定された所定の電流値に対し、異なる CSA ゲイン値を設定しても、測定結果である I1、I2、I3 の生の電流値は変わりません。

現実の単位での電流検出値 I_{CSx}（x = 1、2、3）は、MCC_CONFIG.GDRV レジスタの LS_RES_ON フィールドでのローサイド抵抗設定値にのみ依存し、表 20 に示す式を用いて生の電流値 I1、I2、I3 から計算できます。

表 20. 電流検出の計算

LS_RES_ON[1:0]	CURRENT VALUE CALCULATION (WITH X = 1, 2, 3)
00	I_{CSx} [A] = Ix × 0.00164 [A] / 8 × 1/4 × 1.000
01	I_{CSx} [A] = Ix × 0.00164 [A] / 8 × 2/4 × 0.992
10	I_{CSx} [A] = Ix × 0.00164 [A] / 8 × 3/4 × 0.988
11	I_{CSx} [A] = Ix × 0.00164 [A] / 8 × 4/4 × 0.982

これらの式は、LS_RES_ON の設定に応じて選択される点に注意してください。ただし、これらは、出力のローサイド・オン抵抗 $R_{ON,LS}$ の値（LS_RES_ON フィールドの値によっても変化）からは直接には導出されません。

3 つの電流値 I1、I2、I3 はすべて、ADC.I123 レジスタの I123 フィールドにまとめられます。理想的な合計値である 0 からの偏差が過大である場合、これは電流測定が不正確であることを意味します。デューティサイクルが 100%に近いフェーズで電流を測定する場合に、これが引き起こされることがあります。この不正確な測定には、後述する自動 Kirchhoff 機能を用いて対処できます。

新しい値は、ADC.I123 レジスタの I123_LIMIT フィールドに定義できます。I123 の絶対値が I123_LIMIT フィールドの値を超えると、ADC.STATUS レジスタの I123_FAIL ビットがセットされます。更に、電流が ADC の範囲を超えたことにより I1、I2、または I3 のいずれかの生の電流値がクリップされると、対応する ADC.STATUS レジスタの I1_CLIPPED、I2_CLIPPED、または I3_CLIPPED のビットがセットされます。また、これらのビットは、CHIP.STATUS_FLAGS レジスタの I_CLIPPED_STATUS ビットにマージされ、対応する CHIP.EVENTS レジスタの I_CLIPPED_EVENT ビットにラッチされます。ADC.STATUS レジスタの個々のイベント・ビットをクリアした後でないと、I_CLIPPED_EVENT はクリアできません。

電流検出アンプのゲイン

上述のように、CSA ゲイン設定値は、I1、I2、I3 の生の電流値に対しトランスペアレントです。ゲイン値が高いほど、内部 ADC によって測定される生の値の分解能が高くなり、ゲイン値が低いほど、より大きな電流を測定できます。静的なゲイン・セットアップの場合の CSA ゲイン値は、MCC_ADC.CSA_GAIN レジスタの CSA_GAIN フィールドを通じて定めることができます。表 21 に、様々な LS_RES_ON および CSA_GAIN の値に対して、ADC が測定できる等価最大電流の絶対値を示します。

表 21. 様々なゲイン設定に対して測定可能な最大電流

CSA_GAIN[1:0]	SENSE CURRENT MULTIPLIER (gain_factor)	ABSOLUTE MAXIMUM SENSE CURRENT [A] DEPENDENT ON LS_RES_ON[1:0]:			
		00	01	10	11
00 (Default)	1x	1.68	3.33	4.98	6.60
01	2x	0.84	1.67	2.49	3.30
10	3x	0.56	1.11	1.67	2.20
11	4x	0.42	0.83	1.24	1.65

これらの値は、ADC の測定限界にのみ対応します。TMC6460 の出力段は、最大で $3A_{RMS}$ および $5A_{FS}$ の電流に対し正しく動作することに注意してください。LS_RES_ON の各値に対するフル・スケール電流については、表 28 を参照してください。

生の電流値 I1、I2、または I3 は、LS_RES_ON と CSA_GAIN の所定の組み合わせの制限を超えるとクリップされます。電流がクリップされる生の絶対値は、2^{15}/gain_factor で定義されます。

電流検出アンプの動的ゲイン

一部のアプリケーションでは、動作中に CSA ゲインを調整すると有用である場合があります。TMC6460 は、動的ゲイン・モードに対応しています。これを機能させるためには、MCC_ADC.CSA_GAIN レジスタの DYN_GAIN_2X_EN、DYN_GAIN_3X_EN、DYN_GAIN_4X_EN の各フィールドを通じて、必要なゲイン値を個別に有効化する必要があります。1 倍の gain_factor には個別のビットがありません。このビットは、動的ゲイン・モードの使用時には常に使用できるためです。

個々の動的ゲイン係数を有効化する他、MCC_ADC.CSA_GAIN レジスタの DYN_GAIN_HYST フィールドにおいて、ヒステリシス値を 0 より大きな値に設定することも必要です。このヒステリシス値は、動的ゲイン・モード時にゲインの変化をトリガしない、1 回の PWM サイクルの間に許容可能な電流変化に関係しています。例えば、PWM 周波数 f_{PWM} が 25kHz の場合、24V 電源で L = 0.5mH のモーターの 1 回の PWM サイクルあたりの理論的な最大電流変化は、di = V/L × dt = 24V/0.0005H × 40µs = 1.92A と定められます。

動的ゲイン・モード中にモニタする電流値は、実際の電流値と電流コントローラの目標電流値のどちらか大きい方です。動的なゲイン・モード時には、有効化可能な最大ゲインは、モニタされる電流値に応じて自動的に指定されます。動的ゲイン・モード中に指定されている最新のゲインは、MCC_ADC.CSA_GAIN レジスタの CSA_GAIN フィールドから読み出せます。

動的ゲイン・モード中にどの CSA_GAIN 値が選択されるかを決定するには、使用するゲイン値に対する電流制限値を定める必要があります。MCC_ADC.DYN_GAIN_LIMIT_2X レジスタの DYN_GAIN_LIMIT_2X フィールド、および MCC_ADC.DYN_GAIN_LIMITS_4X_3X レジスタの DYN_GAIN_LIMIT_3X フィールドと DYN_GAIN_LIMIT_4X フィールドで定めます。目標電流値または実際の電流値は一定の制限値を超えた場合、次に低い CSA_GAIN が選択されます。ただし、その特定のゲインが動的ゲイン変化に対し有効化されていることが前提です。有効化されているすべてのゲインの制限値を超過する場合、CSA_GAIN = 00_b が指定されます。

自動 Kirchhoff

電流は、ローサイド MOSFET がオンのときにのみ測定できるため、ハイサイドのデューティサイクルが非常に高い場合に、ADC が十分な精度で電流を測定できない、ということが起こり得ます。この場合、デューティサイクルが最大時の測定値を、他の 2 つの測定値を用いる Kirchhoff の法則を使用して、置き換えることができます。

この機能は、MCC_ADC.I_GEN_CONFIG レジスタの MEAS_MODE フィールドをセットすることにより、オンにできます。これに対応する同じレジスタの STATUS フィールドは、測定が置き換えられていることを示します。MCC_CONFIG.PWM_PERIOD レジスタの AUTO_KIRCHHOFF_LIM フィールドは、FOC.FOC_UW_UU レジスタおよび FOC.FOC_UV レジスタの UU、UV、UW の各フィールドに匹敵するデューティサイクル値を指定します。3 つのデューティサイクル値のいずれかが AUTO_KIRCHHOFF_LIM を超えた場合、最大デューティサイクルの位相に対し自動 Kirchhoff 機能が用いられます。

自動 Kirchhoff 計算の後に測定される電流は、MCC_ADC.IW_IU レジスタと MCC_ADC.IV レジスタの IU、IV、IW の各フィールドでモニタできます。表 22 に、MCC の電流値と ADC の生の電流値の関係を、どの値も自動 Kirchhoff 計算から取得されていない場合について示します。

表 22. ADC の電流値と MCC_ADC の生の値の関係

MCC_ADC REGISTER FIELD	CORRESPONDING ADC REGISTER FIELD
MCC_ADC.IW_IU::IU	ADC.I2_I1_RAW::I1
MCC_ADC.IV::IV	ADC.I2_I1_RAW::I2
MCC_ADC.IW_IU::IW	ADC.VM_I3_RAW::I3

AUTO_KIRCHOFF_LIM を計算するための式は、次の PWM エンジンのセクションで示します。

PWM エンジン

PWM で生成する出力のスイッチング周波数は、MCC_CONFIG.PWM_PERIOD レジスタの MAX_COUNT フィールドを通じて設定できます。動作中に PWM 周波数を変更することは推奨されません。この周波数は、すべての制御ループの基本となる周波数であり、ループの不安定を引き起こす可能性があるためです。MAX_COUNT = 4800 のリセット値は、25kHz の PWM 周波数に相当します。PWM 周波数は、次式を用いて計算できます。サポートされる最大の PWM 周波数は 200kHz です。この周波数を超えると、電流検出と制御ループが正しく動作しなくなる可能性があります。

$$f_{PWM} = \frac{f_{PLL}}{MAX_COUNT} \quad with\ f_{PLL} = 120MHz$$

テスト用に、MCC_CONFIG.PWM レジスタの CHOP フィールドを通じて、PWM チョッパー方式を操作できます。モーターの適切な動作は、CHOP フィールドが CENTERED に設定されている場合にのみ可能である点に注意してください。

通常動作において BLDC モーターを駆動している場合、出力ごとのデューティサイクルが目標出力電圧に基づいて計算され、これが次に FOC 出力変換後の電流コントローラの出力として計算されます。詳細については、電流制御ループのセクションを参照してください。計算される出力電圧は、FOC.PID_UQ_UD_LIMITS レジスタの PID_UQ_UD_LIMITS フィールドで制限値を設定することにより、制約を設けることができます。

計算される出力デューティサイクルが 100%を超えることのないようにするため、出力電圧には、ユーザにより定義される制限値の他に、内部に最大値があります。この内部最大値は、そのときに使用している変調方式に依存します。MCC_CONFIG.PWM レジスタの SV_MODE フィールドに値を設定することにより、4 種類の変調方式のうちいずれか 1 つを選択できます。表 23 に、様々な変調方式についての簡単な説明と、内部最大出力電圧値を示します。

表 23. 空間ベクトル PWM モード

SV_MODE[1:0]	SVPWMモード	出力電圧の内部最大値	説明
00	DISABLED	16383	サインPWMモード。空間ベクトル変調はディスエーブルされます。
01	HARMONIC	18900	空間ベクトル変調がイネーブルされ、デューティサイクルに3次高調波注入が追加されます。それによって、モーター内で電源電圧VSより高い実効ピーク電圧が可能になります。
01	BOTTOM	18900	空間ベクトル変調が、HARMONICモードの変調としてイネーブルされます。すべての位相から最低の相電圧が差し引かれ、1つの位相が常に0%のデューティサイクルになります。
11	BOTTOM_OFFSET	18900	オプションのオフセットがある場合のBOTTOMと同じです。このオフセットは、MCC_CONFIG.PWMレジスタのFLAT_BOTTOM_OFFSETフィールドを用いてすべての位相に付加できます。

デフォルトの変調方式は、HARMONIC モードです。

上記の様々な変調方式に対して生成される PWM デューティサイクル値を、図 37 に示します。SV_MODE = 00_b（ディスエーブル）を用いる場合、出力電圧ピークの計算値が 16383 の制限値になるときに、100%の PWM デューティサイクル（65535）が生成されます。それと対照的に、その他の SV_MODE を用いる場合、100%のデューティサイクルは、出力電圧ピークの計算値が 18900 のときに生成されます。100%の PWM デューティサイクルは、電源電圧 V_S に等しい電圧に相当します。

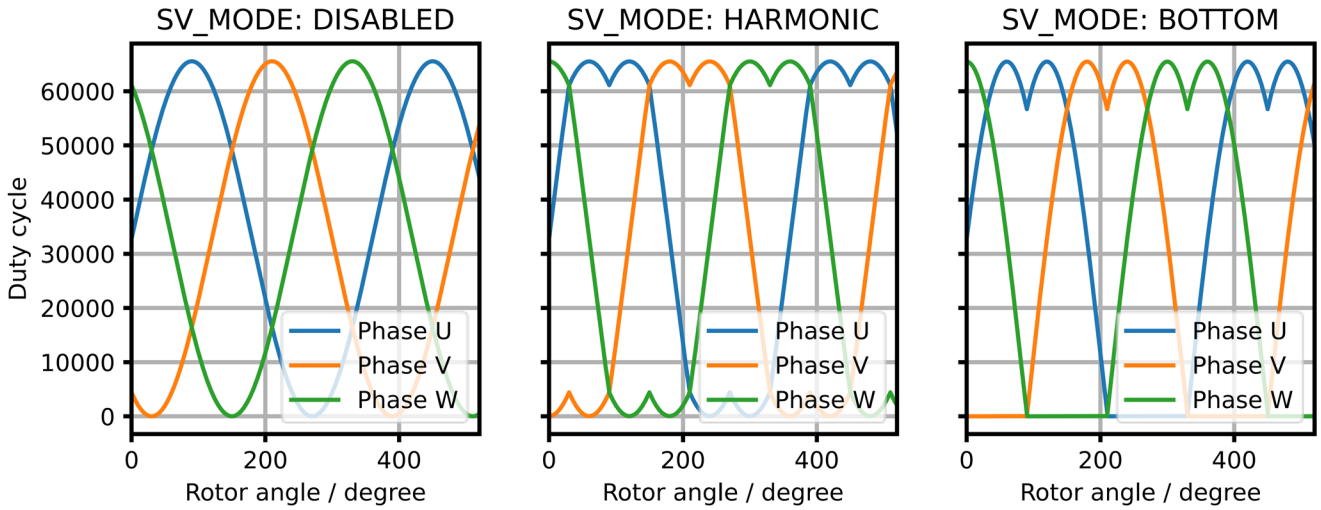


図 37. PWM 変調モード

MCC_ADC.I_GEN_CONFIG レジスタの MEAS_MODE フィールドの自動スイッチ・モード（自動 Kirchhoff 計算）を用いる場合、MCC_CONFIG.PWM_PERIOD レジスタの AUTO_KIRCHHOFF_LIM フィールドも PWM エンジン用に設定する必要があります。デューティサイクルが高すぎるために電流測定が不正確になるのを防止するため、AUTO_KIRCHHOFF_LIM は、デューティサイクルに上限スレッショルドを設定します。1 つの位相のデューティサイクルが設定されたスイッチ制限値を超えると、その特定の位相の電流は測定されません。代わりに、他の 2 つの位相を用いて 3 つ目の位相の電流を計算します。最大スイッチ制限値は、次式で計算できます。

$$AUTO_KIRCHHOFF_LIM_{max} = (1 - T_{set} \times f_{PWM}) \times 2^{16}$$

電流測定が不正確になるのを防ぐため、AUTO_KIRCHHOFF_LIM レジスタは、計算された AUTO_KIRCHHOFF_LIM$_{MAX}$未満に設定する必要があります。この計算値は、設定された PWM 周波数 f_{PWM} と CSA のセトリング・タイム Tset に依存し、このセトリング・タイムは、MCC_CONFIG.GDRV レジスタの SLEW_RATE フィールドの値に依存します。表 24 に、様々なスルー・レートの選択に対して可能な Tset の値を示します。

表 24. CSA セトリング・タイム T_{SET}

SLEW_RATE[1:0]	00	01	10	11
T_{SET} [µs]	2.4	1.8	1.3	1.0

一方、最小の制限値は、選択した SV_MODE に応じ、以下の式で計算できます。

- SV_MODE= DISABLED

$$AUTO_KIRCHHOFF_LIM_{min} = 24576 + 1.5 \times PID_UQ_UD_LIMITS$$

- SV_MODE= HARMONIC

$$AUTO_KIRCHHOFF_LIM_{min} = 32768 + 1.5 \times PID_UQ_UD_LIMITS$$

- SV_MODE= BOTTOM

$$AUTO_KIRCHHOFF_LIM_{min} = 3 \times PID_UQ_UD_LIMITS$$

最小スイッチ制限値は、FOC.PID_UQ_UD_LIMITS レジスタの PID_UQ_UD_LIMITS フィールドで設定されるアクティブな電圧制限値によって異なります。最小制限値は、2 つの位相が同時にスイッチ制限値を超えるような点が波形に存在しないようにするために必要です。そのため、AUTO_KIRCHHOFF_LIM の値は、計算された最小値より大きく、かつ、計算された最大値より小さい値に設定することを推奨します。最小値が最大値より大きいためにこれが不可能な場合は、PID_UQ_UD_LIMITS フィールドに設定された出力電圧制限値を減少する必要があります。それを行うには、上述のように、電圧制限値には、同様に選択した SV_MODE に依存する内部最大値があるのを知っておくことが重要です。PID_UQ_UD_LIMITS をこれより大きな値に調整しても効果はありません。

電流制御ループ

TMC6460 の電流制御ループは、トルク（i_Q）および磁束（i_D）用の PI コントローラ、入力クラークおよびパーク変換、出力電圧リミッタと出力電圧逆パークおよび逆クラーク変換、空間ベクトル PWM ブロックで構成されています。図 38 に電流制御ループの全体構造を示します。

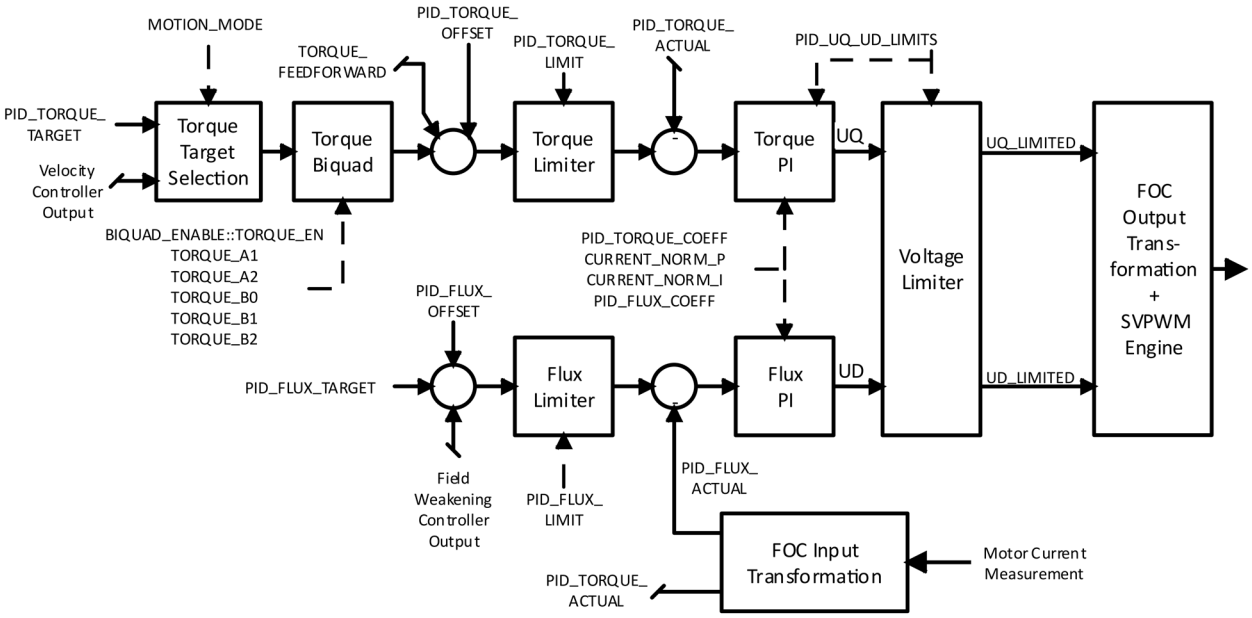


図 38. トルクおよび磁束の制御ループ

入力変換ブロックは、従来用いられている一般的な変換を実行します。FOC.FOC_IBETA_IALPHA レジスタの IALPHA フィールドおよび IBETA フィールドにおいて、暫定的な結果をモニタできます。パーク変換には phi_e の角度が用いられ、通常動作時にフィードバック・エンジンにおいて計算されます。

電流コントローラのメインの入力目標値は、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドに従って切り替えることができます。目標値は、FOC.PID_TORQUE_FLUX_TARGET レジスタの PID_TORQUE_TARGET フィールドおよび PID_FLUX_TARGET フィールドの直接目標を通じて設定されます。あるいは、速度コントローラの出力と弱め界磁コントローラの出力を通じて設定されます。実装されているバイクワッド・フィルタを通じて、トルク・コントローラのメイン入力をフィルタ処理できます。

設定可能なオフセットは、FOC.PID_TORQUE_FLUX_OFFSET レジスタの PID_TORQUE_OFFSET フィールドおよび PID_FLUX_OFFSET フィールドを通じて、電流コントローラに適用できます。

更に、トルク・コントローラに対しては、MCC_CONFIG.TORQUE_FEEDFORWARD レジスタからフィードフォワード成分を追加できます。この合計されたフィードフォワード成分は、加速度、粘性摩擦、クーロン摩擦からなる 3 つのトルク関連フィードフォワード成分を統合しています。次に、個別のトルク・フィードフォワード成分が、ランプ・ジェネレータの速度および加速度出力から生成されます。

各トルク・フィードフォワードは、個別にイネーブルし設定する必要があります。これらをイネーブルするには、MCC_CONFIG.MOTOR_MOTION レジスタの TORQUE_FF_ACC_EN ビット、TORQUE_FF_VISC_FRIC_EN ビット、TORQUE_FF_COULOMB_FRICTION_EN ビットを用い、設定するには、MCC_CONFIG.TORQUE_FF_ACC_CONFIG レジスタ、MCC_CONFIG.TORQUE_FF_VISC_FRIC_CONFIG レジスタ、MCC_CONFIG.TORQUE_FF_COULOMB_FRIC レジスタを用います。

合計された電流コントローラの目標値は、FOC.PID_TORQUE_FLUX_LIMITS レジスタの PID_TORQUE_LIMIT フィールドおよび PID_FLUX_LIMIT フィールドの制限値による制約を受けます。これらの制限された目標値は、FOC.PIDIN_TORQUE_FLUX_TARGET_LIMITED レジスタの PIDIN_TORQUE_TARGET_LIMITED フィールドおよび PIDIN_FLUX_TARGET_LIMITED フィールドで使用できます。

結果として得られる電流コントローラの目標値が実際の値と比較されて、制御電流誤差が計算されます。これは、FOC.PID_TORQUE_ERROR レジスタおよび FOC.PID_FLUX_ERROR レジスタで読み出せます。

トルクおよび磁束の PI コントローラには、固有の比例係数と積分係数があり、これらは、PID_TORQUE_COEFF レジスタの TORQUE_P フィールドと TORQUE_I フィールド、および PID_FLUX_COEFF レジスタの FLUX_P フィールドと FLUX_I フィールドを通じて設定できます。電流コントローラの比例係数と積分係数は、Q16.0 または Q8.8（デフォルト）の固定小数点形式に従い、正規化できます。これは、FOC.PID_CONFIG レジスタの CURRENT_NORM_I フィールド（積分係数）および CURRENT_NORM_P フィールド（比例係数）を通じて設定されます。正規化係数は、トルク・コントローラと磁束コントローラで共有されます。

電流コントローラがアクティブな場合、トルクおよび磁束用の積分器の値は、FOC.PID_TORQUE_INTEGRATOR レジスタおよび FOC.PID_FLUX_INTEGRATOR レジスタを通じて、読出しと上書きができます。

電流コントローラの出力（FOC.FOC_UQ_UD レジスタの UQ フィールドおよび UD フィールド）は、電圧リミッタに転送されます。これらのコントローラの出力は、既に FOC.PID_UQ_UD_LIMITS レジスタ・フィールドによって制限されていますが、その後に更なる制限が追加されます。この電圧リミッタは、必要とされる場合に UD が常に UQ より優先され、また同時に、電圧ベクトルの長さが FOC.PID_UQ_UD_LIMITS レジスタで定められているデューティサイクル制限を超えないような方法で、電圧値を制限します。FOC.FOC_UQ_UD_LIMITED レジスタの UQ_LIMITED フィールドおよび UD_LIMITED フィールドの制限された出力電圧は、その後、出力変換に転送されます。

逆パーク変換では、phi_e の角度を用い、FOC.FOC_UBETA_UALPHA レジスタの UBETA フィールドと UALPHA フィールドの値を計算します。これらの値は、その後、逆クラーク変換に転送され、次にそれによって、FOC.FOC_UW_UU レジスタおよび FOC.FOC_UV レジスタの UU、UV、UW フィールドの値が計算されます。

実際のモーター電圧と合計のモーター電流は、FOC.U_S_ACTUAL_I_S_ACTUAL レジスタの I_S_ACTUAL フィールドおよび U_S_ACTUAL フィールドを通じてモニタできます。実際の目標電流値は、FOC.I_T_ACTUAL レジスタの I_T_ACTUAL フィールドに示されます。これらの値は、次式のように計算されます。

$$I_S_ACTUAL = \sqrt{PID_TORQUE_ACTUAL^2 + PID_FLUX_ACTUAL^2}$$

$$U_S_ACTUAL = \sqrt{U_Q_LIMITED^2 + U_D_LIMITED^2}$$

$$I_T_ACTUAL = \sqrt{PIDIN_TORQUE_TARGET_LIMITED^2 + PIDIN_FLUX_TARGET_LIMITED^2}$$

出力電力は、FOC.P_MOTOR レジスタから読み出すことができます。この値は、I_S_ACTUAL フィールドと U_S_ACTUAL フィールドの値の積として計算されています。

弱め界磁コントローラ

TMC6460 は、設定可能な電圧制限値を超えた場合に負の磁束電流（i_D）を注入する、弱め界磁コントローラを備えています。これはモーターのバック EMF 電圧を実質的に軽減できるため、モーターはより高速で動作できるようになります。ただし、消費電流の増加やトルク（i_Q）の低下を招きます。

この機能を用いる場合には、モーターの合計電流をモニタすることが重要です。この電流が発熱に関連するためです。i_D と i_Q の両方の成分から計算される合計モーター電流は、FOC.U_S_ACTUAL_I_S_ACTUAL レジスタの I_S_ACTUAL フィールドから読み出せます。

図 39 に、速度に応じた弱め界磁コントローラの動作を示します。モーター速度が高まるにつれ、モーターの電圧が増加します。設定された電圧制限値（ここでは公称電圧として記載）までは、電流をトルク発生のためだけに使用できます（i_Q）。電圧制限値に達すると、コントローラは、電流の一部を用いて負の磁束を注入し始めます（i_D）。それにより、トルクは減少しますが、モーターの速度を上げることも可能になります。

図 40 に、弱め界磁コントローラの構造を示します。

弱め界磁コントローラの入力は、FOC.PID_U_S_MAX レジスタの U_S_MAX フィールドに設定されている電圧制限値と、FOC.U_S_ACTUAL_I_S_ACTUAL レジスタの U_S_ACTUAL フィールドにある実際のモーター電圧との差として計算されます。U_S_ACTUAL が U_S_MAX を上回ると、コントローラは、FOC.PID_FIELDWEAK_COEFF レジスタの FIELDWEAK_P フィールドと FIELDWEAK_I フィールドにある比例係数と積分係数を通じ、負の出力値を構成します。どちらの係数も、Q8.8 の固定小数点形式として解釈されます。

弱め界磁コントローラの出力は、その他のユーザにより定義された磁束値と組み合わされ、FOC.PIDIN_FLUX_TARGET レジスタに示されます。この値は、PID_TORQUE_FLUX_LIMITS レジスタの PID_FLUX_LIMIT フィールドに設定された制限値に従う制約を受けます。次にこの制限値は磁束コントローラの入力として用いられ、FOC.PIDIN_TORQUE_FLUX_TARGET_LIMITED レジスタの PIDIN_FLUX_TARGET_LIMITED フィールドで読み出せます。

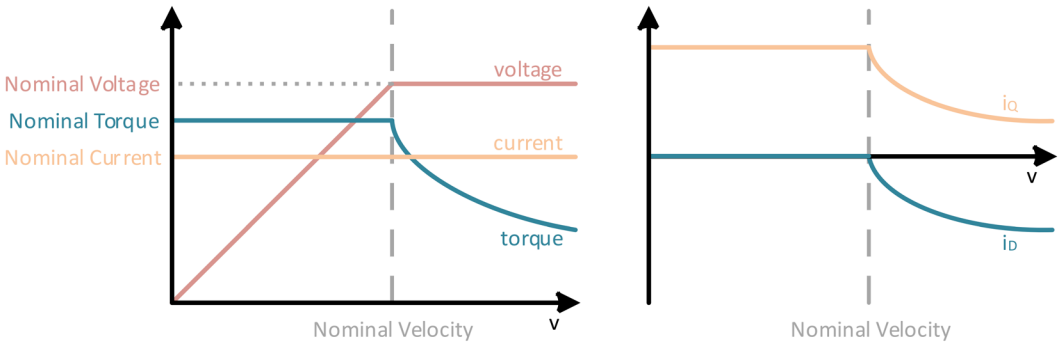


図 39. 弱め界磁の働き

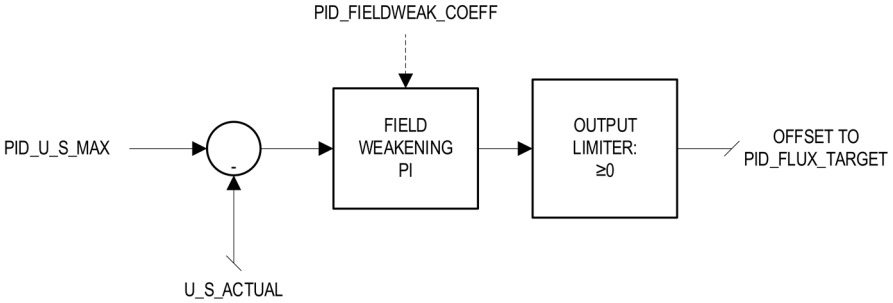


図 40. 弱め界磁の制御ループ

図 38 は、弱め界磁コントローラの出力を磁束コントローラの入力として用いる例を示しています。

速度制御ループ

TMC6460 の速度コントローラは、入力値と出力値に制限のある PI コントローラとして備わっています。図 41 にその構造を示します。

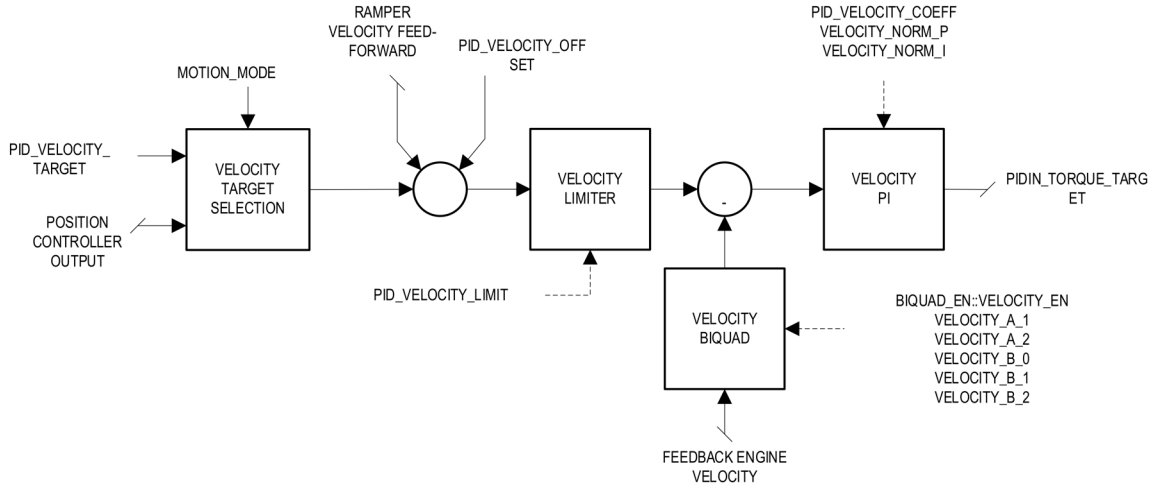


図 41. 速度制御ループ

FOC.PID_VELOCITY_ACTUAL レジスタに示される実際の速度は、速度バイクワッド・フィルタの出力であり、その値はフィードバック・エンジンの速度アクイジション・ブロックから取得されます。

速度コントローラのメインの入力目標値は、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドに従って切り替えることができます。モーション・モードがポジション・モードに設定されている場合、入力の目標値は、ポジション・コントローラの出力から取得されます。モーション・モードが速度モードに設定されている場合は、入力目標値は、ランプ・ジェネレータがイネーブルの場合は、RAMPER.VELOCITY レジスタから取得され、ランプ・ジェネレータがディスエーブルの場合は、FOC.PID_VELOCITY_TARGET レジスタから直接取得されます。ただし、ユーザにより定義された速度モード用の速度目標値は、ランプ・ジェネレータがイネーブルか否かを問わず、常に、FOC.PID_VELOCITY_TARGET レジスタを通じて規定される点に注意してください。その結果得られる入力目標値は、FOC.PIDIN_VELOCITY_TARGET レジスタから読み出せます。

FOC.PID_VELOCITY_OFFSET レジスタを通じて、速度コントローラにオフセットを加えることができます。これは、ユーザにより定義されるフィードフォワード値として用いることを意図されたものです。

更に、ランプ・ジェネレータ・ブロックの RAMPER.V_ACTUAL レジスタからフィードフォワード成分を追加できます。これは、MCC_CONFIG.MOTOR_MOTION レジスタの VELOCITY_FF_EN ビットがセットされている場合に行われます。

速度コントローラの合計目標値は、FOC.PID_VELOCITY_LIMIT レジスタの制限値によって制約を受けます。この制限された目標値は、FOC.PIDIN_TARGET_VELOCITY_LIMITED レジスタに示されています。

その結果得られる速度コントローラの制限入力目標値が実際の速度と比較され、制御速度誤差が計算されます。この誤差は、FOC.PID_VELOCITY_ERROR レジスタで読み出せます。

速度コントローラの比例係数と積分係数は、FOC.PID_VELOCITY_COEFF レジスタの VELOCITY_P フィールドおよび VELOCITY_I フィールドを通じて規定できます。比例係数は、Q16.0、Q8.8（デフォルト）、Q0.16、Q0.24 の固定小数点形式に従って正規化でき、この形式は、FOC.PID_CONFIG レジスタの VELOCITY_NORM_P フィールドで設定されます。同様に、積分係数は、Q8.8、Q0.16（デフォルト）、Q0.24、Q0.32 の固定小数点形式に従って正規化でき、この形式は、同じレジスタの VELOCITY_NORM_I フィールドで設定されます。

個々の正規化係数に加え、FOC.PID_CONFIG レジスタの VELOCITY_SHIFT フィールドに値を設定することにより、速度コントローラの合計出力をシフトできます。FOC.PID_VELOCITY_LIMIT レジスタの制限値は、それに応じてシフトされます。

速度コントローラがアクティブな場合、その積分値は、FOC.PID_VELOCITY_INTEGRATOR レジスタを通じて、読出しと上書きができます。

制限値による制約を無視した、速度コントローラの出力を表す式を以下に示します。

$$PIDIN_TORQUE_TARGET = \frac{\frac{PID_VELOCITY_ERROR \times VELOCITY_P}{2^{VELOCITY_NORM_P}} + PID_VELOCITY_INTEGRATOR}{2^{VELOCITY_SHIFT}}$$

$$with\ PID_VELOCITY_INTEGRATOR = \frac{PID_VELOCITY_ERROR \times VELOCITY_I}{2^{VELOCITY_NORM_I}}$$

$$with\ VELOCITY_NORM_P \in \{0,8,16,24\}, \quad VELOCITY_NORM_I \in \{8,16,24,32\}, \quad VELOCITY_SHIFT \in \{0 \dots 15\}$$

速度コントローラの更新レートf_{vel}は、周波数速度メータのセクションで説明したように、FEEDBACK.VELOCITY_FRQ_CONF レジスタの VELOCITY_SAMPLING フィールドを通じて設定できます。

速度コントローラの出力は、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドの値に応じて、FOC.PIDIN_TORQUE_FLUX_TARGET レジスタの PIDIN_TORQUE_TARGET フィールドに示されます。

ポジション制御ループ

TMC6460 のポジション・コントローラは、入力値と出力値に制限のある PI コントローラとして備わっています。図 42 に、ポジション・コントローラの簡略化した構造図を示します。

実際のポジション値は、FOC.PID_POSITION_ACTUAL レジスタに示されており、この値は、フィードバック・エンジンのポジション・デコーダの出力から取得されます。

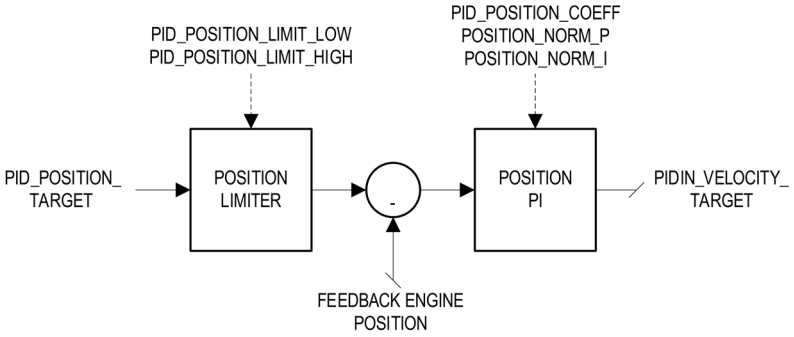


図 42. ポジション制御ループ

ポジション・コントローラの入力目標値は、ランプ・ジェネレータがイネーブルの場合は、RAMPER.POSITION レジスタから取得でき、ランプ・ジェネレータがディスエーブルの場合は、FOC.PID_POSITION_TARGET レジスタから直接取得できます。ただし、ユーザにより定義されたポジション目標値は、ランプ・ジェネレータがイネーブルか否かを問わず、常に、FOC.PID_POSITION_TARGET レジスタを通じて規定される点に注意してください。その結果得られるポジション・コントローラの入力目標値は、FOC.PIDIN_POSITION_TARGET レジスタから読み出せます。

ポジション・コントローラの入力目標値は、FOC.PID_POSITION_LIMIT_LOW レジスタおよび FOC.PID_POSITION_LIMIT_HIGH レジスタの制限値によって制約を受けます。両方の制限値が同じである場合は、何の制限も適用されません。この制限された目標値は、FOC.PIDIN_POSITION_TARGET_LIMITED レジスタに示されています。

その結果得られるポジション・コントローラの制限目標値が実際の値と比較され、制御ポジション誤差が計算されます。この誤差は、FOC.PID_POSITION_ERROR レジスタで読み出せます。

ポジション・コントローラの比例係数と積分係数は、FOC.PID_POSITION_COEFF レジスタの POSITION_P フィールドおよび POSITION_I フィールドを通じて規定できます。比例係数は、Q16.0、Q8.8（デフォルト）、Q0.16、Q0.24 の固定小数点形式に従って正規化でき、この形式は、FOC.PID_CONFIG レジスタの POSITION_NORM_P フィールドで設定されます。同様に、積分係数は、Q8.8、Q0.16（デフォルト）、Q0.24、Q0.32 の固定小数点形式に従って正規化でき、この形式は、同じレジスタの POSITION_NORM_I フィールドで設定されます。

ポジション・コントローラがアクティブな場合、その積分値は、FOC.PID_POSITION_INTEGRATOR レジスタを通じて、読出しと上書きができます。

制限値による制約を無視した、ポジション・コントローラの出力を表す式を以下に示します。

$$PIDIN_VELOCITY_TARGET = \frac{PID_POSITION_ERROR \times POSITION_P}{2^{POSITION_NORM_P}} + PID_POSITION_INTEGRATOR$$

$$PID_POSITION_INTEGRATOR = \frac{PID_POSITION_ERROR \times POSITION_I}{2^{POSITION_NORM_I}}$$

$$with\ POSITION_NORM_P \in \{0, 8, 16, 24\}, \qquad POSITION_NORM_I \in \{8, 16, 24, 32\}$$

ポジション・コントローラの更新レートは、次式のように、速度コントローラの更新レートf_{vel}をダウン・サンプリングした周波数として、FOC.PID_CONFIG レジスタの POSITION_SAMPLING フィールドを通じて設定できます。

$$f_{pos} = \frac{f_{vel}}{POSITION_SAMPLING + 1} = \frac{f_{PWM}}{(VELOCITY_SAMPLING + 1) \times (POSITION_SAMPLING + 1)}$$

VELOCITY_SAMPLING フィールドは、FEEDBACK.VELOCITY_FRQ_CONF レジスタの一部です。その設定については、周波数速度メータのセクションに説明があります。

ポジション・コントローラの出力は、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドの値に応じて、FOC.PIDIN_VELOCITY_TARGET レジスタに示されます。

モーターが目標ポジションに達したら、小さなポジション誤差は無視できます。この場合、制御カスケードはアイドル状態になります。この状態では、ポジション・コントローラがディスエーブルされ、速度コントローラの出力は静的なままになります。この機能を有効化するには、FOC.PID_POSITION_TOLERANCE レジスタと FOC.PID_POSITION_TOLERANCE_DELAY レジスタを、それに対応するよう設定する必要があります。この機能が作用するには、ランプ・ジェネレータもイネーブルする必要があります。

以下の条件が満たされると、制御カスケードはアイドル状態になります。つまり、1) ランプ・ジェネレータの RAMP.POSITION が目標ポジションに達しており、かつ、2) コントローラの誤差の絶対値 FOC.PID_POSITION_ERROR が、少なくとも FOC.PID_POSITION_TOLERANCE_DELAY の値によって定められる PWM サイクル数だけ、FOC.PID_POSITION_TOLERANCE の設定値を下回ることが条件です。ポジション誤差の値が許容帯域から外れるか、目標ポジションが変化すると直ちに、ポジション制御が再びアクティブになり、通常のコントローラ動作が確保されます。

バイクワッド・フィルタ

TMC6460 は、3 つのバイクワッド・フィルタを備えています。そのうちの 2 つは制御カスケード内に配置され、他の 1 つは、内蔵の I/O コントローラが使用します。制御カスケードの場合、フィルタは速度測定のため、および速度コントローラ出力つまりトルク・コントローラ用入力のために用意されています。

速度バイクワッド・フィルタは、フィードバック・エンジンで計算される実際の速度を処理します。その入力は、FEEDBACK.OUTPUT_CONF レジスタの VELOCITY_SELECTION フィールドを通じて設定される、どちらかの速度メータから取得されます。速度バイクワッド・フィルタは、速度コントローラの更新レート f_{vel} でトリガされます。これは、BIQUAD.BIQUAD_EN レジスタの VELOCITY_EN ビットに示されるように、デフォルトでイネーブルされています。フィルタの係数は、BIQUAD.VELOCITY_A1、BIQUAD.VELOCITY_A2、BIQUAD.VELOCITY_B0、BIQUAD.VELOCITY_B1、BIQUAD.VELOCITY_B2 の各レジスタを通じて設定できます。

トルク・バイクワッド・フィルタは、速度コントローラの出力を処理します。トルク・バイクワッド・フィルタも、速度コントローラの更新レート f_{vel} でトリガされます。これは、BIQUAD.BIQUAD_EN レジスタの TORQUE_EN ビットを通じてイネーブルされます。フィルタの係数は、BIQUAD.TORQUE_A1、BIQUAD.TORQUE_A2、BIQUAD.TORQUE_B0、BIQUAD.TORQUE_B1、BIQUAD.TORQUE_B2 の各レジスタを通じて設定できます。

すべての係数は、Q4.20 の固定小数点形式に従って正規化されます。図 43 に、内蔵されているバイクワッド・フィルタのブロック図を示します。伝達関数は以下のように記述できます。

$$G(z) = \frac{y(n)}{x(n)} = \frac{b_0 + b_1 \times z^{-1} + b_2 \times z^{-2}}{1 - a_1 \times z^{-1} - a_2 \times z^{-2}}$$

この場合、係数 A1 および A2 の符号を考慮する必要があります。

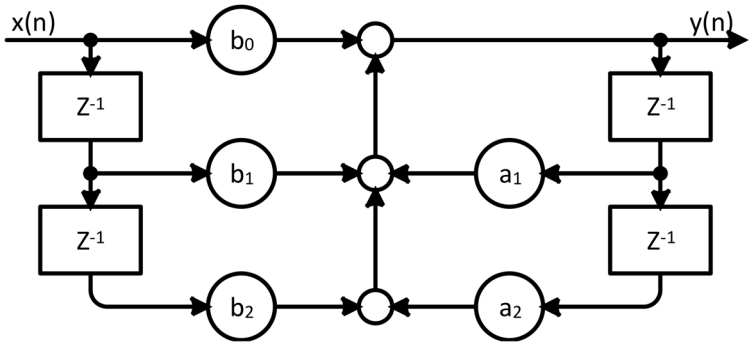


図 43. バイクワッド・フィルタの構造

TMCL-IDE パッケージは、バイクワッド・フィルタ用の調整ツールを提供します。

直接モーション・モード

標準的なクローズド・ループ・モーション・モードの他、TMC6460 には、いくつかの直接制御モードが備わっています。これらは、IC 内の一部の制御ループをバイパスするため、主としてシステム・セットアップおよび試験のために提供されています。

PWM オフ・モードおよび PWM オン・モード

TMC6460 のリセット後は、PWM オフがデフォルトのモードです。このモードは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE を 0（PWM_OFF）に設定することにより、手動で選択できます。このモードでは、PWM エンジンと全制御カスケードがディスエーブルされます。出力段はトライステートです。

MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE を 1（PWM_ON）に設定することにより、PWM エンジンおよび電力段がアクティブになり、すべてのハーフ・ブリッジがトグルを開始して周波数 f_{PWM} の PWM 出力を発生します。このモードでは、MCC_CONFIG.PWM_PERIOD レジスタの MAX_COUNT フィールド、または MCC_CONFIG.PWM レジスタの CHOP フィールドが変化すると、発生する PWM 出力に変化が生じます。f_{PWM} に従う PWM トリガに依存する内部処理も使用可能です。このモードでは、PWM のデューティサイクルは 50%に固定され、カスケード内の各種コントローラは用いられません。つまり、このモードでは、電流がモーターを流れることはなく、モーターの制御はできません。

外部 PWM デューティサイクル・モード

通常動作時、3 つの PWM 出力のデューティサイクルは、電流コントローラの出力電圧に基づいて自動的に計算されます。外部 PWM デューティサイクル・モードを用いることにより、すべての制御ループをバイパスし、出力チャンネルごとに特定のデューティサイクルを直接設定することもできます。

外部 PWM デューティサイクル・モードは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドを 9（PWM_EXT）に設定することで有効化できます。このモードでは、3 つのハーフ・ブリッジのデューティサイクルを、EXT_CTRL.PWM_V_U レジスタおよび EXT_CTRL.PWM_W レジスタの PWM_U、PWM_V、PWM_W の各フィールドを通じて直接設定できます。電流測定は依然として実行されますが、TMC6460 内の制御は行われません。

PWM_U、PWM_V、PWM_W の各フィールドで 0～65535 の値を設定することにより、それに応じてチャンネルごとに 0%～100%のデューティサイクルが生成されます。

オープン・ループ・モーションのモード

モーター・ポジションに関するフィードバックが得られない場合でも、TMC6460 は、オープン・ループ構成でモーターを駆動できます。このモードでは、ランプ・ジェネレータを phi_e の角度のソースとして使用します。これは、一般的には、実際の角度フィードバック・ソースのセットアップ時にのみ役立ちます。

MCC_CONFIG.MOTOR_MOTION レジスタの RAMP_EN ビットおよび RAMP_USE_PHI_E ビットがセットされている場合、RAMPER.PHI_E レジスタの PHI_E フィールドに生成される角度が、FOC エンジンのモーター整流用に用いられます。ランプ・ジェネレータは、MCC_CONFIG.MOTOR_MOTION レジスタの RAMP_MODE フィールドを通じて、速度モードまたはポジション・モードに設定できます。選択したランプ・モードに応じて、対応する目標値を FOC.PID_VELOCITY_TARGET レジスタまたは FOC.PID_POSITION_TARGET レジスタに設定できます。

オープン・ループ動作の場合に対する現実の単位への変換を含む、ランプ・ジェネレータの詳細については、ランプ・ジェネレータのセクションを参照してください。

外部電圧オープン・ループ・モード

通常の FOC 動作では、モーターを流れる電流のトルクおよび磁束の成分（i_Qおよび i_D）を計算するために、位相ごとに測定されるモーター角度のフィードバックおよび電流が用いられます。電流コントローラからは、対応する電圧成分 u_Qおよび u_Dが計算され、これを用いて出力段の PWM デューティサイクルが駆動されます。通常動作とは対照的に、外部電圧モードは、電圧成分 u_Qおよび u_Dを直接設定する方法を提供します。

外部電圧モードは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドを 11（VOLTAGE_EXT）に設定することで有効化できます。このモードでは、EXT_CTRL.VOLTAGE レジスタの UQ フィールドおよび UD フィールドを通じて、電圧を手動で設定できます。電流測定は依然として実行されますが、電流制御は行われません。このモードでは、PWM エンジンの電圧リミッタ部分および出力変換部分はアクティブです。これは、設定された電圧値に応じたデューティサイクルが生成されることを意味します。

外部電圧モード時には、上述のように、ランプ・ジェネレータを用いるオープン・ループ・モードを有効化できます。したがって、EXT_CTRL.VOLTAGE レジスタの UD フィールドに磁束電圧値を設定できます。設定された電圧値が十分に高い場合、ローターは、ランプ・ジェネレータのオープン・ループ角度と、それに対応して発生する磁界に追従します。電圧値が高いほど、ローターは発生する角度に緊密に追従しますが、モーターのコイルに強制的に流れる電流は大きくなります。ローターが発生する磁界に追従できない場合、電圧値を増加する必要があります。

この構成を用いて角度フィードバック・ソースをセットアップする場合、UQ をゼロ、UD をモーターが滑らかに回転するのに十分な大きな正の値に設定することを推奨します。

トルク・オープン・ループ・モード

トルク・オープン・ループ・モードは、電流コントローラがアクティブである点において、外部電圧モードとは異なります。ただし、FOC に用いられる phi_e は、実際の角度フィードバック・ソースから取得されるのではなく、上述のように、ランプ・ジェネレータから取得されます。このモードには、外部電圧モードと比べて、必要な電流を生じるかどうかわからない単なる固定された電圧値の代わりに、既知の測定可能な制御された電流をモーターに印加できるという利点があります。

トルク・オープン・ループ・モードが有効化されるのは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE が 2（TORQUE）に設定され、MCC_CONFIG.MOTOR_MOTION レジスタの RAMP_EN ビットおよび RAMP_USE_PHI_E ビットがセットされている場合です。

この構成を用いて角度フィードバック・ソースをセットアップする場合、PID_TORQUE_TARGET をゼロ、PID_FLUX_TARGET をモーターが滑らかに回転するのに十分な大きな正の値に設定することを推奨します。

注意すべき点は、このモードには電流コントローラが用いられているため、このモードを正しく使用する前に、トルクや磁束のコントローラの係数を、電流制御ループのセクションで説明したように設定する必要があることです。

標準的なモーション・モード

直接モーション・モードおよびオープン・ループ・モーション・モードの他、MCC は標準的な様々なモーション・モードで動作できます。これは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドを通じて設定されます。

トルク・モード（MOTION_MODE＝2）では、電流コントローラ（トルクおよび磁束）と FOC 変換がイネーブルされます。PID_TORQUE_TARGET および PID_FLUX_TARGET の各レジスタ・フィールドは、通常、目標値をコントローラに供給するために用いられます。

速度モード（MOTION_MODE＝3）では、速度コントローラもアクティブ化されます。その出力（PIDIN_TORQUE_TARGET）は、トルク・コントローラに転送されます。レジスタ・フィールド PID_TORQUE_TARGET は、もはやトルク・コントローラの目標値としては用いられません。速度コントローラの目標値は、レジスタ・フィールド PID_VELOCITY_TARGET を通じて供給されます。

ポジション・モード（MOTION_MODE＝4）では、ポジション・コントローラもアクティブ化されます。その出力（PIDIN_VELOCITY_TARGET）は、速度コントローラに転送されます。レジスタ・フィールド PID_VELOCITY_TARGET は、もはや速度コントローラの目標値としては用いられません。ポジション・コントローラの目標値は、レジスタ・フィールド PID_POSITION_TARGET を通じて供給されます。

レジスタ・フィールド PID_FLUX_TARGET は、速度モードおよびポジション・モードの場合でも、磁束コントローラの目標値として引き続き使用される点に注意してください。そのため、速度モードおよびポジション・モードで通常動作させるには、このフィールドを 0 にセットする必要があります。

モーション・モード間の切り替え

動作中に 2 つの異なるモーション・モード間で切り替えを行う最も確実な方法は、モーターを確実に停止状態にし、切り替え先のモードの目標値を慎重に選択することです。更に、ポジション・モードと速度モードの間で切り替えを行うときにランプ・ジェネレータが用いられている場合は、それに応じてランプ・モードを変更することも重要です。

一般的には、動作中およびモーターの稼働中、3 つの標準的なモーション・モードの間で切り替えが可能です。ただし、突然の動作や大きな電流スパイクの発生を防ぐために、いくつかの点を考慮する必要があります。切り替え時の正確な動作は、元のモーション・モードと目的のモーション・モード、および切り替え時の FOC.PID_CONFIG レジスタの OVERWRITE_TARGET ビットの値に依存します。

OVERWRITE_TARGET をセットする際の一般的な考慮事項は、2 つのモーション・モード間でジャークのない遷移が可能であることです。これは、様々なコントローラの適切な積分器や目標の値を自動的に上書きすることによって、実現されます。ランプ・ジェネレータ・モードの変更は、OVERWRITE_TARGET がセットされている場合でも保持されなくてはなりません。

OVERWRITE_TARGET がセットされているときに、1 つのモーション・モードから、制御カスケード内の「下位」レベルにある別のモーション・モードへ切り替え（速度→トルク、ポジション→トルク、または、ポジション→速度）を行うと、変更先のモーション・モードの特定の目標レジスタが自動的に上書きされます。具体的には、ポジション・モードまたは速度モードからトルク・モードに切り替える場合、切り替え時に、PID_TORQUE_TARGET レジスタ・フィールドが、内部のトルク目標値に自動的に上書きされます。ポジション・モードから速度モードに切り替える場合は、PID_VELOCITY_TARGET レジスタ・フィールドが内部の速度目標値に自動的に上書きされます。

対照的に、1 つのモーション・モードから、制御カスケード内の「上位」レベルにある別のモーション・モードへ切り替え（トルク→速度、トルク→ポジション、または、速度→ポジション）を行う場合、滑らかな遷移を確実なものにするためには、OVERWRITE_TARGET がセットされている場合であっても、切り替え先のモーション・モードの特定の目標レジスタを手動で指定する必要があります。トルク・モードからポジション・モードまたは速度モードへ変更するという特定の場合では、ジャークを最小限に抑えるのを支援するために、速度積分器レジスタが、トルク目標レジスタの最新の値に自動的に上書きされます。

PRBS のモーション・モード

システムを識別するために、疑似ランダム 2 進シーケンス（PRBS）モードと呼ばれる、いくつかのモードが使用できます。これらのモードは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドを通じても選択できます。これらのモードでは、対応する制御モードの入力信号を生成するために PRBS ジェネレータが用いられます。PRBS ジェネレータには、20 ビットのフィードバック・シフト・レジスタ（ビット 0～ビット 19）があり、図 44 および次式に示すように、ビット 16 とビット 19 の XOR 操作の結果がレジスタにシフト・バックされます。

$$PRBS_REG[0] = PRBS_REG[16] \;\; xor \;\; PRBS_REG[19]$$

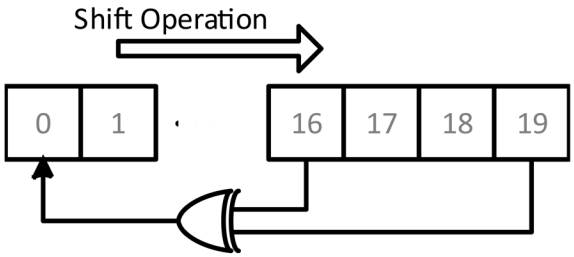


図 44. PRBS シフト・レジスタ

すべてのビットを 1 にセットすることで、PRBS_REG が初期化され、このとき、FOC.PRBS_AMPLITUDE レジスタはゼロになります。シフト動作は、FOC.PRBS_DOWN_SAMPLING_RATIO レジスタを通じて設定可能なダウン・サンプリング・レートを用いて、f_{PWM} に同期して実行されます。したがって、PRBS シフト周波数は次式で定まります。

$$f_{PRBS} = \frac{f_{PWM}}{PRBS_DOWN_SAMPLING_RATIO + 1}$$

各シフトの後、PRBS ジェネレータの出力が形成されます。PRBS_REG[0]が 1 の場合、出力として PRBS_AMPLITUDE が用いられます。PRBS_REG[0]が 0 の場合、-PRBS_AMPLITUDE が用いられます。表 25 に様々なモードを示します。

表 25. PRBS のモーション・モード

MOTION_MODE	PRBS OUTPUT ON	USE CASE
PRBS_UD	UD	Identification of flux controller plant.
PRBS_FLUX	TARGET_FLUX	Identification of flux controller bandwidth.
PRBS_TORQUE	TARGET_TORQUE	Identification of torque controller bandwidth.
PRBS_VELOCITY	TARGET_VELOCITY	Identification of velocity controller bandwidth.
PRBS_POSITION	TARGET_POSITION	Identification of position controller bandwidth.

上述のように、これらのモードは、コントローラの特性を識別するために用いられ、モーション・コントロールのためではありません。

DC モーター・モード

TMC6460 は、三相 BLDC モーターではなく DC モーターを駆動するために使用することもできます。このモードは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTOR_TYPE フィールドを 1（DC）に設定することで有効化できます。

DC モーターは、最初の 2 つの出力チャンネル OUT1 および OUT2 のみを使用します。3 つ目のモーター出力はディスエーブルされます。

電流測定の場合、両方のチャンネルが冗長的に測定されます。ADC.I2_I1_RAW レジスタがその結果を示します。設定された自動 Kirchhoff 制限に応じて、測定された電流値の一方のみが電流制御用に用いられます。そのために、MEASUREMENT_MODE が AUTOMATIC_SWITCH（MCC_ADC.I_GEN_CONFIG レジスタ）に設定されている場合、MCC_CONFIG.PWM_PERIOD レジスタの AUTO_KIRCHHOFF_LIM フィールドを約 50%のデューティサイクル（32768）に設定する必要があります。電流制御用に選択された電流は、IU レジスタ・フィールド（MCC_ADC.IW_IU レジスタ）で読み出せます。DC モーター・モードでは、その他の位相はどれも用いられません。

PWM 変調を、MCC_CONFIG.PWM レジスタの SV_MODE フィールドを用いて調整することもできます。DC モーターでは 3 次高調波注入を用いることはできないため、DISABLED、HARMONIC、BOTTOM_OFFSET の各オプションは、常に約 50%のデューティサイクルのデフォルト変調を行います。BOTTOM モードも使用可能であり、これは、2 つの出力のうちの 1 つが常にゼロでグラウンドに接続されている状態で変調を行います。PWM 変調の詳細については、PWM エンジンのセクションを参照してください。

電気的角度（phi_e）という概念は DC モーターにはないため、DC モーター・モードでは、FOC の入力変換および出力変換は単にバイパスされます。それに代わり、トルク電流 i_Q が、IU フィールドに示される直接相電流測定に等しいと単に仮定され、磁束電流 i_D は計算されません。

そのため、DC モーターの電流の安定化には、トルク・コントローラのみが用いられます。したがって、トルク・モードで動作している場合、目標値は、対応する PID_TORQUE_TARGET レジスタ・フィールドを通じて設定する必要があります。

同様に、PWM の生成には、トルク・コントローラの出力電圧 u_Q のみが用いられます。そのため、外部電圧モードで動作している場合、目標出力電圧は、対応する UQ レジスタ・フィールドで設定する必要があります。

通常は phi_e の角度に依存するその他のプロセスも、DC モーター・モードでは異なったものになります。例えば、ランプの生成した phi_e 角度にモーターを同期させるためにオープン・ループ・モードを用いることは、もはやできません。また、フィードバック角を内部 phi_e に揃えるために必要なプロセスも、もはや不要です。

それ以外、速度およびポジションのモーション・モードは、通常どおり使用できます。その理由は、速度コントローラおよびポジション・コントローラは、phi_e の角度に依存しておらず、外部角度フィードバック・ソースが提供するモーターの機械的角度に依存するためです。

電力段の設定

内蔵のパワー・ドライバ段を正しく動作させるために必要な設定は、わずかです。電流アクイジションのセクションでは、電流検出アンプのゲインや、ローサイドのオン抵抗値など、モーター電流に関する考慮事項について説明しています。

出力段のスルー・レートは、MCC_CONFIG.GDRV レジスタの SLEW_RATE フィールドに値を設定することにより変更できます。表 26 に、様々な SLEW_RATE オプションに対する出力スルー・レート値を示します。

表 26. SLEW_RATE レジスタ・フィールド

SLEW_RATE[1:0]	00	01 (Default)	10	11
OUTPUT SLOPE [V/µs]	100	200	400	800

ドライバ段をイネーブルし、ドライバ・オン状態にするには、DRV_EN_BIT ビットフィールドをセットします。これを引き起こすには、DRV_EN ピンをアサートする必要もあります。CHIP.STATUS_FLAGS レジスタの GDRV_ON_STATE ビットは、システムがドライバ・オン状態になっているかどうかを識別するために使用できます。同じレジスタの SYS_READY_STATE ビットは、システムがドライバ・オフ状態になっているかどうかを示します。

一部のドライバ段で故障イベントが発生すると、ドライバ段が自動的に非アクティブになります。ほとんどの場合、DRV_EN_BIT ビットフィールドも自動的にクリアされます。それにより、ドライバ段の確実でユーザにより制御された再アクティブ化が可能になります。

DRV_EN_BIT ビットフィールドが自動的にクリアされない特別な状況については、以下のセクションで説明します。

過電流保護

TMC6460 には過電流保護（OCP）機構が備わっており、外部の短絡回路や大きな電流ピークを検出した場合に、内部 FET を保護します。2 つの過電流保護モードを備えています。現在アクティブな過電流保護モードは、MCC_CONFIG.GDRV レジスタの OCP_DETECTION_MODE ビットを通じて設定されます。各モードの説明を表 27 に記載します。

表 27. OCP モード

OCP制御モード	OCP_DETECTION_MODE	説明
Current	0	パワーFETを通じて測定された電流に基づく過電流検出。
Voltage	1	パワーFETで測定されたVDSに基づく過電流検出。

OCP スレッショルドを設定することが可能です。それを行うには、MCC_CONFIG.GDRV レジスタの LS_RES_ON フィールドを用います。表 28 に、電流検出モードになっている場合に OCP がトリガされる電流値を示します。

表 28. 過電流保護のスレッショルド

LS_RES_ON[1:0]	FULL-SCALE CURRENT [A]	OCP THRESHOLD MINIMUM [A]
00	I_{FS} × (1/4)	I_{FS} × (1/2)
01	I_{FS} × (1/2)	I_{FS}
10	I_{FS} × (3/4)	I_{FS} × (3/2)
11	I_{FS}	I_{FS} × 2

最大フル・スケール電流 I_{FS} は 5A である点に注意してください。

過電流イベントが生じた場合、ドライバは、特定の PWM サイクルの間、オフに切り替わります。更に、以下に説明するように、短絡の場所に応じて、MCC_CONFIG.MCC_STATUS レジスタのフォルト・フラグがアクティブになります。

- 地絡（x = 1、2、3）：
  - 短絡が続く限り、SHRT2G_CHx_STATUS がセットされます。
  - SHRT2G_CHx_EVENT がセットされ地絡の発生を示します。クリアするには「1」を書き込みます。
  - ハイサイドのドライバのみがディスエーブルになります。
- V_S への短絡（x = 1、2、3）：
  - 短絡が続く限り、SHRT2V_CHx_STATUS がセットされます。
  - SHRT2V_CHx_LATCH がセットされ V_S への短絡の発生を示します。クリアするには「1」を書き込みます。
  - ローサイドとハイサイドのドライバがディスエーブルになります。

6 個のステータス・ビットはすべて、CHIP.STATUS_FLAGS レジスタの SHRT_FAIL_STATUS ビットにマージされます。

3 連続 PWM サイクルで短絡が検出された場合、ドライバ段全体がディスエーブルになり、IC がドライバ・オフ状態（SYS_READY）にセットされて IC を保護します。

MCC_CONFIG.GDRV レジスタの OCP_AUTORETRY フィールドで再試行時間を選択することにより、ドライバ段の自動再始動が 1ms、5ms、または 50ms 後に試行されます。このフィールドが 11_b（デフォルト）に設定されている場合、いかなる自動再始動も試行されません。

短時間のみ検出された過電流状態をマスクするために、デグリッチ時間を定めることができます。MCC_CONFIG.GDRV レジスタの OCP_DEGLITCH フィールドを通じて、0.3µs～2.4µs のデグリッチ時間を選択できます。表 29 に、OCP_DEGLITCH オプションと OCP_AUTORETRY オプションを示します。

表 29. 過電流保護のデグリッチおよび自動再試行時間

OCP_DEGLITCH[1:0]	DEGLITCH TIME [µs]
00	0.3
01	0.6
10	1.2
11	2.4

OCP_AUTORETRY[1:0]	RETRY TIME [ms]
00	1
01	5
10	50
11	No retry

自動再試行機能を用いる場合には、注意が必要です。過電流の根本原因が解決していないと、ドライバは OCP トリガと再始動を繰り返す状態になる可能性があるためです。

ソフト過電流制限

TMC6460 には、ソフト過電流制限機能も付加されています。ADC.I2_I1_RAW レジスタおよび ADC.VM_I3_RAW レジスタの I1、I2、I3 ビットフィールドにおける生の電流測定値のいずれかの絶対値が、MCC_ADC.CURRENT_OVERLOAD レジスタの CURRENT_OVERLOAD フィールドの値を超えた場合、MCC_ADC.EVENTS レジスタの CURRENT_OVERLOAD_STATUS ビットがセットされます。このビットは、CHIP.STATUS_FLAGS レジスタにも転送されます。

過熱保護

TMC6460 は、過熱保護機能も備えています。いずれかの位相でドライバ段の温度が 165ºC を超えた場合、対応する MCC_CONFIG.MCC_STATUS レジスタの OVERTEMP_CHx_STATUS ビット（x = 1、2、3）がセットされ、ドライバ段はオフに切り替わります。

OVERTEMP_CHx_STATUS ビットはすべて、CHIP.STATUS_FLAGS レジスタの OT_FAIL_STATUS ビットにマージされます。

MCC_CONFIG.GDRV レジスタの OVERTEMP_LATCH ビットがセットされておらず（デフォルト）、かつ、温度が 145ºC 未満の場合、ドライバ段は自動的にアクティブになり、それによって、システムはドライバ・オン状態（GDRV_ON）に戻ります。OVERTEMP_LATCH ビットがセットされている場合、ドライバ段を手動で再イネーブルできるためには、その前に、MCC_CONFIG.MCC_STATUS レジスタの OVERTEMP_CHx_EVENT ビットを手動で（それらに「1」を書き込むことにより）クリアする必要があります。

ソフト過熱制限

TMC6460 には、ADC.TEMP_RAW レジスタの TEMP_INT フィールドおよび TEMP_EXT フィールドの内部温度測定値および外部温度アナログ入力値用に、ソフト過熱制限機能が付加されています。

この機能を設定するには、MCC_ADC.TEMP_LIMITS レジスタの TEMP_INT_LIMIT フィールドや TEMP_EXT_LIMIT フィールドの値を設定します。内部温度測定または外部温度アナログ入力が、設定されたこれらの制限値を超えた場合、対応する MCC_ADC.EVENTS レジスタの TEMP_INT_LIMIT_EXCEEDED ビットや TEMP_EXT_LIMIT_EXCEEDED ビットがセットされます。

どちらのビットも、CHIP.STATUS_FLAGS レジスタの OVERTEMP_WARN_STATUS ビットにマージされます。

過電圧故障およびブレーキ・チョッパーの出力

ADC.VM_I3_RAW レジスタの VM フィールドにある内部測定電源電圧が MCC_CONFIG.BRAKE_CHOPPPER_LIMITS レジスタの UPPER_SWITCH_LIM の値を超えた場合、CHIP.STATUS_FLAGS レジスタの OV_VM_LIMIT_FAIL_STATUS ビットがセットされます。このビットは、VM の値が同じレジスタの LOWER_SWITCH_LIM フィールドの値未満に低下するまで、アクティブ状態のままです。それによって、VM 電源電圧モニタリングでのヒステリシスが可能になります。

更に、OV_VM_LIMIT_FAIL_STATUS ビットは、BRAKE 出力に直接接続されています。したがって、取り付けられたブレーキ・チョッパー回路を駆動できます。これが作用するためには、BRAKE 出力が TMC6460 のいずれかのピンの代替機能として割り当てられている必要があります。使用可能なピンについては、ピンの再割り当てのセクションを参照してください。

更に、BRAKE 出力を用いてブレーキ・チョッパー回路を駆動する場合、選択したピンが、駆動しているトランジスタに直接接続されていてはいけない点に注意してください。代わりに、このピンは、ゲート・ドライバ、シュミット・トリガ、バッファ回路などの、外部コンポーネントに接続する必要があります。図 60 および図 61 のアプリケーション例の図に、これを示します。

出力周波数制限

TMC6460 には、599Hz の固定出力周波数制限があります。出力周波数が 599Hz より高いことが検出されると、ドライバ段は自動的に非アクティブになり、CHIP.STATUS レジスタの VEL_FAIL_STATUS ビットがセットされます。対応する CHIP.EVENTS の VEL_FAIL_EVENT ビットもセットされます。

この場合、出力周波数は、モーターのコイルに現れるサイン波電流の周波数を指します。つまり、モーターの速度が毎秒 599 電気的回転に制限されています。rpm 単位での対応するモーター速度制限値は、次式に従い、モーターの極ペアの数の係数として計算されます。

$$v_{599Hz,lim}[rpm] = \frac{599 \times 60}{N_{polepairs}}$$

例えば、モーターの極ペアが 1 つの場合、出力周波数制限による速度制限値は、35940rpm になります。4 極ペア・モーターの場合は速度制限値は 8985rpm になります。その他の場合も同様に計算できます。

出力周波数制限値は、常に、f_{OSC} = 15MHz ± 5%で動作する内部発振器で計算される点に注意してください。実際の出力周波数が確実に 599Hz を超えないようにするために、制限値は、15MHz + 5%の最も厳しいシナリオを仮定して計算されます。そのため、周波数制限は、実際の出力周波数が 599Hz になる前にトリガされる可能性があります。

例えば、特定のサンプルに対する内部発振器が 15MHz で動作している場合、周波数制限は、実際の出力周波数が約 570Hz のときにトリガされます。同様に、特定のサンプルに対する内部発振器が 15MHz - 5%で動作している場合、周波数制限は、実際の出力周波数が約 542Hz のときにトリガされます。

ゼロ電流限界

TMC6460 の電力段のスイッチング特性が必ずしも理想的ではないために、電力段がアクティブに切り替わった後にモーターを流れる電流が正確にゼロになる状態を実現するのは不可能です。この限界による影響は、少なくとも 2 つの電力段チャンネルが同じ PWM のデューティサイクルで動作している場合に現れる可能性があります。そのような状況が発生するのは、例えば、電流コントローラが電流をゼロに安定化しようとする場合であり、これは、速度またはポジションのモーション・モード時に特定のシナリオでも生じる可能性があります。

この限界は、モーター電流のわずかな発振が原因です。発振の振幅と周波数は、モーターのインピーダンス、選択された PWM 周波数、制御ループ・カスケードの設定された PI 調整パラメータによって異なります。つまり、この影響は、与えられたアプリケーションにおいて、多少なりとも現れます。

この限界による影響を軽減する最も直接的な方法は、電流コントローラに対し、より高強度の PI 調整を適用することです。

ランプ・ジェネレータ

TMC6460 のランプ・ジェネレータは、単一の目標値に基づき、滑らかなモーションを実現します。これは、設定されたランプ・パラメータに基づきモーション・プロファイルを自動的に計算し、ジャークを最小限に抑えたモーションを可能にします。生成されるプロファイルのタイプに応じて、ランプ・ジェネレータは、速度モードまたはポジション・モードのどちらかで動作できます。

ランプ・ジェネレータは、目標のポジションまたは目標の速度をリアルタイムで変更できます。また、使用中のモーション・プロファイルの変更も可能です。更に、ランプ・ジェネレータは、リファレンス・スイッチが検出された場合や、目標値からの偏差が大きくなりすぎていることが検出された場合などの、特定のイベントに基づき、自動的にモーションを停止するよう設定できます。

ランプ・ジェネレータは、MCC_CONFIG.MOTOR_MOTION レジスタの RAMP_EN ビットをセットすることによって、イネーブルになります。ランプ・モードは、同じレジスタの RAMP_MODE ビットフィールドを通じて設定されます。

これとは別に、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドが使用可能である点に注意してください。上述のように、このフィールドは、ランプ・モードとは独立した、MCC のモーション・モードを設定します。表 30 に、様々なモーション・モードに推奨されるランプ・モードの使用法を示します。速度またはポジションのモーション・モードの間、不適切なランプ・モードを使用しないよう、注意が必要です。モーターの望ましくない動作の原因になる可能性があるためです。

表 30. 様々な MOTION_MODE の選択に対するランプ・ジェネレータの使用法

MOTION_MODEの選択	ランプ・ジェネレータの使用法（RAMP_MODE）	コメント
VOLTAGE_EXT	RAMP_VELOCITY or RAMP_POSITION	オープン・ループ動作中のモーター回転に必要なランプ・ジェネレータ。詳細については、オープン・ループ・モーションのモードを参照してください。
TORQUE	RAMP_VELOCITY or RAMP_POSITION	オープン・ループ動作中のモーター回転に必要なランプ・ジェネレータ。クローズド・ループ動作には使用しません。
VELOCITY	RAMP_VELOCITY only	速度フィードフォワードをディスエーブルする必要があります。
POSITION	RAMP_POSITION only	速度フィードフォワードを使用できます。

図 45 に、ランプ・モードおよびモーション・モードに応じたコントローラの入力目標を、ブロック図で示します。ランプ・ジェネレータがイネーブルの場合、RAMPER.V_ACTUAL レジスタおよび RAMPER.POSITION レジスタのランプ値は、それぞれ、速度コントローラおよびポジション・コントローラの入力目標値として用いられます。

ユーザにより定義された目標値は、ランプ・ジェネレータがイネーブルか否かに関係なく、常に、FOC.PID_VELOCITY_TARGET レジスタ（速度モーション・モードの場合）および FOC.PID_POSITION_TARGET レジスタ（ポジション・モーション・モードの場合）を通じて設定されます。例えば、速度モーション・モードで動作する場合、速度目標は PID_VELOCITY_TARGET を通じて定められます。ランプ・ジェネレータがディスエーブルの場合、PID_VELOCITY_TARGET の値は、速度コントローラへの入力として、直接取得されます。しかし、ランプ・ジェネレータがイネーブルの場合は、PID_VELOCITY_TARGET の値はランプ・ジェネレータの入力として取得され、RAMPER.V_ACTUAL の値（ランプ・ジェネレータの出力）が速度コントローラの入力として用いられます。

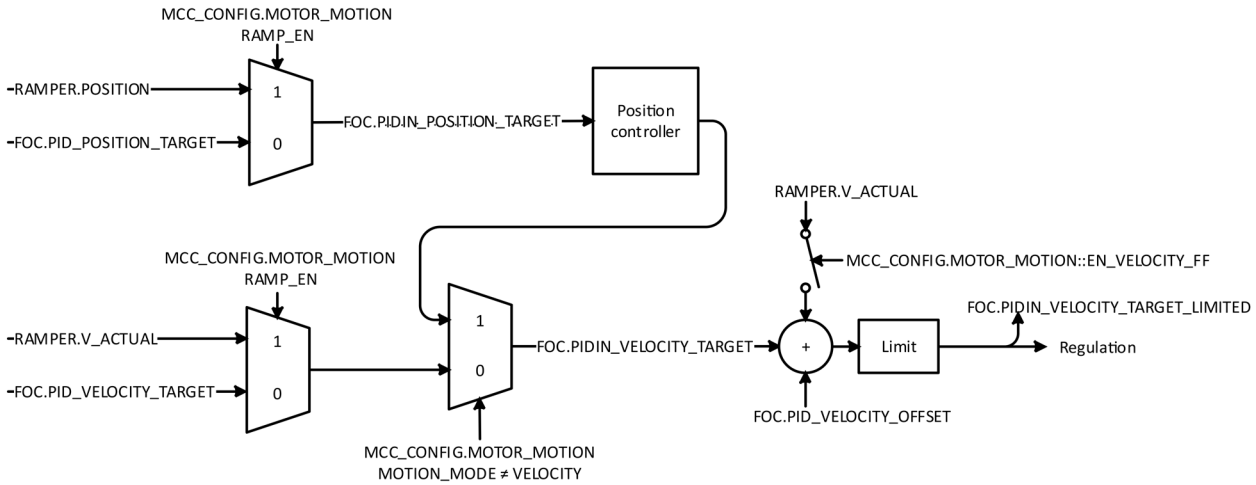


図 45. ランプ・コントローラの目標値の生成

電流制御ループのセクションおよび速度制御ループのセクションで説明したように、ランプ・ジェネレータは、速度コントローラまたはトルク・コントローラへのフィードフォワード入力を生成するために使用できます。RAMPER.V_ACTUAL レジスタの値は、速度コントローラの入力として使用でき、一方、RAMPER.V_ACTUAL レジスタおよび RAMPER.ACCELERATION レジスタの値は、トルク・コントローラの入力として使用できます。

ランプ・ポジショニング・モード

ランプ・ジェネレータは、MCC_CONFIG.MOTOR_MOTION レジスタの RAMP_MODE フィールドを 0_b に設定した場合、ポジショニング・プロファイル用に設定されます。ユーザの定義による新しい目標ポジションが、FOC.PID_POSITION_TARGET レジスタを通じてコマンドされた場合、ランプ・ジェネレータは、目標ポジションに到達するためのポジショニング・モーション・プロファイルを生成する、8 点速度ランプを計算します。

生成されたプロファイルからの最新の速度およびポジションの値は、RAMPER.V_ACTUAL レジスタおよび RAMPER.POSITION レジスタから読み出せます。ランプ・ジェネレータがイネーブルされると、RAMPER.POSITION レジスタの値がポジション・コントローラの入力として用いられます。

8 点速度ランプは、表 31 に示すように、定義されたランプ・パラメータに従って生成されます。図 46 に示すように、8 点ランプ・プロファイルでは、最大 6 個の加速および減速セグメントと 1 つの定速セグメントを区別できます。

表 31. ポジション・モードのランプ・パラメータ

レジスタ・フィールド	概要
POSITION	ランプ・ジェネレータの最新のポジション値。ランプ・ジェネレータがイネーブルされている場合に、ポジション・コントローラの目標値として用いられます。FOC.PID_POSITION_ACTUALレジスタへの書込みアクセス後、POSITIONは、自動的にPID_POSITION_ACTUALの新しい値に上書きされます。
V_ACTUAL	ランプ・ジェネレータの読出し専用の最新速度値。ランプ・ジェネレータがイネーブルされている場合に、速度コントローラの目標値、あるいは、ポジション・コントローラへのフィードフォワード入力値として用いられます。
ACCELERATION	ランプ・ジェネレータの読出し専用の最新加速度値。オプションで、トルク・コントローラへのフィードフォワード入力として用いられます。
PID_POSITION_TARGET	FOC.PID_POSITION_TARGETレジスタの一部。 ランプ・ジェネレータの入力目標ポジション。FOC.PID_CONFIGレジスタのKEEP_POS_TARGETビットがセットされていない限り、FOC.PID_POSITION_ACTUALレジスタへの書込みアクセス後、PID_POSITION_TARGETがPID_POSITION_ACTUALの新しい値に上書きされます。
V_START	8点速度ランプの初期速度。モーションの開始時、または駆動方向が切り替わるとき（速度0に交差するとき）に用いられます。これは符号なしの値です。現在の速度の符号は、ポジション目標の相対的な方向によって決まります。
V1	ランプ・ジェネレータが、加速フェーズ時の場合A1とA2の間、減速フェーズ時の場合D2とD1の間で切り替わる際の速度値。V_STARTより大きいことが必要です。あるいは、A1/D1セクションが不要な場合、これをゼロに設定できます。これは符号なしの値です。現在の速度の符号は、ポジション目標の相対的な方向によって決まります。
V2	ランプ・ジェネレータが、加速フェーズ時の場合A2とA_MAXの間、減速フェーズ時の場合D_MAXとD2の間で切り替わる際の速度値。V1より大きいことが必要です。 あるいは、A2/D2セクションが不要な場合、これをV1と等しい値に設定できます。これは符号なしの値です。現在の速度の符号は、ポジション目標の相対的な方向によって決まります。
V_MAX	8点速度ランプの最大速度。V2より大きいことが必要です。これは符号なしの値です。現在の速度の符号は、ポジション目標の相対的な方向によって決まります。
V_STOP	停止直前の速度値。ポジション目標に到達したとき、または駆動方向が切り替わるとき（速度0に交差するとき）に用いられます。目標ポジションに到達したときに|V_ACTUAL|がV_STOPより小さい場合は、ランプ速度はV_STOPには設定されません。
A1	V_STARTとV1の間の加速度値。
A2	V1とV2の間の加速度値。
A_MAX	V2とVMAXの間の加速度値。
D1	V_STARTからV1への減速度値。
D2	V1からV2への減速度値。
D_MAX	V2からV_MAXへの減速度値。
T_ZEROWAIT	RAMPER.TIME_CONFIGレジスタの一部。 新たな動作を開始するまでT_ZEROWAIT × 8.5μsの間、速度を0に保持します。タイマーが起動するのは、目標ポジションに達した後、V_ACTUALがゼロに交差したとき、または、ハード／ソフト・ストップ・イベントの後です。
T_VMAX	RAMPER.TIME_CONFIGレジスタの一部。 V_MAXに達した後、または、ポジション目標が変更されたことによる加速または減速が行われる前（加速度の符号が変わった場合でも）、少なくともT_VMAX × 8.5μsの間、一定速度が保持されます。

特に指定のない限り、この表に示すレジスタ・フィールドは、RAMPER レジスタ・ブロックに属します。

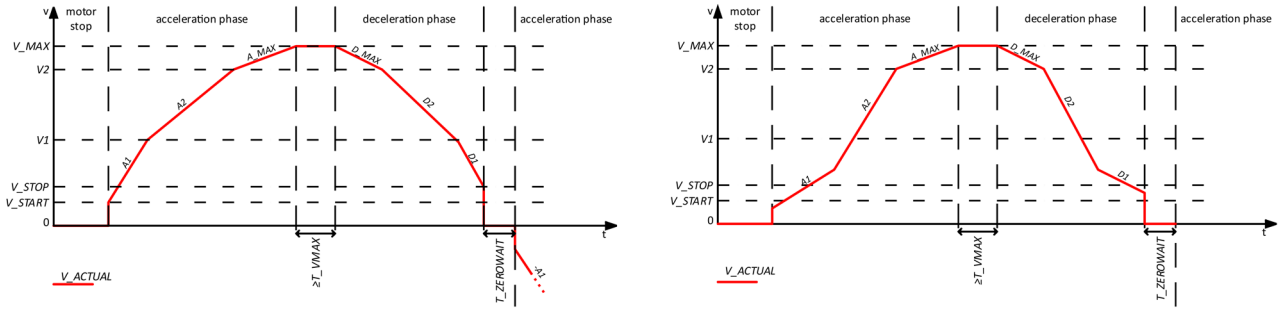


図 46. ポジショニング・モーション・プロファイル用の代表的な 8 点速度ランプ

ランプ・プロファイルの、定速セグメント（RAMPER.TIME_CONFIG レジスタの T_VMAX フィールドで設定）と減速フェーズに必要な距離が FOC.PID_POSITION_TARGET − RAMPER.POSITION の差で与えられる残りの距離より大きい場合、ランプの早期終了が発生します。この場合、修正ランプ・プロファイルが計算され、8 点ランプのすべてのセクションが実行されるとは限らなくなります。図 47 にその例を示します。

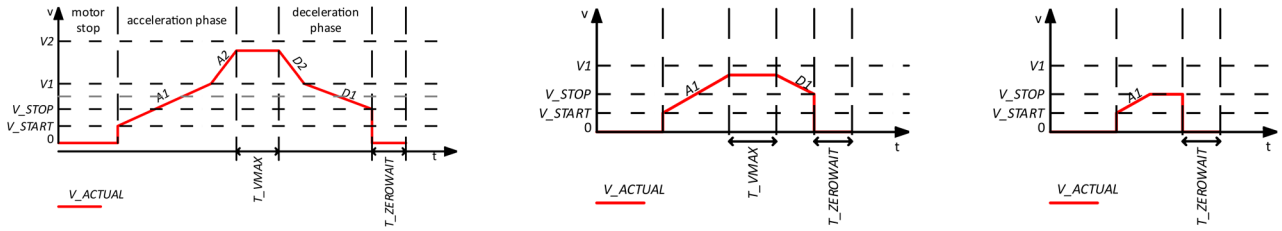


図 47. 修正されたランプ・ポジショニング・プロファイルの例

実行中のランプ・ポジショニング・プロファイルの間、FOC.PID_POSITION_TARGET の目標ポジションに変更が生じると、ランプの早期終了の原因になることがあります。図 48 に様々な状況を示します。

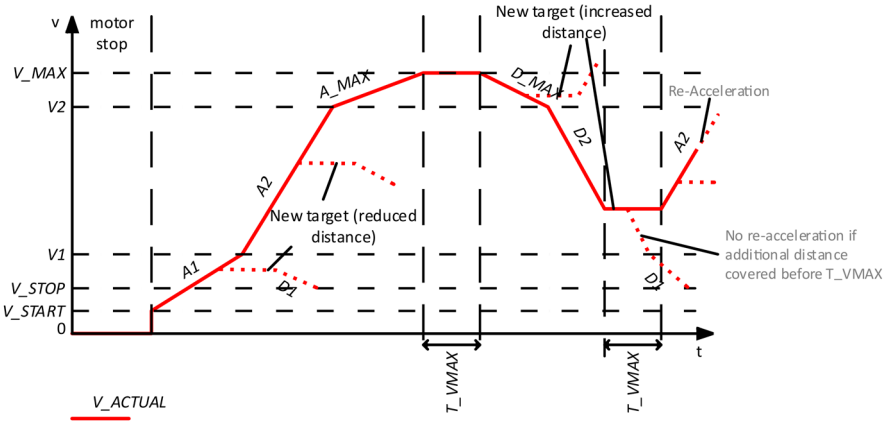


図 48. 目標ポジションの変更を原因としてランプ・ポジショニング・プロファイルに生じる可能性のある修正

ランプ速度モード

ランプ・ジェネレータは、MCC_CONFIG.MOTOR_MOTION レジスタの RAMP_MODE フィールドを 1_b に設定した場合、速度プロファイル用に設定されます。ユーザにより新たに定義された目標速度が FOC.PID_VELOCITY_TARGET レジスタを通じてコマンドされた場合、ランプ・ジェネレータは、現在の速度から設定された目標速度に達するための、速度モーション・プロファイルを計算します。

生成されたプロファイルからの最新の速度値は、RAMPER.V_ACTUAL レジスタから読み出せます。ランプ・ジェネレータがイネーブルされると、RAMPER.VELOCITY レジスタの値が速度コントローラの入力として用いられます。

速度プロファイルは、表 32 に示すように、定義されたランプ・パラメータに従って生成されます。ポジショニング・プロファイルとは対照的に、ランプ速度プロファイルには明確な減速フェーズがない点に注意してください。その代わり、プロファイルが加速しているか減速しているかは、現在の速度と比べた目標速度の大きさにのみ依存します。

ランプ速度プロファイルでは、最大3つの明確な加速／減速セグメントと1つの定速セグメントが生成されます。定速セグメントは、T_VMAX レジスタ・フィールドで定義されるセグメントのみを表します。

T_ZEROWAIT、V_START、V_STOP、V_MAX、D1、D2、D_MAX は、ランプ速度プロファイルには使用されません。

定義されたランプ速度パラメータ V1 および V2 と、それに対応する加速／減速フェーズは、たとえ単一の速度モーション・プロファイルの間であっても、両方の速度符号に対し適用される点に注意してください。例えば、現在の速度の値が、$+v_I \geq V2$ であり、$-v_I$ という新しい速度目標が設定された場合、$+v_I \rightarrow +V2 \rightarrow +V1 \rightarrow -V1 \rightarrow -V2 \rightarrow -v_I$ という速度プロファイル・シーケンスが生成されます。図 49 にこの挙動を示します。ここでは、2 つの速度ランプ・プロファイルが示されています。1 つは、速度 0 から PID_VELOCITY_TARGET へのプロファイル、もう 1 つは、+PID_VELOCITY_TARGET から-PID_VELOCITY_TARGET へのプロファイルです。また、T_VMAX ランプ・パラメータで定義される定速セグメントも示されています。

表 32. 速度モードのランプ・パラメータ

レジスタ・フィールド	概要
POSITION	ランプ・ジェネレータの最新のポジション値。ランプ・ジェネレータがイネーブルされている場合に、ポジション・コントローラの目標値として用いられます。FOC.PID_POSITION_ACTUALレジスタへの書込みアクセス後、POSITIONは、自動的にPID_POSITION_ACTUALの新しい値に上書きされます。
V_ACTUAL	ランプ・ジェネレータの読出し専用の最新速度値。ランプ・ジェネレータがイネーブルされている場合に、速度コントローラの目標値、あるいは、ポジション・コントローラへのフィードフォワード入力値として用いられます。
ACCELERATION	ランプ・ジェネレータの読出し専用の最新加速度値。オプションで、トルク・コントローラへのフィードフォワード入力として用いられます。
PID_POSITION_TARGET	FOC.PID_VELOCITY_TARGETレジスタの一部。 ランプ・ジェネレータの入力目標速度。
V1	ランプ・ジェネレータが加速／減速フェーズ時のA1とA2の間で切り替わる際の速度値。V_STARTより大きいことが必要です。あるいは、A1セクションが不要な場合、これをゼロに設定できます。これは符号なしの値です。現在の速度の符号は、速度目標の相対的な大きさによって決まります。
V2	ランプ・ジェネレータが加速／減速フェーズ時のA2とA_MAXの間で切り替わる際の速度値。V1より大きいことが必要です。あるいは、A2セクションが不要な場合、これをV1と等しい値に設定できます。これは符号なしの値です。現在の速度の符号は、速度目標の相対的な大きさによって決まります。
A1	ランプ・プロファイルを開始する前の速度とV1の間の加速度値。
A2	V1とV2の間の加速度値。
A_MAX	V2とPID_VELOCITY_TARGETの間の加速度値。
T_VMAX	RAMPER.TIME_CONFIGレジスタの一部。 PID_VELOCITY_TARGETに達した後、または、速度目標が変更されたことによる加速／減速が行われる前（加速度の符号が変わった場合でも）、少なくともT_VMAX × 8.5µsの間、一定速度が保持されます。

特に指定のない限り、この表に示すレジスタ・フィールドは、RAMPER レジスタ・ブロックに属します。

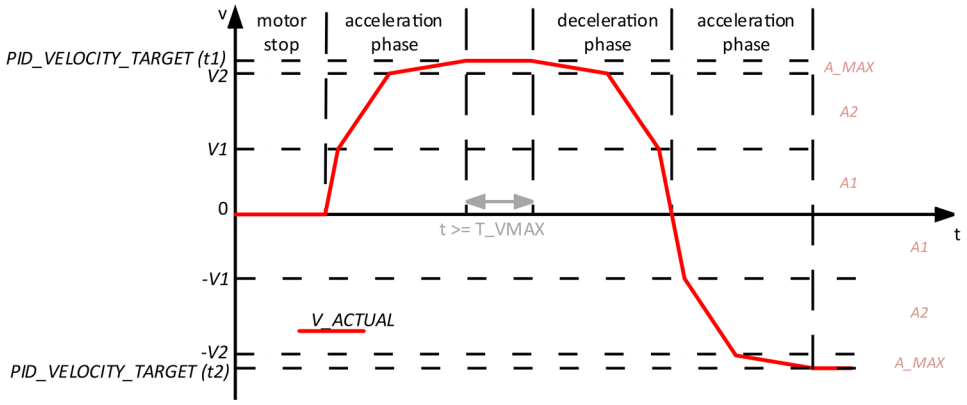


図 49. 代表的なランプ速度プロファイル

図 50 に示すように、実行中のランプ速度プロファイルの間、FOC.PID_VELOCITY_TARGET の目標速度に変更が生じると、ランプの早期終了の原因になることがあります。ここでは、T_VMAX で定義される定速セクションが用いられています。

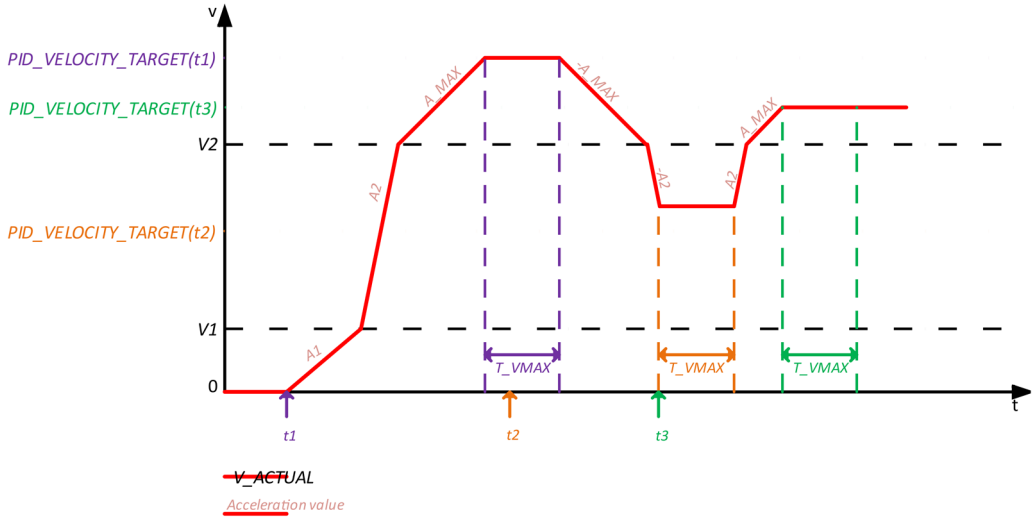


図 50. 目標速度の変更を原因としてランプ速度プロファイルに生じる可能性のある修正

オープン・ループでの現実の単位への変換

TMC6460 は、システム・クロック f_{SYS} のリファレンスとして、内部発振器または外部クロック信号を使用できます。ランプ・ジェネレータの速度および加速度のパラメータは、システム・クロックを基準にしています。そのため、ランプ・モーション・プロファイルにおいて最高の安定性と再現性を確保するために、外付けの水晶発振器を使用するか、マイクロコントローラから外部クロック信号を供給することを推奨します。

TMC6460 は、ランプ・ジェネレータの値に内部スケーリングを使用します。このセクションでは、内部値を現実の単位に変換するための数式を説明します。ここで行われる変換が有効となるのは、オープン・ループ・モードで動作している場合、つまり、ランプ・ジェネレータの phi_e が外部フィードバック・ソースの角度の代わりに使用されている場合（MCC_CONFIG.MOTOR_MOTION レジスタの RAMP_USE_PHI_E ビットがセットされている場合）のみです。

クローズド・ループ動作（外部角度フィードバック・ソースが用いられる場合）の現実の単位への変換については、現実の単位への変換のセクションを参照してください。

ポジションの変換

オープン・ループ動作用のポジションのスケーリングは、モーターの極ペアの数にのみ依存します。内部ポジションP_{int}から電気的および機械的ローター回転数P_eおよびP_mに変換するための式を、以下に示します。

$$P_e[erev] = \frac{P_{int}}{2^{16}}$$

$$P_m[rev] = \frac{P_{int}}{2^{16} \times N_{polepairs}}$$

この場合、P_{int}は、オープン・ループ・モードの場合に用いられるいずれかのポジション・レジスタの値を示しています。これには、次のレジスタが含まれますが、これに限定されるわけではありません:FOC.PID_POSITION_ACTUAL、FOC.PID_POSITION_TARGET、RAMPER.POSITION。

速度の変換

オープン・ループ・モードで動作している場合の内部速度値を RPM 単位のローター速度に変換するには、次式とスケーリング・ファクタk_{OL}を使用します。

$$v\,[rpm] = \frac{v_{int}}{k_{OL}}$$

オープン・ループ動作の速度スケーリング・ファクタk_{OL}は、システム・クロック周波数 f_{SYS} にのみ依存します。

$$k_{OL} = \frac{2^{16} \times 2^{24}}{f_{SYS} \times 60} \quad with\ f_{SYS} = 60MHz$$

この場合、v_{int} は、オープン・ループ・モードの場合に用いられるいずれかのポジション・レジスタの値を示しています。これには、次のレジスタが含まれますが、これに限定されるわけではありません : FOC.PID_VELOCITY_ACTUAL、FOC.PID_VELOCITY_TARGET、およびランプ・ジェネレータ・レジスタからの速度パラメータ

加速度の変換

速度およびポジションとは対照的に、加速度値はランプ・ジェネレータ・ブロックを通じてのみ用いられます。

オープン・ループ・モードで動作している場合の加速度変換は、速度スケーリング・ファクタk_{OL}およびシステム・クロック周波数 f_{SYS} にのみ依存します。内部加速度値a_{int}をローターの加速度（rev/s² 単位）に変換するには、次式を使用します。

$$a\left[\frac{rev}{s^2}\right] = a_{int} \times \frac{f_{SYS}}{k_{OL} \times 60 \times 2^{17}} \quad with\ f_{SYS} = 60MHz$$

この場合、a_{int}は、ランプ・ジェネレータ・レジスタからの加速度パラメータの値です。

リファレンス・スイッチ

TMC6460 は、リニア・モーターまたはリニア段で一般に用いられている、リファレンス・スイッチの検出機能をサポートしています。左スイッチ、右スイッチ、ホーム・スイッチの 3 つのリファレンス・スイッチをサポートしています。リファレンス・スイッチ入力として使用できるデバイスには、機械式スイッチ、フォト・インタラプタ、デジタル・ホール・センサーなどがあります。リファレンス・スイッチ入力は、現在のポジションをラッチしたり、ランプ・モーションを停止したりするために使用できます。この機能の設定は、RAMPER.SWITCH_MODE レジスタに割り当てられます。

該当するスイッチの極性を設定し、それに応じてプルアップ抵抗またはプルダウン抵抗の設定を選択することで、ノーマリ・オープンおよびノーマリ・クローズの両方のスイッチを使用できます。ノーマリ・クローズのスイッチは、スイッチ接続の割込みに対してフェイルセーフです。

ケーブルが長い場合、または信号にノイズが多い場合、TMC6460 のリファレンス入力付近に RC フィルタを追加する必要が生じることもあります。RC フィルタを追加すると、誤った配線によりロジック・レベル入力を破壊してしまうおそれも低減できますが、わずかな遅延が生じるため、この点をアプリケーションに関して考慮する必要があります。図 51 に、左および右のノーマリ・クローズ・スイッチおよび RC フィルタのセットアップ例を示します。

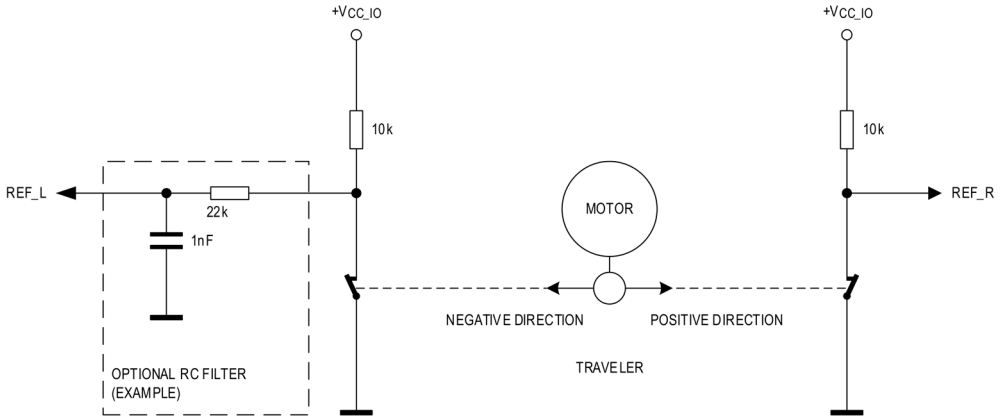


図 51. リファレンス・スイッチの使用例

ピン設定のセクションで説明したように、入力ピン信号のほとんどは、デジタル的に反転されフィルタ処理されます。これらの設定の他に、TMC6460 では、例えば、現在のポジションのラッチやモーションの自動停止など、リファレンス・スイッチ入力を使用するランプ・ジェネレータ・ブロックのすべての機能に対し、アクティブな極性を設定できます。これは、RAMPER.SWITCH_MODE レジスタの RAMP_REF_x_POL ビット（x＝H、L、R）を通じて設定できます。

更に、左および右のリファレンス・スイッチ入力は、RAMPER.SWITCH_MODE レジスタの SWAP_RAMP_REF_LR ビットを設定することにより、入れ替えることができます。

ポジションのラッチ

ランプ・ジェネレータ・ブロックは、いずれかのリファレンス・スイッチ入力の検出時に現在のポジションをラッチする機能を備えています。リファレンス・スイッチ入力の有効化へのエッジにおいてラッチするのか、非有効化へのエッジにおいてラッチするのかを設定することもできます。

この機能を使用するため、および設定するために、ポジション・ラッチをトリガするリファレンス・スイッチ入力に応じて、RAMPER.SWITCH_MODE レジスタの LATCH_REF_x_INACTIVE ビットまたは LATCH_REF_x_ACTIVE ビット（x＝H、L、R）のどちらかをセットします。有効化／非有効化エッジは、同じレジスタの RAMP_REF_x_POL ビットで設定されたスイッチ極性に従って選択されます。

ポジション・ラッチ・トリガが発生すると 2 つのポジション値がラッチされます。すなわち、RAMPER.POSITION レジスタの現在の値は、RAMPER.POSITION_LATCH レジスタに書き込まれ、FOC.PID_POSITION_ACTUAL レジスタの現在の値は、RAMPER.POSITION_ACTUAL_LATCH レジスタに書き込まれます。

RAMPER.EVENTS レジスタの LATCH_REF_x_READY ビット（x＝H、L、R）を用いると、リファレンス・スイッチ・イベントによってポジションがラッチされているかどうかをモニタできます。

停止モード

ランプ・モーションを停止するよう設定できるイベント・タイプが 3 種類あります。それは、リファレンス・スイッチ入力、ポジションまたは速度偏差、手動設定可能トリガです。停止動作は、速度ランプ・モードとポジション・ランプ・モードのどちらに対しても作用します。モーションの停止は、ランプ・ジェネレータの出力で行われる点に注意してください。そのため、この動作を作用させるには、ランプ・ジェネレータをイネーブルする必要があります。

偏差ストール停止信号を有効化してランプ・モーションを停止するには、RAMPER.SWITCH_MODE レジスタの STALL_STOP_EN ビットをセットし、更に、目的の動作に応じて STOP_ON_POS_DEVIATION ビットまたは STOP_ON_V_DEVIATION ビットをセットします。ストール停止信号が生成される偏差の値を設定するには、FOC.PID_POSITION_ERROR_MAX レジスタや FOC.PID_VELOCITY_ERROR_MAX レジスタに適切な値を設定します。偏差ストール停止信号のステータスは、RAMPER.STATUS レジスタの STALL_IN_POSITION_ERR ビットおよび STALL_IN_V_ERR ビットからクエリできます。

ストール停止イベントを手動でトリガするには、RAMPER.SWITCH_MODE レジスタの FORCE_HARD_STOP ビットをセットします。これが作用するためには、同じレジスタの STALL_STOP_EN ビットもセットする必要があります。

いずれかのストール停止イベントが発生すると、RAMPER.EVENTS レジスタの STALL_STOP_EVENT ビットがセットされます。

ランプ・モーションを停止するよう特定のリファレンス・スイッチ入力をイネーブルするには、対応する RAMPER.SWITCH_MODE レジスタの RAMP_REF_x_STOP_EN ビット（x＝H、L、R）をセットします。信号は、同じレジスタの RAMP_REF_x_POL ビットで設定されたアクティブなスイッチ極性に従って選択されます。停止信号のステータスは、RAMPER.STATUS レジスタの RAMP_REF_x_STOP_STATUS ビットからクエリできます。

ハード・ストップとソフト・ストップの比較

デフォルトでは、上記のすべての停止信号は、モーションのハード・ストップをトリガします。つまり、ランプ速度は直ちにゼロに設定されます。ハード・ストップは、RAMPER.TIME_CONFIG レジスタの T_ZEROWAIT フィールドに設定された待機時間を遵守して、ランプ・モーションを解放します。ハード・ストップは、リファレンス・スイッチの検出に適しています。また、モーションが比較的低速度に維持されている場合には常に適しています。しかし、システムの通常動作には適さない可能性があります。

ハード・ストップが適切でない状況では、この他にソフト・ストップ・モードが使用できます。ソフト・ストップ動作をイネーブルするには、RAMPER.STATUS レジスタの SOFT_STOP_EN ビットをセットします。ソフト・ストップ・シーケンスが有効化されると、このシーケンスは、リファレンス・スイッチ停止信号、ポジション／速度偏差信号でのストール、および手動停止トリガ用に用いられます。

ソフト・ストップ・シーケンスは、DMAX、D2、D1、V2、V1、VSTOP、T_ZEROWAIT の各減速ランプ設定値を用いて、ランプ・モーションを停止します。

ソフト・ストップ・モードを、特定のシステムの機械的制限値にあまりに近いリファレンス・スイッチと組み合わせて使用する場合には、注意が必要です。このような場合では、ソフト・ストップ・シーケンスがまだ進行中であっても、ローターまたはリニア段が機械的な制限値に達する可能性があります。それによって、特にランプ・ポジションとモーターの実際のポジションが完全には同期していない場合など、状況によっては、ポジション・コントローラがモーターを最大トルクで機械的制限値まで駆動することになりかねません。こうした状況を避けるために、機械的制限値に対し十分な距離を確保してリファレンス・スイッチを配置することを推奨します。

ホーミング・ルーチンの実行

リファレンス・スイッチ入力およびランプ・ポジション・モードを用いてホーミング・ルーチンを実行するには、以下のシーケンスを用います。

1. スイッチから短い距離を置くなどして、使用するスイッチが押されることのないようにします。
2. 使用するスイッチ・イベントや目的の極性が生じたときにポジションのラッチを有効化します。
3. 使用するスイッチのアクティブ時にモーター（ソフト）ストップを有効化します。
4. ランプ・モーションをスイッチのポジションの方向に開始します（左側スイッチはより負のポジションに移動させ、右側スイッチはより正のポジションに移動させます）。このモーションは、ランプ・ポジショニング・モーションを用いることでタイムアウトできます。
5. スイッチに接触すると直ちに、ポジションがラッチされモーターは停止を始めます。
6. V_ZERO ステータス・フラグをポーリングすることにより、モーターが停止状態になるまで待機します。
7. ラッチされたポジションと実際のポジションの差を計算します。
8. KEEP_POS_TARGET ビットがセットされていないことを確認します。
9. 計算した差を実際のポジション・レジスタに書き込みます。

これでホーミングは終了です。0 のポジションに移動すると、モーターは、スイッチに接触した点に正確に戻ります。

スイッチ・モード・レジスタの概要

表 33. RAMPER.SWITCH_MODE のビットフィールド

レジスタ・フィールド	備考
RAMP_REF_R_POL RAMP_REF_L_POL RAMP_REF_H_POL	これらのビットフィールドは、右、左、またはホーム・リファレンス・スイッチのアクティブな極性を定めます。 ▪ 0に設定すると、アクティブな極性は1になります。 ▪ 1に設定すると、アクティブな極性は0になります（反転）。
RAMP_REF_R_STOP_EN RAMP_REF_L_STOP_EN RAMP_REF_H_STOP_EN	これらは、以下の場合に、ビットフィールドがランプ・モーションの停止イベントをトリガできるようにします。 ▪ 右リファレンス・スイッチがアクティブで、実際の速度値が正（V_ACTUAL > 0）。 ▪ 左リファレンス・スイッチがアクティブで、実際の速度値が負（V_ACTUAL < 0）。 ▪ ホーム・リファレンス・スイッチがアクティブ。
STALL_STOP_EN	FORCE_HARD_STOP、STOP_ON_POS_DEVIATION、STOP_ON_V_DEVIATIONのいずれかがストール停止イベントをトリガできるようにするには、このビットをセットします。
FORCE_HARD_STOP	このビットをアクティブにすると、ストール停止イベントがトリガされます（STALL_STOP_ENがセットされている必要があります）。
STOP_ON_POS_DEVIATION	ポジション・コントローラの絶対ポジション誤差がFOC.PID_POSITION_ERROR_MAXを超えた場合にストール停止イベントをトリガします（STALL_STOP_ENがセットされている必要があります）。
STOP_ON_V_DEVIATION	速度コントローラの絶対速度誤差がFOC.PID_VELOCITY_ERROR_MAXを超えた場合にストール停止イベントをトリガします（STALL_STOP_ENがセットされている必要があります）。
SOFTSTOP_EN	このビットがクリアされている場合、トリガされた任意の停止イベントは、ランプ・モーションのハード・ストップを実行し、RAMPER.V_ACTUALを直ちに0にセットします。 トリガされた任意の停止イベントに対しソフト・ストップ動作を有効にするには、このビットをセットします。その場合、減速ランプ（定義されたすべてのフェーズRAMPER.D1、RAMPER.D2、RAMPER.D_MAXを含む）が実行されます。
LATCH_REF_R_ACTIVE LATCH_REF_L_ACTIVE LATCH_REF_H_ACTIVE	これらのビットのいずれかをセットすると、特定のリファレンス・スイッチ入力の有効化へのエッジで、RAMPER.POSITIONの値がRAMPER.POSITION_LATCHレジスタにラッチされ、FOC.PID_POSITION_ACTUALの値がRAMPER.POSITION_ACTUAL_LATCHレジスタにラッチされます。
LATCH_REF_R_INACTIVE LATCH_REF_L_INACTIVE LATCH_REF_H_INACTIVE	これらのビットのいずれかをセットすると、特定のリファレンス・スイッチ入力の非有効化へのエッジで、RAMPER.POSITIONの値がRAMPER.POSITION_LATCHレジスタにラッチされ、FOC.PID_POSITION_ACTUALの値がRAMPER.POSITION_ACTUAL_LATCHレジスタにラッチされます。
SWAP_RAMP_REF_LR	このビットがセットされている場合、左と右のリファレンス・スイッチの内部処理が交換可能になります。外部回路の変化を避けるのに有用です。

特に指定のない限り、この表に示すレジスタ・フィールドは、RAMPER.SWITCH_MODE レジスタに属します。

ランプ・ジェネレータのステータス・フラグおよびイベント

ランプ・ジェネレータ・ブロックは、リファレンス・スイッチに関連するものを含む、特定のランプ条件を示すいくつかのステータス・フラグを供給します。ステータス・フラグは、特定の条件の最新の状態を示すのに対し（読出し専用）、イベント信号は、特定の条件が発生したことを示します。特定の状態がアクティブである限り、イベント・ビットはアクティブなままであり、対応するビットは手動ではクリアできません。表 34 に、すべてのステータス・フラグとそのビット・ポジションを示します。これらは RAMPER.STATUS レジスタにあります。表 35 には、イベント・ビットを示します。これらのビットは、RAMPER.EVENTS レジスタにあります。一部のステータス・ビットは、CHIP.STATUS_FLAGS レジスタに転送されます。詳細については、ランプ・ジェネレータのステータス・フラグおよびイベントのセクションを参照してください。

表 34. RAMPER.STATUS のビットフィールド

フラグ名	ビット番号	アクセス	説明
RAMP_REF_L_STATUS	0	R	左リファレンス・スイッチのステータス：0 = 非アクティブ、1 = アクティブ
RAMP_REF_R_STATUS	1	R	右リファレンス・スイッチのステータス：0 = 非アクティブ、1 = アクティブ
RAMP_REF_H_STATUS	2	R	ホーム・リファレンス・スイッチのステータス：0 = 非アクティブ、1 = アクティブ
RAMP_REF_L_STOP_STATUS	6	R	アクティブな左停止スイッチにより、停止条件を発信。*
RAMP_REF_R_STOP_STATUS	7	R	アクティブな右停止スイッチにより、停止条件を発信。*
RAMP_REF_H_STOP_STATUS	8	R	アクティブなホーム・リファレンス・スイッチにより、停止条件を発信。*
V_REACHED_STATUS	11	R	最大／目標速度が最新のランプ速度値に等しくなっています。ランプ・ポジション・モード： V_ACTUAL = RAMPER.V_MAX ランプ速度モード： V_ACTUAL = FOC.PID_VELOCITY_TARGET
POSITION_REACHED_STATUS	12	R	最新のランプ速度が0に等しく、目標ポジションが最新のランプ・ポジションに等しくなっています。つまり、RAMPER.POSITION = FOC.PID_POSTION_TARGETです。 このビットは、CHIP.STATUS_FLAGSレジスタに転送されます。
V_ZERO	13	R	最新のランプ速度値RAMPER.V_ACTUALが0に等しくなっています。
T_ZEROWAIT_ACTIVE	14	R	ランプ・モーションの停止後、T_ZEROWAITフェーズがアクティブです。この時間、ランプ・モーションは停止状態です。
STALL_IN_V_ERR	16	R	最大速度差がFOC.PID_VELOCITY_ERROR_MAXの値を超えたため、ランプが停止しています。
STALL_IN_POSITION_ERR	17	R	最大ポジション差がFOC.PID_POSITION_ERROR_MAXの値を超えたため、ランプが停止しています。

*停止条件およびその割込みは、反対方向への移動をコマンドすることによりクリアできます。ソフト・ストップ・モードが有効になっている場合、この条件はモーターがリファレンス・スイッチ方向へのモーションを停止するまでアクティブのままとなります。リファレンス・スイッチまたはその停止機能を無効化することでもこの割込みをクリアできますが、モーターはモーションを続けます。

表 35. RAMPER.EVENTS のビットフィールド

フラグ名	ビット番号	アクセス	説明
POSITION_REACHED_EVENT	0	R/W1C	ランプ・モーションは停止し、目標ポジションに達しています。つまり、RAMPER.POSITION = FOC.PID_POSTION_TARGETです。
STALL_STOP_EVENT	1	R/W1C	速度またはポジションのトラッキング・エラーによりストール停止イベントがアクティブであることを発信します。このビットフィールドに「1」を書き込むと停止条件がクリアされ、モーターはモーションを再開できます（ビットおよび割込み条件は「1」を書き込むとクリアされます）。
SECOND_MOVE_EVENT	2	R/W1C	目標値が更新されたなどの理由で、ランプ・ジェネレータが反対方向への新しいランプ・モーションの移動を計算しました。
LATCH_REF_H_READY	3	R/W1C	ポジションのラッチは、トリガされたホーム・リファレンス・スイッチによって実行されます。
LATCH_REF_L_READY	4	R/W1C	ポジションのラッチは、トリガされた左リファレンス・スイッチによって実行されます。
LATCH_REF_R_READY	5	R/W1C	ポジションのラッチは、トリガされた右リファレンス・スイッチによって実行されます。

この表に示されているビットは W1C アクセスです。つまり、クリアするには「1」を書き込む必要があります。

プログラマブル I/O コントローラ

TMC6460 は、汎用性のあるプログラマブルな I/O コントローラを内蔵しています。そのアーキテクチャおよびコマンド・セットは、シリアル・データをパラレル・データに、および、パラレル・データをシリアル・データに変換するという、特定の目的のために最適化されています。したがって、同期および非同期のビット・ストリームをサポートしています。

ピンの再割り当てのセクションで説明したように、ほとんどの I/O ピンは、I/O コントローラに関連する代替機能に割り当てることができます。TMC6460 の最大 3 つの出力ピンと 2 つの入力ピンは、直接 I/O コントローラを制御したり I/O コントローラとインターフェースしたりできます。I/O コントローラの 4 つ目の出力は、CHIP.STATUS_FLAGS レジスタの IO_CONTROLLER_STATUS ビットに転送されます。このビットは、I/O コントローラのリセット・フラグと組み合わせられるため、I/O コントローラがリセット状態にない場合は、4 つ目の出力は、IO_CONTROLLER_STATUS ビットに反映されるだけです。一部の内部同期信号（PWM_ZERO など）も、I/O コントローラへの入力として割り当てることができます。

更に、SPI または UART を通じて外部ホストから任意のレジスタにアクセスするのと並行して、I/O コントローラも完全なレジスタ・マップにアクセスできます。コントローラは、レジスタ値を取得したり操作したりすることを含むタスクを低レイテンシで周期的にトリガされる信号でサポートできるだけの汎用性を備えています。

TMC6460 の I/O コントローラには、TMC6460 の機能を拡張する基本的な機能セットを備えた ROM コードが内蔵されています。これには、SPI/SSI エンコーダとのインターフェース、外部 EEPROM からのプログラムのロード、バイクワッド・フィルタによるデータのフィルタ処理などがあります。

I/O コントローラは、稼働しているプログラム内の特定のアドレスを用いることによって制御される、様々なアクセラレータ・モジュールを内蔵しています。使用できるアクセラレータ・モジュールは次のとおりです。

- I^2C コントローラ
- 符号なし 32 ビットを 16 ビットで除算する除算器
- 符号付き 16 ビット乗算器
- 符号付きバイクワッド・フィルタ

このデータシートでは、アクセラレータ・モジュールの説明を、内蔵の ROM コード機能のコンテキストにおいてのみ行っています。

1024 × 16 ビット RAM もユーザ・プログラミング用に使用できます。RAM のプログラミングおよび読出しは、内蔵の ROM コード機能を通じて可能です。I/O コントローラのプログラミングはオプションであり、ROM コードによって提供されているもの以外の機能が必要な場合にのみ、必要とされます。

特定の RAM アクセスの更に詳細な説明や、アクセラレータ・モジュール要求、および使用可能なすべてのオペコードは、追加されるアプリケーション・ノートに記載されています。

ROM コード機能

TMC6460 は、カスタム・プログラムを必要とせず、I/O コントローラを用いていくつかの機能を提供します。これらは ROM コード機能と呼ばれます。IO_CONTROLLER.COMMAND レジスタは、メインのインターフェースであり、これを通じて様々な機能が実行されます。一部の ROM コード機能は単一コマンドとして実行されますが、繰り返し実行する機能もあります。一度に実行できる機能は 1 つのみです。表 36 に、使用可能な ROM コード機能の概要を示します。

表 36. ROM コード機能の概要

コマンド	機能	単一（S）／繰り返し（R）実行
GET_VER	ROMコードのバージョンを返す。	S
I2C_INIT_PP	I²Cプロトコルをプッシュプル出力で初期化。	S
I2C_INIT_OD	I²Cプロトコルを初期化。オープン・ドレイン出力。	S
I2C_READ	I²Cクライアントから読出し。	S
I2C_WRITE	I²Cクライアントへ書込み。	S
I2C_EEPROM	インターフェースされているI²C EEPROMの内容をRAMに読出し。	S
RAM_READ	内蔵RAMから1つのアドレスを読出し。	S
RAM_WRITE	内蔵RAMの1つのアドレスを書込み。	S
RAM_EXE	RAMコード実行へ切り替え。	S
FILTER_SETUP	内蔵バイクワッド・フィルタのセットアップ。	S
CALC_FILT	以下の1つまたはすべてを実行： 符号付き乗算、符号なし除算、符号付きバイクワッド・フィルタ処理	S
DEBUG	最小値、最大値、フィルタ処理された値を取得することにより、特定のレジスタをデバッグ。	R
PWM_INPUT	PWM入力ピンの処理を実行し、計算された値を定められたレジスタに格納。	R
AIN_INPUT	AIN入力ピンの処理を実行し、計算された値を定められたレジスタに格納。	R
SPI_ENC	SPIメイン・インターフェースをセットアップして、インターフェースされたSPIエンコーダに単一の読出しまたは連続的な読出しで通信を行い、オプションで、フィードバック・エンジンと併用するために、受信した値を格納。 対応するデータグラム構造を持つSSIエンコーダも、この機能を通じてインターフェースできます。	S/R

単一実行コマンドは、終了コマンドを付加する必要がありませんが、繰り返し実行コマンドは、「アイドル」コマンドを IO_CONTROLLER.COMMAND = 0x00000000 と設定して、終了する必要があります。

一般的に、IO_CONTROLLER.RESPONSE_0 レジスタは、コマンドが成功裏に実行されると直ちに、要求された IO_CONTROLLER.COMMAND と等しい値に設定されます。繰り返し実行コマンドには、ある程度のわずかな変動が発生します。これについては、特定のコマンドのセクションで説明します。

注意すべき重要な点は、新しいコマンド実行を開始するために必要な条件です。IO_CONTROLLER.COMMAND レジスタの値によって定義されるコマンドが実行されるのは、コマンドの値が IO_CONTROLLER.RESPONSE_0 レジスタの値と異なる場合のみです。そのため、2 つの同じコマンドを順次実行する必要がある場合、「アイドル」コマンド（0x00000000）を実行と実行の間にセットする必要があります。これが機能する理由は、IO_CONTROLLER.COMMAND = 0x00000000 と設定すると IO_CONTROLLER.RESPONSE_0 = 0x00000000 になるためです。

IO_CONTROLLER_STATUS ビットに関連する 4 つ目の I/O コントローラ出力は、I/O コントローラによって、内部の IO_CONTROLLER.RESPONSE_0 ステータス・ビットに割り当てられます。IO_CONTROLLER.RESPONSE_0 が IO_CONTROLLER.COMMAND レジスタに等しくなる（これはコマンドが終了したことを意味します）と直ちに、このステータス・ビットが 1 にセットされます。したがって、FAULTN ピンを用いて要求されたコマンドの終了に関する情報を得ることで、IO_CONTROLLER.RESPONSE_0 レジスタを永続的にポーリングするのを避けることができます。

使用する ROM コード機能に応じて、コマンド自体以外の追加パラメータが必要になる場合もあります。そのためには、IO_CONTROLLER.RESPONSE_x レジスタ（x = 1、2、3）の一部または全部を使用する必要があります。詳細は、特定のコマンドのセクションでも説明します。

ROM コード機能のコマンドを説明するために以下の表記が用いられます。小文字で書かれた文字は特定のレジスタの 1 バイトを表し、その値は目的とする設定に合わせて選択する必要があります。大文字で書かれた文字および整数は、コマンドが機能するために必要な、事前に定められた 16 進数値を表します。

GET_VER

表 37. GET_VER のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xAA000000			
2			RESPONSE_0	0xAA000000	コマンド実行後直ちに。
3			RESPONSE_1	0x646000B3	ROMのバージョン番号。

説明：現在の ROM バージョンを返します。

I2C_INIT_PP

表 38. I2C_INIT_PP のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xAA01yyyy			0xyyyy：ボー・レートの設定。
2			RESPONSE_0	0xAA01yyyy	コマンド実行後直ちに。

説明：

16 ビットの「yyyy」で定義された特定の I^2C クロック周波数で、I^2C インターフェースを初期化します。クロック周波数は、$f_{SYS}/4/(yyyy + 1)$ で計算されます。例えば、yyyy = 0x95 とすると、f_{SYS} = 60MHz の場合、f_{I2C} = 100kHz のクロック周波数になります。I^2C の SDA ピンおよび SCK ピンは、フィードバック・エンジン・ピン HALL_U および HALL_V の代替ピン機能としてのみ、使用できます。

I^2C プロトコル用に期待されるとおり、SCL ピンは、プッシュプルとして設定され、SDA は、ACK および読出しフェーズの間、オープン・ドレインとして設定されます。

このコマンドまたは I2C_INIT_OD のコマンドのどちらかを実行してから、EEPROM 読出し／書込みコマンドを使用する必要があります。

I2C_INIT_OD

表 39. I2C_INIT_OD のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xAA02yyyy			0xyyyy：ボー・レートの設定。
2			RESPONSE_0	0xAA02yyyy	コマンド実行後直ちに。

説明：

16 ビットの「yyyy」で定義された特定の I^2C クロック周波数で、I^2C インターフェースを初期化します。クロック周波数は、$f_{SYS}/4/(yyyy + 1)$ で計算されます。例えば、yyyy = 0x95 とすると、f_{SYS} = 60MHz の場合、f_{I2C} = 100kHz のクロック周波数になります。I^2C の SDA ピンおよび SCK ピンは、フィードバック・エンジン・ピン HALL_U および HALL_V の代替ピン機能としてのみ、使用できます。

SCL はプッシュプルとして設定されません。代わりに、SCL は強い 0 と弱い 1 で駆動されます。それによって、接続されている I^2C クライアントのクロック・ストレッチング動作を可能にできます。I^2C プロトコル用に期待されるとおり、SDA は、ACK および読出しフェーズの間、オープン・ドレインとして設定されます。

このコマンドまたは I2C_INIT_PP のコマンドのどちらかを実行してから、EEPROM 読出し／書込みコマンドを使用する必要があります。

I2C_READ

表 40. I2C_READ のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xDyyy0000			0xyyy：I^2Cクライアント用の読出しアドレス。
2			RESPONSE_0	0xDyyy0000	コマンド実行後直ちに。
3			RESPONSE_1	0x0000zzzz	I^2Cクライアントからの読出し応答。

説明：

接続されている I²C クライアントからの単一の読出しアクセスを実行します。「yyy」は指定された 12 ビットのアドレスを表します。

16 ビットの「zzzz」は、クライアントの応答を示すもので、IO_CONTROLLER.RESPONSE_1 レジスタに格納されており、コマンドが実行された後に読み出されます。

I²C インターフェースを I2C_INIT_OD 機能または I2C_INIT_PP 機能で初期化してから、このコマンドを発行する必要があります。

I2C_WRITE

表 41. I2C_WRITE のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xEyyyzzzz			0xyyy：書込みアドレス、 0xzzzz：書込みデータ。
2			RESPONSE_0	0xEyyyzzzz	コマンド実行後直ちに。

説明：

関連する I²C クライアントに単一の書込みアクセスを実行します。16 ビットの「zzzz」は、指定された「yyy」ビットの 12 ビットのアドレスに書き込まれる値を表します。

I²C インターフェースを I2C_INIT_OD 機能または I2C_INIT_PP 機能で初期化してから、このコマンドを発行する必要があります。

I2C_EEPROM

表 42. I2C_EEPROM のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xAA030000			
2			RESPONSE_0	0xAA030000	コマンド実行後直ちに。

説明：

接続されている EEPROM の RAM への完全な読出しを I²C を通じて実行します。

I²C インターフェースを I2C_INIT_OD 機能または I2C_INIT_PP 機能で初期化してから、このコマンドを発行する必要があります。

RAM_READ

表 43. RAM_READ のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xCyyy0000			0xyyy：RAMアドレス
2			RESPONSE_0	0xCyyy0000	コマンド実行後直ちに。
3			RESPONSE_1	0x0000zzzz	アドレスが0xyyyのRAMからの値の読出し。

説明：

内部 RAM からの単一の読出しアクセスを実行します。「yyy」には指定された 10 ビットのアドレスが格納されます。「yyy」の上位 2 ビットは 0 に設定する必要があります。

16 ビットの「zzzz」は、RAM の内容を表すもので、IO_CONTROLLER.RESPONSE_1 レジスタに格納されており、コマンドが実行された後に読み出されます。

RAM_WRITE

表 44. RAM_WRITE のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xByyyzzzz			0xyyy：RAM書込みアドレス、0xzzzz：書込みデータ。
2			RESPONSE_0	0xByyyzzzz	コマンド実行後直ちに。

説明：

内部 RAM への単一の書込みアクセスを実行します。16 ビットの「zzzz」は、指定された「yyy」ビットの 10 ビットに書き込まれる値を表します。「yyy」の上位 2 ビットは 0 に設定する必要があります。

RAM_EXE

表 45. RAM_EXE のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0xFFFFFFFF			
2			RESPONSE_0	0xFFFFFFFF	コマンド実行後直ちに、RAMコードの実行の直前。

説明：

ROM コードの実行から RAM コードの実行への遷移を実行します。それ以上の I/O コントローラのアクションは、RAM のオペコードで定義されます。

FILTER_SETUP

表 46. FILTER_SETUP のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	RESPONSE_1	0xzzpppppp			0xpppppp：係数b0、0xzz：a2の一部
2	RESPONSE_2	0xyyssssss			0xssssss：係数b1、0xyy：a2の一部
3	RESPONSE_3	0xwwtttttt			0xtttttt：係数b2、0xww：a2の一部
4	COMMAND	0x70uuuuuu			0xuuuuuu：係数a1
5			RESPONSE_0	0x70uuuuuu	コマンド実行後直ちに。

説明：

I/O コントローラに接続されるバイクワッド・フィルタをその全 24 ビット・パラメータで設定します（バイクワッドのパラメータ化の詳細については、バイクワッド・フィルタのセクションを参照してください）。

- 係数 b0 [23:0] = pppppp
- 係数 b1 [23:0] = ssssss
- 係数 b2 [23:0] = tttttt
- 係数 a1 [23:0] = uuuuuu
- 係数 a2 [23:0] = wwyyzz

係数 b0、b1、b2、a2 は、コマンドを実行する前に設定する必要があります。そのため、係数 a1 を最後に設定しなくてはなりません。

CALC_FILT

表 47. CALC_FILT のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	RESPONSE_1	0xzzzzzzzz			分子（u32）
2	RESPONSE_2	0xsssswwww			0xsssssと0xwwww：乗数（s16）
3	RESPONSE_3	0xyyyyyyyy			バイクワッド入力（s32）
4	COMMAND	0x7100uuuu			0xuuuu：除数（u16）
5			RESPONSE_0	0x7100uuuu	コマンド実行後直ちに。
6			RESPONSE_1	0xqqqqqqqq	符号なしの除算結果（u32）
			RESPONSE_2	0xpppppppp	符号付きの乗算結果（s32）
			RESPONSE_3	0xtttttttt	バイクワッド・フィルタ出力（s32）

説明：

I/O コントローラのアクセラレータ・モジュールを使用して 3 種類の計算を実行します。

- 符号なしの除算（分子[31:0]/除数[15:0] = 商[31:0]）
  - 計算： 0xzzzzzzzz/0x0000uuuu = 0xqqqqqqqq
  - RESPONSE_1 に除算結果が格納されます。
  - すべての数値に符号が付きません。
- 符号付き乗算（乗数[15:0] × 乗数[15:0] = 積[31:0]）
  - 計算： 0xssss × 0xwwww = 0xpppppppp
  - RESPONSE_2 に乗算結果が格納されます。
  - すべての数値が符号付きです。
- バイクワッド・フィルタ。
  - 0xyyyyyyyy は、バイクワッド・フィルタの符号付き入力です。
  - 0xttttttt（RESPONSE_3）にバイクワッド・フィルタの出力が格納されます。
  - すべての数値が符号付きです。
  - バイクワッド・フィルタは、FILTER_SETUP コマンドを用いて、事前にセットアップしておく必要があります。
  - 詳細については、バイクワッド・フィルタのセクションを参照してください。

すべての計算は並行して行われます。目的の計算に必要なパラメータのみが必要です。

DEBUG

表 48. DEBUG のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	COMMAND	0x6yyyzwuu			0xyyy：アドレス、0xz：符号付き／符号なし 0xw：データ・マスク、0xuu：トリガ・セットアップ
2			RESPONSE_0	0x55yyzwuu	計算中、応答データは有効ではありません。
3			RESPONSE_0	0x6yyyzwuu	計算フェーズ後、応答データは有効です。
			RESPONSE_1	0xqqqqqqqq	デバッグ開始後の最大値（u32）。
			RESPONSE_2	0xpppppppp	中間（s32）
			RESPONSE_3	0xttttttt	デバッグ開始後の最小値（s32）。
4	COMMAND	0x00000000			デバッグ機能の終了。

説明：

特定のレジスタを繰り返しデバッグします。新しいトリガごとに、新しい計算フェーズが実行されます。各計算フェーズで、定義されたデバッグ・レジスタの値が読み込まれ、以下についてチェックされます。

- 格納される正の最大値（IO_CONTROLLER.RESPONSE_1 = 0xpppppppp）
- 格納される負の最小値（IO_CONTROLLER.RESPONSE_3 = 0xnnnnnnnn）

IO_CONTROLLER.RESPONSE_1 および IO_CONTROLLER.REPONSE_3 の値は、それぞれ、正の値および負の値にのみ関連する点に注意してください。例えば、デバッグされたレジスタについて負の値のみが取得された場合、IO_CONTROLLER.RESPONSE_1 には、絶対値が最小の負の値ではなく、ゼロの値が格納されます。反対に、正の値のみが取得された場合は、IO_CONTROLLER.RESPONSE_3 には、正の最小値ではなく-1 の値が格納されます。

IO_CONTROLLER.RESPONSE_2 にはフィルタ処理された値が格納されます。新しい値ごとに、接続されたバイクワッド・フィルタでフィルタ処理されます。そのため、FILTER_SETUP コマンドを用いて、DEBUG の開始前にフィルタをセットアップしておく必要があります。

デバッグされるレジスタは、「yyy」ビットの 10 ビットのアドレスで定義されます。「yyy」の上位 2 ビットは 0 に設定する必要があります。「z」ビットは、値が符号付き（= 1）か符号なし（= 0）かを定めます。

「w」ビットは、デバッグされるレジスタのマスク値を定めます。

- このバイトの各ビットは、デバッグされるレジスタの保持すべきバイトを表します。
- 例えば、
  - $w = 0011_b$ は、デバッグされるレジスタの下位 2 バイト（レジスタ・ビット[15:0]）を処理します。
  - $w = 0110_b$ は、デバッグされるレジスタのレジスタ・ビット[23:8]を処理します。
  - $w = 1111_b$ は、デバッグされるレジスタの全レジスタ・ビット[31:0]を処理します。

「uu」のビットは、各アクイジション／計算フェーズのトリガを定義します。

- 新しいアクイジション／計算フェーズごとに、定められた信号でトリガされます。
- 信号の 0 から 1 への変化が検出されると、新しいアクイジションおよび計算サイクルが処理されます。
- 処理の間、IO_CONTROLLER.RESPONSE_0 の最上位バイトが 0x55 に変化し、応答データが有効でないことを示します。
- 処理の終了後、IO_CONTROLLER.RESPONSE_0 の最上位バイトが 0x6y に変化し、応答データが有効であることを示します。

表 49 に、デバッグ・トリガに使用できる信号を示します。複数のトリガ信号を選択できる点に注意してください。ただし、トリガを見落とさないようにするため、選択する信号は最大で 1 つにすることを推奨します。

表 49. ROM コードの DEBUG：トリガ・オプション

“uu”	TRIGGER	“uu”	TRIGGER
0x01	PWM_IN input signal	0x10	ADC_I_VALID: Internal CSA calculations done.
0x02	DIR_IN input signal	0x20	ADC_EXT_VALID: External ADC calculations done.
0x04	Internal PWM_ZERO signal	0x40	FOC_VALID: FOC calculations done.
0x08	Internal PWM_CENTER signal	0x80	Internal BRAKE signal

「uu」が 0x00 の場合、PWM_CENTER 信号はトリガ信号として定義されます。

デバッグを終了するには、IO_CONTROLLER.COMMAND = 0x00000000 の「アイドル」コマンドを送信します。応答データは、新しいコマンドが発行されない限り、保持されます。

PWM_INPUT

表 50. PWM_INPUT のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	RESPONSE_1	0xttttzzzz			デューティサイクルの正規化された制限値（u16）。
2	RESPONSE_2	0xuuuuuuuu			最終出力値の下限値（s32）。
3	RESPONSE_3	0xvvvvvvvv			最終出力値の上限値（s32）。
4	COMMAND	0x9yyymsww			0xyyy：目標アドレス、 0xm：バイト・マスク、0xs：バイト・シフト、 0xww：サンプル・レート遅延。
5			RESPONSE_0	0x55yymsww	計算中、応答データは有効ではありません。
6			RESPONSE_0	0x9yyymsww	計算フェーズ後、応答データは有効です。
			RESPONSE_1	0xnnnnnnnn	最後の目標値（s32）。
			RESPONSE_2	0xqqqqpppp	0xqqqq：最後にフィルタ処理されたデューティサイクル値、 0xpppp：最後の相対デューティサイクル値。
			RESPONSE_3	0xkkkkhhhh	0xkkkk：PWM = 1カウント。 0xhhhh：PWM = 0カウント。
7	COMMAND	0x00000000			PWM_INPUT機能の終了。

説明：

PWM_IN 入力とオプションで DIR_IN 入力を繰り返し処理し、それらを任意の目標レジスタの制御信号として使用します。この処理中、目標レジスタは、応答レジスタの有効性とは関係なく、新しい値に繰り返し書き込まれます。応答レジスタは、デバッグ目的でのみ使用されます。

PWM_IN および DIR_IN 入力信号のオプションのデジタル入力フィルタや信号反転は、このコマンドを発行する前に設定する必要があります。詳細については、ピン設定およびピンの再割り当てのセクションを参照してください。

このコマンドに対し様々なタイプのデューティサイクルが計算されます。デューティサイクルは、0%～100%のパーセンテージに対応する、16 ビットの値で表されます。入力 PWM 信号は、0%または 100%のデューティサイクルにはできない点に注意してください。異なる計算ステップを開始するには、1→0 および 0→1 の遷移のエッジが必須であるためです。入力のデューティサイクルを正規化できるため、処理されるデューティサイクル値は、実際にはいずれかの制限値に達することができます。

デューティサイクルは 16 ビットの値で表されるため、現実のデューティサイクルのパーセンテージは、内部値を 65536 で除算することによって得られます。例えば、8192 の内部デューティサイクル値は、8192/65536 × 100% = 12.5%に対応します。

入力 PWM 信号の基本サンプリング・レートは、$f_{SYS}/30 = 2MHz$ です。この周波数は、必要に応じて減少することもできます。コマンド・データグラムの「ww」ビットは、sampling_rate = 2MHz/(ww + 1)の式を用いる基本サンプリング・レートの除数を定義します。例えば、ww = 4 の場合、サンプリング・レートは 400kHz に減少しますが、ww = 0 の場合は、2MHz の最大サンプリング・レートになります。

入力 PWM 信号の周期が $1/2MHz \times 2^{16} = 32.768ms$（30.52Hz に相当）より大きい場合、サンプリング・レートを遅くする必要が生じる可能性があります。これは、入力カウンタが 16 ビット値に制限されている場合に該当します。

ほとんどの計算が、入力 PWM 信号の 1→0 の遷移のエッジ後に実行されるため、この入力のローのフェーズは、すべての PWM_IN デューティサイクルを正しく処理できるよう、少なくとも 6µs の長さであることが必要です。

計算フェーズの間、IO_CONTROLLER.RESPONSE_0 の最上位バイトが 0x55 に設定され、応答データが有効でないことを示します。すべての計算ステップが実行され、目標の割り当てが終了した後、その同じバイトが再度 0x9y に設定されて、有効な応答レジスタであることを示します。その後、応答レジスタには次のデータが格納されます。

- IO_CONTROLLER.RESPONSE_1 = 0xnnnnnnnn：最後の目標割り当て値。
- IO_CONTROLLER.RESPONSE_2[15:0] = 0xpppp：最後にモニタされた相対デューティサイクル値 $DC_{REL} \times 2^{16}$。
- IO_CONTROLLER.RESPONSE_2[31:16] = 0xqqqq：最後にフィルタ処理されたデューティサイクル値 $DC_{FILT} \times 2^{16}$。
- IO_CONTROLLER.RESPONSE_3[15:0] = 0xhhhh：最後にモニタされた PWM_IN = 1 のカウント値。
- IO_CONTROLLER.RESPONSE_3[31:16] = 0xkkkk：最後にモニタされた PWM_IN = 0 のカウント値。

PWM 入力処理を終了するには、IO_CONTROLLER.COMMAND = 0x00000000 の「アイドル」コマンドを送信します。応答データは、新しいコマンドが発行されない限り、保持されます。

基本的なアクイジションおよび計算のシーケンスを、以下に説明します。PWM_IN = 0→1 のエッジ後、相対デューティサイクルは次式で計算されます。

$$DC_{REL} = \frac{count\{PWM_IN=1\}}{count\{PWM_IN=1\}+count\{PWM_IN=0\}}$$

PWM_IN = 1→0 のエッジ後、DIR_IN の極性が格納され、残りの計算がすべて実行されて、PWM_IN 処理の目標値の割り当てが開始されます。これを以下のプロセスで説明します。

1. デューティサイクル制限値に従って相対デューティサイクル DCREL を正規化します。
   - COMMAND が発行されると、IO_CONTROLLER.RESPONSE_1 が 16 ビットの制限値を設定します。これは、正規化計算の最小および最大の相対デューティサイクル値を定めます。
   - 最小相対デューティサイクル DC_{MIN} = 0xzzzz/65536（RESPONSE_1[15:0]で定義）：
     - 相対デューティサイクルがこの制限値未満の場合、正規化デューティサイクル DC_{NORM} は 0 に設定されます。
   - 最大相対デューティサイクル DC_{MAX} = 0xtttt/65536（RESPONSE_1[31:16]で定義）：
     - 相対デューティサイクルがこの制限値を超えた場合、正規化デューティサイクル DC_{NORM} は 1 に設定されます。
   - DC_{REL} が DC_{MIN} と DC_{MAX} の間にある場合、正規化デューティサイクル DC_{NORM} は、これらの制限値に比例して計算されます。
   - デューティサイクルの反転が有効化されており（COMMAND のビット 27 がセット）、かつ、PWM_IN 1→0 のエッジで DIR_IN = 1 の場合、正規化デューティサイクル DC_{NORM} は反転します。
   - このフェーズでの計算と割り当てをまとめると次のようになります。
     - If $DC_{REL} < DC_{MIN}$: $DC_{NORM} = 0$
     - Else if $DC_{REL} > DC_{MAX}$: $DC_{NORM} = 65535 / 65536 \approx 1.0$
     - Else: $DC_{NORM} = (DC_{REL} - DC_{MIN})/(DC_{MAX} - DC_{MIN})$
     - If duty cycle inversion is active and DIR_IN = 1: $DC_{NORM} = 1.0 - DC_{NORM}$
2. 正規化デューティサイクル DC_{NORM} の値を、接続されているバイクワッド・フィルタでフィルタ処理します。詳細についてはバイクワッド・フィルタのセクションを参照してください。
   - バイクワッド・フィルタの出力は、以下において DC_{FILT} と名付けられています。
   - FILTER_SETUP コマンドを用いることによって、バイクワッド・フィルタを事前にセットアップする必要がある点に注意してください。
3. フィルタ処理されたデューティサイクル DC_{FILT} および設定された制限値を用いて、最終的な出力値を計算します。COMMAND が発行されると、32 ビットの符号付き制限値が、IO_CONTROLLER.RESPONSE_2 および IO_CONTROLLER.RESPONSE_3 によって定まります。
   - 最小出力値 OUT_{MIN} = 0xuuuuuuuu（IO_CONTROLLER.RESPONSE_2 で定義、符号付き）。
   - 最大出力値 OUT_{MAX} = 0xvvvvvvvv（IO_CONTROLLER.RESPONSE_3 で定義、符号付き）。
   - 最終出力値 OUT_{FINAL} は、これらの制限値に比例して計算されます。
   - 出力値の反転が有効化されており（COMMAND のビット 27 がセット）、かつ、PWM_IN 1→0 のエッジで DIR_IN = 1 の場合、最終出力値 OUT_{FINAL} は反転します。
   - このフェーズでの計算と割り当てをまとめると次のようになります。
     - 最終出力の計算： $OUT_{FINAL} = (OUT_{MAX} - OUT_{MIN}) \times DC_{FILT} + OUT_{MIN})$
     - 出力値反転がアクティブであり、かつ、DIR_IN = 1 の場合： $OUT_{FINAL} = -1.0 \times OUT_{FINAL}$

4. 出力値をシフトし目標レジスタ値をマスクします。
   - 目標レジスタが複数のレジスタ・フィールドを含んでいる場合、出力値 OUT_{FINAL} をシフトでき、元のレジスタ値を、特定のレジスタ・フィールドのみが上書きされるようマスクできます。
   - 「m」バイトは目標レジスタ値のマスクを定義します。「m」の各ビットは、目標レジスタのバイトを表します。いずれかのビットが 0 の場合、目標レジスタへの書込みアクセスの間、特定のバイトが保持されます。以下の例を参照してください。
     - m = 0xC の場合、目標レジスタの上位 2 バイトが最終出力値によって上書きされます。
     - m = 0xF の場合、すべてのバイトが最終出力値 OUT_{FINAL} によって上書きされます。
   - 「s」は、最終出力値 OUT_{FINAL} の可能性のあるシフト動作を定めます。
     - 例えば、「s」が 0x3 の場合、最終出力値は、目標レジスタ・フィールドに合うよう、2 バイトだけ左にシフトされます。
     - 最終出力値に対しては、以下のシフト動作が使用可能です（「s」が 0x0 の場合は、シフトは実行されません）。

“s” =	0x1	0x3	0x7	0x9	0xB	0xF
Shift	Left by 1 byte	Left by 2 bytes	Left by 3 bytes	Right by 1 byte	Right by 2 bytes	Right by 3 bytes

5. 出力値を目標レジスタに転送します。
   - 目標レジスタ・アドレスは、COMMAND の「yyy」部によって定まります。
   - 下位 10 ビットは、目標レジスタのアドレスを表します。
   - 先立って行われたすべての計算の後、目標レジスタ値は、シフトされた OUT_{FINAL} の値によって上書きされます。
   - マスクされるバイト（「m」の特定のビットが 0 に等しい場合）は、この目標レジスタの以前の値から転送されます。

DIR_IN 入力信号は、上述のように、デューティサイクル値または最終出力値を反転するために使用できます。DIR_IN の評価は PWM_IN 1→0 のエッジで常に行われる、という点に注意することが重要です。両方の場合で DIR_IN が 0 に場合、反転は実行されません。

図 52 は、例として挙げた状況での PWM_INPUT の使用方法を示しています。ここでは、FOC.PID_TORQUE_FLUX_TARGET レジスタの PID_TORQUE_TARGET フィールドは、バイト・シフトおよびマスク設定を利用することにより、同じレジスタの PID_FLUX_TARGET フィールドの値を変更せずに、PWM 入力によって調整されています。

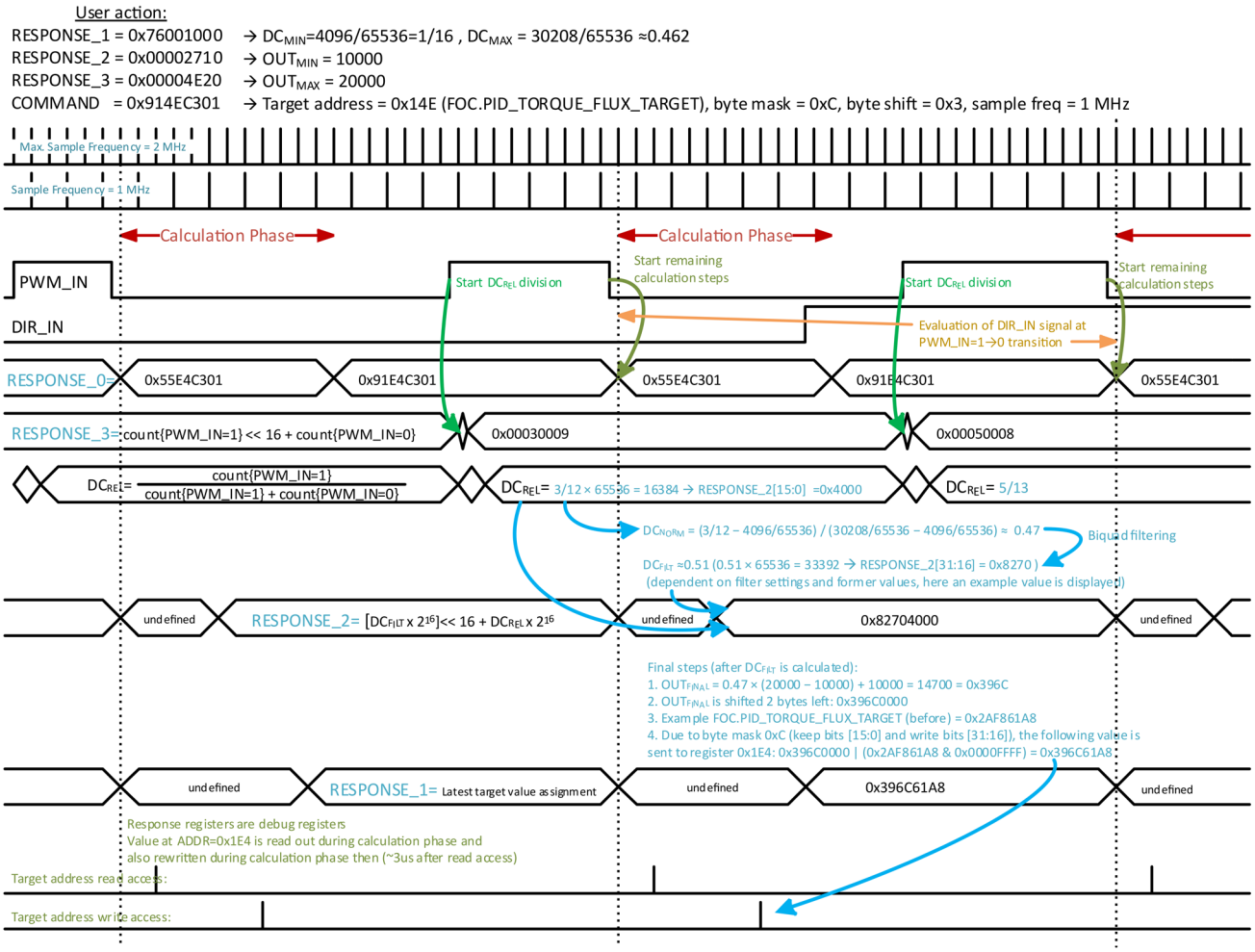


図 52. PWM_INPUT ROM コード機能の例

AIN_INPUT

表 51. AIN_INPUT のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	RESPONSE_1	0xttttzzzz			正規化されたデューティサイクルの制限値（u16）
2	RESPONSE_2	0xuuuuuuuu			最終出力値の下限値（s32）
3	RESPONSE_3	0xvvvvvvvv			最終出力値の上限値（s32）
4	COMMAND	0x8yyyms00			0xyyy：目標アドレス、0xm：バイト・マスク： 0xs：バイト・シフト
5			RESPONSE_0	0x55yyms00	計算中、応答データは有効ではありません。
6			RESPONSE_0	0x8yyyms00	計算フェーズ後、応答データは有効です。
			RESPONSE_1	0xnnnnnnnn	最後の目標値（s32）
			RESPONSE_2	0xqqqqpppp	0xqqqq：最後にフィルタ処理されたデューティサイクル値 0xpppp：最後の相対デューティサイクル値
7	COMMAND	0x00000000			AIN_INPUT機能の終了。

説明：

AINアナログ入力とオプションでDIR_IN入力を繰り返し処理し、それらを任意の目標レジスタの制御信号として使用します。この処理中、目標レジスタは、応答レジスタの有効性とは関係なく、新しい値に繰り返し書き込まれます。応答レジスタは、デバッグ目的でのみ使用されます。

ほとんどの計算ステップは、PWM_INPUT コマンドと同じです。唯一の相違は、相対デューティサイクル DC_{REL} のアクイジションにあります。AIN_INPUT の場合、DC_{REL} は、PWM サイクルごとに 1 回、アナログ入力測定から直接取得されます。そのため、RESPONSE_3 レジスタは応答レジスタとしては用いられず、COMMAND 発行時の最終出力値の上限値を定めるためにのみ用いられます。読出し可能な最大アナログ値は$(2^{15}-1)$である点に注意してください。最小および最大のデューティサイクル値（「zzzz」および「tttt」の値）の定義は、それに従って設定する必要があります。AIN の負のアナログ入力値は 0 にクリップされます。

図 53 に、例として挙げた状況での AIN_INPUT の使用方法を示します。ここでは、PID_POSITION_TARGET レジスタ全体は、バイト・シフト設定を利用することにより、AIN 入力によって調整されています。

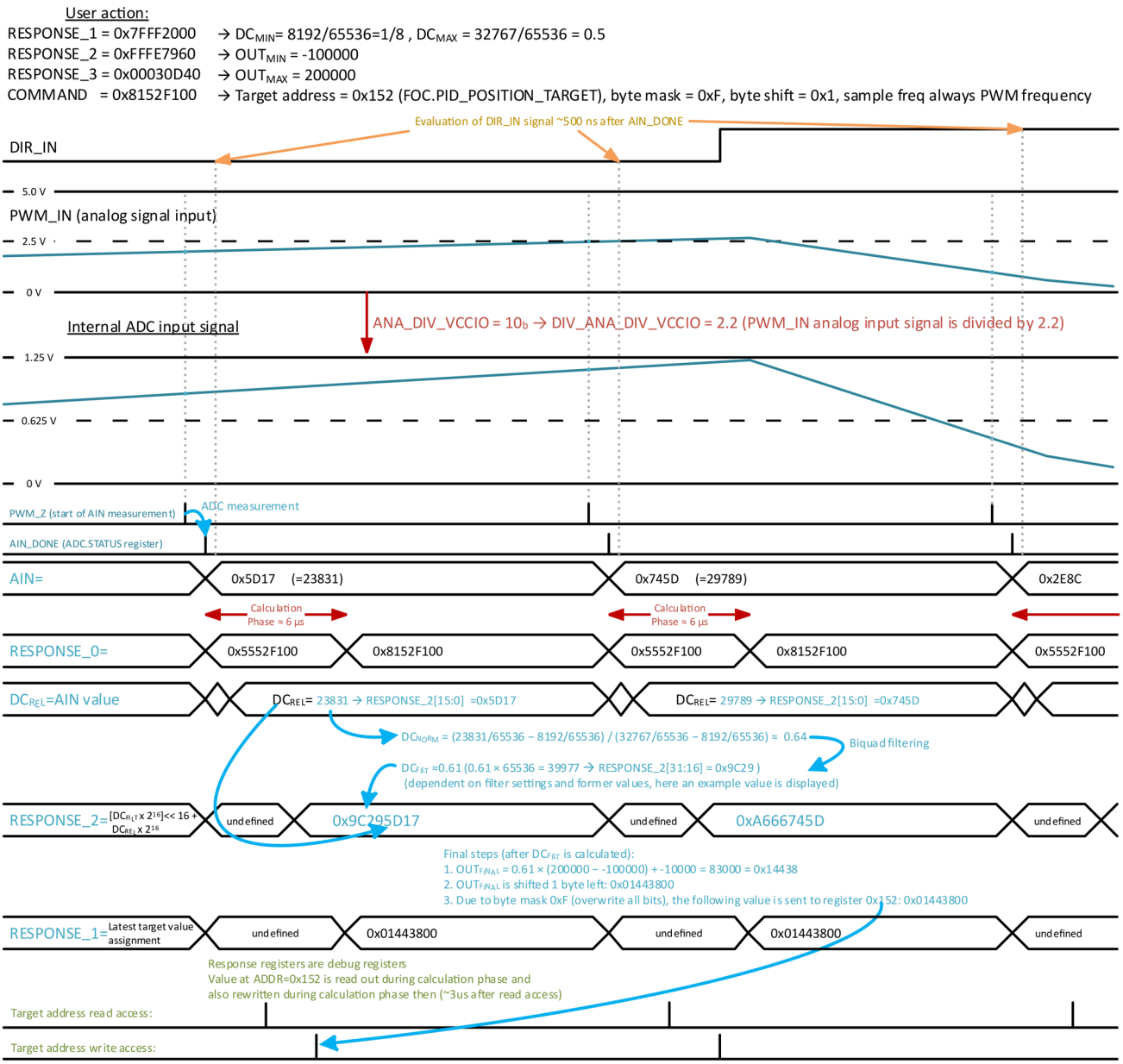


図 53. AIN_INPUT ROM コード機能の例

SPI_ENC

表 52. SPI/ENC のコマンドおよび応答

	ユーザによるアクション：レジスタへの書込み		I/Oコントローラの応答：レジスタからの読出し		注釈
	IO_CONTROLLER...	値	IO_CONTROLLER...	値	
1	RESPONSE_1	0xzzzzzzzz			SPIクライアントに送信されるアドレス／データ・ビット。
2	RESPONSE_2	0xssssssss			応答ビット・マスク
3	COMMAND	0xFywwuuvv			0xy：モード、0xww：CSNの時間、0xuu：ピン・セットアップ + ビット・カウント、0xvv：周期。
4			RESPONSE_0	0x55wwuuvv	計算中、応答データは有効ではありません。
5			RESPONSE_0	0xFywwuuvv	計算フェーズ後、応答データは有効です。
			RESPONSE_3	0xrrrrrrrr	完全なクライアント応答
6	COMMAND	0x00000000			SPI_ENC機能の終了。

説明：

SPI または SSI エンコーダなどのシリアル・クライアントからの角度データを要求し処理します。

SPI_ENC コマンドにより、単独のクライアント要求、つまり、エンコーダのレジスタとの間での読出しや書込みを行うことができます。したがって、インターフェースされているエンコーダを、セットアップ中に設定できます。

更に、SPI_ENC コマンドにより、SPI エンコーダから生のデータを繰り返し読み出すことができます。生のデータは、処理した後、FEEDBACK.PHI_EXT_A レジスタ、または FEEDBACK.PHI_EXT_B レジスタのどちらかに書き込むことができます。したがって、SPI エンコーダの角度は、フィードバック・エンジンによって更に利用できます。

以下において、SPI エンコーダ要求について、処理された角度データがフィードバック・エンジン・レジスタに書き込まれる時点までの詳細を説明します。その他のステップについては、フィードバック・エンジンのセクションを参照してください。

表 53 に、4 つの SPI 通信ラインとして用いられる I/O コントローラ信号を示します。

表 53. SPI/SSI エンコーダの機能ピンの割り当て

u[7]	CSN	SCK	SDO	SDI
0	DIRECT_OUT[2]	DIRECT_OUT[0]	DIRECT_OUT[1]	DIRECT_IN[0]
1	DIRECT_OUT[1]	DIRECT_OUT[0]	DIRECT_OUT[2]	DIRECT_IN[0]

表 53 において、u[7]は、IO_CONTROLLER.COMMAND からの 0xuu バイトのビット 7 を表します。

I/O コントローラの入力および出力と TMC6460 の物理的な I/O ピンとの間の接続は、順応性がある点に注意してください。表 54 を、関連する I/O コントローラの入力および出力として再割り当てできるピンのリファレンスとして、使用してください。詳細については、ピンの再割り当てのセクションの CHIP.IO_MATRIX レジスタの設定を参照してください。

表 54. I/O コントローラの入力および出力に対するピンの可用性

PIN NAME	DIRECT_OUT[2]	DIRECT_OUT[0]	DIRECT_OUT[1]	DIRECT_IN[0]
ENC_A		X		
ENC_B				X
ENC_N			X	
HALL_W	X			
REF_L				X
REF_R			X	
SDI				X
SDO		X		
UART_TXD		X	X	
UART_RXD				X

また、通信ピン（SDI、SDO、UART_TXD、UART_RXD）は V_{CC_IO} 電源を基準とするのに対し、表 54 のその他のピンは、VCC_IOF 電源を基準とする点にも注意してください。エンコーダで使用するピンを選択する場合には、この点を考慮する必要があります。

I/O コントローラのその他の入力および出力とは対照的に、DIRECT_OUT[2]出力信号は、TMC6460 の 1 つのピン（HALL_W）のみの代替機能として使用できます。そのため、SPI エンコーダを使用する予定である場合、デジタルおよびアナログのホール・センサー、および、

SinCos エンコーダは、第二の同時フィードバック・ソースとしては使用できなくなります。これらはすべて、HALL_W ピンが機能することが必要であるためです。第二の ABN エンコーダも一部が使用できなくなります。その N 信号も HALL_W ピンを使用するためです。

SPI エンコーダとは対照的に、SSI エンコーダは、データ入力がないため、2 つ（SCK および SDO）または場合によって 3 つ（SCK、SDO、CSN）の信号のみを使用します。CSN 信号と SDO 信号を DIRECT_OUT[1]出力と DIRECT_OUT[2]出力の間で交換できるようにすることで、SSI エンコーダと、HALL_U、HALL_V、HALL_W の各ピンとそれらの代替機能を通じて使用可能な第二フィードバック・ソースを、同時に利用できます。

COMMAND 自体、SPI 通信用にいくつかの設定を内蔵しています。

- 「y」バイトは、SPI モードおよび要求タイプを定義します。
  - COMMAND[27:26]は、用いられている一般的な SPI モード（SPI モード 0、1、2、または 3）を設定します。
  - COMMAND[25:24]は、要求タイプを定義します。

  00_b： 1 つのデータグラムが送信され、その応答（標準的な SPI 通信の直前のデータグラムの応答）は IO_CONTROLLER.RESPONSE_3 に格納されます。SPI 書込み要求に有用です。

  01_b： 2 つの同一のデータグラムが送信されます。最初に送信されたデータグラムの応答は、第二のデータグラムと共に受信され、IO_CONTROLLER.RESPONSE_3 に格納されます。SPI 読出し要求に有用です。

  10_b： データグラムは繰り返し送信され、SPI 生データを取得します。取得されたデータは処理され、最終的に、通常のフィードバック・データとして FEEDBACK.PHI_EXT_A レジスタに書き込まれます。

  11_b： データグラムは繰り返し送信され、SPI 生データを取得します。取得されたデータは処理され、最終的に、通常のフィードバック・データとして FEEDBACK.PHI_EXT_B レジスタに書き込まれます。

- 2 つの「vv」バイト（COMMAND[7:0]）は、エンコーダのクロック周期の半分を表す SCK_HALF_PERIOD を定義します。
  - SCK のスイッチング周波数はこの値により、定まります。
  - この値が 10 未満の場合、可能な最速の SCK 周波数（SCK_FREQ）である、$f_{SYS}/20 = 3MHz$ が用いられます。
  - そうでない場合は、設定された値が、次式に従い SCK 周波数を定めます。

    - $SCK_FREQ = \frac{f_{SYS}}{(SCK_HALF_PERIOD+1)\times 2}$ $with\ f_{SYS} = 60\ MHz$

  - これにより、SCK の周波数範囲は、117.2kHz～3MHz になります。
- 2 つの「uu」バイトは、上述のピン割り当て（COMMAND[15]）、および各 SPI データグラム（SPI_BIT_CNT）に送受信されるデータ・ビットの数（COMMAND[12:8]）を定めます。
  - SPI_BIT_CNT = COMMAND[12:8] + 1
  - 最大ビット・カウント数は SPI_BIT_CNT = 32 です。これより長いデータグラムは、この ROM 機能ではサポートされません。
  - 2 つの「ww」バイト（COMMAND[23:16]）は、2 つの連続する SPI データグラムの間で待機するシステム・クロックサイクル数 SPI_CSH_CNT を定めます（CSN = 1 の場合）。それによって、アイドル時間 t_{SPI_CSH} が生じます。SPI 要求タイプが 0 に等しい場合、この機能ではこの設定は考慮されず、1 つのデータグラムのみが送信されます。
  - t_{SPI_CSH} は、SPI_CSH_CNT の二乗に比例しますが、可能な最短時間は 6µs です。

    - $t_{SPI_CSH} > \frac{SPI_CSH_CNT\times(SPI_CSH_CNT+1)}{f_{SYS}} + t_{SPI_CSH_OFST}$ $with\ f_{SYS} = 60\ MHz$

  - SPI 要求タイプが 1 の場合、$t_{SPI_CSH_OFST} \approx 2\mu s$、それ以外の場合、$t_{SPI_CSH_OFST} \approx 3\mu s$ です。
    - すなわち、tSPI_CSH は、繰り返し要求と 2 データグラム要求とでわずかに異なります。

SPI エンコーダ・クライアントに送信されるアドレスおよびオプションのデータは、IO_CONTROLLER.RESPONSE_1 レジスタの 0xzzzzzzzz で定められます。データは MSB ファーストで送信される点に注意してください。つまり、常にビット[31]が最初に送信されます。例えば、12 ビットの 0xABC データをクライアントに送信する必要がある場合、IO_CONTROLLER.RESPONSE_1 を 0xABC00000 に設定しなくてはなりません。

エンコーダ・クライアントの応答は、IO_CONTROLLER.RESPONSE_3 に格納されます。そこでは、応答データは下位側のバイトに割り当てられます。12 ビット・データの前述の例の場合、0xABC の要求に対する応答として、応答が 0xDEF で構成されるならば、IO_CONTROLLER.RESPONSE_3 は、0x00ABCDEF となります。応答には、送信されたデータグラムの一部が含まれます。内部では出力レジスタは入力レジスタと同じであるためです。この場合、下位 12 ビットのみが必須です。これらがエンコーダ・クライアントからの応答であるためです。

繰り返し要求タイプの場合、応答値は、選択した要求タイプに応じて自動的に処理され、フィードバック・エンジンの定められたレジスタに格納されます。

自動角度データ・アクイジションおよび処理が機能するための制限事項は次のとおりです。1) 角度値を含む応答データは、32 ビットを超えてはならず、かつ、2) 角度値は、単一の応答データグラムから取得できなくてはなりません。例えば、2 つの連続するデータグラムからのデータを結合することはできません。ほとんどの SPI エンコーダ・クライアントは、データが複数の連続するレジスタに分割されている場合に、生の角度値を 1 つのデータグラムでストリーム・アウトするオプションを備えています。

SPI エンコーダ・クライアントによっては、特定のステータス・ビットがデータグラムの一部になっているなどの理由により、角度データを生の応答データグラムから直接解釈することができません。そのような場合のために、SPI_ENC 機能には、生の応答データを自動的に処理するマスク機能が備わっています。

IO_CONTROLLER.RESPONSE_2（= 0xssssssss）の値のセットは、マスク・データグラムの役割を担います。エンコーダの角度値に関連するクライアント応答データグラムのデータ・ビットはすべて、マスク・データグラムにおいて 1 に設定する必要があり、省略するデータ・ビットは 0 に設定する必要があります。ステータス・ビットが角度データ・ビットの間に配置される場合、これらのビットは、バイトの末尾または先頭になくてはならないという点を唯一の制限として、マスクできます。最終的な角度値は、マスクしないビットを右にシフトすることで配列されます。表 55 に、様々なマスク値の例を示します。

表 55. SPI/SSI エンコーダのマスク／シフト例

SPI_BIT_CNT	BIT MASK (RESPONSE_2)	ENCODER RAW DATA	FINAL FEEDBACK DATA	MASKED RAW DATA BITS	COMMENT
12	0x000003FC	0x00000789	0x000000E2	[11:10] [1:0]	Final value 8 bits wide: [7:0] = Raw[9:2]
12	0x00000E7F	0x00000789	0x00000189	[8:7]	Final value 10 bits wide: [9:0] = Raw[11:9][6:0]
14	0x00001CFC	0x00003E9F	0x000001E7	[13] [9:8] [1:0]	Final value 9 bits wide: [8:0] = Raw[12:10][7:2]
14	0x000006FF	0x00003E9F	0x0000039F	[13:11] [8]	Final value 8 bits wide: [7:0] = Raw[10:9][7:0]
16	0x0000FFF0	0x00008049	0x00000804	[3:0]	Final value 12 bits wide: [11:0] = Raw[15:4]
16	0x00007E1F	0x00008049	0x00000009	[15] [8:5]	Final value 11 bits wide: [10:0] = Raw[14:9][4:0]
16	0x00006C1F				**Not valid**: Masked bits are in the middle of byte2.
24	0x00F83F78	0x007C7864	0x00003F8C	[18:14] [7] [2:0]	Final value 15 bits wide: [14:0] = Raw[23:19][13:8][6:3]

生の応答データが配列（マスクおよびシフト）された後、その結果生じた角度値は、目標レジスタ FEEDBACK.PHI_EXT_A または FEEDBACK.PHI_EXT_B に自動的に書き込まれ、フィードバック・エンジンによって、更に処理されます。

角度データの更新レートは、上述のとおり、SCK 周波数およびアイドル時間によって定められます。

繰り返し SPI エンコーダ要求を終了するには、IO_CONTROLLER.COMMAND = 0x00000000 の「アイドル」コマンドを送信します。応答データは、新しいコマンドが発行されない限り、保持されます。

図 54 に、連続 SPI エンコーダ（SPI モード 3）データの読出しの例を示します。転送された 8 個のビットのうち 4 個のみが、角度値として用いられます（残りはステータス・フラグなどで構成されます）。これは、IO_CONTROLLER.RESPONSE_2 レジスタのバイト・マスクによって定められます。この図には、上述のいくつかのタイミングも示されています。更に、IO_CONTROLLER.RESPONSE_1 レジスタは、0x56023456 という値で満たされています。これには、SPI エンコーダに送信されたデータ（0x56）と、コマンドの発行後 SPI エンコーダから受信したその他のいくつかのデータ（0x023456）が含まれます。これらの最後のビットは、SPI エンコーダに送信されず、内部の FEEDBACK.PHI_EXT_A レジスタにも転送されません。これらのビットは、ここでは、入力と出力のストリーム・データを組み合わせた使用を説明するためだけに示されています。

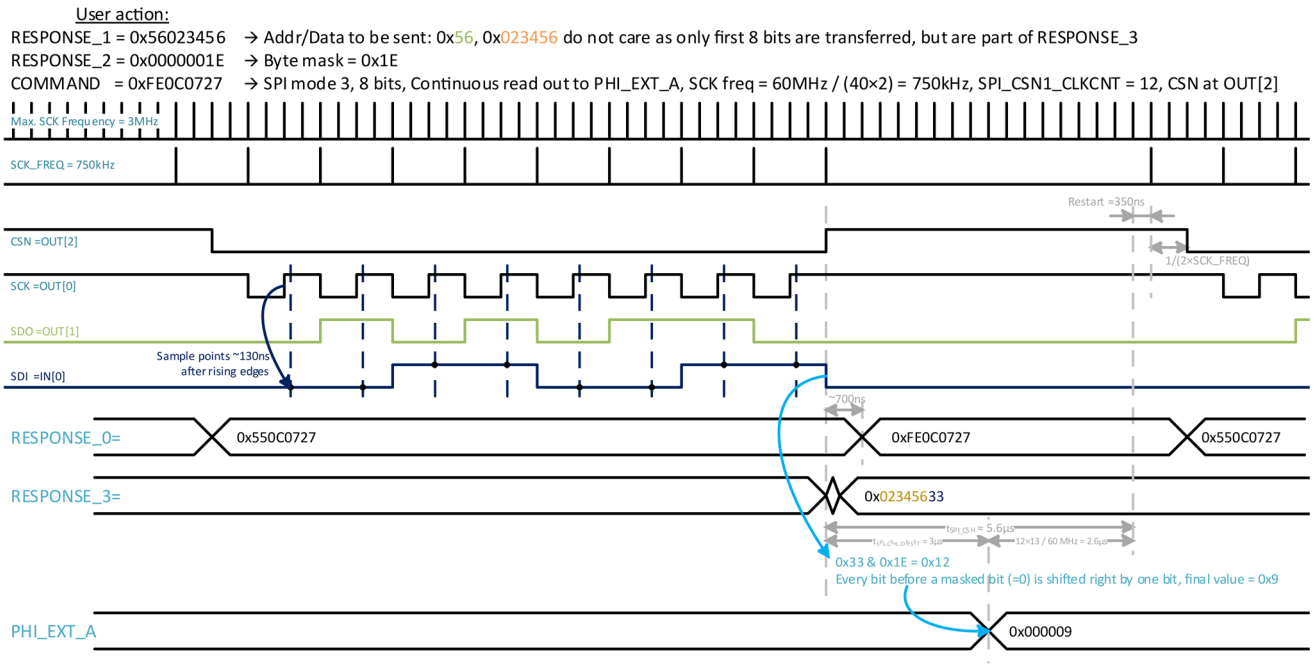


図 54. 連続 SPI エンコーダ読出しの例。目標レジスタは PHI_EXT_A。

図 55 に、ワンタイム SPI エンコーダ（SPI モード 0）読出しの例を示します。これは、COMMAND[25:24] = 01_b を用いており、2 つのデータグラムのみが送信されます。その後、第二のデータグラムに含まれるクライアントの応答が、IO_CONTROLLER.RESPONSE_3 レジスタから読み出せます。IO_CONTROLLER.RESPONSE_2 レジスタを満たすことは必須ではありません。FEEDBACK.PHI_EXT_A レジスタや FEEDBACK.PHI_EXT_B レジスタへのデータ転送がないためです。

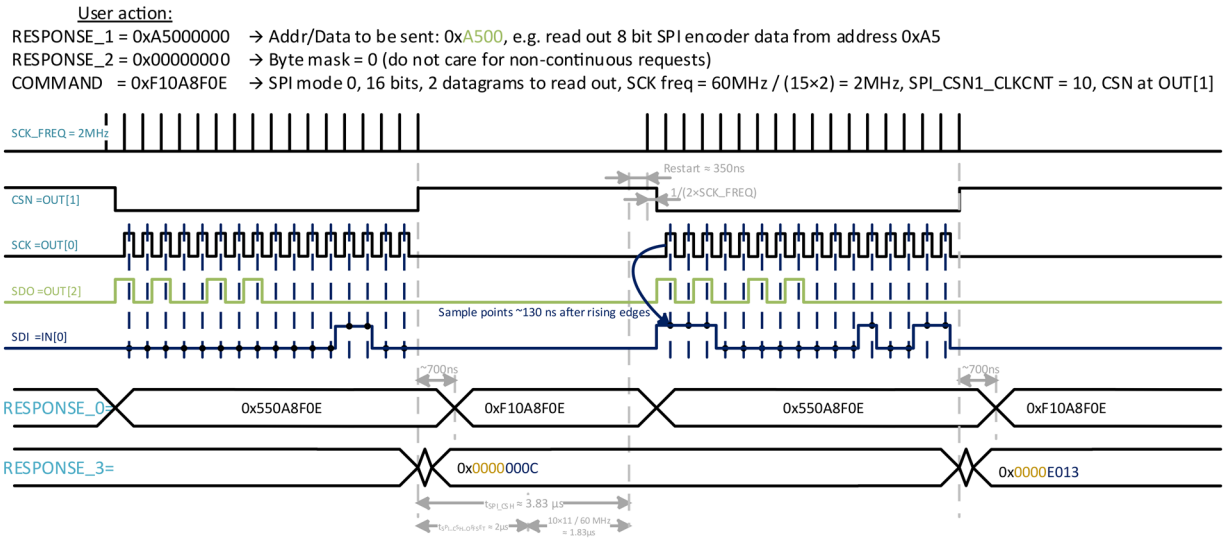


図 55. 同一のデータグラムを 2 回送信して RESPONSE_3 レジスタに応答を得ることによる、ワンタイム SPI エンコーダ読出しの例

最後に、同じコマンド・シーケンスで、SSI エンコーダも使用できます。図 56 にこのアプリケーションのセットアップを示します。SDO から SSI エンコーダへのデータ経路は不要です。SSI エンコーダが 1 つのクロック入力および 1 つのデータ出力のみを使用する場合、CSN はオプションです。この例では、バイト・マスクの使用についても示しています。ほとんどのエンコーダは、ギャップに満ちたデータ・ストリームを供給することはありません。しかし、角度値の一部ではないビットがバイトの先頭または末尾に配置されている限り、この ROM コード機能は、マスク処理とシフト処理を行い、内部の FEEDBACK.PHI_EXT_A レジスタまたは FEEDBACK.PHI_EXT_B レジスタに有効な角度値を提供できます。

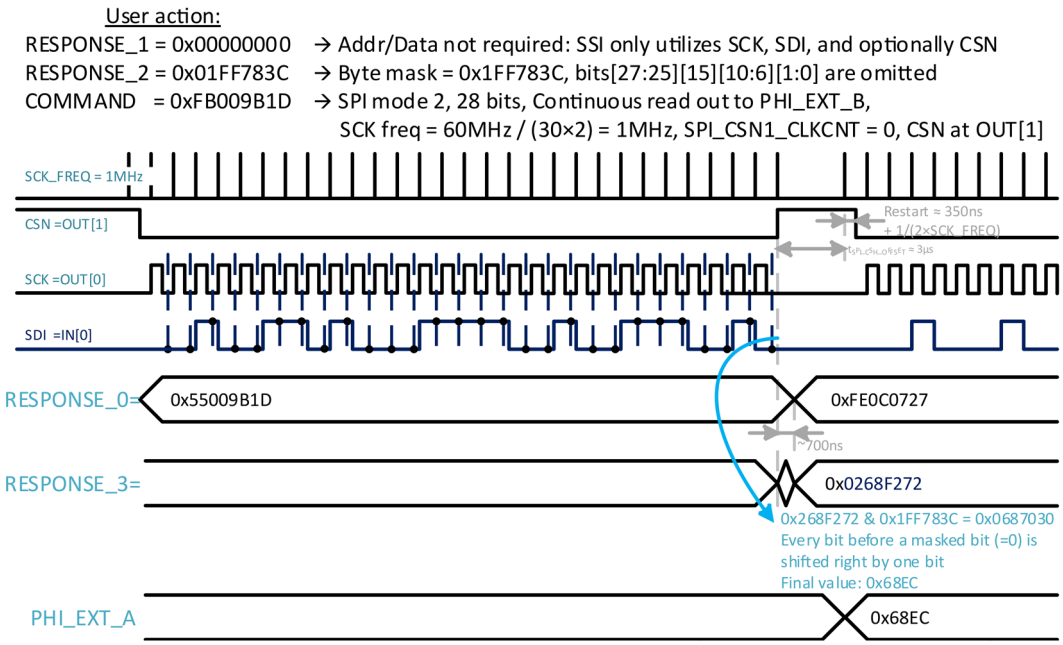


図 56. 連続 SSI エンコーダ読出しの例。目標レジスタは PHI_EXT_B。

アプリケーション情報

TMC6460 は、様々なフィードバック・システムで BLDC モーター／三相 PMSM を駆動できます。以下のアプリケーションのブロック図にいくつかの例を示します。

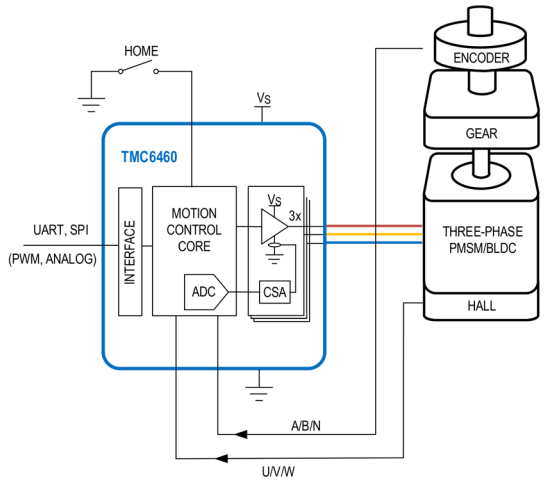


図 57. ギアボックス後段にインクリメンタル光エンコーダ、モーターにデジタル・ホール・センサーを備えた BLDC

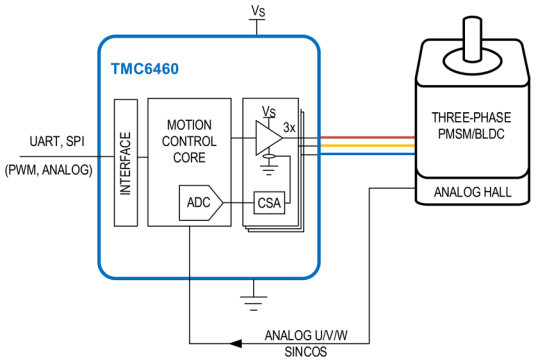


図 58. アナログ・ホール・センサーまたは SinCos エンコーダを備えた BLDC

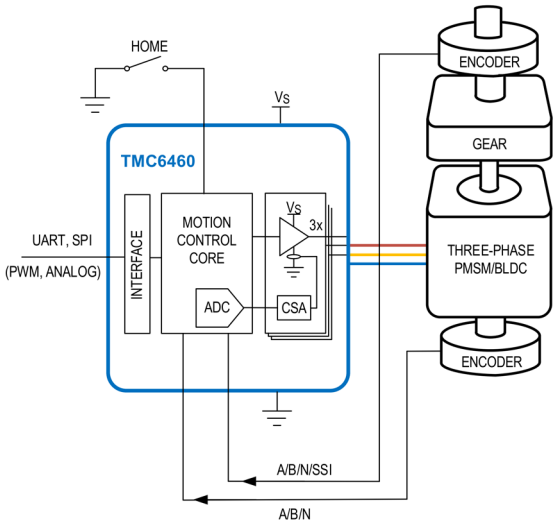


図 59. 1 つはインクリメンタル A/B/N エンコーダ、もう 1 つはインクリメンタルまたは絶対（SSI など）エンコーダの、2 つのエンコーダを備えた BLDC。1 つのエンコーダはモーターに取り付けられ、もう 1 つはギアボックスの後段にあります。

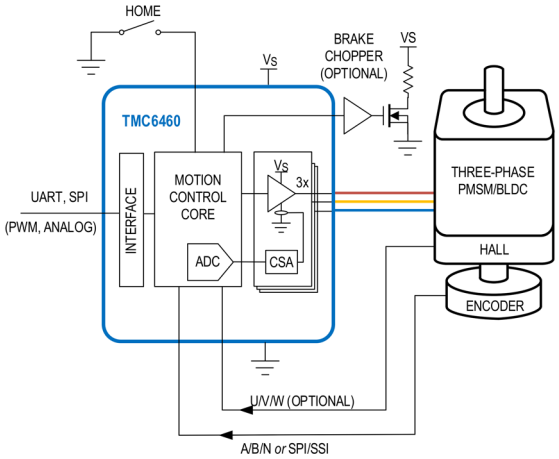


図 60. モーターに光エンコーダとデジタル・ホールを備え、更にブレーキ・チョッパーを使用する BLDC

標準的なアプリケーション回路

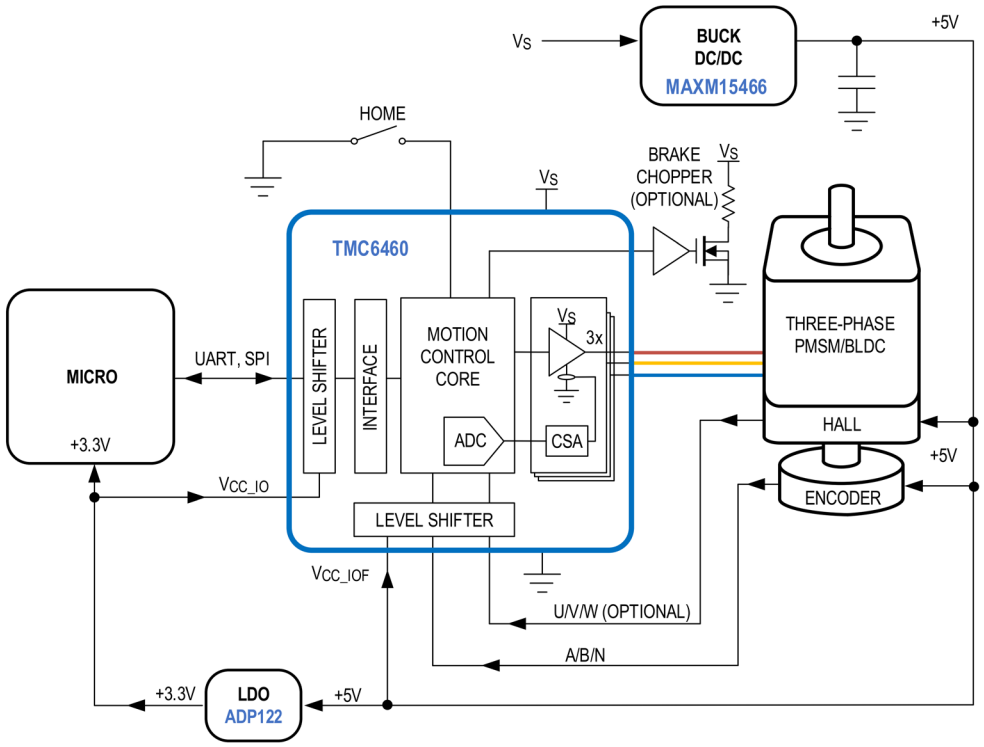


図 61. 標準的なアプリケーション回路のブロック図

レジスタ・マップ

レジスタの概要

TMC6460 のすべてのレジスタの概要を以下に示します。

ADDRESS AND NAME	FIELDS							
0x000 CHIP.ID	ID[31:24]							
	ID[23:16]							
	ID[15:8]							
	ID[7:0]							
0x001 CHIP.VARIANT	VARIANT[31:24]							
	VARIANT[23:16]							
	VARIANT[15:8]							
	VARIANT[7:0]							
0x002 CHIP.REVISION		REVISION[30:24]						
	REVISION[23:16]							
	REVISION[15:8]							
	REVISION[7:0]							
0x004 CHIP.INPUTS_RAW	PWM_IN	DIR_IN	DIRECT_IN1	DIRECT_IN0	REF_H	REF_R	REF_L	HALL_W
	HALL_V	HALL_U	ENC2_N	ENC2_B	ENC2_A	ENC_N	ENC_B	ENC_A
	PIN_DRV_EN		PIN_UART_RXD_RAW	PIN_PWM_IN_RAW	PIN_CSN_RAW	PIN_SCK_RAW	PIN_SDO_RAW	PIN_SDI_RAW
	PIN_REF_R_RAW	PIN_REF_L_RAW	PIN_HALL_W_RAW	PIN_HALL_V_RAW	PIN_HALL_U_RAW	PIN_ENC_N_RAW	PIN_ENC_B_RAW	PIN_ENC_A_RAW
0x005 CHIP.OUTPUTS_RAW						INVALID_DIFF_ENC_N	INVALID_DIFF_ENC_B	INVALID_DIFF_ENC_A
	FAULTN_OUT	INT_OUT	PWM_W_H	PWM_W_L	PWM_V_H	PWM_V_L	PWM_U_H	PWM_U_L
0x006 CHIP.IO_MATRIX	PIN_FAULTN		PIN_UART_TXD		PIN_UART_RXD		PIN_PWM_IN	
	PIN_SCK	PIN_SDO		PIN_SDI		PIN_REF_R		PIN_REF_L[1]
	PIN_REF_L[0]	PIN_HALL_W			PIN_HALL_V			PIN_HALL_U[2]
	PIN_HALL_U[1:0]		PIN_ENC_N		PIN_ENC_B		PIN_ENC_A	
0x007 CHIP.IO_PU_PD					PIN_UART_RXD_PD	PIN_PWM_IN_PD	PIN_SCK_PD	PIN_SDI_PD
		PIN_FAULTN_PU		PIN_SDO_PU	PIN_UART_RXD_PU	PIN_PWM_IN_PU	PIN_SCK_PU	PIN_SDI_PU
	PIN_REF_R_PD	PIN_REF_L_PD	PIN_HALL_W_PD	PIN_HALL_V_PD	PIN_HALL_U_PD	PIN_ENC_N_PD	PIN_ENC_B_PD	PIN_ENC_A_PD
	PIN_REF_R_PU	PIN_REF_L_PU	PIN_HALL_W_PU	PIN_HALL_V_PU	PIN_HALL_U_PU	PIN_ENC_N_PU	PIN_ENC_B_PU	PIN_ENC_A_PU
0x008 CHIP.IO_CONFIG	DIFF_DELAY						USE_PIN_REF_R_AS_REF_L	USE_PIN_REF_L_AS_REF_R
	ANA_DIV_VCCIOF		ANA_DIV_VCCIO		PWM_IN_FLT		REF_FLT	
	HALL_FLT		ENC_FLT		INV_PIN_RXD	INV_PIN_PWM_IN	INV_PIN_SCK	INV_PIN_SDI
	INV_PIN_REF_R	INV_PIN_REF_L	INV_PIN_HALL_W	INV_PIN_HALL_V	INV_PIN_HALL_U	INV_PIN_ENC_N	INV_PIN_ENC_B	INV_PIN_ENC_A

ADDRESS AND NAME	FIELDS				MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)			
0x009 CHIP. STATUS_FLAGS	GDRV_ON_STATE	SYS_READY_STATE	OVERTEMP_WARN_STATUS	OV_VM_LIMIT_FAIL_STATUS	VM_TEMP_CLIPPED_STATUS	AIN_CLIPPED_STATUS	I_CLIPPED_STATUS	CURRENT_OVERLOAD_STATUS
	HALL_FAIL_STATUS	ABN_FAIL_STATUS	RAMP_REF_H_STATUS	RAMP_REF_LR_STATUS	RAMP_V_ZERO_STATUS	RAMP_V_REACHED_STATUS	RAMP_TARGET_REACHED_STATUS	RAMP_STALL_FAIL_STATUS
	IO_CONTROLLER_STATUS	DIFF_ENC_FAIL_STATUS	OT_FAIL_STATUS	SHRT_FAIL_STATUS	VEL_FAIL_STATUS	TEMP_LIMIT_FAIL_STATUS	ADC_FAIL_STATUS	EXT_RES_FAIL_STATUS
	CLK_PLL_FAIL_STATUS	POWER_FAIL_STATUS	UV_VM_STATUS	CP_FAIL_STATUS	VCC_IOF_FAIL_STATUS	UART_FAIL_STATUS	SPI_FAIL_STATUS	PWRUP_FAIL_STATUS
0x00A CHIP.EVENTS	GDRV_ON_EVENT	SYS_READY_EVENT	OVERTEMP_WARN_EVENT	OV_VM_LIMIT_FAIL_EVENT	VM_TEMP_CLIPPED_EVENT	AIN_CLIPPED_EVENT	I_CLIPPED_EVENT	CURRENT_OVERLOAD_EVENT
	HALL_FAIL_EVENT	ABN_FAIL_EVENT	RAMP_REF_H_EVENT	RAMP_REF_LR_EVENT	RAMP_V_ZERO_EVENT	RAMP_V_REACHED_EVENT	RAMP_TARGET_REACHED_EVENT	RAMP_STALL_FAIL_EVENT
	IO_CONTROLLER_EVENT	DIFF_ENC_FAIL_EVENT	OT_FAIL_EVENT	SHRT_FAIL_EVENT	VEL_FAIL_EVENT	TEMP_LIMIT_FAIL_EVENT	ADC_FAIL_EVENT	EXT_RES_FAIL_EVENT
	CLK_PLL_FAIL_EVENT	POWER_FAIL_EVENT	UV_VM_EVENT	CP_FAIL_EVENT	VCCIOF_FAIL_EVENT	UART_FAIL_EVENT	SPI_FAIL_EVENT	RST_EVENT
0x00B CHIP. FAULTN_INT_MASK	FAULTN_INT_MASK[31:24]							
	FAULTN_INT_MASK[23:16]							
	FAULTN_INT_MASK[15:8]							
	FAULTN_INT_MASK[7:0]							
0x00C CHIP. SPI_STATUS_MASK	SPI_STATUS_MASK[31:24]							
	SPI_STATUS_MASK[23:16]							
	SPI_STATUS_MASK[15:8]							
	SPI_STATUS_MASK[7:0]							
0x040 CLK_CTRL.CONFIG								
				CLOCK_DIVIDER				
			CLK_FSM_EN	PWM_CLK_EN	ADC_CLK_EN	PLL_EN	PLL_SRC	COMMIT
0x041 CLK_CTRL.STATUS	CLK_FSM_FLG							
	CLK_FSM_ERR_RPT							
		PLL_READY	PLL_LOCK_ON	PLL_ERR	CLK_STUCK	CLK_FSM_ERR	CLK_1M0_TIMEOUT	CLK_1M0_OK
0x080 ADC.CONFIG								
					CSA_AZ_FLT_EXP			
					NRST_ADC_1	NRST_ADC_0		

ADDRESS AND NAME	FIELDS							MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x081 ADC.ADC_VERSION	VERSION_NUMBER[31:24]							
	VERSION_NUMBER[23:16]							
	VERSION_NUMBER[15:8]							
	VERSION_NUMBER[7:0]							
0x082 ADC.I2_I1_RAW	I2[15:8]							
	I2[7:0]							
	I1[15:8]							
	I1[7:0]							
0x083 ADC.VM_I3_RAW	VM[15:8]							
	VM[7:0]							
	I3[15:8]							
	I3[7:0]							
0x084 ADC.TEMP_RAW	TEMP_INT[15:8]							
	TEMP_INT[7:0]							
	TEMP_EXT[15:8]							
	TEMP_EXT[7:0]							
0x085 ADC. AIN_V_AIN_U_RAW	AIN_V[15:8]							
	AIN_V[7:0]							
	AIN_U[15:8]							
	AIN_U[7:0]							
0x086 ADC. AIN_AIN_W_RAW	AIN[15:8]							
	AIN[7:0]							
	AIN_W[15:8]							
	AIN_W[7:0]							
0x08A ADC.STATUS	I123_FAIL	NO_ACK_ ADC			USE_ADC_ EXT_ VALID_EXT	ADC_EXT_ VALID_EXT	USE_ADC_ I_VALID_ EXT	ADC_I_ VALID_EXT
	ADC_EXT_ VALID	ADC_I_ VALID	AIN_DONE	AIN_W_ DONE	AIN_V_ DONE	AIN_U_ DONE	I3_DONE	I2_DONE
	I1_DONE	TEMP_ EXT_DONE	TEMP_ EXT_ CLIPPED	TEMP_INT_ DONE	TEMP_INT_ CLIPPED	VM_DONE	VM_ CLIPPED	AIN_ CLIPPED
	AIN_W_ CLIPPED	AIN_V_ CLIPPED	AIN_U_ CLIPPED	I3_CLIPPED	I2_CLIPPED	I1_CLIPPED	ADC_1_ READY	ADC_0_ READY
0x08C ADC.I123	I123_LIMIT[15:8]							
	I123_LIMIT[7:0]							
	I123[15:8]							
	I123[7:0]							
0x0C0 MCC_ADC. I_GEN_CONFIG								
			STATUS					
								MEAS_ MODE
0x0C1 MCC_ADC.IW_IU	IW[15:8]							
	IW[7:0]							
	IU[15:8]							
	IU[7:0]							
0x0C2 MCC_ADC.IV								
	IV[15:8]							
	IV[7:0]							

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)						
0x0C3 MCC_ADC.CSA_GAIN								
	DYN_GAIN_HYST							
		DYN_GAIN_4X_EN	DYN_GAIN_3X_EN	DYN_GAIN_2X_EN	DYN_GAIN_ACT		CSA_GAIN	
0x0C4 MCC_ADC.EVENTS								
		CURRENT_OVERLOAD_STATUS	TEMP_INT_LIMIT_EXCEEDED	TEMP_EXT_LIMIT_EXCEEDED			MEAS_DONE	ADC_CLIPPED
0x0C5 MCC_ADC. DYN_GAIN_LIMITS_4X_3X	DYN_GAIN_LIMIT_4X[15:8]							
	DYN_GAIN_LIMIT_4X[7:0]							
	DYN_GAIN_LIMIT_3X[15:8]							
	DYN_GAIN_LIMIT_3X[7:0]							
0x0C6 MCC_ADC. DYN_GAIN_LIMIT_2X								
	DYN_GAIN_LIMIT_2X[15:8]							
	DYN_GAIN_LIMIT_2X[7:0]							
0x0C7 MCC_ADC. TEMP_LIMITS	TEMP_INT_LIMIT[15:8]							
	TEMP_INT_LIMIT[7:0]							
	TEMP_EXT_LIMIT[15:8]							
	TEMP_EXT_LIMIT[7:0]							
0x0C8 MCC_ADC. CURRENT_OVERLOAD								
	CURRENT_OVERLOAD[15:8]							
	CURRENT_OVERLOAD[7:0]							
0x100 MCC_CONFIG. MOTOR_MOTION								
					VELOCITY_FF_EN	TORQUE_FF_VISC_FRIC_EN	TORQUE_FF_COULOMB_FRICTION_EN	TORQUE_FF_ACC_EN
	RAMP_USE_PHI_E	RAMP_EN	RAMP_MODE	MOTION_MODE				MOTOR_TYPE[1]
	MOTOR_TYPE[0]	N_POLE_PAIRS						
0x101 MCC_CONFIG.GDRV	CHARGE_PUMP_EN			LP_MODE_EN				USE_INTERNAL_R_REF
				OVERTEMP_LATCH				DRV_EN_BIT
			OCP_AUTORETRY		OCP_DEGLITCH		OCP_DETECTION_MODE	
			LS_RES_ON				SLEW_RATE	
0x102 MCC_CONFIG.PWM	FLAT_BOTTOM_OFFSET[15:8]							
	FLAT_BOTTOM_OFFSET[7:0]							
			SV_MODE		CHOP			
0x103 MCC_CONFIG. PWM_PERIOD	AUTO_KIRCHOFF_LIM[15:8]							
	AUTO_KIRCHOFF_LIM[7:0]							
	MAX_COUNT[15:8]							
	MAX_COUNT[7:0]							

ADDRESS AND NAME	FIELDS							MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x104 MCC_CONFIG. BRAKE_CHOPPER_ LIMITS	UPPER_SWITCH_LIM[15:8]							
	UPPER_SWITCH_LIM[7:0]							
	LOWER_SWITCH_LIM[15:8]							
	LOWER_SWITCH_LIM[7:0]							
0x105 MCC_CONFIG. MCC_STATUS								OVERTEMP _CH3_ EVENT
	OVERTEMP _CH2_ EVENT	OVERTEMP _CH1_ EVENT	SHRT2V_ CH3_ EVENT	SHRT2V_ CH2_ EVENT	SHRT2V_ CH1_ EVENT	SHRT2G_ CH3_ EVENT	SHRT2G_ CH2_ EVENT	SHRT2G_ CH1_ EVENT
								OVERTEMP _CH3_ STATUS
	OVERTEMP _CH2_ STATUS	OVERTEMP _CH1_ STATUS	SHRT2V_ CH3_ STATUS	SHRT2V_ CH2_ STATUS	SHRT2V_ CH1_ STATUS	SHRT2G_ CH3_ STATUS	SHRT2G_ CH2_ STATUS	SHRT2G_ CH1_ STATUS
0x106 MCC_CONFIG. TORQUE_FF_ACC_ CONFIG								
						RAMP_ACC_SHIFT		
	RAMP_ACC_GAIN[15:8]							
	RAMP_ACC_GAIN[7:0]							
0x107 MCC_CONFIG. TORQUE_FF_VISC_ FRIC_CONFIG								
						RAMP_VEL_SHIFT		
	RAMP_VEL_GAIN[15:8]							
	RAMP_VEL_GAIN[7:0]							
0x108 MCC_CONFIG. TORQUE_FF_ COULOMB_FRIC								
	TORQUE_FF_COULOMB_FRIC[15:8]							
	TORQUE_FF_COULOMB_FRIC[7:0]							
0x109 MCC_CONFIG. TORQUE_ FEEDFORWARD								
	TORQUE_FEEDFORWARD[15:8]							
	TORQUE_FEEDFORWARD[7:0]							
0x140 FOC.PID_CONFIG								
	POSITION_SAMPLING							
	VELOCITY_SHIFT				POSITION_NORM_I		POSITION_NORM_P	
	VELOCITY_NORM_I		VELOCITY_NORM_P		CURRENT_ NORM_I	CURRENT_ NORM_P	OVERWRIT E_TARGET	KEEP_ POS_ TARGET
0x141 FOC.PID_U_S_MAX								
	U_S_MAX[15:8]							
	U_S_MAX[7:0]							
0x142 FOC. PID_FLUX_COEFF	FLUX_P[15:8]							
	FLUX_P[7:0]							
	FLUX_I[15:8]							
	FLUX_I[7:0]							
0x143 FOC. PID_TORQUE_COEFF	TORQUE_P[15:8]							
	TORQUE_P[7:0]							
	TORQUE_I[15:8]							
	TORQUE_I[7:0]							
0x144 FOC. PID_FIELDWEAK_ COEFF	FIELDWEAK_P[15:8]							
	FIELDWEAK_P[7:0]							
	FIELDWEAK_I[15:8]							
	FIELDWEAK_I[7:0]							

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x145 FOC. PID_VELOCITY_COEFF	VELOCITY_P[15:8]	
	VELOCITY_P[7:0]	
	VELOCITY_I[15:8]	
	VELOCITY_I[7:0]	
0x146 FOC. PID_POSITION_COEFF	POSITION_P[15:8]	
	POSITION_P[7:0]	
	POSITION_I[15:8]	
	POSITION_I[7:0]	
0x147 FOC. PID_POSITION_ TOLERANCE	PID_POSITION_TOLERANCE[30:24]	
	PID_POSITION_TOLERANCE[23:16]	
	PID_POSITION_TOLERANCE[15:8]	
	PID_POSITION_TOLERANCE[7:0]	
0x148 FOC. PID_POSITION_ TOLERANCE_DELAY		
	PID_POSITION_TOLERANCE_DELAY[15:8]	
	PID_POSITION_TOLERANCE_DELAY[7:0]	
0x149 FOC. PID_UQ_UD_LIMITS		
	PID_UQ_UD_LIMITS[15:8]	
	PID_UQ_UD_LIMITS[7:0]	
0x14A FOC. PID_TORQUE_FLUX_ LIMITS	PID_TORQUE_LIMIT[14:8]	
	PID_TORQUE_LIMIT[7:0]	
	PID_FLUX_LIMIT[14:8]	
	PID_FLUX_LIMIT[7:0]	
0x14B FOC. PID_VELOCITY_LIMIT	PID_VELOCITY_LIMIT[30:24]	
	PID_VELOCITY_LIMIT[23:16]	
	PID_VELOCITY_LIMIT[15:8]	
	PID_VELOCITY_LIMIT[7:0]	
0x14C FOC. PID_POSITION_LIMIT_ LOW	PID_POSITION_LIMIT_LOW[31:24]	
	PID_POSITION_LIMIT_LOW[23:16]	
	PID_POSITION_LIMIT_LOW[15:8]	
	PID_POSITION_LIMIT_LOW[7:0]	
0x14D FOC. PID_POSITION_LIMIT_ HIGH	PID_POSITION_LIMIT_HIGH[31:24]	
	PID_POSITION_LIMIT_HIGH[23:16]	
	PID_POSITION_LIMIT_HIGH[15:8]	
	PID_POSITION_LIMIT_HIGH[7:0]	
0x14E FOC. PID_TORQUE_FLUX_ TARGET	PID_TORQUE_TARGET[15:8]	
	PID_TORQUE_TARGET[7:0]	
	PID_FLUX_TARGET[15:8]	
	PID_FLUX_TARGET[7:0]	
0x14F FOC. PID_TORQUE_FLUX_ OFFSET	PID_TORQUE_OFFSET[15:8]	
	PID_TORQUE_OFFSET[7:0]	
	PID_FLUX_OFFSET[15:8]	
	PID_FLUX_OFFSET[7:0]	
0x150 FOC. PID_VELOCITY_ TARGET	PID_VELOCITY_TARGET[31:24]	
	PID_VELOCITY_TARGET[23:16]	
	PID_VELOCITY_TARGET[15:8]	
	PID_VELOCITY_TARGET[7:0]	
0x151 FOC. PID_VELOCITY_ OFFSET	PID_VELOCITY_OFFSET[31:24]	
	PID_VELOCITY_OFFSET[23:16]	
	PID_VELOCITY_OFFSET[15:8]	
	PID_VELOCITY_OFFSET[7:0]	

ADDRESS AND NAME	FIELDS MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x152 FOC. PID_POSITION_TARGET	PID_POSITION_TARGET[31:24]
	PID_POSITION_TARGET[23:16]
	PID_POSITION_TARGET[15:8]
	PID_POSITION_TARGET[7:0]
0x153 FOC. PID_TORQUE_FLUX_ ACTUAL	PID_TORQUE_ACTUAL[15:8]
	PID_TORQUE_ACTUAL[7:0]
	PID_FLUX_ACTUAL[15:8]
	PID_FLUX_ACTUAL[7:0]
0x154 FOC. PID_VELOCITY_ ACTUAL	PID_VELOCITY_ACTUAL[31:24]
	PID_VELOCITY_ACTUAL[23:16]
	PID_VELOCITY_ACTUAL[15:8]
	PID_VELOCITY_ACTUAL[7:0]
0x155 FOC. PID_POSITION_ACTUAL	PID_POSITION_ACTUAL[31:24]
	PID_POSITION_ACTUAL[23:16]
	PID_POSITION_ACTUAL[15:8]
	PID_POSITION_ACTUAL[7:0]
0x156 FOC. PID_POSITION_ ACTUAL_OFFSET	PID_POSITION_ACTUAL_OFFSET[31:24]
	PID_POSITION_ACTUAL_OFFSET[23:16]
	PID_POSITION_ACTUAL_OFFSET[15:8]
	PID_POSITION_ACTUAL_OFFSET[7:0]
0x157 FOC. PID_TORQUE_ERROR	
	PID_TORQUE_ERROR[15:8]
	PID_TORQUE_ERROR[7:0]
0x158 FOC. PID_FLUX_ERROR	
	PID_FLUX_ERROR[15:8]
	PID_FLUX_ERROR[7:0]
0x159 FOC. PID_VELOCITY_ERROR	PID_VELOCITY_ERROR[31:24]
	PID_VELOCITY_ERROR[23:16]
	PID_VELOCITY_ERROR[15:8]
	PID_VELOCITY_ERROR[7:0]
0x15A FOC. PID_VELOCITY_ ERROR_MAX	PID_VELOCITY_ERROR_MAX[30:24]
	PID_VELOCITY_ERROR_MAX[23:16]
	PID_VELOCITY_ERROR_MAX[15:8]
	PID_VELOCITY_ERROR_MAX[7:0]
0x15B FOC. PID_POSITION_ERROR	PID_POSITION_ERROR[31:24]
	PID_POSITION_ERROR[23:16]
	PID_POSITION_ERROR[15:8]
	PID_POSITION_ERROR[7:0]
0x15C FOC. PID_POSITION_ ERROR_MAX	PID_POSITION_ERROR_MAX[30:24]
	PID_POSITION_ERROR_MAX[23:16]
	PID_POSITION_ERROR_MAX[15:8]
	PID_POSITION_ERROR_MAX[7:0]
0x15D FOC. PID_TORQUE_ INTEGRATOR	PID_TORQUE_INTEGRATOR[31:24]
	PID_TORQUE_INTEGRATOR[23:16]
	PID_TORQUE_INTEGRATOR[15:8]
	PID_TORQUE_INTEGRATOR[7:0]
0x15E FOC. PID_FLUX_ INTEGRATOR	PID_FLUX_INTEGRATOR[31:24]
	PID_FLUX_INTEGRATOR[23:16]
	PID_FLUX_INTEGRATOR[15:8]
	PID_FLUX_INTEGRATOR[7:0]

ADDRESS AND NAME	FIELDS MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x15F FOC. PID_VELOCITY_ INTEGRATOR	PID_VELOCITY_INTEGRATOR[31:24]
	PID_VELOCITY_INTEGRATOR[23:16]
	PID_VELOCITY_INTEGRATOR[15:8]
	PID_VELOCITY_INTEGRATOR[7:0]
0x160 FOC. PID_POSITION_ INTEGRATOR	PID_POSITION_INTEGRATOR[31:24]
	PID_POSITION_INTEGRATOR[23:16]
	PID_POSITION_INTEGRATOR[15:8]
	PID_POSITION_INTEGRATOR[7:0]
0x161 FOC. PIDIN_TORQUE_FLUX_ TARGET	PIDIN_TORQUE_TARGET[15:8]
	PIDIN_TORQUE_TARGET[7:0]
	PIDIN_FLUX_TARGET[15:8]
	PIDIN_FLUX_TARGET[7:0]
0x162 FOC. PIDIN_VELOCITY_ TARGET	PIDIN_VELOCITY_TARGET[31:24]
	PIDIN_VELOCITY_TARGET[23:16]
	PIDIN_VELOCITY_TARGET[15:8]
	PIDIN_VELOCITY_TARGET[7:0]
0x163 FOC. PIDIN_POSITION_ TARGET	PIDIN_POSITION_TARGET[31:24]
	PIDIN_POSITION_TARGET[23:16]
	PIDIN_POSITION_TARGET[15:8]
	PIDIN_POSITION_TARGET[7:0]
0x164 FOC. PIDIN_TORQUE_FLUX_ TARGET_LIMITED	PIDIN_TORQUE_TARGET_LIMITED[15:8]
	PIDIN_TORQUE_TARGET_LIMITED[7:0]
	PIDIN_FLUX_TARGET_LIMITED[15:8]
	PIDIN_FLUX_TARGET_LIMITED[7:0]
0x165 FOC. PIDIN_VELOCITY_ TARGET_LIMITED	PIDIN_VELOCITY_TARGET_LIMITED[31:24]
	PIDIN_VELOCITY_TARGET_LIMITED[23:16]
	PIDIN_VELOCITY_TARGET_LIMITED[15:8]
	PIDIN_VELOCITY_TARGET_LIMITED[7:0]
0x166 FOC. PIDIN_POSITION_ TARGET_LIMITED	PIDIN_POSITION_TARGET_LIMITED[31:24]
	PIDIN_POSITION_TARGET_LIMITED[23:16]
	PIDIN_POSITION_TARGET_LIMITED[15:8]
	PIDIN_POSITION_TARGET_LIMITED[7:0]
0x167 FOC. FOC_IBETA_IALPHA	IBETA[15:8]
	IBETA[7:0]
	IALPHA[15:8]
	IALPHA[7:0]
0x168 FOC.FOC_IQ_ID	IQ[15:8]
	IQ[7:0]
	ID[15:8]
	ID[7:0]
0x169 FOC.FOC_UQ_UD	UQ[15:8]
	UQ[7:0]
	UD[15:8]
	UD[7:0]
0x16A FOC. FOC_UQ_UD_LIMITED	UQ_LIMITED[15:8]
	UQ_LIMITED[7:0]
	UD_LIMITED[15:8]
	UD_LIMITED[7:0]
0x16B FOC. FOC_UBETA_UALPHA	UBETA[15:8]
	UBETA[7:0]
	UALPHA[15:8]
	UALPHA[7:0]

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)							
0x16C FOC.FOC_UW_UU	UW[15:8]								
	UW[7:0]								
	UU[15:8]								
	UU[7:0]								
0x16D FOC.FOC_UV									
	UV[15:8]								
	UV[7:0]								
0x16E FOC.PWM_V_U	PWM_V[15:8]								
	PWM_V[7:0]								
	PWM_U[15:8]								
	PWM_U[7:0]								
0x16F FOC.PWM_W									
	PWM_W[15:8]								
	PWM_W[7:0]								
0x170 FOC.FOC_STATUS									
					FOC_STATUS				
0x171 FOC. U_S_ACTUAL_I_S_ ACTUAL	U_S_ACTUAL[15:8]								
	U_S_ACTUAL[7:0]								
	I_S_ACTUAL[15:8]								
	I_S_ACTUAL[7:0]								
0x172 FOC.P_MOTOR	P_MOTOR[31:24]								
	P_MOTOR[23:16]								
	P_MOTOR[15:8]								
	P_MOTOR[7:0]								
0x173 FOC.I_T_ACTUAL									
	I_T_ACTUAL[15:8]								
	I_T_ACTUAL[7:0]								
0x174 FOC. PRBS_AMPLITUDE	PRBS_AMPLITUDE[31:24]								
	PRBS_AMPLITUDE[23:16]								
	PRBS_AMPLITUDE[15:8]								
	PRBS_AMPLITUDE[7:0]								
0x175 FOC. PRBS_DOWN_ SAMPLING_RATIO									
	PRBS_DOWN_SAMPLING_RATIO								
0x180 BIQUAD.BIQUAD_EN									
							TORQUE_ EN	VELOCITY_ EN	
0x181 BIQUAD. VELOCITY_A1									
	COEFF_VELOCITY_A1[23:16]								
	COEFF_VELOCITY_A1[15:8]								
	COEFF_VELOCITY_A1[7:0]								
0x182 BIQUAD. VELOCITY_A2									
	COEFF_VELOCITY_A2[23:16]								
	COEFF_VELOCITY_A2[15:8]								
	COEFF_VELOCITY_A2[7:0]								

ADDRESS AND NAME	FIELDS							
	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)							
0x183 BIQUAD. VELOCITY_B0								
	COEFF_VELOCITY_B0[23:16]							
	COEFF_VELOCITY_B0[15:8]							
	COEFF_VELOCITY_B0[7:0]							
0x184 BIQUAD. VELOCITY_B1								
	COEFF_VELOCITY_B1[23:16]							
	COEFF_VELOCITY_B1[15:8]							
	COEFF_VELOCITY_B1[7:0]							
0x185 BIQUAD. VELOCITY_B2								
	COEFF_VELOCITY_B2[23:16]							
	COEFF_VELOCITY_B2[15:8]							
	COEFF_VELOCITY_B2[7:0]							
0x186 BIQUAD.TORQUE_A1								
	COEFF_TORQUE_A1[23:16]							
	COEFF_TORQUE_A1[15:8]							
	COEFF_TORQUE_A1[7:0]							
0x187 BIQUAD.TORQUE_A2								
	COEFF_TORQUE_A2[23:16]							
	COEFF_TORQUE_A2[15:8]							
	COEFF_TORQUE_A2[7:0]							
0x188 BIQUAD.TORQUE_B0								
	COEFF_TORQUE_B0[23:16]							
	COEFF_TORQUE_B0[15:8]							
	COEFF_TORQUE_B0[7:0]							
0x189 BIQUAD.TORQUE_B1								
	COEFF_TORQUE_B1[23:16]							
	COEFF_TORQUE_B1[15:8]							
	COEFF_TORQUE_B1[7:0]							
0x18A BIQUAD.TORQUE_B2								
	COEFF_TORQUE_B2[23:16]							
	COEFF_TORQUE_B2[15:8]							
	COEFF_TORQUE_B2[7:0]							
0x1C0 RAMPER. TIME_CONFIG	T_VMAX[15:8]							
	T_VMAX[7:0]							
	T_ZEROWAIT[15:8]							
	T_ZEROWAIT[7:0]							
0x1C1 RAMPER. SWITCH_MODE								
						STOP_ON_V_DEVIATION	STOP_ON_POS_DEVIATION	FORCE_HARD_STOP
	SOFT_STOP_EN	STALL_STOP_EN		LATCH_REF_H_INACTIVE	LATCH_REF_H_ACTIVE	LATCH_REF_R_INACTIVE	LATCH_REF_R_ACTIVE	LATCH_REF_L_INACTIVE
	LATCH_REF_L_ACTIVE	SWAP_RAMP_REF_LR	RAMP_REF_H_POL	RAMP_REF_R_POL	RAMP_REF_L_POL	RAMP_REF_H_STOP_EN	RAMP_REF_R_STOP_EN	RAMP_REF_L_STOP_EN
0x1C3 RAMPER.PHI_E	PHI_E_OFFSET[15:8]							
	PHI_E_OFFSET[7:0]							
	PHI_E[15:8]							
	PHI_E[7:0]							
0x1C4 RAMPER.A1								
		A1[22:16]						
	A1[15:8]							
	A1[7:0]							

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x1C5 RAMPER.A2		
	A2[22:16]	
	A2[15:8]	
	A2[7:0]	
0x1C6 RAMPER.A_MAX		
	A_MAX[22:16]	
	A_MAX[15:8]	
	A_MAX[7:0]	
0x1C7 RAMPER.D1		
	D1[22:16]	
	D1[15:8]	
	D1[7:0]	
0x1C8 RAMPER.D2		
	D2[22:16]	
	D2[15:8]	
	D2[7:0]	
0x1C9 RAMPER.D_MAX		
	D_MAX[22:16]	
	D_MAX[15:8]	
	D_MAX[7:0]	
0x1CA RAMPER.V_START		
	V_START[22:16]	
	V_START[15:8]	
	V_START[7:0]	
0x1CB RAMPER.V1		
	V1[26:24]	
	V1[23:16]	
	V1[15:8]	
	V1[7:0]	
0x1CC RAMPER.V2		
	V2[26:24]	
	V2[23:16]	
	V2[15:8]	
	V2[7:0]	
0x1CD RAMPER.V_STOP		
	V_STOP[22:16]	
	V_STOP[15:8]	
	V_STOP[7:0]	
0x1CE RAMPER.V_MAX		
	V_MAX[26:24]	
	V_MAX[23:16]	
	V_MAX[15:8]	
	V_MAX[7:0]	
0x1CF RAMPER. ACCELERATION		
	ACCELERATION[23:16]	
	ACCELERATION[15:8]	
	ACCELERATION[7:0]	
0x1D0 RAMPER.V_ACTUAL		
	V_ACTUAL[27:24]	
	V_ACTUAL[23:16]	
	V_ACTUAL[15:8]	
	V_ACTUAL[7:0]	
0x1D1 RAMPER.POSITION		
	POSITION[31:24]	
	POSITION[23:16]	
	POSITION[15:8]	
	POSITION[7:0]	

ADDRESS AND NAME	FIELDS				MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)			
0x1D2 RAMPER. POSITION_LATCH	POSITION_LATCH[31:24]							
	POSITION_LATCH[23:16]							
	POSITION_LATCH[15:8]							
	POSITION_LATCH[7:0]							
0x1D3 RAMPER. POSITION_ACTUAL_ LATCH	POSITION_ACTUAL_LATCH[31:24]							
	POSITION_ACTUAL_LATCH[23:16]							
	POSITION_ACTUAL_LATCH[15:8]							
	POSITION_ACTUAL_LATCH[7:0]							
0x1D4 RAMPER.STATUS								
							STALL_IN_ POSITION_ ERR	STALL_IN_ V_ERR
		T_ ZEROWAIT _ACTIVE	V_ZERO	POSITION_ REACHED_ STATUS	V_ REACHED_ STATUS			RAMP_ REF_H_ STOP_ STATUS
	RAMP_ REF_R_ STOP_ STATUS	RAMP_ REF_L_ STOP_ STATUS				RAMP_ REF_H_ STATUS	RAMP_ REF_R_ STATUS	RAMP_ REF_L_ STATUS
0x1D5 RAMPER.EVENTS								
			LATCH_ REF_R_ READY	LATCH_ REF_L_ READY	LATCH_ REF_H_ READY	SECOND_ MOVE_ EVENT	STALL_ STOP_ EVENT	POSITION_ REACHED_ EVENT
0x200 EXT_CTRL.VOLTAGE	UQ[15:8]							
	UQ[7:0]							
	UD[15:8]							
	UD[7:0]							
0x202 EXT_CTRL.PWM_V_U	PWM_V[15:8]							
	PWM_V[7:0]							
	PWM_U[15:8]							
	PWM_U[7:0]							
0x203 EXT_CTRL.PWM_W								
	PWM_W[15:8]							
	PWM_W[7:0]							
0x240 FEEDBACK. CONF_CH_A					SRC_SEL_A			
	CPR_INV_A[23:16]							
	CPR_INV_A[15:8]							
	CPR_INV_A[7:0]							
0x241 FEEDBACK. CONF_CH_B					SRC_SEL_B			
	CPR_INV_B[23:16]							
	CPR_INV_B[15:8]							
	CPR_INV_B[7:0]							
0x242 FEEDBACK. PHI_E_OFFSET								
	PHI_E_OFFSET[15:8]							
	PHI_E_OFFSET[7:0]							

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)							
0x243 FEEDBACK.LUT	RDATA								
	ADDR								
		LOOKUP_B_GAIN				LOOKUP_A_GAIN			
						LOOKUP_B_EN	LOOKUP_A_EN	SPLIT_MODE_EN	
0x244 FEEDBACK. VELOCITY_FRQ_CONF									
	VELOCITY_SCALING[15:8]								
	VELOCITY_SCALING[7:0]								
	VELOCITY_SAMPLING								VELOCITY_SYNC_SRC
0x245 FEEDBACK. VELOCITY_PER_CONF	POS_DEV_TIMER[15:8]								
	POS_DEV_TIMER[7:0]								
		POS_DEV_MIN[14:8]							
	POS_DEV_MIN[7:0]								
0x246 FEEDBACK. VELOCITY_PER_FILTER									
						FILTER_WIDTH			
0x247 FEEDBACK. PHI_CONVERTED	PHI_CONVERTED_B[15:8]								
	PHI_CONVERTED_B[7:0]								
	PHI_CONVERTED_A[15:8]								
	PHI_CONVERTED_A[7:0]								
0x248 FEEDBACK.CH_A	PHI_EXTRAPOLATED_AB[15:8]								
	PHI_EXTRAPOLATED_AB[7:0]								
	PHI_LOOKUP_A[15:8]								
	PHI_LOOKUP_A[7:0]								
0x249 FEEDBACK.CH_B									
	PHI_LOOKUP_B[15:8]								
	PHI_LOOKUP_B[7:0]								
0x24A FEEDBACK. VELOCITY_FRQ	VELOCITY_FRQ[31:24]								
	VELOCITY_FRQ[23:16]								
	VELOCITY_FRQ[15:8]								
	VELOCITY_FRQ[7:0]								
0x24B FEEDBACK. VELOCITY_PER	VELOCITY_PER[31:24]								
	VELOCITY_PER[23:16]								
	VELOCITY_PER[15:8]								
	VELOCITY_PER[7:0]								
0x24C FEEDBACK. LUT_WDATA	WDATA[31:24]								
	WDATA[23:16]								
	WDATA[15:8]								
	WDATA[7:0]								
0x24D FEEDBACK. PHI_EXT_A									
	PHI_EXT_A[23:16]								
	PHI_EXT_A[15:8]								
	PHI_EXT_A[7:0]								
0x24E FEEDBACK. PHI_EXT_B									
	PHI_EXT_B[23:16]								
	PHI_EXT_B[15:8]								
	PHI_EXT_B[7:0]								

ADDRESS AND NAME	FIELDS	MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)						
0x24F FEEDBACK. VELOCITY_EXT	VELOCITY_EXT[31:24]							
	VELOCITY_EXT[23:16]							
	VELOCITY_EXT[15:8]							
	VELOCITY_EXT[7:0]							
0x250 FEEDBACK. OUTPUT_CONF								
	VELOCITY_SELECTION		VELOCITY_ SRC	POSITION_ SRC			PHI_E_SRC	
	PHI_E_MUL_FACTOR							
0x251 FEEDBACK.PHI_E								
	PHI_E_FOC[15:8]							
	PHI_E_FOC[7:0]							
0x252 FEEDBACK. PHI_DIFF_LIMIT								
	PHI_DIFF_LIMIT[15:8]							
	PHI_DIFF_LIMIT[7:0]							
0x280 ABN.CONFIG	CPR[23:16]							
	CPR[15:8]							
	CPR[7:0]							
					INV_DIR		CLN	COMBINED _N
0x281 ABN.COUNT								
	COUNT[23:16]							
	COUNT[15:8]							
	COUNT[7:0]							
0x282 ABN. COUNT_N_CAPTURE								
	COUNT_N_CAPTURE[23:16]							
	COUNT_N_CAPTURE[15:8]							
	COUNT_N_CAPTURE[7:0]							
0x283 ABN. COUNT_N_WRITE								
	COUNT_N_WRITE[23:16]							
	COUNT_N_WRITE[15:8]							
	COUNT_N_WRITE[7:0]							
0x284 ABN.EVENTS								
							N_EVENT	INVALID_ SIGNAL_ EVENT
0x2C0 ABN2.CONFIG	CPR[23:16]							
	CPR[15:8]							
	CPR[7:0]							
					INV_DIR		CLN	COMBINED _N
0x2C1 ABN2.COUNT								
	COUNT[23:16]							
	COUNT[15:8]							
	COUNT[7:0]							
0x2C2 ABN2. COUNT_N_CAPTURE								
	COUNT_N_CAPTURE[23:16]							
	COUNT_N_CAPTURE[15:8]							
	COUNT_N_CAPTURE[7:0]							

ADDRESS AND NAME	FIELDS				MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)			
0x2C3 ABN2. COUNT_N_WRITE	COUNT_N_WRITE[23:16]							
	COUNT_N_WRITE[15:8]							
	COUNT_N_WRITE[7:0]							
0x2C4 ABN2.EVENTS								
							N_EVENT	INVALID_ SIGNAL_ EVENT
0x300 HALL.MAP_CONFIG								
		ANA_HALL_MAP				HALL_MAP		
0x301 HALL.DIG_COUNT								
						COUNT		
0x302 HALL.ANA_CONFIG								
		ADC_ CLIPPED	N_PULSE_ OUT	USE_N_ PULSE	TWO_ CYCLE_ MODE_EN	N_PULSE_ POL	N_PULSE_ EDGE	ANA_MODE
	N_PULSE_THRESHOLD[15:8]							
	N_PULSE_THRESHOLD[7:0]							
0x303 HALL. ANA_UX_CONFIG	UX_SCALE[15:8]							
	UX_SCALE[7:0]							
	UX_OFFSET[15:8]							
	UX_OFFSET[7:0]							
0x304 HALL. ANA_VN_CONFIG	VN_SCALE[15:8]							
	VN_SCALE[7:0]							
	VN_OFFSET[15:8]							
	VN_OFFSET[7:0]							
0x305 HALL. ANA_WY_CONFIG	WY_SCALE[15:8]							
	WY_SCALE[7:0]							
	WY_OFFSET[15:8]							
	WY_OFFSET[7:0]							
0x306 HALL.ANA_UX_OUT								
	UX_OUT[15:8]							
	UX_OUT[7:0]							
0x307 HALL.ANA_VN_OUT								
	VN_OUT[15:8]							
	VN_OUT[7:0]							
0x308 HALL.ANA_WY_OUT								
	WY_OUT[15:8]							
	WY_OUT[7:0]							
0x309 HALL.ANA_OUT	ANA_PHI[15:8]							
	ANA_PHI[7:0]							
	ANA_PHI_N_CAPTURE[15:8]							
	ANA_PHI_N_CAPTURE[7:0]							

ADDRESS AND NAME	FIELDS			MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)				
0x30A HALL.DIG_EVENTS								
								INVALID_ SIGNAL_ EVENT
0x340 UART.CONTROL	RTMI_SAMPLING							
		RTMI_EN	RTMI_CRC_ EN	NORMAL_ CRC_EN	RX_ FILTER_EN	AUTOBAUD _EN		TIMEOUT_ PRE_ DIVIDER_ EN
				MANTISSA_LIMIT[12:8]				
	MANTISSA_LIMIT[7:0]							
0x341 UART.TIMEOUT	COUNTER[15:8]							
	COUNTER[7:0]							
	LIMIT[15:8]							
	LIMIT[7:0]							
0x342 UART.STATUS								
				AUTOBAUD _VALID		TX_ACTIVE		RX_ACTIVE
0x343 UART.EVENTS								
	INVALID_ CRC_ EVENT	INVALID_ STOP_BIT_ EVENT	INVALID_ START_ BIT_EVENT	RX_IDLE_ TIMEOUT_ EVENT	RTMI_ INTERRUPT _EVENT	NOISY_ DATA_ EVENT		AUTOBAUD _SYNC_ FAIL_ EVENT
0x344 UART.RTMI_CH_0								
								CH_0_EN
							CH_0_ADDR[9:8]	
	CH_0_ADDR[7:0]							
0x345 UART.RTMI_CH_1								
								CH_1_EN
							CH_1_ADDR[9:8]	
	CH_1_ADDR[7:0]							
0x346 UART.RTMI_CH_2								
								CH_2_EN
							CH_2_ADDR[9:8]	
	CH_2_ADDR[7:0]							
0x347 UART.RTMI_CH_3								
								CH_3_EN
							CH_3_ADDR[9:8]	
	CH_3_ADDR[7:0]							
0x348 UART.RTMI_CH_4								
								CH_4_EN
							CH_4_ADDR[9:8]	
	CH_4_ADDR[7:0]							
0x349 UART.RTMI_CH_5								
								CH_5_EN
							CH_5_ADDR[9:8]	
	CH_5_ADDR[7:0]							

ADDRESS AND NAME	FIELDS							MSB (TOP LEFT) TO LSB (BOTTOM RIGHT)
0x34A UART.RTMI_CH_6								
								CH_6_EN
							CH_6_ADDR[9:8]	
	CH_6_ADDR[7:0]							
0x34B UART.RTMI_CH_7								
								CH_7_EN
							CH_7_ADDR[9:8]	
	CH_7_ADDR[7:0]							
0x380 IO_CONTROLLER. CONTROL								
								RESET
0x381 IO_CONTROLLER. COMMAND	COMMAND[31:24]							
	COMMAND[23:16]							
	COMMAND[15:8]							
	COMMAND[7:0]							
0x382 IO_CONTROLLER. RESPONSE_0	RESPONSE_0[31:24]							
	RESPONSE_0[23:16]							
	RESPONSE_0[15:8]							
	RESPONSE_0[7:0]							
0x383 IO_CONTROLLER. RESPONSE_1	RESPONSE_1[31:24]							
	RESPONSE_1[23:16]							
	RESPONSE_1[15:8]							
	RESPONSE_1[7:0]							
0x384 IO_CONTROLLER. RESPONSE_2	RESPONSE_2[31:24]							
	RESPONSE_2[23:16]							
	RESPONSE_2[15:8]							
	RESPONSE_2[7:0]							
0x385 IO_CONTROLLER. RESPONSE_3	RESPONSE_3[31:24]							
	RESPONSE_3[23:16]							
	RESPONSE_3[15:8]							
	RESPONSE_3[7:0]							

レジスタの詳細な説明

TMC6460 のすべてのレジスタの詳細な説明を以下に示します。アクセス性を示すレジスタ・フィールド・タイプは次のとおりです：R（読出し専用）、RW（読出し／書込み）、W1C（「1」を書き込んでクリア）、W1S（「1」を書き込んでセット）。

0x000：CHIP.ID

デフォルト：0x36343630

ダイの ID 番号。

ビットおよび名称	タイプおよびリセット	説明
[31:0] ID	R, unsigned 0x36343630	チップID

0x001：CHIP.VARIANT

デフォルト：0x00000000

ダイの品種。

ビットおよび名称	タイプおよびリセット	説明
[31:0] VARIANT	R, unsigned 0x00000000	チップ品種

0x002：CHIP.REVISION

デフォルト：0x00000013

ダイのリビジョン。

ビットおよび名称	タイプおよびリセット	説明
30:0] REVISION	R, unsigned 0x00000013	ICのリビジョン

0x004：CHIP.INPUTS_RAW

デフォルト：0x00000000

PWM_IN、DIR_IN、リファレンス・スイッチ入力、デジタル・ホール入力、デジタル ABN エンコーダ入力、I/O コントローラ入力、通信ピン入力への入力信号。システムのセットアップ時および検証時に直接読出しを行うために使用できる信号です。生の信号は、物理ピンで読み出される実際の入力レベルです。生でない信号は、ピン信号がマトリックスを介してルーティングされ、入力処理ステップを通過した後の、論理レベルです。

ビットおよび名称	タイプおよびリセット	説明
[31] PWM_IN	R 0x0	内部指定されたPWM_IN入力値
[30] DIR_IN	R 0x0	内部指定された方向入力値（I/Oコントローラに転送）。
[29] DIRECT_IN1	R 0x0	直接入力値1：内部指定された入力1（I/Oコントローラに転送）の入力値。
[28] DIRECT_IN0	R 0x0	直接入力値0：内部指定された入力0（I/Oコントローラに転送）の入力値。
[27] REF_H	R 0x0	内部指定されたリファレンス値REF_H
[26] REF_R	R 0x0	内部指定されたリファレンス値REF_R
[25] REF_L	R 0x0	内部指定されたリファレンス値REF_L
[24] HALL_W	R 0x0	内部指定されたホール信号W
[23] HALL_V	R 0x0	内部指定されたホール信号V

ビットおよび名称	タイプおよびリセット	説明
[22] HALL_U	R 0x0	内部指定されたホール信号U
[21] ENC2_N	R 0x0	ABN2モジュール用に内部指定されたエンコーダ信号N。
[20] ENC2_B	R 0x0	ABN2モジュール用に内部指定されたエンコーダ信号B。
[19] ENC2_A	R 0x0	ABN2モジュール用に内部指定されたエンコーダ信号A。
[18] ENC_N	R 0x0	ABNモジュール用に内部指定されたエンコーダ信号N。
[17] ENC_B	R 0x0	ABNモジュール用に内部指定されたエンコーダ信号B。
[16] ENC_A	R 0x0	ABNモジュール用に内部指定されたエンコーダ信号A。
[15] PIN_DRV_EN	R 0x0	DRV_ENピンでの無処理入力信号。
[13] PIN_UART_RXD_RAW	R 0x0	UART_RXDピンでの無処理入力信号。
[12] PIN_PWM_IN_RAW	R 0x0	PWM_INピンでの無処理入力信号。
[11] PIN_CSN_RAW	R 0x0	CSNピンでの無処理COM入力信号。
[10] PIN_SCK_RAW	R 0x0	SCKピンでの無処理入力信号。
[9] PIN_SDO_RAW	R 0x0	SDOピンでの無処理入力信号。
[8] PIN_SDI_RAW	R 0x0	SDIピンでの無処理入力信号。
[7] PIN_REF_R_RAW	R 0x0	REF_Rピンでの無処理入力信号。
[6] PIN_REF_L_RAW	R 0x0	REF_Lピンでの無処理入力信号。
[5] PIN_HALL_W_RAW	R 0x0	HALL_Wピンでの無処理入力信号。
[4] PIN_HALL_V_RAW	R 0x0	HALL_Vピンでの無処理入力信号。
[3] PIN_HALL_U_RAW	R 0x0	HALL_Uピンでの無処理入力信号。
[2] PIN_ENC_N_RAW	R 0x0	ENC_Nピンでの無処理入力信号。
[1] PIN_ENC_B_RAW	R 0x0	ENC_Bピンでの無処理入力信号。
[0] PIN_ENC_A_RAW	R 0x0	ENC_Aピンでの無処理入力信号。

0x005：CHIP.OUTPUTS_RAW

デフォルト：0x00000000

FAULTN、INT、PWM の各内部出力信号のロジック・レベル、およびデジタル差動 ABN エンコーダ信号のエラー・フラグのロジック・レベル。

ビットおよび名称	タイプおよびリセット	説明
[26] INVALID_DIFF_ENC_N	R 0x0	ENC_N入力でのデジタル差動エンコーダの故障。
[25] INVALID_DIFF_ENC_B	R 0x0	ENC_B入力でのデジタル差動エンコーダの故障。
[24] INVALID_DIFF_ENC_A	R 0x0	ENC_A入力でのデジタル差動エンコーダの故障。
[7] FAULTN_OUT	R 0x0	内部FAULTNの出力値
[6] INT_OUT	R 0x0	内部INTの出力値
[5] PWM_W_H	R 0x0	OUT3のハイサイドでの内部デジタルPWM出力値。
[4] PWM_W_L	R 0x0	OUT3のローサイドでの内部デジタルPWM出力値。
[3] PWM_V_H	R 0x0	OUT2のハイサイドでの内部デジタルPWM出力値。
[2] PWM_V_L	R 0x0	OUT2のローサイドでの内部デジタルPWM出力値。
[1] PWM_U_H	R 0x0	OUT1のハイサイドでの内部デジタルPWM出力値。
[0] PWM_U_L	R 0x0	OUT1のローサイドでの内部デジタルPWM出力値。

0x006：CHIP.IO_MATRIX

デフォルト：0x00000000

すべての IO ピンに対する代替機能の割り当て。I/O コントローラに関連する一部の代替機能では、ピン・ロジックが入力から出力に変化する場合がある点に注意してください。

ビットおよび名称	タイプおよびリセット	説明
[31:30] PIN_FAULTN	RW 0x0	FAULTNピンの機能の割り当て
	0: FAULTN_OD	メイン機能：デジタル出力FAULTN、オープン・ドレイン。 CHIP.FAULTN_INT_MASKと組み合わせたCHIP.STATUS_FLAGSによって制御。
	1: INT_OD	代替機能1：デジタル出力INT、オープン・ドレイン。CHIP.FAULTN_INT_MASKと組み合わせたCHIP.EVENTSによって制御。
	2: FAULTN_PP	代替機能2：デジタル出力FAULTN、プッシュプル。CHIP.FAULTN_INT_MASKと組み合わせたCHIP.STATUS_FLAGSによって制御。
	3: INT_PP	代替機能3：デジタル出力INT、プッシュプル。CHIP.FAULTN_INT_MASKと組み合わせたCHIP.EVENTSによって制御。
[29:28] PIN_UART_TXD	RW 0x0	UART_TXDピンの機能の割り当て
	0: UART_TXD	メイン機能：デジタル出力TXD。UART通信のインターフェース。
	1: BRAKE	代替機能1：デジタルBRAKE出力。
	2: DIRECT_OUT0	代替機能2：I/Oコントローラのデジタル出力0。
	3: DIRECT_OUT1	代替機能3：I/Oコントローラのデジタル出力1。
[27:26] PIN_UART_RXD	RW 0x0	UART_RXDピンの機能の割り当て
	0: UART_RXD	メイン機能：デジタル入力RXD。UART通信のインターフェース。
	1: DIR_IN	代替機能1：I/Oコントローラのデジタル入力DIR_IN。
	3: DIRECT_IN0	代替機能3：I/Oコントローラのデジタル入力0。

ビットおよび名称	タイプおよびリセット	説明
[25:24] PIN_PWM_IN	RW 0x0	PWM_INピンの機能の割り当て
	0: PWM_IN	メイン機能：I/Oコントローラのデジタル入力PWM_IN。
	1: AIN	代替機能1：AINの値のアナログ入力。
	3: DIRECT_IN1	代替機能3：I/Oコントローラのデジタル入力1。
[23] PIN_SCK	RW 0x0	SCKピンの機能の割り当て
	0: SCK	メイン機能：デジタル入力SCK。SPI通信のインターフェース。
	1: DIRECT_IN1	代替機能1：I/Oコントローラのデジタル入力1。
[22:21] PIN_SDO	RW 0x0	SDOピンの機能の割り当て
	0: SDO	メイン機能：デジタル出力SDO。SPI通信のインターフェース。
	1: BRAKE	代替機能1：デジタルBRAKE出力。
	3: DIRECT_OUT0	代替機能3：I/Oコントローラのデジタル出力0。
[20:19] PIN_SDI	RW 0x0	SDIピンの機能の割り当て
	0: SDI	メイン機能：デジタル入力SDI。SPI通信のインターフェース。
	1: DIR_IN	代替機能1：I/Oコントローラのデジタル入力DIR_IN。
	3: DIRECT_IN0	代替機能3：I/Oコントローラのデジタル入力0。
[18:17] PIN_REF_R	RW 0x0	REF_Rピンの機能の割り当て。
	0: REF_R	メイン機能：デジタル入力REF_R。
	1: REF_H	代替機能1：デジタル入力選択。
	2: BRAKE	代替機能2：デジタルBRAKE出力。
	3: DIRECT_OUT1	代替機能3：I/Oコントローラのデジタル出力1。
[16:15] PIN_REF_L	RW 0x0	REF_Lピンの機能の割り当て
	0: REF_L	メイン機能：デジタル入力REF_L。
	1: REF_H	代替機能1：デジタル入力選択。
	3: DIRECT_IN0	代替機能3：I/Oコントローラのデジタル入力0。
[14:12] PIN_HALL_W	RW 0x0	HALL_Wピンの機能の割り当て
	0: HALL_W	メイン機能：デジタル・ホール入力W。
	1: AINP_W	代替機能1：アナログ・ホールWまたはSinCos Yの正のアナログ入力。
	2: ENC2_N	代替機能2：デジタル第二エンコーダ入力N。
	3: DIRECT_OUT2	代替機能3：I/Oコントローラのデジタル出力2。
	4: REF_H	代替機能4：デジタルREF_H入力。
	6: DIG_DIFF_ENC_N_MONITOR	代替機能6：ENC_Nの負の差動デジタル入力。モニタリング専用。
	7: DIG_DIFF_ENC_N_N	代替機能7：ENC_Nの負の差動デジタル入力。
[11:9] PIN_HALL_V	RW 0x0	HALL_Vピンの機能の割り当て
	0: HALL_V	メイン機能：デジタル・ホール入力V。
	1: AINP_V	代替機能1：アナログ・ホールVまたはSinCos Nの正のアナログ入力。
	2: ENC2_B	代替機能2：デジタル第二エンコーダ入力B。
	3: SCL	代替機能3：I/Oコントローラのデジタル出力。I2C通信インターフェースのSCL。プッシュプル。
	4: BRAKE	代替機能4：デジタルBRAKE出力。
	5: SCL_OD	代替機能5：I/Oコントローラのデジタル出力。I2C通信インターフェースのSCL。オープン・ドレイン。
	6: DIG_DIFF_ENC_B_MONITOR	代替機能6：ENC_Bの負の差動デジタル入力。モニタリング専用。
	7: DIG_DIFF_ENC_B_N	代替機能7：ENC_Bの負の差動デジタル入力。

ビットおよび名称	タイプおよびリセット	説明
[8:6] PIN_HALL_U	RW 0x0	HALL_Uピンの機能の割り当て
	0: HALL_U	メイン機能：デジタル・ホール入力U。
	1: AINP_U	代替機能1：アナログ・ホールUまたはSinCos Xの正のアナログ入力。
	2: ENC2_A	代替機能2：デジタル第二エンコーダ入力A。
	3: SDA	代替機能3：I/Oコントローラのデジタル IO。I²C通信インターフェースのSDA。
	6: DIG_DIFF_ENC_A_MONITOR	代替機能6：ENC_Aの負の差動デジタル入力。モニタリング専用。
	7: DIG_DIFF_ENC_A_N	代替機能7：ENC_Aの負の差動デジタル入力。
[5:4] PIN_ENC_N	RW 0x0	ENC_Nピンの機能の割り当て
	0: ENC_N	メイン機能：デジタル・エンコーダ入力N。
	1: AINN_W	代替機能1：アナログ・ホールWまたはSinCos Yの負のアナログ入力。
	2: HALL_W	代替機能2：デジタル・ホール入力W。
	3: DIRECT_IN1_OUT1	代替機能3：I/Oコントローラのデジタル入力1または出力1。 I/Oの割り当ては、I/Oコントローラの割り当てにより定められます。
[3:2] PIN_ENC_B	RW 0x0	ENC_Bピンの機能の割り当て
	0: ENC_B	メイン機能：デジタル・エンコーダ入力B。
	1: AINN_V	代替機能1：アナログ・ホールVまたはSinCos Nの負のアナログ入力。
	2: HALL_V	代替機能2：デジタル・ホール入力V。
	3: DIRECT_IN0	代替機能3：I/Oコントローラのデジタル入力0。
[1:0] PIN_ENC_A	RW 0x0	ENC_Aピンの機能の割り当て
	0: ENC_A	メイン機能：デジタル・エンコーダ入力A。
	1: AINN_U	代替機能1：アナログ・ホールUまたはSinCos Xの負のアナログ入力。
	2: HALL_U	代替機能2：デジタル・ホール入力U。
	3: DIRECT_OUT0	代替機能3：I/Oコントローラのデジタル出力0。

0x007：CHIP.IO_PU_PD

デフォルト：0x075800FF

プルアップ抵抗およびプルダウン抵抗をイネーブルまたはディスエーブルします。このレジスタに書き込む場合、不使用ビットのデフォルト値は維持しておく必要があります。

ビットおよび名称	タイプおよびリセット	説明
[27] PIN_UART_RXD_PD	RW 0x0	UART_RXDピンのプルダウンをイネーブルします
[26] PIN_PWM_IN_PD	RW 0x1	PWM_INピンのプルダウンをイネーブルします
[25] PIN_SCK_PD	RW 0x1	SCKピンのプルダウンをイネーブルします
[24] PIN_SDI_PD	RW 0x1	SDIピンのプルダウンをイネーブルします
[22] PIN_FAULTN_PU	RW 0x1	FAULTNピンのプルアップをイネーブルします。オープン・ドレイン割り当て時のみアクティブです。
[20] PIN_SDO_PU	RW 0x1	SDOピンのプルアップをイネーブルします。これがアクティブになるのは、CSNが「1」（SPI通信）の場合、または、I/Oコントローラの出力イネーブルがDIRECT_OUT0に対しアクティブでない場合のみです。
[19] PIN_UART_RXD_PU	RW 0x1	UART_RXDピンのプルアップをイネーブルします
[18] PIN_PWM_IN_PU	RW 0x0	PWM_INピンのプルアップをイネーブルします
[17] PIN_SCK_PU	RW 0x0	SCKピンのプルアップをイネーブルします

ビットおよび名称	タイプおよびリセット	説明
[16] PIN_SDI_PU	RW 0x0	SDIピンのプルアップをイネーブルします
[15] PIN_REF_R_PD	RW 0x0	REF_Rピンのプルダウンをイネーブルします
[14] PIN_REF_L_PD	RW 0x0	REF_Lピンのプルダウンをイネーブルします
[13] PIN_HALL_W_PD	RW 0x0	HALL_Wピンのプルダウンをイネーブルします
[12] PIN_HALL_V_PD	RW 0x0	HALL_Vピンのプルダウンをイネーブルします
[11] PIN_HALL_U_PD	RW 0x0	HALL_Uピンのプルダウンをイネーブルします
[10] PIN_ENC_N_PD	RW 0x0	ENC_Nピンのプルダウンをイネーブルします
[9] PIN_ENC_B_PD	RW 0x0	ENC_Bピンのプルダウンをイネーブルします
[8] PIN_ENC_A_PD	RW 0x0	ENC_Aピンのプルダウンをイネーブルします
[7] PIN_REF_R_PU	RW 0x1	REF_Rピンのプルアップをイネーブルします
[6] PIN_REF_L_PU	RW 0x1	REF_Lピンのプルアップをイネーブルします
[5] PIN_HALL_W_PU	RW 0x1	HALL_Wピンのプルアップをイネーブルします
[4] PIN_HALL_V_PU	RW 0x1	HALL_Vピンのプルアップをイネーブルします
[3] PIN_HALL_U_PU	RW 0x1	HALL_Uピンのプルアップをイネーブルします
[2] PIN_ENC_N_PU	RW 0x1	ENC_Nピンのプルアップをイネーブルします
[1] PIN_ENC_B_PU	RW 0x1	ENC_Bピンのプルアップをイネーブルします
[0] PIN_ENC_A_PU	RW 0x1	ENC_Aピンのプルアップをイネーブルします

0x008：CHIP.IO_CONFIG

デフォルト：0x70055000

一部のピンの追加入力処理機能の有効化および設定を行います。例えば、アナログ入力の分圧器の選択、デジタル入力レベルの反転、スプリアス信号を除去するデジタル・フィルタの有効化などです。

ビットおよび名称	タイプおよびリセット	説明
[31:28] DIFF_DELAY	RW, unsigned 0x7	デジタル差動入力の空白時間（単位：クロック・サイクル）：DIG_DIFF_のモニタリング時の入力でのレベル変化後、この時間内ではフォルトが生成されません。
[25] USE_PIN_REF_R_AS_REF_L	RW 0x0	REF_Rピンの入力信号は、内部指定されたREF_L信号にも割り当てられます。例えば、両方向を1つのピンで停止できるようにするなど、REF_R信号とREF_L信号の両方を1つの物理ピンREF_Rのみで使用できるようにするために用います。
[24] USE_PIN_REF_L_AS_REF_R	RW 0x0	REF_Lピンの入力信号は、内部指定されたREF_R信号にも割り当てられます。例えば、両方向を1つのピンで停止できるようにするなど、REF_R信号とREF_L信号の両方を1つの物理ピンREF_Lのみで使用できるようにするために用います。

ビットおよび名称	タイプおよびリセット	説明
[23:22] ANA_DIV_VCCIOF	RW 0x0	V_{CC_IOF}領域でのアナログ入力（ENC_xとHALL_xの各ピン）の分圧比。
	0: DIV_BY_4P5	アナログ入力を4.5の比で分圧
	1: DIV_BY_3P0	アナログ入力を3.0の比で分圧
	2: DIV_BY_2P2	アナログ入力を2.2の比で分圧
	3: DIV_BY_1P0	アナログ入力を1.0の比で分圧
[21:20] ANA_DIV_VCCIO	RW 0x0	V_{CC_IO}領域でのアナログ入力（PWM_INとTEMPの各ピン）の分圧比。
	0: DIV_BY_4P5	アナログ入力を4.5の比で分圧
	1: DIV_BY_3P0	アナログ入力を3.0の比で分圧
	2: DIV_BY_2P2	アナログ入力を2.2の比で分圧
	3: DIV_BY_1P0	アナログ入力を1.0の比で分圧
[19:18] PWM_IN_FLT	RW 0x1	PWM_IN入力に割り当てられるデジタル・フィルタの設定。
	0: NO_FLT	フィルタなし
	1: FLT_1M	1MHzのデジタル・フィルタ
	2: FLT_500K	500kHzのデジタル・フィルタ
	3: FLT_100K	100kHzのデジタル・フィルタ
[17:16] REF_FLT	RW 0x1	リファレンス入力ピンのデジタル・フィルタの設定。
	0: NO_FLT	フィルタなし
	1: FLT_1M	1MHzのデジタル・フィルタ
	2: FLT_500K	500kHzのデジタル・フィルタ
	3: FLT_100K	100kHzのデジタル・フィルタ
[15:14] HALL_FLT	RW 0x1	ホール入力のデジタル・フィルタの設定。
	0: NO_FLT	フィルタなし
	1: FLT_1M	1MHzのデジタル・フィルタ
	2: FLT_500K	500kHzのデジタル・フィルタ
	3: FLT_100K	100kHzのデジタル・フィルタ
[13:12] ENC_FLT	RW 0x1	エンコーダ入力のデジタル・フィルタ。
	0: NO_FLT	フィルタなし
	1: FLT_1M	1MHzのデジタル・フィルタ
	2: FLT_500K	500kHzのデジタル・フィルタ
	3: FLT_100K	100kHzのデジタル・フィルタ
[11] INV_PIN_RXD	RW 0x0	受信するPIN_RXD_RAW入力信号を反転して、更に処理を行います。
[10] INV_PIN_PWM_IN	RW 0x0	受信するPIN_PWM_IN_RAW入力信号を反転して、更に処理を行います。
[9] INV_PIN_SCK	RW 0x0	受信するPIN_SCK_RAW入力信号を反転して、更に処理を行います。SPI信号には適用されません。
[8] INV_PIN_SDI	RW 0x0	受信するPIN_SDI_RAW入力信号を反転して、更に処理を行います。SPI信号には適用されません。
[7] INV_PIN_REF_R	RW 0x0	受信するPIN_REF_R_RAW入力信号を反転して、更に処理を行います。
[6] INV_PIN_REF_L	RW 0x0	受信するPIN_REF_L_RAW入力信号を反転して、更に処理を行います。
[5] INV_PIN_HALL_W	RW 0x0	受信するPIN_HALL_W_RAW入力信号を反転して、更に処理を行います。
[4] INV_PIN_HALL_V	RW 0x0	受信するPIN_HALL_V_RAW入力信号を反転して、更に処理を行います。
[3] INV_PIN_HALL_U	RW 0x0	受信するPIN_HALL_U_RAW入力信号を反転して、更に処理を行います。

ビットおよび名称	タイプおよびリセット	説明
[2] INV_PIN_ENC_N	RW 0x0	受信するPIN_ENC_N_RAW入力信号を反転して、更に処理を行います。
[1] INV_PIN_ENC_B	RW 0x0	受信するPIN_ENC_B_RAW入力信号を反転して、更に処理を行います。
[0] INV_PIN_ENC_A	RW 0x0	受信するPIN_ENC_A_RAW入力信号を反転して、更に処理を行います。

0x009：CHIP.STATUS_FLAGS

デフォルト：0x00000000

個々の、あるいはマージされた、エラー状態およびステータスを示す様々なビット。このレジスタは、対応するステータス・ビットのアクティブ化された状態をラッチする、CHIP.EVENTS レジスタのベースとなるものです。

ビットおよび名称	タイプおよびリセット	説明
[31] GDRV_ON_STATE	R 0x0	GDRV_ONステータス・ビット：ゲート・ドライバはアクティブです。特定のエラー条件が満たされる場合、この状態はSYS_READYのままになります。
[30] SYS_READY_STATE	R 0x0	SYS_READYステータス・ビット：チップの基本的なレディ状態はアクティブです。この状態に達するのは、パワーアップ後ゲート・ドライバ段がアクティブになる（したがってDRV_ENピンとDRV_EN_BITが「1」に設定される）前、および、特定のエラー条件によりGDRV_ON状態が自動的にそのままになっている場合です。
[29] OVERTEMP_WARN_STATUS	R 0x0	内部取得または外部取得された温度が、レジスタベースの過熱制限値を超過しています。
[28] OV_VM_LIMIT_FAIL_STATUS	R 0x0	取得されたVM電圧がレジスタベースの過電圧制限値（UPPER_SWITCH_LIMIT）を超過しています。
[27] VM_TEMP_CLIPPED_STATUS	R 0x0	VMや外部／内部温度値が最後のアクイジション・サイクルの間にクリップされています。
[26] AIN_CLIPPED_STATUS	R 0x0	アナログ入力電流値がクリップされていることを示すステータス・ビット
[25] I_CLIPPED_STATUS	R 0x0	いずれかの相電流値が最後の電流アクイジション・サイクルの間にクリップされています。
[24] CURRENT_OVERLOAD_STATUS	R 0x0	実際の電流が、レジスタベースの電流過負荷制限値を超えています。
[23] HALL_FAIL_STATUS	R 0x0	ホール・フェイル・ステータス・ビット：3つのホール信号（U、V、W）がすべて同じ値です。
[22] ABN_FAIL_STATUS	R 0x0	ABNフェイル・ステータス・ビット。ABNとABN2を連結。特定のABN.EVENTSレジスタを参照してください。
[21] RAMP_REF_H_STATUS	R 0x0	ランプ・モジュールのアクティブなREF_H条件。
[20] RAMP_REF_LR_STATUS	R 0x0	ランプ・モジュールのアクティブなREF_LまたはREF_R条件。
[19] RAMP_V_ZERO_STATUS	R 0x0	ランプV_ZEROステータス・ビット：速度値が0になっています。
[18] RAMP_V_REACHED_STATUS	R 0x0	ランプ速度到達のステータス・ビット：速度値が実際のポジション値に一致しています。
[17] RAMP_TARGET_REACHED_STATUS	R 0x0	ランプ目標到達のステータス・ビット：目標値が実際のポジション値に一致しています。
[16] RAMP_STALL_FAIL_STATUS	R 0x0	アクティブなランプ・ストール条件：ポジション値や速度値が指定された制限値を超過しています。
[15] IO_CONTROLLER_STATUS	R 0x0	I/Oコントローラがディスエーブル、またはI/Oコントローラの生成したステータス・ビットがアクティブ。
[14] DIFF_ENC_FAIL_STATUS	R 0x0	差動エンコーダ・フェイル・ステータス・ビット：デジタル（および差動に指定された）エンコーダ入力が差動になっていません。

ビットおよび名称	タイプおよびリセット	説明
[13] OT_FAIL_STATUS	R 0x0	過熱フェイル・ステータス・ビット：いずれかの位相での内部測定温度が制限値を超えています。ドライバ出力段は、この過熱故障がアクティブである限りディスエーブルされます。
[12] SHRT_FAIL_STATUS	R 0x0	短絡フェイル・ステータス・ビット：いずれの位相出力においても地絡／V_Sへの短絡がモニタされます。
[11] VEL_FAIL_STATUS	R 0x0	出力周波数が599Hzの制限値を超過。
[10] TEMP_LIMIT_FAIL_STATUS	R 0x0	ダイのサーマル・シャットダウン・フェイル・ステータス・ビット：内部モニタされた温度が制限値を超えています。
[9] ADC_FAIL_STATUS	R 0x0	ADCフェイル・ステータス・ビット。ADC.STATUSレジスタを参照してください。
[8] EXT_RES_FAIL_STATUS	R 0x0	外部抵抗フェイル・ステータス・ビット：外部抵抗値が所定範囲外になっています。
[7] CLK_PLL_FAIL_STATUS	R 0x0	クロック／PLLフェイル・ステータス・ビット
[6] POWER_FAIL_STATUS	R 0x0	電圧フェイル・ステータス・ビット：VMやチャージ・ポンプが正しく起動しません。
[5] UV_VM_STATUS	R 0x0	低電圧VMフェイル・ステータス・ビット：VM電圧レベルが低すぎます。
[4] CP_FAIL_STATUS	R 0x0	チャージ・ポンプ・フェイル・ステータス・ビット：
[3] VCC_IOF_FAIL_STATUS	R 0x0	V_{CCIOF}の電圧レベルが低すぎます。
[2] UART_FAIL_STATUS	R 0x0	UARTフェイル・ステータス・ビット：何らかのUART通信エラーがアクティブです。
[1] SPI_FAIL_STATUS	R 0x0	SPIフェイル・ステータス：タイムアウトまたは割込み故障がアクティブです。
[0] PWRUP_FAIL_STATUS	R 0x0	電源投入プロセスに失敗しました。

0x00A：CHIP.EVENTS

デフォルト：0x00000001

CHIP.STATUS_FLAGS レジスタにおいてアクティブ化された対応するステータス・ビットのラッチされた状態。各々のイベントは、レジスタ・アクセスを用いてアクティブにクリアされるまで、セットされたままになります。セットされたビットのクリアは、その特定のビットに「1」を書き込むことによって実行できます。

ビットおよび名称	タイプおよびリセット	説明
[31] GDRV_ON_EVENT	RW, W1C 0x0	GDRV_ONイベント（特定のステータス・ビットに基づく）。
[30] SYS_READY_EVENT	RW, W1C 0x0	SYS_READYイベント（特定のステータス・ビットに基づく）。
[29] OVERTEMP_WARN_EVENT	RW, W1C 0x0	内部または外部過熱制限値フェイル・イベント（特定のステータス・ビットに基づく）。
[28] OV_VM_LIMIT_FAIL_EVENT	RW, W1C 0x0	過電圧（VM）制限値フェイル・イベント（特定のステータス・ビットに基づく）。
[27] VM_TEMP_CLIPPED_EVENT	RW, W1C 0x0	VMまたは温度値クリップ・イベント（特定のステータス・ビットに基づく）。
[26] AIN_CLIPPED_EVENT	RW, W1C 0x0	アナログ入力電流値クリップ・イベント（特定のステータス・ビットに基づく）。
[25] I_CLIPPED_EVENT	RW, W1C 0x0	相電流値クリップ・イベント（特定のステータス・ビットに基づく）。
[24] CURRENT_OVERLOAD_EVENT	RW, W1C 0x0	レジスタベースの電流過負荷制限値を超過。
[23] HALL_FAIL_EVENT	RW, W1C 0x0	ホール・フェイル・イベント：HALL.DIG_EVENTSレジスタでクリアしてからこのフィールドをクリアします。

ビットおよび名称	タイプおよびリセット	説明
[22] ABN_FAIL_EVENT	RW, W1C 0x0	ABNフェイル・イベント。ABNとABN2を連結（特定のステータス・ビットに基づく）。特定のABN.EVENTSレジスタでクリアしてからこのフィールドをクリアします。
[21] RAMP_REF_H_EVENT	RW, W1C 0x0	ランプ・ホーム・イベント（特定のステータス・ビットに基づく）。
[20] RAMP_REF_LR_EVENT	RW, W1C 0x0	ランプ・リファレンス・イベント（特定のステータス・ビットに基づく）。
[19] RAMP_V_ZERO_EVENT	RW, W1C 0x0	ランプV_ZEROイベント（特定のステータス・ビットに基づく）。
[18] RAMP_V_REACHED_EVENT	RW, W1C 0x0	ランプ速度到達イベント（特定のステータス・ビットに基づく）。
[17] RAMP_TARGET_REACHED_ EVENT	RW, W1C 0x0	ランプ目標到達イベント（特定のステータス・ビットに基づく）。
[16] RAMP_STALL_FAIL_EVENT	RW, W1C 0x0	ランプ・ストール・イベント（特定のステータス・ビットに基づく）。
[15] IO_CONTROLLER_EVENT	RW, W1C 0x0	I/Oコントローラ・イベント（特定のステータス・ビットに基づく）。
[14] DIFF_ENC_FAIL_EVENT	RW, W1C 0x0	差動エンコーダ・フェイル・イベント（特定のステータス・ビットに基づく）。
[13] OT_FAIL_EVENT	RW, W1C 0x0	過熱フェイル・イベント（特定のステータス・ビットに基づく）。
[12] SHRT_FAIL_EVENT	RW, W1C 0x0	短絡フェイル・イベント（特定のステータス・ビットに基づく）。
[11] VEL_FAIL_EVENT	RW, W1C 0x0	速度フェイル・イベント（特定のステータス・ビットに基づく）。
[10] TEMP_LIMIT_FAIL_EVENT	RW, W1C 0x0	ダイ・サーマル・シャットダウン・フェイル・イベント
[9] ADC_FAIL_EVENT	RW, W1C 0x0	ADCフェイル・イベント（特定のステータス・ビットに基づく）。
[8] EXT_RES_FAIL_EVENT	RW, W1C 0x0	外部抵抗フェイル・イベント（特定のステータス・ビットに基づく）。
[7] CLK_PLL_FAIL_EVENT	RW, W1C 0x0	クロック／PLLフェイル・イベント（特定のステータス・ビットに基づく）。
[6] POWER_FAIL_EVENT	RW, W1C 0x0	電圧フェイル・イベント（特定のステータス・ビットに基づく）。
[5] UV_VM_EVENT	RW, W1C 0x0	低電圧VMフェイル・イベント（特定のステータス・ビットに基づく）。
[4] CP_FAIL_EVENT	RW, W1C 0x0	チャージ・ポンプ・フェイル・イベント（特定のステータス・ビットに基づく）。
[3] VCCIOF_FAIL_EVENT	RW, W1C 0x0	VCCIOFフェイル・イベント（特定のステータス・ビットに基づく）。
[2] UART_FAIL_EVENT	RW, W1C 0x0	UARTフェイル・イベント
[1] SPI_FAIL_EVENT	RW, W1C 0x0	SPIフェイル・イベント（特定のステータス・ビットに基づく）。
[0] RST_EVENT	RW, W1C 0x1	リセット・イベント：リセットがトリガされています。

0x00B：CHIP.FAULTN_INT_MASK

デフォルト：0x00000000

CHIP.STATUS_FLAGS レジスタまたは CHIP.EVENTS レジスタのどちらのビットが FAULTN ピンを駆動するかを選択するためのマスク。CHIP.IO_MATRIX.PIN_FAULTN フィールドの値は、ステータス・フラグ・ビット（FAULTN_x オプション）またはイベント・ビット（INT_x オプション）を使用して出力を駆動するかどうかを決定します。

ビットおよび名称	タイプおよびリセット	説明
[31:0] FAULTN_INT_MASK	RW, unsigned 0x00000000	FAULTN出力ピンを駆動するビットを選択するためのマスク。マスクのビットを「1」に設定すると、対応するCHIP.STATUS_FLAGSレジスタまたはCHIP.EVENTSレジスタのビットが選択されます。選択するレジスタは、CHIP.IO_MATRIX.PIN_FAULTNフィールドの値で決まります。マスクにおいて2個以上のビットが設定されている場合、対応するビットすべてがOR接続されます。

0x00C：CHIP.SPI_STATUS_MASK

デフォルト：0x00000000

各 SPI コマンド応答で CHIP.EVENTS レジスタのどのビットが含まれるかを選択するためのマスク。最大 6 ビットを選択できます。

ビットおよび名称	タイプおよびリセット	説明
[31:0] SPI_STATUS_MASK	RW, unsigned 0x00000000	各SPIコマンド応答に含まれるビットを選択するためのマスク。マスクのビットを「1」に設定すると、対応するCHIP.EVENTSレジスタのビットが選択されます。最大6ビットを選択できます。これより多くのビットが選択された場合、ビット番号の小さい方が優先されます。

0x040：CLK_CTRL.CONFIG

デフォルト：0x00000E3C

システム・クロックの設定。

ビットおよび名称	タイプおよびリセット	説明
[12:8] CLOCK_DIVIDER	RW, unsigned 0x0E	PLLのソース・クロックのクロック分周比。PLLの入力周波数は1MHzでなくてはなりません。デフォルト値14は、内部発振器（15MHz）用に設定されています。pll_input_freq = source_clk_freq/(CLOCK_DIVIDER + 1)
[5] CLK_FSM_EN	RW 0x1	クロックおよびPLLコントローラFSMをイネーブル。設定が正しくないためにFSMが特定のポイントでスタックした場合、このビットを手動でクリアしてFSMをリセットしてください。
[4] PWM_CLK_EN	RW 0x1	120MHzのPWMクロックをイネーブルします。
[3] ADC_CLK_EN	RW 0x1	60MHzのADCクロックをイネーブルします。
[2] PLL_EN	RW 0x0	システムのメイン・クロックとして、内部発振器クロック（15MHz）またはPLL出力クロック（120MHz）を選択します。
	0: INT_OSC	内部発振器クロック（15MHz）をシステム・クロック、PWMクロック、ADCクロックのソースとして使用します。
	1: PLL	PLL出力クロック（120MHz）をシステム・クロック、PWMクロック、ADCクロックのソースとして使用します。 システム・クロック周波数 = 60MHz ADCクロック周波数 = 60MHz PWMクロック周波数 = 120MHz
[1] PLL_SRC	RW 0x0	PLLの入力ソースとして、内部発振器クロック（15MHz）または外部クロックを選択します。
	0: INT_CLK	内部発振器クロック（15MHz）をPLL入力クロックとして使用します。
	1: EXT_CLK	外部クロックをPLL入力クロックとして使用します。
[0] COMMIT	RW, W1S 0x0	このビットに1を書き込むことによって、クロックおよびPLLの設定を適用します。クロック設定の終了後、このビットは自動的にクリアされます。

0x041：CLK_CTRL.STATUS

デフォルト：0x00000000

システム・クロックの一部に関連する、ステータス・ビットおよびステータス・フィールド。

ビットおよび名称	タイプおよびリセット	説明
[31:24] CLK_FSM_FLG	R, unsigned 0x00	このビットフィールドは、クロックFSMの最新の状態を保持します。
[23:16] CLK_FSM_ERR_RPT	RW 0x0	CLK_FSM_ERRの場合、これらのフィールドがエラー状態をレポートします。
	0: NO_ERR_RPT	エラーは生じていません。
	1: NOT_USED	
	2: DET_ERR_RPT	PLL入力クロック用の1MHzは検出されず、外部クロックがソースとして用いています。
	4: STK_ERR_RPT	PLL出力クロックがスタックしています。
	8: PLL_ERR_RPT	PLLアナログ・エラーが発生しました。
	255: UNK_ERR_RPT	未知のエラー。
[6] PLL_READY	R 0x0	PLL設定が終了すると、このビットは1に設定されます。120MHzおよび60MHzの出力クロックが生成され使用可能な状態になります。 このビットが0の場合、PLLの出力クロックは用いられません。チップは、15MHzの内部発振器クロックを用いています。
[5] PLL_LOCK_ON	R 0x0	起動プロシージャの間PLLがロックされています。
[4] PLL_ERR	R 0x0	アナログPLLエラーが発生しました。
[3] CLK_STUCK	R 0x0	PLL出力クロックがスタックしています。ウォッチドッグは、内部発振器クロックを用いて、PLL出力クロックをモニタしています。
[2] CLK_FSM_ERR	R 0x0	クロック・コントローラFSMがエラーを検出しました。エラー・コードはCLK_FSM_ERR_RPTで読み出せます。このビットがセットされた場合、PLLはオフになり、クロック・コントローラは新しいCOMMITを待ちます。
[1] CLK_1M0_TIMEOUT	R 0x0	1MHzのクロックがPLLをイネーブルするのを待つ間にタイムアウト時間が経過しました。
[0] CLK_1M0_OK	R 0x0	PLL入力クロック用に1MHzが検出されています。

0x080：ADC.CONFIG

デフォルト：0x00000400

ADC 設定。このレジスタに書き込む場合、不使用ビットのデフォルト値は維持しておく必要があります。

ビットおよび名称	タイプおよびリセット	説明
[11:8] CSA_AZ_FLT_EXP	RW, unsigned 0x4	生のすべてのAZ（オートゼロ）値に対するデジタル・フィルタ（移動平均スタイル）のフィルタ長の指数。
[3] NRST_ADC_1	RW 0x0	ADC1のアクティブ・ローのソフトウェア・リセット。ADCのリセット後、I/Oコントローラをリセットします。
	0: RESET	ADC1がリセットされます。
	1: ENABLE	ADC1がイネーブルされます。
[2] NRST_ADC_0	RW 0x0	ADC0のアクティブ・ローのソフトウェア・リセット。ADCのリセット後、I/Oコントローラをリセットします。
	0: RESET	ADC0がリセットされます。
	1: ENABLE	ADC0がイネーブルされます。

0x081：ADC.ADC_VERSION

デフォルト：0x00000002

ADC のバージョン番号。

ビットおよび名称	タイプおよびリセット	説明
[31:0] VERSION_NUMBER	R, unsigned 0x00000002	ADCのバージョン番号。

0x082：ADC.I2_I1_RAW

デフォルト：0x00000000

相電流 I2 および I1 の生の値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] I2	RW, signed 0x0000	生の相電流I2
[15:0] I1	RW, signed 0x0000	生の相電流I1

0x083：ADC.VM_I3_RAW

デフォルト：0x00000000

電源電圧および相電流 I3 の生の値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] VM	RW, signed 0x0000	生のアナログ電圧モニタリング値
[15:0] I3	RW, signed 0x0000	生の相電流I3

0x084：ADC.TEMP_RAW

デフォルト：0x00000000

生の内部ダイ温度の値および TEMP アナログ入力ピンの生の電圧値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] TEMP_INT	RW, signed 0x0000	生の内部ダイ温度
[15:0] TEMP_EXT	RW, signed 0x0000	TEMPアナログ入力ピンの外部温度測定値の生の電圧値。

0x085：ADC.AIN_V_AIN_U_RAW

デフォルト：0x00000000

ポジション・フィードバック・アナログ・センサーの生の電圧値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] AIN_V	RW, signed 0x0000	HALL_Vピンの生のアナログ入力（代替機能使用）（オプションでHALL_VピンとENC_Bピンとの間の生のアナログ差動入力）。
[15:0] AIN_U	RW, signed 0x0000	HALL_Uピンの生のアナログ入力（代替機能使用）（オプションでHALL_UピンとENC_Aピンと間の生のアナログ差動入力）。

0x086：ADC.AIN_AIN_W_RAW

デフォルト：0x00000000

ポジション・フィードバック・アナログ・センサーの生の電圧値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] AIN	RW, signed 0x0000	生のアナログ入力AIN
[15:0] AIN_W	RW, signed 0x0000	HALL_Wピンの生のアナログ入力（代替機能使用）（オプションでHALL_WピンとENC_Nピンと間の生のアナログ差動入力）。

0x08A：ADC.STATUS

デフォルト：0x00000000

ADC のステータス・ビット

ビットおよび名称	タイプおよびリセット	説明
[31] I123_FAIL	R 0x0	生の相電流値すべての合計が、ADC.I123.I123_LIMITレジスタ・フィールドで設定された絶対制限値を超過しています。
[30] NO_ACK_ADC	RW, W1C 0x0	ADCが正しくアクノリッジされていません。
[27] USE_ADC_EXT_VALID_EXT	RW 0x0	内部生成されたADC_EXT_VALIDではなく、ADC_EXT_VALID_EXTがフィードバック・エンジンに用いられています。
[26] ADC_EXT_VALID_EXT	RW 0x0	ADC_EXT_VALID信号が外部生成されています。
[25] USE_ADC_I_VALID_EXT	RW 0x0	内部生成されたADC_I_VALIDではなく、ADC_I_VALID_EXTがFOCエンジンに用いられています。
[24] ADC_I_VALID_EXT	RW 0x0	ADC_I_VALID信号が外部生成されています。
[23] ADC_EXT_VALID	R 0x0	外部ADC測定が完了したことを示します。
[22] ADC_I_VALID	R 0x0	CSA測定が完了したことを示します。
[21] AIN_DONE	RW, W1C 0x0	AINチャンネルの測定が完了しています。
[20] AIN_W_DONE	RW, W1C 0x0	HALL_Wピンのアナログ測定が完了しています。
[19] AIN_V_DONE	RW, W1C 0x0	HALL_Vピンのアナログ測定が完了しています。
[18] AIN_U_DONE	RW, W1C 0x0	HALL_Uピンのアナログ測定が完了しています。
[17] I3_DONE	RW, W1C 0x0	I3のADC測定が完了しています。
[16] I2_DONE	RW, W1C 0x0	I2のADC測定が完了しています。
[15] I1_DONE	RW, W1C 0x0	I1のADC測定が完了しています。
[14] TEMP_EXT_DONE	RW, W1C 0x0	TEMPピンの外部温度入力の生のアナログ値測定が完了しています。
[13] TEMP_EXT_CLIPPED	RW, W1C 0x0	TEMPピンの外部温度入力の生のアナログ値がクリップされています。
[12] TEMP_INT_DONE	RW, W1C 0x0	生の内部ダイ温度測定が完了しています。
[11] TEMP_INT_CLIPPED	RW, W1C 0x0	生の内部ダイ温度値がクリップされています。
[10] VM_DONE	RW, W1C 0x0	生のアナログ電圧モニタリング測定が完了しています。

ビットおよび名称	タイプおよびリセット	説明
[9] VM_CLIPPED	RW, W1C 0x0	生のアナログ電圧モニタリング値がクリップされています。
[8] AIN_CLIPPED	RW, W1C 0x0	AINチャンネル値がクリップされています。
[7] AIN_W_CLIPPED	RW, W1C 0x0	HALL_Wピンのアナログ測定値がクリップされています。
[6] AIN_V_CLIPPED	RW, W1C 0x0	HALL_Vピンのアナログ測定値がクリップされています。
[5] AIN_U_CLIPPED	RW, W1C 0x0	HALL_Uピンのアナログ測定値がクリップされています。
[4] I3_CLIPPED	RW, W1C 0x0	I3のADC値がクリップされています。
[3] I2_CLIPPED	RW, W1C 0x0	I2のADC値がクリップされています。
[2] I1_CLIPPED	RW, W1C 0x0	I1のADC値がクリップされています。
[1] ADC_1_READY	R 0x0	ADC1レディ・ステータス・ビット
	0: ONGOING	ADC_1のキャリブレーションが進行中です。
	1: READY	ADC_1がキャリブレーションされ、通常動作が可能な状態になっています。
[0] ADC_0_READY	R 0x0	ADC0レディ・ステータス・ビット
	0: ONGOING	ADC_0のキャリブレーションが進行中です。
	1: READY	ADC_0がキャリブレーションされ、通常動作が可能な状態になっています。

0x08C：ADC.I123

デフォルト：0xFFFF0000

現在のすべての値の合計。

ビットおよび名称	タイプおよびリセット	説明
[31:16] I123_LIMIT	RW, unsigned 0xFFFF	ADC.STATUS.I123_FAILレジスタ・ビットをトリガする前の、生の相電流の絶対値すべての合計に対する制限値。
[15:0] I123	R, signed 0x0000	すべての生の相電流が加算されています。

0x0C0：MCC_ADC.I_GEN_CONFIG

デフォルト：0x00000000

電流 ADC の全般的な設定。このレジスタに書き込む場合、不使用ビットのデフォルト値は維持しておく必要があります。

ビットおよび名称	タイプおよびリセット	説明
[13:12] STATUS	R, unsigned 0x0	自動Kirchhoff計算のステータス：0 = 電流計算は置き換えられていません。この値が0より大きい場合、電流測定が自動Kirchhoff機能に置き換えられる特定の相番号が転送されます（1 = IU、2 = IV、3 = IW）。
[0] MEAS_MODE	RW 0x0	電流測定モード
	0: AUTOMATIC_SWITCH	自動Kirchhoff機能が有効化されています：PWMのデューティサイクルが最大の電流測定はスキップされ（AUTO_KIRCHHOFF_LIMに依存）、他の2つの相電流値を援用して計算される場合があります。
	1: NO_SWITCH	自動Kirchhoff機能が無効化されています：すべての電流測定値は、対応するFOC入力値に直接転送されます。

0x0C1：MCC_ADC.IW_IU

デフォルト：0x00000000

FOC モジュールに送信される、スケーリングとオフセットにより補償された U および W の相電流に対する ADC 測定値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] IW	R, signed 0x0000	スケーリングとオフセットにより補償されたW相電流に対するADC測定値。
[15:0] IU	R, signed 0x0000	スケーリングとオフセットにより補償されたU相電流に対するADC測定値。

0x0C2：MCC_ADC.IV

デフォルト：0x00000000

FOC モジュールに送信される、スケーリングとオフセットにより補償された V 相電流に対する ADC 測定値。

ビットおよび名称	タイプおよびリセット	説明
[15:0] IV	R, signed 0x0000	スケーリングとオフセットにより補償されたV相電流に対するADC測定値。

0x0C3：MCC_ADC.CSA_GAIN

デフォルト：0x00000000

CSA のゲイン設定。

ビットおよび名称	タイプおよびリセット	説明
[15:8] DYN_GAIN_HYST	RW, unsigned 0x00	動的ゲインのヒステリシスの選択
[6] DYN_GAIN_4X_EN	RW 0x0	4倍の動的ゲイン係数を有効にします。
[5] DYN_GAIN_3X_EN	RW 0x0	3倍の動的ゲイン係数を有効にします。
[4] DYN_GAIN_2X_EN	RW 0x0	2倍の動的ゲイン係数を有効にします。
[3:2] DYN_GAIN_ACT	R, unsigned 0x0	実際のゲイン係数
[1:0] CSA_GAIN	RW 0x0	CSAのゲイン値。この値は動的ゲインが有効でない場合にアクティブになります。
	0: X1 1: X2 2: X3 3: X4	CSAゲイン = 1 CSAゲイン = 2 CSAゲイン = 3 CSAゲイン = 4

0x0C4：MCC_ADC.EVENTS

デフォルト：0x00000000

一部の ADC 測定のステータス・ビットおよびイベント。

ビットおよび名称	タイプおよびリセット	説明
[6] CURRENT_OVERLOAD_STATUS	R 0x0	MCC_ADC.CURRENT_OVERLOADレジスタで設定されたソフト過負荷制限値との電流測定比較のステータス。
[5] TEMP_INT_LIMIT_EXCEEDED	R 0x0	MCC_ADC.TEMP_LIMITS.TEMP_INT_LIMITレジスタで設定されたソフト制限値との内部温度比較のステータス。
[4] TEMP_EXT_LIMIT_EXCEEDED	R 0x0	MCC_ADC.TEMP_LIMITS.TEMP_EXT_LIMITレジスタで設定されたソフト制限値との外部温度比較のステータス。

ビットおよび名称	タイプおよびリセット	説明
[1] MEAS_DONE	RW, W1C 0x0	電流測定完了フラグ（MCC_ADC処理中）。このビットをクリアするには、「1」を書き込みます。
[0] ADC_CLIPPED	RW, W1C 0x0	いずれかのADC測定がクリップされるとこのビットがセットされます。このビットをクリアするには、「1」を書き込みます。

0x0C5：MCC_ADC.DYN_GAIN_LIMITS_4X_3X

デフォルト：0x40008000

自動ゲイン・スイッチングの電流制限値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] DYN_GAIN_LIMIT_4X	RW, unsigned 0x4000	4xのゲインがその次に低倍のアクティブな動的ゲインに自動的に切り替わる前の電流制限値。
[15:0] DYN_GAIN_LIMIT_3X	RW, unsigned 0x8000	3xのゲインがその次に低倍のアクティブな動的ゲインに自動的に切り替わる前の電流制限値。

0x0C6：MCC_ADC.DYN_GAIN_LIMIT_2X

デフォルト：0x0000C000

自動ゲイン・スイッチングの電流制限値。

ビットおよび名称	タイプおよびリセット	説明
[15:0] DYN_GAIN_LIMIT_2X	RW, unsigned 0xC000	2xのゲインがその次に低倍のアクティブな動的ゲインに自動的に切り替わる前の電流制限値。

0x0C7：MCC_ADC.TEMP_LIMITS

デフォルト：0xFFFFFFFF

ソフト温度 ADC 測定制限値。値は、特定のステータス・ビットをトリガするためのスレッショルドとして設定できます。

ビットおよび名称	タイプおよびリセット	説明
[31:16] TEMP_INT_LIMIT	RW, unsigned 0xFFFF	内部ダイ温度ADC測定のソフト制限値。これを超えると、MCC_ADC.EVENTS.TEMP_INT_LIMIT_EXCEEDEDレジスタ・ビットがトリガされます。
[15:0] TEMP_EXT_LIMIT	RW, unsigned 0xFFFF	TEMPピンADC測定のソフト制限値。これを超えると、MCC_ADC.EVENTS.TEMP_EXT_LIMIT_EXCEEDEDレジスタのビットがトリガされます。

0x0C8：MCC_ADC.CURRENT_OVERLOAD

デフォルト：0x0000FFFF

ソフト電流 ADC 測定制限値。値は、特定のステータス・ビットをトリガするためのスレッショルドとして設定できます。

ビットおよび名称	タイプおよびリセット	説明
[15:0] CURRENT_OVERLOAD	RW, unsigned 0xFFFF	いずれかの相電流（IU、IV、IW）に対するADC測定のソフト制限値。この制限値を超えると、MCC_ADC.EVENTS.CURRENT_OVERLOAD_STATUSレジスタ・ビットがトリガされます。

0x100：MCC_CONFIG.MOTOR_MOTION

デフォルト：0x00000000

モーター・タイプおよびアクティブなモーション・モードの全般的な設定。

ビットおよび名称	タイプおよびリセット	説明
[19] VELOCITY_FF_EN	RW 0x0	RAMPER.RAMPER_VELOCITY値の速度PIDコントローラへのフィードフォワードをイネーブルします。MOTION_MODE = POSITIONの場合にのみイネーブルし、それ以外の場合はディスエーブルする必要があります。
[18] TORQUE_FF_VISC_FRIC_EN	RW 0x0	MCC_CONFIG.TORQUE_FEEDFORWARDレジスタの値を計算する際に、トルク・フィードフォワードの粘性摩擦成分（torque_ff_visc_fric）を考慮します。
[17] TORQUE_FF_COULOMB_FRICTION_EN	RW 0x0	MCC_CONFIG.TORQUE_FEEDFORWARDレジスタの値を計算する際に、トルク・フィードフォワードのクーロン摩擦成分（torque_ff_coulomb_fric）を考慮します。
[16] TORQUE_FF_ACC_EN	RW 0x0	MCC_CONFIG.TORQUE_FEEDFORWARDレジスタの値を計算する際に、トルク・フィードフォワードの加速度成分（torque_ff_acc）を考慮します。
[15] RAMP_USE_PHI_E	RW 0x0	このビットをイネーブルすると、FOCコントローラ用にRAMPER.PHI_E.PHI_Eレジスタ・フィールドの値をphi_e値として用い、FEEDBACK.OUTPUT_CONF.PHI_E_SRCレジスタ・フィールドでの選択をバイパスできます。
[14] RAMP_EN	RW 0x0	このビットがセットされると、コントローラはオープン・ループ・モードで動作します。ランプ・モジュールをイネーブルし、ランピング動作のプロファイルを生成します。
[13] RAMP_MODE	RW 0x0	ランプ・モジュールで速度動作プロファイルを生成するかポジショニング動作プロファイルを生成するかを定めます。該当のモーション・モードは、MOTION_MODEレジスタ・フィールドでも設定する必要があります。
	0: RAMP_POSITION	ポジション・モード。
	1: RAMP_VELOCITY	速度モード。
[12:9] MOTION_MODE	RW 0x0	モーション・コントローラ設定セットアップ。
	0: PWM_OFF	PWM出力をディスエーブル。
	1: PWM_ON	PWM出力をイネーブル。
	2: TORQUE	トルク・モード。ユーザはトルク目標値を設定します。トルクPIDのみを使用します。
	3: VELOCITY	速度モード。ユーザは速度目標を設定します。速度->トルクのカスケードを使用します。
	4: POSITION	ポジション・モード。ユーザはポジション目標を設定します。完全にカスケード化されたコントローラを使用します。
	5: PRBS_UD	UDとしてのPRBS。
	6: PRBS_FLUX	磁束コントローラ目標としてのPRBS。
	7: PRBS_TORQUE	トルク・コントローラ目標としてのPRBS。
	8: PRBS_VELOCITY	速度コントローラ目標としてのPRBS。
	9: PRBS_POSITION	ポジション・コントローラ目標としてのPRBS。
	10: PWM_EXT	外部PWMモード。PWMデューティサイクルを手動設定するために使用します。
	11: VOLTAGE_EXT	外部電圧モード。オープン・ループ制御に使用します。
[8:7] MOTOR_TYPE	RW 0x0	モーターのタイプ。使用可能なハーフ・ブリッジの使用法に影響します。
	0: NONE	モーターなし。
	1: DC	単相DCモーター。1つのコイルを駆動するのにUおよびVのハーフ・ブリッジ出力を使用します。
	2: RESERVED	
	3: BLDC	三相BLDCモーター。U、V、Wのハーフ・ブリッジ出力で三相電流を駆動します。
[6:0] N_POLE_PAIRS	RW, unsigned 0x00	BLDCモーターの極ペア数。この値は、phi_m = phi_e × N_POLE_PAIRSの式でモーターの機械的角度を計算するために用いられます。DCモーターを用いる場合、1の値に設定します。

0x101：MCC_CONFIG.GDRV

デフォルト：0x81003431

ゲート駆動の設定。

ビットおよび名称	タイプおよびリセット		説明
[31] CHARGE_PUMP_EN	RW 0x1		チャージ・ポンプをイネーブルまたはディスエーブルします。
[28] LP_MODE_EN	RW 0x0		低電力モード。
[24] USE_INTERNAL_R_REF	RW 0x1		内部リファレンス抵抗を使用。
[20] OVERTEMP_LATCH	RW 0x0		過熱フェイル・イベントがCHIP.EVENTS.OT_FAIL_EVENTレジスタ・ビットにラッチされます。このビットは、ドライバ段が再びイネーブルされる前にイベント・ビットをクリアする必要があるかどうかを制御します。
	0:	AUTO_RESTART	過熱ステータス・ビットがクリアされた後に自動で再始動します。
	1:	CLEAR	ドライバ段を再イネーブルする前に、過熱イベント・ビットをクリアする必要があります。
[16] DRV_EN_BIT	RW 0x0		ドライバ段をイネーブルまたはディスエーブルします。モーターのモーション動作には、ドライバ段がイネーブルされている（DRV_EN_BIT=1かつDRV_EN入力がハイ）必要があります。
[13:12] OCP_AUTORETRY	RW 0x3		過電流保護（OCP）再試行時間。設定されている場合、ゲート・ドライバは、無期限のOCPイベントが発生した後、一定の時間が経過した後に自動的に再始動します。
	0:	RETRY_AFTER_1MS	1ms後に再試行。
	1:	RETRY_AFTER_5MS	5ms後に再試行。
	2:	RETRY_AFTER_50MS	50ms後に再試行。
	3:	NO_RETRY	再試行は行われません。
[11:10] OCP_DEGLITCH	RW 0x1		過電流保護（OCP）デグリッチ時間。過電流ステータスの検出後、過電流状態がここで指定された時間存在する場合のみ、OCPイベントがトリガされます。
	0:	DEGLITCH_300_NS	300ns
	1:	DEGLITCH_600_NS	600ns
	2:	DEGLITCH_1200_NS	1200ns
	3:	DEGLITCH_2400_NS	2400ns
[9] OCP_DETECTION_MODE	RW 0x0		過電流保護（OCP）の検出を負荷電流をモニタして行うか、出力電圧をモニタして行うかの選択。
	0:	LOAD_CURRENT	OCPは負荷電流をチェック。
	1:	OUTPUT_VOLTAGE	OCPは出力電圧をチェック。
[5:4] LS_RES_ON	RW 0x3		ローサイド電力抵抗のR_{ON}の選択。
	0:	RES_200_MOHM	200mΩ
	1:	RES_112_MOHM	112mΩ
	2:	RES_78_MOHM	78mΩ
	3:	RES_55_MOHM	55mΩ
[1:0] SLEW_RATE	RW 0x1		出力のスルー・レートの制御。
	0:	SR_100_V_PER_US	100V/µs
	1:	SR_200_V_PER_US	200V/µs
	2:	SR_400_V_PER_US	400V/µs
	3:	SR_800_V_PER_US	800V/µs

0x102：MCC_CONFIG.PWM

デフォルト：0x00000017

PWM エンジンの全般的な設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] FLAT_BOTTOM_OFFSET	RW, unsigned 0x0000	オフセット・モードを用いてフラット・ボトム変調を行うためのPWMのデューティサイクルのオフセット。SV_MODE = BOTTOM_OFFSETの場合に使用します。
[5:4] SV_MODE	RW 0x1	空間ベクトルPWMモード。
	0: DISABLED	空間ベクトルPWMをディスエーブル。
	1: HARMONIC	3次高調波注入をイネーブル。
	2: BOTTOM	フラット・ボトム変調。
	3: BOTTOM_ OFFSET	オフセット付きのフラット・ボトム変調。
[2:0] CHOP	RW 0x7	CENTEREDモードをオン、OFFモードをオフにして使用します。その他のモードはテスト用です。
	0: OFF_FREE	PWMオフ。自走状態。
	1: OFF_LSON	PWMオフ。ローサイド（LS）が常時オン。
	2: OFF_HSON	PWMオフ。ハイサイド（HS）が常時オン。
	3: OFF_FREE2	PWMオフ。自走状態。
	4: OFF_FREE3	PWMオフ。自走状態。
	5: LSPWM_HSOFF	ローサイド（LS）PWM。ハイサイド（HS）オフ。
	6: HSPWM_LSOFF	ハイサイド（HS）PWM。ローサイド（LS）オフ。
	7: CENTERED	FOC用にPWMを中央に設定。

0x103：MCC_CONFIG.PWM_PERIOD

デフォルト：0xFFFF12C0

PWM 周波数と自動 Kirchhoff デューティサイクル制限値の設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] AUTO_KIRCHOFF_LIM	RW, unsigned 0xFFFF	いずれかの位相のPWMデューティサイクルがこの制限値を超えた場合、破損したその位相のADC測定の代わりに、Kirchhoffルールを用いて3番目の電流を計算します。
[15:0] MAX_COUNT	RW, unsigned 0x12C0	出力段を駆動するPWM信号を生成する、120MHzのクロック・サイクルの数。 PWM周波数とFOCコントローラのメインの更新レートが決まります。 MAX_COUNT = 120MHz/f_pwmで計算します。 デフォルト値の4800は、25kHzのPWM周波数に相当します。

0x104：MCC_CONFIG.BRAKE_CHOPPER_LIMITS

デフォルト：0xFFFF0000

ソフト過電圧制限値を超過すると直ちにアクティブになるようブレーキ・チョッパー出力を制御するために、内部測定電源電圧と比較されるスレッショルド値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] UPPER_SWITCH_LIM	RW, unsigned 0xFFFF	VM測定のブレーキ・チョッパー用上限スイッチング制限値。この制限値を超えると、OV_VM_LIMIT_FAIL_STATUSステータス・ビットまたはBRAKE出力（この出力がCHIP.IO_MATRIXによってイネーブルされている場合）がアクティブになります。
[15:0] LOWER_SWITCH_LIM	RW, unsigned 0x0000	VM測定のブレーキ・チョッパー用下限スイッチング制限値（低ヒステリシス値）。過電圧状態（OV_VM_LIMIT_FAIL_STATUSビットがアクティブ）の間、この制限値を下回ると、OV_VM_LIMIT_FAIL_STATUSビットまたはBRAKE出力（この出力がCHIP.IO_MATRIXによってイネーブルされている場合）が非アクティブになります。

0x105：MCC_CONFIG.MCC_STATUS

デフォルト：0x00000000

ドライバ段に関連するステータス・ビットおよびイベント・ビット。

ビットおよび名称	タイプおよびリセット	説明
[24] OVERTEMP_CH3_EVENT	RW, W1C 0x0	チャンネル3で過熱を検出
[23] OVERTEMP_CH2_EVENT	RW, W1C 0x0	チャンネル2で過熱を検出
[22] OVERTEMP_CH1_EVENT	RW, W1C 0x0	チャンネル1で過熱を検出
[21] SHRT2V_CH3_EVENT	RW, W1C 0x0	チャンネル3でV_Sへの短絡を検出
[20] SHRT2V_CH2_EVENT	RW, W1C 0x0	チャンネル2でV_Sへの短絡を検出
[19] SHRT2V_CH1_EVENT	RW, W1C 0x0	チャンネル1でV_Sへの短絡を検出
[18] SHRT2G_CH3_EVENT	RW, W1C 0x0	チャンネル3で地絡を検出
[17] SHRT2G_CH2_EVENT	RW, W1C 0x0	チャンネル2で地絡を検出
[16] SHRT2G_CH1_EVENT	RW, W1C 0x0	チャンネル1で地絡を検出
[8] OVERTEMP_CH3_STATUS	R 0x0	チャンネル3が過熱状態
[7] OVERTEMP_CH2_STATUS	R 0x0	チャンネル2が過熱状態
[6] OVERTEMP_CH1_STATUS	R 0x0	チャンネル1が過熱状態
[5] SHRT2V_CH3_STATUS	R 0x0	チャンネル3がV_Sへの短絡状態
[4] SHRT2V_CH2_STATUS	R 0x0	チャンネル2がV_Sへの短絡状態
[3] SHRT2V_CH1_STATUS	R 0x0	チャンネル1がV_Sへの短絡状態
[2] SHRT2G_CH3_STATUS	R 0x0	チャンネル3が地絡状態
[1] SHRT2G_CH2_STATUS	R 0x0	チャンネル2が地絡状態
[0] SHRT2G_CH1_STATUS	R 0x0	チャンネル1が地絡状態

0x106：MCC_CONFIG.TORQUE_FF_ACC_CONFIG

デフォルト：0x00060000

トルク・フィードフォワード加速度成分の設定。

ビットおよび名称	タイプおよびリセット	説明
[18:16] RAMP_ACC_SHIFT	RW 0x6	トルク・フィードフォワード加速度成分を計算するための、ランプ加速度に対するシフト係数。 torque_ff_acc = (RAMPER.ACCELERATION × RAMP_ACC_GAIN) >> (RAMP_ACC_SHIFT × 4)
	0: NO_ACC_SHIFT 1: ACC_SHIFT_BY_4 2: ACC_SHIFT_BY_8 3: ACC_SHIFT_BY_12 4: ACC_SHIFT_BY_16 5: ACC_SHIFT_BY_20 6: ACC_SHIFT_BY_24	
[15:0] RAMP_ACC_GAIN	RW, signed 0x0000	トルク・フィードフォワード加速度成分を計算するための、ランプ加速度をスケーリングするゲイン係数。 torque_ff_acc = (RAMPER.ACCELERATION × RAMP_ACC_GAIN) >> (RAMP_ACC_SHIFT × 4)

0x107：MCC_CONFIG.TORQUE_FF_VISC_FRIC_CONFIG

デフォルト：0x00070000

トルク・フィードフォワード粘性摩擦成分の設定。

ビットおよび名称	タイプおよびリセット	説明
[18:16] RAMP_VEL_SHIFT	RW 0x7	トルク・フィードフォワード粘性摩擦成分を計算するための、ランプ速度に対するシフト係数。 (RAMPER.V_ACTUAL × RAMP_VEL_GAIN) >> (RAMP_VEL_SHIFT × 4)
	0: NO_VEL_SHIFT 1: VEL_SHIFT_BY_4 2: VEL_SHIFT_BY_8 3: VEL_SHIFT_BY_12 4: VEL_SHIFT_BY_16 5: VEL_SHIFT_BY_20 6: VEL_SHIFT_BY_24 7: VEL_SHIFT_BY_28	
[15:0] RAMP_VEL_GAIN	RW, signed 0x0000	トルク・フィードフォワード粘性摩擦成分を計算するための、ランプ速度をスケーリングするゲイン係数。 torque_ff_visc_fric = (RAMPER.V_ACTUAL × RAMP_VEL_GAIN) >> (RAMP_VEL_SHIFT × 4)

0x108：MCC_CONFIG.TORQUE_FF_COULOMB_FRIC

デフォルト：0x00000000

トルク・フィードフォワード・クーロン摩擦成分の設定。

ビットおよび名称	タイプおよびリセット	説明
[15:0] TORQUE_FF_COULOMB_FRIC	RW, signed 0x0000	トルク・フィードフォワード・クーロン摩擦成分のフィードフォワード値。 RAMPER.V_ACTUAL > 0の場合、このフィールドの値が、 torque_ff_coulomb_fricに用いられ、 RAMPER.V_ACTUAL < 0の場合は、反転した値が用いられます。また、 その他の場合は、torque_ff_coulomb_fric = 0です。

0x109：MCC_CONFIG.TORQUE_FEEDFORWARD

デフォルト：0x00000000

トルク PID コントローラ用のフィードフォワード値。

ビットおよび名称	タイプおよびリセット	説明
[15:0] TORQUE_FEEDFORWARD	R, signed 0x0000	トルクPIDコントローラ用のフィードフォワード値。 MCC_CONFIG.MOTOR_MOTIONレジスタのイネーブルされたトルク・フィードフォワード成分に応じて、この値は次式によって求まります。 torque_ff_visc_fric + torque_ff_coulomb_fric + torque_ff_acc

0x140：FOC.PID_CONFIG

デフォルト：0x0000855C

PID コントローラの全般的な設定。

ビットおよび名称	タイプおよびリセット	説明
[22:16] POSITION_SAMPLING	RW, unsigned 0x00	position_controller_sampling_freq = pwm_freq/(VELOCITY_SAMPLING + 1)/(POSITION_SAMPLING + 1)の式に従って計算し、ポジション・コントローラのサンプリング周波数を設定します。 VELOCITY_SAMPLINGは、FEEDBACK.VELOCITY_FRQ_CONFの一部です。
[15:12] VELOCITY_SHIFT	RW, unsigned 0x8	速度コントローラ出力のシフト係数。MOTION_MODE = VELOCITYまたはMOTION_MODE = POSITIONの場合、シフトされた値が、トルク・コントローラの目標値になります。VELOCITY_SHIFTビット分だけ右にシフトします。
[11:10] POSITION_NORM_I	RW 0x1	ポジション・コントローラのKi係数に対する正規化係数。 POSITON_NORM_Iの数が増えるごとに、POSITION_Iフィールドの固定小数点表記を、デフォルトのQ8.8（POSITION_NORM_I = 0）から右に8ビット分だけ変化させます。それにより、実質的に、レジスタ・フィールドの係数Iの値が256^(POSITION_NORM_I+1)によって除算され、結果のフラクショナル部は保持されます。
	0: I_SHIFT_BY_8 1: I_SHIFT_BY_16 2: I_SHIFT_BY_24 3: I_SHIFT_BY_32	ポジション・シフト係数Ki：POSITION_Iを右に8ビット分シフトします。 ポジション・シフト係数Ki：POSITION_Iを右に16ビット分シフトします。 ポジション・シフト係数Ki：POSITION_Iを右に24ビット分シフトします。 ポジション・シフト係数Ki：POSITION_Iを右に32ビット分シフトします。
[9:8] POSITION_NORM_P	RW 0x1	ポジション・コントローラのKp係数に対する正規化係数。POSITON_NORM_Pの数が増えるごとに、POSITION_Pフィールドの固定小数点表記を、デフォルトのQ16.0（POSITION_NORM_P = 0）から右に8ビット分だけ変化させます。それにより、実質的に、レジスタ・フィールドの係数Pの値が256^POSITION_NORM_Pによって除算され、結果のフラクショナル部は保持されます。
	0: P_NO_SHIFT 1: P_SHIFT_BY_8 2: P_SHIFT_BY_16 3: P_SHIFT_BY_24	ポジション・シフト係数Kp：POSITION_Pのシフトはありません。 ポジション・シフト係数Kp：POSITION_Pを右に8ビット分シフトします。 ポジション・シフト係数Kp：POSITION_Pを右に16ビット分シフトします。 ポジション・シフト係数Kp：POSITION_Pを右に24ビット分シフトします。
[7:6] VELOCITY_NORM_I	RW 0x1	速度コントローラのKi係数に対する正規化係数。VELOCITY_NORM_Iの数が増えるごとに、VELOCITY_Iフィールドの固定小数点表記を、デフォルトのQ8.8（VELOCITY_NORM_I = 0）から右に8ビット分だけ変化させます。それにより、実質的に、レジスタ・フィールドの係数Iの値が256^(VELOCITY_NORM_I+1)によって除算され、結果のフラクショナル部は保持されます。
	0: I_SHIFT_BY_8 1: P_SHIFT_BY_16 2: P_SHIFT_BY_24 3: P_SHIFT_BY_32	速度シフト係数Ki：VELOCITY_Iを右に8ビット分シフトします。 速度シフト係数Ki：VELOCITY_Iを右に16ビット分シフトします。 速度シフト係数Ki：VELOCITY_Iを右に24ビット分シフトします。 速度シフト係数Ki：VELOCITY_Iを右に32ビット分シフトします。

ビットおよび名称	タイプおよびリセット		説明
[5:4] VELOCITY_NORM_P	RW 0x1		速度コントローラのKp係数に対する正規化係数。VELOCITY_NORM_Pの数が増えるごとに、VELOCITY_Pフィールドの固定小数点表記を、デフォルトのQ16.0（VELOCITY_NORM_P = 0）から右に8ビット分だけ変化させます。それにより、実質的に、レジスタ・フィールドの係数Iの値が256^VELOCITY_NORM_Pによって除算され、結果のフラクショナル部は保持されます。
	0:	P_NO_SHIFT	速度シフト係数Kp：VELOCITY_Pのシフトはありません。
	1:	P_SHIFT_BY_8	速度シフト係数Kp：VELOCITY_Pを右に8ビット分シフトします。
	2:	P_SHIFT_BY_16	速度シフト係数Kp：VELOCITY_Pを右に16ビット分シフトします。
	3:	P_SHIFT_BY_24	速度シフト係数Kp：VELOCITY_Pを右に24ビット分シフトします。
[3] CURRENT_NORM_I	RW 0x1		トルクおよび磁束コントローラのKi係数に対する正規化係数。TORQUE_Iフィールドおよび FLUX_Iフィールドの固定小数点表記を、フラクショナル・ビットの数を右に8ビット分だけシフトすることにより、Q16.0（CURRENT_NORM_I = 0）から（Q8.8に）変化させます。それにより、実質的に、レジスタ・フィールドの係数の値が256^CURRENT_NORM_Iによって除算され、結果のフラクショナル部は保持されます。
	0:	CUR_I_NO_SHIFT	電流シフト係数Ki：シフトはありません。
	1:	CUR_I_SHIFT_BY_8	電流シフト係数Ki：FLUX_IおよびTORQUE_Iを右に8ビット分シフトします。
[2] CURRENT_NORM_P	RW 0x1		トルクおよび磁束コントローラのKp係数に対する正規化係数。TORQUE_Pフィールドおよび FLUX_Pフィールドの固定小数点表記を、フラクショナル・ビットの数を右に8ビット分だけシフトすることにより、Q16.0（CURRENT_NORM_P = 0）から（Q8.8に）変化させます。それにより、実質的に、レジスタ・フィールドの係数の値が256^CURRENT_NORM_Pによって除算され、結果のフラクショナル部は保持されます。
	0:	CUR_P_NO_SHIFT	電流シフト係数Kp：シフトはありません。
	1:	CUR_P_SHIFT_BY_8	電流シフト係数Kp：FLUX_PおよびTORQUE_Pを右に8ビット分シフトします。
[1] OVERWRITE_TARGET	RW 0x0		イネーブルされていると、コントローラの目標値（pid_velocity_target、pid_torque_target）および積分器（pid_velocity_integrator）は、モーション・モードの円滑な遷移を実現できるよう、モジュールの出力によって直接上書きされます。
[0] KEEP_POS_TARGET	RW 0x0		セットされていると、PID_POSITION_ACTUALレジスタに手動で書込みを行った場合に、PID_POSITION_TARGETは自動的には上書きされません。

0x141：FOC.PID_U_S_MAX

デフォルト：0x00007FFF

最大許容バック EMF モーター電圧。これを超えると弱め界磁コントローラがアクティブになります。

ビットおよび名称	タイプおよびリセット	説明
[15:0] U_S_MAX	RW, unsigned 0x7FFF	最大許容バックEMFモーター電圧。これを超えると弱め界磁コントローラがアクティブになります。

0x142：FOC.PID_FLUX_COEFF

デフォルト：0x00000000

PI 磁束コントローラのゲインの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] FLUX_P	RW, signed 0x0000	PI磁束コントローラの比例ゲイン。
[15:0] FLUX_I	RW, signed 0x0000	PI磁束コントローラの積分ゲイン。

0x143：FOC.PID_TORQUE_COEFF

デフォルト：0x00000000

PI トルク・コントローラのゲインの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] TORQUE_P	RW, signed 0x0000	PIトルク・コントローラの比例ゲイン。
[15:0] TORQUE_I	RW, signed 0x0000	PIトルク・コントローラの積分ゲイン。

0x144：FOC.PID_FIELDWEAK_COEFF

デフォルト：0x00000000

PI 弱め界磁コントローラのゲインの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] FIELDWEAK_P	RW, signed 0x0000	PI弱め界磁コントローラの比例ゲイン。
[15:0] FIELDWEAK_I	RW, signed 0x0000	PI弱め界磁コントローラの積分ゲイン。

0x145：FOC.PID_VELOCITY_COEFF

デフォルト：0x00000000

PI 速度コントローラのゲインの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] VELOCITY_P	RW, signed 0x0000	PI速度コントローラの比例ゲイン。
[15:0] VELOCITY_I	RW, signed 0x0000	PI速度コントローラの積分ゲイン。

0x146：FOC.PID_POSITION_COEFF

デフォルト：0x00000000

PI ポジション・コントローラのゲインの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] POSITION_P	RW, signed 0x0000	PIポジション・コントローラの比例ゲイン。
[15:0] POSITION_I	RW, signed 0x0000	PIポジション・コントローラの積分ゲイン。

0x147：FOC.PID_POSITION_TOLERANCE

デフォルト：0x00000000

コントローラの許容ポジション誤差の設定。

ビットおよび名称	タイプおよびリセット	説明
[30:0] PID_POSITION_TOLERANCE	RW, unsigned 0x00000000	ポジション・コントローラは、この値より小さなポジション誤差を無視します。これを有効にするには、以下の2つの条件を満たす必要があります： position_reached_event割込み信号がセットされ、かつ、設定された PID_POSITION_TOLERANCE_DELAY PWMサイクル以上の間、誤差がこの値より小さくなっていなくてはなりません。

0x148：FOC.PID_POSITION_TOLERANCE_DELAY

デフォルト：0x00000000

コントローラの許容ポジション誤差の時間設定。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PID_POSITION_TOLERANCE_DELAY	RW, unsigned 0x0000	ポジション・コントローラが誤差を無視するために、絶対PID_POSITION_ERRORの値がPID_POSITION_TOLERANCEの設定値以内に収まっていなくてはならないPWM周期の数。

0x149：FOC.PID_UQ_UD_LIMITS

デフォルト：0x00005A81

電圧コントローラの目標値に対する制限値の設定。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PID_UQ_UD_LIMITS	RW, unsigned 0x5A81	電圧コントローラの絶対目標値に対する制限値。

0x14A：FOC.PID_TORQUE_FLUX_LIMITS

デフォルト：0x7FFF7FFF

トルクおよび磁束コントローラの絶対目標値に対する制限値の設定。

ビットおよび名称	タイプおよびリセット	説明
[30:16] PID_TORQUE_LIMIT	RW, unsigned 0x7FFF	トルク・コントローラの絶対目標値に対する制限値。
[14:0] PID_FLUX_LIMIT	RW, unsigned 0x7FFF	磁束コントローラの絶対目標値に対する制限値。

0x14B：FOC.PID_VELOCITY_LIMIT

デフォルト：0x7FFFFFFF

速度コントローラの絶対目標値に対する制限値の設定。

ビットおよび名称	タイプおよびリセット	説明
[30:0] PID_VELOCITY_LIMIT	RW, unsigned 0x7FFFFFFF	速度コントローラの絶対目標値に対する制限値。

0x14C：FOC.PID_POSITION_LIMIT_LOW

デフォルト：0x80000001

ポジション・コントローラの目標値に対する下限値の設定。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_POSITION_LIMIT_LOW	RW, signed 0x80000001	ポジション・コントローラの目標値に対する下限値。この下限値は正でも負でも構いませんが、PID_POSITION_LIMIT_HIGHより小さい値でなくてはなりません。

0x14D：FOC.PID_POSITION_LIMIT_HIGH

デフォルト：0x7FFFFFFF

ポジション・コントローラの目標値に対する上限値の設定。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_POSITION_LIMIT_HIGH	RW, signed 0x7FFFFFFF	ポジション・コントローラの目標値に対する上限値。この下限値は正でも負でも構いませんが、PID_POSITION_LIMIT_LOWより大きい値でなくてはなりません。

0x14E：FOC.PID_TORQUE_FLUX_TARGET

デフォルト：0x00000000

トルクおよび磁束コントローラに外部から書込み可能な PID 目標値（MOTION_MODE = TORQUE）。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PID_TORQUE_TARGET	RW, signed 0x0000	トルク・コントローラに外部から書込み可能なPID目標値 （MOTION_MODE = TORQUEの場合に使用）。
[15:0] PID_FLUX_TARGET	RW, signed 0x0000	磁束コントローラに外部から書込み可能なPID目標値 （MOTION_MODE = TORQUEの場合に使用）。

0x14F：FOC.PID_TORQUE_FLUX_OFFSET

デフォルト：0x00000000

PID のトルクおよび磁束オフセット。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PID_TORQUE_OFFSET	RW, signed 0x0000	フィードフォワード制御用のトルク・オフセット。
[15:0] PID_FLUX_OFFSET	RW, signed 0x0000	フィードフォワード制御用の磁束オフセット。

0x150：FOC.PID_VELOCITY_TARGET

デフォルト：0x00000000

速度コントローラに外部から書込み可能な PID 目標値（MOTION_MODE = VELOCITY）。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_VELOCITY_TARGET	RW, signed 0x00000000	速度コントローラに外部から書込み可能なPID目標値 （MOTION_MODE = VELOCITYの場合に使用）。

0x151：FOC.PID_VELOCITY_OFFSET

デフォルト：0x00000000

フィードフォワード制御用の PID 速度オフセット。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_VELOCITY_OFFSET	RW, signed 0x00000000	フィードフォワード制御用のPID速度オフセット。

0x152：FOC.PID_POSITION_TARGET

デフォルト：0x00000000

ポジション・コントローラに外部から書込み可能な PID 目標値（MOTION_MODE = POSITION）。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_POSITION_TARGET	RW, signed 0x00000000	ポジション・コントローラに外部から書込み可能なPID目標値 （MOTION_MODE = POSITIONの場合に使用）。

0x153：FOC.PID_TORQUE_FLUX_ACTUAL

デフォルト：0x00000000

トルクおよび磁束コントローラの PID の実際の値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PID_TORQUE_ACTUAL	R, signed 0x0000	トルク・コントローラのPIDの実際の値。
[15:0] PID_FLUX_ACTUAL	R, signed 0x0000	磁束コントローラのPIDの実際の値。

0x154：FOC.PID_VELOCITY_ACTUAL

デフォルト：0x00000000

速度コントローラの PID の実際の値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_VELOCITY_ACTUAL	R, signed 0x00000000	速度コントローラのPIDの実際の値。

0x155：FOC.PID_POSITION_ACTUAL

デフォルト：0x00000000

ポジション・コントローラの PID の実際の値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_POSITION_ACTUAL	RW, signed 0x00000000	ポジション・コントローラのPIDの実際の値。KEEP_POS_TARGETビットがセットされていない限り、このフィールドに手動で値の書込みを行うと、予期せぬ動作を招かぬよう、同じ値がPID_POSITION_TARGETにも書き込まれます。

0x156：FOC.PID_POSITION_ACTUAL_OFFSET

デフォルト：0x00000000

PID ポジションの実際のオフセット。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_POSITION_ACTUAL_OFFSET	RW, signed 0x00000000	PIDポジションの実際のオフセット。

0x157：FOC.PID_TORQUE_ERROR

デフォルト：0x00000000

トルク・コントローラによる PID 誤差値。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PID_TORQUE_ERROR	R, signed 0x0000	トルク・コントローラによるPID誤差値。

0x158：FOC.PID_FLUX_ERROR

デフォルト：0x00000000

磁束コントローラによる PID 誤差値。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PID_FLUX_ERROR	R, signed 0x0000	磁束コントローラによるPID誤差値。

0x159：FOC.PID_VELOCITY_ERROR

デフォルト：0x00000000

速度コントローラによる PID 誤差値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_VELOCITY_ERROR	R, signed 0x00000000	速度コントローラによるPID誤差値。

0x15A：FOC.PID_VELOCITY_ERROR_MAX

デフォルト：0x7FFFFFFF

許容可能な最大絶対 PID 速度誤差の値。

ビットおよび名称	タイプおよびリセット	説明
[30:0] PID_VELOCITY_ERROR_MAX	RW, unsigned 0x7FFFFFFF	許容可能な最大PID速度誤差の値。これを超えると RAMPER.STATUS.STALL_IN_V_ERRステータス・ビットがセットされます。

0x15B：FOC.PID_POSITION_ERROR

デフォルト：0x00000000

ポジション・コントローラによる PID 誤差値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_POSITION_ERROR	R, signed 0x00000000	ポジション・コントローラによるPID誤差値。

0x15C：FOC.PID_POSITION_ERROR_MAX

デフォルト：0x7FFFFFFF

許容可能な最大絶対 PID 速度誤差の値。

ビットおよび名称	タイプおよびリセット	説明
[30:0] PID_POSITION_ERROR_MAX	RW, unsigned 0x7FFFFFFF	許容可能な最大PID速度誤差の値。これを超えると RAMPER.STATUS.STALL_IN_POSITION_ERRステータス・ビットがセットされます。

0x15D：FOC.PID_TORQUE_INTEGRATOR

デフォルト：0x00000000

PID トルク・コントローラの積分成分。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_TORQUE_INTEGRATOR	RW, signed 0x00000000	PIDトルク・コントローラの積分成分。

0x15E：FOC.PID_FLUX_INTEGRATOR

デフォルト：0x00000000

PID 磁束コントローラの積分成分。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_FLUX_INTEGRATOR	RW, signed 0x00000000	PID磁束コントローラの積分成分。

0x15F：FOC.PID_VELOCITY_INTEGRATOR

デフォルト：0x00000000

PID 速度コントローラの積分成分。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_VELOCITY_INTEGRATOR	RW, signed 0x00000000	PID速度コントローラの積分成分。

0x160：FOC.PID_POSITION_INTEGRATOR

デフォルト：0x00000000

PID ポジション・コントローラの積分成分。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PID_POSITION_INTEGRATOR	RW, signed 0x00000000	PIDポジション・コントローラの積分成分。

0x161：FOC.PIDIN_TORQUE_FLUX_TARGET

デフォルト：0x00000000

いずれかのフィルタ、オフセット、フィードフォワード補償または制限値が適用される前の、PID トルクおよび磁束目標値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PIDIN_TORQUE_TARGET	R, signed 0x0000	いずれかのフィルタ、オフセット、フィードフォワード補償または制限値が適用される前の、PIDトルク目標値。
[15:0] PIDIN_FLUX_TARGET	R, signed 0x0000	いずれかのフィルタ、オフセット、フィードフォワード補償または制限値が適用される前の、PID磁束目標値。

0x162：FOC.PIDIN_VELOCITY_TARGET

デフォルト：0x00000000

いずれかのオフセット、フィードフォワード補償または制限値が適用される前の、PID 速度目標値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PIDIN_VELOCITY_TARGET	R, signed 0x00000000	いずれかのオフセット、フィードフォワード補償または制限値が適用される前の、PID速度目標値。

0x163：FOC.PIDIN_POSITION_TARGET

デフォルト：0x00000000

いずれかの制限値が適用される前の PID ポジション目標値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PIDIN_POSITION_TARGET	R, signed 0x00000000	いずれかの制限値が適用される前のPIDポジション目標値。

0x164：FOC.PIDIN_TORQUE_FLUX_TARGET_LIMITED

デフォルト：0x00000000

設定されたフィルタ、オフセット、フィードフォワード補償または制限値が適用された後の、トルクおよび磁束コントローラに対する PID 目標値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PIDIN_TORQUE_TARGET_LIMITED	R, signed 0x0000	設定されたフィルタ、オフセット、フィードフォワード補償または制限値が適用された後の、トルク・コントローラに対するPID目標値。
[15:0] PIDIN_FLUX_TARGET_LIMITED	R, signed 0x0000	設定されたフィルタ、オフセット、フィードフォワード補償または制限値が適用された後の、磁束コントローラに対するPID目標値。

0x165：FOC.PIDIN_VELOCITY_TARGET_LIMITED

デフォルト：0x00000000

設定されたオフセット、フィードフォワード補償または制限値が適用された後の、速度コントローラに対する PID 目標値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PIDIN_VELOCITY_TARGET_LIMITED	R, signed 0x00000000	設定されたオフセット、フィードフォワード補償または制限値が適用された後の、速度コントローラに対するPID目標値。

0x166：FOC.PIDIN_POSITION_TARGET_LIMITED

デフォルト：0x00000000

設定された制限値が適用された後の、ポジション・コントローラに対する PID 目標値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PIDIN_POSITION_TARGET_ LIMITED	R, signed 0x00000000	設定された制限値が適用された後の、ポジション・コントローラに対するPID目標値。

0x167：FOC.FOC_IBETA_IALPHA

デフォルト：0x00000000

FOC モジュールの Ibeta および Ialpha の項に対する暫定的な結果。

ビットおよび名称	タイプおよびリセット	説明
[31:16] IBETA	R, signed 0x0000	FOCモジュールのIbeta項に対する暫定的な結果。
[15:0] IALPHA	R, signed 0x0000	FOCモジュールのIalpha項に対する暫定的な結果。

0x168：FOC.FOC_IQ_ID

デフォルト：0x00000000

FOC モジュールの Iq および Id の項に対する暫定的な結果。

ビットおよび名称	タイプおよびリセット	説明
[31:16] IQ	R, signed 0x0000	FOCモジュールのIq項に対する暫定的な結果。
[15:0] ID	R, signed 0x0000	FOCモジュールのId項に対する暫定的な結果。

0x169：FOC.FOC_UQ_UD

デフォルト：0x00000000

いずれかの制限値が適用される前の、FOC モジュールの Uq および Ud の項に対する暫定的な結果。

ビットおよび名称	タイプおよびリセット	説明
[31:16] UQ	R, signed 0x0000	いずれかの制限値が適用される前の、FOCモジュールのUq項に対する暫定的な結果。
[15:0] UD	R, signed 0x0000	いずれかの制限値が適用される前の、FOCモジュールのUd項に対する暫定的な結果。

0x16A：FOC.FOC_UQ_UD_LIMITED

デフォルト：0x00000000

設定された制限値が適用された後の、FOC モジュールの Uq および Ud の項に対する暫定的な結果。

ビットおよび名称	タイプおよびリセット	説明
[31:16] UQ_LIMITED	R, signed 0x0000	設定された制限値が適用された後の、FOCモジュールのUq項に対する暫定的な結果。
[15:0] UD_LIMITED	R, signed 0x0000	設定された制限値が適用された後の、FOCモジュールのUd項に対する暫定的な結果。

0x16B：FOC.FOC_UBETA_UALPHA

デフォルト：0x00000000

FOC モジュールの Ubeta および Ualpha の項に対する暫定的な結果。

ビットおよび名称	タイプおよびリセット	説明
[31:16] UBETA	R, signed 0x0000	FOCモジュールのUbetaに対する暫定的な結果。
[15:0] UALPHA	R, signed 0x0000	FOCモジュールのUalphaに対する暫定的な結果。

0x16C：FOC.FOC_UW_UU

デフォルト：0x00000000

FOC モジュールの UW および UU の項に対する暫定的な結果。

ビットおよび名称	タイプおよびリセット	説明
[31:16] UW	R, signed 0x0000	FOCモジュールのUW項に対する暫定的な結果。
[15:0] UU	R, signed 0x0000	FOCモジュールのUU項に対する暫定的な結果。

0x16D：FOC.FOC_UV

デフォルト：0x00000000

FOC モジュールの UV 項に対する暫定的な結果。

ビットおよび名称	タイプおよびリセット	説明
[15:0] UV	R, signed 0x0000	FOCモジュールのUV項に対する暫定的な結果。

0x16E：FOC.PWM_V_U

デフォルト：0x00000000

V 相および U 相に対する PWM デューティサイクル。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PWM_V	R, unsigned 0x0000	V相に対するPWMデューティサイクル。
[15:0] PWM_U	R, unsigned 0x0000	U相に対するPWMデューティサイクル。

0x16F：FOC.PWM_W

デフォルト：0x00000000

W 相に対する PWM デューティサイクル。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PWM_W	R, unsigned 0x0000	W相に対するPWMデューティサイクル。

0x170：FOC.FOC_STATUS

デフォルト：0x00000000

FOC モジュールの現在の状態。

ビットおよび名称	タイプおよびリセット	説明
[3:0] FOC_STATUS	R, unsigned 0x0	FOCモジュールの現在の状態。

0x171：FOC.U_S_ACTUAL_I_S_ACTUAL

デフォルト：0x00000000

モーターの実際の電圧および電流値。

ビットおよび名称	タイプおよびリセット	説明
[31:16] U_S_ACTUAL	R, unsigned 0x0000	実際のモーター電圧値。U_S_ACTUAL = sqrt(u_d^2 + u_q^2)の式で計算。
[15:0] I_S_ACTUAL	R, unsigned 0x0000	実際のモーター電流値。I_S_ACTUAL = sqrt(i_d^2 + i_q^2)の式で計算。

0x172：FOC.P_MOTOR

デフォルト：0x00000000

モーターに印加された実際の電力。

ビットおよび名称	タイプおよびリセット	説明
[31:0] P_MOTOR	R, unsigned 0x00000000	モーターに印加された実際の電力。 P_MOTOR = U_S_ACTUAL × I_S_ACTUALの式で計算。

0x173：FOC.I_T_ACTUAL

デフォルト：0x00000000

実際の目標電流値。

ビットおよび名称	タイプおよびリセット	説明
[15:0] I_T_ACTUAL	R, unsigned 0x0000	実際の目標電流値。 I_T_ACTUAL = sqrt(i_d_target^2 + i_q_target^2)の式で計算。

0x174：FOC.PRBS_AMPLITUDE

デフォルト：0x00000000

PRBS 信号の振幅。システムの識別に使用。

ビットおよび名称	タイプおよびリセット	説明
[31:0] PRBS_AMPLITUDE	RW, unsigned 0x00000000	PRBS信号の振幅。

0x175：FOC.PRBS_DOWN_SAMPLING_RATIO

デフォルト：0x00000000

PRBS 信号のダウン・サンプリング・レート。システムの識別に使用。

ビットおよび名称	タイプおよびリセット	説明
[7:0] PRBS_DOWN_SAMPLING_RATIO	RW, unsigned 0x00	PRBS信号のダウン・サンプリング・レート。 prbs_freq = pwm_freq/(PRBS_DOWN_SAMPLING_RATIO + 1)

0x180：BIQUAD.BIQUAD_EN

デフォルト：0x00000001

速度およびトルクのバイクワッド・フィルタ用信号をイネーブルします。速度フィルタは速度測定に対して用いられ、トルク・フィルタはトルク・コントローラの目標値に対して用いられる点に注意してください。

ビットおよび名称	タイプおよびリセット	説明
[1] TORQUE_EN	RW 0x0	トルク・コントローラの目標値に対するバイクワッド・フィルタをイネーブルします。
[0] VELOCITY_EN	RW 0x1	速度の実際の測定値に対するバイクワッド・フィルタをイネーブルします。

0x181：BIQUAD.VELOCITY_A1

デフォルト：0x001C376B

速度バイクワッド・フィルタ係数 a1。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_VELOCITY_A1	RW, signed 0x1C376B	速度バイクワッド・フィルタ係数a1

0x182：BIQUAD.VELOCITY_A2

デフォルト：0x00F38F52

速度バイクワッド・フィルタ係数 a2。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_VELOCITY_A2	RW, signed 0xF38F52	速度バイクワッド・フィルタ係数a2

0x183：BIQUAD.VELOCITY_B0

デフォルト：0x00000E51

速度バイクワッド・フィルタ係数 b0。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_VELOCITY_B0	RW, signed 0x000E51	速度バイクワッド・フィルタ係数b0

0x184：BIQUAD.VELOCITY_B1

デフォルト：0x00001CA1

速度バイクワッド・フィルタ係数 b1。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_VELOCITY_B1	RW, signed 0x001CA1	速度バイクワッド・フィルタ係数b1

0x185：BIQUAD.VELOCITY_B2

デフォルト：0x00000E51

速度バイクワッド・フィルタ係数 b2。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_VELOCITY_B2	RW, signed 0x000E51	速度バイクワッド・フィルタ係数b2

0x186：BIQUAD.TORQUE_A1

デフォルト：0x00000000

トルク・バイクワッド・フィルタ係数 a1。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_TORQUE_A1	RW, signed 0x000000	トルク・バイクワッド・フィルタ係数a1

0x187：BIQUAD.TORQUE_A2

デフォルト：0x00000000

トルク・バイクワッド・フィルタ係数 a2。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_TORQUE_A2	RW, signed 0x000000	トルク・バイクワッド・フィルタ係数a2

0x188：BIQUAD.TORQUE_B0

デフォルト：0x00100000

トルク・バイクワッド・フィルタ係数 b0。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_TORQUE_B0	RW, signed 0x100000	トルク・バイクワッド・フィルタ係数b0

0x189：BIQUAD.TORQUE_B1

デフォルト：0x00000000

トルク・バイクワッド・フィルタ係数 b1。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_TORQUE_B1	RW, signed 0x000000	トルク・バイクワッド・フィルタ係数b1

0x18A：BIQUAD.TORQUE_B2

デフォルト：0x00000000

トルク・バイクワッド・フィルタ係数 b2。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COEFF_TORQUE_B2	RW, signed 0x000000	トルク・バイクワッド・フィルタ係数b2

0x1C0：RAMPER.TIME_CONFIG

デフォルト：0x00000000

ランプ・タイミング・パラメータの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] T_VMAX	RW, unsigned 0x0000	減速を開始する前のVMAXにおける最小時間。
[15:0] T_ZEROWAIT	RW, unsigned 0x0000	ランプ・ポジショニング動作を終了してから新たに設定された動作を開始するまでの最小時間。

0x1C1：RAMPER.SWITCH_MODE

デフォルト：0x00000000

ランプ・リファレンス・スイッチおよび停止信号の設定。

ビットおよび名称	タイプおよびリセット	説明
[18] STOP_ON_V_DEVIATION	RW 0x0	STALL_IN_V_ERR信号がランプ・コントローラにストール停止イベントをトリガできるようにします。停止イベントを生成するには、STALL_STOP_ENをセットする必要もあります。
[17] STOP_ON_POS_DEVIATION	RW 0x0	STALL_IN_POSITION_ERR信号がランプ・コントローラにストール停止イベントをトリガできるようにします。停止イベントを生成するには、STALL_STOP_ENをセットする必要もあります。
[16] FORCE_HARD_STOP	RW 0x0	手動でランプ・コントローラにストール停止イベントをトリガします。停止イベントを生成するには、STALL_STOP_ENをセットする必要もあります。
[15] SOFT_STOP_EN	RW 0x0	リファレンス・スイッチの停止イベントに、ハード・ストップではなくソフト・ストップ・シーケンスを使用します。リファレンス・スイッチが停止イベントをトリガするには、RAMP_REF_H_STOP_EN、RAMP_REF_R_STOP_EN、RAMP_REF_L_STOP_EN、STALL_STOP_ENの対応するビットをセットする必要があります。ソフト・ストップでは、DMAX、D2、D1、V2、V1、VSTOP、T_ZEROWAITの減速ランプ設定を用いてモーターを停止させます。ハード・ストップも、モーターが解放される前にT_ZEROWAITを用います。
[14] STALL_STOP_EN	RW 0x0	ランプ・コントローラを通じてモーターを停止するには、STOP_ON_V_DEVIATION、STOP_ON_POS_DEVIATION、FORCE_HARD_STOPからストール停止イベントをイネーブルします。ディスエーブルの場合、EVENTS.STALL_STOP_EVENTビットが自動的にクリアされます。
[12] LATCH_REF_H_INACTIVE	RW 0x0	ホーム・リファレンス・スイッチ入力において非有効化へのエッジが発生したときに、ポジションをPOSITION_LATCHおよびPOSITION_ACTUAL_LATCHにラッチします。アクティブ・レベルの極性はRAMP_REF_H_POLで定められます。ラッチされたポジションを取得できるかどうかをモニタするには、LATCH_REF_H_READYイベント・フラグを用います。
[11]L ATCH_REF_H_ACTIVE	RW 0x0	ホーム・リファレンス・スイッチ入力において有効化へのエッジが発生したときに、ポジションをPOSITION_LATCHおよびPOSITION_ACTUAL_LATCHにラッチします。アクティブ・レベルの極性はRAMP_REF_H_POLで定められます。ラッチされたポジションを取得できるかどうかをモニタするには、LATCH_REF_H_READYイベント・フラグを用います。
[10] LATCH_REF_R_INACTIVE	RW 0x0	右リファレンス・スイッチ入力において非有効化へのエッジが発生したときに、ポジションをPOSITION_LATCHおよびPOSITION_ACTUAL_LATCHにラッチします。アクティブ・レベルの極性はRAMP_REF_R_POLで定められます。ラッチされたポジションを取得できるかどうかをモニタするには、LATCH_REF_R_READYイベント・フラグを用います。
[9] LATCH_REF_R_ACTIVE	RW 0x0	右リファレンス・スイッチ入力において有効化へのエッジが発生したときに、ポジションをPOSITION_LATCHおよびPOSITION_ACTUAL_LATCHにラッチします。アクティブ・レベルの極性はRAMP_REF_R_POLで定められます。ラッチされたポジションを取得できるかどうかをモニタするには、LATCH_REF_R_READYイベント・フラグを用います。

ビットおよび名称	タイプおよびリセット	説明
[8] LATCH_REF_L_INACTIVE	RW 0x0	左リファレンス・スイッチ入力において非有効化へのエッジが発生したときに、ポジションをPOSITION_LATCHおよびPOSITION_ACTUAL_LATCHにラッチします。アクティブ・レベルの極性はRAMP_REF_L_POLで定められます。ラッチされたポジションを取得できるかどうかをモニタするには、LATCH_REF_L_READYイベント・フラグを用います。
[7] LATCH_REF_L_ACTIVE	RW 0x0	左リファレンス・スイッチ入力において有効化へのエッジが発生したときに、ポジションをPOSITION_LATCHおよびPOSITION_ACTUAL_LATCHにラッチします。アクティブ・レベルの極性はRAMP_REF_L_POLで定められます。ラッチされたポジションを取得できるかどうかをモニタするには、LATCH_REF_L_READYイベント・フラグを用います。
[6] SWAP_RAMP_REF_LR	RW 0x0	左リファレンス・スイッチ入力REF_Lと右リファレンス・スイッチ入力REF_Rを入れ替えます
[5] RAMP_REF_H_POL	RW 0x0	ホーム・リファレンス・スイッチ入力のアクティブな極性を設定します。
	0: NON_INVERTED 1: INVERTED	非反転、アクティブ・ハイ。 反転、アクティブ・ロー。
[4] RAMP_REF_R_POL	RW 0x0	右リファレンス・スイッチ入力のアクティブな極性を設定します。
	0: NON_INVERTED 1: INVERTED	非反転、アクティブ・ハイ。 反転、アクティブ・ロー。
[3] RAMP_REF_L_POL	RW 0x0	左リファレンス・スイッチ入力のアクティブな極性を設定します。
	0: NON_INVERTED 1: INVERTED	非反転、アクティブ・ハイ。 反転、アクティブ・ロー。
[2] RAMP_REF_H_STOP_EN	RW 0x0	ホーム・リファレンス・スイッチ入力がアクティブとなっている間にランプ・コントローラを通じて自動モーター停止をイネーブルします。停止信号のステータスは、RAMP_REF_H_STOP_STATUSビットから取得できます。
[1] RAMP_REF_R_STOP_EN	RW 0x0	右リファレンス・スイッチ入力がアクティブとなっている間にランプ・コントローラを通じて自動モーター停止をイネーブルします。スイッチ入力がアクティブのときに速度の符号が正であると、停止信号がトリガされます。停止信号のステータスは、RAMP_REF_R_STOP_STATUSビットから取得できます。
[0] RAMP_REF_L_STOP_EN	RW 0x0	左リファレンス・スイッチ入力がアクティブとなっている間にランプ・コントローラを通じて自動モーター停止をイネーブルします。スイッチ入力がアクティブのときに速度の符号が負であると、停止信号がトリガされます。停止信号のステータスは、RAMP_REF_L_STOP_STATUSビットから取得できます。

0x1C3：RAMPER.PHI_E

デフォルト：0x00000000

ランプ・ポジションに基づくランプ phi_e。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PHI_E_OFFSET	RW, unsigned 0x0000	ランプphi_e計算のオフセット。
[15:0] PHI_E	R, unsigned 0x0000	オープン・ループphi_e。POSITION.POSITION + PHI_E_OFFSETから計算。MCC_CONFIG.MOTOR_MOTION.RAMP_USE_PHI_Eがセットされている場合に使用。

0x1C4：RAMPER.A1

デフォルト：0x00010000

V_START から V1 までの 8 点ランプ加速。

ビットおよび名称	タイプおよびリセット	説明
[22:0] A1	RW, unsigned 0x010000	V_STARTからV1までの8点ランプ加速。

0x1C5：RAMPER.A2

デフォルト：0x00010000

V1 から V2 までの 8 点ランプ加速。

ビットおよび名称	タイプおよびリセット	説明
[22:0] A2	RW, unsigned 0x010000	V1からV2までの8点ランプ加速。

0x1C6：RAMPER.A_MAX

デフォルト：0x00010000

V2 から V_MAX までの 8 点ランプ加速。

ビットおよび名称	タイプおよびリセット	説明
[22:0] A_MAX	RW, unsigned 0x010000	V2からV_MAXまでの8点ランプ加速。

0x1C7：RAMPER.D1

デフォルト：0x00010000

V1 から V_STOP までの 8 点ランプ減速。

ビットおよび名称	タイプおよびリセット	説明
[22:0] D1	RW, unsigned 0x010000	V1からV_STOPまでの8点ランプ減速。

0x1C8：RAMPER.D2

デフォルト：0x00010000

V2 から V1 までの 8 点ランプ減速。

ビットおよび名称	タイプおよびリセット	説明
[22:0] D2	RW, unsigned 0x010000	V2からV1までの8点ランプ減速。

0x1C9：RAMPER.D_MAX

デフォルト：0x00010000

V_MAX から V2 までの 8 点ランプ減速。

ビットおよび名称	タイプおよびリセット	説明
[22:0] D_MAX	RW, unsigned 0x010000	V_MAXからV2までの8点ランプ減速。

0x1CA：RAMPER.V_START

デフォルト：0x00000000

8 点ランプの開始速度。

ビットおよび名称	タイプおよびリセット	説明
[22:0] V_START	RW, unsigned 0x000000	8点ランプの開始速度。

0x1CB：RAMPER.V1

デフォルト：0x00000000

8 点ランプの最初の切り替え速度。

ビットおよび名称	タイプおよびリセット	説明
[26:0] V1	RW, unsigned 0x0000000	8点ランプの最初の速度。

0x1CC：RAMPER.V2

デフォルト：0x00000000

8 点ランプの 2 つ目の切り替え速度。

ビットおよび名称	タイプおよびリセット	説明
[26:0] V2	RW, unsigned 0x0000000	8点ランプの2つ目の速度。

0x1CD：RAMPER.V_STOP

デフォルト：0x00000100

8 点ランプの停止速度。

ビットおよび名称	タイプおよびリセット	説明
[22:0] V_STOP	RW, unsigned 0x000100	8点ランプの停止速度。

0x1CE：RAMPER.V_MAX

デフォルト：0x07FFFFFF

8 点ランプの最大（3 つ目）の速度。

ビットおよび名称	タイプおよびリセット	説明
[26:0] V_MAX	RW, unsigned 0x7FFFFFF	8点ランプの最大（3つ目）の速度。

0x1CF：RAMPER.ACCELERATION

デフォルト：0x00000000

ランプ・コントローラの現在の加速度出力値。

ビットおよび名称	タイプおよびリセット	説明
[23:0] ACCELERATION	R, signed 0x000000	ランプ・コントローラの現在の加速度出力値。

0x1D0：RAMPER.V_ACTUAL

デフォルト：0x00000000

ランプ・コントローラの現在の速度出力値。

ビットおよび名称	タイプおよびリセット	説明
[27:0] V_ACTUAL	R, signed 0x0000000	ランプ・コントローラの現在の速度出力値。

0x1D1：RAMPER.POSITION

デフォルト：0x00000000

ランプ・コントローラの現在のマルチターン・ポジション・出力値。

ビットおよび名称	タイプおよびリセット	説明
[31:0] POSITION	RW, signed 0x00000000	ランプ・コントローラの現在のマルチターン・ポジション・出力値。

0x1D2：RAMPER.POSITION_LATCH

デフォルト：0x00000000

リファレンス・スイッチ・トリガでラッチされた RAMPER.POSITION。トリガは RAMPER.SWITCH_MODE で設定する必要があります。

ビットおよび名称	タイプおよびリセット	説明
[31:0] POSITION_LATCH	R, signed 0x00000000	リファレンス・スイッチ・トリガでラッチされたRAMPER.POSITION。トリガはRAMPER.SWITCH_MODEで設定する必要があります。

0x1D3：RAMPER.POSITION_ACTUAL_LATCH

デフォルト：0x00000000

リファレンス・スイッチ・トリガでラッチされた FOC.PID_POSITION_ACTUAL。トリガは RAMPER.SWITCH_MODE で設定する必要があります。

ビットおよび名称	タイプおよびリセット	説明
[31:0] POSITION_ACTUAL_LATCH	R, signed 0x00000000	リファレンス・スイッチ・トリガでラッチされたFOC.PID_POSITION_ACTUAL。トリガはRAMPER.SWITCH_MODEで設定する必要があります。

0x1D4：RAMPER.STATUS

デフォルト：0x00000000

ランプ・コントローラ用のステータス・ビット。

ビットおよび名称	タイプおよびリセット	説明
[17] STALL_IN_POSITION_ERR	R 0x0	現在の絶対ポジション偏差がFOC.PID_POSITION_ERROR_MAXを超えると、このビットがセットされます。
[16] STALL_IN_V_ERR	R 0x0	現在の絶対速度偏差がFOC.PID_VELOCITY_ERROR_MAXを超えると、このビットがセットされます。
[14] T_ZEROWAIT_ACTIVE	R 0x0	ランプのT_ZEROWAIT部がアクティブであり、モーターが停止状態であることを示している場合、このビットがセットされます。
[13] V_ZERO	R 0x0	V_ACTUALの値がゼロの場合、このビットがセットされます。
[12] POSITION_REACHED_STATUS	R 0x0	POSITION.POSITIONとFOC.PID_POSITION_TARGETが一致すると、このビットがセットされます。
[11] V_REACHED_STATUS	R 0x0	RAMP_MODE = RAMP_POSITIONでありV_ACTUALがV_MAXに一致している間、または、RAMP_MODE = RAMP_VELOCITYでありV_ACTUALがFOC.PID_VELOCITY_TARGETに一致している間、アクティブになります。RAMP_MODEはMCC_CONFIG.MOTOR_MOTIONレジスタの一部である点に注意してください。

ビットおよび名称	タイプおよびリセット	説明
[8] RAMP_REF_H_STOP_STATUS	R 0x0	ホーム・リファレンス・スイッチの停止条件がアクティブであることを示します。ソフト・ストップがイネーブルの場合、この条件は、ソフト・ストップ・シーケンスが終了した後ではなく、スイッチが検出されると直ちにアクティブになります。
[7] RAMP_REF_R_STOP_STATUS	R 0x0	右リファレンス・スイッチの停止条件がアクティブであることを示します。ソフト・ストップがイネーブルの場合、この条件は、ソフト・ストップ・シーケンスが終了した後ではなく、スイッチが検出されると直ちにアクティブになります。割込み条件および停止条件は、反対方向への移動をコマンドすることによりクリアできます。
[6] RAMP_REF_L_STOP_STATUS	R 0x0	左リファレンス・スイッチの停止条件がアクティブであることを示します。ソフト・ストップがイネーブルの場合、この条件は、ソフト・ストップ・シーケンスが終了した後ではなく、スイッチが検出されると直ちにアクティブになります。割込み条件および停止条件は、反対方向への移動をコマンドすることによりクリアできます。
[2] RAMP_REF_H_STATUS	R 0x0	ランプ・コントローラで設定されたホーム・リファレンス・スイッチのステータス。
[1] RAMP_REF_R_STATUS	R 0x0	ランプ・コントローラで設定された右リファレンスのステータス。
[0] RAMP_REF_L_STATUS	R 0x0	ランプ・コントローラで設定された左リファレンスのステータス。

0x1D5：RAMPER.EVENTS

デフォルト：0x00000000

ランプ・コントローラ用のイベント・フラグ。

ビットおよび名称	タイプおよびリセット	説明
[5] LATCH_REF_R_READY	RW, W1C 0x0	右リファレンス・スイッチ・イベント後にポジション値がラッチされていることを示します。ポジションのラッチを行うには、必要なエッジに対し、対応するSWITCH_MODEレジスタのビットでイネーブルする必要があります。このビットをクリアするには、「1」を書き込みます。
[4] LATCH_REF_L_READY	RW, W1C 0x0	左リファレンス・スイッチ・イベント後にポジション値がラッチされていることを示します。ポジションのラッチを行うには、必要なエッジに対し、対応するSWITCH_MODEレジスタのビットでイネーブルする必要があります。このビットをクリアするには、「1」を書き込みます。
[3] LATCH_REF_H_READY	RW, W1C 0x0	ホーム・リファレンス・スイッチ・イベント後にポジション値がラッチされていることを示します。ポジションのラッチを行うには、必要なエッジに対し、対応するSWITCH_MODEレジスタのビットでイネーブルする必要があります。このビットをクリアするには、「1」を書き込みます。
[2] SECOND_MOVE_EVENT	RW, W1C 0x0	モーション中にパラメータが変更されたなどの理由によって、ランプ・コントローラが反対方向への自動的な移動を行ったことを示します。このビットをクリアするには、「1」を書き込みます。
[1] STALL_STOP_EVENT	RW, W1C 0x0	ストール停止イベントが発生したことを示します。SWITCH_MODEレジスタでSTALL_STOP_ENや必要なトリガ条件ビットがセットされる必要があります。このビットが手動でクリアされると、モーターのモーションが再開される可能性がある点に注意してください。このビットおよび対応する割込み条件をクリアするには、「1」を書き込みます。
[0] POSITION_REACHED_EVENT	RW, W1C 0x0	目標ポジションに達したこと、つまり、POSITION.POSITIONとFOC.PID_POSITION_TARGETが等しくなったことを示します。このビットおよび対応する割込み条件をクリアするには、「1」を書き込みます。

0x200：EXT_CTRL.VOLTAGE

デフォルト：0x00000000

外部から書込み可能な電圧成分の値。MOTION_MODE = VOLTAGE_EXT の場合はオープン・ループ電圧制御に使用。

ビットおよび名称	タイプおよびリセット	説明
[31:16] UQ	RW, signed 0x0000	外部から書込み可能な電圧成分Uqの値。MOTION_MODE = VOLTAGE_EXTの場合はオープン・ループ電圧制御に使用。
[15:0] UD	RW, signed 0x0000	外部から書込み可能な電圧成分Udの値。MOTION_MODE = VOLTAGE_EXTの場合はオープン・ループ電圧制御に使用。

0x202：EXT_CTRL.PWM_V_U

デフォルト：0x00000000

外部から書込み可能な PWM デューティサイクルの値。MOTION_MODE = PWM_EXT の場合に使用。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PWM_V	RW, unsigned 0x0000	モーターのV相のために外部から書込み可能なPWMデューティサイクルの値。MOTION_MODE = PWM_EXTの場合に使用。
[15:0] PWM_U	RW, unsigned 0x0000	モーターのU相のために外部から書込み可能なPWMデューティサイクルの値。MOTION_MODE = PWM_EXTの場合に使用。

0x203：EXT_CTRL.PWM_W

デフォルト：0x00000000

外部から書込み可能な PWM デューティサイクルの値。MOTION_MODE = PWM_EXT の場合に使用。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PWM_W	RW, unsigned 0x0000	モーターのW相のために外部から書込み可能なPWMデューティサイクルの値。MOTION_MODE = PWM_EXTの場合に使用。

0x240：FEEDBACK.CONF_CH_A

デフォルト：0x00000000

フィードバック・エンジンのチャンネル A の設定。

ビットおよび名称	タイプおよびリセット	説明
[27:24] SRC_SEL_A	RW 0x0	フィードバック・エンジンのチャンネルAに関する角度ソースの選択。入力値はCPR_INV_Aでスケーリングされ、その後補正LUTモジュールに供給されます。
	0: ABN_1	ENC_xピンに接続されたABNエンコーダからのシングル・ターン・ポジションをデコードした値。
	1: ABN_1_FREE	ENC_xピンに接続されたABNエンコーダの「マルチ・ターン」カウント。このオプションは、速度またはポジションの測定にのみ選択し、phi_e計算用には選択しないでください。
	2: ABN_2	HALL_xピン（代替機能）に接続されたABNエンコーダからのシングル・ターン・ポジションをデコードした値。
	3: ABN_2_FREE	HALL_xピン（代替機能）に接続されたABNエンコーダの「マルチ・ターン」カウント。このオプションは、速度またはポジションの測定にのみ選択し、phi_e計算用には選択しないでください。
	4: ANALOG_SENSOR	HALL_xピン（代替機能）に接続されたアナログ・ホールまたはSinCosエンコーダからのアナログ・センサー・ポジションをデコードした値。
	5: HALL	HALL_xピンまたはENC_xピン（代替機能のENC_x）に接続されたホール・センサーからのポジションをデコードした値。このオプションはphi_eの計算にのみ選択し、速度またはポジションの測定用には選択しないでください。
	6: HALL_FREE	HALL_xピンまたはENC_xピン（代替機能のENC_x）に接続されたホール・センサーからの「マルチ・ターン」カウント。このオプションは、速度またはポジションの測定にのみ選択し、phi_e計算用には選択しないでください。
	7: PHI_EXT_A	FEEDBACK.PHI_EXT_Aレジスタからの値を使用。このオプションは、I/Oコントローラを通じてインターフェースされたエンコーダ（SPIエンコーダなど）に対し選択するか、レジスタ・アクセスを通じて外部からPHI_EXT_Aを更新する場合に選択します。
	8: PHI_EXT_B	FEEDBACK.PHI_EXT_Bレジスタからの値を使用。このオプションは、I/Oコントローラを通じてインターフェースされたエンコーダ（SPIエンコーダなど）に対し選択するか、レジスタ・アクセスを通じて外部からPHI_EXT_Bを更新する場合に選択します。
[23:0] CPR_INV_A	RW, unsigned 0x000000	16ビット変換動作のスケーリング・ファクタ。変換された出力は次式により求まります。 converted_angle = (selected_source_angle × CPR_INV_A) >> 8 SRC_SEL_AからABN_1_FREE、ABN_2_FREE、またはHALL_FREEのいずれかのオプションを選択する場合、この値を256以上の2の累乗に設定します。 それ以外のSRC_SEL_Aのオプションの場合は、この値を2^24/(選択したフィードバック・ソースのCPR)に設定します。

0x241：FEEDBACK.CONF_CH_B

デフォルト：0x00000000

フィードバック・エンジンのチャンネル B の設定。

ビットおよび名称	タイプおよびリセット	説明
[27:24] SRC_SEL_B	RW 0x0	フィードバック・エンジンのチャンネルBに関する角度ソースの選択。入力値はCPR_INV_Bでスケーリングされ、その後補正LUTモジュールに供給されます。
	0: ABN_1	ENC_xピンに接続されたABNエンコーダからのシングル・ターン・ポジションをデコードした値。
	1: ABN_1_FREE	ENC_xピンに接続されたABNエンコーダの「マルチ・ターン」カウント。このオプションは、速度またはポジションの測定にのみ選択し、phi_e計算用には選択しないでください。
	2: ABN_2	HALL_xピン（代替機能）に接続されたABNエンコーダからのシングル・ターン・ポジションをデコードした値。
	3: ABN_2_FREE	HALL_xピン（代替機能）に接続されたABNエンコーダの「マルチ・ターン」カウント。このオプションは、速度またはポジションの測定にのみ選択し、phi_e計算用には選択しないでください。
	4: ANALOG_SENSOR	HALL_xピン（代替機能）に接続されたアナログ・ホールまたはSinCosエンコーダからのアナログ・センサー・ポジションをデコードした値。
	5: HALL	HALL_xピンまたはENC_xピン（代替機能のENC_x）に接続されたホール・センサーからのポジションをデコードした値。このオプションはphi_eの計算にのみ選択し、速度またはポジションの測定用には選択しないでください。
	6: HALL_FREE	HALL_xピンまたはENC_xピン（代替機能のENC_x）に接続されたホール・センサーからの「マルチ・ターン」カウント。このオプションは、速度またはポジションの測定にのみ選択し、phi_e計算用には選択しないでください。
	7: PHI_EXT_A	FEEDBACK.PHI_EXT_Aレジスタからの値を使用。このオプションは、I/Oコントローラを通じてインターフェースされたエンコーダ（SPIエンコーダなど）に対し選択するか、レジスタ・アクセスを通じて外部からPHI_EXT_Aを更新する場合に選択します。
	8: PHI_EXT_B	FEEDBACK.PHI_EXT_Bレジスタからの値を使用。このオプションは、I/Oコントローラを通じてインターフェースされたエンコーダ（SPIエンコーダなど）に対し選択するか、レジスタ・アクセスを通じて外部からPHI_EXT_Bを更新する場合に選択します。
[23:0] CPR_INV_B	RW, unsigned 0x000000	16ビット変換動作のスケーリング・ファクタ。変換された出力は次式により求まります。 converted_angle = (selected_source_angle × CPR_INV_B) >> 8 SRC_SEL_BからABN_1_FREE、ABN_2_FREE、またはHALL_FREEのいずれかのオプションを選択する場合、この値を256以上の2の累乗に設定します。 それ以外のSRC_SEL_Bのオプションの場合は、この値を2^24/(選択したフィードバック・ソースのcounts_per_rotation)に設定します。

0x242：FEEDBACK.PHI_E_OFFSET

デフォルト：0x00000000

フィードバック・エンジンの計算された phi_e の角度に対するオフセット。選択されたチャンネルの角度に PHI_E_MUL_FACTOR が乗じられた後に適用されます。このオフセットを使用して、フィードバック・エンジンの計算された phi_e を内部生成された phi_e に一致させます。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PHI_E_OFFSET	RW, unsigned 0x0000	PHI_E_OFFSETビットフィールドは、phi_eに付加されるphi_eオフセット値を保持します。

0x243：FEEDBACK.LUT

デフォルト：0x00000000

補正 LUT モジュールの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:24] RDATA	R, signed 0x00	LUTの1つの項目からリード・バックされた値を保持します。このフィールドの値は、新しいADDR値が書き込まれた後に更新されます。
[23:16] ADDR	RW, unsigned 0x00	ルックアップ・テーブルとの間で値を読書きするためのLUTアドレスを設定します。LUT_WDATAレジスタを通じたLUT値の書込みは、一度に4項目ずつ行われるため、LUTへの入力時に、例えば、ADDR = 0、4、8、12、というように設定して、アドレスを4つおきに指定する必要があります。LUT値のRDATAフィールドへの読出しは、一度に1項目ずつ行われます。そのため、ADDR値は、LUTの項目ごとに設定する必要があります。
[14:12] LOOKUP_B_GAIN	RW 0x0	補正LUTモジュールのチャンネルBのゲイン値。
	0: GAIN1 1: GAIN2 2: GAIN4 3: GAIN8 4: GAIN16	1xゲイン 2xゲイン 4xゲイン 8xゲイン 16xゲイン
[10:8] LOOKUP_A_GAIN	RW 0x0	補正LUTモジュールのチャンネルAのゲイン値。
	0: GAIN1 1: GAIN2 2: GAIN4 3: GAIN8 4: GAIN16	1xゲイン 2xゲイン 4xゲイン 8xゲイン 16xゲイン
[2] LOOKUP_B_EN	RW 0x0	補正LUTモジュールのチャンネルBをイネーブルするには、このビットをセットします。SPLIT_MODE_ENがセットされていない場合、チャンネルAとチャンネルBのどちらか一方のみがイネーブルできます。
[1] LOOKUP_A_EN	RW 0x0	補正LUTモジュールのチャンネルAをイネーブルするには、このビットをセットします。SPLIT_MODE_ENがセットされていない場合、チャンネルAとチャンネルBのどちらか一方のみがイネーブルできます。
[0] SPLIT_MODE_EN	RW 0x0	補正LUTのスプリット・モードを有効化します。スプリット・モードを用いる場合は、チャンネルAとチャンネルBの両方をイネーブルする必要があります。

0x244：FEEDBACK.VELOCITY_FRQ_CONF

デフォルト：0x00000000

周波数速度メータの設定。

ビットおよび名称	タイプおよびリセット	説明
[23:8] VELOCITY_SCALING	RW, unsigned 0x0000	周波数速度メータのスケーリング・ファクタを設定します。この出力と周期速度メータの出力が同じスケールとなるように用いられ、それによって、動作時でも両者の切り替えが可能になります。 VELOCITY_SCALING = 2^24/6E7 × velocity_update_rateとなるように設定します。velocity_update_rateを計算する方法については、VELOCITY_SAMPLINGフィールドを参照してください。
[7:1] VELOCITY_SAMPLING	RW, unsigned 0x00	速度コントローラの更新レートを設定します。それにより、周波数速度メータのサンプリング周波数も設定されます。更新レートは次式で計算されます。 velocity_update_rate = pwm_freq / (VELOCITY_SAMPLING+1) 速度コントローラの全体としての更新レートがこのフィールドで設定されるため、周波数速度メータの代わりに周期速度メータが用いられる場合でも、このフィールドを設定する必要がある点に注意してください。
[14:12] LOOKUP_B_GAIN	RW 0x0	速度メータの同期信号としてどちらのトリガ（単一のPWMサイクルを基準）を用いるかを選択します。
	0: PWM_Z 1: PWM_C	PWM_Z（PWM周期の始め）を使用します。 PWM_C（PWM周期の中央）を使用します。

0x245：FEEDBACK.VELOCITY_PER_CONF

デフォルト：0xFFF00001

周期速度メータの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:16] POS_DEV_TIMER	RW, unsigned 0xFFF0	周期速度測定が更新されるためにPOS_DEV_MINで設定された最小角度変化が従う必要のある、タイムアウトを設定します。タイムアウトはPOS_DEV_TIMER/f_CLKSYSで計算されます。デフォルト値では、1.092msのタイムアウトになります。
[14:0] POS_DEV_MIN	RW, unsigned 0x0001	選択したチャンネルの角度変化がこの値を超えた場合のみ、周期速度測定が更新されます。更新が生じるためには、この変化がPOS_DEV_TIMERで定められた時間内に生じる必要もあります。

0x246：FEEDBACK.VELOCITY_PER_FILTER

デフォルト：0x00000000

周期速度メータの移動平均フィルタの設定。

ビットおよび名称	タイプおよびリセット	説明
[2:0] FILTER_WIDTH	RW, unsigned 0x0	周期速度測定のための移動平均フィルタのウィンドウ幅を設定します。幅を広くするとより滑らかになりますが、出力の遅延が増加します。

0x247：FEEDBACK.PHI_CONVERTED

デフォルト：0x00000000

CONF_CH_A レジスタおよび CONF_CH_B レジスタでの設定に従って 16 ビットへの変換／スケーリングを行った後、チャンネル A およびチャンネル B へのフィードバック・データを保持します。このデータは、補正 LUT モジュールの入力です。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PHI_CONVERTED_B	R, unsigned 0x0000	CONF_CH_Bレジスタでの設定に従って16ビットへの変換／スケーリングを行った後、チャンネルBへのフィードバック・データを保持します。このデータは、補正LUTモジュールの入力です。
[15:0] PHI_CONVERTED_A	R, unsigned 0x0000	CONF_CH_Aレジスタでの設定に従って16ビットへの変換／スケーリングを行った後、チャンネルAへのフィードバック・データを保持します。このデータは、補正LUTの入力です。

0x248：FEEDBACK.CH_A

デフォルト：0x00000000

LUT レジスタでの設定に従う補正 LUT モジュール後のチャンネル A へのフィードバック・データを保持します。このレジスタは、PHI_E_SRC フィールドおよび PHI_DIFF_LIMIT フィールドの設定に従う（チャンネル A またはチャンネル B のいずれかの）phi 外挿機能モジュール後のデータも保持します。

ビットおよび名称	タイプおよびリセット	説明
[31:16] PHI_EXTRAPOLATED_AB	R, unsigned 0x0000	PHI_E_SRCフィールドおよびPHI_DIFF_LIMITフィールドの設定に従う（チャンネルAまたはチャンネルBのいずれかの）phi外挿機能後のデータを保持します。
[15:0] PHI_LOOKUP_A	R, unsigned 0x0000	LUTレジスタでの設定に従う補正LUTモジュール後のチャンネルAへのフィードバック・データを保持します。LOOKUP_A_ENがセットされていない場合、PHI_CONVERTED_Aからのデータは不変のままである点に注意してください。

0x249：FEEDBACK.CH_B

デフォルト：0x00000000

LUT レジスタでの設定に従う補正 LUT モジュール後のチャンネル B へのフィードバック・データを保持します。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PHI_LOOKUP_B	R, unsigned 0x0000	LUTレジスタでの設定に従う補正LUTモジュール後のチャンネルBへのフィードバック・データを保持します。LOOKUP_B_ENがセットされていない場合、PHI_CONVERTED_Bからのデータは不変のままである点に注意してください。

0x24A：FEEDBACK.VELOCITY_FRQ

デフォルト：0x00000000

周波数速度メータの出力を保持します。

ビットおよび名称	タイプおよびリセット	説明
[31:0] VELOCITY_FRQ	R, signed 0x00000000	周波数速度メータの出力を保持します。

0x24B：FEEDBACK.VELOCITY_PER

デフォルト：0x00000000

周期速度メータの出力を保持します。

ビットおよび名称	タイプおよびリセット	説明
[31:0] VELOCITY_PER	R, signed 0x00000000	周期速度メータの出力を保持します。

0x24C：FEEDBACK.LUT_WDATA

デフォルト：0x00000000

補正 LUT に書き込まれるデータを保持します。

ビットおよび名称	タイプおよびリセット	説明
[31:0] WDATA	RW, unsigned 0x00000000	4バイトからなる32ビット値を書き込みます。LUTは、ビット[7:0]をADDRフィールドで指定されるテーブル項目に、ビット[15:8]を(ADDR+1)のテーブル項目に、というように自動的に書き込みます。

0x24D：FEEDBACK.PHI_EXT_A

デフォルト：0x00000000

外部から書込み可能なフィードバック・エンジン入力値。チャンネル A または B の 16 ビット変換／スケーリング・モジュールに転送されます。

ビットおよび名称	タイプおよびリセット	説明
[23:0] PHI_EXT_A	RW, unsigned 0x000000	対応するソースの選択がPHI_EXT_Aオプションで設定されている場合に、外部から書込み可能なフィードバック・エンジン入力値。チャンネルAまたはBの16ビット変換／スケーリング・モジュールに転送されます。このレジスタは、実行可能なエンコーダを用いている場合のI/Oコントローラにより（例えば、SPIエンコーダをアクティブ化することにより（ROMコード機能））、書き込めます。

0x24E：FEEDBACK.PHI_EXT_B

デフォルト：0x00000000

外部から書込み可能なフィードバック・エンジン入力値。チャンネル A または B の 16 ビット変換／スケーリング・モジュールに転送されます。

ビットおよび名称	タイプおよびリセット	説明
[23:0] PHI_EXT_B	RW, unsigned 0x000000	外部から書込み可能なフィードバック・エンジン入力値。対応するソースの選択がPHI_EXT_Bオプションで設定されている場合に、チャンネルAまたはBの16ビット変換／スケーリング・モジュールの入力として使用されます。このレジスタは、実行可能なエンコーダを用いている場合のI/Oコントローラにより（例えば、SPIエンコーダをアクティブ化することにより（ROMコード機能））、書き込めます。

0x24F：FEEDBACK.VELOCITY_EXT

デフォルト：0x00000000

外部から書込み可能な速度値。VELOCITY_SELECTION = VELOCITY_EXT の場合に、速度コントローラへの入力として用います。

ビットおよび名称	タイプおよびリセット	説明
[31:0] VELOCITY_EXT	RW, signed 0x00000000	外部から書込み可能な速度値。VELOCITY_SELECTION = VELOCITY_EXTの場合に、速度コントローラへの入力として用います。

0x250：FEEDBACK.OUTPUT_CONF

デフォルト：0x00000001

選択された角度ソースおよび設定された A および B チャンネルに基づく、フィードバック・エンジンの様々な出力チャンネルの設定。

ビットおよび名称	タイプおよびリセット		説明
[23:22] VELOCITY_SELECTION	RW 0x0		FOCモジュールの速度コントローラに用いられる速度ソースを選択します。
	0:	VELOCITY_FRQ	周波数速度メータ。ほとんどのアプリケーションに対し問題なく機能します。中速度から高速度の範囲において最も良好に機能します。
	1:	VELOCITY_PER	周期速度メータ。このメータは低速度アプリケーションにより適しています。ゼロ速度に安定化させる場合にも良好に機能します。
	2:	VELOCITY_EXT	外部コントローラが速度測定を処理する場合、あるいはI/Oコントローラを使用することによって、測定された速度はVELOCITY_EXTに手動で書き込まれます。
[21] VELOCITY_SRC	RW 0x0		周波数速度メータおよび周期速度メータの両方の入力として用いるLUTチャンネルを選択します。
	0:	LOOKUP_A	LUTチャンネルA
	1:	LOOKUP_B	LUTチャンネルB
[20] POSITION_SRC	RW 0x0		ポジション・マルチターン・デコーダの入力として用いるLUTチャンネルを選択します。
	0:	LOOKUP_A	LUTチャンネルA
	1:	LOOKUP_B	LUTチャンネルB
[17:16] PHI_E_SRC	RW 0x0		フィードバック・エンジンのphi_e計算モジュールのソースを選択します。
	0:	LOOKUP_A	LUTチャンネルA
	1:	LOOKUP_B	LUTチャンネルB
	2:	EXTRAPOLATOR_A	外挿された角度チャンネルA
	3:	EXTRAPOLATOR_B	外挿された角度チャンネルB
[7:0] PHI_E_MUL_FACTOR	RW, unsigned 0x01		選択されたフィードバック角度とphi_e角度の間の逓倍係数を設定します。ABNエンコーダやSinCosエンコーダなど、機械的角度を出力するフィードバック・ソースの、モーターの極ペア数に等しい値を設定します。デジタルまたはアナログのホール・センサーなど、電気的角度を出力するフィードバック・ソースに対しては、1の値をセットします。

0x251：FEEDBACK.PHI_E

デフォルト：0x00000000

クローズド・ループ動作の場合に、フィードバック・エンジンの計算された phi_e を保持します。この角度は、MCC_CONFIG.MOTOR_MOTION.RAMP_USE_PHI_E がセットされていない場合に、FOC モジュールへの入力として用いられます。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PHI_E_FOC	R, unsigned 0x0000	クローズド・ループ動作の場合に、フィードバック・エンジンの計算されたphi_eを保持します。PHI_E_SRCで選択されたチャンネルの角度をPHI_E_MUL_FACTORと掛け合わせPHI_E_OFFSETを適用することにより、計算されます。この角度は、MCC_CONFIG.MOTOR_MOTION.RAMP_USE_PHI_Eがセットされていない場合に、FOCモジュールへの入力として用いられます。

0x252：FEEDBACK.PHI_DIFF_LIMIT

デフォルト：0x00004000

このレジスタは、外挿モジュールの入力と出力の許容可能な最大偏差、つまり、実際のフィードバック角度と外挿された予測角度との偏差を示します。

ビットおよび名称	タイプおよびリセット	説明
[15:0] PHI_DIFF_LIMIT	RW, unsigned 0x4000	このレジスタは、外挿モジュールの入力と出力の許容可能な最大偏差、つまり、実際のフィードバック角度と外挿された予測角度との偏差を保持します。

0x280：ABN.CONFIG

デフォルト：0xFFFFFF02

ENC_x ピンに接続された ABN エンコーダの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:8] CPR	RW, unsigned 0xFFFFFF	COUNTをリセットする最大値を保持します。エンコーダのCPR-1に設定します。
[3] INV_DIR	RW 0x0	カウント方向を反転します。
[1] CLN	RW 0x1	Nパルスの検出時にCOUNT_N_WRITEのCOUNTへの転送をイネーブルします。
	0: OFF 1: ON	オフ オン
[0] COMBINED_N	RW 0x0	Nパルスを判定するためにA、B、Nの3つの信号のすべての論理積を使用します。
	0: ONLY_N 1: ALL	AおよびBを無視して、ヌル信号としてNパルスのみを用います。 3つの信号すべてをヌル信号として用います。

0x281：ABN.COUNT

デフォルト：0x00000000

生の ABN デコーダ・カウントを保持します。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COUNT	RW, unsigned 0x000000	生のABNデコーダ・カウントを保持します。

0x282：ABN.COUNT_N_CAPTURE

デフォルト：0x00000000

N パルス検出時にラッチされた ABN デコーダ・カウントを保持します。N_EVENT ビットをモニタして、N パルス・イベントが発生したかどうかを知ることができます。CONFIG.CLN ビットがセットされている場合、ラッチされた値は、単に COUNT_N_WRITE に設定された値に一致するだけです。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COUNT_N_CAPTURE	R, unsigned 0x000000	Nパルス検出時にラッチされたABNデコーダ・カウントを保持します。 N_EVENTビットをモニタして、Nパルス・イベントが発生したかどうかを知ることができます。CONFIG.CLNビットがセットされている場合、ラッチされた値は、単にCOUNT_N_WRITEフィールドに設定された値に一致するだけです。

0x283：ABN.COUNT_N_WRITE

デフォルト：0x00000000

N パルス検出時に COUNT に書き込まれる値を設定します。これは CONFIG.CLN ビットがセットされている場合にのみ有効です。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COUNT_N_WRITE	RW, unsigned 0x000000	Nパルス検出時にCOUNTに書き込まれる値を設定します。この値は、 CONFIG.CLNビットがセットされている場合にのみ書き込まれます。

0x284：ABN.EVENTS

デフォルト：0x00000000

ABN デコーダのイベント・ビット。

ビットおよび名称	タイプおよびリセット	説明
[1] N_EVENT	RW, W1C 0x0	Nパルス・イベントが発生しました。このビットをクリアするには、「1」を書き込みます。
[0] INVALID_SIGNAL_EVENT	RW, W1C 0x0	このビットは、AとBが同時に変化した場合にセットされます。このビットをクリアするには、「1」を書き込みます。

0x2C0：ABN2.CONFIG

デフォルト：0xFFFFFF02

HALL_x ピン（代替機能）に接続された ABN エンコーダの設定。

ビットおよび名称	タイプおよびリセット	説明
[31:8] CPR	RW, unsigned 0xFFFFFF	デコーダ・カウントをリセットする必要のある、1回転あたりの最大カウントを保持します。この値はCPR-1に設定します。
[3] INV_DIR	RW 0x0	カウント方向を反転します。
[1] CLN	RW 0x1	Nパルスの検出時にCOUNT_N_WRITEのCOUNTへの転送をイネーブルします。
	0: OFF 1: ON	オフ オン
[0] COMBINED_N	RW 0x0	Nパルスを判定するためにA、B、Nの3つの信号のすべての論理積を使用します。
	0: ONLY_N 1: ALL	AおよびBを無視して、ヌル信号としてNパルスのみを用います。 3つの信号すべてをヌル信号として用います。

0x2C1：ABN2.COUNT

デフォルト：0x00000000

生の ABN2 デコーダ・カウントを保持します。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COUNT	RW, unsigned 0x000000	生のABN2デコーダ・カウントを保持します。

0x2C2：ABN2.COUNT_N_CAPTURE

デフォルト：0x00000000

N パルス検出時にラッチされた ABN2 デコーダ入力を保持します。N_EVENT ビットをモニタして、N パルス・イベントが発生したかどうかを知ることができます。CONFIG.CLN ビットがセットされている場合、ラッチされた値は、単に COUNT_N_WRITE に設定された値に一致するだけです。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COUNT_N_CAPTURE	R, unsigned 0x000000	Nパルス検出時にラッチされたABN2デコーダ入力を保持します。N_EVENTビットをモニタして、Nパルス・イベントが発生したかどうかを知ることができます。CONFIG.CLNビットがセットされている場合、ラッチされた値は、単にCOUNT_N_WRITEフィールドに設定された値に一致するだけです。

0x2C3：ABN2.COUNT_N_WRITE

デフォルト：0x00000000

N パルス検出時に COUNT に書き込まれる値を設定します。これは CONFIG.CLN ビットがセットされている場合にのみ有効です。

ビットおよび名称	タイプおよびリセット	説明
[23:0] COUNT_N_WRITE	RW, unsigned 0x000000	Nパルス検出時にCOUNTに書き込まれる値を設定します。この値は、CONFIG.CLNビットがセットされている場合にのみ書き込まれます。

0x2C4：ABN2.EVENTS

デフォルト：0x00000000

ABN2 デコーダのイベント・ビット。

ビットおよび名称	タイプおよびリセット	説明
[1] N_EVENT	RW, W1C 0x0	Nパルス・イベントが発生しました。このビットをクリアするには、「1」を書き込みます。
[0] INVALID_SIGNAL_EVENT	RW, W1C 0x0	このビットは、AとBが同時に変化した場合にセットされます。このビットをクリアするには、「1」を書き込みます。

0x300：HALL.MAP_CONFIG

デフォルト：0x00000000

デジタル・ホール信号およびアナログ・ホール信号の割り当ての設定。

ビットおよび名称	タイプおよびリセット	説明
[6:4] ANA_HALL_MAP	RW 0x0	U/X、V/N、W/Yの各アナログ・センサー信号に対する代替割り当て。
	0: ANA_UVW	それぞれ、AIN_U、AIN_V、AIN_Wからのアナログ・センサーU/X、V/N、W/Y。
	1: ANA_UWV	それぞれ、AIN_U、AIN_W、AIN_Vからのアナログ・センサーU/X、V/N、W/Y。
	2: ANA_VUW	それぞれ、AIN_V、AIN_U、AIN_Wからのアナログ・センサーU/X、V/N、W/Y。
	3: ANA_WUV	それぞれ、AIN_W、AIN_U、AIN_Vからのアナログ・センサーU/X、V/N、W/Y。
	4: ANA_VWU	それぞれ、AIN_V、AIN_W、AIN_Uからのアナログ・センサーU/X、V/N、W/Y。
	5: ANA_WVU	それぞれ、AIN_W、AIN_V、AIN_Uからのアナログ・センサーU/X、V/N、W/Y。
[2:0] HALL_MAP	RW 0x0	ホールUVW信号の代替割り当て。
	0: UVW	それぞれ、PIN_HALL_U_RAW、PIN_HALL_V_RAW、PIN_HALL_W_RAWからのホール・センサーU、V、W。
	1: UWV	それぞれ、PIN_HALL_U_RAW、PIN_HALL_W_RAW、PIN_HALL_V_RAWからのホール・センサーU、V、W。
	2: VUW	それぞれ、PIN_HALL_V_RAW、PIN_HALL_U_RAW、PIN_HALL_W_RAWからのホール・センサーU、V、W。
	3: WUV	それぞれ、PIN_HALL_W_RAW、PIN_HALL_U_RAW、PIN_HALL_V_RAWからのホール・センサーU、V、W。
	4: VWU	それぞれ、PIN_HALL_V_RAW、PIN_HALL_W_RAW、PIN_HALL_U_RAWからのホール・センサーU、V、W。
	5: WVU	それぞれ、PIN_HALL_W_RAW、PIN_HALL_V_RAW、PIN_HALL_U_RAWからのホール・センサーU、V、W。

0x301：HALL.DIG_COUNT

デフォルト：0x00000000

デジタル・ホール・デコーダのカウント、つまり現在のホール・センサーのポジションを保持します。

ビットおよび名称	タイプおよびリセット	説明
[2:0] COUNT	R, unsigned 0x0	デジタル・ホール・デコーダのカウント、つまり現在のホール・センサーのポジションを保持します。

0x302：HALL.ANA_CONFIG

デフォルト：0x00000000

アナログ・ホールや SinCos エンコーダなどのアナログ・フィードバック・センサーの設定およびステータス・フラグ。

ビットおよび名称	タイプおよびリセット	説明
[22] ADC_CLIPPED	R 0x0	角度の計算中に少なくとも1つのアナログ・センサーの値がクリップされたことを示します。
[21] N_PULSE_OUT	R 0x0	N_PULSE_POLおよびN_PULSE_THRESHOLDに従いV/Nアナログ・センサー信号から生成されたNパルスの値を保持します。
[20] USE_N_PULSE	RW 0x0	アクティブな場合、V/Nは、SinCosエンコーダのNチャンネルとして機能します。Nイベントを表すアナログ値を定義するには、N_PULSE_THRESHOLDも設定する必要があります。
[19] TWO_CYCLE_MODE_EN	RW 0x0	1回転あたり2サイクルを利用するアナログ・センサーの場合にはこのビットをセットします。
[18] N_PULSE_POL	RW 0x0	Nパルスの極性を選択します。これは、N_PULSE_OUTがセットされるのが、V/N信号の値がN_PULSE_THRESHOLDより大きい（非反転）場合か、小さい（反転）場合かを定めます。
	0: NON_INVERTED	非反転、アクティブ・ハイ。
	1: INVERTED	反転、アクティブ・ロー。
[17] N_PULSE_EDGE	RW 0x0	USE_N_PULSEがセットされている場合に現在のアナログ・センサーの角度のラッチをトリガする、N_PULSE_OUTビットのエッジを選択します。
	0: RISING	n_pulseの立上がりエッジを使用するにはローにセットします。
	1: FALLING	Nパルスの立下がりエッジ。
[16] ANA_MODE	RW 0x0	アナログ・センサーのモードを選択します。アナログ・ホール・センサーの場合はUVWモード、SinCosエンコーダの場合はXYモードを使用します。
	0: XY	XYモード。
	1: UVW	UVWモード。
[15:0] N_PULSE_THRESHOLD	RW, signed 0x0000	V/Nアナログ・センサー信号がN_PULSE_OUTにNパルスを発生するためのスレッショルド値を書き込みます。スレッショルドは、オフセット処理およびスケーリング処理を行う前のV/N信号についてチェックされる点に注意してください。

0x303：HALL.ANA_UX_CONFIG

デフォルト：0x00000000

U/X アナログ・センサー入力のオフセットおよびスケーリングを設定します。

ビットおよび名称	タイプおよびリセット	説明
[31:16] UX_SCALE	RW, signed 0x0000	オフセットを加えた後にU/Xアナログ・センサー入力をスケーリングするために用いられる値を設定します。
[15:0] UX_OFFSET	RW, signed 0x0000	スケーリング前のU/Xアナログ・センサー入力に加えられる値を設定します。

0x304：HALL.ANA_VN_CONFIG

デフォルト：0x00000000

V/N アナログ・センサー入力のオフセットおよびスケーリングを設定します。

ビットおよび名称	タイプおよびリセット	説明
[31:16] VN_SCALE	RW, signed 0x0000	オフセットを加えた後にV/Nアナログ・センサー入力をスケーリングするために用いられる値を設定します。
[15:0] VN_OFFSET	RW, signed 0x0000	スケーリング前のV/Nアナログ・センサー入力に加えられる値を設定します。

0x305：HALL.ANA_WY_CONFIG

デフォルト：0x00000000

W/Y アナログ・センサー入力のオフセットおよびスケーリングを設定します。

ビットおよび名称	タイプおよびリセット	説明
[31:16] WY_SCALE	RW, signed 0x0000	オフセットを加えた後にW/Yアナログ・センサー入力をスケーリングするために用いられる値を設定します。
[15:0] WY_OFFSET	RW, signed 0x0000	スケーリング前のW/Yアナログ・センサー入力に加えられる値を設定します。

0x306：HALL.ANA_UX_OUT

デフォルト：0x00000000

オフセット処理およびスケーリング処理後の U/X アナログ・センサー値を保持します。

ビットおよび名称	タイプおよびリセット	説明
[15:0] UX_OUT	R, signed 0x0000	オフセット処理およびスケーリング処理後のU/Xアナログ・センサー値を保持します。

0x307：HALL.ANA_VN_OUT

デフォルト：0x00000000

オフセット処理およびスケーリング処理後の V/N アナログ・センサー値を保持します。

ビットおよび名称	タイプおよびリセット	説明
[15:0] VN_OUT	R, signed 0x0000	オフセット処理およびスケーリング処理後のV/Nアナログ・センサー値を保持します。

0x308：HALL.ANA_WY_OUT

デフォルト：0x00000000

オフセット処理およびスケーリング処理後の W/Y アナログ・センサー値を保持します。

ビットおよび名称	タイプおよびリセット	説明
[15:0] WY_OUT	R, signed 0x0000	オフセット処理およびスケーリング処理後のW/Yアナログ・センサー値を保持します。

0x309：HALL.ANA_OUT

デフォルト：0x00000000

計算されたアナログ・センサー角度およびオプションの N パルス検出時にラッチされた値を保持します。

ビットおよび名称	タイプおよびリセット	説明
[31:16] ANA_PHI	R, unsigned 0x0000	入力の逆正接計算から求められたアナログ・センサー角度を保持します。
[15:0] ANA_PHI_N_CAPTURE	R, unsigned 0x0000	Nパルス検出時にキャプチャされたアナログ・センサー角度を保持します。 Nパルス検出は、ANA_CONFIGレジスタで設定します。キャプチャが行われるようにするには、USE_N_PULSEビットをセットする必要があります。

0x30A：HALL.DIG_EVENTS

デフォルト：0x00000000

デジタル・ホール・デコーダのイベント・ビット。

ビットおよび名称	タイプおよびリセット	説明
[0] INVALID_SIGNAL_EVENT	RW, W1C 0x0	このビットがセットされるのは、デジタル・ホール信号がすべてゼロまたはすべて1の場合です。このビットをクリアするには、「1」を書き込みます。

0x340：UART.CONTROL

デフォルト：0x00040000

UART の設定

ビットおよび名称	タイプおよびリセット	説明
[31:24] RTMI_SAMPLING	RW, unsigned 0x00	RTMIブロックをトリガするためのダウン・サンプリング値。 RTMIトリガ・レート = f_PWM/(RTMI_SAMPLING + 1)
[22] RTMI_EN	RW 0x0	RTMI機能を有効化します。
[21] RTMI_CRC_EN	RW 0x0	RTMI読出し／書込みデータグラムおよび通常の書込み応答データグラムについて、CRC評価または生成をイネーブルするかどうかを定めます。
[20] NORMAL_CRC_EN	RW 0x0	通常の読出しデータグラムおよび書込み要求について、CRC評価または生成をイネーブルするかどうかを定めます。
[19] RX_FILTER_EN	RW 0x0	入力フィルタがイネーブルされている場合、受信したUART_RXDデータは、各ビット周期の予想される中央で、3回サンプリングされます。最終的なビット値は、多数決方式で決まります。つまり、3つのうちの2つで最終ビット値が決まります。これは、信頼できるデータ受信を確保し、ノイズやタイミングの変動による影響を軽減するのに役立ちます。
[18] AUTOBAUD_EN	RW 0x1	自動的なボー・レート検出をイネーブルし、MANTISSA_LIMITの設定を無視します。
[16] TIMEOUT_PRE_DIVIDER_EN	RW 0x0	タイムアウト・カウンタの4096前置分周器によるシステム・クロック分周をイネーブルします。
[12:0] MANTISSA_LIMIT	RW, unsigned 0x0000	ボー・レートをsystem_clock/MANTISSA_LIMITに定めます。最小値は5です。

0x341：UART.TIMEOUT

デフォルト：0x0000FFFF

UART のタイムアウト制限およびタイムアウト・カウント

ビットおよび名称	タイプおよびリセット	説明
[31:16] COUNTER	R, unsigned 0x0000	RXラインがアイドル中に受信した停止ビット後にカウントを開始します。カウンタは、システム・クロックまたは4096分周されたシステム・クロック（TIMEOUT_PRE_DIVIDER_ENを参照）を用いて、タイムアウト制限に達するまで、常に増加方向にカウントします。
[15:0] LIMIT	RW, unsigned 0xFFFF	タイムアウト・カウンタの上限値。RXラインがアイドル中にカウンタがこの制限値に達すると、タイムアウト・フラグがセットされます。カウントについては、TIMEOUT_PRE_DIVIDER_ENフィールドに応じて、システム・クロックまたは4096分周されたシステム・クロックが用いられます。この上限値をゼロに設定すると、タイムアウト・カウンタは非アクティブになります。

0x342：UART.STATUS

デフォルト：0x00000000

UART ステータス・ビット

ビットおよび名称	タイプおよびリセット	説明
[4] AUTOBAUD_VALID	RW 0x0	データの受信後オートボーが正しく検出された場合に、アクティブになります。
[2] TX_ACTIVE	R 0x0	UART_TXDがデータを送信している場合にアクティブになります。
[0] RX_ACTIVE	R 0x0	UART_RXDがデータを受信している場合にアクティブになります。

0x343：UART.EVENTS

デフォルト：0x00000000

UART 故障イベント。この特定のイベント・ビットに「1」を書く込むことにより、すべてのイベントがクリアされます。

ビットおよび名称	タイプおよびリセット	説明
[7] INVALID_CRC_EVENT	RW, W1C 0x0	CRCがイネーブルされ、計算されたCRCがデータグラムに付加されたCRCと異なる場合に、アクティブになります。この場合、データグラムの破棄も行われます。
[6] INVALID_STOP_BIT_EVENT	RW, W1C 0x0	いずれかのデータグラムのバイトの停止ビットが破損した場合にアクティブになります。このビットをクリアするには、「1」を書き込みます。
[5] INVALID_START_BIT_EVENT	RW, W1C 0x0	いずれかのデータグラムのバイトの開始ビットが破損した場合にアクティブになります。このビットをクリアするには、「1」を書き込みます。
[4] RX_IDLE_TIMEOUT_EVENT	RW, W1C 0x0	データグラムのバイトとバイトの間で、指定されたタイムアウト時間より長くUART_RXDピンがアイドルになっている場合に、アクティブになります。このビットをクリアするには、「1」を書き込みます。
[3] RTMI_INTERRUPT_EVENT	RW, W1C 0x0	このフラグは、まだ出力ストリームを処理している間に（PWMサイクル時間が経過したことにより）新しいPWMサイクルが再開した場合に、RTMIブロックの送信が中断されたかどうかを示します。クリアするには「1」を書き込みます。
[2] NOISY_DATA_EVENT	RW, W1C 0x0	UARTフィルタ処理がアクティブな場合に3つのサンプリング・ポイントすべてが同一（000_bまたは111_b）ではないと、アクティブになります。UARTフィルタ処理中に多数決方式を用いて最終ビット値が決まるため、UARTデータグラムはいずれにせよ処理されます。このビットをクリアするには、「1」を書き込みます。
[0] UART_SYNC_FAIL_EVENT	RW, W1C 0x0	UARTモジュールがビットの何らかの破損を検出したか、（オートボーがイネーブルされている場合に）ビットを正しく受信しなかった場合、または、データグラムが（バイト0の）誤った同期ビットで送信され、オートボーがディスエーブルされている場合に、アクティブになります。このビットをクリアするには、「1」を書き込みます。

0x344：UART.RTMI_CH_0

デフォルト：0x00000000

RTMI チャンネル 0 の設定。

ビットおよび名称	タイプおよびリセット	説明
[16] CH_0_EN	RW 0x0	RTMIがイネーブルの場合に、UART_TXDを通じたチャンネル0のデータ送信をイネーブルします。
[9:0] CH_0_ADDR	RW, unsigned 0x000	レジスタ・バンク・アドレスに割り当てられたRTMIチャンネル0のアドレス。

0x345：UART.RTMI_CH_1

デフォルト：0x00000000

RTMI チャンネル 1 の設定。

ビットおよび名称	タイプおよびリセット	説明
[16] CH_1_EN	RW 0x0	RTMIがイネーブルの場合に、UART_TXDを通じたチャンネル1のデータ送信をイネーブルします。
[9:0] CH_1_ADDR	RW, unsigned 0x000	レジスタ・バンク・アドレスに割り当てられたRTMIチャンネル1のアドレス。

0x346：UART.RTMI_CH_2

デフォルト：0x00000000

RTMI チャンネル 2 の設定。

ビットおよび名称	タイプおよびリセット	説明
[16] CH_2_EN	RW 0x0	RTMIがイネーブルの場合に、UART_TXDを通じたチャンネル2のデータ送信をイネーブルします。
[9:0] CH_2_ADDR	RW, unsigned 0x000	レジスタ・バンク・アドレスに割り当てられたRTMIチャンネル2のアドレス。

0x347：UART.RTMI_CH_3

デフォルト：0x00000000

RTMI チャンネル 3 の設定。

ビットおよび名称	タイプおよびリセット	説明
[16] CH_3_EN	RW 0x0	RTMIがイネーブルの場合に、UART_TXDを通じたチャンネル3のデータ送信をイネーブルします。
[9:0] CH_3_ADDR	RW, unsigned 0x000	レジスタ・バンク・アドレスに割り当てられたRTMIチャンネル3のアドレス。

0x348：UART.RTMI_CH_4

デフォルト：0x00000000

RTMI チャンネル 4 の設定。

ビットおよび名称	タイプおよびリセット	説明
[16] CH_4_EN	RW 0x0	RTMIがイネーブルの場合に、UART_TXDを通じたチャンネル4のデータ送信をイネーブルします。
[9:0] CH_4_ADDR	RW, unsigned 0x000	レジスタ・バンク・アドレスに割り当てられたRTMIチャンネル4のアドレス。

0x349：UART.RTMI_CH_5

デフォルト：0x00000000

RTMI チャンネル 5 の設定。

ビットおよび名称	タイプおよびリセット	説明
[16] CH_5_EN	RW 0x0	RTMIがイネーブルの場合に、UART_TXDを通じたチャンネル5のデータ送信をイネーブルします。
[9:0] CH_5_ADDR	RW, unsigned 0x000	レジスタ・バンク・アドレスに割り当てられたRTMIチャンネル5のアドレス。

0x34A：UART.RTMI_CH_6

デフォルト：0x00000000

RTMI チャンネル 6 の設定。

ビットおよび名称	タイプおよびリセット	説明
[16] CH_6_EN	RW 0x0	RTMIがイネーブルの場合に、UART_TXDを通じたチャンネル6のデータ送信をイネーブルします。
[9:0] CH_6_ADDR	RW, unsigned 0x000	レジスタ・バンク・アドレスに割り当てられたRTMIチャンネル6のアドレス。

0x34B：UART.RTMI_CH_7

デフォルト：0x00010000

RTMI チャンネル 7 の設定。UART 書込み要求への応答を取得するには、チャンネル 7 をイネーブルにする必要もあります。

ビットおよび名称	タイプおよびリセット	説明
[16] CH_7_EN	RW 0x1	RTMIがイネーブルの場合に、UART_TXDを通じたチャンネル7のデータ送信をイネーブルします。デフォルトでは、RTMIがディスエーブル時に通常の書込み要求データグラムに対する応答を送信するために、このチャンネルはイネーブルされています。
[9:0] CH_7_ADDR	RW, unsigned 0x000	RTMIがイネーブルされている場合、レジスタ・バンク・アドレスに割り当てられたRTMIチャンネル7のアドレス。RTMIがディスエーブルの場合は、RTMIチャンネル7のアドレスが、以前の書込みアドレスに割り当てられ、レジスタ・バンクに書き込まれた以前の値が取得されます。

0x380：IO_CONTROLLER.CONTROL

デフォルト：0x00000000

I/O コントローラの制御信号。

ビットおよび名称	タイプおよびリセット	説明
[0] RESET	RW 0x0	I/Oコントローラをリセットするには、このビットをセットします。 GDRV_ON状態の間にアクティブにすると、SYS_READY状態にリセットされるだけです。あるいは、SYS_READY状態の間にアクティブにすると、I/Oコントローラはリセットされ、チップのスタートアップ・プロセス（ADCの再設定を含む）が実行されます。また低電力モードも終了します。

0x381：IO_CONTROLLER.COMMAND

デフォルト：0x00000000

I/O コントローラのコマンド・レジスタ。

ビットおよび名称	タイプおよびリセット	説明
[31:0] COMMAND	RW, unsigned 0x00000000	I/Oコントローラのコマンド・レジスタ。

0x382：IO_CONTROLLER.RESPONSE_0

デフォルト：0x00000000

I/O コントローラの応答 0 レジスタ。

ビットおよび名称	タイプおよびリセット	説明
[31:0] RESPONSE_0	RW, unsigned 0x00000000	I/Oコントローラの応答0レジスタ。応答は、要求されたコマンドに依存します。すべてのROMコード機能に対しハンドシェイク・レジスタとして機能します。RESPONSE_0がCOMMANDレジスタの値に等しい場合、要求されたROMコードは、ワンタイム要求の場合は終了し、繰り返し要求の場合は有効な応答データ（RESPONSE_1～3）を示します。

0x383：IO_CONTROLLER.RESPONSE_1

デフォルト：0x00000000

I/O コントローラの応答 1 レジスタ。

ビットおよび名称	タイプおよびリセット	説明
[31:0] RESPONSE_1	RW, unsigned 0x00000000	I/Oコントローラの応答1レジスタ。応答は、要求されたコマンドに依存します。

0x384：IO_CONTROLLER.RESPONSE_2

デフォルト：0x00000000

I/O コントローラの応答 2 レジスタ。

ビットおよび名称	タイプおよびリセット	説明
[31:0] RESPONSE_2	RW, unsigned 0x00000000	I/Oコントローラの応答2レジスタ。応答は、要求されたコマンドに依存します。

0x385：IO_CONTROLLER.RESPONSE_3

デフォルト：0x00000000

I/O コントローラの応答 3 レジスタ。

ビットおよび名称	タイプおよびリセット	説明
[31:0] RESPONSE_3	RW, unsigned 0x00000000	I/Oコントローラの応答3レジスタ。応答は、要求されたコマンドに依存します。

型番

PART NUMBER	TEMPERATURE RANGE	PIN-PACKAGE
TMC6460ATU+	-40°C to +125°C	38 TQFN 5mm × 7mm
TMC6460ATU+T	-40°C to +125°C	38 TQFN 5mm × 7mm

+は鉛（Pb）フリー／RoHS 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：　TMC6460
対象となるデータシートのリビジョン(Rev)：Rev.0
訂正箇所：46 頁、上から 10 行目

【誤】
「次のフィルタ・カットオフ周波数となるように設定できます。つまり、設定値が 00b、01b、10b、11b に対し、それぞれ 1MHz、500kHz、100kHz です。」

【正】
「次のフィルタ・カットオフ周波数となるように設定できます。つまり、設定値が 00b、01b、10b、11b に対し、それぞれスルー、1MHz、500kHz、100kHz 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：　TMC6460
対象となるデータシートのリビジョン(Rev)：Rev.0
訂正箇所：65 頁、箇条書き 5 の欄、最後の文

【誤】
「両チャンネルに対し LUT をイネーブルする場合は、FEEDBACK.LUT レジスタの SPLIT_MODE_EN ビットをセットし、LOOKUP_EN_A および **LOOKUP_EN_A** の両方のビットをセットします。」

【正】
「両チャンネルに対し LUT をイネーブルする場合は、FEEDBACK.LUT レジスタの SPLIT_MODE_EN ビットをセットし、LOOKUP_EN_A および **LOOKUP_EN_B** の両方のビットをセットします。」

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：　TMC6460
対象となるデータシートのリビジョン(Rev)：Rev.0
訂正箇所：86 頁、「外部 PWM デューティサイクル・モード」の項、上から 4 行目

【誤】
「外部 PWM デューティサイクル・モードは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドを **9（PWM_EXT）** に設定することで有効化できます」

【正】
「外部 PWM デューティサイクル・モードは、MCC_CONFIG.MOTOR_MOTION レジスタの MOTION_MODE フィールドを **10（PWM_EXT）** に設定することで有効化できます」

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：　TMC6460
対象となるデータシートのリビジョン(Rev)：Rev.0
訂正箇所：161 頁、「[5:4] VELOCITY_NORM_P」の説明欄

【誤】
「それにより、実質的に、レジスタ・フィールドの**係数 I** の値が 256^VELOCITY_NORM_P によって除算され、結果のフラクショナル部は保持されます。」

【正】
「それにより、実質的に、レジスタ・フィールドの**係数 P** の値が 256^VELOCITY_NORM_P によって除算され、結果のフラクショナル部は保持されます。」

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