

Design Guide: TIDM-02013

EV/HEV (電気自動車とハイブリッド電気自動車) 向け、GaN 採用、7.4kW 双方向オンボードチャージャのリファレンスデザイン



概要

PMP22650 リファレンス デザインは、7.4kW の双方向オンボードチャージャを提示します。このデザインは、アクティブ同期整流機能付きの 2 相トータムポール PFC とフルブリッジ CLLLC コンバータを搭載しています。CLLLC は周波数変調と位相変調の両方を活用し、電圧の全範囲にわたって出力のレギュレーションを実施します。このデザインでは、単一の TMS320F280039C マイクロコントローラを使用して PFC 段と DCDC 段の両方を制御します。また、このデザインは TMS320F28P65x マイクロコントローラでもサポートされています。複数の高速 GaN スイッチ (LMG3522-Q1) を使用して、高密度を達成しています。電力密度が 3.8kW/リットルのオープンフレーム電源との組み合わせで、96.5% のピークシステム効率を達成しています。

このデザインでは、閉電圧および閉電流ループモードで、単一の C2000™ マイクロコントローラを使ってこの電源トポロジを制御する方法を示します。このリファレンスデザインのハードウェアとソフトウェアは開発期間の短縮を可能にします。

リソース

TIDM-02013、PMP22650	デザインフォルダ
TMS320F280039C、TMS320F28P650DK	プロダクトフォルダ
AMC3330-Q1、AMC3302-Q1、UCC21222-Q1	プロダクトフォルダ
C2000WARE-DIGITAL-POWERSDK	ソフトウェアフォルダ
TMDSCNCD280039C、TMDSCNCD28P65X	ツールフォルダ



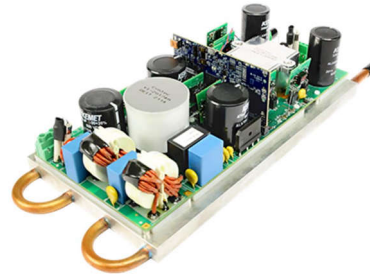
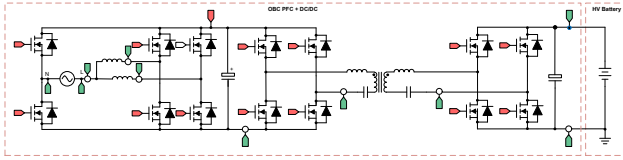
テキサス・インスツルメンツの™ E2E サポート エキスパートにお問い合わせください。

特長

- 最大電力:7.4kW、ピーク効率:96.5%
- AC 90V~264V、AC:AC 240V TYP
- Vprim:公称 DC 400V、Vsec:DC 250V~450V
- 公称 500kHz の PWM スイッチング (200kHz~800kHz の範囲) を行う CLLLC 共振タンクにより電力密度を向上
- 1 次側のソフト スイッチングおよびゼロ電圧スイッチング (ZVS) と、2 次側のゼロ電流スイッチング (ZCS) および ZVS により効率を向上
- ロゴスキー コイル センサを使用したアクティブ同期整流方式を実装することで効率を向上
- 制御補償器アクセラレータ (CLA) を搭載した TMS320F28003x デバイスをソフトウェアでサポートしているため、単一の C2000 マイクロコントローラを使用して AC/DC および DC/DC 制御を統合した OBC 設計が実現可能
- 1 つの CPU (CPU1) で AC/DC 段と DC/DC 段を制御する TMS320F28P65x デバイスのソフトウェア サポートを提供
- TMS320F28P65x の新しいハードウェア オーバーサンプリング機能により CPU オーバーヘッドを低減

アプリケーション

- ハイブリッド車、電気自動車、パワートレイン システム
- DC 高速充電ステーション
- 電力変換システム (PCS)



1 CLLLC システムの説明

オンボードチャージャ (OBC) は、電気自動車 (EV) やハイブリッド電気自動車 (HV) に不可欠な部品です。図 1-1 に示すように、OBC は通常、AC/DC (力率補正 (PFC) 整流段) と絶縁型 DC/DC コンバータで構成されています。C2000 マイクロコントローラは、車載アプリケーションで求められる高度なデジタル電源制御を実装するように設計されています。詳細については、『C2000 デジタル電源』と『C2000 EV』を参照してください。

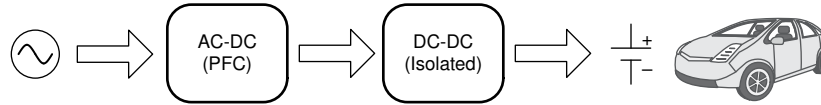


図 1-1. 代表的な OBC アーキテクチャ

ほとんどの EV レベル 1 およびレベル 2 のチャージャでは、一晩でバッテリーをフル充電できることが強く求められています。バッテリー容量が増加するにつれて、OBC はさらに大電力用に設計される必要があります。OBC の電力容量が大きくなると、車内のスペースや冷却能力に余裕がなくなるため、電力密度や効率などの仕様がますます重要になります。

CLLLC (コンデンサ - インダクタ - インダクタ - インダクタ - コンデンサ) は、対称型のタンク、ソフトスイッチング特性、高周波数でのスイッチング能力を備えており、このようなアプリケーションに適しています。このデザインでは、図 1-1 に示す CLLLC トポロジの制御と実装について説明します。

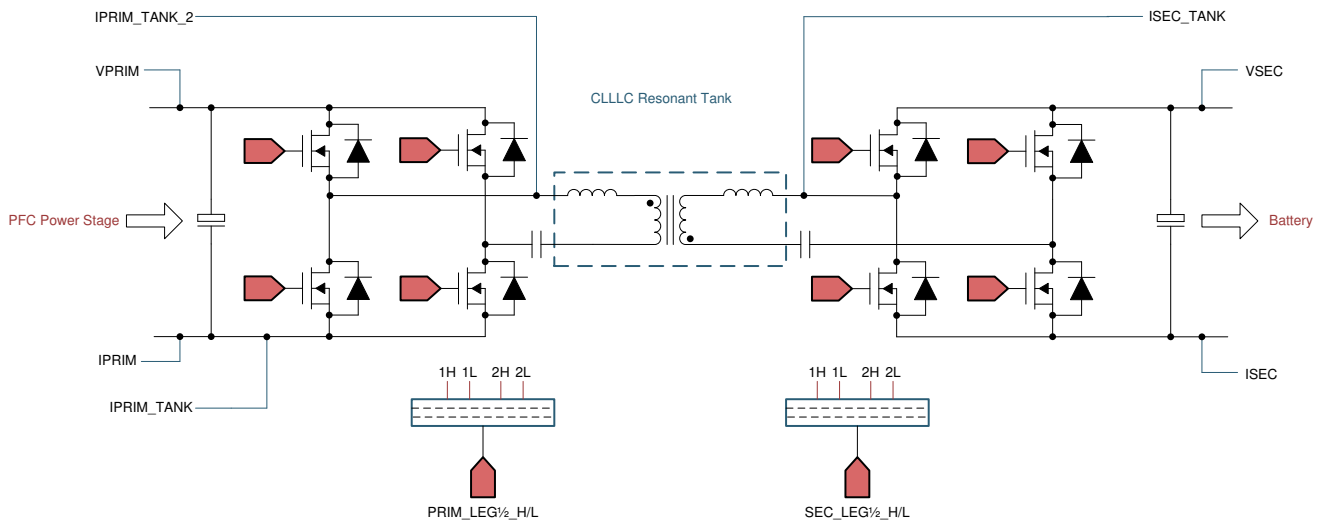


図 1-2. 絶縁型 DC/DC コンバータの CLLLC トポロジ

図 1-1 の用語は以下のとおりです。

VPRIM	1 次側電圧 (通常は PFC コンバータから供給されます)
IPRIM	1 次側のリターン電流。保護と監視に使用できます。
IPRIM_TANK, IPRIM_TANK_2	1 次側のタンク電流。シャント電流センシングとログスキー コイルを使用した 2 つのセンシング方法があります。一つだけが必要で、たとえば、2 次側から 1 次側への逆方向の同期整流に使用されます。また、保護にも使用されます。
VSEC	2 次側電圧 (通常はバッテリーです)
ISEC	2 次側のリターン電流。バッテリー電流制御ループに使用されます。
ISEC_TANK	2 次側のタンク電流。たとえば、1 次側から 2 次側への順方向の電力フローの同期整流に使用されます。
PRIM_LEG1/2_H/L	1 次側フルブリッジ用 PWM
SEC_LEG1/2_H/L	2 次側フルブリッジ用 PWM

1.1 主なシステム仕様

CLLLC リファレンス デザインの電力仕様を [表 1-1](#) に示します。

表 1-1. 主なシステム仕様

パラメータ	仕様
1 次側電圧 (Vprim)	DC 400V~450V (平均)
2 次側電圧 (Vsec)	DC 250V~450V 最大値
順方向電流定格	7.4kW
出力電流 (I _{out})	20 最大
効率 (CLLLC)	ピーク 98%
PWM スイッチング周波数	公称 500kHz (200kHz~800kHz の範囲)



警告

テキサス・インスツルメンツは、このリファレンス デザインをラボ環境のみで使用するものとし、一般消費者向けの完成品とはみなしておりません。このデザインは室温で動作することを意図しており、他の周囲温度での動作はテストされていません。

テキサス・インスツルメンツは、このリファレンス デザインを高電圧電気機械部品、システム、およびサブシステムの取り扱いに伴うリスクを熟知した**有資格のエンジニアおよび技術者**のみが使用するものとしています。

基板上は高電圧状態になっており、接触するおそれがあります。基板は、不適切に取り扱ったり、使用した場合に感電、火災、けがの原因となる電圧および電流で動作します。けがをしたり、物品を破損しないために必要な注意と適切な対策をもって機器を使用してください。



注意

電源を入れたままその場を離れないでください。



警告

高電圧！基板上は高電圧状態になっており、接触するおそれがあります。感電する可能性があります。基板は、不適切に取り扱った場合に感電、火災、けがの原因となる電圧および電流で動作します。けがをしたり、物品を破損しないために必要な注意と適切な対策をもって機器を使用してください。安全のため、過電圧/過電流保護機能を備え、絶縁された試験装置の使用を強くお勧めします。

TI は、電圧/絶縁要件を確認 理解した上で基板やシミュレーションにて電圧を加えることをユーザーの責任と考えます。電源投入中は、回路およびその接続部品には触れないでください。



警告

表面は高温！触れるとやけどの原因になることがあります。触れないでください！

基板の電源を入れると、一部の部品は 55°C を超える高温に達することがあります。動作中は常に、また動作直後も高温の状態が続く可能性があるため、基板に触れてはいけません。

2 CLLLC システムの概要

2.1 ブロック図

図 1-1 に、CLLLC トポロジのブロック図を示します。

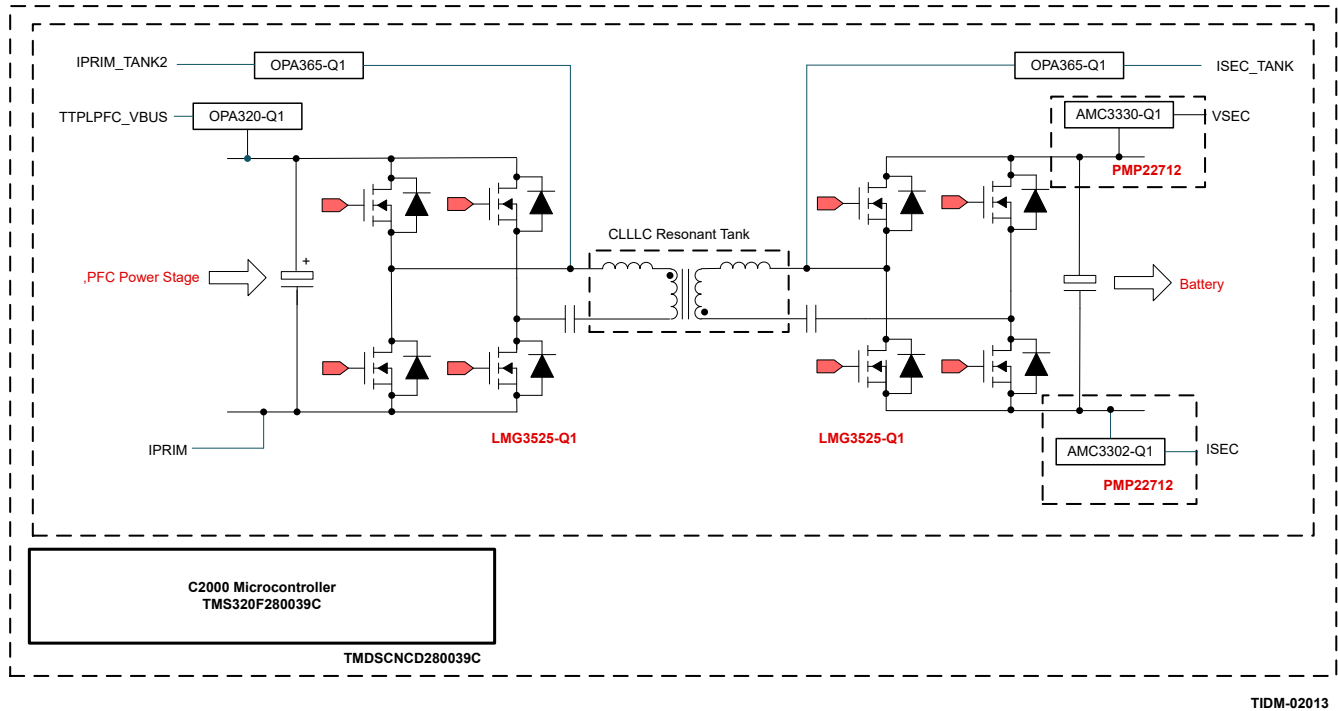


図 2-1. CLLLC のブロック図

このリファレンス デザインでは、このガイドに記載されている動作を実現するために、以下の評価基板と PMP デザインを使用しています。

1. OBC ベース ボード: PMP22650 (PMP22712 + PMP22773 ドーターカード搭載)
2. F280039C controlCARD 評価基板: TMDSCNCD280039C

以降のセクションでは、ハードウェア、ソフトウェア、システム設計の詳細について説明します。

2.2 設計上の考慮事項とシステム設計理論

LLC コンバータは、1 次側で ZVS、2 次側で ZCS を実現することができるため、広く普及しています。代表的な LLC 直列共振コンバータ (SRC) を図 1-1 に示します。このコンバータの 1 次側はハーフブリッジであるため、ボルト秒の観点から考えると、トランスの利用率は半分になります。さらに、スイッチの電流定格は、フルブリッジ構造を使用する場合に比べて、必要な電流の 2 倍となっています。

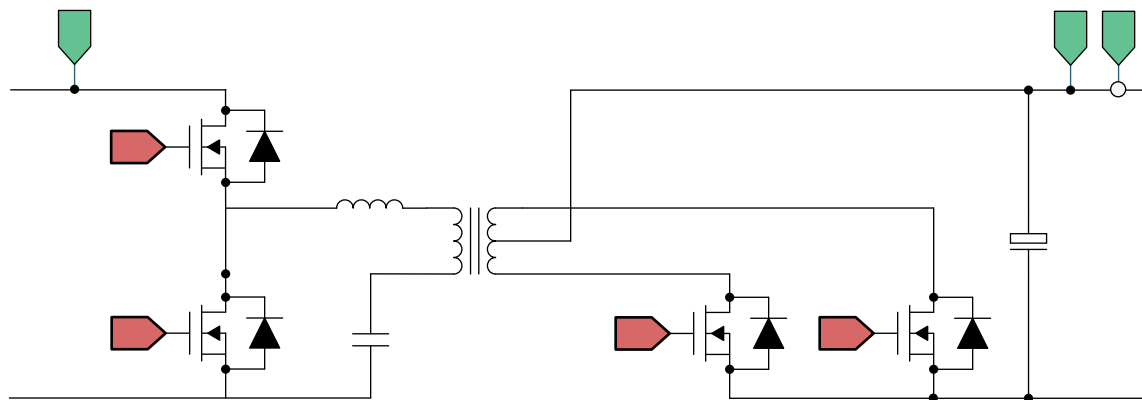


図 2-2. LLC ハーフブリッジ SRC

ハーフブリッジ LLC SRC はコスト的には低消費電力で魅力的ですが、大電力で高密度のアプリケーションでは、次の理由からフルブリッジ LLC SRC が必要とされます。

1. フルブリッジ LLC コンバータは、2 次側と 1 次側の両方でトランスの磁気コアをより効果的に活用するため、より高い電力密度を提供することができます。
2. フルブリッジ LLC コンバータは電流定格を低減するため、配線内の銅にかかるコストが削減できます。また、このコンバータは、同じ銅線で大電力 (ハーフブリッジ SRC と比べて) を実現することができます。

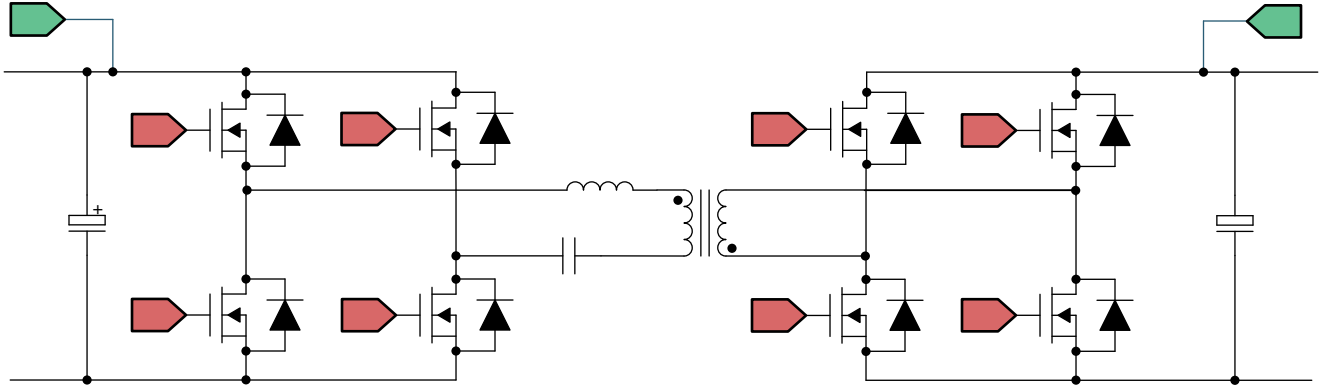


図 2-3. フルブリッジ LLC コンバータ

図 1-1 に示すように、フルブリッジ LLC コンバータは、広い意味でデュアル アクティブ ブリッジ (DAB) コンバータに分類されます。DAB コンバータでは、コンバータはモデルまたは動作に基づいて分類できます。

1. 位相シフト型 DAB コンバータは、歴史的に見て、最も広く普及しているコンバータの一つです。
2. 共振 DAB コンバータでは、共振タンクにさまざまなバリエーションがあります (LC、LLC、CLLLC、CLLLC など)。

共振 DAB コンバータは、高効率、大電力、高密度が実現できることから、高い関心を集めています。CLLLC は、対称型のタンクによって双方向動作が可能です。LLC 構造を双方向に使用する際の問題点は、逆方向の電力フローモードで動作する場合のスイッチング周波数が、トランスの巻線容量と漏れインダクタンスに左右されることです。このため、電力段のゲインやスイッチング周波数はほぼ制御できません。このことから、CLLLC タイプの構造は、スイッチング周波数がより制御しやすく、ゲインの自由度が増すため、好まれています。

2.2.1 タンク的设计

このセクションでは、必要な電圧ゲインやソフト スwitching 特性に基づいて CLLLC のタンク パラメータの選択について説明し、CLLLC に基づいてチャージャに適した電力プロファイルを選択します。

その他の計算や情報については、ソフトウェア インストール パッケージ (C2000Ware_DigitalPower_SDK_<ver>/solution/tidm_02013/hardware/) 内にある以下のファイルを参照してください。

2.2.1.1 電圧ゲイン

タンク的设计を理解するには、最初に、第 1 高調波近似を使用した第 1 高調波分析 (FHA) で、バッテリー充電モードと逆方向パワーフローモードの両方のゲインを分析する必要があります。共振タンクの概略図を図 1-1 に示します。

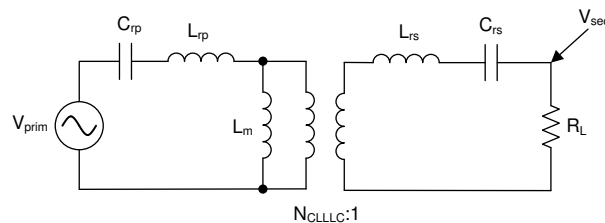


図 2-4. バッテリー充電モード (BCM) 時の CLLLC 共振タンクの FHA モデル

図 1-1 の用語は以下のとおりです。

V_{prim} (TTPLPFC_V _{BUS})	1 次側の入力電圧
L_{rp}	1 次側の共振インダクタ
C_{rp}	1 次側の共振コンデンサ
N_{CLLLC}	トランスの巻線比
L_{m}	磁化インダクタ
V_{sec}	2 次側の出力電圧
L_{rs}	2 次側の共振インダクタ
C_{rs}	2 次側の共振コンデンサ
R_{L}	2 次側出力で FHA を使用した場合の実効負荷

$$R_{\text{L}} = \left(\frac{8}{\pi^2} \right) R_{\text{L_dc}}$$

ここで、実効 R_{L} は $R_{\text{L_dc}}$ として計算され、 $R_{\text{L_dc}}$ は出力における DC 抵抗性負荷であることに注意してください。

1 次側で 2 次側の数量を参照すると、以下のようになります。

- L'_{rs} は $L_{\text{rs}} * N_{\text{CLLLC}} * N_{\text{CLLLC}}$ と等しい
- C'_{rs} は $C_{\text{rs}} / (N_{\text{CLLLC}} * N_{\text{CLLLC}})$ と等しい
- R'_{L} は $R_{\text{L}} * (N_{\text{CLLLC}} * N_{\text{CLLLC}})$ と等しい
- V'_{rs} は $V_{\text{rs}} * N_{\text{CLLLC}}$ と等しい

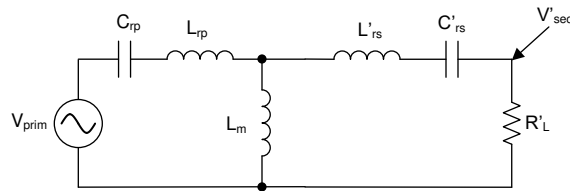


図 2-5. BCM で 1 次側の数量を参照した FHA CLLLC

KCI と KVL を使用すると、ゲイン計算式は 式 1 のように記述できます。

$$\frac{V_{\text{sec}}}{V_{\text{prim}}} = \frac{\left[Z_{\text{m}} \parallel (Z'_{\text{rs}} + R'_{\text{L}}) \right] R'_{\text{L}}}{\left(Z_{\text{rp}} + \left[Z_{\text{m}} \parallel (Z'_{\text{rs}} + R'_{\text{L}}) \right] \right) (Z'_{\text{rs}} + R'_{\text{L}}) N_{\text{CLLLC}}} \quad (1)$$

同様に、逆方向の電力フローについても、回路は 図 1-1 に示すように簡略化でき、ゲインは 式 2 のように記述できます。

$$\frac{V_{\text{prim}}}{V_{\text{sec}}} = \frac{N_{\text{CLLLC}} \left[Z_{\text{m}} \parallel (Z_{\text{rp}} + R_{\text{L}}) \right] R_{\text{L}}}{\left(Z'_{\text{rs}} + \left[Z_{\text{m}} \parallel (Z_{\text{rp}} + R_{\text{L}}) \right] \right) (Z_{\text{rp}} + R_{\text{L}})} \quad (2)$$

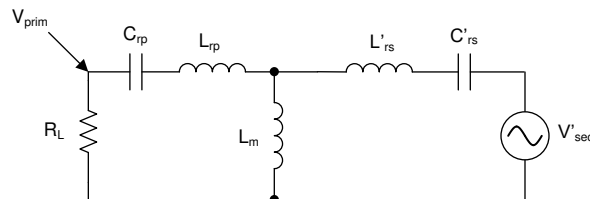


図 2-6. RCM におけるゲイン計算の FHA モデル

式 1 と 式 2 は、設計用に選択されたパラメータに基づいて電圧ゲインを調べるために、以下のセクションで使用されません。

2.2.1.2 トランス ゲイン比の設計 (N_{CLLLC})

共振コンバータは通常、共振周波数ちょうどまたはその付近で動作するときに最も効率的です。今回の回路は双方向バッテリーチャージャなので、このデザインは特定の出力電圧範囲に対応する必要があります。このことは、 I^2R 損失を低減するために、コンバータができるだけ小さい電流で動作できるように、 n を選択する必要があることを暗黙的に意味します。このデザインコンセプトに従えば、最大電力を供給する必要がある最小出力電圧で、最大出力電流が流れることとなります。コンバータができるだけ共振に近い状態で動作するこのポイントに合わせて、 n を設定します。このデザインでは、巻線比は 1.1:1 になります。これにより、広い出力電圧範囲を許容しながら、損失を最小限に抑えることができます。

2.2.1.3 磁化インダクタンスの選択 (L_m)

1 次側 FET の ZVS 動作を確実に実現するには、共振タンクに蓄積されたエネルギーが FET の出力コンデンサに蓄積されたエネルギーより大きいことを確認する必要があります。式 3 を使用して、フルブリッジ LLC SRC に必要な L_m を割り出すことができます。

$$L_m \leq \frac{T_{\text{dead}}}{16 * C_{\text{OSS}}} \quad (3)$$

ここで、コンバータの想定されるスイッチング周波数は 500kHz であるため、 $T = 1/(500 * 10^3)$ となり、パワー デバイスに基づいています。 t_{dead} や C_{OSS} のような選択されたパラメータは、パワー デバイスのデータシートから特定することもできます。通常、実効 C_{OSS} は、曲線近似を使用して計算する必要があります。このデザインでは、説明した設計パラメータに基づき、 L_m は 20 μH 未満でなければなりません。上記の計算で考慮されている要素に加えて、実際のトランスには共振タンク電流によって放電される必要のある巻線間静電容量があります。したがって、シミュレーションを使用して、コンバータの動作範囲全体で ZVS を確実に実現するために 14 μH の値が選択されました。この値は以降の選択プロセスで使用されます。

2.2.1.4 共振インダクタとコンデンサの選択 (L_{rp} と C_{rp})

L_{rp} を選択する際には、 L_m と L_{rp} の比が設計パラメータとして広く使用されます。

$$L_n = \frac{L_m}{L_{rp}} \quad (4)$$

L_n 値は、共振タンクの電圧ゲインがコンバータの動作範囲全体で十分になるように選択されます。このデザインでは、入力電圧が PFC 段から供給され、推定 10% のリップルが発生するため、10% 以上のゲイン変動が必要です。この基準と、インダクタ値、およびそれによる損失を低減するために L_n を高く維持する必要があるということを考慮に入れて、負荷によって L_n が変化する FHA のプロットに基づいて、このデザインでは L_n は 14 が選択されます(図 1-1 を参照)。

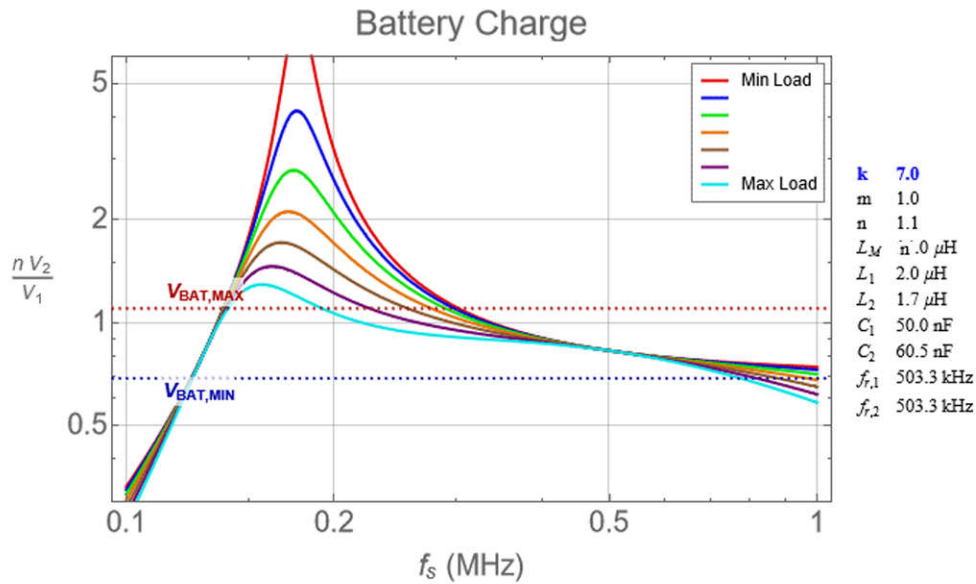


図 2-7. L_n の変化による CLLLC タンクのゲイン変動

L_n を選択したので、式 4 を使用して L_{rp} を計算できます。 L_{rp} と C_{rp} はコンバータの直列共振周波数を決定し、式 5 で表されます。

$$f_{res} = \frac{1}{2\pi\sqrt{L_{rp}C_{rp}}} \quad (5)$$

式 5 は、設計に必要な C_{rp} を計算するために使用できます。ただし、部品の入手可能状況により、設計には次に近い値の C_{rp} が使用されます。これらの部品値による BCM ゲインは、図 1-1 のとおりです。

図 1-1 では、負荷が増加する ($R_{L_{dc}}$ が低下する) と、直列共振周波数未満の領域でゲイン曲線は非単調になります。これによって、ZVS が 1 次側 FET で失われ、さらに深刻なことに、制御ができなくなることにつながります。このため、公称 V_{out} での最大負荷を想定した場合、負荷は $R_{L_{dc}} = 30\Omega$ に制限またはクランプされ、このときゲインは単調になります (図 1-1 を参照)。

さらに 図 1-1 は、BCM では、200kHz~800kHz の動作周波数全体で十分なゲインがあり、すべての動作条件に対応できます。最後に、注目すべきは、PFC リップルを低減できれば、想定される入力範囲も小さくなるということです。これにより、必要なゲイン範囲が小さくなり、最終的にはすべての負荷条件に対応するために必要な周波数変動を低減できます。

2.2.2 電流および電圧センシング

以下のセクションでは、設計上のさまざまな電流および電圧に対するセンシング方式について説明します。設計上、ユーザーがアプリケーションのニーズに適した方式を選択できるように、複数の方式が実装されています。

2.2.2.1 VPRIM 電圧センシング

C2000 マイクロコントローラは 1 次側でバイアスされているため、1 次側電圧は、基板のグラウンドに接続された抵抗デバイダによってセンシングされます。オーバーサンプリングが使用されるため、図 1-1 に示すように、ボルテージ フォロワ構成のオペアンプを使用して ADC の信号をバッファします。バッファは ADC で認識されるインピーダンスを低減するのに役立つため、より高速なサンプリング レートを使用することができます。そうでなければ、サンプリングは通常は高い抵抗分圧器の抵抗の時定数によって制限されるため、低速サンプリングしかできません。

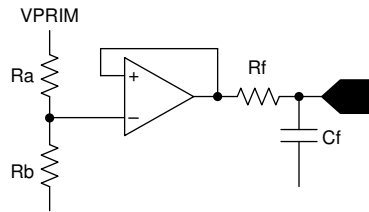


図 2-8. VPRIM 電圧センシング回路

2.2.2.2 VSEC 電圧センシング

図 1-1 に示すように、2 次側電圧は AMC3330-Q1 を使用して絶縁された方法でセンシングされます。

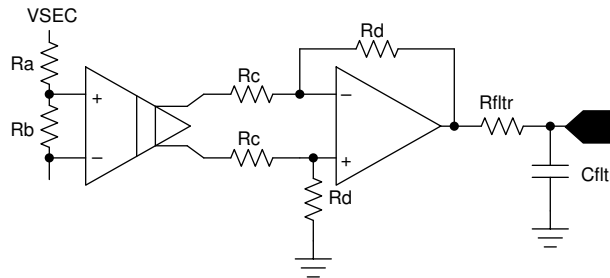


図 2-9. VSEC 電圧センシング回路

2.2.2.3 ISEC 電流センシング

図 1-1 に示すように、2 次側出力電流は AMC3302-Q1 を使用して絶縁された方法でセンシングされます。

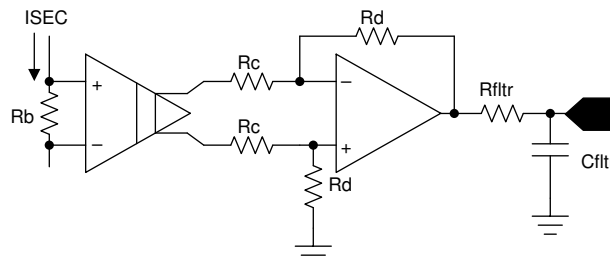


図 2-10. ISEC 電流センシング回路

2.2.2.4 ISEC タンクおよび IPRIM タンク

図 1-1 に示すように、1 次側と 2 次側のタンクにおける高周波電流を絶縁された方法でセンシングするために、ロゴスキー コイルをベースにしたセンシング メカニズムが選択されています。ADC ピンはコンパレータ サブシステム (CMPSS) に内部接続されており、これによって、同期整流に必要な動作を実現するために、X-Bar を介して PWM に流れる適切なパルスを生成することができます。

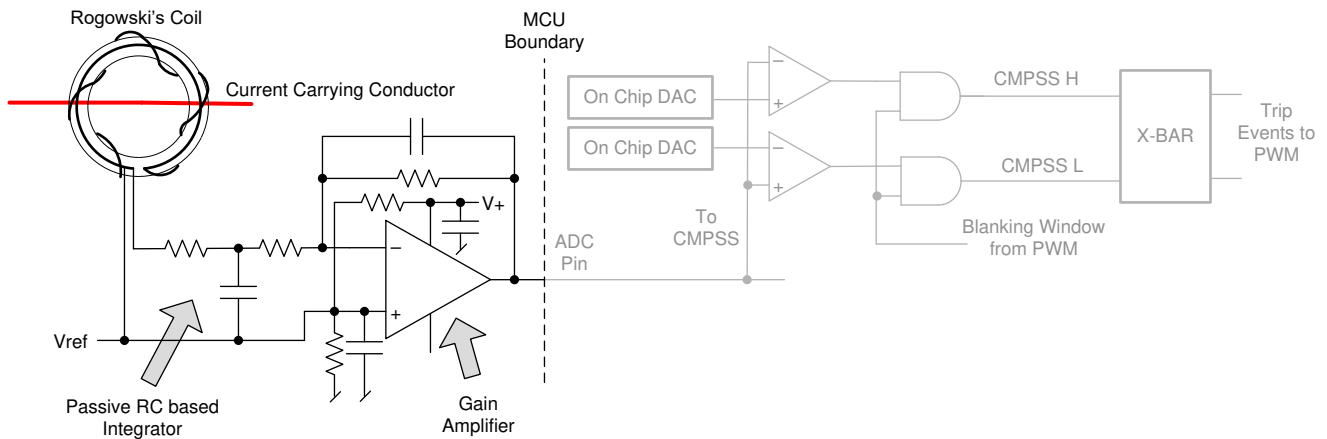


図 2-11. ロゴスキー コイルを使用した ISEC タンクの電流センシング

2.2.2.5 IPRIM 電流センシング

1 次側電流 IPRIM は、LMV796-Q1 を使用してセンシングされます (図 1-1 を参照)。

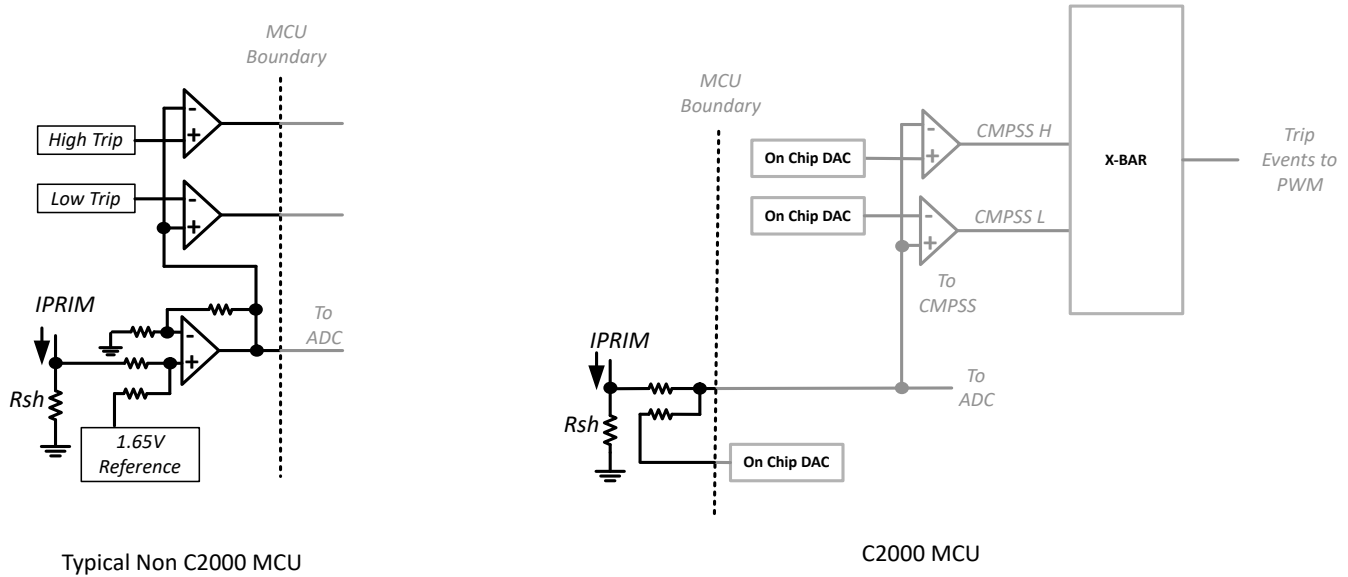


図 2-12. IPRIM 電流センシング回路、代表的なマイクロコントローラと C2000 マイクロコントローラとの比較

2.2.2.6 保護 (CMPSS および X-Bar)

大半のパワー エレクトロニクス コンバータは、過電流イベントから保護する必要があります。このデザインでは、複数のコンパレータが必要であり、トリップ ポイントの基準を生成する必要があります。コンパレータ サブシステム (CMPSS) の一部として、トリップ設定ポイント用の 12 ビット DAC とともにオンチップのウィンドウ付きコンパレータを搭載した、TMS320F280039 などの C2000 マイクロコントローラを PWM モジュールに内部接続することで、外部ハードウェアを使用することなく、PWM の高速トリップが可能になります。これにより、DAC、コンパレータ、ADC などのオンチップ リソースを使用することで追加部品が不要になるため、最終アプリケーションで基板面積とコストが削減できます。これらのリソースはすべて、外部接続を追加することなく、同時使用が可能です。さらに、CMPSS で生成された信号は X-Bar に送られ、さまざまな独自の方法で組み合わされて、複数のソースからの特定のトリップ イベントにフラグを付与することができます。

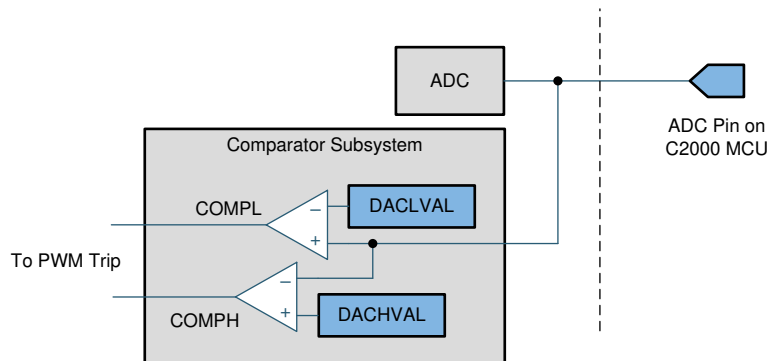


図 2-13. 過電流保護用のコンパレータ サブシステム (CMPSS)

2.2.3 PWM 変調

図 1-1 に、このデザインで使用する PWM の波形構成を示します。

高分解能 PWM は 1 次側レグと 2 次側レグに使用され、PWM の生成にはアップダウン カウント モードが使用されます。高分解能 PWM を使用する場合、PRIM_LEG1_H PWM パルスは周期イベントの中心に配置され、タイム ベースはアップダウン カウントに構成されます。その後、相補型スイッチ用に、高分解能デッドタイムを持つ相補型パルスが生成されます。LEG1 と LEG2 の間には、フルブリッジ動作のために 180 度の位相シフトがあります。これは、PWM モジュールの機能を使用して xA 出力と xB 出力を入れ替えることで実現します。(別の方法として、位相シフトを実装することもできますが、このデザインでは必要ありません)。

2 次側への PWM パルスはアイソレータを通るため、さらに伝搬遅延が生じます。この伝搬遅延を考慮して、PWM を少し進める必要があります。これは、1 次側アクティブ PWM パルスの立ち下がりエッジに対する位相シフト遅延によって行われます。2 次側の位相シフトは、図 1-1 に示すように、アイソレータに必要な周期と遅延の組み合わせによるものです。アクティブ同期整流方式を使用しているため、立ち上がりエッジは 1 次側 PWM スwitchのタイミングによって制御されます。スイッチング イベントはノイズが多い可能性があるため、ブランキング ウィンドウを使用します。2 次側タンクの電流は、動作周波数や負荷に応じて不連続になることがあります。そのため、立ち下がりエッジは、2 次側電流がゼロになった直後にトリガされるトリップ動作によって制御されます。その後、次のゼロ イベントまたは周期イベントが発生するまでトリップがラッチされ、ノイズによる 2 次側スイッチの意図しないオン動作を回避します。ブランキング パルスは PWM タイム ベースによって生成されますが、トリップ ラッチ動作やブランキング動作は CMPSS の一部として行われます。タンク電流の正の半分であるか負の半分であるかに応じて、2 つの異なるトリップ信号が生成され、X-Bar 経由で PWM モジュールに送信されます。C2000 マイクロコントローラのタイプ 4 PWM は、これらのイベントを独自に使用して、アップ カウント中に xA パルスを、ダウン カウント中に xB パルスをトリップさせることができます。詳細については、セクション 5.1.2 を参照し、ソリューションの HAL ファイルである関数 CLLLC_HAL_setupSynchronousRectificationAction() 内のコードを参照してください。

タイプ 4 PWM のグローバルリンク メカニズムは、レジスタの更新に必要なサイクル数を減らし、高周波動作を可能にするために使用されます。たとえば、関数 CLLLC_HAL_setupPWM() 内の以下のコードは、すべての PWM レグの TBPRD レジスタをリンクしています。このリンクを使用すると、PRIM_LEG1 TBPRD レジスタへの書き込みを 1 回行うだけで、PRIM_LEF2、SEC_LEG1、SEC_LEG2 に値が書き込まれます。

```
EPWM_setupEPWMLinks(CLLLC_PRIM_LEG2_PWM_BASE,
                    EPWM_LINK_WITH_EPWM_1,
                    EPWM_LINK_TBPRD);
```

```
EPWM_setupEPWMLinks(CLLLC_SEC_LEG1_PWM_BASE,
                    EPWM_LINK_WITH_EPWM_1,
                    EPWM_LINK_TBPRD);
```

```
EPWM_setupEPWMLinks(CLLLC_SEC_LEG2_PWM_BASE,
                    EPWM_LINK_WITH_EPWM_1,
                    EPWM_LINK_TBPRD);
```

高分解能 PWM は前のサイクルでの残りの計算結果を次のサイクルに持ち越すため、位相関係を維持するために 1 次側 PWM と 2 次側 PWM の間で周期的な同期を使用することはできません。周波数の変化やデューティの変化が検出されるたびに、高速割り込みサービス ルーチンを使用して、ワンタイム同期が行われます (ISR1、セクション 5.1.2.2 を参照)。

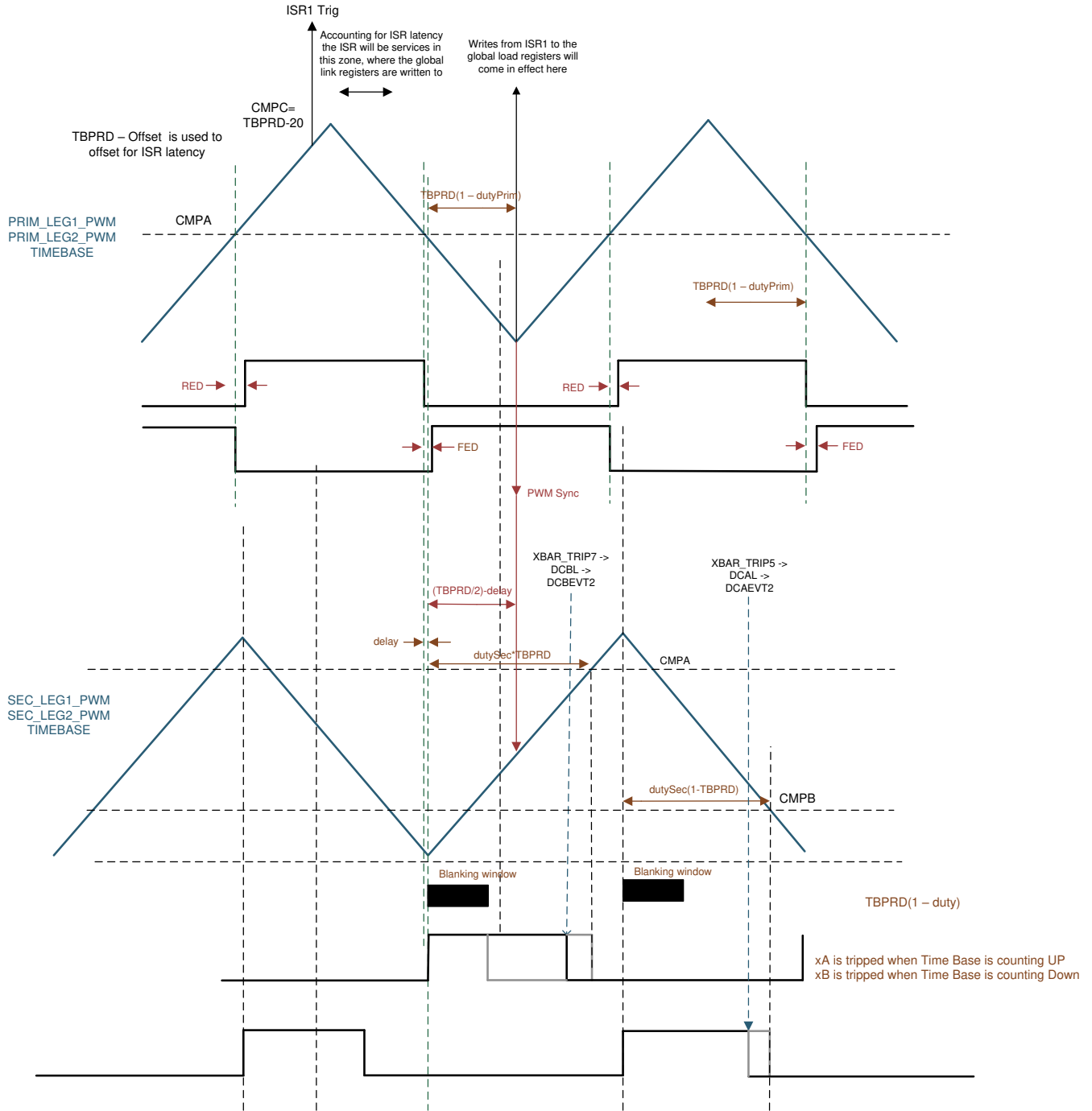


図 2-14. CLLLC 設計に使用される PWM 方式、1 次側から 2 次側への電力フローによるアクティブ同期整流の場合

逆方向の電力フローについても同様に、使用されている PWM 構成を 図 1-1 に示します。

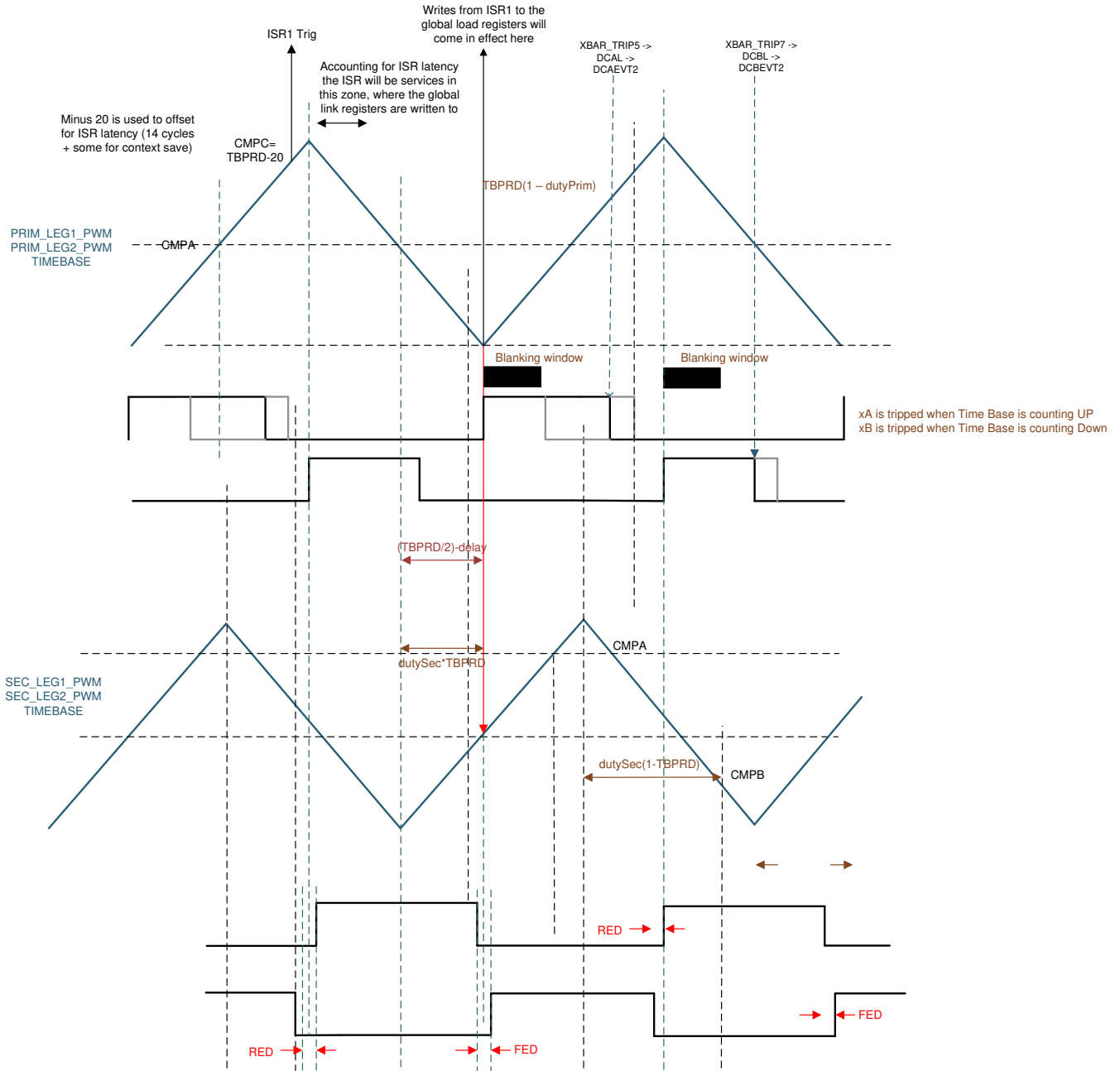




図 2-15. CLLLC 設計に使用される PWM 方式、2 次側から 1 次側への電力フローによるアクティブ同期整流の場合

3 トータムポール PFC システムの説明


警告



デバイスの電源を入れたままその場を離れないでください。



高電圧！ 基板上は高電圧状態になっており、接触するおそれがあります。感電する可能性があります。基板は、不適切に取り扱った場合に感電、火災、けがの原因となる電圧および電流で動作します。負傷や物品の破損を避けるために、必要な注意と適切な対策をもって機器を使用してください。安全のため、過電圧および過電流対応の絶縁された機器の使用を強くお勧めします。テキサス・インスツルメンツは、基板への電源投入やシミュレーション実行の前に、電圧要件および絶縁要件を確認し理解することがユーザーの責任であると考えます。電源投入中は、回路およびその接続部品には触れないでください。



表面は高温！ 触れるとやけどの原因になることがあります。触れないでください！基板の電源を入れると、一部の部品は **55°C** を超える高温に達することがあります。動作中は常に、また動作直後も高温の状態が続く可能性があるため、基板に触れてはいけません。

3.1 トータムポール ブリッジレス PFC の利点

すべてのプラグイン ハイブリッド電気自動車 (PHEV) では、車両内の電力グリッドと高電圧バッテリー パックの間にオンボードチャージャ (OBC) が必要です。AC/DC 電力変換を行うために電力グリッドに直接接続し、ダウンストリームの DC/DC コンバータに流れる実際の電力を最大化するには、力率改善 (PFC) コンバータの実装が必須になります。

従来型の PFC コンバータには整流用に、現在ではパッシブ PFC 技術として知られているパッシブ ダイオードブリッジが実装されています。この方式の利点として、設計が簡単、信頼性が高い、低速のシステム制御ループ、低コストであることが挙げられます。ただし欠点も明らかで、パッシブ部品は重量が重く、力率も低く、電力損失が大きいと、ヒートシンクがかさばり、放熱が多くなります。この問題をさらに調査すると、幅広い主電源アプリケーションの低電圧ラインでは、入力ブリッジが入力電力の約 2% を消費することが判明しました。設計者が直列ダイオードのいずれかを抑制できれば、入力電力を 1% 節約でき、その結果として効率を 94% から 95% に高めることができます (Turchi, Dalal, Wang, Lenck, 2014 年)。前述の欠点があるため、ブリッジ接続の従来型 PFC の定格電力は数百ワット未満に制限されています。特に、スペースと重量の削減が重要な設計パラメータであるハイブリッド電気自動車 (HEV) や電気自動車 (EV) ではそれが顕著です。

その結果、従来のダイオードブリッジをなくした、ブリッジレスアーキテクチャへと移行し続けています。OBC はシリコンパワー デバイスをベースとしており、低効率、低電力密度、高重量などの制限があります。SiC MOSFET の利点により、設計者は、高速スイッチング、低逆方向回復電荷、低 $R_{DS(ON)}$ という優れた性能を活用することで、これらの制限を大幅に改善することができます。

図 1-1 に、トータムポール ブリッジレス PFC 昇圧整流器の基本構造を示します。1 つの昇圧インダクタ、2 つの高周波昇圧 GaN スイッチまたは SiC スイッチ (下図の SiC_1 と SiC_2)、およびライン周波数で電流を流すための 2 つの部品で構成されています。図 1-1 に示すように、ライン周波数の関連部品には 2 つの低速ダイオードを使用しても問題ありません。(A) に 2 つのシリコン ダイオード (D_1 と D_2) を示しています。(B) は、 Si_1 と Si_2 を使用することで効率がさらに向上することを示しています。

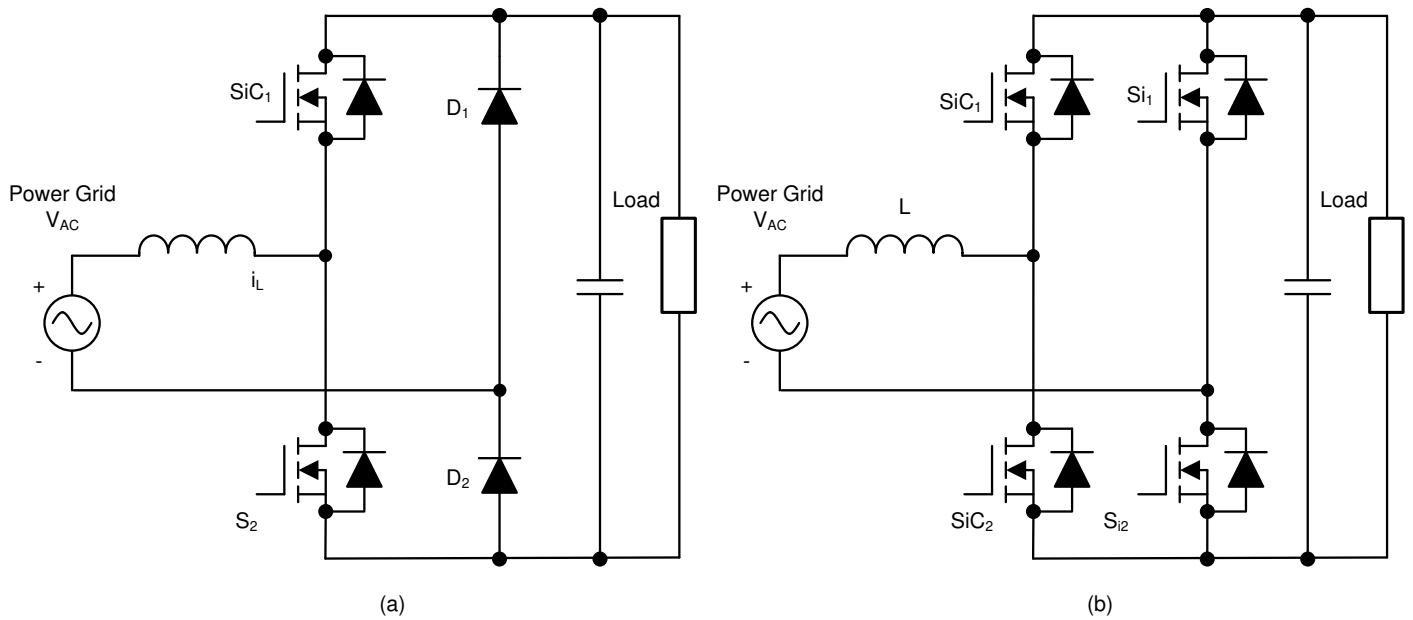


図 3-1. トータムポールブリッジレス PFC 昇圧コンバータのトポロジ: (A) ライン整流用ダイオード、(B) ライン整流用 MOSFET

トータムポール PFC に固有の問題は、AC 電圧ゼロクロス時の動作モードの遷移です。AC 入力ゼロクロス時に正のハーフラインから負のハーフラインになると、ローサイド高周波スイッチ SiC_2 のデューティ比が 100% から 0% に変化し、 SiC_1 のデューティサイクルが 0% から 100% に変化します。ハイサイドダイオード (または MOSFET のボディダイオード) の逆回復が遅いため、 D_2 のカソードの電圧はグランドから $DC+$ 電圧に瞬時にジャンプすることはできません (これにより大きな電流スパイクが発生します)。この問題が原因で、設計者は Si MOSFET を連続導通モード (CCM) のトータムポール PFC に使用することはできません。したがって、 SiC_1 と SiC_2 は、逆回復性能が低い窒化ガリウム (GaN) または SiC MOSFET の電界効果トランジスタ (FET) でなければなりません。TIDM-02013 では GaN FET を選択しました。

トータムポール PFC の最大の利点は、導通経路での電力損失が低減されることです。表 3-1 に、従来型 PFC とトータムポール PFC のデバイスの比較を示します。

表 3-1. ブリッジ接続の従来型 PFC とトータムポールブリッジレス PFC のデバイスの比較

パラメータ	低周波ダイオード	高周波ダイオード	高周波スイッチ	導通経路
ブリッジ接続の従来型 PFC	4	1	1	低速ダイオード 2 個 + スイッチ 1 個または (低速ダイオード 2 個 + 高速ダイオード 1 個)
トータムポールブリッジレス PFC	2	ゼロ	2	高速 GaN スイッチ 1 個 + 低速 Si (または SiC) MOSFET 1 個

トータムポール PFC の利点については、以下のとおりです。

- 従来型の PFC 昇圧コンバータは最も一般的なトポロジですが、その効率はフロントエンドのダイオードブリッジ整流器の導通損失の影響を受け、双方向ではありません。トータムポール PFC は本来、双方向動作ができるようになっています。
- ブリッジレス PFC 昇圧コンバータは、ダイオード数を大幅に削減し、電力密度を高め、効率を向上させています。
- この PFC は、高い効率、小さい同相モードノイズ、小さい AC 電流リップル、小さい逆方向回復電流、少ない部品点数という点で優れています。
- GaN ボディダイオードの低逆回復電荷、GaN FET の低ターンオン抵抗によって、このコンバータは双方向オンボードチャージャ向けの効率的でコスト効率の優れたソリューションとなります。

3.2 トータムポールブリッジレス PFC の動作

トータムポール PFC は、AC 主電源入力の正サイクルと負サイクルでそれぞれ動作し、高周波 GaN MOSFET がどのようにスイッチングされるかに応じて電流フローを決定します (それぞれ [図 1-1](#) と [図 1-1](#) を参照)。

高周波 GaN MOSFET とインダクタを組み合わせ、同期モード昇圧コンバータを構成します。正の半サイクル中、 S_2 はデューティサイクル D で駆動される昇圧スイッチで、 S_1 は相補型パルス幅変調 (PWM) 信号 $(1-D)$ で駆動されます。[図 1-1 \(A\)](#) は電流が流れる方向を示しています。同様に、 S_2 が $1-D$ でスイッチングされている間、 S_1 は D でスイッチングされます。[図 1-1 \(B\)](#) は電流が流れる方向を示しています。このサイクル中、 S_{D2} は連続的に導通しています。

負の半サイクル中は、ハイサイドとローサイドの高周波スイッチの役割が入れ替わることを除けば、動作はほぼ同じです。[図 1-1](#) は電流が流れる方向を示しています。このサイクル中、 S_{D1} は連続的に導通しています。

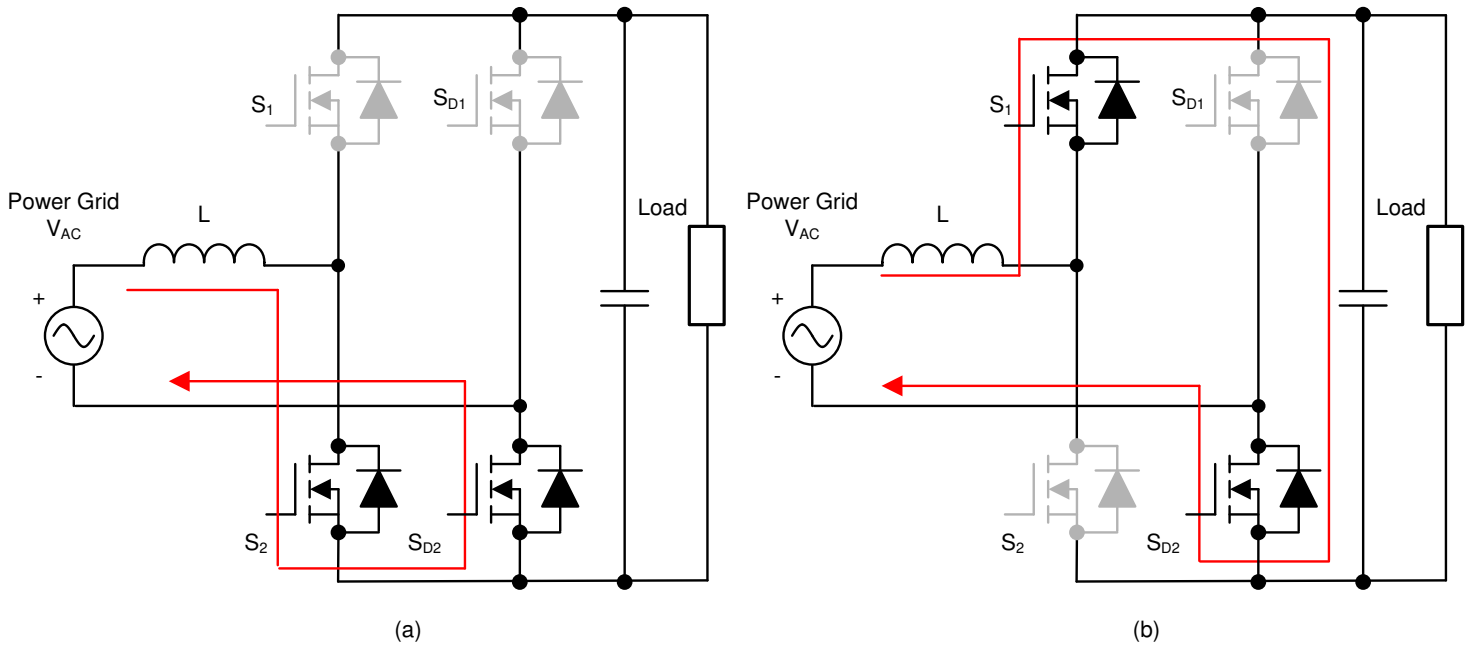


図 3-2. 正のハーフサイクル中のトータムポールブリッジレス PFC の動作: (A) S_2 ON 時、(B) S_2 OFF 時

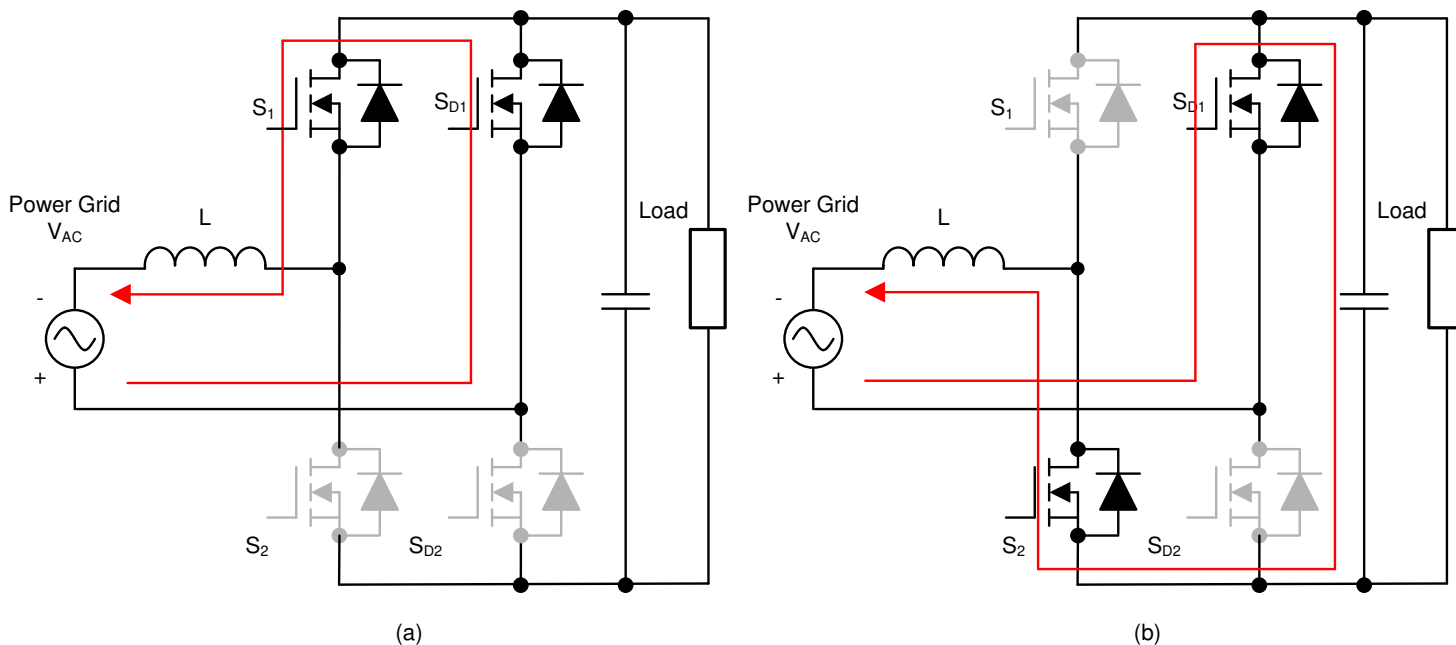


図 3-3. 負のハーフサイクル中のトータムポールブリッジレス PFC の動作: (A) S_1 ON 時、(B) S_1 OFF 時

このリファレンス デザインは、GaN FET (LMG3522R030-Q1) とテキサス・インスツルメンツの C2000™ Piccolo™ (TMS320F280039C) 高性能マイクロコントローラを使用しています。高周波 GaN FET は 120kHz のスイッチング周波数で動作し、1 対の Si MOSFET はライン周波数 (約 45Hz~60Hz) で動作します。したがって、導通経路には 1 つの GaN スイッチと 1 つの低周波 Si スイッチがあり、導通損失が大幅に低減されています。導通損失と入力電流リップルを低減するために 2 チャンネル インターリーブが使用されており、テスト結果では 98.5% を上回る高効率を実証しています。

3.3 主なシステム仕様

このデザインの主なシステム仕様を [表 3-2](#) に示します。

表 3-2. TIDA-02013 PFC の主なシステム仕様

パラメータ	仕様
入力	<ul style="list-style-type: none"> • 単相 • 電圧: 約 90V AC_{RMS} ~ 264V AC_{RMS} • AC ライン周波数範囲: 50Hz ~ 60 Hz • 入力電流: 240V で 32A_{RMS_MAX}、120V で 32A_{RMS_MAX} • 力率: ≥ 0.99
出力	<ul style="list-style-type: none"> • PFC 出力: 約 400V (代表値) • 最大出力電力: 約 400V で 7.4kW • ピーク効率: 98.5%
性能	<ul style="list-style-type: none"> • 高電圧リチウムイオン バッテリ OBC 用 PFC 段 • スイッチング周波数: 120 kHz • 絶縁: 強化型 • 入力 AC センシング • PFC 出力電圧センシング
保護	<ul style="list-style-type: none"> • 過熱保護 • 短絡保護 • 過電流保護 • 低電圧保護 • 過電圧保護

3.4 システム概要

3.4.1 ブロック図

図 1-1 に、TIDM-02013 リファレンス デザインのシステム ブロック図を示します。このリファレンス デザインには、以下の要素が含まれています。

- パワー スイッチ G1～G4 は高周波 GaN MOSFET で、各ハーフブリッジレッグ間に 180° の位相シフトがあります。G5 と G6 は低周波数 (40Hz～60Hz) の同期整流ブリッジを形成し、実質的にはスイッチング損失がありません。この 2 つのデバイスには低導通損失の特性が求められます。
- TMS320F280039C C2000 リアルタイム マイクロコントローラがコントローラとして機能し、すべての電圧と電流のセンサ入力を有し、G1～G6 用の正しい PWM 信号を生成します。また、コントローラはゲートドライバ ボードからフォルト信号を読み取り、フォルトが発生した場合にシステムをシャットダウンします。リセット機能は、起動時またはフォルトがクリアされたときに使用されます。
- ホール センサは、合計入力電流と各チャネルの電流のセンシングに使用されます。分圧器は、入力ライン電圧、ニュートラル電圧、および出力 DC バス電圧のセンシングに使用されます。

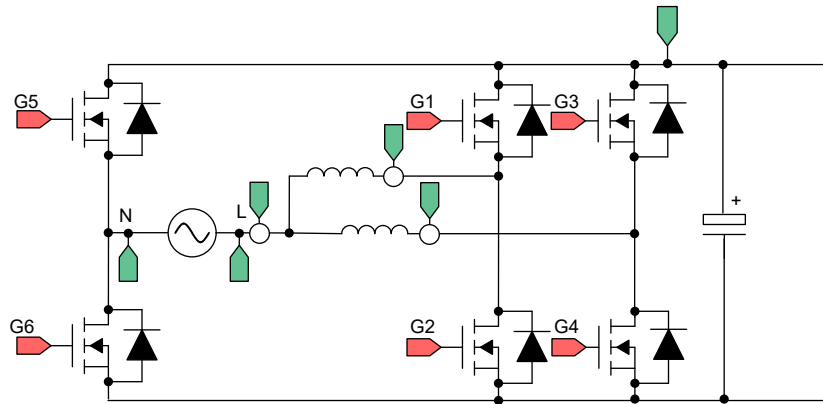


図 3-4. TIDM-02013 PFC のブロック図

3.5 システム設計理論

3.5.1 PWM

図 1-1 にインターリーブ TTPL PFC 方式の单相概略図を示します。この整流器を制御するには、デューティ サイクルを制御することによって、電圧を直接制御します。この電圧制御は、ソフトウェア変数 Duty または D が 1 と等しいとき Q3 が常時オンになるように設定した場合に可能であり、この設定により電圧 V_{xiN} は電圧 V_{bus} と等しくなります。Duty を 0 に設定すると、Q3 はオンになりません。Q4 は常時 DC バスの負側に接続され、電圧が 0 になります。

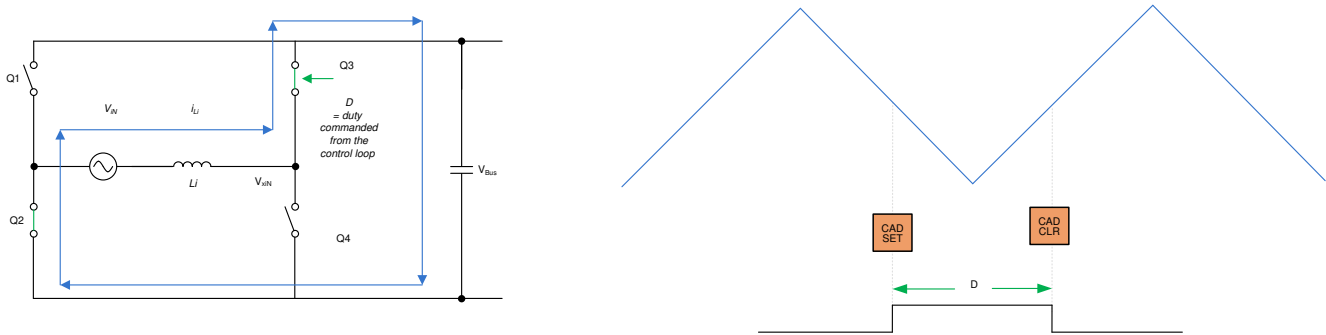


図 3-5. TTPL PFC の单相図

3.5.2 電流ループモデル

電流ループモデルを理解するには、まずインダクタ電流に注目します。図 1-1 では、スイッチ Q3 および Q4 に接続された PWM 変調器にデューティ サイクル (D) が設定されます。ここから、式 6 を次のように表します。

$$V_{xiN} = D \times V_{bus} \tag{6}$$

注

D を 1 に設定すると、Q3 は常時オンになり、D を 0 に設定すると、Q3 は常時オフになります。

インダクタを流れる電流を変調するには、Q3 スイッチ と Q4 スイッチのデューティ サイクル制御を用いて電圧 V_{xiN} を制御します。電流の方向は AC ラインから整流器に流れ込む方向で正となり、DC バス フィードフォワードおよび AC 電圧フィードフォワードの使用時にはグリッドがかなり強力になると想定されます。図 1-1 に電流ループの概略図を示し、電流ループプラントモデルを式 7 のように表します。

$$H_{p_i} = \frac{i_{Li}^*}{D} = \frac{1}{K_v_gain} \times K_i_gain \times G_d \times \frac{1}{Z_i} \tag{7}$$

ここで、

- K_v_gain は、センシングされた最大バス電圧の逆数、 $\frac{1}{V_{busMaxSense}}$ です。
- K_i_gain は、センシングされた最大 AC 電流の逆数、 $\frac{1}{I_{AC_MaxSense}}$ です。
- K_i_fltr は、電流センサから ADC ピンに接続された RC フィルタの応答です。
- G_d は、PWM 更新に伴うデジタル遅延であり、デジタル制御は電流指令です。
- i_{Li}^* は電流指令です。

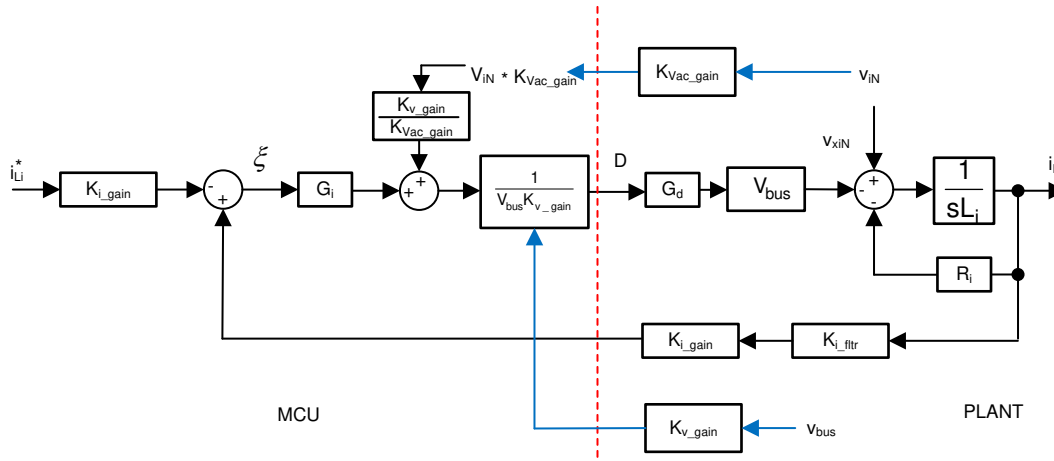


図 3-6. 電流ループ制御モデル

注

基準の負記号は、電流ループが電圧 V_{xiN} を制御していると考えられるためです。電流を上げるには、 V_{xiN} を下げる必要があります。このため、図 1-1 の基準と帰還では記号が逆になっています。

次に、この電流ループモデルを用いて電流補償器を設計します。電流ループには、単純な比例積分(PI)コントローラを使用します。

2 相インターリーブの場合は、各レッグに同じデューティ サイクルが設定されるため、電流は単純に 2 倍になります。このため、プラントモデルは式 8 のようになります。

$$H_{p_i} = \frac{i_{Li}^*}{D} = 2 \times \frac{1}{K_v_gain} \times K_i_gain \times K_i_fltr \times G_d \times \frac{1}{Z_i} \tag{8}$$

3.5.3 DC バス電圧制御ループ

DC バス電圧制御ループは、基準電力を提供するものとされています。基準電力を入力電圧 RMS の 2 乗で割って導電率を出し、さらに入力電圧を乗じて瞬時電流指令を求めます。

DC バス電圧制御ループの小信号モデルは、動作点周りで式 9 を線形化して作成します。

$$\hat{i}_{DC} V_{bus} = \eta V_{Nrms} \hat{i}_{Nrms} \rightarrow \hat{i}_{DC} = \eta \frac{\bar{V}_{Nrms}}{V_{bus}} \hat{i}_{Li} \quad (9)$$

抵抗性負荷の場合、バス電圧と電流は式 10 に示すような関係にあります。

$$\hat{V}_{bus} = \frac{R_L}{1 + sR_L C_o} \hat{i}_{DC} \quad (10)$$

DC 電圧ループ制御モデルは、図 1-1 に示すように描写できます。制御ループがバス電圧に依存しないようにするために、追加の V_{bus} フィードフォワードが適用されます。したがって、バス制御のプラントモデルは式 11 のように記述できます。

$$H_{p_bus} = H_{load} * \eta * \frac{1}{K_{i_gain}} * K_{v_gain} * K_{v_ftr} * \left(\frac{K_{v_gain}}{K_{vac_gain}} \right) \quad (11)$$

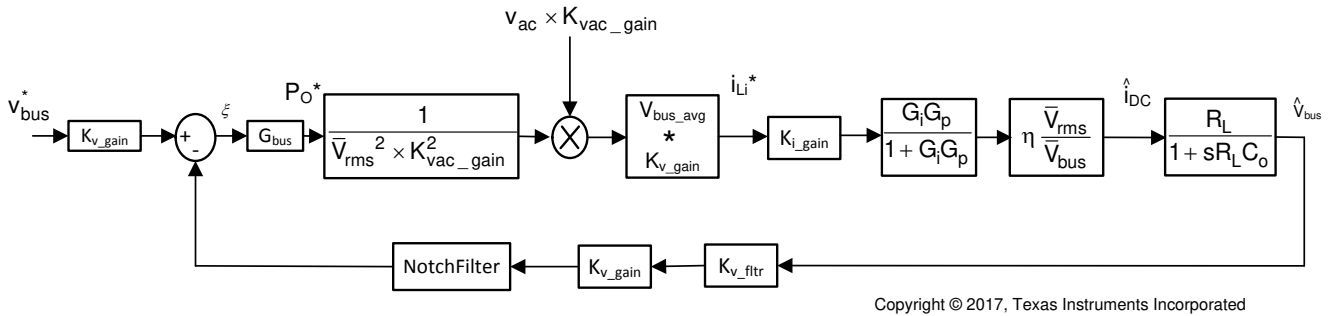


図 3-7. DC 電圧ループ制御モデル

図 1-1 を用いて、電圧ループ用に比例積分(PI)補償器を設計します。このループの帯域幅は、定常状態で THD と衝突するため、狭く維持します。

3.5.4 電流スパイクを除去または低減するゼロクロス付近のソフトスタート

ゼロクロス電流スパイクは、TTPL PFC トポロジにおける一つの課題です。これは、ステート マシンによるソフトスタートの手法を実装して、一定のシーケンスでスイッチのオン/オフを切り替えることで解決されます。

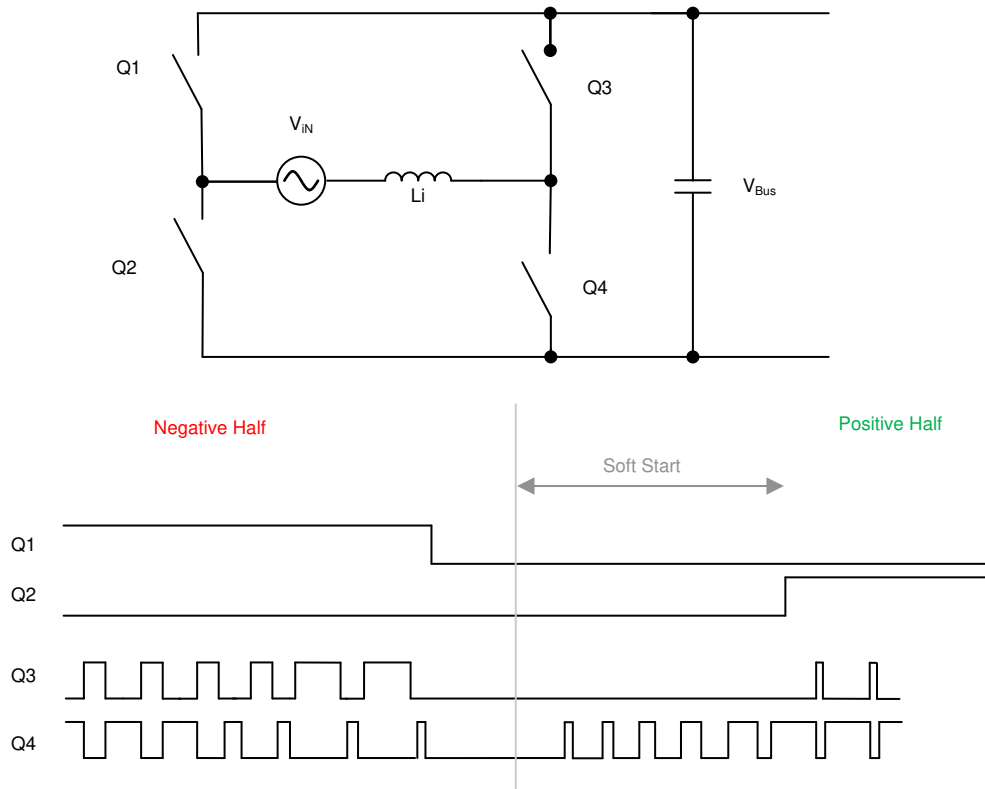


図 3-8. ソフトスタートによる PWM シーケンスでゼロクロス時の電流スパイクを低減する

図 1-1 は、AC 波が負から正に移行するときのスイッチング シーケンスを示しています。負の半サイクル中は、Q1 がオン、Q3 がアクティブ FET、Q4 が同期 FET となります。この間、Q2 を通る電圧は DC バス電圧となります。AC サイクルが変わると、Q2 は 100% またはほぼ 100% オンになる必要があります。Q2 がすぐにオンになると、非常に大きな正のスパイクが生じます。このため、図 1-1 に示すようにソフトスタート シーケンスを用いて Q4 をオンにします。このソフトスタートの調整は、インダクタンス値やその他の電力段パラメータ (デバイスの C_{oss} など) に依存します。

ゼロクロス付近で負の電流スパイクが生じるもう一つの理由は、ゼロクロス付近の AC 電圧が比較的低いことです。Q3 がオンになると、デューティ サイクルが低くても、高電圧差となり、高い負の電流スパイクが生じる可能性があります。このため、Q3 が再びスイッチング動作を開始する前に、十分な遅延を要します。

また、ソフトスタートの開始後に、いくらかの遅延ののち Q2 がオンになります。

3.5.5 電流の計算

式 12 で計算される最大入力電流に基づいて、入力ヒューズ、フィルタ電流定格を選択してください。

$$I_{inrms} = \frac{P_{out_max}}{\eta \cdot V_{inrms} \cdot PF} = 28.2 \text{ A} \quad (12)$$

ここで、

- P_{OUT_MAX} は最大出力電力 6.6kW です。
- η は効率です (98.6% と想定)。
- V_{IN_RMS} は入力電圧 RMS の値 (240V) です。
- PF は力率です (0.99 と想定)。

3.5.6 インダクタの計算

インダクタは、システム効率、電流リップル、全体のサイズに影響を及ぼす重要な役割を果たします。効率と電力密度のバランスを常に考慮する必要があります。インダクタンスの値は、入力電圧、出力電圧、ワースト ケースのリップルに基づいて計算します。

デューティサイクルは次のように計算されます。

$$D = 1 - \frac{V_{in}}{V_{out}} \quad (13)$$

インダクタへの電流リップルの計算は、次の 3 つの期間に区別できます。

$$I_{ripple} = \left(\frac{V_{in}}{L} - 2 \times \frac{V_{out} - V_{in}}{L} \right) \times D \times T_s \leftarrow \text{For } D \leq 1/3 \quad (14)$$

$$I_{ripple} = \left(\frac{2 \times V_{in}}{L} - \frac{V_{out} - V_{in}}{L} \right) \times \left(D - \frac{1}{3} \right) \times T_s \leftarrow \text{For } 1/3 < D < 2/3 \quad (15)$$

$$I_{ripple} = \left(\frac{3 \times V_{in}}{L} \right) \times \left(D - \frac{2}{3} \right) \times T_s \leftarrow \text{For } D \geq 2/3 \quad (16)$$

ワーストケースでは、式は次のようになります。

$$I_{ripple} = \frac{V_{out} \times T_s}{12 \times L} \quad (17)$$

このデザインでは、最大入力電力と最大 AC 電流で 10% の電流リップルを達成することを目標にしています。

$$I_{ripple} < 10\% \times \frac{\sqrt{2} \times P_{out_max}}{V_{in_max} \times \eta} \quad (18)$$

ここで、

- P_{out_max} は最大出力電力です。
- η は効率です。
- V_{in_max} は最大入力電圧です。

その結果、インダクタンスは 12A RMS 電流で 126 μ H と計算されます。

3.5.7 出力コンデンサの計算

DC リンク コンデンサにおける入力のダブルライン周波数リップルに起因して、DC リンク コンデンサの容量は、式 19 で計算される出力電圧リップルに主に影響を受けます。

$$C_{out(min)} \geq \frac{P_{out}/V_{out}}{4 \cdot \pi \cdot f_{line_min} \cdot V_{ripple_max}} = 860\mu F \quad (19)$$

ここで、

- P_{OUT} は出力電力です。
- V_{OUT} は出力電圧です。
- f_{LINE_MIN} は最小ライン周波数です。
- V_{RIPPLE} は出力リップルです。

実際に使用するコンデンサは 1410 μ F (3 x 470 μ F) です。

3.5.8 電流および電圧センシング

ホール効果センサ ACS733KLATR-40AB-T は、[図 1-1](#) に示すように、合計入力電流のセンシングに使用されます。OPA320 ベースのアンプ回路は、センサの低出力電圧をより高いレベルに調整し、この電圧をコントローラの ADC ピンに送ります。ACS733KLATR-40AB-T デバイスは、インターリーブされた各位相の電流をセンシングして、位相電流バランスを可能にします。

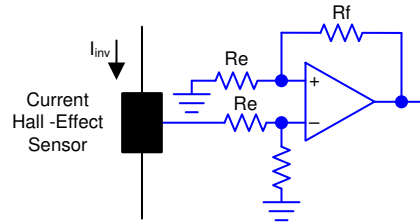


図 3-9. ホール効果センサのシグナル コンディショニング回路

シグナル コンディショニング回路からの出力電圧は、[図 1-1](#) に示すように、回路を使用して ADC 範囲と一致するようスケールリングされます。電圧は次のように計算されます。

$$I_{out} = \frac{R_f}{R_e} \left(I_{inv} \times \frac{V_{no\text{最小値}al}}{I_{no\text{最小値}al_max}} + V_{offset} \right) \quad (20)$$

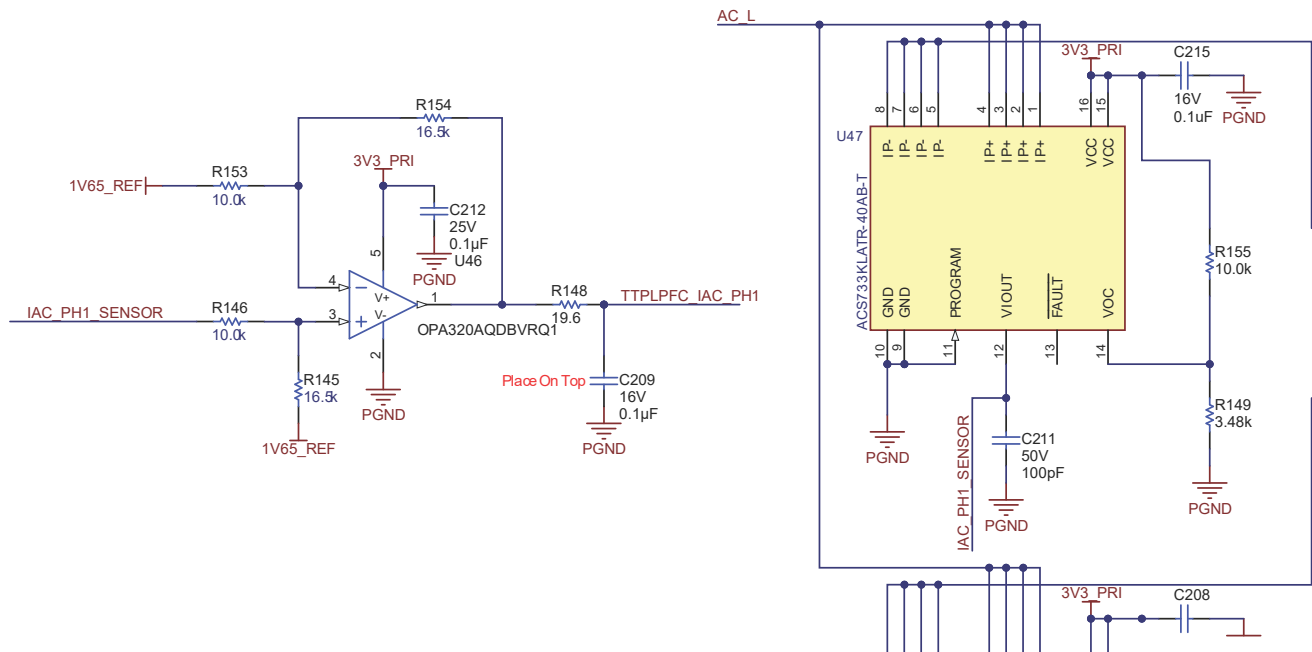


図 3-10. 電流センシングの回路図

図 1-1 に示すように、入力 AC 電圧はラインをセンシングすることで差動的にセンシングされ、ニュートラル入力は 2 つの分圧器で個別に制御グラウンドを参照します。制御グラウンドは DC リンクの負端子であるため、単一の分圧器を使用して DC バス電圧をセンシングできます。RC フィルタにより、コントローラに接続する前に信号がフィルタ処理されます。このデザインでは、すべてのセンシング信号に共通の RC フィルタが使用されています。

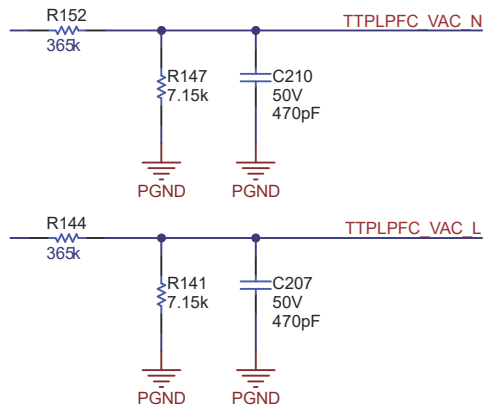


図 3-11. AC 入力電圧センシング用分圧器の回路図

4 主な使用製品

4.1 C2000 マイクロコントローラ TMS320F28003x

C2000™ 32 ビット マイクロコントローラは、処理、センシング、アクチュエーションに最適化されており、リアルタイム制御アプリケーション、たとえば産業用モータードライブ、ソーラー インバータおよびデジタル電源、電気自動車および輸送、モーター制御、センシングおよび信号処理などにおける閉ループ性能が向上しています。

TMS320F28003x (F28003x) は、重要な制御ペリフェラル、差別化されたアナログ、不揮発性メモリを 1 つのデバイスに組み込むことができる、強力な 32 ビット浮動小数点マイクロコントローラ ユニット (MCU) です。

CLA により、一般的なタスクの負荷の多くをメインの C28x CPU から取り除くことができます。CLA は独立の 32 ビット浮動小数点演算アクセラレータであり、CPU と並列に実行されます。さらに、CLA には独自の専用メモリリソースがあり、一般的な制御システムで必要となる主要なペリフェラルに直接アクセスできます。ANSI C のサブセット、およびハードウェアブレイクポイントやハードウェアによるタスク切り替えなどの主要な機能が標準でサポートされています。

F28003x マイクロコントローラには高性能のアナログ ブロックが内蔵されており、システムのさらなる統合が可能です。3 つの独立した 12 ビット ADC により、複数のアナログ信号を正確かつ効率的に管理でき、最終的にシステムのスループットが向上します。4 つのアナログ コンパレータ モジュールが、トリップ条件の有無を判断するために入力電圧レベルを継続的に監視します。

TMS320C2000™ デバイスには、周波数に依存しない拡張パルス幅変調器 / 高分解能パルス幅変調器 (ePWM/HRPWM) と拡張キャプチャ (eCAP) モジュールを備えた、業界をリードする制御ペリフェラルが搭載されており、クラス最高レベルのシステム制御が可能です。シグマ デルタ フィルタ モジュール (SDFM) が内蔵されているため、絶縁バリアを通して、オーバーサンプリング シグマ デルタ変調器をシームレスに統合できます。

さまざまな業界標準の通信ポート (シリアル ペリフェラル インターフェイス (SPI)、シリアル通信インターフェイス (SCI)、IC の相互接続 (I2C)、コントローラ エリア ネットワーク (CAN) など) により接続性がサポートされており、さまざまなアプリケーションにおいて最適な信号配置を行うための複数の多重化オプションを備えています。C2000™ プラットフォームの新機能として、完全準拠のパワー マネージメント バス (PMBus) が追加されました。さらに、業界で初めて高速シリアル インターフェイス (FSI) による高速かつ堅牢な通信が可能になり、本デバイスに組み込まれている一連の豊富なペリフェラルを補完します。

特別仕様の TMS320F28003xC では、構成可能ロジック ブロック (CLB) を利用して追加のインターフェイス機能を実現できます。詳細については、『TMS320F28003x マイクロコントローラ データ マニュアル』の「デバイスの比較」表を参照してください。

組み込みのリアルタイム分析および診断 (ERAD) モジュールにより、追加のハードウェア ブレイクポイントやプロファイリング用のカウンタを使用できるようになり、デバイスのデバッグおよびシステム分析機能が強化されます。

高周波 CLLLC トポロジの制御を可能にするために、このデザインで特に強調されている C2000 マイクロコントローラの機能の一部は、以下のとおりです。

- 高分解能 PWM:** ピコ秒単位の分解能を実現する C2000 マイクロコントローラの ePWM モジュールは、高周波 PWM を高精度に生成できます。タイプ 4 PWM の高分解能周期制御に加えて、高分解能デューティ制御、高分解能デッドバンド制御、高分解能位相シフト制御が可能です。これによって共振タンクの励起に適した平衡なパルスを生成できるようになり、高周波パワー コンバータを実現する機能と言えます。
- アクティブ同期整流用 ePWM 付きコンパレータ サブシステム (CMPSS):** アクティブ同期整流を使用すると、高い効率が実現されます。共振点よりも低い周波数および高い周波数の両方で動作する高周波共振コンバータの場合、トポロジに必要な機能となります。C2000 マイクロコントローラの内蔵 CMPSS では、内蔵コンパレータと内蔵 D/A コンバータ (DAC) を使用することで、アクティブ同期整流パルスを生成できます。(セクション 2.2.2.4 を参照。)
- ブランキング ウィンドウ:** スイッチング コンバータでは避けられないノイズのため、ブランキング ウィンドウ機能を使用して、ノイズの多いスイッチング イベント中の CMPSS 出力を抑制します。このブランキング ウィンドウは ePWM タイムベースによって提供され、PWM サイクルの異なるタイミングで適用できます。(セクション 2.2.3 を参照。)
- X-Bar:** 1 つのパワー コンバータで複数のトリップ ソースを使用することができます。X-bar により、異なる CMPSS または GPIO からの異なるトリップ動作を組み合わせることで、外部ロジックなしで ePWM に必要なトリップ動作を生成できます。

5. **制御補償器アクセラレータ (CLA)** により、複数のトポロジの制御を単一のコントローラに統合できます。このデザインで提供されるソフトウェアには、CLA または C28x で制御ループを実行するオプションがあります。
6. **PWM モジュールのグローバルリンク機能** により、1 回の書き込みで複数の PWM を更新できるため、CPU の負荷が軽減され、より周波数の高いコンバータを容易に制御できます。

4.2 LMG352xR30-Q1

LMG352xR30-Q1 は、ドライバ、保護機能、温度レポート機能を内蔵した、車載対応の 650V、30mΩ の GaN FET です。内蔵ドライバにより、最大 150V/ns のスイッチング速度を実現できます。テキサス・インスツルメンツの統合型高精度ゲートバイアスは、外部ディスクリートゲートドライバと比較して、より広いスイッチング SOA をもたらします。この統合と低インダクタンスパッケージの組み合わせにより、ハードスイッチング電源トポロジでノイズの少ないスイッチングとリンギングの最小化を実現できます。EMI を制御するための調整可能なゲートドライブ強度、過熱保護、フォルト表示付きの堅牢な過電流保護を含むその他の機能を使うと、BOM コスト、基板サイズ、フットプリントを最適化できます。高度な電源管理機能には、デジタル温度レポートが含まれます。GaN FET の温度は可変デューティサイクル PWM 出力により通知されるため、システムは負荷を最適に管理できます。

4.3 UCC21222-Q1

UCC21222 は、プログラミング可能なデッドタイムを備えた絶縁型デュアルチャネルゲートドライバです。ピーク電流はソース 4A、シンク 6A で、パワー MOSFET、IGBT、GaN トランジスタを駆動するように設計されています。

UCC21222 デバイスは、2 つのローサイドドライバ、2 つのハイサイドドライバ、または 1 つのハーフブリッジドライバとして構成可能です。5ns の遅延マッチング性能により、内部貫通電流のリスクを伴わずに、2 つの出力を並列化して 2 倍の駆動力で重負荷条件に対応できます。

入力側は 3.0kV_{RMS} の絶縁バリアによって 2 つの出力ドライバと分離され、同相過渡耐性(CMTI)は最小で 100V/ns です。

抵抗によるデッドタイムのプログラミングが可能のため、システムの制約に合わせてデッドタイムを調整することにより、効率を高め、出力のオーバーラップを防止できます。その他の保護機能:DIS を High に設定した場合に 2 つの出力を同時にシャットダウンするディセーブル機能、5ns 未満の入力過渡を除去する内蔵グリッチ除去フィルタ、入力 / 出力ピンでの 200ns にわたる最大 -2V のスパイクに対応する負電圧処理機能があります。すべての電源が UVLO 機能を備えています。

4.4 AMC3330-Q1

AMC3330 は、デバイスのローサイドから単一電源で動作できる完全統合型絶縁 DC/DC コンバータを備えた高精度絶縁型アンプです。その容量性強化絶縁バリアは、VDE V 0884-11 および UL1577 に準じて認定済みであり、異なる同相電圧レベルで動作するシステムの各部を分離し、低電圧部分を損傷から保護します。AMC3330 の入力、高電圧信号を検出するために、抵抗分圧ネットワークなどの高インピーダンス電圧信号ソースに直接接続するよう最適化されています。内蔵の絶縁 DC/DC コンバータにより非グランド基準信号を測定できるため、ノイズが多くスペースに制約があるアプリケーション向けの独自の設計として活用できます。

4.5 AMC3302-Q1

AMC3302 は、シャントを用いた電流測定に最適化された高精度絶縁型アンプです。完全に統合された絶縁型 DC/DC コンバータのおかげで、本デバイスの低電圧側から電力を供給する単一電源動作が可能であるため、スペースに制約があるアプリケーション向けのユニークなソリューションとして活用できます。その容量性強化絶縁バリアは、VDE V 0884-11 および UL1577 により認証済みであり、最大 1.2kV_{RMS} の使用電圧に対応しています。この絶縁バリアは、異なる同相電圧レベルで動作するシステム領域を分離し、危険な電圧と損傷から低電圧側を保護します。AMC3302 の入力、低インピーダンスのシャント抵抗またはその他の信号レベルが小さい低インピーダンス電圧源と直接接続できるように最適化されています。優れた DC 精度と小さい温度ドリフトにより、拡張産業用温度範囲 (-40°C ~ +125°C) にわたる高精度電流測定に対応できます。

5 ハードウェア、ソフトウェア、試験要件、試験結果

5.1 必要なハードウェアとソフトウェア

このセクションでは、ハードウェアの詳細について解説し、ボード上のさまざまなセクションと、この設計ガイドで説明しているように、実験用にそれらを設定する方法について説明します。

5.1.1 ハードウェアの設定

このデザインは高速エッジカード (HSEC) の制御カード コンセプトを採用しており、HSEC 制御カードを利用できる C2000 マイクロコントローラ製品ファミリのデバイスであれば使用できる可能性があります。マイクロコントローラの電力段の制御に使用される主要なリソースを [表 5-1](#) に示します。このリファレンス デザインの主要な電力段とコネクタを [図 1-1](#) に示します。[表 5-3](#) に主要なコネクタとその機能を示します。

1. 基板に電源が接続されていないことを確認します。
2. J25 スロットに制御カードを挿入します。
3. [図 1-1](#) に示す J15 に、12V バイアス電源 (+12V、2A) の電源を接続します (電源は投入しないでください)。
4. バイアス電源の電源をオンにします。制御カードの緑色の LED が点灯します。これは、C2000 マイクロコントローラ デバイスに電力が供給されていることを示します。注: マイクロコントローラの電源バイアスは電力段と分離されているため、この一連の指示でシステムを安全に立ち上げることができます。
5. JTAG を接続するには、制御カードから USB ケーブルを使用してホスト コンピュータに接続します。
6. TTPPLPFC 段を動作させるには、AC 入力を J33 (90V~264V) に接続する必要があります。テストには 10kW を超える電源が使用されていますが、低消費電力テストのみを実施する場合は、クリーンで安定した低定格電源を使用できます。
7. PFC 段のスタンドアロン動作の場合、負荷を J37 と J38 に接続することができますし、代わりに CLLLC を使用して PFC 段に負荷をかけることもできます。
8. CLLLC 段のスタンドアロン動作の場合、DC 電源 (400V) を J15 の VBUS に接続することができます。この場合、ソフトウェアで TTPPLPFC を起動せず、上記ステップ 6 で説明した AC 電源は接続しないでください。
9. 使用時は、CLLLC コンバータの 2 次側に負荷を接続する必要があります。このような負荷の接続には、J7 と J10 を使用できます。
10. PFC 段と DCDC 段の両方を動作させる場合は、上記ステップ 6 のように AC 電源を接続し、上記ステップ 9 のように負荷を接続します。VBUS への接続は必要はありませんが、OBC 実行後に余分な電圧を迅速に逃がすために、電流ブリード抵抗が役立つ場合があります。
11. 電流プローブと電圧プローブを接続して、1 次側と 2 次側のタンク電流を観測できます。オプションとして、電力計を接続して効率を測定することもできます。

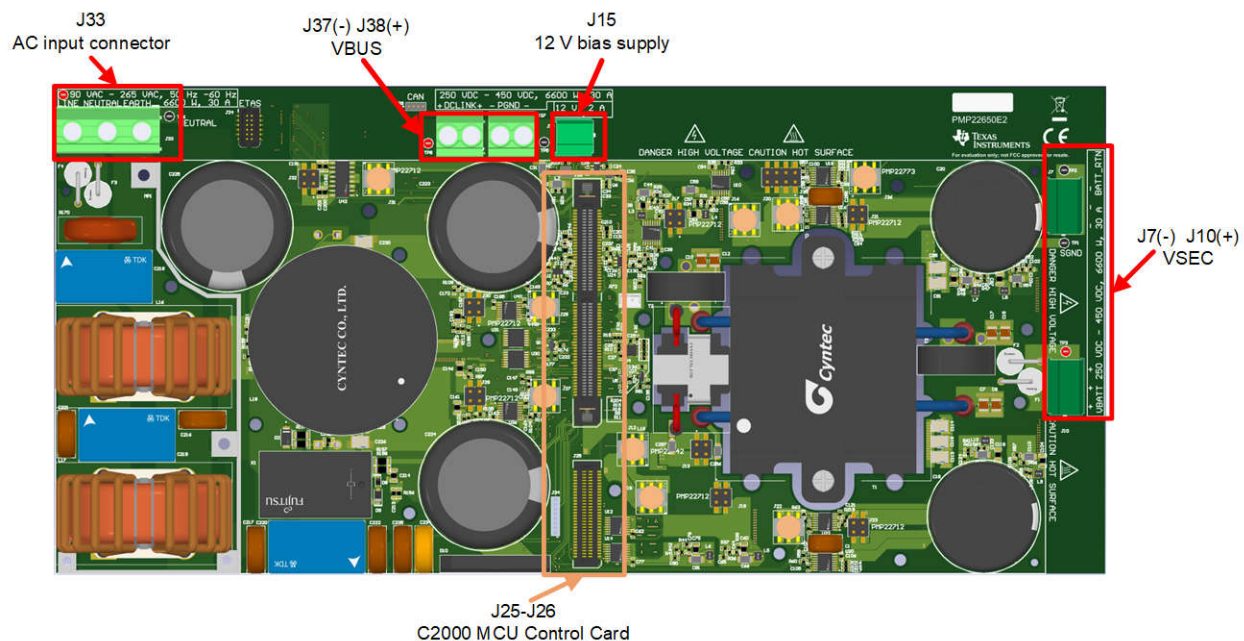


図 5-1. 基板の概要

赤色で示したバイアス電源ドーターカード 7 枚が必要です。

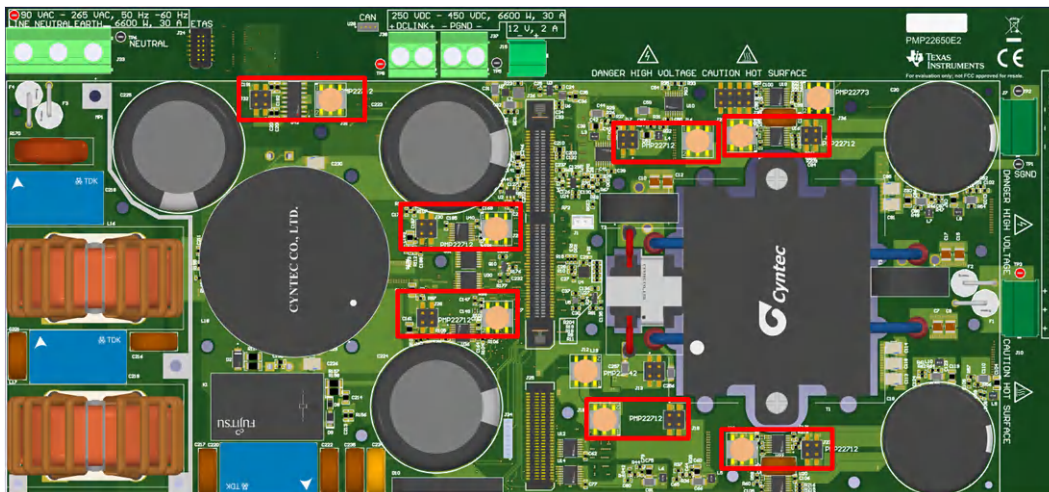


図 5-2. PMP22712 - バイアス電源

赤色で示したフィードバック絶縁ドーターカード PMP22773 1 枚が必要です。

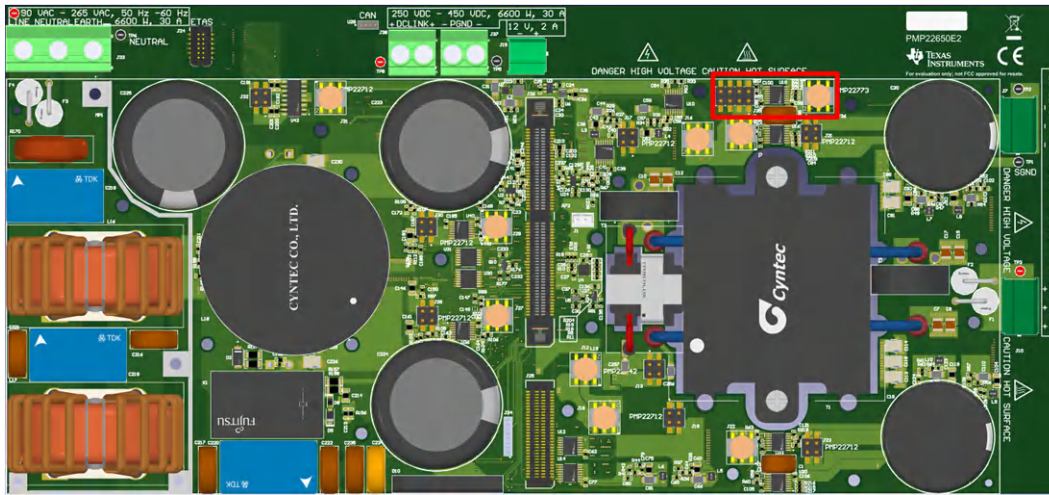


図 5-3. PMP22773 -フィードバック絶縁ドーター カード

表 5-1. 主なデジタルピン構成

信号名	HSEC ピン番号	F28003x パリフェラル
SYSTEM_ISR Trigger	-	ECAP1
CLLLC_CONTROL_OUTPUT_DAC_PIN	14	DACA
CLLLC_PRIM_LEG1_H/L	49/51	EPWM1 (A/B)
CLLLC_PRIM_LEG2_H/L	53/55	EPWM2 (A/B)
CLLLC_SEC_LEG1_H/L	50/52	EPWM3 (A/B)
CLLLC_SEC_LEG2_H/L	54/56	EPWM4 (A/B)
CLLLC_FAULTn	74	GPIO-23 → INPUTXBAR2
CLLLC_LC_CHANGE	62	GPIO-14
CLLLC_SEC_SIDE_DIAG	80	GPIO-30
TTPLPFC_LOW_FREQ_H/L	57/59	EPWM5 (A/B)
TTPLPFC_HIGH_FREQ_PH1_H/L	61/63	EPWM6 (A/B)
TTPLPFC_HIGH_FREQ_PH2_H/L	58/60	EPWM7 (A/B)
TTPLPFC_FAULTn	72	GPIO-22 → INPUTXBAR1
TTPLPFC_INRUSH_RELAY_CTRL	64	GPIO-15
ERRORSTSn	102	GPIO55
SYSTEM_WATCHDOG_OUT	75	GPIO24
SYSTEM_WATCHDOG_DISABLE	77	GPIO25 (抵抗オプション)
SYSTEM_PMIC_SPI (予約済み)	79	GPIO26 (抵抗オプション)
SYSTEM_PMIC_SPI (予約済み)	81	GPIO27 (抵抗オプション)
SYSTEM_DISABLE_FET_SUPPLY	85	GPIO32
SYSTEM_TEMP_MUX_OUT1	91	GPIO41 -> ECAP2 → INPUTXBAR3
SYSTEM_TEMP_MUX_OUT2	96	GPIO60 -> ECAP3 → INPUTXBAR4
SYSTEM_TEMP_MUX_SEL_1-3	93 94 95	GPIO47 GPIO58 GPIO59
SYSTEM_PROFILING1~3	89 92 101	GPIO40 GPIO44 GPIO49
FSI_TX_D0	101	GPIO-49/FSITXA_D0
FSI_TX_D1	103	GPIO-50/FSITXA_D1
FSI_TX_CLK	105	GPIO-51/FSITXA_CLK

表 5-1. 主なデジタルピン構成 (続き)

信号名	HSEC ピン番号	F28003x ペリフェラル
LED1	82	GPIO-31 → LED1
LED2	86	GPIO-34 → LED2 (SFRA)

この表はリファレンス デザインのサンプリング方法を示し、上部の列はそれぞれ 1 つの独立した ADC を表しています。各 ADC は互いに完全に独立して動作します。各信号には、1 つまたは複数の変換開始 (SOC) が割り当てられます。各 SOC はそのチャンネルの独立した 1 つの読み取りを表し、たとえば、TTPLPFC_IAC_PH1 は ADCA 内の SOC0 と SOC1 に割り当てられています。つまり、この信号はサイクルごとに 2 回サンプリングされ、1 回は ePWM6_SOC_A によってトリガされ、もう 1 回は ePWM6_SOC_B によってトリガされるのです。このトリガは 120kHz で動作しているため、この信号は各 120kHz のサンプリング期間中、実質的には 2 倍のオーバーサンプリングが行われます。同様に、CLLLC_ISEC では 11 倍のオーバーサンプリングが行われ、CLLLC_IPRIM はオーバーサンプリングされません。また、この表には、低周波サンプリング信号がいくつか示されており、これらの信号が異なる SOC 信号を使用していることがわかります。最後に、SOC を番号順に処理するためにラウンド ロビン カウンタが使用されているため、表はサンプリングの順に上から下へ時系列に読み取れます。

表 5-2. 主なアナログ信号

	ADC-A	ADC-B	ADC-C
最も優先度の高い信号 (120kHz)	TTPLPFC_IAC_PH1 (A2, CMPSS1) SOC0 → ADC_TRIGGER_EPWM6_SOC_A SOC1 → ADC_TRIGGER_EPWM6_SOC_B	TTPLPFC_IAC_PH2 (B12, CMPSS3) SOC0 → ADC_TRIGGER_EPWM6_SOC_A SOC1 → ADC_TRIGGER_EPWM6_SOC_B	TTPLPFC_VAC (C7) SOC0 → ADC_TRIGGER_EPWM6_SOC_A SOC1 → ADC_TRIGGER_EPWM6_SOC_B
	CLLLC_ISEC (A5, CMPSS2) SOC2 → ADC_TRIGGER_EPWM6_SOC_A SOC3 → ADC_TRIGGER_EPWM6_SOC_A SOC4 → ADC_TRIGGER_EPWM6_SOC_A SOC5 → ADC_TRIGGER_EPWM6_SOC_A SOC6 → ADC_TRIGGER_EPWM6_SOC_A SOC7 → ADC_TRIGGER_EPWM6_SOC_A SOC8 → ADC_TRIGGER_EPWM6_SOC_B SOC9 → ADC_TRIGGER_EPWM6_SOC_B SOC10 → ADC_TRIGGER_EPWM6_SOC_B SOC11 → ADC_TRIGGER_EPWM6_SOC_B SOC12 → ADC_TRIGGER_EPWM6_SOC_B	TTPLPFC_VBUS / CLLLC_VBUS (B4) SOC2 → ADC_TRIGGER_EPWM6_SOC_A SOC3 → ADC_TRIGGER_EPWM6_SOC_B SOC4 → ADC_TRIGGER_EPWM7_SOC_A SOC5 → ADC_TRIGGER_EPWM7_SOC_B	CLLLC_VSEC (C11, CMPSS2) SOC2 → ADC_TRIGGER_EPWM6_SOC_A SOC3 → ADC_TRIGGER_EPWM6_SOC_A SOC4 → ADC_TRIGGER_EPWM6_SOC_A SOC5 → ADC_TRIGGER_EPWM6_SOC_A SOC6 → ADC_TRIGGER_EPWM6_SOC_A SOC7 → ADC_TRIGGER_EPWM6_SOC_A SOC8 → ADC_TRIGGER_EPWM6_SOC_A SOC9 → ADC_TRIGGER_EPWM6_SOC_A SOC10 → ADC_TRIGGER_EPWM6_SOC_A SOC11 → ADC_TRIGGER_EPWM6_SOC_A SOC12 → ADC_TRIGGER_EPWM6_SOC_A
	CLLLC_IPRIM (A9, CMPSS2) SOC13 → ADC_TRIGGER_EPWM1_SOC_A		

表 5-2. 主なアナログ信号 (続き)

	ADC-A	ADC-B	ADC-C
低周波サンプリング信号 (10kHz)	TTPLPFC_VAC_L (A4) SOC14 → ADC_TRIGGER_CPU1_TINT2	TTPLPFC_VAC_N (B2) SOC10 → ADC_TRIGGER_CPU1_TINT2	TTPLPFC_VBUS2 (C10, CMPSS2) SOC14 → ADC_TRIGGER_CPU1_TINT2
	SYSTEM_TEMP_1 (A11) SOC15 → ADC_TRIGGER_CPU1_TINT2	SYSTEM_VREF_1_65 (B5) SOC11 → ADC_TRIGGER_CPU1_TINT2	CLLLC_VSEC (C11, CMPSS2) VSEC13 → SOC15 → ADC_TRIGGER_CPU1_TINT2
サンプリングなし、CMPSS のみ		CLLLC_IPRIM_TANK (A12/C5、 CMPSS2)	CLLLC_ISEC_TANK (C1, CMPSS4)

表 5-3. 主なコネクタと機能

コネクタ名	機能
J33	AC 入力
J37/J38	VBUS 接続、PFC 出力、DCDC VPRIM
J7/J10	DCDC 出力接続、DCDC VSEC
J15	12V、2A 電源
J25/J26	HSEC 制御カードのコネクタ スロット

5.1.1.1 制御カードの設定

デバイスコントロールカードの設定には、JTAG 経由の通信が必要であり、絶縁 UART ボードを使います。また、適切な ADC 基準電圧も提供する必要があります。以下は F280039C 制御カードのリビジョン A に必要な設定です。ユーザーは、C2000Ware (\c2000ware\boards\controlcards\TMDSCNCD280039C) にある情報シートを参照するか、または『F280039 controlCARD 情報ガイド』から情報を入手することもできます。

1. デバイスに対する JTAG 接続と SFRA GUI に対する UART 接続を確立するには、制御カードの S1:A を両端で ON (左) に設定する必要があります。このスイッチが OFF (右) になっていると、制御カードに内蔵された絶縁 JTAG を使用したり、SFRA GUI でデバイスと通信したりすることはできません。
2. J1:A は、USB ケーブル用のコネクタで、Code Composer Studio™ 統合開発環境 (IDE) が実行されているホスト PC からデバイスへの通信に使用されます。
3. このデザインの制御ループ調整には 3.3V の基準電圧が必要です。F28003x の内部基準電圧が使用されますが、そのためには、S3 スwitch を最も高い位置 (INT の方向) にする必要があります。
4. コンデンサは、制御カードの絶縁グラウンドの間 (C7:A) に接続されています。HV 電源を接続する前に取り外してください。

最高の性能を実現するには、複数の ADC チャンネルに RC フィルタ キャップを追加する必要があります。各部品の指示子はシルクスクリーンで明示されており、アセンブリ図は TMDSCNCD280039C 用の C2000Ware (\c2000ware\boards\controlcards\TMDSCNCD280039C) にあります。

信号	コンデンサ部品の指示子	コンデンサ値
TTPLPFC_VBUS	C46	1μF 0603
CLLLC_VSEC	C43	1μF 0603
TTPLPFC_VAC	C32	1μF 0603
TTPLPFC_IAC_PH1	C31	1μF 0603
TTPLPFC_IAC_PH2	C49	1μF 0603
CLLLC_ISEC_TANK	C50	560pF 0603
CLLLC_ISEC	C34	1μF 0603

5.1.2 ソフトウェア

このデザインのソフトウェアは、C2000 マイクロコントローラ向けデジタル電源ソフトウェア開発キット (SDK) (C2000WARE-DIGITALPOWER-SDK) に同梱済みです。

5.1.2.1 Code Composer Studio 内でプロジェクトを開く

まず

- Code Composer Studio (CCS) 統合開発環境 (IDE) ツール フォルダから Code Composer Studio をインストールします。バージョン 12.0 またはそれ以降をお勧めします。
- C2000WARE-DIGITAL-POWER-SDK を次のいずれかの方法でインストールします。
 - C2000Ware Digital Power SDK ツール フォルダからダウンロードします。
 - CCS にアクセスし、[View] → [Resource Explorer] に進みます。テキサス・インスツルメンツの Resource Explorer 下で C2000WARE-DIGITAL-POWER-SDK にアクセスし、[Install] ボタンをクリックします。
- インストールが完了したら、CCS を閉じて、新しいワークスペースを開きます。CCS により自動的に powerSUITE が検出されます。変更を有効にするために、CCS の再起動を求められることがあります。

注

デフォルトでは、SDK とともに powerSUITE がインストールされます。

ファームウェア プロジェクトは次のようにインポートできます。

CCS 内で [Project] → [Import CCS Projects] をクリックし、<SDK>/solutions/tidm_02013/f28003x/ccs にあるソリューション フォルダを参照して、プロジェクトを直接インポートすることもできます。

プロジェクトの様子が表示され、これをクリックすると、すべての依存関係を含んだプロジェクトの自己完結型フォルダが作成されます。

5.1.2.2 プロジェクト構造

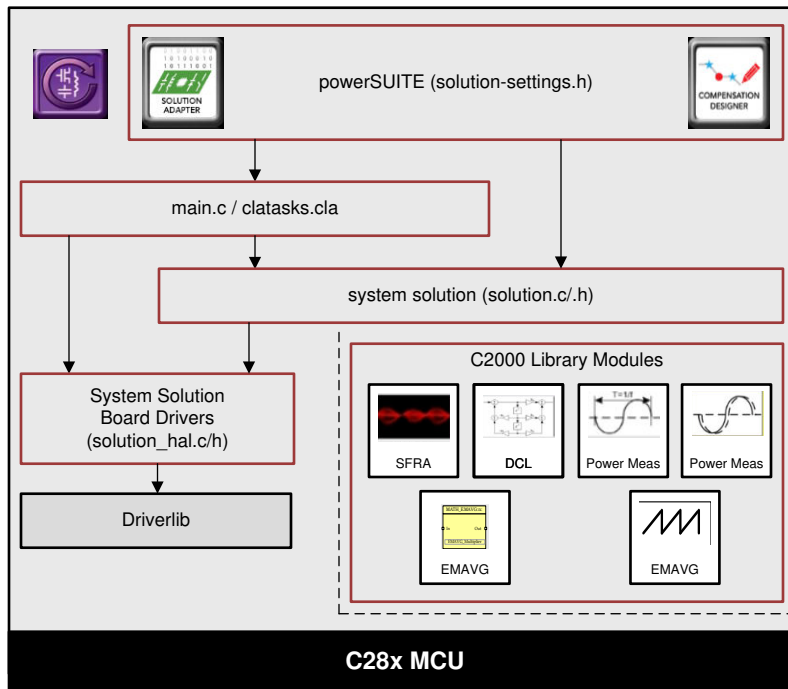


図 5-4. プロジェクト構造の概要

プロジェクトの一般構造を図 1-1 に示します。プロジェクトがインポートされると、図 1-1 に示すように CCS 内に Project Explorer が表示されます。

コア アルゴリズム コードで構成される、デバイスに依存しないソリューション固有のファイルは、<solution>.c/h にあります。たとえば、TTPPLPFC.c または CLLLC.h です。

基板固有およびデバイス固有のファイルは、<solution>_hal.c/h にあります。このファイルは、ソリューションを実行するデバイス特定のドライバで構成されています。別の変調方式やデバイスを使用する場合、プロジェクト内のデバイス サポートファイルを変更する以外に変更を加える必要があるのは、これらのファイルのみです。

<solution>-main.c ファイルは、プロジェクトのメイン フレームワークで構成されています。このファイルは、システム フレームワークの作成に役立つボード ファイルとソリューション ファイルの呼び出し、割り込みサービス ルーチン (ISR)、低速なバックグラウンド タスクで構成されています。

このデザインには、obc_7_4kw、c11lc、ttplpfc の 3 つの <solution> 名があります。最大限の柔軟性を維持するために、CLLLC コード ベースと TTPPLPFC コード ベースをできるだけ独立させることにしました。一方で、obc_7_4kw ファイルは、TTPPFC や CLLLC に依存しない設定を含めるために必要な場所に追加されました。これにより、エンド ユーザーは各段を独立して動作させ、必要に応じて最終設計に PFC 段または DCDC 段のさまざまなトポロジを簡単に組み込むことができます。

<solution>_settings.h ファイルには、どのラボをビルドするかなどのコード構成設定が含まれています。一方、<solution>_user_settings.h には、ADC マッピングや GPIO などの #define マクロのような基板レベルの構成が含まれています。

solution.js ファイルには、各ラボの実行中に観測対象に関連する変数を入力するのに役立つスクリプト ファイルが含まれています。これらのスクリプトを使用するには、スクリプト コンソール ([View] - [Scripting Console]) を開きます。solution.js ファイルの内容をスクリプト コンソールに貼り付け、[Enter] キーを押してください。これにより、後でデバッグに使用するための [Expressions] ウィンドウに入力されます。

ソリューション名は、ソリューションで使用されるすべての変数のモジュール名および定義としても使用されます。そのため、すべての変数および関数呼び出しの前に CLLLC 名が追加されます (CLLLC_vSecSensed_pu など)。この命名規則により、名前の競合を回避しながら、異なるソリューションを組み合わせることができます。

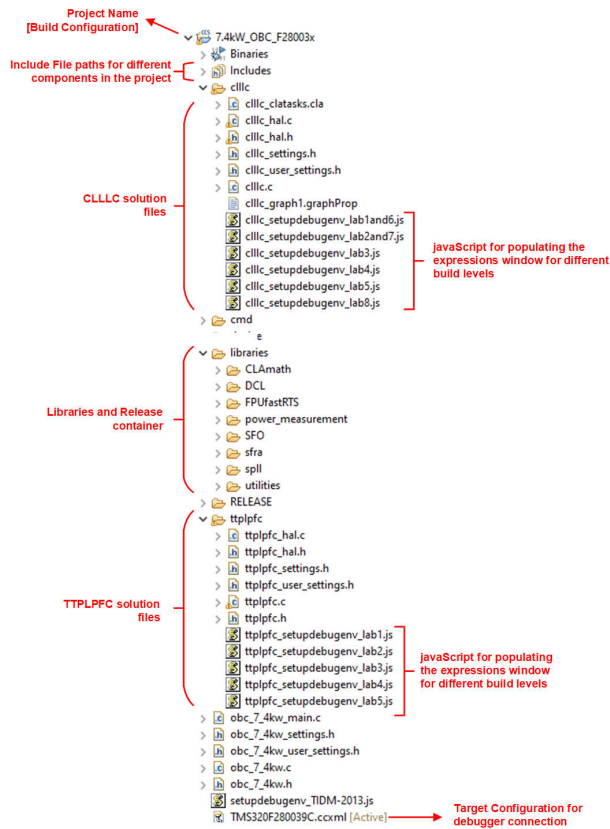


図 5-5. CLLLC プロジェクトの Project Explorer ビュー

OBC プロジェクトは、C28x コア と CLA コアの 2 つのコアで動作する 3 つの ISR (ISR1、ISR2、ISR3) で構成されています。ISR1 のトリガに ePWM、ISR2 のトリガに eCAP、ISR3 のトリガに ADC を使用することで、ISR の優先順位をハードウェアで完全に制御できます。表 5-4 に、各 ISR がどのように分割され、どのようなタスクが実行されるかを示します。

表 5-4. ISR の分割とタスク

ISR	トリガソース	C28x	CLA
ISR1 (120kHz)	ePWM	該当なし	CLLLC PWM 値の更新
ISR2 (120kHz)	eCAP	PFC 電流ループ	CLLLC 制御コード、ISR1 有効化
ISR3 (10kHz)	ADC	PFC 電圧ループ、計測	該当なし

ISR1 は、PWM 更新のために予約された最速かつネスト不可能な ISR で、すべて CLA で実行されます。ISR1 は、PRIM_LEG1_PWM_BASE → EPWM_INT_TBCTR_U_CMPC イベントでトリガされます。一般に、この割り込みは、PRIM_LEG1_PWM_BASE に対して考えられるすべての TBPRD レジスタ値よりも大きい値を CMPC に書き込むことによって無効にされます。これは、CLLLC_HAL_setupISR1Trigger 関数で行います。この ISR に関連する定義は以下のとおりです。

```
#define CLLLC_ISR1_PERIPHERAL_TRIG_BASE CLLLC_PRIM_LEG1_PWM_BASE
#define CLLLC_ISR1_TRIG INT_EPWM1
#define CLLLC_ISR1_PIE_GROUP INTERRUPT_ACK_GROUP3
```

```
#define CLLLC_ISR1_TRIG_CLA CLA_TRIGGER_EPWM1INT
```

ISR2 は両方のコアに分割されます。これにより、TTPPLPFC と CLLLC のコードを簡単にモジュール化できます。C28x で実行される ISR2 と CLA で実行される ISR2 は、どちらも同じソースによってトリガされ、同時に動作します。C28x コアは TTPPLPFC に関連するタスクを実行し、CLA コアは CLLLC の実行に関連するタスクを実行します。

ISR2 は、ISR1 が必要なときに、CMPC への書き込みによって ISR1 をトリガするために有効な値を書き込む役割を担っています。(注:CMPC は、これを可能にするためにグローバル負荷メカニズムに接続されていません。また、CMPC のシャドウ負荷は無効にされています。)CMPC 値を調整して、ISR1 から目的のタイミングを得ることができます。ISR1 が有効になるたびに、2 回トリガされます。最初の ISR1 では、PWM レジスタが更新され、同期が有効になります。2 回目の ISR1 では、PWM 同期が無効にされ、CMPC は ISR1 が再度トリガされないような値に設定されます。分かりやすくするために、ソフトウェア構成図と構造には、最初にトリガされた ISR1 のみが示されています。

ISR2 は ISR2_FREQUENCY で定期的にトリガされます。予備の CAP モジュールを使用して、タイム ベースを生成し、割り込みをトリガします。予備の ePWM モジュールも同じタイム ベースで構成されており、ADC 変換のトリガに使用されます。ISR2 は、制御規則を実行し、PWM に必要なクロック ティックを計算する役割を担っています。シャドウレジスタへの書き込みが完了すると、ISR2 は有効な値 (現在の TBPRD レジスタより小さい値) を CMPC レジスタに書き込むことで、ISR1 トリガを有効にします。ISR2 には、1 次側から 2 次側への電力フロー用と 2 次側から 1 次側への電力フロー用の ISR2_primToSecPowerFlow と ISR2_secToPrimPowerFlow の 2 つのバリエーションがあります。これは、さまざまな方向の電力フローを制御するときに CPU サイクルを最適化するために行われます。分かりやすくするために、各ラボではどちらも ISR2 と呼ばれています。タイミングによっては、ISR1 は、タイミングが非常に重要な更新の書き込みのために ISR2 をネストすることがあります。この ISR に関連する定義は以下のとおりです。

```
#define CLLLC_ISR2_ECAP_BASE ECAP1_BASE
#define CLLLC_ISR2_PWM_BASE EPWM5_BASE
#define CLLLC_ISR2_TRIG INT_ECAP1
#define CLLLC_ISR2_PIE_GROUP INTERRUPT_ACK_GROUP4
```

```
#define CLLLC_ISR2_TRIG_CLA CLA_TRIGGER_ADCA2
```

ISR3 はすべて C28x コアで実行され、ADCINT2 によってトリガされます。ADCINT2 は、CPU タイマを使用して開始される変換によって開始されます。TTPPLPFC の電圧ループの実行や、電流と電圧信号の移動平均を計算してノイズを除

去するなどのハウスキーピング機能に使用されます。コマンドリファレンスのスルーレート機能を実行するのにも使用されます。

```
#define CLLLC_ISR3_TIMEBASE CLLLC_TASKC_CPUTIMER_BASE
#define CLLLC_ISR3_PERIPHERAL_TRIG_BASE ADCC_BASE
#define CLLLC_ISR3_TRIG INT_ADCC2
```

```
#define CLLLC_ISR3_PIE_GROUP INTERRUPT_ACK_GROUP10
```

これにより、割り込みを簡単にネストできます。図 1-1 は、3 つの割り込みのネストを示しています。この図は、開ループのシステムで、[Watch] ウィンドウを通して周期の変化が開始され、ISR1 トリガが 1 回だけ観測されたときのものです。閉ループのシステムの場合、周期はある制御 ISR サイクルから他の制御 ISR サイクルへとわずかに変化するだけであるため、ISR1 は繰り返しトリガされます。

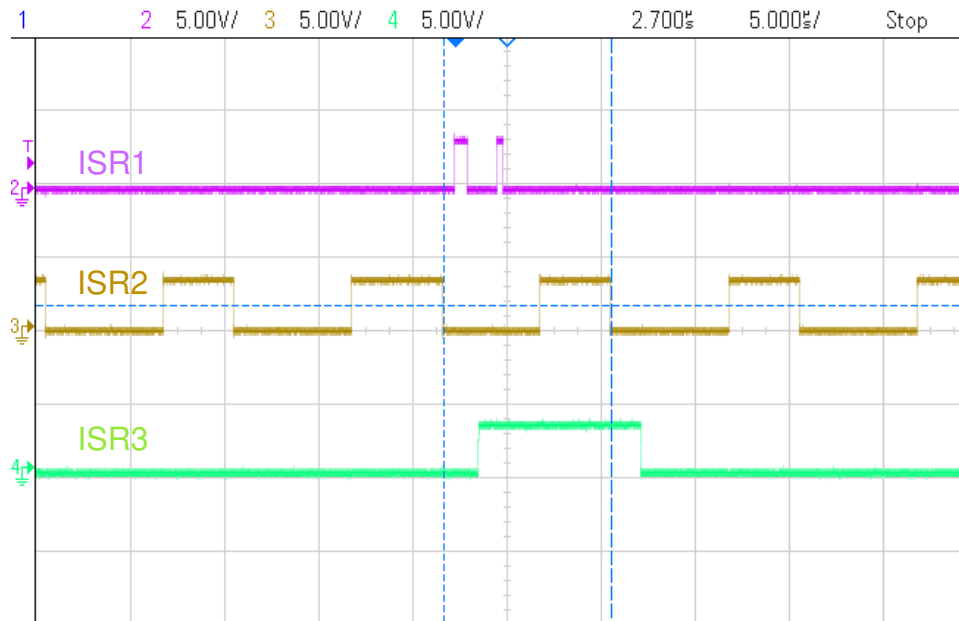


図 5-6. 3 階層にネストされた ISR

さらに、CPU タイマは、低速なバックグラウンド タスクのトリガに使用されます (割り込み駆動ではなく、ポーリング)。

A タスクは、100Hz の TASKA_FREQ でトリガされます。SFRA GUI はこのレートで呼び出す必要があります。1 つのタスク A1 がこのレートで実行されます。

B タスクは、10Hz の TASKB_FREQ でトリガされます。これらは、一部の基本的な LED トグルや、タイミングが重要ではないステート マシンの項目に使用されます。3 つのタスク (B1、B2、B3) がこれによって処理されるため、それぞれの実行レートは 3.33Hz です。

```
#define TASKA_FREQ 100
#define TASKB_FREQ 10
```

このリファレンス デザインのソフトウェアは、ソリューションごとに分けられたラボで構成されており、それぞれにインクリメンタルビルド (INCR_BUILD) があります。これらのテストによって、システムの立ち上げや設計が簡略化されます。

CLLLC ラボ

ラボ 1:1 次側から 2 次側への電力フロー、PWM ドライバの開ループ チェック (基板に大電力が印加されていない状態)。セクション 5.2.2.1 を参照してください。

ラボ 2:1 次側から 2 次側への電力フロー、PWM ドライバおよび保護付き ADC の開ループ チェック (2 次側に抵抗性負荷が接続されている状態)。セクション 5.2.2.2 を参照してください。

ラボ 3:1 次側から 2 次側への電力フロー、閉電圧ループ チェック (2 次側に抵抗性負荷が接続されている状態)。セクション 5.2.2.3 を参照してください。

ラボ 4:1 次側から 2 次側への電力フロー、閉電流ループ チェック (2 次側に抵抗性負荷が接続されている状態)。セクション 5.2.2.4 を参照してください。

ラボ 5:1 次側から 2 次側への電力フロー、閉電流ループ チェック (2 次側で抵抗性負荷が電圧源と並列に接続されてバッテリー接続をエミュレートしている状態)。セクション 5.2.2.5 を参照してください。

これらの定義は settings.h ファイル内にあり、そのファイル内で直接変更できます。

5.2 テストと結果

5.2.1 テストのセットアップ (初期設定)

このデザインのテストを開始するためのハードウェア テストのセットアップを、[図 1-1](#) 示します。ハードウェアの詳細なセットアップについては、[セクション 5.1.1](#) で説明しています。

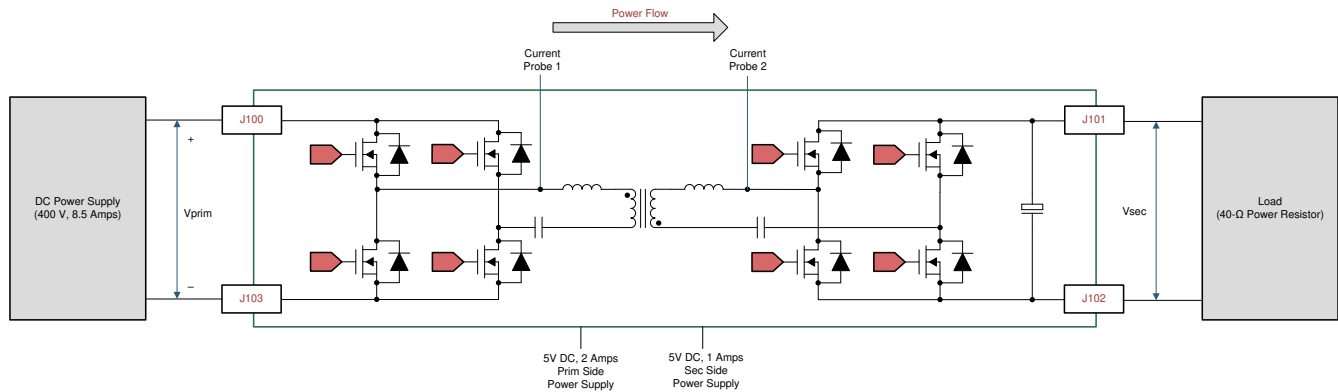


図 5-7. ソフトウェアを実行するためのハードウェアセットアップ

5.2.2 CLLLC のテスト手順

5.2.2.1 ラボ 1.1 次側から 2 次側への電力フロー、PWM ドライバの開ループ チェック

このラボ オプションは、リファレンス デザインのハードウェア接続とは無関係に実行できるように、主にソフトウェアの観点から PWM だけにフォーカスしたテストとして行われます。このラボでは、C2000 の制御カードまたはローンチパッドでコードを実行し、PWM 波形を観測することができます。

ロードと実行に使用される手順は、[セクション 5.2.2.2](#) と同様です。

PWM ドライバに変更がなければ、このラボをスキップしてラボ 2 に直接進むこともできます。このことから、このラボの手順は、主に PWM ドライバの開発とデバッグが目的であるため、文書化されていません。

5.2.2.2 ラボ 2.1 次側から 2 次側への電力フロー、PWM ドライバおよび保護付き ADC の開ループ チェック (2 次側に抵抗性負荷が接続されている状態)

このビルドでは、基板は [Watch] ウィンドウから変更可能な指定された周波数で開ループ方式で励起されます。周波数は CLLLC_pwmPeriodRef_pu 変数で制御されます。

このビルドでは、電力段からの帰還値のセンシングと PWM ゲートドライバの動作を検証し、ハードウェアに問題がないことを確認します。また、このビルドでは入出力電圧センシングの較正も実行できます。このビルドのソフトウェア構成図を、[図 1-1](#) に示します。

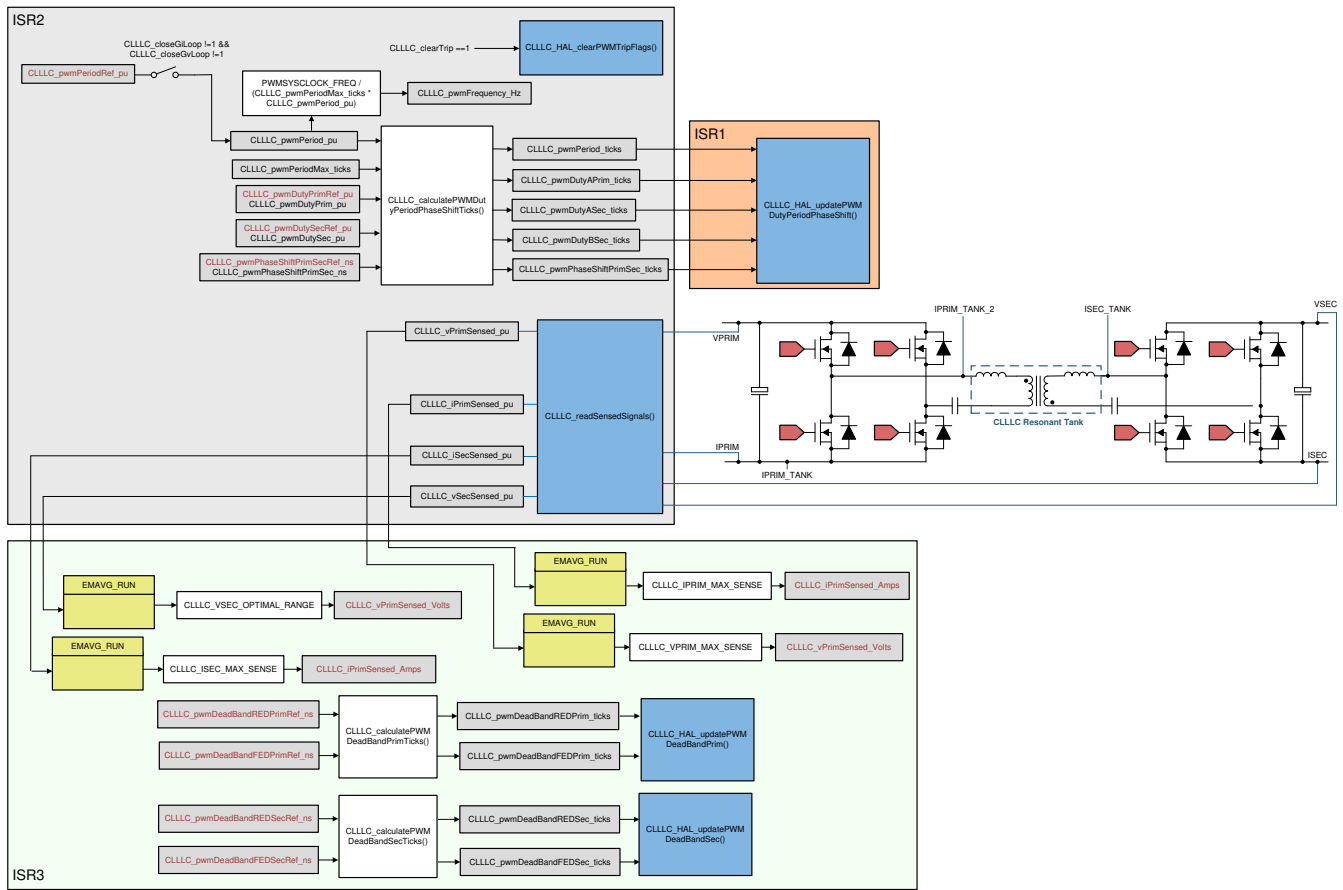


図 5-8. ラボ 1 とラボ 2 のソフトウェア構成図

5.2.2.2.1 ラボ 2 のソフトウェアオプションの設定

1. 開始するには、[セクション 5.1.2.1](#) で説明したように、CCS プロジェクトを開きます。
2. このビルドでは、`settings.h` ファイルに以下の定義が設定されています。

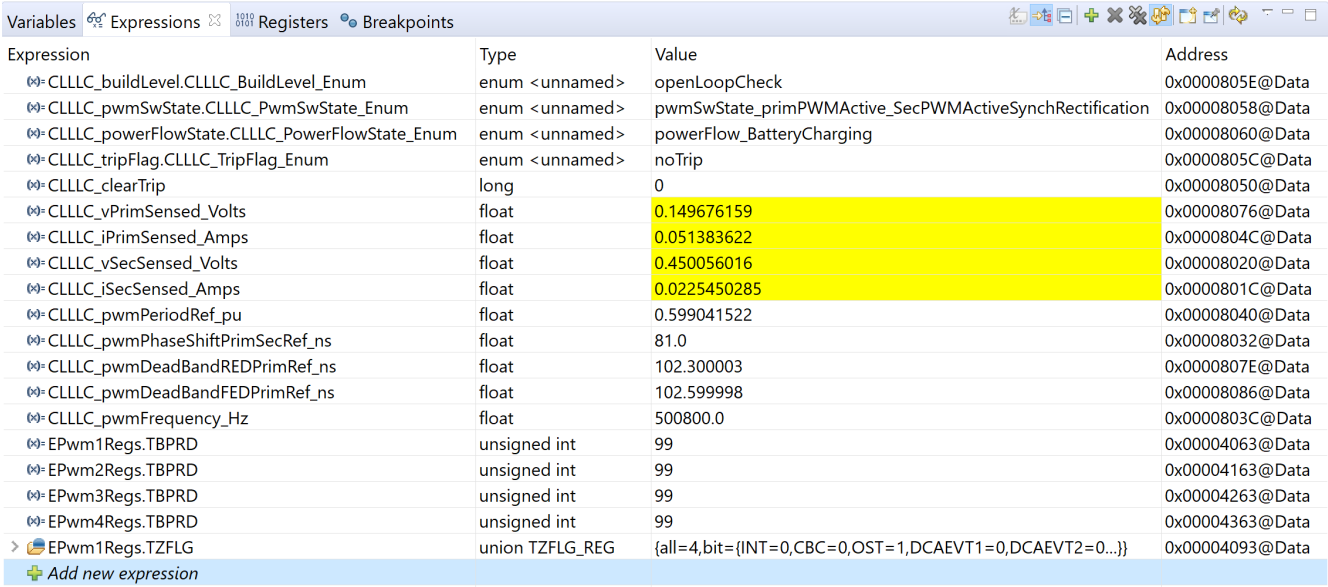
```
#if CLLLC_LAB == 2
#define CLLLC_CONTROL_RUNNING_ON CLA_CORE
#define CLLLC_POWER_FLOW CLLLC_POWER_FLOW_PRIM_SEC
#define CLLLC_INCR_BUILD CLLLC_OPEN_LOOP_BUILD
#define CLLLC_TEST_SETUP CLLLC_TEST_SETUP_RES_LOAD
#define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED

#if CLLLC_SFRA_ALLOWED == 1
#define CLLLC_SFRA_TYPE CLLLC_SFRA_VOLTAGE
#else
#define CLLLC_SFRA_TYPE CLLLC_SFRA_DISABLED
#endif

#define CLLLC_SFRA_AMPLITUDE (float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL2
#endif
```

5.2.2.2.2 プロジェクトのビルドおよびロードとデバッグ環境の設定

1. プロジェクト名を右クリックし、[Rebuild Project] をクリックします。
2. プロジェクトが正常にビルドされます。
3. [Project Explorer] で、targetConfigs 下で適切な目標構成ファイルが有効になっていることを確認します (図 1-1 を参照)。
4. その後、[Run] → [Debug] をクリックしてデバッグ セッションを開始します。デュアル CPU デバイスの場合、デバッグ を実行する CPU を選択するウィンドウが表示されます。ここでは、CPU1 を選択します。
5. プロジェクトがデバイスにロードされ、CCS デバッグ ビューが有効になります。コードはメイン ルーチンの開始で停止 します。
6. [Watch] および [Expressions] ウィンドウに変数を追加するには、[View] → [Scripting Console] をクリックして、 [Scripting Console] ダイアログ ボックスを開きます。このコンソールの右上隅で、[Open] をクリックして、プロジェクト フォルダ内にある setupdebugenv_lab2and7.js スクリプト ファイルを参照します。このファイルにより、[Watch] ウィン ドウに、システムをデバッグするのに必要な適切な変数が入力されます。
7. [Watch] ウィンドウで [Continuous Refresh] ボタン (🔄) をクリックして、コントローラからの値の連続更新を有効にし ます。図 1-1 に示すように [Watch] ウィンドウが表示されます。




Expression	Type	Value	Address
CLLLC_buildLevel.CLLLC_BuildLevel_Enum	enum <unnamed>	openLoopCheck	0x0000805E@Data
CLLLC_pwmSwState.CLLLC_PwmSwState_Enum	enum <unnamed>	pwmSwState_primPWMActive_SecPWMActiveSynchRectification	0x00008058@Data
CLLLC_powerFlowState.CLLLC_PowerFlowState_Enum	enum <unnamed>	powerFlow_BatteryCharging	0x00008060@Data
CLLLC_tripFlag.CLLLC_TripFlag_Enum	enum <unnamed>	noTrip	0x0000805C@Data
CLLLC_clearTrip	long	0	0x00008050@Data
CLLLC_vPrimSensed_Volts	float	0.149676159	0x00008076@Data
CLLLC_iPrimSensed_Amps	float	0.051383622	0x0000804C@Data
CLLLC_vSecSensed_Volts	float	0.450056016	0x00008020@Data
CLLLC_iSecSensed_Amps	float	0.0225450285	0x0000801C@Data
CLLLC_pwmPeriodRef_pu	float	0.599041522	0x00008040@Data
CLLLC_pwmPhaseShiftPrimSecRef_ns	float	81.0	0x00008032@Data
CLLLC_pwmDeadBandREDPrimRef_ns	float	102.300003	0x0000807E@Data
CLLLC_pwmDeadBandFEDPrimRef_ns	float	102.599998	0x00008086@Data
CLLLC_pwmFrequency_Hz	float	500800.0	0x0000803C@Data
EPwm1Regs.TBPRD	unsigned int	99	0x00004063@Data
EPwm2Regs.TBPRD	unsigned int	99	0x00004163@Data
EPwm3Regs.TBPRD	unsigned int	99	0x00004263@Data
EPwm4Regs.TBPRD	unsigned int	99	0x00004363@Data
EPwm1Regs.TZFLG	union TZFLG_REG	{all=4,bit={INT=0,CBC=0,OST=1,DCAEVT1=0,DCAEVT2=0...}}	0x00004093@Data

図 5-9. ラボ 2 [Expression] ウィンドウ

5.2.2.2.3 リアルタイム エミュレーションの使用

リアルタイム エミュレーションは、マイクロコントローラ動作中に Code Composer Studio 内のウィンドウを更新できる特別なエミュレーション機能です。この機能により、グラフおよび Watch ビューが更新されるだけでなく、[Watch] ウィンドウや [Memory] ウィンドウで値を変更したり、プロセッサを停止することなくシステムにおける変更の反映を確認したりできます。

1. マウス ポインタを水平ツールバーのボタンの上に置き、 **Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running)** ボタンをクリックして、リアルタイム モードを有効に します。
2. メッセージ ボックスが表示されることがあります。その場合は [YES] を選択して、デバッグ イベントを有効にします。こ れにより、ステータス レジスタ 1 (ST1) のビット 1 (DGBM ビット) が 0 に設定されます。DGBM は、デバッグ イネー ブル マスク ビットです。DGBM ビットが「0」に設定されると、メモリ値とレジスタ値がホスト プロセッサに渡されて、デバ ッガのウィンドウが更新できるようになります。

5.2.2.2.4 コードの実行

1.  をクリックしてプロジェクトを実行します。
2. ここで、CLLLC_clearTrip 変数に 1 を書き込んで検出をクリアします。
3. Watch ビューで、CLLLC_vPrimSensed_Volts、CLLLC_iPrimSensed_Amps、CLLLC_vSecSensed_Volts、CLLLC_iSecSensed_Amps の各変数が定期的に更新されているかどうかチェックします。(注:この時点では電力が印加されていないため、これらの値はゼロに近くなっています。)
4. 入力 VPRIM DC 電圧を 0V から 400V に徐々に上げます。CLLLC_vPrimSensed_Volts に正しい値が表示されていることを確認します。
5. デフォルトでは CLLLC_pwmPeriodRef_pu 変数は、[図 1-1](#) に示すように 0.599 に設定されており、これは 500.8kHz です。これは、コンバータの直列共振周波数に近いですが、実際のハードウェアに搭載されている部品のばらつきが原因で、直列共振周波数より低くなったり、高くなったりします。たとえば、[図 1-1](#) では、直列共振周波数よりもわずかに低い周波数が確認できます。
6. VSEC 変数は、設計したタンク ゲインに応じて、300V 近い電圧を示します。CLLLC_vSecSensed_Volts が正しい電圧を示していることを確認します。これにより、基板の電圧センシングが検証されます。

Expression	Type	Value	Address
CLLLC_buildLevel.CLLLC_BuildLevel_Enum	enum <unnamed>	openLoopCheck	0x0000805E@Data
CLLLC_pwmSwState.CLLLC_PwmSwState_Enum	enum <unnamed>	pwmSwState_primPWMActive_SecPWMActiveSynchRectification	0x00008058@Data
CLLLC_powerFlowState.CLLLC_PowerFlowState_Enum	enum <unnamed>	powerFlow_BatteryCharging	0x00008060@Data
CLLLC_tripFlag.CLLLC_TripFlag_Enum	enum <unnamed>	noTrip	0x0000805C@Data
CLLLC_clearTrip	long	0	0x00008050@Data
CLLLC_vPrimSensed_Volts	float	403.606567	0x00008076@Data
CLLLC_iPrimSensed_Amps	float	4.82742071	0x0000804C@Data
CLLLC_vSecSensed_Volts	float	296.258301	0x00008020@Data
CLLLC_iSecSensed_Amps	float	6.27206373	0x0000801C@Data
CLLLC_pwmPeriodRef_pu	float	0.599041522	0x00008040@Data
CLLLC_pwmPhaseShiftPrimSecRef_ns	float	81.0	0x00008032@Data
CLLLC_pwmDeadBandREDPrimRef_ns	float	102.300003	0x0000807E@Data
CLLLC_pwmDeadBandFEDPrimRef_ns	float	102.599998	0x00008086@Data
CLLLC_pwmFrequency_Hz	float	500800.0	0x0000803C@Data
EPwm1Regs.TBPRD	unsigned int	99	0x00004063@Data
EPwm2Regs.TBPRD	unsigned int	99	0x00004163@Data
EPwm3Regs.TBPRD	unsigned int	99	0x00004263@Data
EPwm4Regs.TBPRD	unsigned int	99	0x00004363@Data
EPwm1Regs.TZFLG	union TZFLG_REG	{all=0,bit={INT=0,CBC=0,OST=0,DCAEVT1=0,DCAEVT2=0...}}	0x00004093@Data

図 5-10. ラボ 2 [Expression] ウィンドウ、共振時

1. テスト条件で指定された負荷では、PRIM 側と SEC 側からの電流は、CLLLC_iPrimSensed_Amps で 4.8A、CLLLC_iSecSensed_Amps で 6.8A になります。

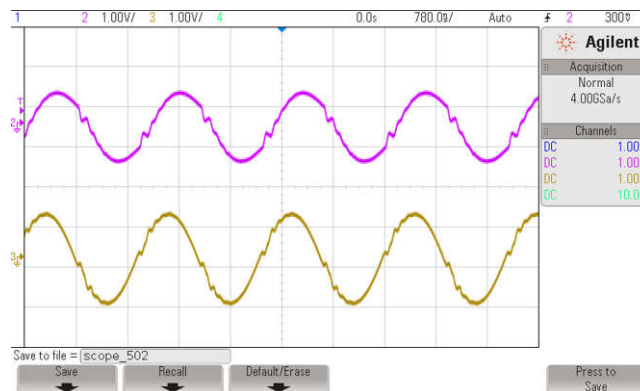


図 5-11. ラボ 2、共振時の 1 次側 (ch2) 電流と 2 次側 (ch3) 電流

- 次に、さまざまな周波数 (共振より高い周波数、共振より低い周波数) での動作を確認するには、CLLLC_pwmPeriodRef_pu 変数を 639kHz の周波数に対応する 0.47 に変更します。この条件での波形を [図 1-1](#) に示します。

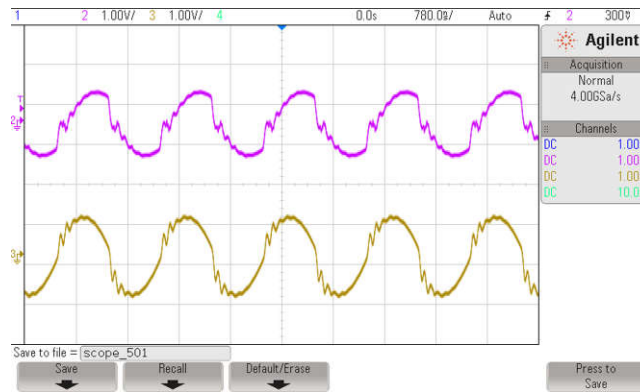


図 5-12. ラボ 2、直列共振周波数を上回る 1 次側 (ch2) 電流と 2 次側 (ch3) 電流

- 次に、CLLLC_pwmPeriodRef_pu に 0.8 を入力して、直列共振周波数を下回る状態で動作をテストします。これにより、生成周波数は 374kHz になります。この場合、1 次側電流は不連続になり、2 次側のデューティ サイクルが変調して、[図 1-1](#) に示すダイオード エミュレーションが実現されます。
- これにより、基本的なレベルで PWM ドライバとハードウェアの接続を検証できます。

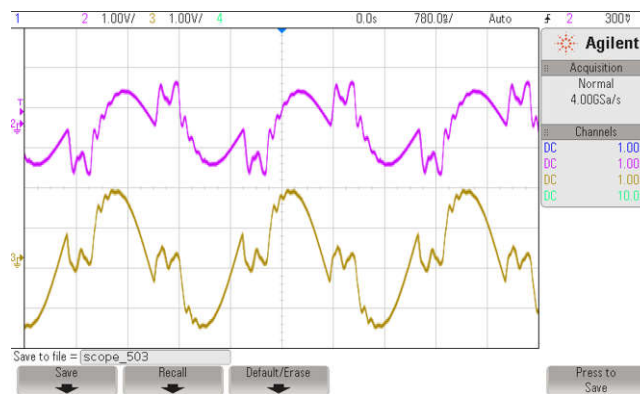


図 5-13. ラボ 2、直列共振周波数を下回る 1 次側 (ch2) 電流と 2 次側 (ch3) 電流

5.2.2.2.5 電圧ループに対する SFRA プラントの測定

- このビルドのソフトウェアには SFRA が統合されているため、プラントの応答を測定して、補償器の設計に使用できます。SFRA を実行するには、プロジェクトを実行したまま、<Install directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\SFRA_GUI.exe に移動します。
- SFRA GUI でデバイスのオプションを選択します。たとえば、F280039 の場合は浮動小数点を選択します。[Setup Connection] をクリックします。ポップアップ ウィンドウで [Boot on Connect] オプションを選択解除し、適切な COM ポートを選択します。[OK] をクリックします。SFRA GUI に戻り、[Connect] をクリックします。
- SFRA GUI がデバイスに接続します。これで [Start Sweep] をクリックして、SFRA 掃引を開始できるようになりました。SFRA 掃引が完了するまでには数分かかります。SFRA GUI のプログレス バーを確認したり、UART の動作を示す制御カード裏面の青色 LED の点滅をチェックすることで、動作を監視できます。完了すると、[図 1-1](#) に示すように測定結果のグラフが表示されます。(ループが閉じていないため、ラボでは開ループ測定は無効です。プラント測定のみを参照してください)。

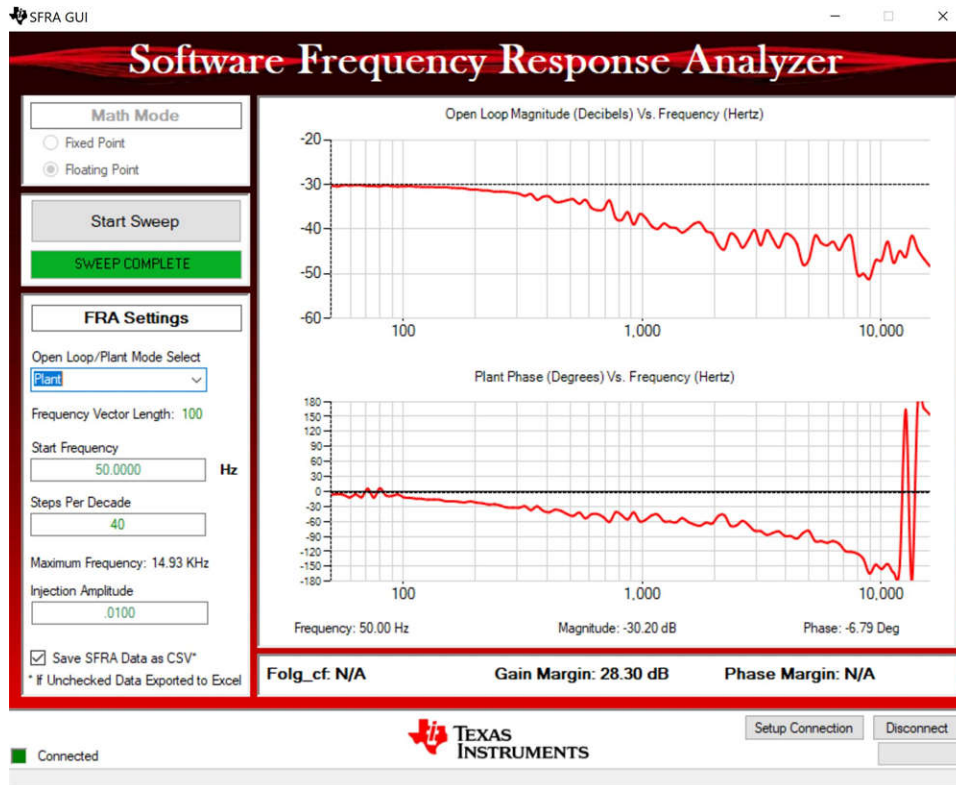


図 5-14. 閉電圧ループに対する SFRA 開ループプロット (Vprim 400V、Vsec 300V、電力 1.972kW、Fsw 500kHz)

また、周波数応答データは SFRA データフォルダ下のプロジェクトフォルダに保存され、SFRA 実行時のタイムスタンプが記録されます。SFRA は、システムの動作範囲をカバーするために、異なる周波数設定ポイントで実行できます。補償器は、これらの測定プロットを使用して次のラボで設計されるため、このタイムスタンプを覚えておくか、SFRA.csv ファイルの名前をわかりやすいものに変更してください。

異なる周波数ポイントで分析を繰り返してください。プラントゲインは周波数ポイントによって異なります。333kHz で測定されたゲインについては 図 1-1 を、680kHz で測定されたゲインについては 図 1-1 を参照してください。このことから、コンバータの周波数範囲全体で安定した補償器を選択する必要があります。すべての実行結果は CSV ファイルに保存され、Compensation Designer にインポートして動作範囲全体の安定性をチェックできます。

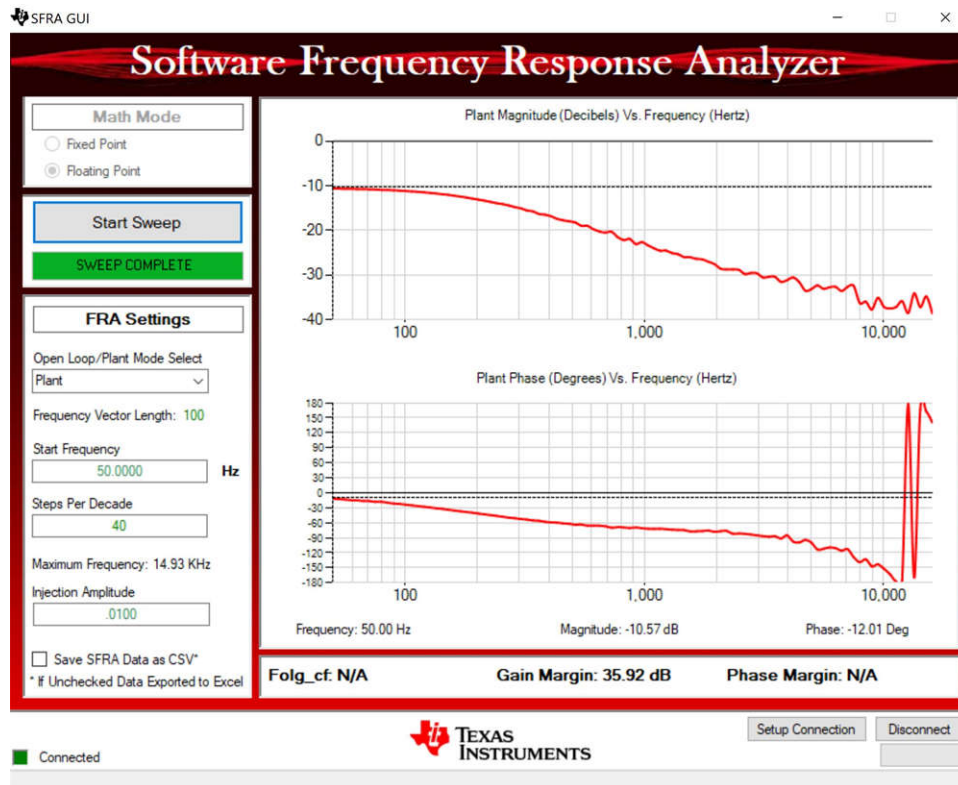


図 5-15. 閉電圧ループに対する SFRA 開ループプロット (V_{prim} 400V、 V_{sec} 320V、電力 2.174kW、 F_{sw} 333kHz)

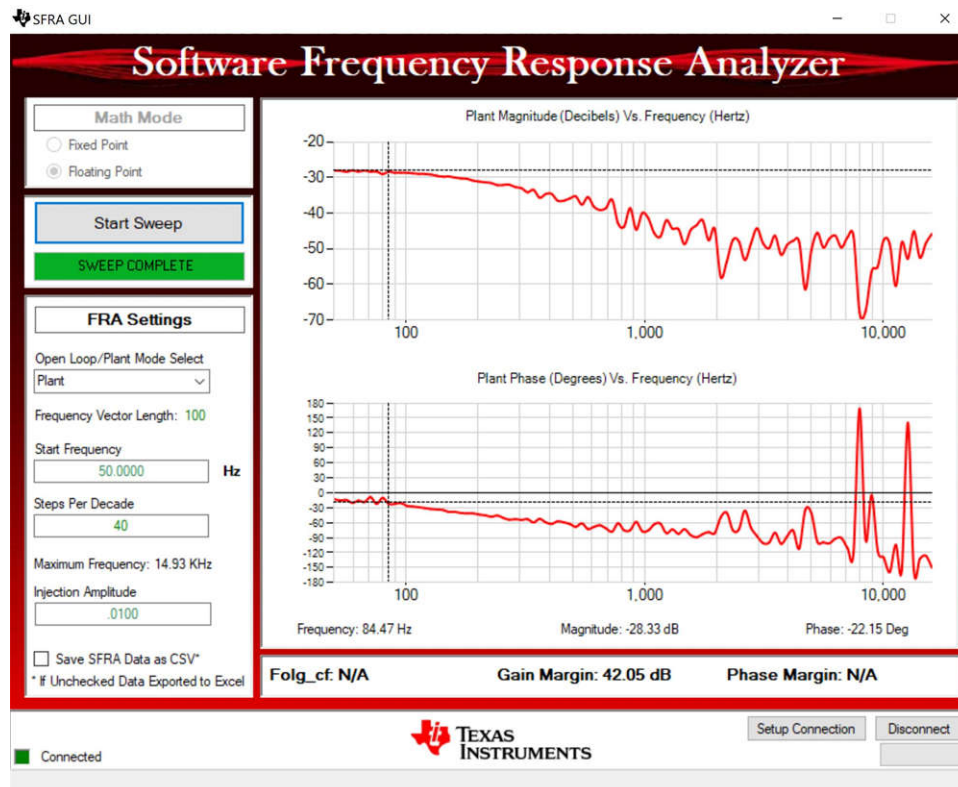


図 5-16. 閉電圧ループに対する SFRA 開ループプロット (V_{prim} 400V、 V_{sec} 293V、電力 1.828kW、 F_{sw} 680kHz)

5.2.2.2.6 アクティブ同期整流の検証

- オプションとして、アクティブ同期整流を検証するために、PWM 信号をプローブしてデューティサイクルの変化を確認することもできます。プローブを接続するには、まず以下の手順で出力段を停止し、テスト対象の回路すべての電源をオフにする必要があります。

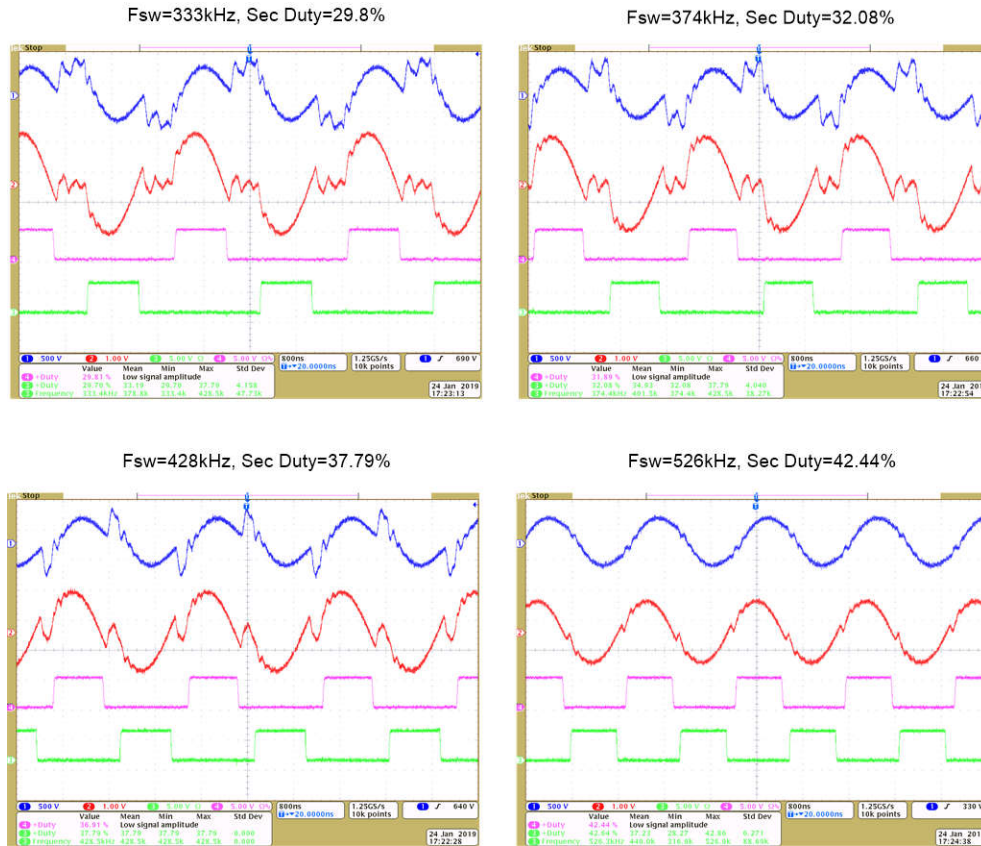






図 5-17. アクティブ同期整流のチェック、ch1 → IPRIM_TANK、ch2 → ISEC_TANK、ch3/ch4 → SEC_LEG1_PWMH/L

- 終了したら、入力電圧 VPRIM をゼロまで下げます。[Watch] ウィンドウの電圧がゼロまで下がるのを確認します。
- リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの

[Halt] ボタン () を使用するか、[Target] → [Halt] の順にクリックして、プロセッサを停止します。次に、  をクリックして、マイクロコントローラをリアルタイム モードから解除します。最後に、マイクロコントローラ () をリセットします。

- [Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグセッションを終了します。

5.2.2.2.7 電流ループに対する SFRA プラントの測定

- SYSCFG ページに戻り、SFRA オプションで電流を選択して電流ループ プラントを測定します
- プロジェクトをリビルドし、リロードします。2 (セクション 5.2.2.2.1) から セクション 5.2.2.2.7 (セクション 5.2.2.2.5) を繰り返します。今回は、SFRA 掃引によってプラントの電流ループが測定されます。この CSV ファイルは、後でラボ 3 で使用するために保存してください。すべての動作条件がカバーされていることを確認するために、複数ポイントでの測定が可能です。図 1-1 に、500kHz での電流ループ プラントの測定結果を示します。

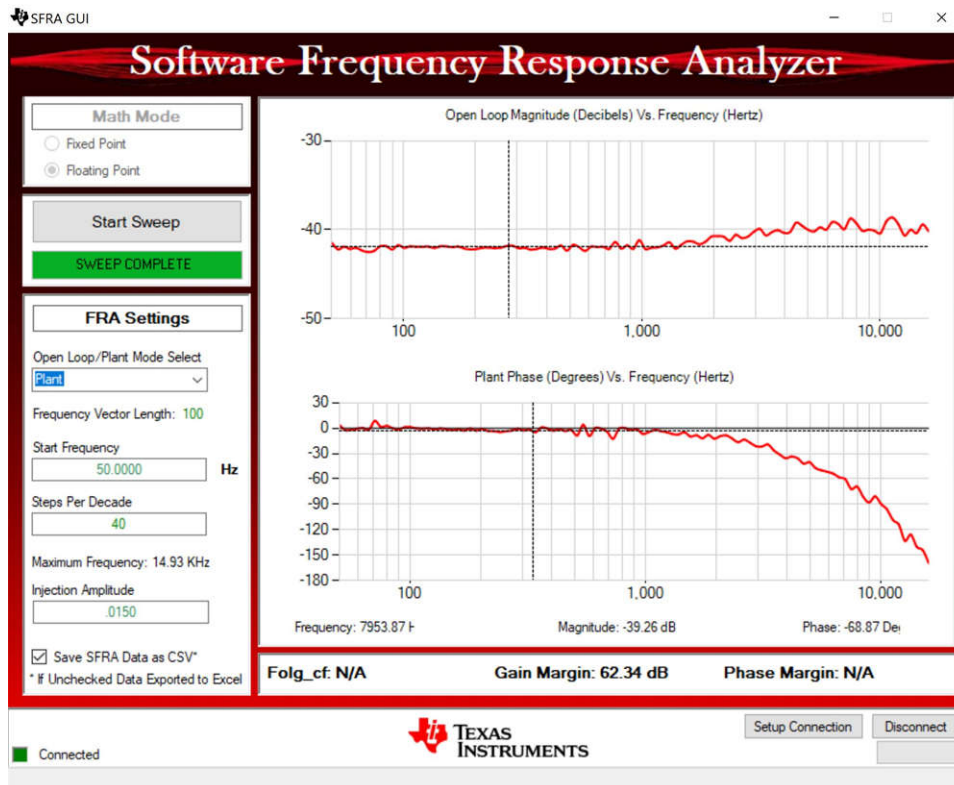






図 5-18. 電流ループに対する SFRA プラントの測定 (Vprim 400V、Vsec 295V、スイッチング周波数 500kHz、1887W)

3. これでこのビルドのチェックは完了し、このビルドが正常に終了した時点で次の項目を検証します。
 - a. 電圧および電流の検出とスケールリングが適正であること
 - b. ISR1、ISR2、ISR3 における BUILD 1 コードの割り込み生成および実行
 - c. PWM ドライバおよびスイッチング
 - d. 電流と電圧ループのプラント測定

問題が確認された場合には、ビルドの問題を解消するためにハードウェアを慎重に点検する必要があります。

4. これでコントローラを停止し、デバッグ接続を終了できます。
5. リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの

[Halt] ボタン () を使用するか、[Target] → [Halt] の順にクリックして、プロセッサを停止します。次に、  をクリックして、マイクロコントローラをリアルタイム モードから解除します。最後に、マイクロコントローラ () をリセットします。

6. [Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグセッションを終了します。

5.2.2.3 ラボ 3.1 次側から 2 次側への電力フロー、閉電圧ループ チェック (2 次側に抵抗性負荷が接続されている状態)

このラボでは、電圧ループ G_V は出力側に抵抗性負荷が接続された状態で閉じています。このビルドのソフトウェア構成図を [図 1-1](#) に示します。ハードウェアは、[図 1-1](#) に示すように設定されているものとします。

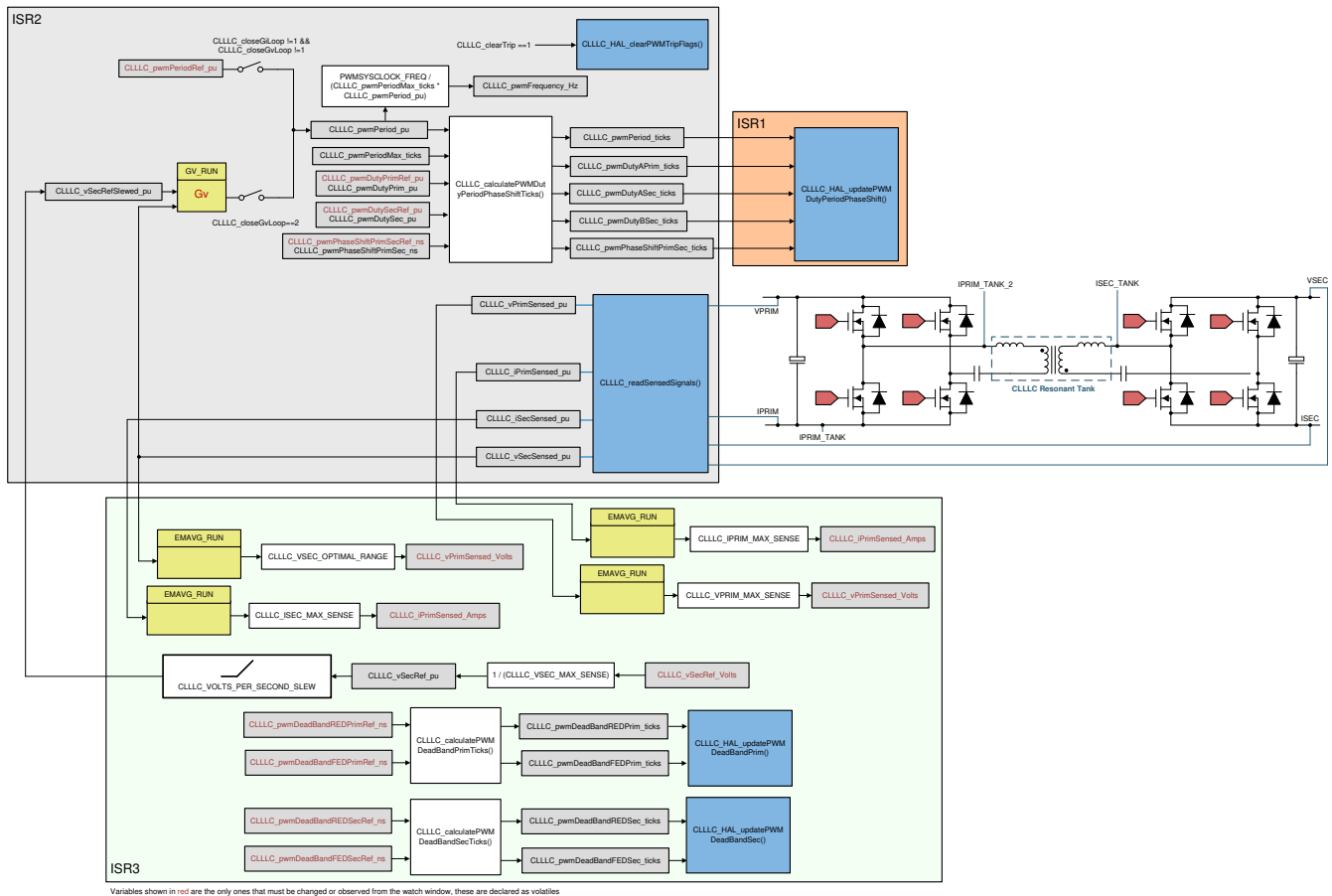


図 5-19. ソフトウェア構成図: 閉電圧ループ、1 次側から 2 次側への電力フロー

5.2.2.3.1 ラボ 3 のソフトウェアオプションの設定

1. このラボを実行するには、前のセクション、[図 1-1](#) で説明したようにハードウェアが設定されていることを確認します。高電圧電力はまだ基板に供給しないでください。
2. <install Directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\CompDesigner.exe を開きます。
3. Compensation Designer が起動し、有効な SFRA データ ファイルを選択するように促されます。ラボ 1 の実行から補償デザイナーに SFRA データをインポートし、2 極、2 ゼロの補償器を設計します。ループが閉じたときにシステムが安定するように、設計のこの段階でより多くのマージンを確保しておくことをお勧めします。SFRA のさまざまな実行から得られたプラント データをチェックして、あらゆる条件下で安定したシステムを得ることができます。たとえば、設計した補償器を使用した 500kHz と 300kHz における 2 つの実行は、[図 1-1](#) と [図 1-1](#) に示すように安定していることがわかります。



図 5-20. 電圧ループの SFRA ベースのプラント測定を用いた補償器の設計、出力に抵抗性負荷が接続されている場合 (500kHz の測定データ)

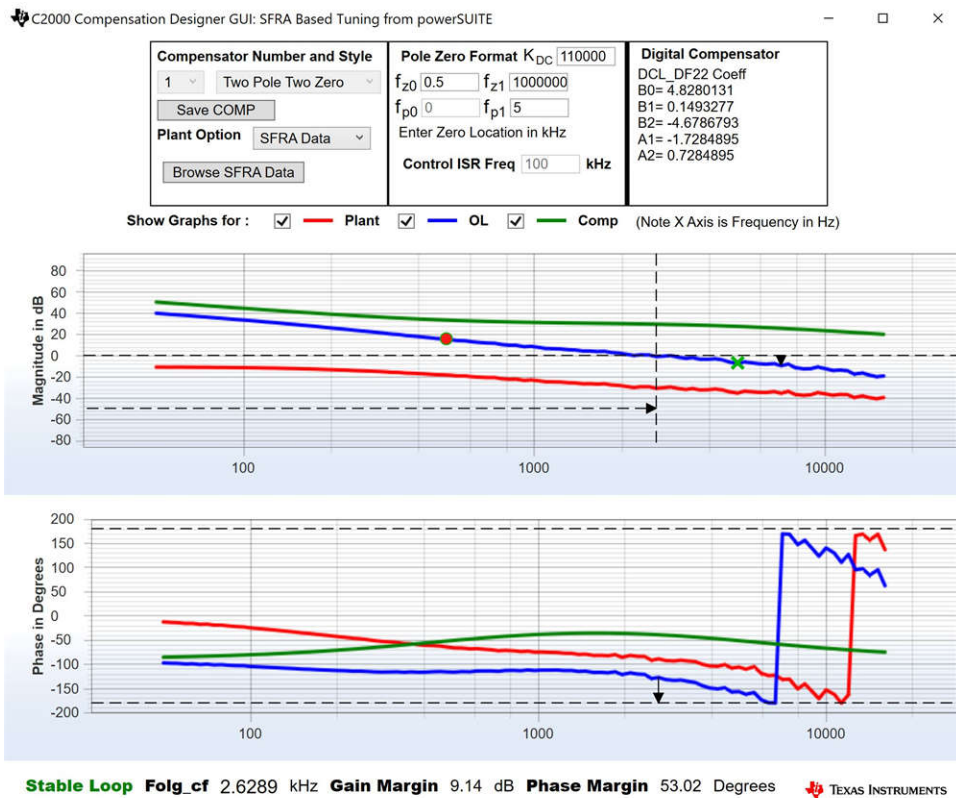


図 5-21. 電圧ループの SFRA ベースのプラント測定を用いた補償器の設計、出力に抵抗性負荷が接続されている場合 (333kHz の測定データ)

注



調整は DF22 方式で行われますが、DF13 はソフトウェアで実行します。これは、DF22 構造ではできないのですが、DF13 ではソフトスタートが簡単なためです。どちらの場合も係数は同じです。本書の作成時点では、DF12 構造は DCL では使用できません。

4. 補償器の設計が完了したら、CLLLC_settings.h ファイルで補償器の値を更新できます。
5. Compensation Designer を閉じます。
6. このビルドでは、settings.h ファイルに以下の定義が設定されています。

```
#if CLLLC_LAB == 3 #define CLLLC_CONTROL_RUNNING_ON CLA_CORE #define CLLLC_POWER_FLOW
CLLLC_POWER_FLOW_PRIM_SEC #define CLLLC_INCR_BUILD CLLLC_CLOSED_LOOP_BUILD #define
CLLLC_CONTROL_MODE CLLLC_VOLTAGE_MODE #define CLLLC_TEST_SETUP CLLLC_TEST_SETUP_RES_LOAD
#define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED #if CLLLC_SFRA_ALLOWED == 1 #define
CLLLC_SFRA_TYPE CLLLC_SFRA_VOLTAGE #else #define CLLLC_SFRA_TYPE CLLLC_SFRA_DISABLED #endif
#define CLLLC_SFRA_AMPLITUDE (float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL1 #endif
```

5.2.2.3.2 プロジェクトのビルドおよびロードとデバッグ環境の設定

1. プロジェクト名を右クリックし、[Rebuild Project] をクリックします。
2. プロジェクトが正常にビルドされます。
3. [Run] → [Debug] をクリックしてデバッグ セッションを開始します。デュアル CPU デバイスの場合、デバッグを実行する CPU を選択するウィンドウが表示されます。ここでは、CPU1 を選択します。
4. プロジェクトがデバイスにロードされ、CCS デバッグ ビューが有効になります。コードはメイン ルーチンの開始で停止します。
5. [Watch] および [Expressions] ウィンドウに変数を追加するには、[View] → [Scripting Console] をクリックして、[Scripting Console] ダイアログ ボックスを開きます。このコンソールの右上隅で、[Open] をクリックして、プロジェクトフォルダ内にある setupdebugenv_lab3.js スクリプト ファイルを参照します。このファイルにより、[Watch] ウィンドウに、システムをデバッグするのに必要な適切な変数が入力されます。

6. [Watch] ウィンドウで [Continuous Refresh] ボタン () をクリックして、コントローラからの値の連続更新を有効にします。
7. マウス ポインタを水平ツールバーのボタンの上に置き、

`Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running)`

 ボタンをクリックして、リアルタイム モードを有効にします。

5.2.2.3.3 コードの実行

1.  をクリックしてプロジェクトを実行します。
2. ここで、CLLLC_clearTrip 変数に 1 を書き込んで検出をクリアします。CLLLC_closeGvLoop 変数がまだ 0 に設定されていないため、コンバータは開ループで動作します。ファームウェアにはソフトスタートが実装されていないため、まず 1 次側と 2 次側の電圧を手動でソフトスタートします。
3. Watch ビューで、CLLLC_vPrimSensed_Volts、CLLLC_iPrimSensed_Amps、CLLLC_vSecSensed_Volts、CLLLC_iSecSensed_Amps の各変数が定期的に更新されているかどうかチェックします。(注:この時点では電力が印加されていないため、これらの値はゼロに近くなっています。)
4. 入力 PRIM DC 電圧を 0V から 400V に徐々に上げて、コンバータをソフトスタートします。CLLLC_vPrimSensed_Volts に VPRIM の正しい値 (400V に近い値) が表示されていることを確認します。
5. デフォルトでは、CLLLC_pwmPeriodRef_pu 変数は 0.599 (500.8kHz) に設定されています。これは、コンバータの直列共振周波数に近いですが、実際のハードウェアに搭載されている部品のばらつきが原因で、直列共振周波数より低くなったり、高くなったりします。
6. 400V の 1 次側入力で、巻線比が 1.33 の場合、CLLLC_vSecSensed_Volts 変数は 300V 近くになります。CLLLC_vSecRef_Volts 変数を 300V に設定します。
7. CLLLC_closeGvLoop 変数を 1 に設定します。これで電圧ループが閉じて、コントローラによる電圧制御が試行されます。
8. CLLC_vSecRef_Volts を 295V から 320V に変化させて、開ループ動作をテストすると、このコマンドリファレンスに CLLLC_vSecSensed_Volts が追従することが確認できます。コンバータは、直列共振周波数以下、共振周波数、共振周波数以上で動作します。電圧を 300V に戻して SFRA を実行します

5.2.2.3.4 閉電圧ループに対する SFRA の測定

1. このビルドのソフトウェアには SFRA が統合されているため、ハードウェアを測定して、設計した補償器が十分なゲイン マージンと位相マージンを提供していることを検証できます。SFRA を実行するには、プロジェクトを実行したまま、<Install directory >\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\SFRA_GUI.exe に移動します。
2. SFRA GUI でデバイスのオプションを選択します。たとえば、F280039 の場合は浮動小数点を選択します。[Setup Connection] をクリックします。ポップアップ ウィンドウで [Boot on Connect] オプションを選択解除し、適切な COM ポートを選択します。[OK] をクリックします。SFRA GUI に戻り、[Connect] をクリックします。
3. SFRA GUI がデバイスに接続します。これで [Start Sweep] をクリックして、SFRA 掃引を開始できるようになりました。SFRA 掃引が完了するまでには数分かかります。SFRA GUI のプログレス バーを確認したり、UART の動作を示す制御カード裏面の青色 LED の点滅をチェックすることで、動作を監視できます。終了すると、[図 1-1](#) のように開ループプロットによるグラフが表示されます。

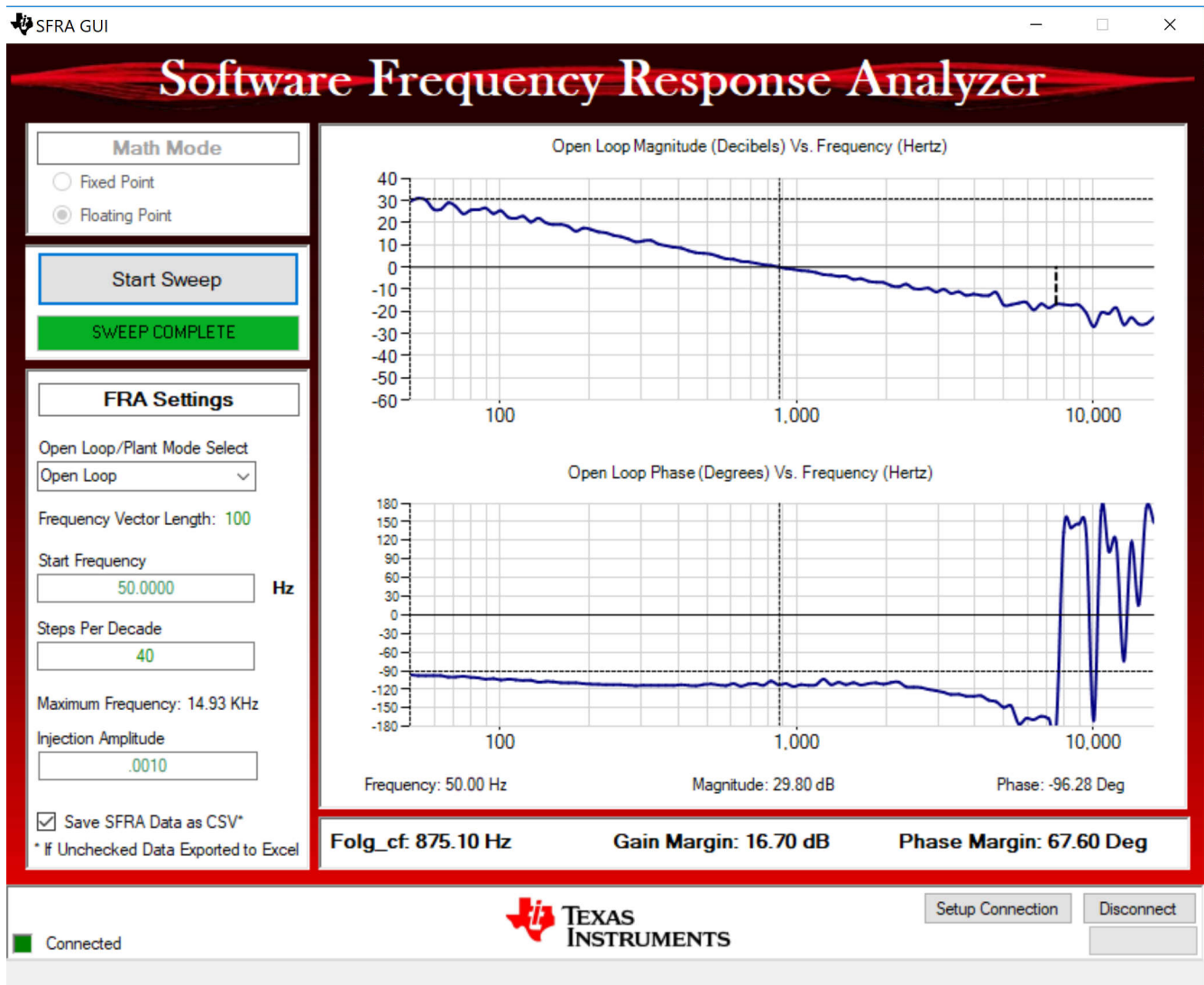





図 5-22. 閉電圧ループに対する SFRA 開ループプロット (Vprim 400V、Vsec 300V、電力 1.972kW、出力に抵抗性負荷あり)

また、周波数応答データは SFRA データフォルダ下のプロジェクトフォルダに保存され、SFRA 実行時のタイムスタンプが記録されます。

このデータは設計した補償器とほぼ一致していますが、開ループでの測定は、小信号の導入によってコンバータの DC ポイントがドリフトする可能性があり、それによって誤差の影響を受けやすいため、偏差が生じるのは当然のことです。

さまざまな電圧で SFRA をテストし、システムが動作可能な範囲全体で安定していることを確認します。

4. これにより、電圧ループの設計を検証できます。
5. システムを安全に停止させるには、入力 VPRIM 電圧をゼロまで下げます。[Watch] ウィンドウの電圧と電流がゼロに下がるのを観測します。
6. リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの

[Halt] ボタン () を使用するか、[Target] → [Halt] の順にクリックして、プロセッサを停止します。次に、  をクリックして、マイクロコントローラをリアルタイム モードから解除します。最後に、マイクロコントローラ () をリセットします。

7. [Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグセッションを終了します。

5.2.2.4 ラボ 4.1 次側から 2 次側への電力フロー、閉電流ループ チェック (2 次側に抵抗性負荷が接続されている状態)

このラボでは、出力電流制御ループは閉じています。このビルドのソフトウェア構成図を 図 1-1 に示します。ハードウェアは、図 1-1 に示すように設定されているものとします。

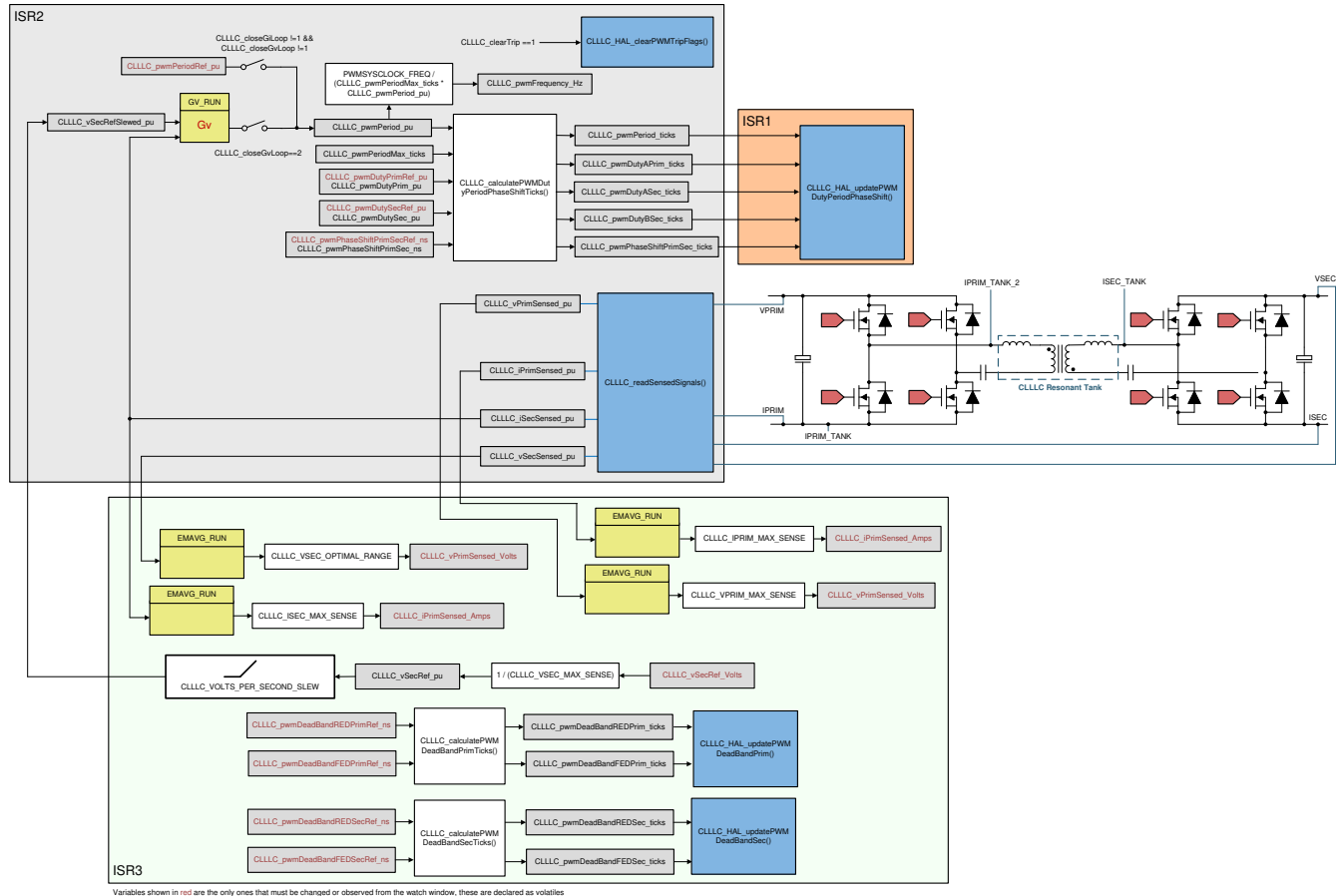


図 5-23. ラボ 4 の制御ソフトウェア構成図: 閉電流ループ

5.2.2.4.1 ラボ 4 のソフトウェアオプションの設定

1. <install Directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\CompDesigner.exe を開きます。
2. Compensation Designer が起動し、有効な SFRA データファイルを選択するように促されます。電流ループに対するラボ 1 の実行から Compensation Designer に SFRA データをインポートし、2 極、2 ゼロの補償器を設計します。ループが閉じたときにシステムが安定するように、設計のこの段階でより多くのマージンを確保しておくことをお勧めします。SFRA のさまざまな実行から得られたプラント データをチェックして、あらゆる条件下で安定したシステムを得ることができます。

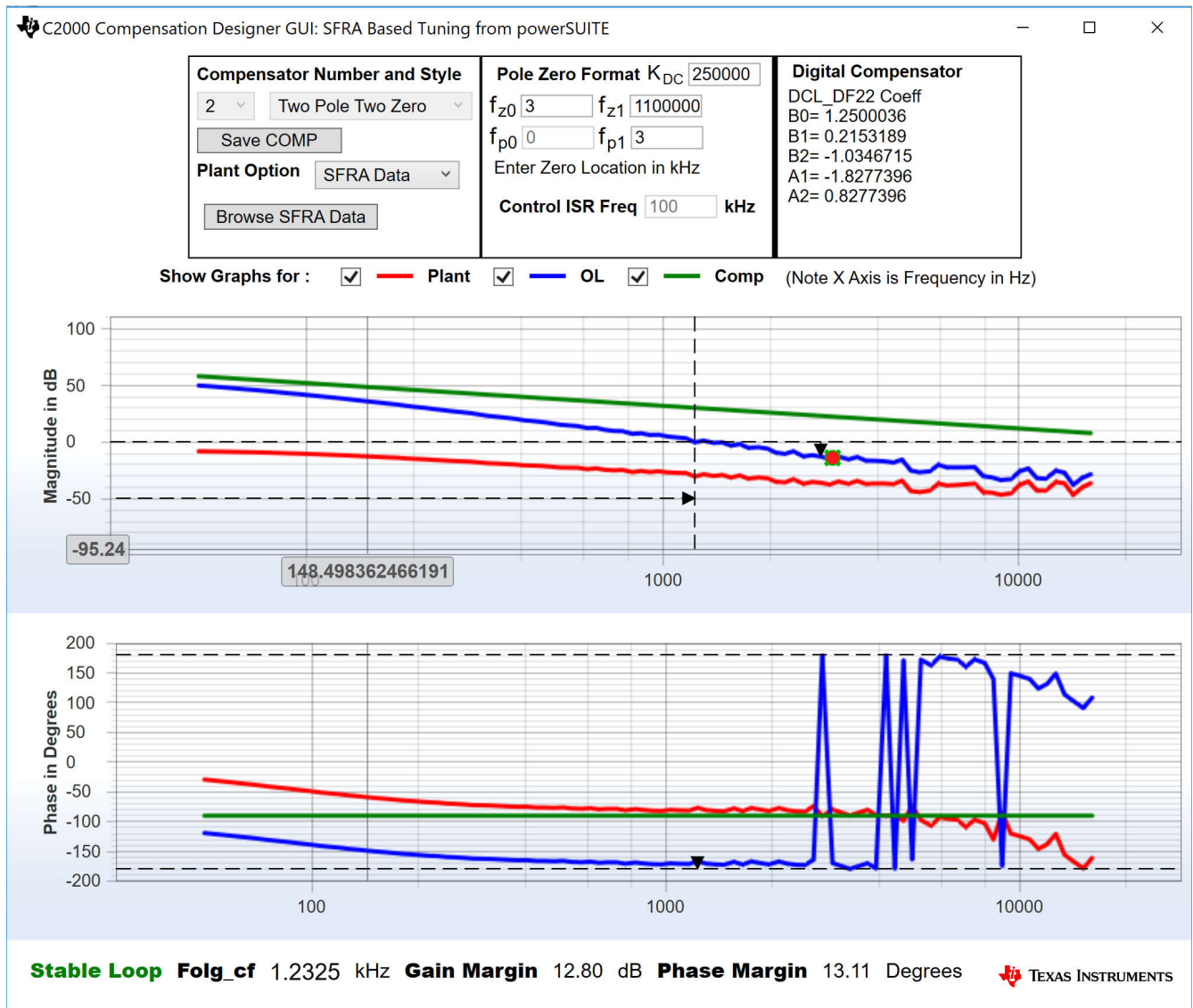


図 5-24. 電流ループの SFRA ベースのプラント測定を用いた補償器の設計、ラボ 4

注


調整は DF22 方式で行われますが、DF13 はソフトウェアで実行します。これは、DF22 構造ではできないのですが、DF13 ではソフトスタートが簡単なためです。どちらの場合も係数は同じです。本書の作成時点では、DF12 構造は DCL では使用できません。

3. 補償器の設計が完了したら、CLLLC_settings.h ファイルで補償器の値を更新できます。
4. Compensation Designer を閉じます。


5. このビルドでは、**settings.h** ファイルに以下の定義が設定されています。

```
#if CLLLC_LAB == 4 #define CLLLC_CONTROL_RUNNING_ON CLA_CORE #define CLLLC_POWER_FLOW
CLLLC_POWER_FLOW_PRIM_SEC #define CLLLC_INCR_BUILD CLLLC_CLOSED_LOOP_BUILD #define
CLLLC_CONTROL_MODE CLLLC_CURRENT_MODE #define CLLLC_TEST_SETUP CLLLC_TEST_SETUP_RES_LOAD
#define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED #if CLLLC_SFRA_ALLOWED == 1 #define
CLLLC_SFRA_TYPE CLLLC_SFRA_CURRENT #else #define CLLLC_SFRA_TYPE CLLLC_SFRA_DISABLED #endif
#define CLLLC_SFRA_AMPLITUDE (float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL1 #endif
```

5.2.2.4.2 プロジェクトのビルドおよびロードとデバッグの設定

- プロジェクト名を右クリックし、[Rebuild Project] をクリックします。
- プロジェクトが正常にビルドされます。
- [Run] → [Debug] をクリックしてデバッグ セッションを開始します。デュアル CPU デバイスの場合、デバッグを実行する CPU を選択するウィンドウが表示されます。ここでは、CPU1 を選択します。
- プロジェクトがデバイスにロードされ、CCS デバッグ ビューが有効になります。コードはメイン ルーチンの開始で停止します。
- [Watch] および [Expressions] ウィンドウに変数を追加するには、[View] → [Scripting Console] をクリックして、[Scripting Console] ダイアログ ボックスを開きます。このコンソールの右上隅で、[Open] をクリックして、プロジェクトフォルダ内にある **setupdebugenv_build4c.js** スクリプト ファイルを参照します。このファイルにより、[Watch] ウィンドウに、システムをデバッグするのに必要な適切な変数が入力されます。
- [Watch] ウィンドウで [Continuous Refresh] ボタン (🔄) をクリックして、コントローラからの値の連続更新を有効にします。
- マウス ポインタを水平ツールバーのボタンの上に置き、 **Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running)** ボタンをクリックして、リアルタイム モードを有効にします。

5.2.2.4.3 コードの実行

-  をクリックしてプロジェクトを実行します。
- CLLLC_clearTrip 変数に 1 を書き込み、検出をクリアします。CLLLC_closeGvLoop 変数がまだ 0 に設定されていないため、コンバータは開ループで動作します。ファームウェアにはソフトスタートが実装されていないため、まず 1 次側と 2 次側の電圧を手動でソフトスタートします。
- Watch ビューで、CLLLC_vPrimSensed_Volts、CLLLC_iPrimSensed_Amps、CLLLC_vSecSensed_Volts、CLLLC_iSecSensed_Amps の各変数が定期的に更新されているかどうかチェックします。(注:この時点では電力が印加されていないため、これらの値はゼロに近くなっています。)
- 入力 PRIM DC 電圧を 0V から 400V に徐々に上げて、コンバータをソフトスタートします。CLLLC_vPrimSensed_Volts に VPRIM の正しい値 (400V に近い値) が表示されていることを確認します。
- デフォルトでは、CLLLC_pwmPeriodRef_pu 変数は 0.6 (500.8kHz) に設定されています。これは、コンバータの直列共振周波数に近いですが、実際のハードウェアに搭載されている部品のばらつきが原因で、直列共振周波数より低くなったり、高くなったりします。
- 400V の 1 次側入力 で、巻線比が 1.33 の場合、CLLLC_vSecSensed_Volts 変数は 300V 近くになります。また、テスト条件で指定されている負荷の場合、負荷は 6.5A 近くになります。CLLLC_iSecRef_Amps 変数を 6.5A に設定します。何らかの理由で測定電流が 6.5A と異なる場合は、Ref を測定値に近い値に設定してください。ソフトウェアにはソフトスタートがないため、この基準を動作ポイントに近づけることが重要です。
- CLLLC_closeGvLoop 変数を 1 に設定します。これで電流ループが閉じて、コントローラによる電流制御が試行されます。
- CLLLC_iSecRef_Amps を 6.3A から 6.8A に変化させて、閉ループ動作をテストします。出力に抵抗性負荷が接続されており、その電圧はバッテリーよりも電流に対して大きく変化するため、ユーザーは電流を大きく変化させることはできません。このような急激な電圧上昇によって、コンバータはすぐに固定された VPRIM の制御可能範囲を超える可能性があります。狭い範囲内であれば、電流トラッキングを確認できます。

5.2.2.4.4 閉電流ループに対する SFRA の測定

1. このビルドのソフトウェアには SFRA が統合されているため、ハードウェアを測定して、設計した補償器が十分なゲイン マージンと位相マージンを提供していることを検証できます。SFRA を実行するには、プロジェクトを実行したまま、<Install directory >\C2000Ware_DigitalPower_SDK_<version>\libraries\sfragui\SFRA_GUI.exe に移動します。SFRA GUI が表示されます。
2. SFRA GUI でデバイスのオプションを選択します。たとえば、F280039 の場合は浮動小数点を選択します。[Setup Connection] をクリックします。ポップアップ ウィンドウで [Boot on Connect] オプションを選択解除し、適切な COM ポートを選択して、[OK] をクリックします。SFRA GUI に戻り、[Connect] をクリックします。
3. SFRA GUI がデバイスに接続します。これで [Start Sweep] をクリックして、SFRA 掃引を開始できるようになりました。SFRA 掃引が完了するまでには数分かかります。SFRA GUI のプログレス バーを確認したり、UART の動作を示す制御カード裏面の青色 LED の点滅をチェックすることで、動作を監視できます。終了すると、[図 1-1](#) のように開ループプロットによるグラフが表示されます。

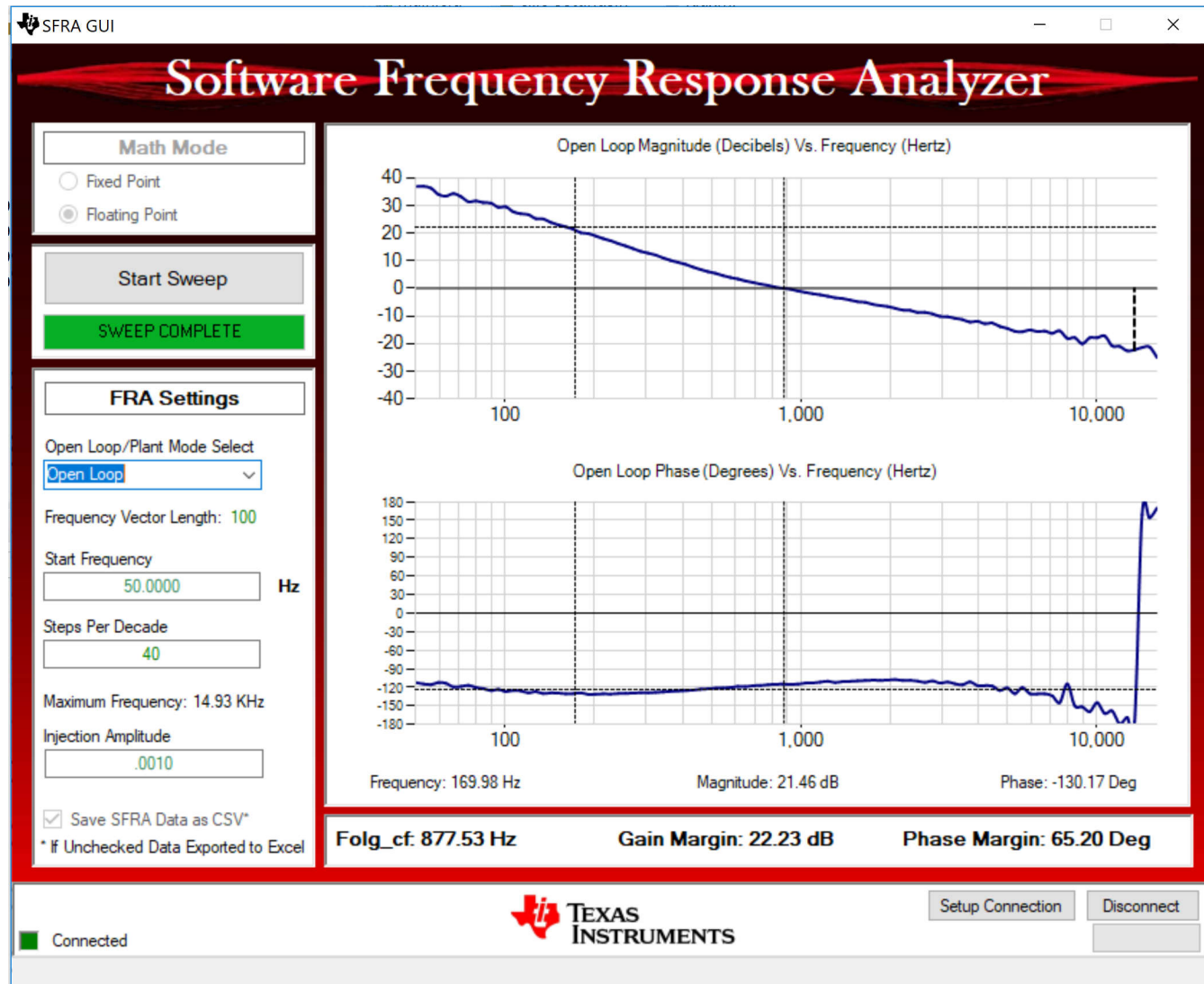





図 5-25. 閉電流ループに対する SFRA 開ループプロット (V_{prim} 400V、 V_{sec} 300V、電力 1.972kW、ラボ 4)

また、周波数応答データは SFRA データフォルダ下のプロジェクトフォルダに保存され、SFRA 実行時のタイムスタンプが記録されます。

このデータは設計した補償器とほぼ一致していますが、開ループでの測定は、小信号の導入によってコンバータの DC ポイントがドリフトする可能性があり、それによって誤差の影響を受けやすいため、偏差が生じるのは当然のことです。

さまざまな電流設定ポイントで SFRA をテストし、周期が制限されていないことを確認して、システムが動作可能な範囲全体で安定していることを確認します。

4. これにより、ラボ 4 の電流ループの設計を検証できます。
5. システムを安全に停止させるには、入力 VPRIM 電圧をゼロまで下げます。[Watch] ウィンドウの電圧と電流がゼロに下がるのを観測します。
6. リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの [Halt] ボタン () を使用するか、[Target] → [Halt] の順にクリックして、プロセッサを停止します。次に、  をクリックして、マイクロコントローラをリアルタイム モードから解除します。最後に、マイクロコントローラ () をリセットします。
7. [Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグセッションを終了します。

5.2.2.5 ラボ 5.1 次側から 2 次側への電力フロー、閉電流ループ チェック (2 次側で抵抗性負荷が電圧源と並列に接続されてバッテリー接続をエミュレートしている状態)

このラボでは、出力電流制御ループが閉じており、2 次側で抵抗性負荷が電圧源と並列に接続され、バッテリー接続をエミュレートしています。ハードウェアは、[図 1-1](#) に示すように設定されているものとします。このビルドのソフトウェア構成図を [図 1-1](#) に示します。

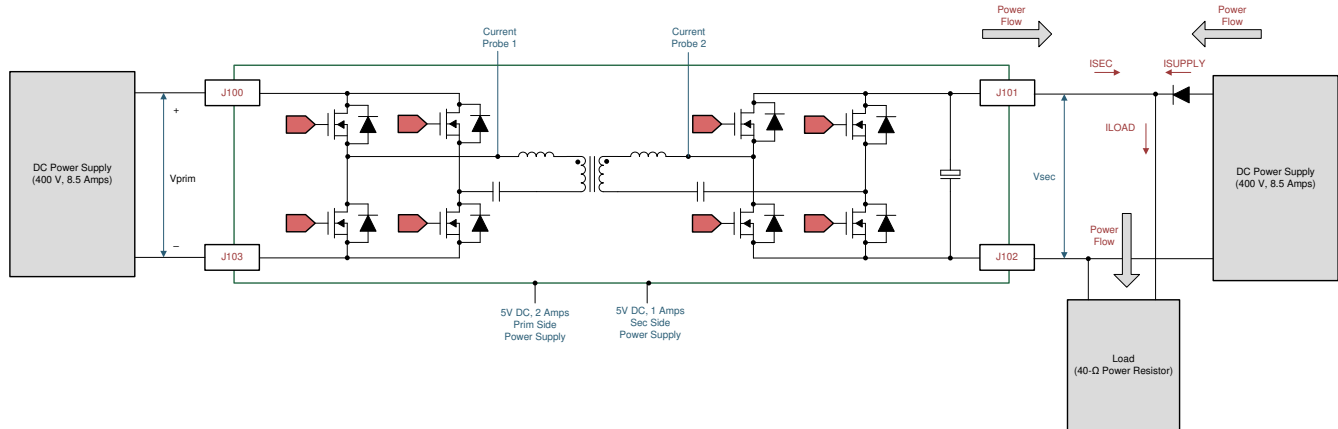


図 5-26. ラボ 5 のハードウェア セットアップ

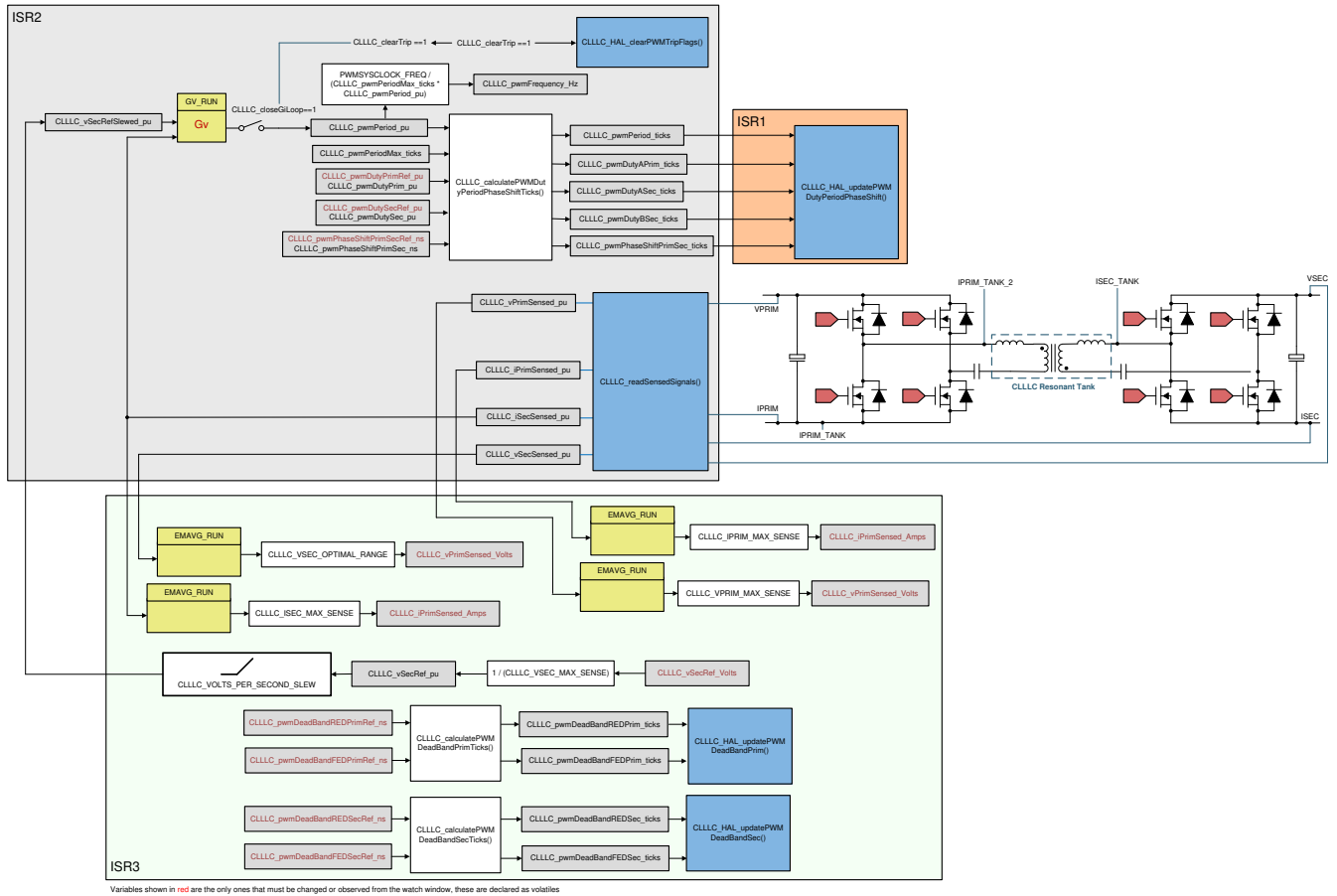


図 5-27. ラボ 5 のソフトウェア構成図

5.2.2.5.1 ラボ5 のソフトウェア オプションの設定

1. <install Directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\CompDesigner.exe を開きます。

5.2.2.5.2 電流ループ補償器の設計

1. Compensation Designer が起動します。現在のところ、数学モデルが利用できないため、この基板で行われた調整を利用して、以下のような補償が設計されています。バッテリー エミュレーション モードのプラントではゲインが大きくなるため、それに対応するために係数を下げる必要があります。図 1-1 に、このラボでこのデザインに使用されている各係数を示します。

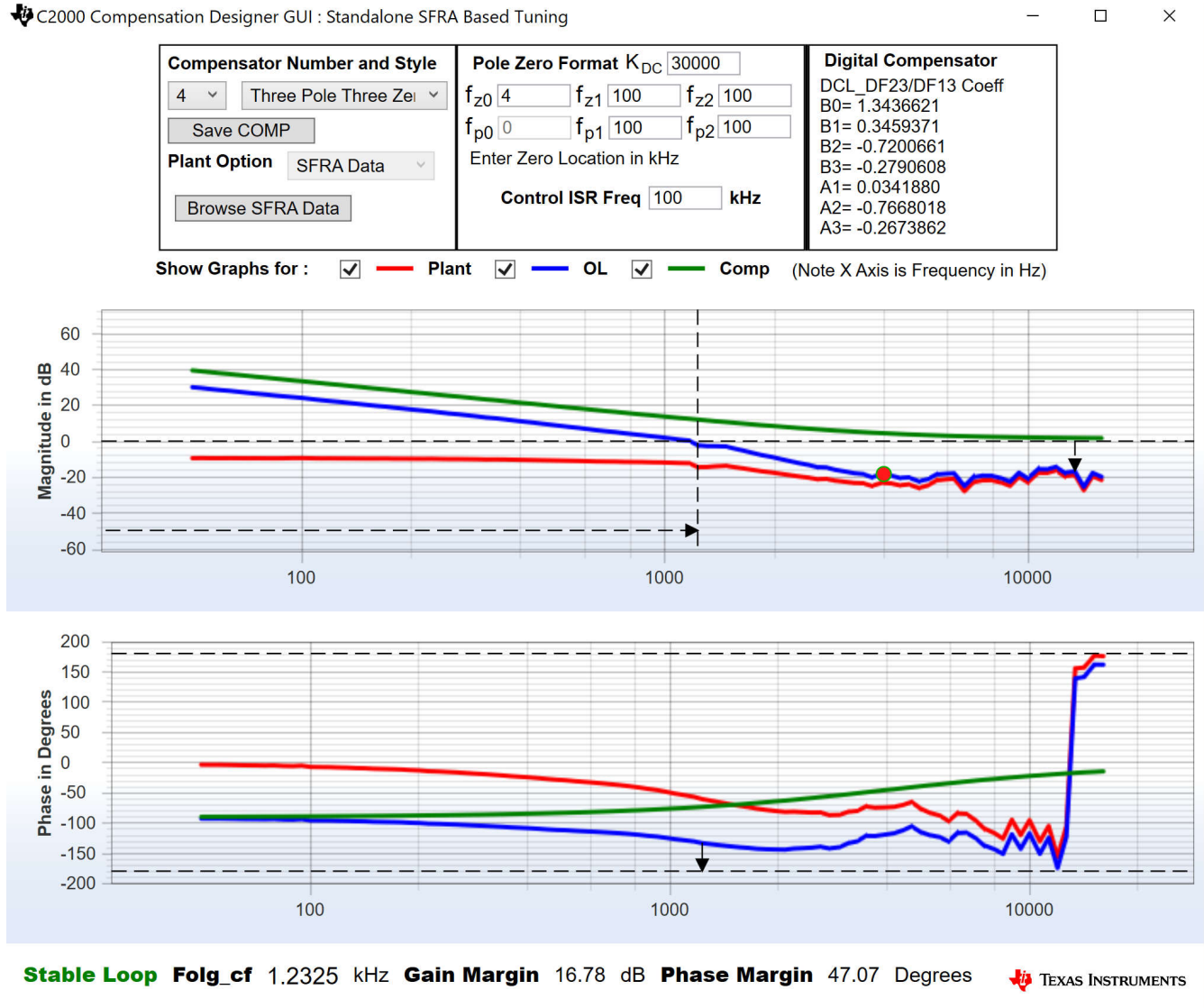



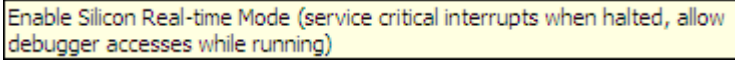
図 5-28. ラボ5、Compensation Designer

2. 補償器の設計が完了したら、CLLLC_settings.h ファイルで補償器の値を更新できます。係数は控えめに、ラボ 3 で使用したものよりもはるかに低くするのが最善です。
3. Compensation Designer を閉じます。
4. このビルドでは、settings.h ファイルに以下の定義が設定されています。


```
#if CLLLC_LAB == 5 #define CLLLC_CONTROL_RUNNING_ON 1 #define CLLLC_POWER_FLOW
CLLLC_POWER_FLOW_PRIM_SEC #define CLLLC_INCR_BUILD CLLLC_CLOSED_LOOP_BUILD #define
CLLLC_CONTROL_MODE CLLLC_CURRENT_MODE #define CLLLC_TEST_SETUP
CLLLC_TEST_SETUP_EMULATED_BATTERY #define CLLLC_PROTECTION CLLLC_PROTECTION_ENABLED #if
```

```
CLLLC_SFRA_ALLOWED == 1 #define CLLLC_SFRA_TYPE CLLLC_SFRA_CURRENT #else #define  
CLLLC_SFRA_TYPE CLLLC_SFRA_DISABLED #endif #define CLLLC_SFRA_AMPLITUDE  
(float32_t)CLLLC_SFRA_INJECTION_AMPLITUDE_LEVEL1 #endif
```

5.2.2.5.3 プロジェクトのビルドおよびロードとデバッグの設定

1. プロジェクト名を右クリックし、[Rebuild Project] をクリックします。
2. プロジェクトが正常にビルドされます。
3. [Run] → [Debug] をクリックしてデバッグ セッションを開始します。デュアル CPU デバイスの場合、デバッグを実行する CPU を選択するウィンドウが表示されます。ここでは、CPU1 を選択します。
4. プロジェクトがデバイスにロードされ、CCS デバッグ ビューが有効になります。コードはメイン ルーチンの開始で停止します。
5. [Watch] および [Expressions] ウィンドウに変数を追加するには、[View] → [Scripting Console] をクリックして、[Scripting Console] ダイアログ ボックスを開きます。このコンソールの右上隅で、[Open] をクリックして、プロジェクトフォルダ内にある setupdebugenv_build4.js スクリプト ファイルを参照します。このファイルにより、[Watch] ウィンドウに、システムをデバッグするのに必要な適切な変数が入力されます。
6. [Watch] ウィンドウで [Continuous Refresh] ボタン (🔄) をクリックして、コントローラからの値の連続更新を有効にします。
7. マウス ポインタを水平ツールバーのボタンの上に置き、  ボタンをクリックして、リアルタイム モードを有効にします。

5.2.2.5.4 コードの実行

1.  をクリックしてプロジェクトを実行します。
2. 入力 PRIM DC 電圧を 0V から 400V に徐々に上げます。CLLLC_vPrimSensed_Volts に VPRIM の正しい値 (400V に近い値) が表示されていることを確認します。この時点で PWM がトリップし、1 次側からは電流が流れません。
3. VSEC を 300V に上げます。2 次側に接続されている電源からすべての電流が負荷に流れ込み、6.5A 近くになります。
4. CLLLC_iSecRef_Amps 変数を 0.1A に設定します。
5. CLLLC_clearTrip 変数に 1 を書き込み、検出をクリアします。このラボのソフトウェアによって CLLLC_closeGiLoop 変数が自動的に 1 に設定されます。
6. 固定された 1 次側電圧での電圧範囲が狭いことが原因で、コンバータは最も高い周波数まで飽和し、ISEC での引き込み電流は 0.1A を上回ります。ユーザーは、CLLLC_pwmFrequency_Hz 変数を監視することでこの状態を観察することができます。CLLLC_pwmFrequency_Hz 変数は、飽和上限時には 800kHz 近くなり、飽和下限時に 200kHz 近くなります。
7. 電流を徐々に 2A~3A に上げます。これで、電流は 2 次側に接続された電圧源とテスト中デバイス (DUT) とで共有されます。

5.2.2.5.5 バッテリ エミュレーション モードでの閉電流ループに対する SFRA 測定

1. このビルドのソフトウェアには SFRA が統合されているため、ハードウェアを測定して、設計した補償器が十分なゲイン マージンと位相マージンを提供していることを検証できます。SFRA を実行するには、プロジェクトを実行したまま、<Install directory >\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\SFRA_GUI.exe に移動します。SFRA GUI が表示されます。
2. SFRA GUI でデバイスのオプションを選択します。たとえば、F280039 の場合は浮動小数点を選択します。[Setup Connection] をクリックします。ポップアップ ウィンドウで [Boot on Connect] オプションを選択解除し、適切な COM ポートを選択して、[OK] をクリックします。SFRA GUI に戻り、[Connect] をクリックします。
3. SFRA GUI がデバイスに接続します。これで [Start Sweep] をクリックして、SFRA 掃引を開始できるようになりました。SFRA 掃引が完了するまでには数分かかります。SFRA GUI のプログレス バーを確認したり、UART の動作を示す制御カード裏面の青色 LED の点滅をチェックすることで、動作を監視できます。終了すると、[図 1-1](#) のように開ループ プロットによるグラフが表示されます。

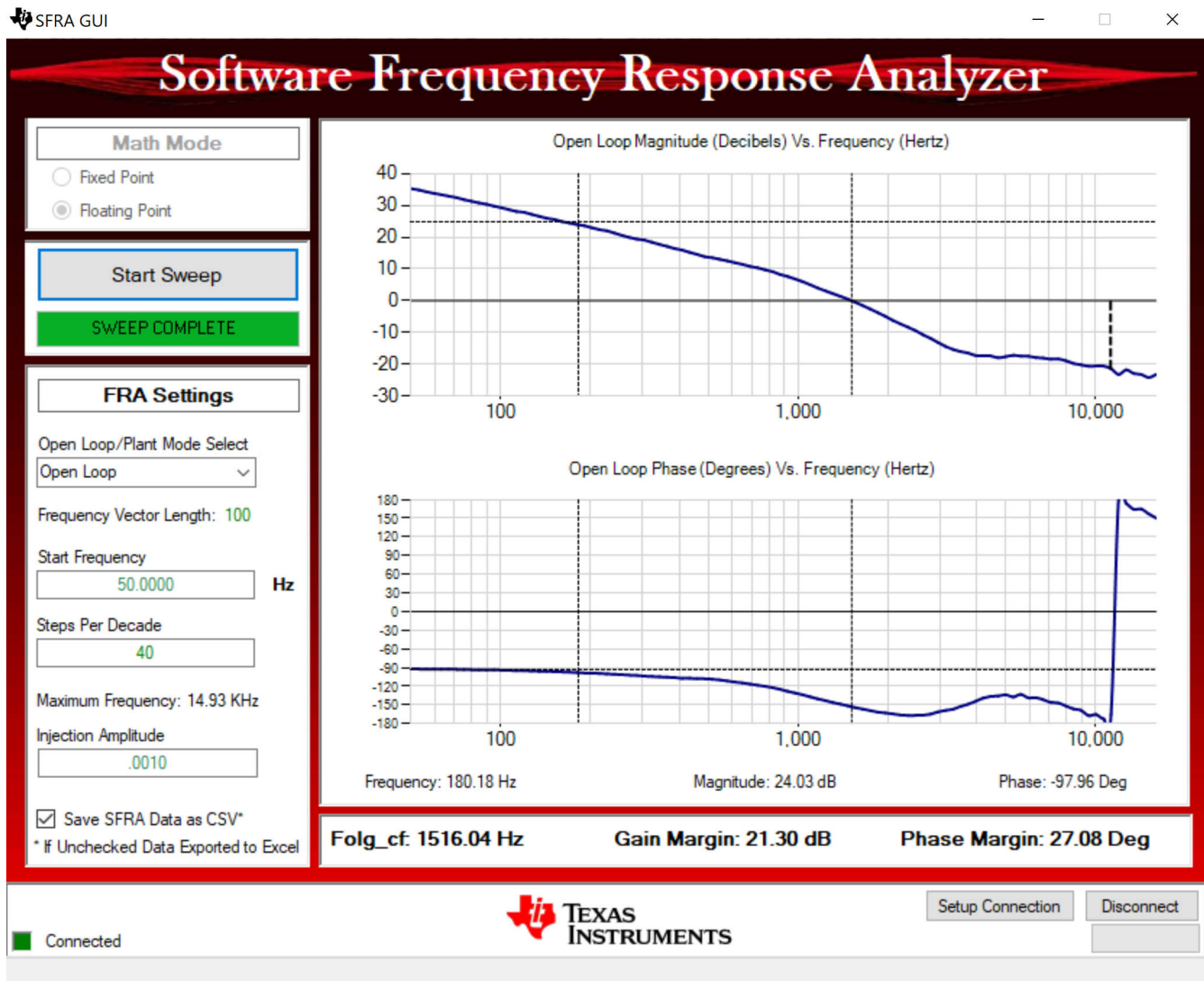





図 5-29. バッテリ接続をエミュレートした状態の閉電流ループに対する SFRA 開ループプロット (Vprim 400V、Vsec 300V、電力 1.972kW、ラボ 5)

また、周波数応答データは SFRA データフォルダ下のプロジェクトフォルダに保存され、SFRA 実行時のタイムスタンプが記録されます。

さまざまな電流設定ポイントで SFRA をテストし、周期が制限されていないことを確認して、システムが動作可能な範囲全体で安定していることを確認します。

4. これにより、ラボ 5 の電流ループ設計を検証できます。
5. システムを安全に停止させるには、入力 VPRIM 電圧をゼロまで下げます。[Watch] ウィンドウの電圧と電流がゼロに下がるのを観測します。
6. リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの [Halt] ボタン () を使用するか、[Target] → [Halt] の順にクリックして、プロセッサを停止します。次に、  をクリックして、マイクロコントローラをリアルタイム モードから解除します。最後に、マイクロコントローラ () をリセットします。
7. [Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグセッションを終了します。

5.2.3 TTPLPFC のテスト手順

5.2.3.1 ラボ 1: 開ループ、DC

このビルドでは、固定デューティ サイクルにより基板は開ループ方式で励起されます。デューティ サイクルは `dutyPU_DC` 変数で制御されます。このビルドでは、電力段からの帰還値のセンシングと PWM ゲートドライバの動作を検証し、ハードウェアに問題がないことを確認します。また、このビルドでは入出力電圧センシングの較正も実行できます。このビルドのソフトウェア構造を [図 1-1](#) に示します。このシステムには、電流ループ用の高速 ISR、電圧ループおよび計測機能を実行する低速 ISR という 2 つの ISR があります。各 ISR で実行されるモジュールを [図 1-1](#) に示します (TIDM-02013 は 2 相インターリーブ TTPPLPFC であることに注意してください)。

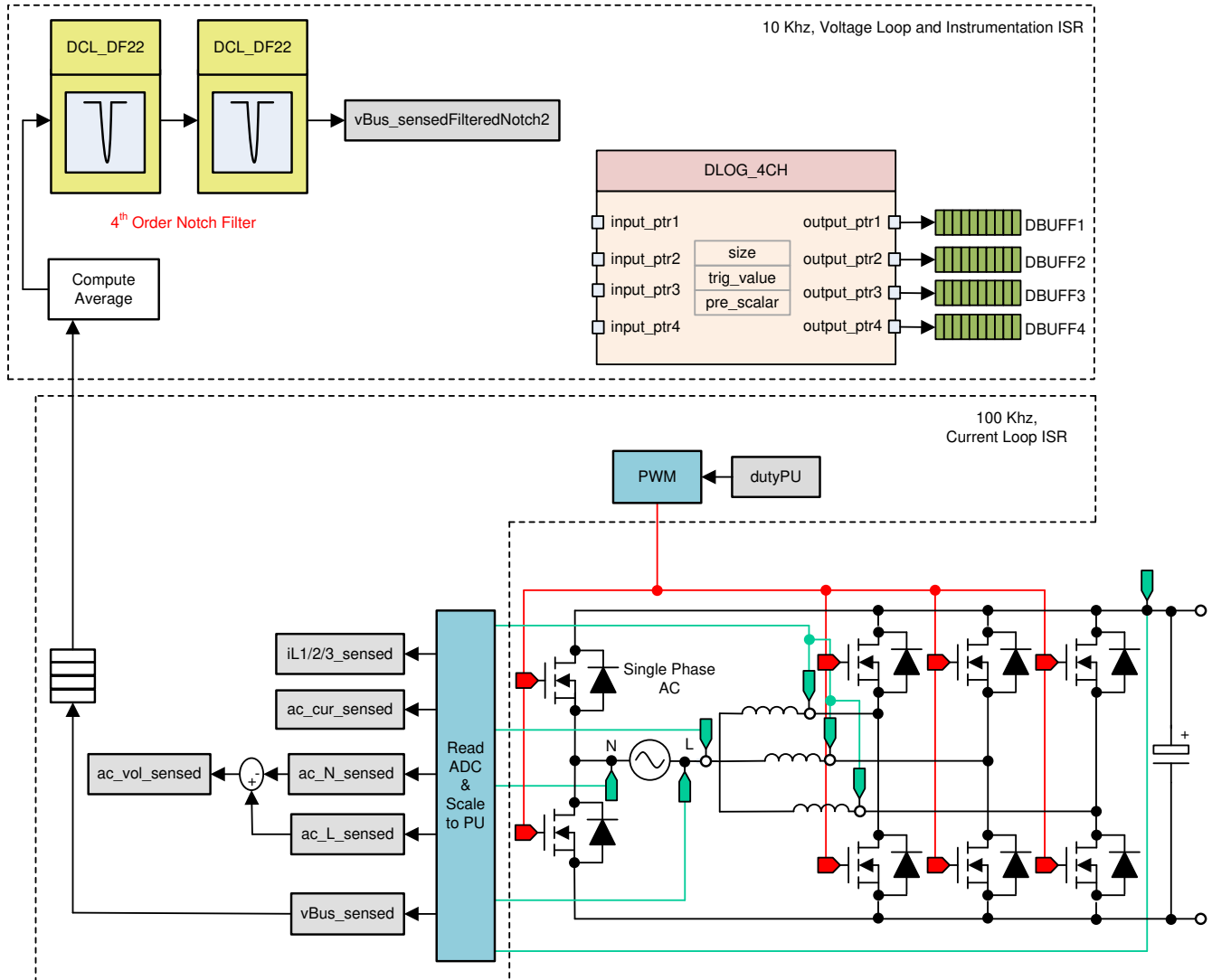


図 5-30. ビルドレベル 1 の制御ソフトウェア構成図: 開ループ プロジェクト

5.2.3.1.1 BUILD 1 のソフトウェアオプションの設定

TTPLPFC_settings.h を開き、Lab1 を有効にします。

```
#define TTPLPFC_LAB 1
```

5.2.3.1.2 プロジェクトのビルドおよびロード

プロジェクト名を右クリックし、[Rebuild Project] をクリックします。

プロジェクトが正常にビルドされます。

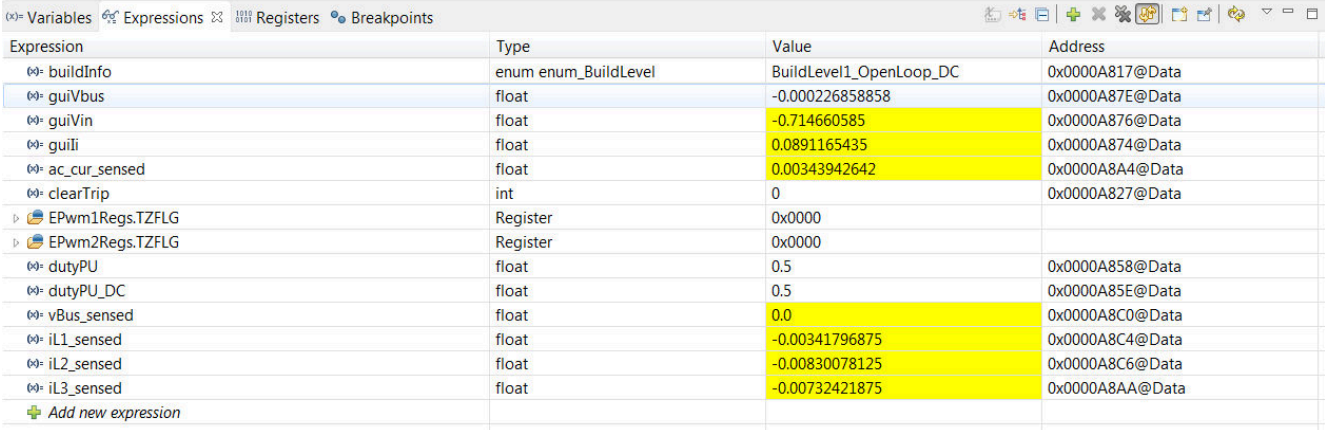
Project Explorer で、targetConfigs 下で適切な目標構成ファイルが有効になっていることを確認します。

[Run] → [Debug] をクリックします。この操作によりデバッグ セッションが起動します。デュアル CPU デバイスの場合には、ウィンドウが表示され、デバッグを実行する必要がある CPU を選択できます。ここでは、CPU1 を選択します。

するとプロジェクトがデバイスにロードされ、CCS デバッグ ビューが有効になります。メイン ルーチンの開始時にコードは停止します。

5.2.3.1.3 デバッグ環境設定ウィンドウ

[Watch] および [Expressions] ウィンドウに変数を追加するには、[View] → [Scripting Console] をクリックして、[Scripting Console] ダイアログ ボックスを開きます。このコンソールの右上隅で、[Open] をクリックして、プロジェクトフォルダ内にある setupdebugenv_lab1.js スクリプト ファイルを参照します。このスクリプトファイルにより、Watch ウィンドウに、システムをデバッグするのに必要な適切な変数が入力されます。[Watch] ウィンドウで [Continuous Refresh] ボタンをクリックして、コントローラからの値の連続更新を有効にします。図 1-1 に示すように Watch ウィンドウが表示されます。



Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel1_OpenLoop_DC	0x0000A817@Data
guiVbus	float	-0.000226858858	0x0000A87E@Data
guiVin	float	-0.714660585	0x0000A876@Data
guiIi	float	0.0891165435	0x0000A874@Data
ac_cur_sensed	float	0.00343942642	0x0000A8A4@Data
clearTrip	int	0	0x0000A827@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	0.5	0x0000A858@Data
dutyPU_DC	float	0.5	0x0000A85E@Data
vBus_sensed	float	0.0	0x0000A8C0@Data
iL1_sensed	float	-0.00341796875	0x0000A8C4@Data
iL2_sensed	float	-0.00830078125	0x0000A8C6@Data
iL3_sensed	float	-0.00732421875	0x0000A8AA@Data
+ Add new expression			

図 5-31. ビルドレベル 1 の Expressions ビュー

 をクリックしてプロジェクトを実行します。

ここでツールバーの [Halt] ボタン () をクリックして、プロセッサを停止します。

5.2.3.1.4 リアルタイム エミュレーションの使用


リアルタイム エミュレーションは、マイクロコントローラ動作中に CCS 内のウィンドウを更新できる特別なエミュレーション機能です。この機能により、グラフおよび Watch ビューが更新されるだけでなく、[Watch] ウィンドウや [Memory] ウィンドウで値を変更したり、プロセッサを停止することなくシステムにおける変更の反映を確認したりできます。

マウス ポインタを水平ツールバーのボタンの上に置き、 ボタンをクリックして、リアルタイム モードを有効にします。

Enable Silicon Real-time Mode (service critical interrupts when halted, allow debugger accesses while running)

メッセージ ボックスが表示されることがあります。その場合は [YES] を選択して、デバッグ イベントを有効にします。この操作により、ステータスレジスタ 1 (ST1) のビット 1 (DGBM ビット) が「0」に設定されます。DGBM は、デバッグ イネーブル マスク ビットです。DGBM ビットが「0」に設定されると、メモリ値とレジスタ値がホストプロセッサに渡されて、デバッグのウィンドウが更新できるようになります。

5.2.3.1.5 コードの実行

 をクリックしてプロジェクトを再度実行します。

数秒後、突入リレーのクリック音がします。DC によりこのビルドレベルでそれを実行するようにソフトウェアがプログラミングされます。検出がクリアされ、デューティサイクル 0.5 が適用されます。

Watch ビューで、guiVin、guiVbus、guili の各変数が定期的に更新されているかどうかチェックします。電力が印加されていないため、この値はゼロに近くなっています。

入力 DC 電圧をゼロから 120V まで徐々に上げていきます。デフォルト設定で 0.5PU の安定したデューティサイクルが適用されるため、出力電圧は昇圧を示します。大電流が引き出された場合、電圧端子が逆接続されていないかどうかを確認します。そうであれば、まず電圧をゼロに下げ、問題を修正してからテストを再開します。

TTPLPFC_vBusAvg_pu に正しい値が表示されていることを確認して、電圧センシングを検証します。DC 120V 入力の場合、これにより、基板の電圧検出をある程度検証できます。

dutyPU_DC 変数を変更して、さまざまな昇圧条件下での動作を確認できます。これにより、PWM ドライバとハードウェアの接続を基本的なレベルで検証できます。




終了したら、入力電圧をゼロに下げ、バス電圧がゼロまで下がるのを見届けます。


これで、このビルドのチェックは完了です。このビルドが正常に完了すると、以下の項目が検証されます。

- 電圧と電流のセンシングとスケーリングの精度
- 電流ループ ISR および電圧ループ計測 ISR における BUILD 1 コードの割り込み生成および実行
- PWM ドライバおよびスイッチング

問題が確認された場合には、ビルドの問題など解消するためにハードウェアを慎重に点検する必要があります。

これでコントローラを停止し、デバッグ接続を終了できます。

リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの [Halt] ボタン () を使用するか [Target] → [Halt] の順にクリックしてプロセッサを停止します。次に、  をクリックして、マイクロコントローラをリアルタイム モードから解除します。最後に、  をクリックしてマイクロコントローラをリセットします。

[Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグ セッションを終了します。

5.2.3.2 ラボ 2: 閉電流ループ DC

BUILD 2 では、内部電流ループが閉じているため、電流補償器 Gi を使用してインダクタ電流を制御します。DC バスと出力電圧フィード フォワードの両方をこの電流補償器の出力に印加して、インバータのデューティサイクルを生成します。これにより、電流補償器用構成が簡素になり、比例(P)コントローラを使用して内部電流のループを調整できます。電流ループのモデルが導き出されています。このビルドのソフトウェア構成図を [図 1-1](#) に示します (TIDM-02013 は 2 相インターリーブ TTPPLPFC であることに注意してください)。

$$\text{duty1PU} = \frac{(\text{ac_cur_meas} - \text{ac_cur_ref_inst}) \times G_i + \text{ac_vol_sensed}}{\text{vBus_sensed}} \quad (21)$$

このビルドのソフトウェア構成図を [図 1-1](#) に示します。

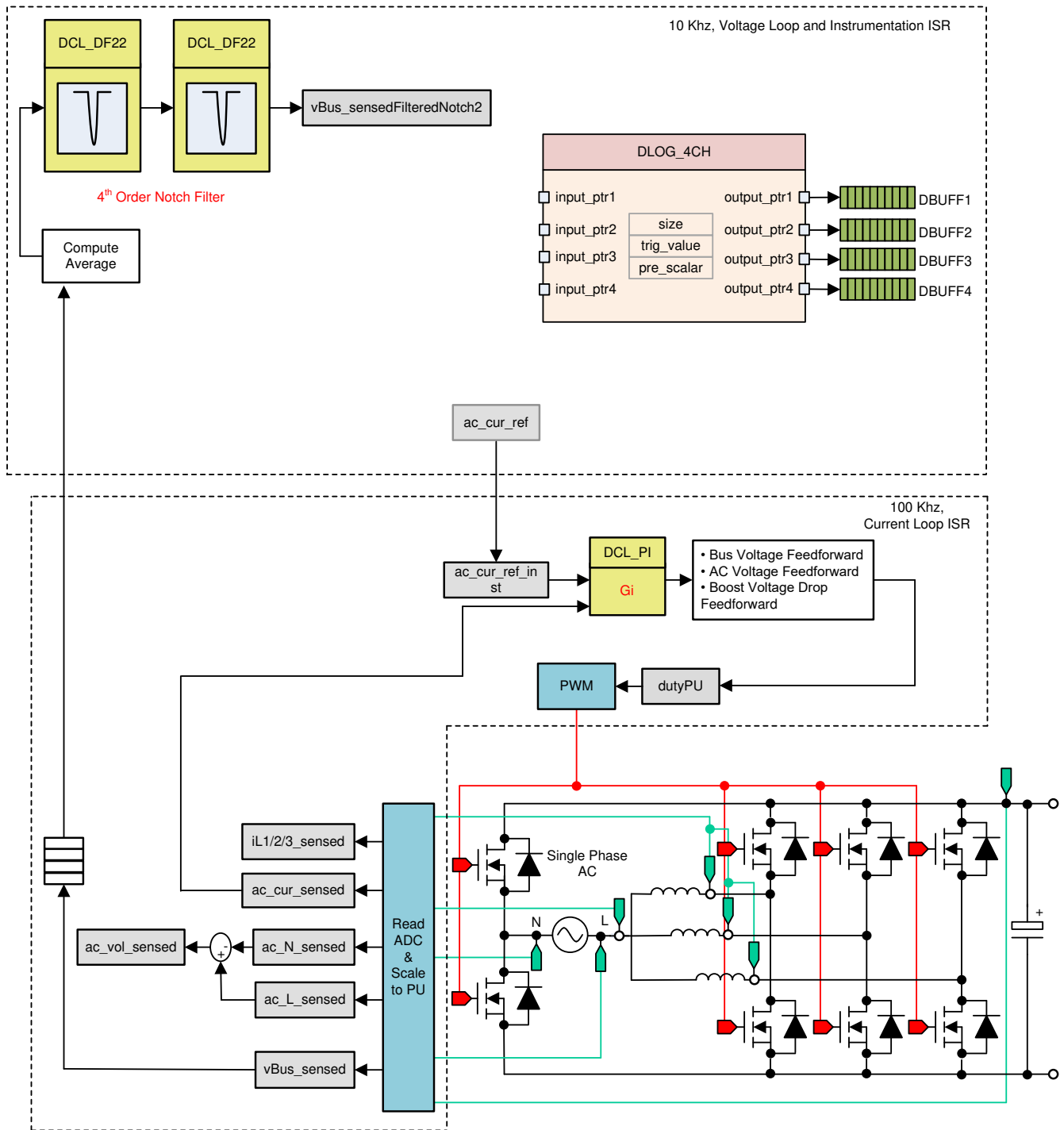


図 5-32. ビルドレベル 2 の制御ソフトウェア構成図: 閉電流ループ

5.2.3.2.1 BUILD 2 のソフトウェアオプションの設定

セクション 5.1.1 で説明されているように、ハードウェアがスタンドアローン PFC 動作用に設定されていることを確認します。高電圧(HV)電力はまだ基板に供給しないでください。

TTPLPFC_settings.h を開き、Lab2 を有効にします。

```
#define TTPLPFC_LAB 2
```

その他のすべてのオプションがセクション 5.2.3.2 で指定したものと同一であることを確認します。

1. Compensation Designer (<install Directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\CompDesigner.exe) を開きます。

5.2.3.2.2 電流ループ補償器の設計

Compensation Designer が起動します。PI コントローラを極零の観点から調整して、安定した閉ループ動作を確保できます。設計した補償器を使用するシステムの安定性は、図 1-1 に示す Compensation Designer の開ループ伝達関数プロットで利得マージンと位相マージンを観測して検証できます。

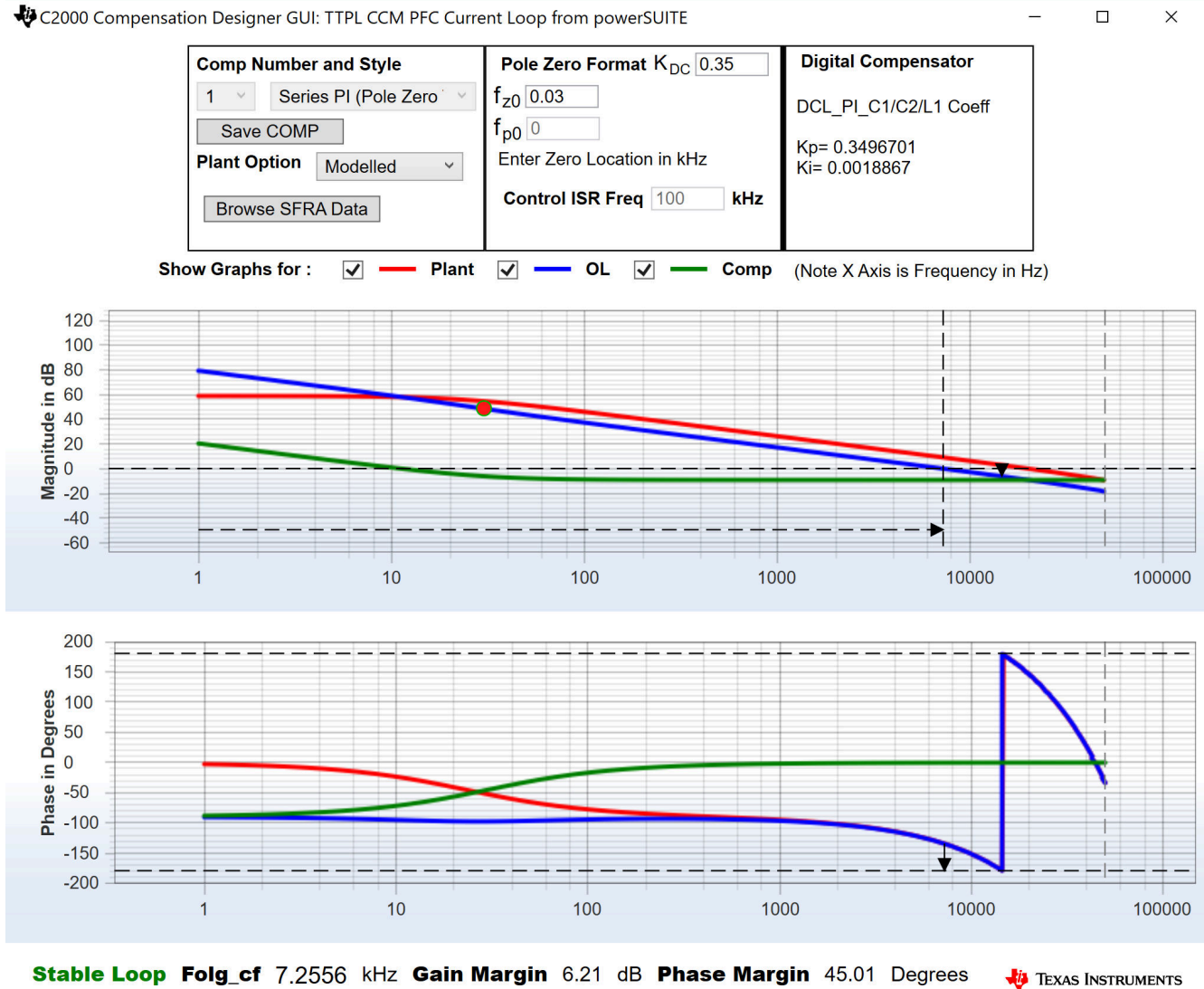



図 5-33. Compensation Designer を使用した電流ループ設計

開ループ ゲインが完了したら、ttlplfc_settings.h ファイルで補償器の値を更新できます。

Compensation Designer を閉じます。

5.2.3.2.3 プロジェクトのビルドおよびロードとデバッグの設定

プロジェクト名を右クリックし、[Rebuild Project] をクリックします。プロジェクトが正常にビルドされます。Run → Debug をクリックして、デバッグセッションを起動します。デュアル CPU デバイスの場合、ウィンドウが表示され、デバッグを実行する必要がある CPU を選択できます。ここでは、CPU1 を選択します。するとプロジェクトがデバイスにロードされ、CCS デバッグ ビューが有効になります。メイン ルーチンの開始時にコードは停止します。

[Watch] および [Expressions] ウィンドウに変数を追加するには、[View] → [Scripting Console] をクリックして、[Scripting Console] ダイアログ ボックスを開きます。このコンソールの右上隅で、[Open] をクリックして、プロジェクトフォルダ内にある `setupdebugenv_lab2.js` スクリプト ファイルを参照します。このファイルにより、[Watch] ウィンドウに、システムをデバッグするのに必要な適切な変数が入力されます。[Watch] ウィンドウで [Continuous Refresh] ボタン () をクリックして、コントローラからの値の連続更新を有効にします。 [図 1-1](#) のように [Watch] ウィンドウが表示されます。

Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel2_CurrentLoop_DC	0x0000A80C@Data
boardStatus	enum enum_boardStatus	boardStatus_Idle	0x0000A804@Data
clearTrip	int	0	0x0000A824@Data
closeGilLoop	int	0	0x0000A819@Data
ac_cur_ref	float	0.0299999993	0x0000A838@Data
ac_cur_sensed	float	0.0118730068	0x0000A8A4@Data
guiVbus	float	0.347434014	0x0000A846@Data
guiVin	float	-0.678055584	0x0000A86C@Data
guiIi	float	0.274688691	0x0000A86A@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
dutyPU	float	0.00999999978	0x0000A84E@Data
dutyPU_DC	float	0.5	0x0000A84C@Data
iL1_sensed	float	-0.00537109375	0x0000A8AE@Data
iL2_sensed	float	-0.00537109375	0x0000A8AC@Data
iL3_sensed	float	-0.00634765625	0x0000A8AA@Data
autoStartSlew	unsigned long	14	0x0000A87C@Data
+ Add new expression			

図 5-34. ビルド レベル 2: 閉電流ループ Expressions ビュー


マウス ポインタを水平ツールバーのボタンの上に置き、  ボタンをクリックして、リアルタイム モードを有効にします。



をクリックしてプロジェクトを実行します。

ここでツールバーの [Halt] ボタン () をクリックして、プロセッサを停止します。

5.2.3.2.4 コードの実行

このプロジェクトは、設定時間(`autoStartSlew==100`)を過ぎたら突入リレーを駆動し、検出をクリアするようにプログラミングされています。DC によりこのビルドレベルでそれを実行するようにソフトウェアがプログラミングされます。Run ボタンをクリックしてからこの `autoslew` カウンタが 100 に達するまでに、入力電圧を印加する必要があります。入力時に電圧を印加する前にカウンタが 100 に達した場合は、コードをリセットする必要があります。このためコントローラをリアルタイムモードから解除する必要があり、リセットを実行し、再起動します。 [セクション 5.2.3.2.3](#) のステップを繰り返し、マウス ポインタを水平ツールバーのボタンの上に置き、  ボタンをクリックして、リアルタイム モードを有効にします。



をクリックしてプロジェクトを実行します。

`autoStartSlew` が 100 に達する前に、約 50V の入力電圧を印加します。`autoStartSlew` が 100 に達するとすぐに、突入リレーが駆動され、PWM 検出がクリアされると同時に電流ループフラグを閉じます。

Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel2_CurrentLoop_DC	0x0000A80C@Data
boardStatus	enum enum_boardStatus	boardStatus_Idle	0x0000A804@Data
clearTrip	int	1	0x0000A824@Data
closeGilLoop	int	1	0x0000A819@Data
ac_cur_ref	float	0.0299999993	0x0000A838@Data
ac_cur_sensed	float	0.0300658941	0x0000A8A4@Data
guiVbus	float	127.377548	0x0000A846@Data
guiVin	float	48.3203316	0x0000A86C@Data
guiIi	float	0.707000256	0x0000A86A@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	0.386497527	0x0000A84E@Data
dutyPU_DC	float	0.5	0x0000A84C@Data
iL1_sensed	float	0.0107421875	0x0000A8AE@Data
iL2_sensed	float	0.0087890625	0x0000A8AC@Data
iL3_sensed	float	0.009765625	0x0000A8AA@Data
autoStartSlew	unsigned long	101	0x0000A87C@Data

図 5-35. Watch 式、ビルド ラボ 2、閉電流ループ動作開始後の DC

入力電流は約 1.5A に制御され、出力電圧は約 193V に上昇します。
 ac_cur_ref を 0.045 すなわち 2.4A 入力まで徐々に上げていきます。
 V_{in} を 120V に徐々に上げていくと、出力電圧が 370V を上回ります。

Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel2_CurrentLoop_DC	0x0000A80C@Data
boardStatus	enum enum_boardStatus	boardStatus_Idle	0x0000A804@Data
clearTrip	int	1	0x0000A824@Data
closeGilLoop	int	1	0x0000A819@Data
ac_cur_ref	float	0.100000001	0x0000A838@Data
ac_cur_sensed	float	0.0993705988	0x0000A8A4@Data
guiVbus	float	380.123596	0x0000A846@Data
guiVin	float	117.478661	0x0000A86C@Data
guiIi	float	2.46380639	0x0000A86A@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	0.308701962	0x0000A84E@Data
dutyPU_DC	float	0.5	0x0000A84C@Data
iL1_sensed	float	0.0493164063	0x0000A8AE@Data
iL2_sensed	float	0.052734375	0x0000A8AC@Data
iL3_sensed	float	0.0458984375	0x0000A8AA@Data
autoStartSlew	unsigned long	101	0x0000A87C@Data

図 5-36. Watch 式、ビルド ラボ 2、フル電圧での閉電流ループ動作開始後の DC

このビルドのソフトウェアには SFRA が統合されているため、ハードウェアを測定して、設計した補償器が十分なゲイン マージンと位相マージンを提供していることを検証できます。SFRA を実行するには、プロジェクトを実行したまま、<Install directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\SFRA_GUI.exe に移動します。SFRA GUI が表示されます。




SFRA GUI でデバイスのオプションを選択します。例として、F28003x の場合には浮動小数点を選択します。[Setup Connection] をクリックします。ポップアップ ウィンドウで [Boot on Connect] オプションを選択解除し、適切な COM ポートを選択します。[Boot on Connect] が選択解除されていることを確認します。[OK] をクリックします。SFRA GUI に戻り、[Connect] をクリックします。


SFRA GUI がデバイスに接続します。これで [Start Sweep] をクリックして、SFRA 掃引を開始できるようになりました。SFRA 掃引が完了するまでには数分かかります。SFRA GUI のプログレスバーを確認したり、UART の動作を示す制御カード裏面の青色 LED の点滅をチェックすることで、動作を監視できます。終了すると、開ループプロットによるグラフが表示されます。また、周波数応答データは SFRA データフォルダ下のプロジェクトフォルダに保存され、SFRA 実行時のタイムスタンプが記録されます。

さらに、プラントの周波数応答の測定値は、Compensation Designer(<install Directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\CompDesigner.exe) を使用して電流補償器を設計するために使用できます。

GUI でプラントオプションに SFRA Data を選択します。これは、測定したプラントの情報を用いて補償器を設計するものです。このオプションを使用して補償を微調整できます。デフォルトでは、Compensation Designer は最新の SFRA 実行を示しています。過去の SFRA 実行時のプラント情報を使用する必要がある場合、[Browse SFRA Data] をクリックし、参照して SFRADData.csv ファイルを選択してください。この操作により、電流補償器設計を検証できます。

入力 DC 電圧をゼロまで下げて、システムを安全に停止させます。guiVBus もゼロに下がっていることを確認します。

リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの [Halt] ボタン () を使用するか [Target] → [Halt] の順にクリックしてプロセッサを停止します。次に、  をクリックしてマイクロコントローラをリアルタイム モードから解除します。最後に、マイクロコントローラ () をリセットします。

[Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグセッションを終了します。

5.2.3.3 ラボ 3: 閉電流ループ、AC

ラボ 3 では、内部電流ループが閉じているため、電流補償器 G_i を使用してインダクタ電流を制御します。DC バスと出力電圧フィードフォワードの両方をこの電流補償器の出力に印加して、ゼロクロス周りの PWM ソフトスタートとともにインバータのデューティサイクルを生成します。

このビルドのソフトウェア構成図を [図 1-1](#) に示します (TIDM-02013 は 2 相インターリーブ TTPPLPFC であることに注意してください)。

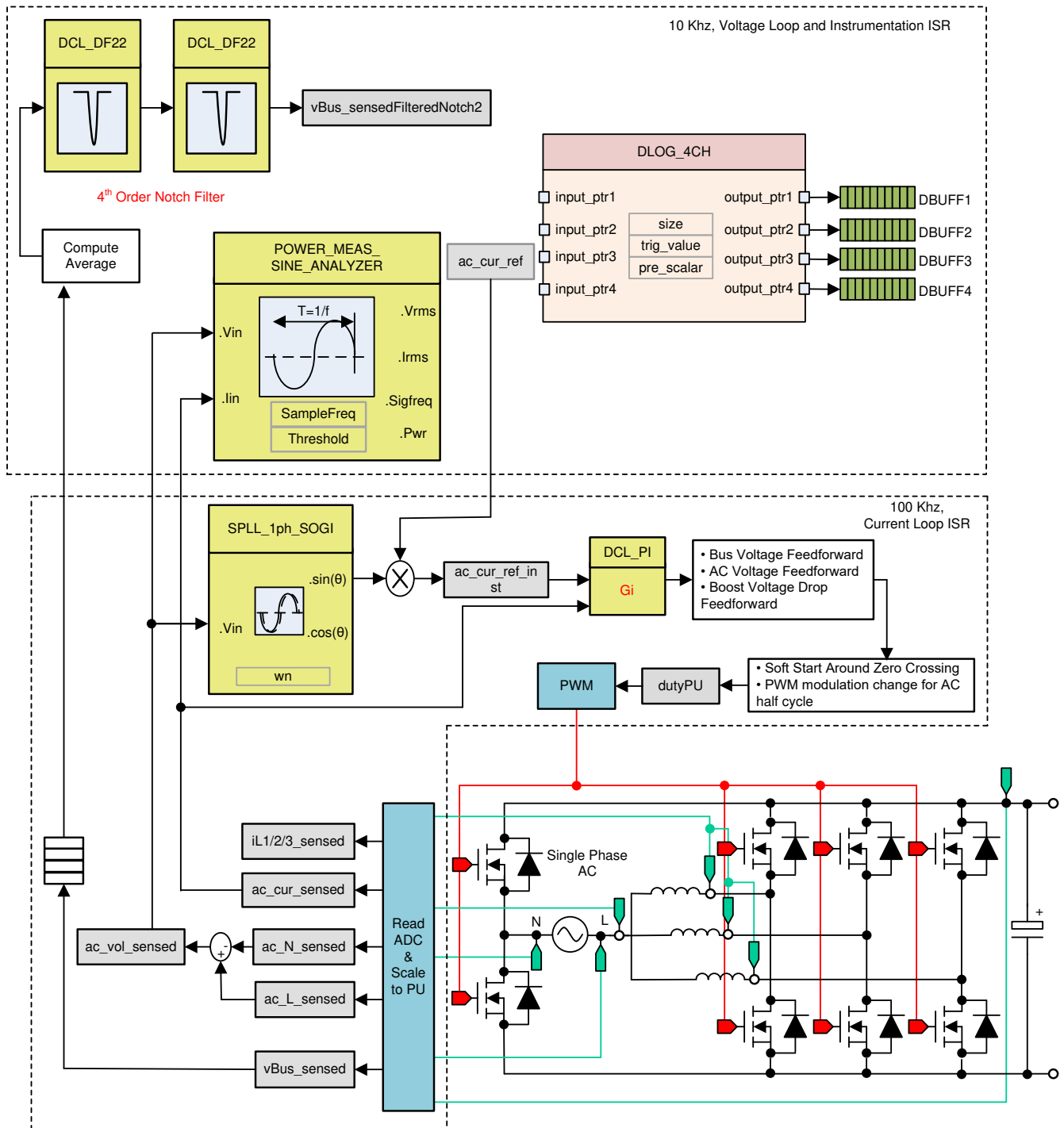


図 5-37. ビルド ラボ 3 の制御ソフトウェア構成図: 閉電流ループ AC

5.2.3.3.1 ラボ 3 のソフトウェアオプションの設定


TTPLPFC_settings.h を開き、Lab3 を有効にします。

```
#define TTPLPFC_LAB 3
```

過去のビルドの電流補償器をこのビルドで再利用するため、ビルドの電流ループを調整する追加手順は必要ありません。

5.2.3.3.2 プロジェクトのビルドおよびロードとデバッグの設定

プロジェクト名を右クリックし、[Rebuild Project] をクリックします。プロジェクトが正常にビルドされます。*Run* → *Debug* をクリックして、デバッグセッションを起動します。デュアル CPU デバイスの場合、ウィンドウが表示され、デバッグを実行する必要がある CPU を選択できます。ここでは、CPU1 を選択します。するとプロジェクトがデバイスにロードされ、CCS デバッグビューが有効になります。メインルーチンの開始時にコードは停止します。

[Watch] および [Expressions] ウィンドウに変数を追加するには、[View] → [Scripting Console] をクリックして、[Scripting Console] ダイアログ ボックスを開きます。このコンソールの右上隅で、[Open] をクリックして、プロジェクトフォルダ内にある `setupdebugenv_Lab3.js` スクリプト ファイルを参照します。このファイルにより、[Watch] ウィンドウに、システムをデバッグするのに必要な適切な変数が入力されます。[Watch] ウィンドウで [Continuous Refresh] ボタン () をクリックして、コントローラからの値の連続更新を有効にします。図 1-1 のように [Watch] ウィンドウが表示されます。

Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel2_CurrentLoop_AC	0x0000A806@Data
pwmSwState	enum enum_pwmSwState	pwmSwState_defaultState	0x0000A824@Data
boardStatus	enum enum_boardStatus	boardStatus_InputUnderVoltageTrip	0x0000A814@Data
clearTrip	int	0	0x0000A809@Data
closeGilLoop	int	0	0x0000A807@Data
ac_cur_ref	float	0.02999999993	0x0000A840@Data
ac_cur_sensed	float	0.010755837	0x0000A8A8@Data
guiVbus	float	0.344914794	0x0000A874@Data
guiVin	float	-0.563325524	0x0000A882@Data
guiVrms	float	0.0	0x0000A872@Data
guiIrms	float	0.0	0x0000A878@Data
guiPrms	float	0.0	0x0000A86E@Data
guiFreqAvg	float	0.0	0x0000A880@Data
guiPowerFactor	float	0.0	0x0000A87C@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
dutyPU	float	0.009999999978	0x0000A86C@Data
dutyPU_DC	float	0.5	0x0000A862@Data
iL1_sensed	float	-0.00634765625	0x0000A8C4@Data
iL2_sensed	float	-0.00830078125	0x0000A8BE@Data
iL3_sensed	float	-0.0112304688	0x0000A8C0@Data
autoStartSlew	unsigned long	0	0x0000A85E@Data
+ Add new expression			

図 5-38. ラボ 3 AC: 閉電流ループ Expressions ビュー

マウス ポインタを水平ツールバーのボタンの上に置き、 ボタンをクリックして、リアルタイム モードを有効にします。

5.2.3.3.3 コードの実行

このプロジェクトは、入力電圧が約 70V_{rms} を上回るまで待機してから突入リレーを駆動し、検出をクリアするようにプログラミングされています。

 をクリックしてプロジェクトを実行します。

約 120V の入力電圧を印加すると、基板は低電圧状態から脱し、突入リレーが駆動されます。検出がクリアされ、約 1.3A RMS の少量の電流が流れます。Watch ウィンドウの外観は図 1-1 と同様です。バス電圧はほぼ 270 V です。

Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel2_CurrentLoop_AC	0x0000A806@Data
pwmSwState	enum enum_pwmSwState	pwmSwState_positiveHalf	0x0000A824@Data
boardStatus	enum enum_boardStatus	boardStatus_NoFault	0x0000A814@Data
clearTrip	int	1	0x0000A809@Data
closeGilLoop	int	1	0x0000A807@Data
ac_cur_ref	float	0.0299999993	0x0000A840@Data
ac_cur_sensed	float	-0.00663924217	0x0000A8A8@Data
guiVbus	float	180.061981	0x0000A874@Data
guiVin	float	-49.6501122	0x0000A882@Data
guiVrms	float	117.459831	0x0000A872@Data
guiIrms	float	0.551513135	0x0000A878@Data
guiPrms	float	64.2371902	0x0000A86E@Data
guiFreqAvg	float	59.8999023	0x0000A880@Data
guiPowerFactor	float	0.978407621	0x0000A87C@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	-0.880984187	0x0000A86C@Data
dutyPU_DC	float	0.5	0x0000A862@Data
iL1_sensed	float	0.0180664063	0x0000A8C4@Data
iL2_sensed	float	-0.0048828125	0x0000A8BE@Data
iL3_sensed	float	-0.0283203125	0x0000A8C0@Data
autoStartSlew	unsigned long	5	0x0000A85E@Data
+ Add new expression			

図 5-39. Watch 式、ラボ 2、閉電流ループ動作開始後の AC

ac_cur_ref を 0.078、すなわち 2.4A 入力まで徐々に上げていくと、バス電圧が 400V まで上昇します。電圧と電流の波形を 図 1-1 に示します。

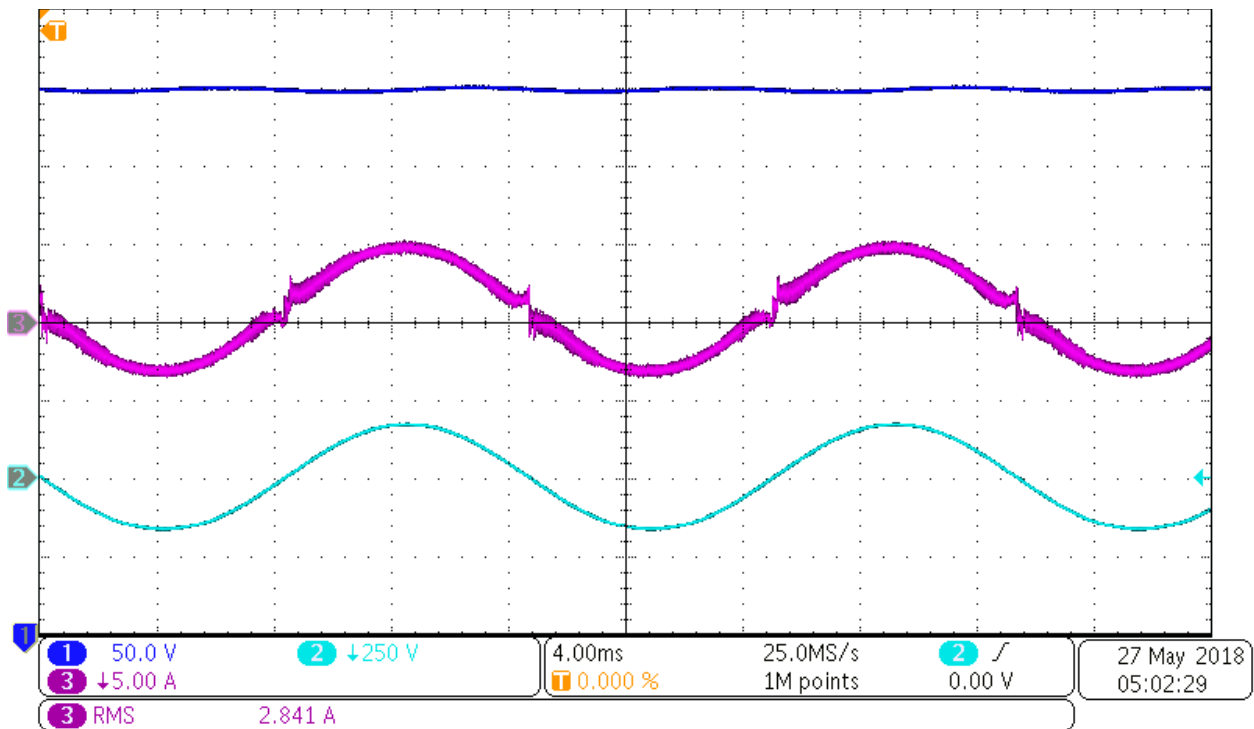





図 5-40. 入力 AC 電流と出力 DC 電圧の波形

このビルドのソフトウェアには SFRA が統合されているため、ハードウェアを測定して、設計した補償器が十分なゲイン マージンと位相マージンを提供していることを検証できます。SFRA を実行するには、プロジェクトを実行したまま、<Install directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\SFRA_GUI.exe に移動します。SFRA GUI が表示されます。

SFRA GUI でデバイスのオプションを選択します。例として、F280039 の場合には浮動小数点を選択します。[Setup Connection] をクリックします。ポップアップ ウィンドウで [Boot on Connect] オプションを選択解除し、適切な COM ポートを選択します。[OK] をクリックします。SFRA GUI に戻り、[Connect] をクリックします。

SFRA GUI がデバイスに接続します。これで [Start Sweep] をクリックして、SFRA 掃引を開始できるようになりました。SFRA 掃引が完了するまでには数分かかります。SFRA GUI のプログレス バーを確認したり、UART の動作を示す制御カード裏面の青色 LED の点滅をチェックすることで、動作を監視できます。終了すると、開ループプロットによるグラフが表示されます。これは DC 条件下のプロットと似ていますが、測定周波数付近に AC 高調波による若干の追加ノイズが確認されます。BW、PM、GM の数値は DC の場合と非常に似通っています。

システムを安全に停止させるには、AC 電源からの出力をオフにし、入力 AC 電圧をゼロまで下げます。guiVbus もゼロに下がっていることを確認します。

リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの [Halt] ボタン () を使用するか、[Target] → [Halt] の順にクリックして、プロセッサを停止します。次に、 をクリックしてマイクロコントローラをリアルタイム モードから解除します。最後に、マイクロコントローラ () をリセットします。

[Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグセッションを終了します。

5.2.3.4 ラボ 4: 閉電圧および電流ループ

このビルドでは、外部電圧ループも内部電流ループも閉じています。外部電圧ループのモデルは [図 1-1](#) に示されています (TIDM-02013 は 2 相インターリーブ TTPPLPFC であることに注意してください)。PI 補償器を使用して、外部電圧ループを Compensation Designer で調整します。

[図 1-1](#) にこのビルドのソフトウェア構成図を示します。

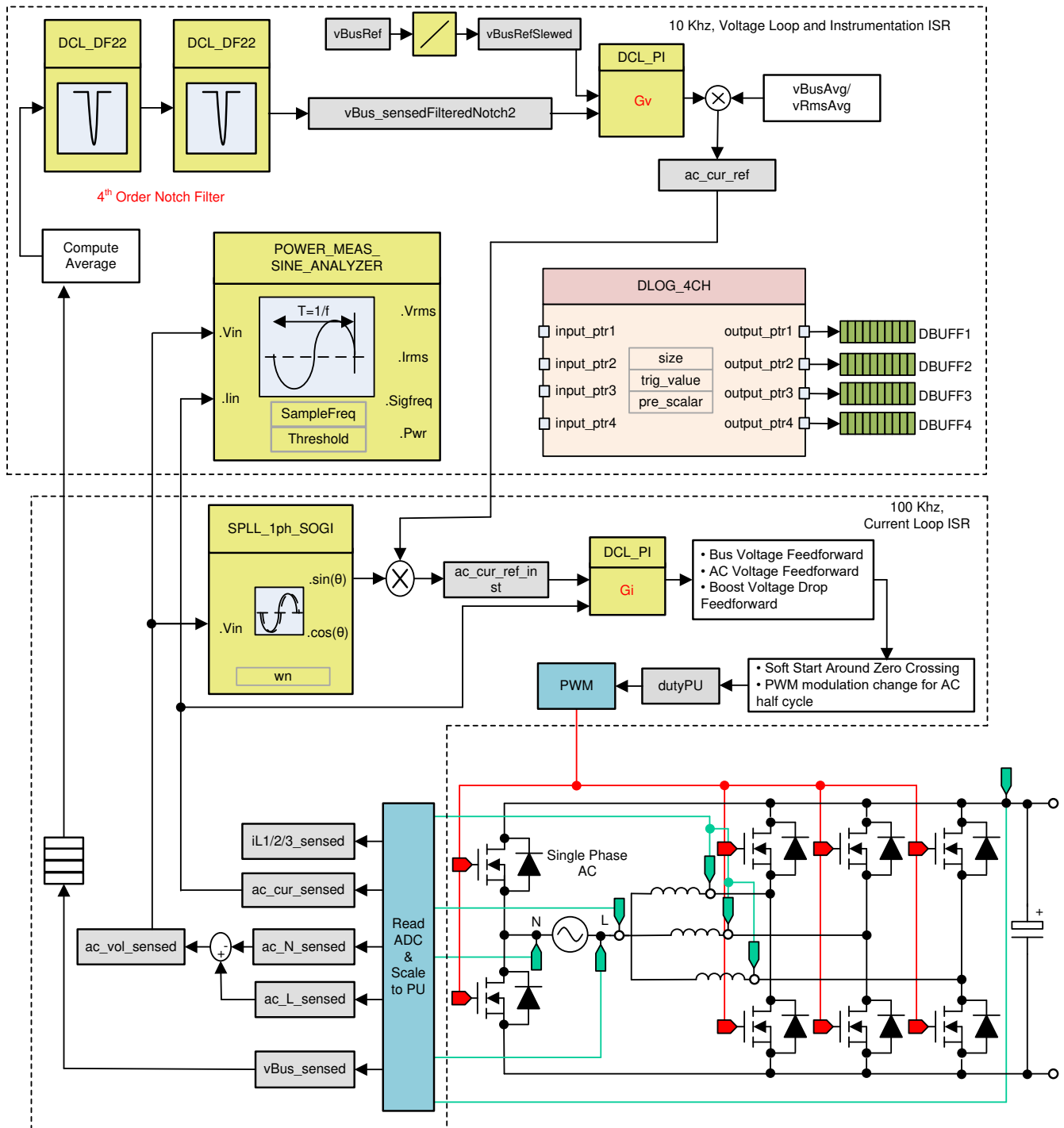


図 5-41. ビルドレベル 4 の制御図: 内部電流ループによる出力電圧制御

5.2.3.4.1 BUILD 4 のソフトウェアオプションの設定

セクション 5.1.1 で説明されているように、ハードウェアがスタンダアローン PFC 動作用に設定されていることを確認します。高電圧(HV)電力はまだ基板に供給しないでください。

TTPLPFC_settings.h を開き、Lab4 を有効にします。

```
#define TTPLPFC_LAB 4
```


その他のすべてのオプションが 図 1-1 で指定したものと同一であることを確認します。


1. Compensation Designer (<install

Directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfragui\CompDesigner.exe) を開きます。

5.2.3.4.2 プロジェクトのビルドおよびロードとデバッグの設定


プロジェクト名を右クリックし、[Rebuild Project] をクリックします。プロジェクトが正常にビルドされます。**Run** → **Debug** をクリックして、デバッグセッションを起動します。デュアル CPU デバイスの場合、ウィンドウが表示され、デバッグを実行する必要がある CPU を選択できます。ここでは、CPU1 を選択します。するとプロジェクトがデバイスにロードされ、CCS デバッグビューが有効になります。メインルーチンの開始時にコードは停止します。

[Watch] および [Expressions] ウィンドウに変数を追加するには、[View] → [Scripting Console] をクリックして、[Scripting Console] ダイアログボックスを開きます。このコンソールの右上隅で、[Open] をクリックして、プロジェクトフォルダ内にある setupdebugenv_lab4.js スクリプトファイルを参照します。このファイルにより、[Watch] ウィンドウに、システムをデバッグするのに必要な適切な変数が入力されます。[Watch] ウィンドウで [Continuous Refresh] ボタン () をクリックして、コントローラからの値の連続更新を有効にします。

 1-1 に示すように [Watch] ウィンドウが表示されます。

Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel3_VoltageLoop_AC	0x0000A811@Data
pwmSwState	enum enum_pwmSwState	pwmSwState_defaultState	0x0000A826@Data
boardStatus	enum enum_boardStatus	boardStatus_InputUnderVoltageTrip	0x0000A80F@Data
clearTrip	int	0	0x0000A81A@Data
closeGvLoop	int	0	0x0000A819@Data
vBusRef	float	0.821337461	0x0000A850@Data
vBus_sensed	float	0.000651041686	0x0000A8C0@Data
closeGiLoop	int	0	0x0000A81C@Data
ac_cur_senseOffset	float	0.502499998	0x0000A888@Data
guiVbus	float	0.353723764	0x0000A858@Data
guiVin	float	0.375192761	0x0000A82C@Data
guiVrms	float	0.0	0x0000A842@Data
guiIrms	float	0.0	0x0000A832@Data
guiPrms	float	0.0	0x0000A834@Data
guiPowerFactor	float	0.0	0x0000A82E@Data
guiFreqAvg	float	0.0	0x0000A844@Data
EPwm1Regs.TZFLG	Register	0x0004	
EPwm2Regs.TZFLG	Register	0x0004	
dutyPU	float	0.00999999978	0x0000A86E@Data
dutyPU_DC	float	0.5	0x0000A86C@Data
il1_sensed	float	-0.00390625	0x0000A884@Data
il2_sensed	float	-0.00634765625	0x0000A880@Data
il3_sensed	float	-0.00244140625	0x0000A882@Data
autoStartSlew	unsigned long	0	0x0000A83E@Data
+ Add new expression			

図 5-42. ビルドラボ 4: Expressions ビュー

マウスポインタを水平ツールバーのボタンの上に置き、 ボタンをクリックして、リアルタイムモードを有効にします。

 をクリックしてプロジェクトを実行します。

ここでツールバーの [Halt] ボタン () をクリックして、プロセッサを停止します。

5.2.3.4.3 コードの実行

このプロジェクトは、入力電圧が約 70V_{rms} を上回るまで待機してから突入リレーを駆動し、検出をクリアするようにプログラミングされています。

 をクリックしてプロジェクトを実行します。

約 120V の入力電圧を印加すると、基板は低電圧状態から脱し、突入リレーが駆動されます。検出がクリアされ、出力が DC 380V まで上昇します。AC 入力から正弦波電流が引き出されます。この段でプログラムを実行しているときの [Watch] ウィンドウを [図 1-1](#) に示します。

Expression	Type	Value	Address
buildInfo	enum enum_BuildLevel	BuildLevel3_VoltageLoop_AC	0x0000A811@Data
pwmSwState	enum enum_pwmSwState	pwmSwState_negativeHalf	0x0000A826@Data
boardStatus	enum enum_boardStatus	boardStatus_NoFault	0x0000A80F@Data
clearTrip	int	1	0x0000A81A@Data
closeGvLoop	int	1	0x0000A819@Data
vBusRef	float	0.821337461	0x0000A850@Data
vBus_sensed	float	0.822998047	0x0000A8C0@Data
closeGilLoop	int	1	0x0000A81C@Data
ac_cur_senseOffset	float	0.502499998	0x0000A888@Data
guiVbus	float	380.081421	0x0000A858@Data
guiVin	float	-152.073486	0x0000A82C@Data
guiVrms	float	120.093376	0x0000A842@Data
guiIrms	float	2.40836215	0x0000A832@Data
guiPrms	float	277.007263	0x0000A834@Data
guiPowerFactor	float	0.990778685	0x0000A82E@Data
guiFreqAvg	float	60.0219727	0x0000A844@Data
EPwm1Regs.TZFLG	Register	0x0000	
EPwm2Regs.TZFLG	Register	0x0000	
dutyPU	float	-0.4262546	0x0000A86E@Data
dutyPU_DC	float	0.5	0x0000A86C@Data
il1_sensed	float	0.0561523438	0x0000A884@Data
il2_sensed	float	-0.0673828125	0x0000A880@Data
il3_sensed	float	-0.0434570313	0x0000A882@Data
autoStartSlew	unsigned long	5	0x0000A83E@Data
+ Add new expression			

図 5-43. ビルドラボ 4: AC 電圧を印加した後の Expressions ビュー

このビルドのソフトウェアには SFRA が統合されているため、ハードウェアを測定して、設計した補償器が十分なゲイン マージンと位相マージンを提供していることを検証できます。SFRA を実行するには、プロジェクトを実行したまま、<Install directory>\C2000Ware_DigitalPower_SDK_<version>\libraries\sfra\gui\SFRA_GUI.exe に移動します。SFRA GUI が表示されます。

SFRA GUI でデバイスのオプションを選択します。例として、F28003x の場合には浮動小数点を選択します。[Setup Connection] をクリックし、ポップアップ ウィンドウで [Boot on Connect] オプションを選択解除し、適切な COM ポートを選択します。[OK] をクリックします。SFRA GUI に戻り、[Connect] をクリックします。

SFRA GUI がデバイスに接続します。これで [Start Sweep] をクリックして、SFRA 掃引を開始できるようになりました。SFRA 掃引が完了するまでには数分かかります。SFRA GUI のプログレスバーを確認したり、UART の動作を示す制御カード裏面の青色 LED の点滅をチェックすることで、動作を監視できます。終了すると、[図 1-1](#) のように開ループプロットによるグラフが表示されます。この操作により、設計した補償器が確かに安定していることを検証できます。

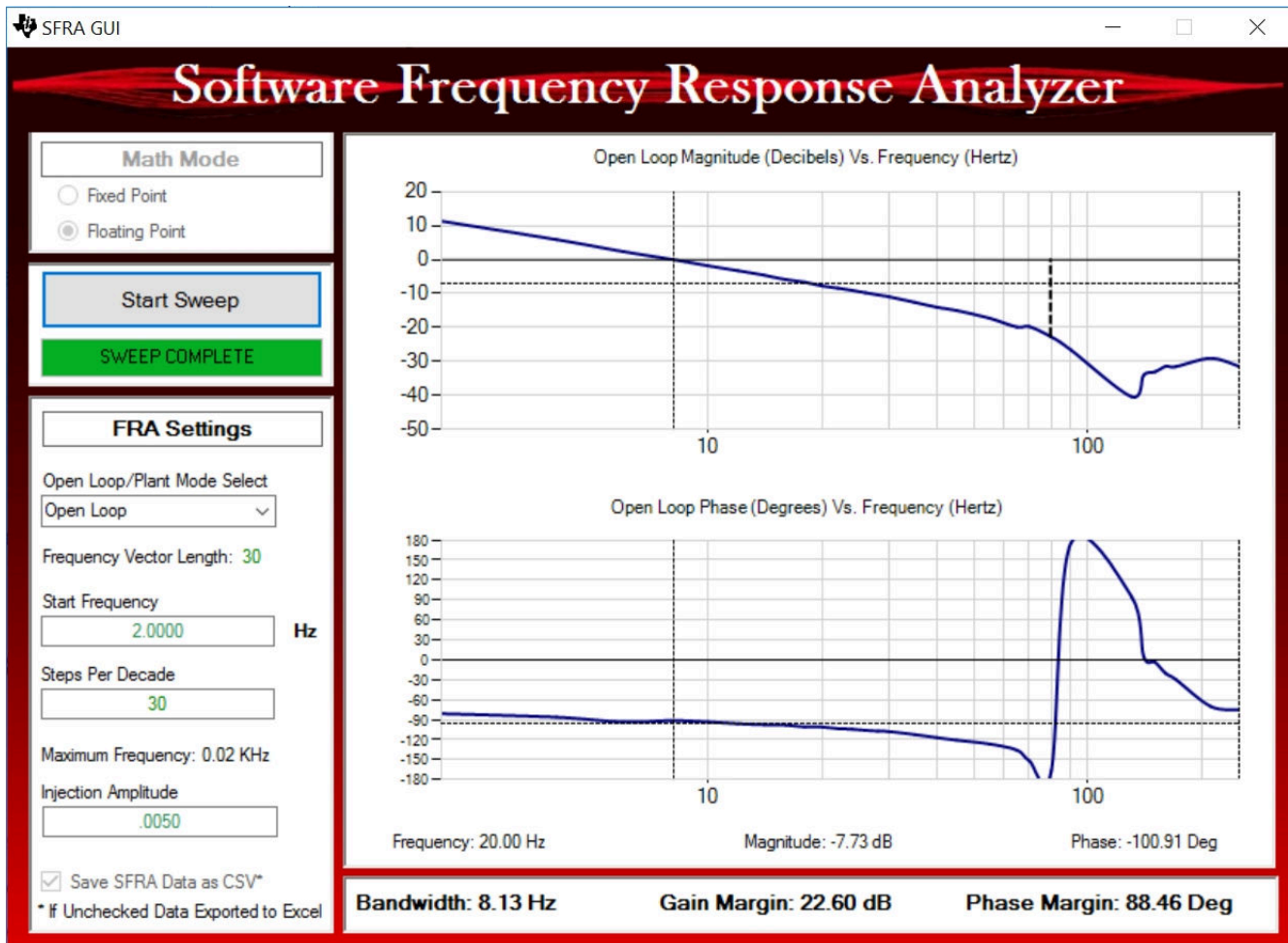






図 5-44. 閉電圧ループでの SFRA 実行

別の方法として、Compensation Designer を再度開き、GUI でプラントオプションに SFRA Data を選択します。このオプションにより、測定したプラント情報を用いて補償器を設計し、補償を微調整できます。デフォルトでは、Compensation Designer は最新の SFRA 実行を示しています。過去の SFRA 実行時のプラント情報を使用する必要がある場合、[Browse SFRA Data] をクリックし、参照して SFRADData.csv ファイルを選択してください。Compensation Designer を閉じて、電圧補償器の設計を検証します。

システムを安全に停止させるには、入力 AC 電圧をゼロまで下げます。guiVBus もゼロに下がっていることを確認します。リアルタイム モードのマイクロコントローラを完全に停止するには、2 段階の手順を踏みます。まず、ツールバーの [Halt] ボタン () を使用するか、[Target] → [Halt] の順にクリックして、プロセッサを停止します。次に、 をクリックしてマイクロコントローラをリアルタイム モードから解除します。最後に、マイクロコントローラ () をリセットします。

[Terminate Debug Session] () ([Target] → [Terminate all]) をクリックして、CCS デバッグ セッションを終了します。

5.2.4 テスト結果

このデザインで実現できる電力密度は 3.8kW/L (62.5W/in³) です。システム全体の効率は 96.5% で、PFC の効率は 98.5%、CLLLC の効率は 98% です。

5.2.4.1 効率

12V バイアス電源の有無による効率データを以下の各グラフに示します。このバイアス電源は、制御、アイソレータ、ゲートドライブに電力を供給します。図 1-1 のグラフは、次の条件で取得したものです。

- $V_{IN,RMS} = 240V$
- $V_{OUT} = 400V$
- クーラント温度: $20^{\circ}C$

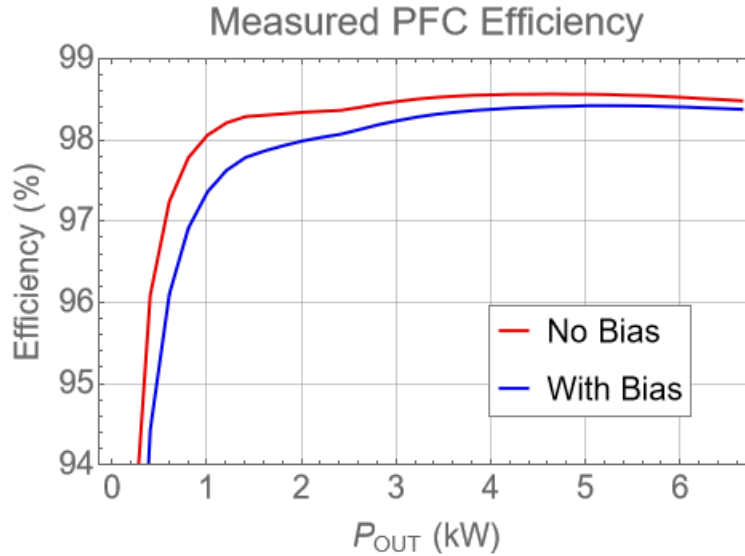


図 5-45. PFC の効率

図 1-1 のグラフは、次の条件で取得したものです。

- $V_{IN} = 400V$
- $V_{OUT} = 350V$
- クーラント温度: $20^{\circ}C$

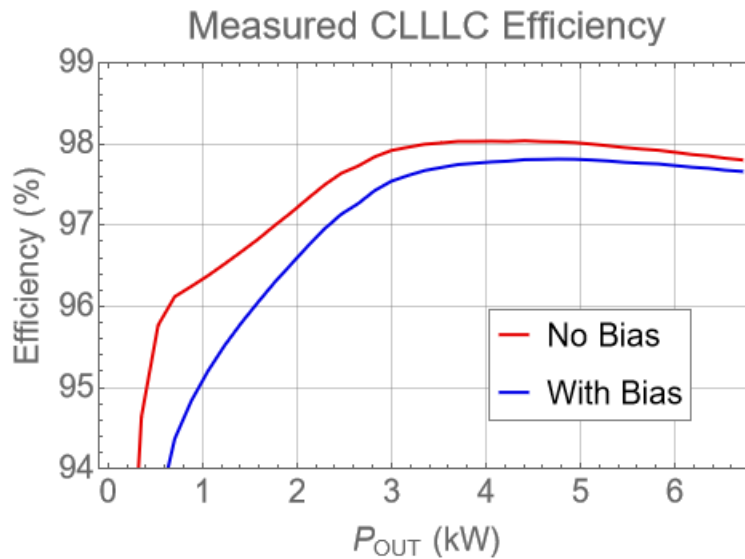


図 5-46. CLLC の効率

図 1-1 のグラフは、次の条件で取得したものです。

- $V_{IN,RMS} = 240V$
- $V_{OUT} = 350V$
- クーラント温度: $20^{\circ}C$

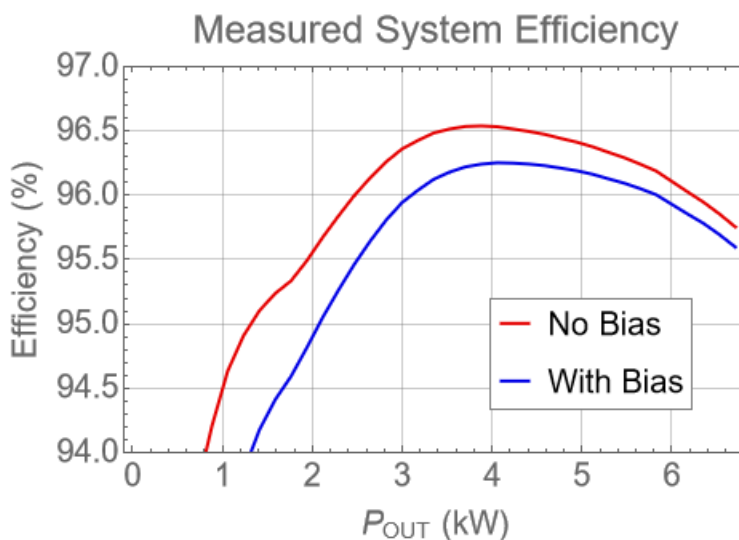


図 5-47. システム効率

5.2.4.2 システム性能

以下の図は、システム全体の効率、システム損失、全高調波歪み (THD)、正規化された出力電圧レギュレーション精度をまとめたものです。

このデザインで実現できる電力密度は 3.8kW/L (62.5W/in^3) です。これによって、システム全体の効率は 96.5% に達します。 1.5kW を超える負荷の THD は 5% 未満で、出力電圧のレギュレーション精度はおおむね $\pm 0.06\%$ 以内です。

図 1-1 のグラフは、次の条件を使用しています。

- $V_{\text{IN,RMS}} = 240\text{V}$
- $V_{\text{OUT}} = 350\text{V}$
- クーラント温度: 20°C

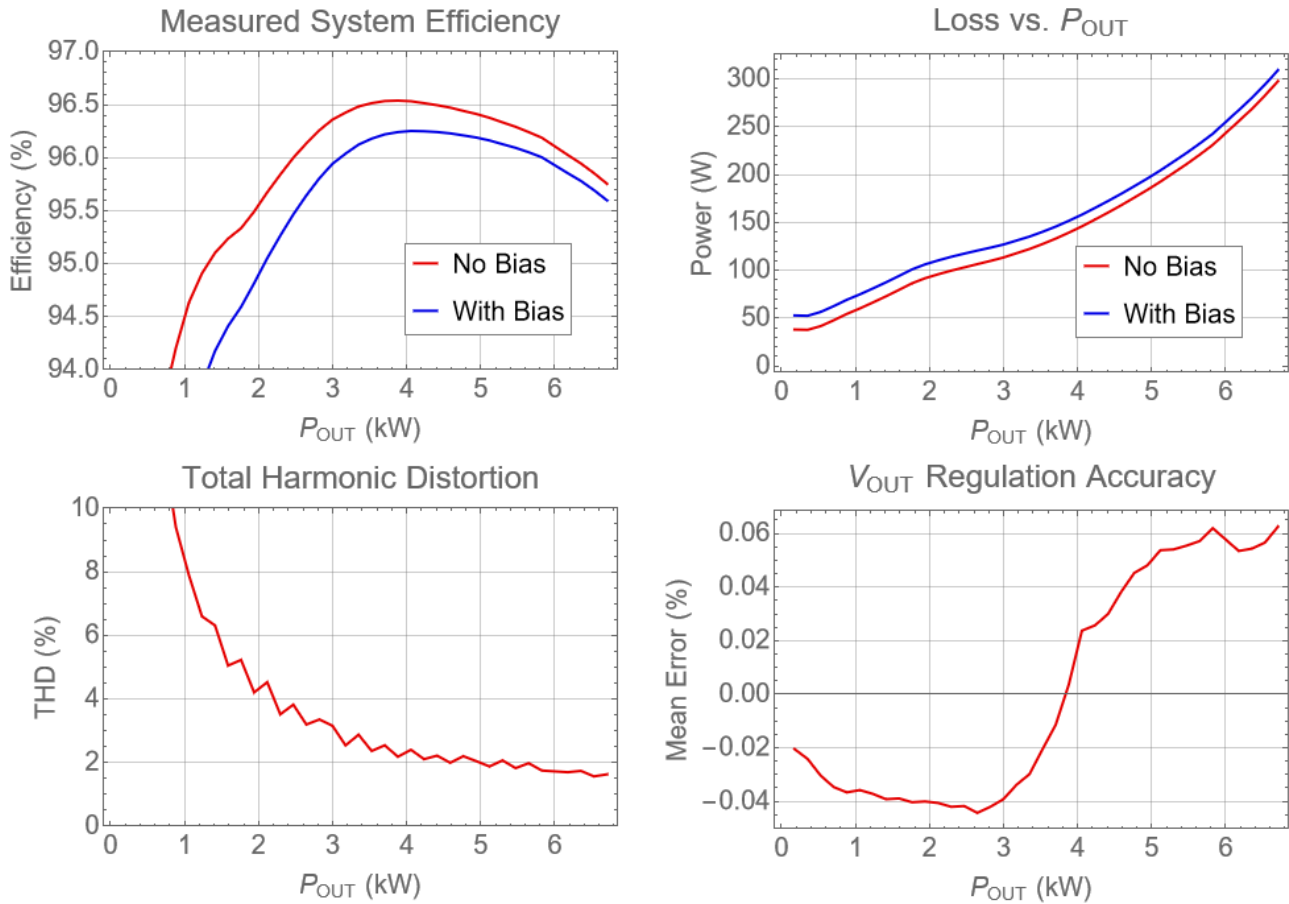


図 5-48. システム性能

5.2.4.3 ボード線図

以下のボード線図は、TMS320F28388D マイクロコントローラに搭載されているオンボードソフトウェア周波数応答アナライザを使用して取得したものです。テストで使用された負荷は定電流シンクとして構成され、マイクロコントローラは一定の出力電圧を調整するように構成されています。帯域幅はおおよそ 1kHz~2.5kHz で、位相マージンは 45°C を超えています。

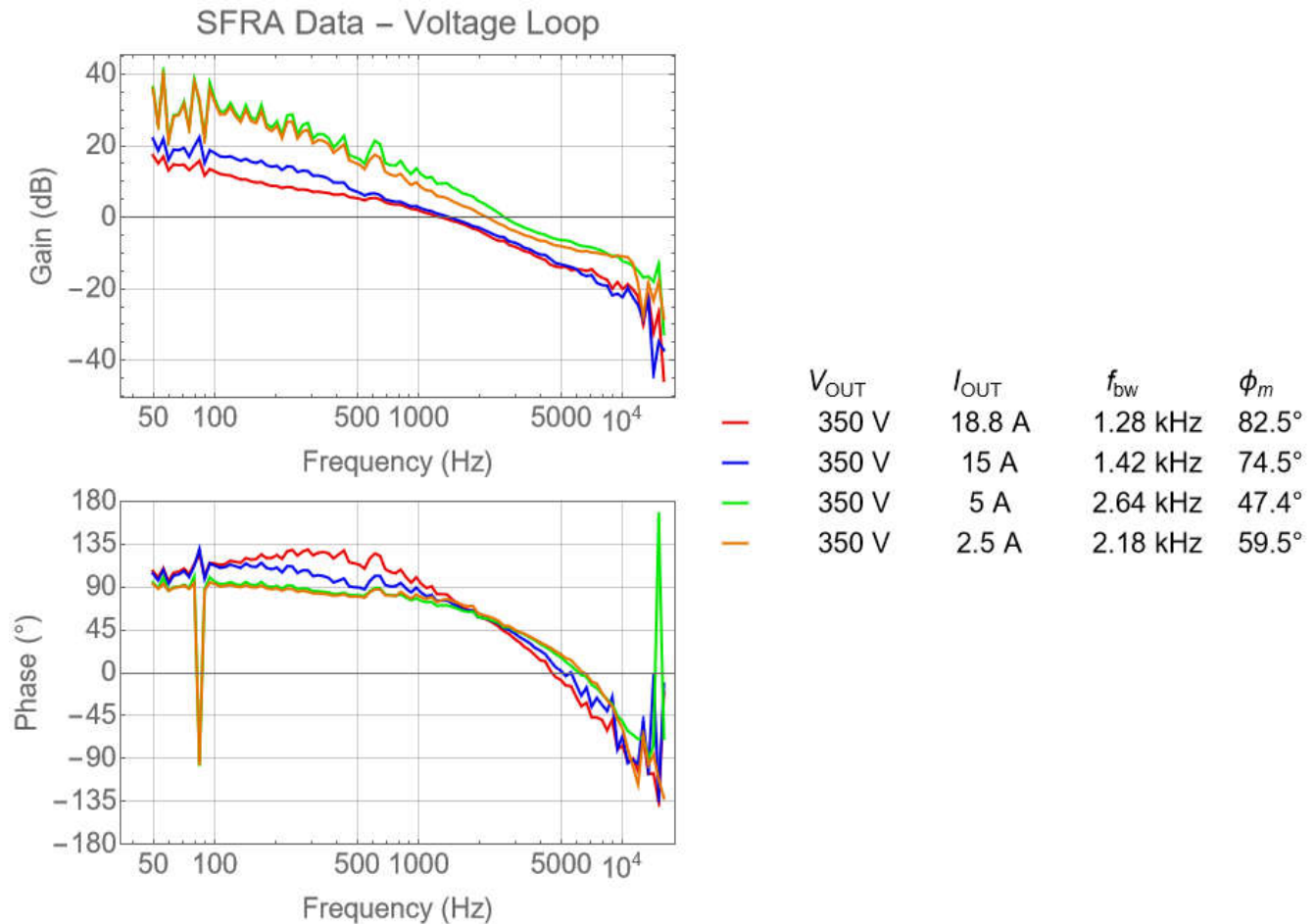


図 5-49. 電圧ループのボード線図

以下のボード線図は、TMS320F28388D マイクロコントローラに搭載されているオンボードソフトウェア周波数応答アナライザを使用して取得したものです。テストで使用された負荷は定電圧として構成され、マイクロコントローラは一定の出力電流を調整するように構成されています。帯域幅はおおよそ 1kHz~2.5kHz で、位相マージンは 60° を超えています。

SFRA Data – Current Loop

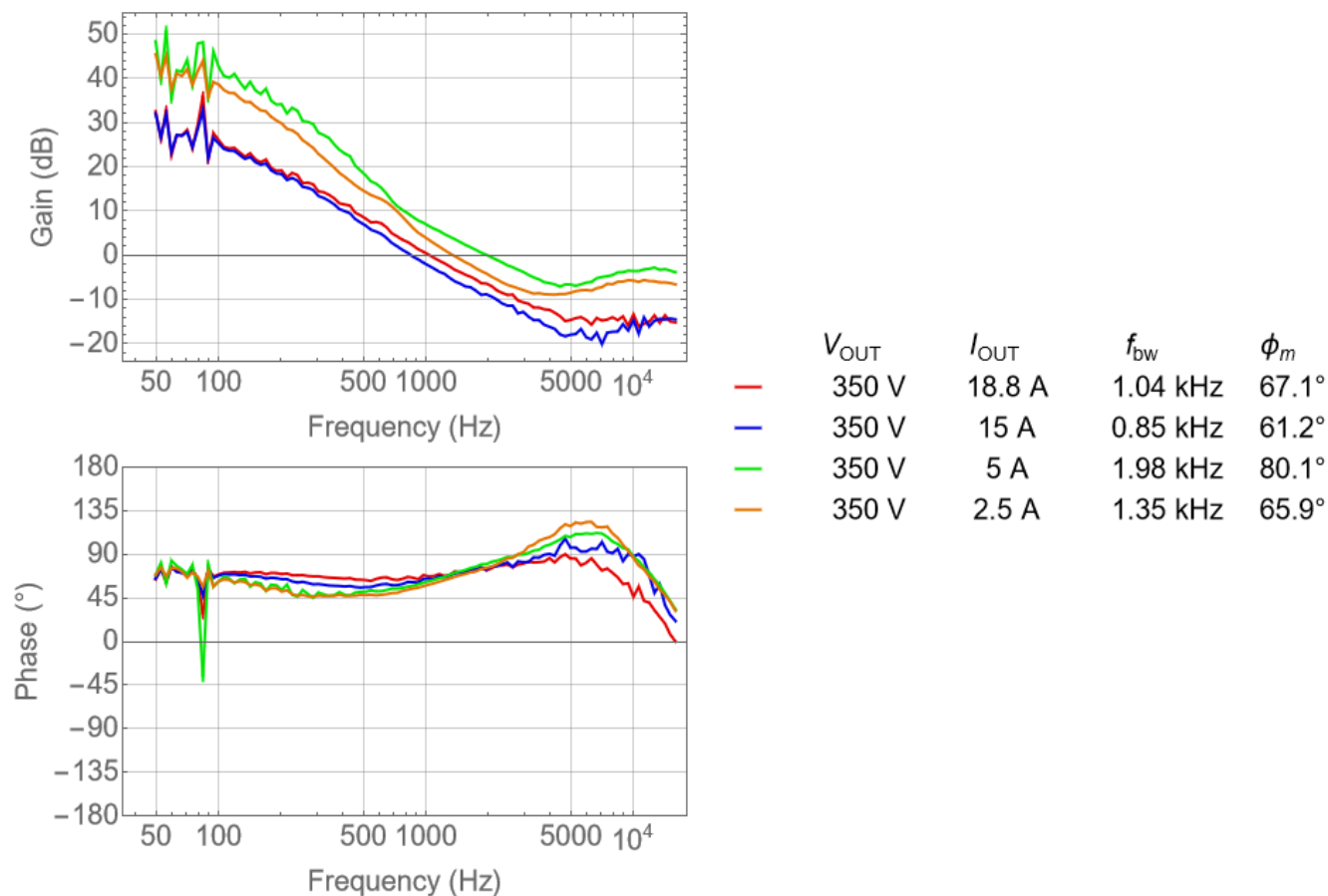


図 5-50. 電圧ループのボード線図 (定電圧負荷)

5.2.4.4 効率とレギュレーションのデータ

次の表は効率とレギュレーションのデータを示します。

V _{OUT} (V)	I _{OUT} (A)	P _{OUT} (W)	V _{IN} (V)	I _{IN} (A)	P _{IN} (W)	V _{BIAS} (V)	I _{BIAS} (A)	P _{BIAS} (W)	効率 (%) バ イアスなし	効率 (%) バ イアスあり
352.76	0.5	177.35	240.12	1.17	215.36	11.92	1.24	14.73	82.35	77.08
352.74	1	354.12	240.05	1.79	391.84	11.92	1.23	14.67	90.37	87.11
352.72	1.5	530.65	239.98	2.49	572.25	11.92	1.24	14.8	92.73	90.39
352.71	2	706.27	239.91	3.23	753.97	11.92	1.24	14.78	93.67	91.87
352.7	2.5	882.52	239.84	3.97	936.83	11.92	1.24	14.76	94.2	92.74
352.7	3	1058.9	239.76	4.72	1118.95	11.92	1.23	14.71	94.63	93.41
352.7	3.5	1235.3	239.69	5.48	1301.58	11.92	1.23	14.65	94.91	93.85
352.69	4	1411.74	239.62	6.23	1484.44	11.92	1.23	14.61	95.1	94.18
352.69	4.5	1587.99	239.55	6.99	1667.43	11.92	1.22	14.57	95.24	94.41
352.69	5	1763.62	239.47	7.75	1849.94	11.92	1.22	14.51	95.33	94.59
352.69	5.5	1939.96	239.4	8.51	2031.56	11.93	1.21	14.48	95.49	94.82
352.69	6	2116.85	239.32	9.27	2212.59	11.93	1.21	14.48	95.67	95.05
352.68	6.5	2293.18	239.25	10.02	2392.72	11.93	1.21	14.46	95.84	95.26
352.68	7	2469.51	239.17	10.78	2572.43	11.93	1.21	14.4	96	95.46
352.67	7.5	2645.17	239.1	11.53	2751.5	11.93	1.19	14.16	96.14	95.64
352.68	8	2821.8	239.02	12.29	2931.48	11.94	1.16	13.85	96.26	95.81
352.69	8.5	2998.24	238.95	13.04	3111.5	11.94	1.14	13.56	96.36	95.94
352.71	9	3174.78	238.87	13.8	3292.5	11.95	1.11	13.26	96.43	96.04
352.72	9.5	3350.46	238.79	14.56	3472.6	11.95	1.09	12.99	96.48	96.12
352.76	10	3527.33	238.71	15.33	3654.7	11.95	1.07	12.76	96.51	96.18
352.79	10.5	3704.19	238.63	16.09	3837.2	11.96	1.05	12.52	96.53	96.22
352.84	11	3881.08	238.54	16.87	4020.3	11.96	1.03	12.37	96.54	96.24
352.91	11.5	4058.4	238.46	17.64	4204.2	11.96	1.02	12.24	96.53	96.25
352.92	12	4235	238.37	18.42	4387.9	11.96	1.02	12.16	96.51	96.25
352.93	12.5	4411.3	238.28	19.2	4571.5	11.96	1.01	12.08	96.5	96.24
352.96	13	4588.2	238.18	19.98	4756	11.96	1	12	96.47	96.23
352.99	13.5	4765.1	238.09	20.77	4940.9	11.96	1	11.92	96.44	96.21
353	14	4941.9	237.99	21.55	5125.8	11.97	0.99	11.86	96.41	96.19
353.02	14.5	5118.9	237.89	22.34	5311.4	11.97	0.99	11.79	96.38	96.16
353.02	15	5294.8	237.79	23.13	5496.4	11.97	0.98	11.73	96.33	96.13
353.02	15.5	5471.6	237.68	23.92	5682.4	11.97	0.98	11.69	96.29	96.09
353.03	16	5648.3	237.62	24.71	5868.9	11.97	0.97	11.64	96.24	96.05
353.05	16.5	5825.2	237.51	25.51	6056.1	11.97	0.97	11.59	96.19	96
353.02	17.5	6176.6	237.26	27.13	6432.3	11.97	0.96	11.52	96.03	95.85
353.02	18	6353.2	237.14	27.94	6621.6	11.97	0.96	11.49	95.95	95.78
353.03	18.5	6529.8	237.01	28.76	6812.2	11.97	0.96	11.47	95.85	95.69
353.05	19	6706.8	236.87	29.59	7004.3	11.97	0.96	11.44	95.75	95.6

5.2.4.5 熱データ

図 1-1 は全負荷動作時に撮影されたものです。発熱が著しい部品はすべて、基板底面のコールドプレートに接続されています。この画像で最も高温に見える部品は、EMI フィルタ内にある同相モード インダクタに起因するものです。これらの部品はコールドプレートに接続されておらず、冷却はすべて周囲の空気を通してのみ行われます。

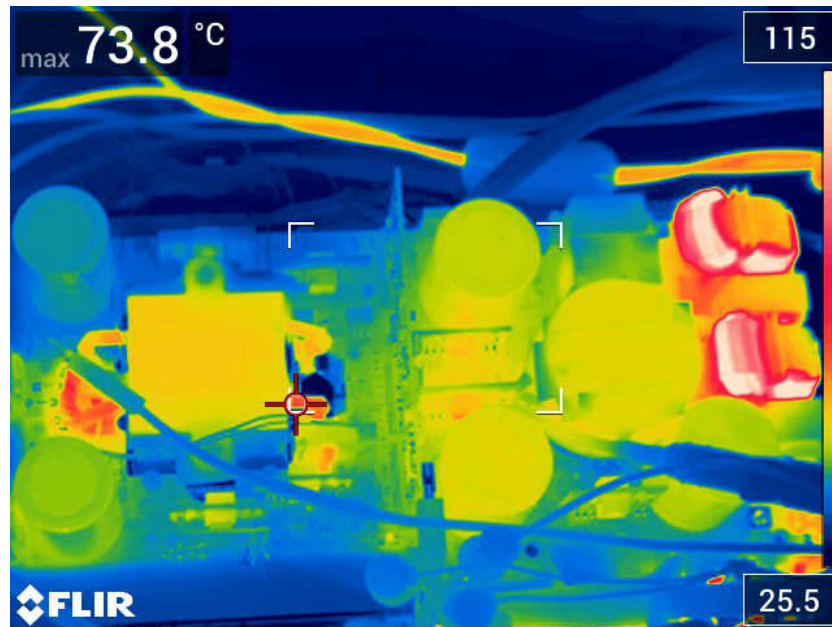


図 5-51. 上面の熱画像

GaN FET の温度は、LMG3522 デバイスに搭載されているオンボード温度センサで測定されます。全負荷条件では、すべての FET 温度は 75°C を下回ります。

表 5-5 に、以下の条件における GaN FET の測定温度を示します。

- $V_{IN,AC}$: 240V
- $V_{DC,LINK}$: 400V
- クーラント温度: 33°C

表 5-5. GaN FET 測定温度

GaN FET	温度 (°C)
PFC	66.8
CLLLC 1 次側 (350V/19A)	58.1
CLLLC 2 次側 (350V/19A)	59.5
CLLLC 1 次側 (300V/19A)	61.0
CLLLC 2 次側 (300V/19A)	74.0

図 1-1 に、以下の条件におけるトランスの臨界温度を示します。

- クーラント温度: 33°C
- トランスの温度測定位置
 - PRI 1 – 1 次側巻線の内側表面で測定
 - PRI 2 – 1 次側巻線の外側表面で測定
 - SEC 1 – 2 次側巻線の内側表面で測定
 - SEC 2 – 2 次側巻線の外側表面で測定
 - CORE 1 – コアのセンターレグ上部で測定
 - CORE 2 – コアのセンターレグ下部で測定
 - CORE 3 – コアの側面で測定
 - CORE 4 – コアの上部で測定

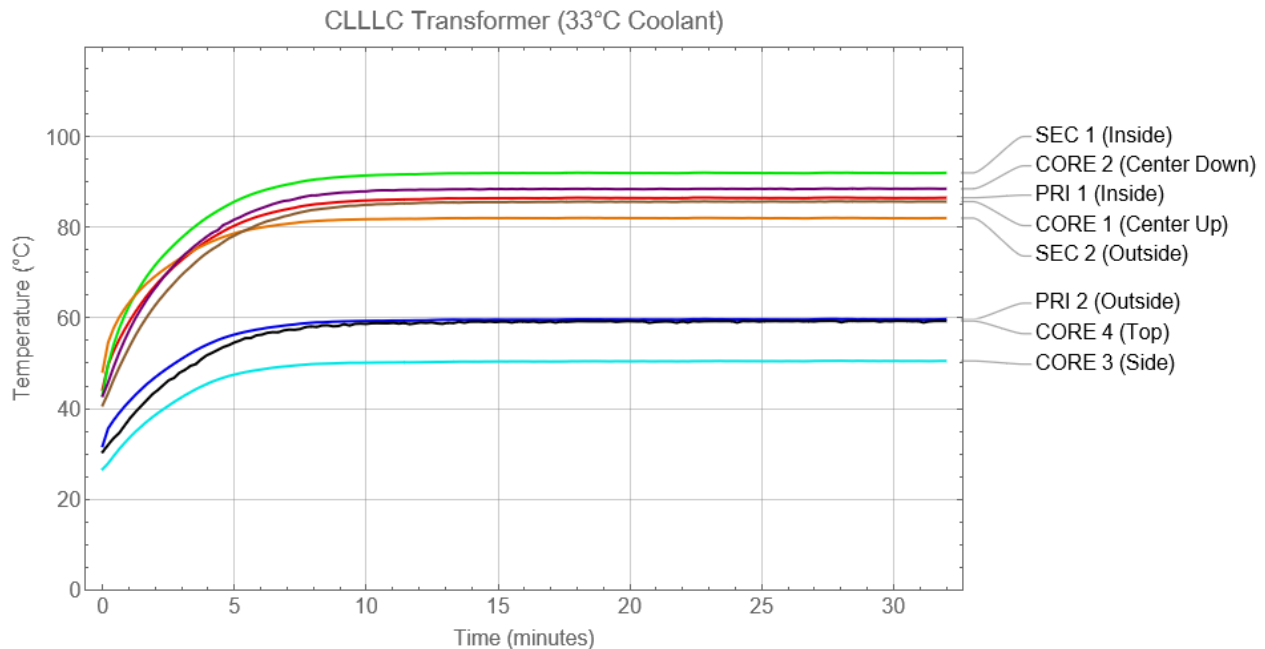


図 5-52. CLLLC トランスの温度

5.2.4.6 PFC の波形

図 1-1 に、以下のパラメータで測定された PFC 入力電圧および入力電流の波形を示します。

- トレース
 - C2: V_{IN}
 - C4: I_{IN}
- 条件
 - $V_{IN} = 208V$
 - $V_{OUT} = 400V$
 - $R_{OUT} = 43\Omega$

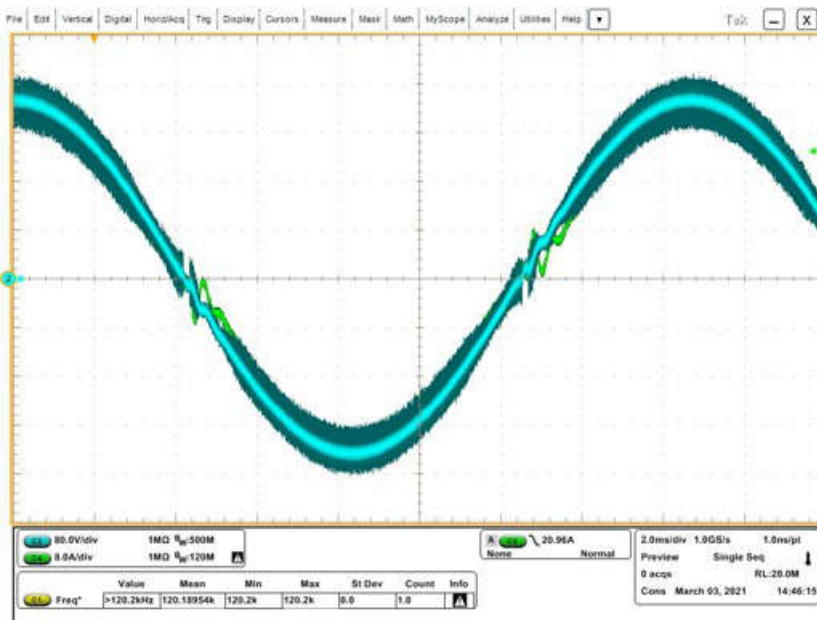


図 5-53. PFC の入力電圧と入力電流

図 1-1 に、次のパラメータで測定された PFC の GaN ドレイン電圧の波形を示します。

- トレース
 - C1: GaN スwitch のノードドレイン電圧
 - C2: V_{IN}
 - C4: I_{IN}
- 条件
 - $V_{IN} = 208V$
 - $V_{OUT} = 400V$
 - $R_{OUT} = 43\Omega$

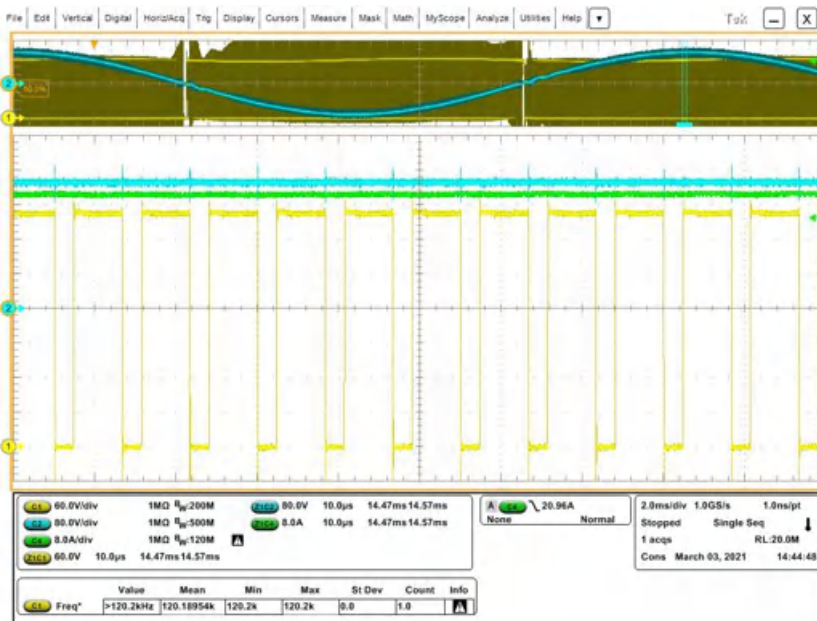


図 5-54. PFC の GaN ドレイン電圧

GaN スwitchのドレイン - ソース間の電圧遷移を拡大すると、[図 1-1](#) に示すように約 20ns となります。このような急激な遷移は、LMG3522 の C_{OSS} が低いことに起因しています。

[図 1-1](#) の波形は、次のパラメータで測定されたものです。

- トレース
 - C1: GaN スwitchのノードドレイン電圧
 - C2: V_{IN}
 - C4: I_{IN}
- 条件
 - $V_{IN} = 208V$
 - $V_{OUT} = 400V$
 - $R_{OUT} = 43\Omega$

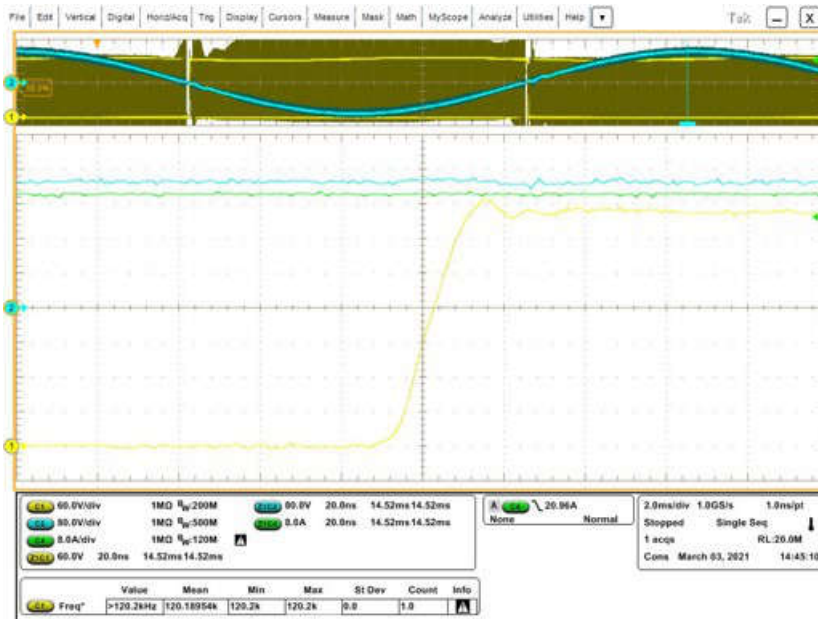
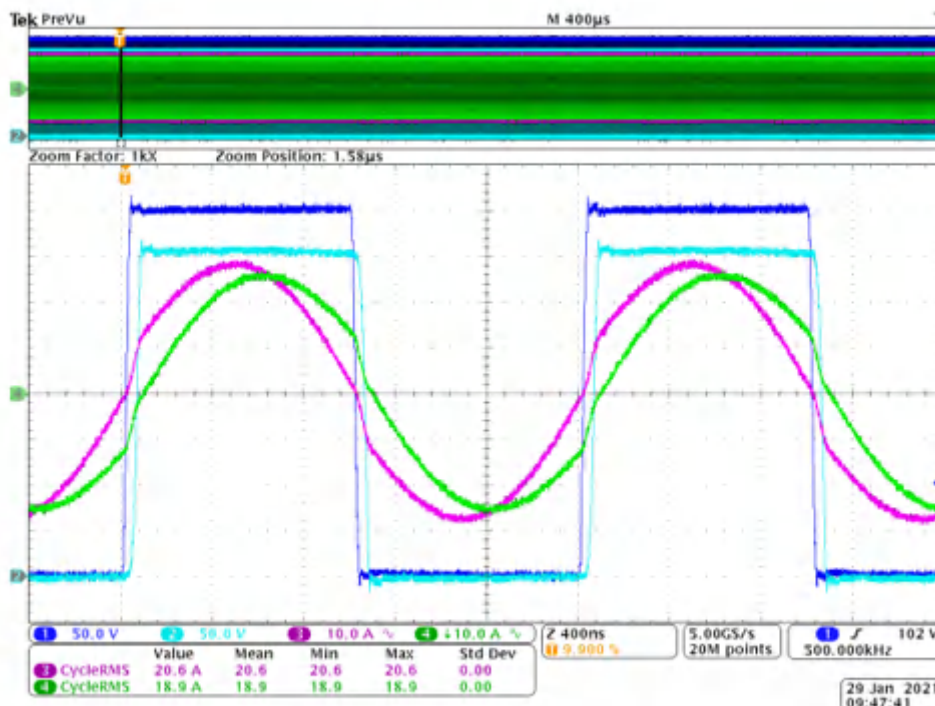


図 5-55. PFC の GaN ドレイン電圧 - 遷移

5.2.4.7 CLLLC の波形

図 1-1 に、次のパラメータを用いた 19A (6.6kW) での CLLLC 動作を示します。

- トレース
 - C1: GaN 1 次側スイッチ ノードのドレイン電圧
 - C2: GaN 2 次側スイッチ ノードのドレイン電圧
 - C3: トランス 1 次側電流
 - C4: トランス 2 次側電流
- 条件
 - $V_{IN} = 400V$
 - $V_{OUT} = 350V$
 - $I_{OUT} = 19A$



DPO4104B - 8:52:04 AM 1/29/2021

図 5-56. CLLLC 動作、19A (6.6kW)

図 1-1 に、次のパラメータを用いた 10A での CLLLC 動作を示します。

- トレース
 - C1: GaN スwitch のノードドレイン電圧レグ 1
 - C2: GaN スwitch のノードドレイン電圧レグ 2
 - C3: トランス 1 次側電流
 - C4: トランス 2 次側電流
- 条件
 - $V_{IN} = 400V$
 - $V_{OUT} = 350V$
 - $I_{OUT} = 10A$

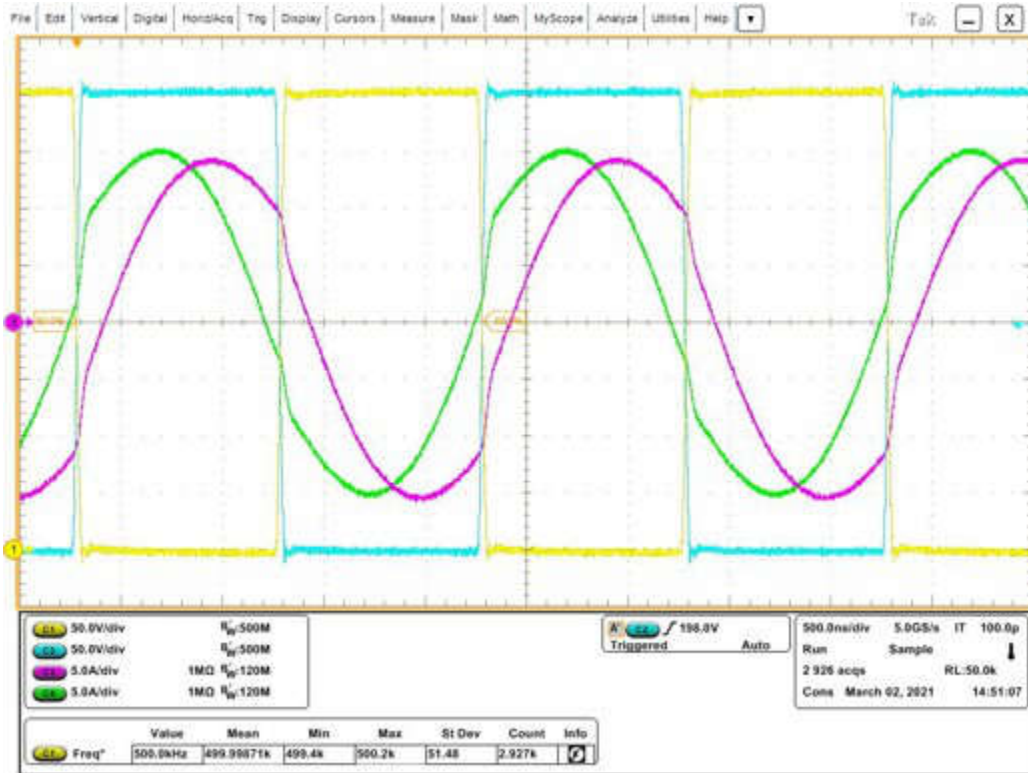


図 5-57. CLLLC 動作、10A

GaN スイッチのドレイン - ソース間の電圧遷移を拡大すると、[図 1-1](#) に示すように約 40 ns となります。このような急激な遷移は、LMG3522 の C_{OSS} が低いことに起因しています。

[図 1-1](#) の波形は、次のパラメータで測定されたものです。

- トレース
 - C1: GaN スイッチのノードドレイン電圧レグ 1
 - C2: GaN スイッチのノードドレイン電圧レグ 2
 - C3: トランス 1 次側電流
 - C4: トランス 2 次側電流
- 条件
 - $V_{IN} = 400V$
 - $V_{OUT} = 350V$
 - $I_{OUT} = 10A$

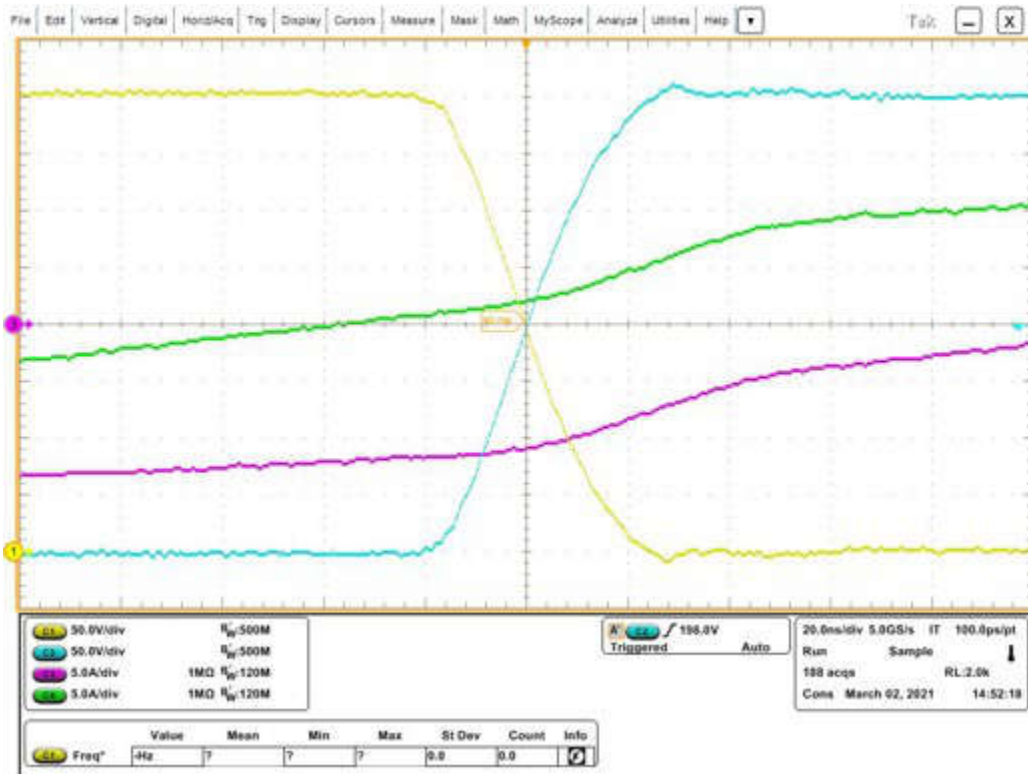


図 5-58. CLLLC 動作、10A - GaN FET の遷移

図 1-1 の波形は、次のパラメータで測定されたものです。

- トレース
 - C1: GaN スwitch のノードドレイン電圧レグ 1
 - C2: GaN スwitch のノードドレイン電圧レグ 2
 - C3: トランス 1 次側電流
 - C4: トランス 2 次側電流
- 条件
 - $V_{IN} = 400V$
 - $V_{OUT} = 350V$
 - $I_{OUT} = 2A$

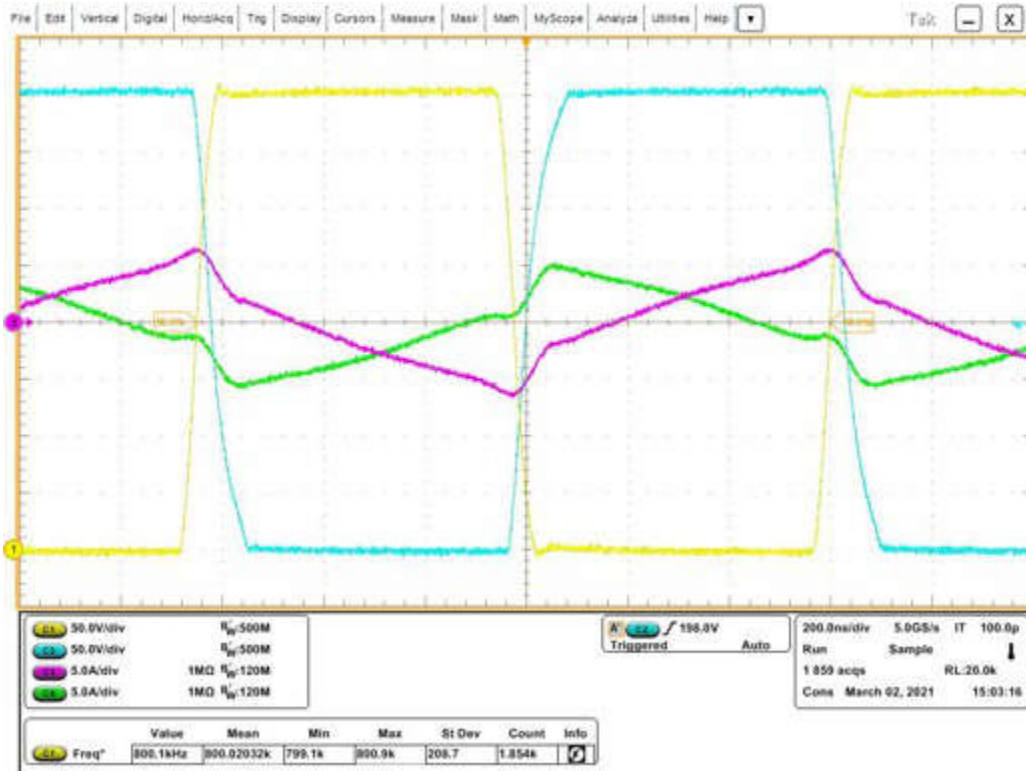


図 5-59. CLLC 動作、2A

GaN スイッチのドレイン - ソース間の電圧遷移を拡大すると、[図 1-1](#) に示すように約 **75 ns** となります。このような急激な遷移は、LMG3522 の C_{OSS} が低いことに起因しています。この図で遷移時間がわずかに長くなっているのは、負荷が軽く、その結果として電流量が減少しているためです。

[図 1-1](#) の波形は、次のパラメータで測定されたものです。

- トレース
 - C1: GaN スイッチのノードドレイン電圧レグ 1
 - C2: GaN スイッチのノードドレイン電圧レグ 2
 - C3: トランス 1 次側電流
 - C4: トランス 2 次側電流
- 条件
 - $V_{IN} = 400V$
 - $V_{OUT} = 350V$
 - $I_{OUT} = 2A$

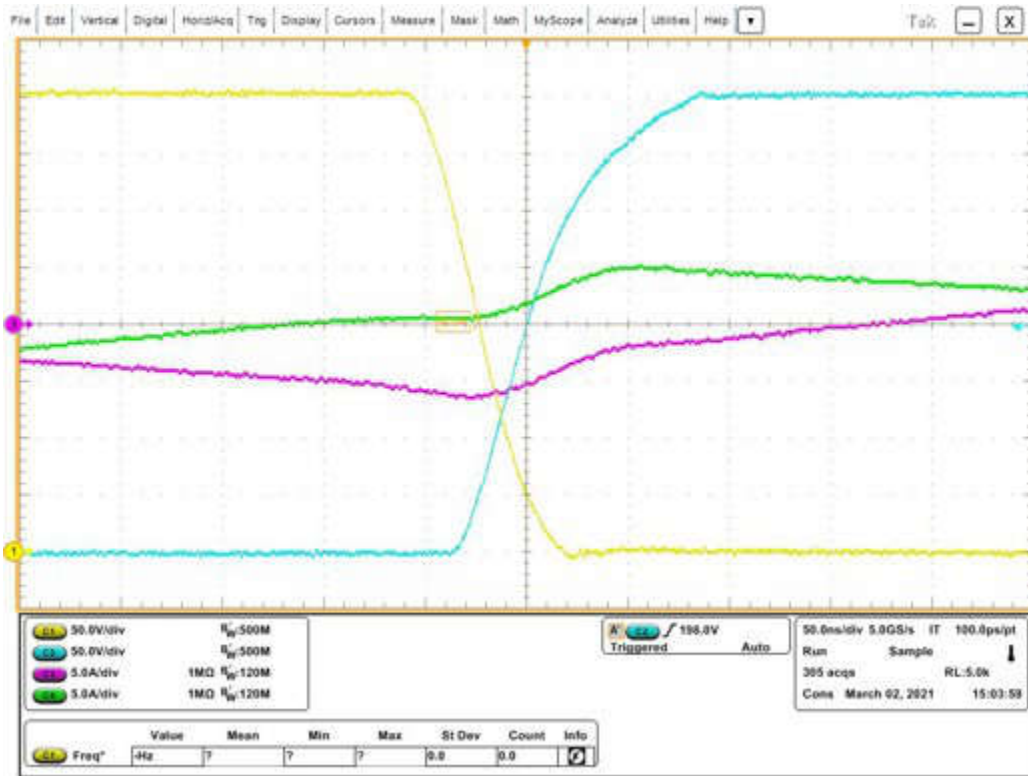


図 5-60. CLLLC 動作、2A - GaN FET の遷移

6 デザイン ファイル

6.1 回路図

回路図をダウンロードするには、[TIDM-02013](#) のデザイン ファイルを参照してください。

6.2 部品表 (BOM)

部品表 (BOM) をダウンロードするには、[TIDM-02013](#) のデザイン ファイルを参照してください。

6.3 Altium プロジェクト

Altium Designer® のプロジェクト ファイルをダウンロードするには、[TIDM-02013](#) のデザイン ファイルを参照してください。

6.4 ガーバー ファイル

ガーバー ファイルをダウンロードするには、[TIDM-02013](#) のデザイン ファイルを参照してください。

7 ソフトウェア ファイル

このリファレンス デザインのソフトウェアをダウンロードするには、[C2000](#) マイクロコントローラ向けデジタル電源ソフトウェア開発キット (SDK) のサイトにアクセスしてください。

8 関連資料

1. テキサス・インスツルメンツ、『[TMS320F28003x リアルタイム マイクロコントローラ](#)』データシート
2. テキサス・インスツルメンツ、『[SDK フレームワークでの C2000™ ソフトウェア周波数応答アナライザ \(SFRA\) ライブラリおよび Compensation Designer](#)』、ユーザー ガイド
3. Zaka Ullah Zahid, Zakariya M. Dalala, Rui Chen, Baifeng Chen, Jih-Sheng Lai, 『[V2G アプリケーションのための双方向 DC-DC 共振コンバータの設計](#)』、『IEEE Transactions on Transportation Electrification』、Vol. 1, No. 3, 2015 年 10 月、pp. 232~244
4. Zakariya M. Dalala, Zaka Ullah Zahid, Osama S. Saadeh, Jih-Sheng Lai, 『[双方向共振コンバータ バッテリ チャージャのモデリングとコントローラ設計](#)』、『IEEE Access』、Vol. 6, 2018 年 4 月、pp. 23338~23350
5. Biao Zhao, Qiang Song, Wenhua Liu, Yandong Sun, 『[高周波リンク電力変換システム用デュアル アクティブ ブリッジ絶縁型双方向 DC-DC コンバータの概要](#)』、Vol. 29, No. 8, 2014 年 8 月、pp. 4091~4106
6. Joel Turchi, J.T., Dhaval Dalal, D.D., Patrick Wang, P.T., Laurent Lenck, L.L.(2014 年)『[力率改善 \(PFC\) ハンドブック: 適切な力率コントローラ ソリューションの選択](#)』、リビジョン 5、<http://www.onsemi.com/pub/Collateral/HBD853-D.PDF>
7. テキサス・インスツルメンツ、『[トータムポール PFC における制御の課題](#)』、アナログ アプリケーション ジャーナル

8.1 商標

C2000™, テキサス・インスツルメンツの™, Piccolo™, TMS320C2000™, and Code Composer Studio™ are trademarks of Texas Instruments.

Altium Designer® is a registered trademark of Altium LLC or its affiliated companies.

すべての商標は、それぞれの所有者に帰属します。

9 用語

略称	定義
BCM	バッテリー充電モード
BW	帯域幅
CAN	コントローラ エリア ネットワーク
CCM	連続導通モード
CCS	Code Composer Studio
CLA	制御補償器アクセラレータ (CLA)
CLB	構成可能なロジック ブロック
CLLLC	コンデンサ、インダクタ、インダクタ、インダクタ、コンデンサ
CMPSS	コンパレータ サブシステム
CMTI	同相過渡耐性
DAB	デュアル アクティブ ブリッジ
DAC	D/A コンバータ
DCM	不連続導通モード
DLOG	データ ロガー
DT	デッド タイム
DUT (試験対象デバイス)	テスト対象の設計
eCAP	拡張キャプチャ
ePWM	拡張パルス幅変調器
ERAD	組み込みのリアルタイム分析および診断
EV	電気自動車
FET	電界効果トランジスタ
FHA	第 1 高調波分析
FSI	高速シリアル インターフェイス
GaN	窒化ガリウム
HEV	ハイブリッド電気自動車
HRPWM	高分解能パルス幅変調器
HSEC	高速エッジ カード
I2C	IC の相互接続
IDE に対応したプロジェクト例を含むソースコードを提供	統合開発環境
IGBT	絶縁ゲートバイポーラトランジスタ
ISR	割り込みサービス ルーチン
KCL	キルヒホッフの電流則
KVL	キルヒホッフの電圧則
LIN	ローカル相互接続ネットワーク
MOSFET	金属酸化膜半導体電界効果トランジスタ
OBC	オンボード チャージャ
PFC	力率補正
PGA	プログラマブル ゲイン アンプ
PMBus	パワー マネージメント バス
SCI	シリアル通信インターフェイス
SDFM	シグマ-デルタ フィルタ モジュール
SFRA	ソフトウェア周波数応答アナライザ
SiC	シリコン カーバイド
SPI	シリアル パリフェラル インターフェイス
SRC	直列共振コンバータ
ZCS	ゼロ電流スイッチング
ZVS	ゼロ電圧スイッチング

10 著者について

Cody Watkins は、テキサス・インスツルメンツにおける C2000 マイクロコントローラ グループのアプリケーション エンジニアです。2015 年にシンシナティ大学で電気工学の学士号を取得しており、代替エネルギー、持続可能性、自給自足エネルギーなどに高い関心があります。

11 改訂履歴

資料番号末尾の英字は改訂を表しています。その改訂履歴は英語版に準じています。

Changes from Revision * (October 2022) to Revision A (February 2024)

Page

-
- このデザインに TMS320F28P65x マイクロコントローラによるソフトウェア サポートを追加。..... **1**
-

重要なお知らせと免責事項

テキサス・インスツルメンツは、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、テキサス・インスツルメンツ製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した テキサス・インスツルメンツ製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている テキサス・インスツルメンツ製品を使用するアプリケーションの開発の目的でのみ、テキサス・インスツルメンツはその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。テキサス・インスツルメンツや第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、テキサス・インスツルメンツおよびその代理人を完全に補償するものとし、テキサス・インスツルメンツは一切の責任を拒否します。

テキサス・インスツルメンツの製品は、[テキサス・インスツルメンツの販売条件](#)、または [ti.com](https://www.ti.com) やかかる テキサス・インスツルメンツ製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。テキサス・インスツルメンツがこれらのリソースを提供することは、適用されるテキサス・インスツルメンツの保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、テキサス・インスツルメンツはそれらに異議を唱え、拒否します。

郵送先住所: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2024, Texas Instruments Incorporated

重要なお知らせと免責事項

TI は、技術データと信頼性データ(データシートを含みます)、設計リソース(リファレンス・デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、TI 製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した TI 製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとし、

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている TI 製品を使用するアプリケーションの開発の目的でのみ、TI はその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。TI や第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、TI およびその代理人を完全に補償するものとし、TI は一切の責任を拒否します。

TI の製品は、[TI の販売条件](#)、または [ti.com](#) やかかる TI 製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。TI がこれらのリソースを提供することは、適用される TI の保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、TI はそれらに異議を唱え、拒否します。

郵送先住所 : Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2024, Texas Instruments Incorporated