

# Using the UCD92xx Digital Point-of-Load Controller

## Design Guide



Literature Number: SLUU490

April 2011



<b>Preface</b> .....	<b>9</b>
<b>1 Quick-Start Guide</b> .....	<b>13</b>
1.1 Procedure .....	14
1.2 Summary .....	21
<b>2 Circuit Considerations Using the UCD92xx</b> .....	<b>23</b>
2.1 Circuit Board Design .....	24
2.1.1 Power Stage Schematics .....	24
2.1.2 Differential Sense Lines .....	25
2.1.3 3.3-V and 1.8-V Core Voltage Decoupling .....	26
2.2 Temperature Monitoring .....	28
2.2.1 Internal Temperature Sense .....	28
2.2.2 External Temperature Sense .....	28
2.2.3 Thermistor Temperature Sense .....	30
2.3 Current Monitoring .....	31
2.3.1 Calculation of the Current Sense Voltage Using Inductor DCR .....	32
2.3.2 Calculation of Attenuation Resistor $R_{ATT}$ .....	33
2.4 Inductor Derating Used by the Fusion GUI .....	33
2.5 Determining IOUT_CAL_GAIN and IOUT_CAL_OFFSET .....	34
2.6 Calibration of IOUT_CAL_GAIN and IOUT_CAL_OFFSET .....	34
2.7 Filtering Requirements on the UCD723x CS Outputs .....	36
2.8 Filtering Requirements on the UCD7242 and UCD74106 CS Outputs .....	36
2.9 Calculation of UCD7230A ILIM Voltage .....	37
2.10 Calculation of UCD7231 ILIM Voltage .....	38
<b>3 Voltage Regulation</b> .....	<b>39</b>
3.1 Loop Compensation .....	40
3.1.1 Total Loop Gain .....	40
3.2 Implementation of the Compensating Filter $H(s)$ .....	41
3.2.1 Compensating Filter Architecture .....	43
3.2.2 Fixed-Point Math .....	44
3.2.3 Cascaded First-Order Filter .....	45
3.2.4 Type II versus Type III Compensation .....	45
3.3 Modeling Delay in the Control Loop .....	45
3.4 Phase Loss as a Result of On-Time and Multiple Power Stages .....	47
3.5 Quantization Effects .....	48
3.6 Anti-Saturation Feature .....	48
3.7 Autotune Operation .....	49
3.7.1 Determining the Compensating Zeros and Compensating Gain .....	50
3.7.1.1 Caveats .....	51
3.8 How to Tweak the Compensation .....	51
3.8.1 Tuning by Pole/Zero Placement .....	51
3.8.2 Tuning by PID Gains .....	53
3.9 Making Loop Transfer Function Measurements .....	54
3.9.1 Types of Excitation .....	54
3.9.2 Generation of Sinewave Excitation .....	54
3.9.3 Response Measurement Implementation .....	55

<b>4</b>	<b>Start-Up Sequencing</b> .....	<b>57</b>
4.1	Sequencing .....	58
4.2	Start Indications .....	58
4.3	TON_DELAY and TON_RISE .....	60
4.4	Group Commands .....	61
4.5	Daisy-Chaining CTRL and PowerGood Pins .....	62
4.6	Voltage Tracking .....	62
4.7	GPIO_SEQ_CONFIG .....	64
4.7.1	Configuration for Sequencing Output .....	65
4.7.2	Configuration for Sequencing Inputs .....	67
4.7.3	Fault Slaves .....	68
4.7.4	Sequencing Timeout .....	70
4.8	Alternative/Additional PowerGood Outputs .....	71
<b>5</b>	<b>Device Operation</b> .....	<b>73</b>
5.1	AFE Overview .....	74
5.2	Synchronizing Multiple PWM Signals .....	75
5.2.1	Description of the Digital Demodulator Logic .....	76
5.2.2	Effect of EV1 .....	77
5.2.3	Setting EADC_SAMPLE_TRIGGER Value .....	77
5.3	The Soft-Start Ramp .....	78
5.3.1	Starting into a Prebias State .....	79
5.3.2	Effect of a Large DRIVER_MIN_PULSE .....	80
5.4	PWM Timing .....	80
5.5	Inrush Current During Kick-Start .....	80
5.5.1	Inductor Ripple Current .....	83
5.6	Current Balance .....	83
5.7	Temperature Balance .....	86
<b>6</b>	<b>Using the Fusion Designer Software</b> .....	<b>87</b>
6.1	Downloading the Project File .....	88
6.1.1	Communicating to a Previously Configured UCD92xx Device .....	90
6.2	Copying One Design to Another Voltage Rail .....	90
6.2.1	Method 1: Copy Coefficients .....	91
6.2.2	Method 2: Copy a Design .....	92
6.3	Design@Device Synchronization .....	92
<b>A</b>	<b>PMBus Communication</b> .....	<b>95</b>
A.1	Overview .....	95
A.2	LINEAR11 Format .....	96
A.3	Clock Stretching .....	97
<b>B</b>	<b>Locating the Compensating Zeros</b> .....	<b>99</b>
B.1	Overview .....	99

## List of Figures

1.	UCD9248 Block Diagram .....	10
1-1.	Fusion Digital Power Designer Software GUI: Startup Screen (Offline Mode) .....	14
1-2.	UCD92xx Naming Convention .....	14
1-3.	Fusion GUI: Phase Assignment Selection Menu .....	15
1-4.	Fusion GUI: Voltage Configuration Menu .....	15
1-5.	Fusion GUI: Start Delay and Soft-Start Ramp Time Configuration to Illustrate PMBus Commands .....	16
1-6.	Fusion GUI: Turn-On and Turn-Off Timing (Voltage Configuration) .....	17
1-7.	Fusion GUI: Tooltip Example .....	17
1-8.	Fusion GUI: $I_{OUT}$ , $V_{IN}$ , and Temperature Configuration .....	17
1-9.	Fusion GUI: Design Menu .....	18
1-10.	Fusion GUI: Design Schematic Example .....	19
1-11.	Fusion GUI: Bode Plots and Simulation Graphs .....	20
2-1.	Typical Multi-Voltage-Rail Board .....	24
2-2.	Single Board $V_{OUT}$ Sense .....	25
2-3.	10- $\Omega$ Backup Circuit for Remote Sense .....	26
2-4.	Recommended Decoupling Capacitor Layout .....	27
2-5.	Providing 3.3-V Power for the Controller .....	27
2-6.	External Temperature Sense Multiplexer .....	29
2-7.	Thermistor Circuit .....	30
2-8.	Thermistor Circuit Response .....	31
2-9.	Block Diagram of the 12-Bit ADC Used for Monitoring .....	31
2-10.	Current Sense Schematic: Direct Current Resistance (DCR) Current Sense .....	32
2-11.	Adding an Additional Load to Check/Calibrate $I_{OUT\_CAL\_GAIN}$ .....	35
2-12.	Current Sense for Drivers with Internal MOSFETS .....	36
2-13.	Setting UCD7230A CLF/FLT Threshold .....	37
3-1.	UCD92xx Fusion Peripheral Block Diagram .....	40
3-2.	UCD92xx Fusion Peripheral Block Diagram: Compensating Filter Overview .....	43
3-3.	Nonlinear Compensation Block and Default Boost Characteristic .....	44
3-4.	PWM Computational Delay .....	46
3-5.	Latency for an Example Three-Phase Power Supply .....	47
3-6.	Anti-Saturation Feature Activation and Recovery .....	48
3-7.	Default Non-Linear Boost Controls .....	49
3-8.	Autotune Criteria .....	50
3-9.	Fusion GUI: PID Gains Control .....	53
3-10.	Transfer Function Analysis Block Diagram .....	54
3-11.	Modeled versus Measured Power Stage Transfer Function .....	56
4-1.	Fusion GUI: ON_OFF_CONFIG Options .....	58
4-2.	Fusion GUI: OPERATION Command Options .....	59
4-3.	Fusion GUI: Setting TON_DELAY and TON_RISE .....	61
4-4.	Daisy-Chain Sequence Example .....	62
4-5.	Fusion GUI: Voltage Tracking Control Options .....	63
4-6.	Tracking Example .....	63
4-7.	Fusion GUI: GPIO Config Tab .....	66
4-8.	Fusion GUI: Dependency Control Options .....	67
4-9.	Fusion GUI: Fault Shutdown Slave Control .....	68
4-10.	Fault on Rail #1; Rail #2 has a Rail #1 Stay-on Dependency with Soft-Off .....	69
4-11.	Fault on Rail #1; Rail #2 has a Rail #1 Stay-on Dependency with Immediate-Off .....	69

---

4-12.	Fault on Rail #1—Rail #2 is a Fault Slave for Rail #1; Shutdown After Retries.....	69
4-13.	Fusion GUI: Sequencing Timeout Control Screen .....	70
5-1.	UCD92xx Analog Front-End .....	74
5-2.	Synchronization Options.....	75
5-3.	PWM Timing with Internal Synchronization .....	76
5-4.	PWM Timing with External Synchronization .....	77
5-5.	Soft-Start Ramp.....	78
5-6.	Soft-Start Ramp: $V_{PREBIAS}$ Less Than $V_{START}$ .....	79
5-7.	Soft-Start Ramp: $V_{PREBIAS}$ Greater Than $V_{START}$ .....	79
5-8.	Soft-Start with Large DRIVER_MIN_PULSE .....	80
5-9.	Simplified Schematic for Estimating Inrush Current .....	81
5-10.	Voltage and Current during the <i>Kick-Start</i> Period.....	83
5-11.	Example Current Balance Action for a Dual-Phase Voltage Rail .....	85
6-1.	Fusion GUI: Project File Download Menu.....	88
6-2.	Fusion GUI: Project File Selection .....	89
6-3.	Fusion GUI: Project File Configuration .....	89
6-4.	Fusion GUI: CLA Coefficients Banks Tab.....	91
6-5.	Fusion GUI: Copy Design Window.....	92
A-1.	TON_DELAY and TON_RISE .....	95
A-2.	Trace C: PMBus Clock, Trace D: PMBus Data.....	97
B-1.	Locus of Zeroes with Changing Q .....	100
B-2.	Real and Imaginary Frequency vs Q .....	101

## List of Tables

2-1.	Example Assignment of PWM Outputs .....	29
2-2.	Resulting Power Stage to Multiplexer Address Assignment .....	29
4-1.	Tracking Options.....	62
4-2.	Available UCD92xx Sequencing Pins.....	65
4-3.	Output Dependency Configuration Bytes .....	71
5-1.	AFE settings.....	74
5-2.	Example Circuit Component Values .....	82
5-3.	Current Balance Update Rates .....	84
B-1.	Example Spreading Factor ( $\beta$ ) Values .....	100

Design is a registered trademark of Texas Instruments.  
ARM7 is a trademark of ARM Ltd.  
I<sup>2</sup>C is a trademark of NXP Semiconductors.  
PMBus is a trademark of SMIF, Inc.  
All other trademarks are the property of their respective owners.



## ***Introduction***

---

---

---

### **Overview**

The [UCD92xx](#) family of digital pulse-width modulation (PWM) controllers are multi-rail, multi-phase synchronous buck digital devices designed for non-isolated dc/dc power applications. These controllers integrate dedicated circuitry for dc/dc loop management with flash memory and a serial interface to support configurability, monitoring, and management. These devices provide a wide variety of desirable features for non-isolated dc/dc converter applications while minimizing the total system component count by reducing external circuits. The solution integrates multi-loop management with sequencing, margining, tracking, and intelligent phase management to optimize the solution for total system efficiency. Additionally, loop compensation and calibration are supported without the need for additional external components. See [Figure 1](#) for a block diagram of the [UCD9248](#) as an example.

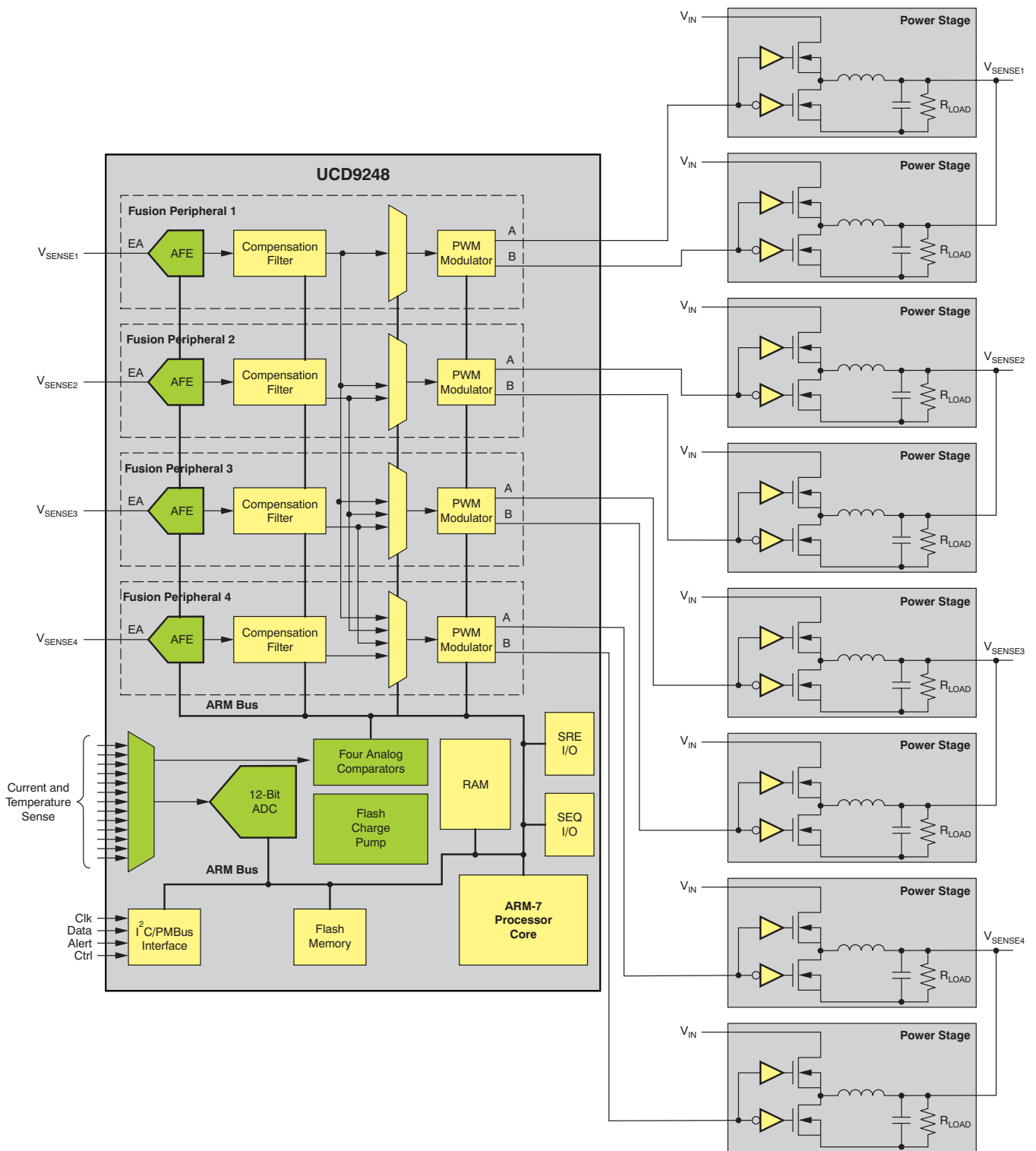


Figure 1. UCD9248 Block Diagram

The UCD92xx digital PWM controllers are PMBus™-compliant devices; they comply with and operate according to the [PMBus version 1.1 or version 1.2](#) standards. The PMBus standard states that all configuration commands received by a device are first stored in volatile memory (RAM or hardware registers). These settings take effect immediately. To permanently configure the device, the PMBus command **STORE\_DEFAULT\_ALL** is issued. This command instructs the device to store all volatile configuration settings into nonvolatile memory (flash). In addition, the [Fusion Digital Power Designer](#) software graphical user interface (GUI) also maintains a level of configuration information called *design information* in the form of a project file that is stored as a Windows file.

To send design information to the device from the GUI, a **Write to Hardware** operation command is issued (normally by pressing a button on the left hand side of the GUI screen). The actual PMBus commands sent are shown in the log at the bottom right of the GUI. You must execute this command each time you make a design change that affects the configuration. To permanently store the configuration into the device, you must issue a **Store RAM to Flash** command. (This operation is always available under the *Device* drop-down menu.) To save your particular design information, which includes the power stage schematic that is not downloaded to the device, save the parameters from the Fusion GUI into an .xml project file.

The normal process of designing a power supply based on the UCD92xx series is to first create a project file using the GUI in *off-line mode* (that is, the software is not connected to a hardware device). In this mode, users select the voltage, switching frequency, rated current, and power stage circuit for each regulated voltage rail to be controlled by the UCD92xx device. Use the recommended component values and/or part numbers suggested by the Fusion software for the power stage circuits in order to complete the board schematic and layout. Once these parameters are established, fabricate and assemble the board with the related components.

Then apply power to the completed printed circuit board (PCB) and connect the PMBus to a PC with the *Fusion Digital Power Designer* GUI installed. The PC is generally connected to the PMBus with the [TI USB Adapter](#). An unconfigured UCD92xx controller has all outputs disabled until the configuration stored in the previously created project file is downloaded to the device. Depending on the setting of ON\_OFF\_CONFIG, the power supply is now ready to accept a command to start regulating each controlled voltage rail. Once you have confirmed that each voltage rail can come up to the regulated voltage without errors, store any changes made to the configuration to the device flash memory, and then store the configuration to the project file.

Once the UCD92xx devices are configured to provide basic regulated power, they can be integrated into the overall management function of a system by communicating over the PMBus (with an I<sup>2</sup>C™ interface) from a microcontroller or FPGA. There are over 120 commands implemented in the device. The standard command syntax is found in the [PMBus standard](#). The commands that are unique to the UCD92xx devices are found in the [UCD92xx Command Reference](#), available for download at [www.ti.com](http://www.ti.com).

### **Information About Cautions and Warnings**

This document contains caution statements.

#### **CAUTION**

This is an example of a caution statement. A caution statement describes a situation that could potentially damage your software or equipment.

The information in a caution or a warning is provided for your protection. Please read each caution carefully.

## Applications Questions

There is a dedicated discussion forum for this device at [http://e2e.ti.com/support/power\\_management/digital\\_power/default.aspx](http://e2e.ti.com/support/power_management/digital_power/default.aspx) on the TI website.

If you have questions about other Texas Instruments PWM controllers, post a question in the broader *Power* forum at <http://e2e.ti.com>. Include in the subject heading the product in which you are interested.

## Acknowledgements

Much of the contents of this application note are based on correspondence between the design team at Texas Instruments and early customers of the device family. This information includes contributions on the operation of the PMBus interface from Karl Northrup; contributions on the behavior of the monitoring features from Mark Heminger; contributions on sequencing from Eric Oettinger; and descriptions of the operation of the Fusion Digital Power Designer software from Mike Muegel. I am also indebted to Brad Higgins, TI factory applications, for a review of the material, and to the work of the field application engineers who have helped numerous customers to use this device in their respective applications.

## Electrostatic Discharge Warning

Many of the components on the UCD92xx are susceptible to damage by electrostatic discharge (ESD). Customers are advised to observe proper ESD handling precautions when unpacking and handling this device, including the use of a grounded wrist strap at an approved ESD workstation.

### CAUTION

Failure to observe ESD handling procedures may result in damage to the device.

## Quick-Start Guide

---

---

---

This chapter presents 10 steps that show how to generate a project file that you can use to configure a device with the Fusion Digital Power Designer program.

Topic	Page
1.1 Procedure .....	14
1.2 Summary .....	21

## 1.1 Procedure

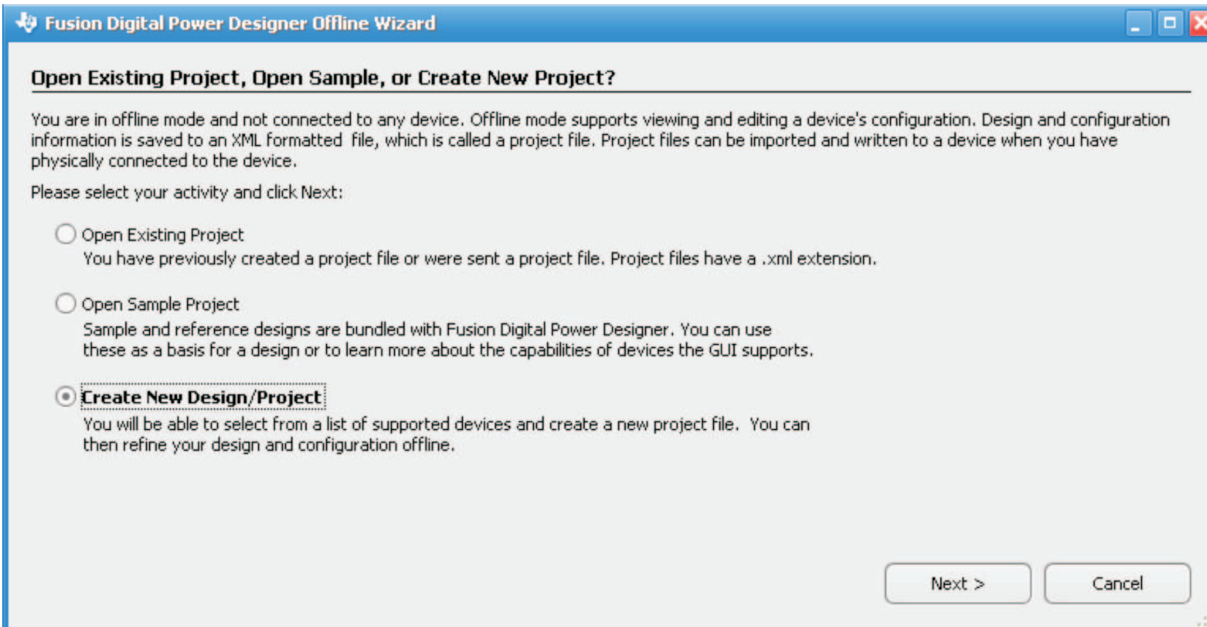
Follow these steps to set up, configure, and create a project design file for use with the UCD92xx series of PWM controllers.

### Step 1.

Download and install the latest version of the [Fusion Digital Power Designer](#) PC program, referred to as the **Fusion GUI** throughout this document.

### Step 2.

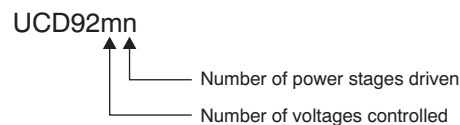
After installation, start the Fusion GUI. The Fusion GUI launches in offline mode, and prompts you to select the type of project that you wish to work on. Select *Create a New Design*, as [Figure 1-1](#) shows.



**Figure 1-1. Fusion Digital Power Designer Software GUI: Startup Screen (Offline Mode)**

### Step 3.

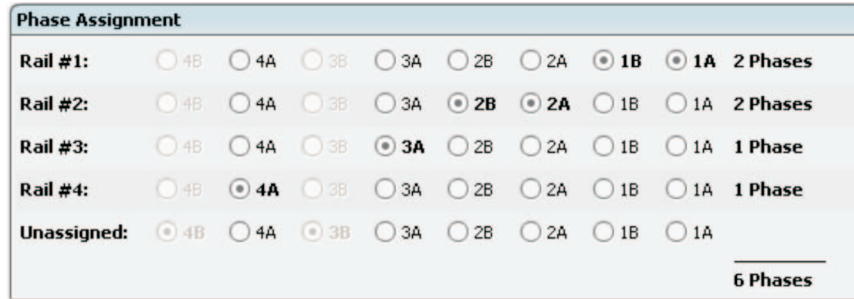
Pick the specific device you want to work with. Note that for most of the UCD92xx devices, the part number itself denotes the number of voltages that can be regulated and the number of power stages that can drive those voltage rails, as shown in [Figure 1-2](#).



**Figure 1-2. UCD92xx Naming Convention**

**Step 4.**

After you select the controller device, the GUI prompts you to define which PWM outputs are assigned to the respective  $V_{OUT}$  sense inputs. Note that in the Fusion GUI software, a MOSFET half-bridge is a *phase* and a regulated voltage is a *rail*. Figure 1-3 shows the phase assignment selection menu.

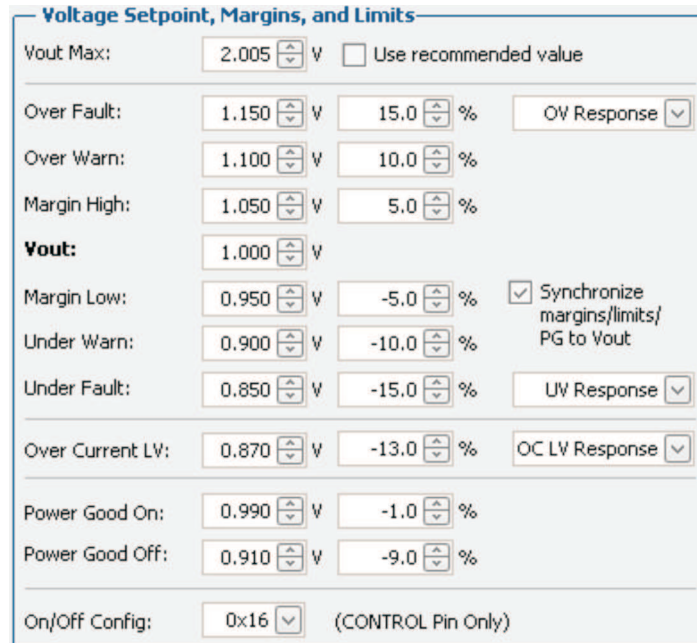


**Figure 1-3. Fusion GUI: Phase Assignment Selection Menu**

Regulated voltage Rail #1 is controlled by sensing its voltage at the  $EA_{P1}/EA_{N1}$  pins; voltage Rail #2 is controlled by sensing its voltage at  $EA_{P2}/EA_{N2}$ , and so on. The underlying PMBus command for this configuration is the PHASE\_INFO command.

**Step 5.**

The GUI then takes you to the *Vout Config* page. This screen allows users to define the nominal regulated output voltage for each rail. The Fusion GUI then uses this value and a set of default percentages to define margin-high, margin-low, overvoltage (OV) warn level, OV fault level, undervoltage (UV) warn level, UV fault level, etc. Each of these output voltage related commands can be overridden with user-specific values. For this discussion, we simply select the default percentages presented, as Figure 1-4 shows.



**Figure 1-4. Fusion GUI: Voltage Configuration Menu**

**Step 6.**

Next, you must determine the event(s) that tell your power supply to start. The PMBus standard identifies four events that can initiate regulation:

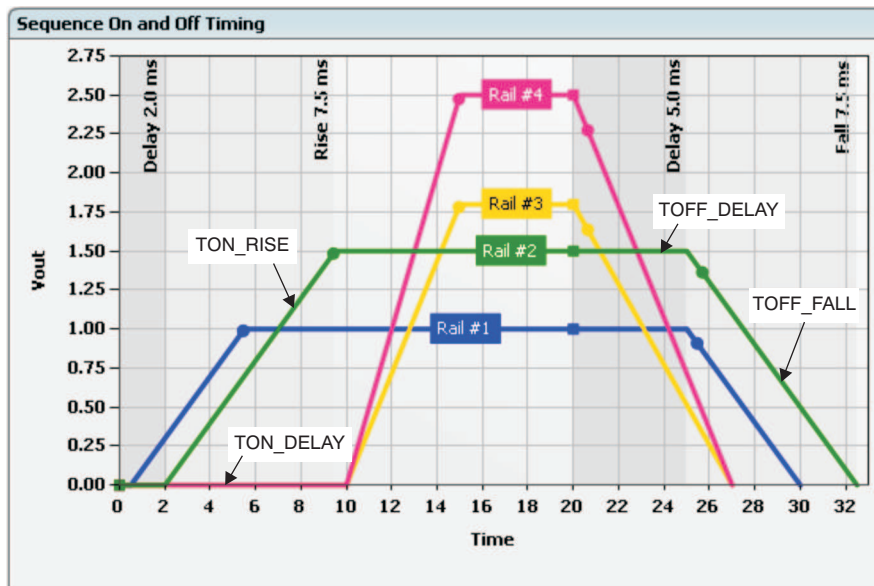
- The input voltage is greater than a configurable threshold.
- The CTRL line to the device goes active.
- The device receives an OPERATION command on the PMBus.
- The device receives an OPERATION command on the PMBus, *and* the CTRL line is active.

Each regulated voltage rail can be configured to have a different event initiate regulation. To configure the specific event that starts a particular voltage rail, click on the *On/Off Config* edit box; the Fusion GUI presents the list of events to select from.

To prepare for the initial device power-on, it is a good idea to set the On/Off Configuration to start on an OPERATION command. With this option, the output voltages are off when power is first applied to the board. Each voltage rail can then be turned on separately using the Fusion GUI. After you have tested each voltage rail, you can change the On/Off configuration to start with  $V_{IN}$  or with the CTRL signal.

Set up the desired sequencing arrangement between the voltage rails. The UCD92xx controllers can perform complex sequencing procedures where one regulated voltage rail depends on other voltage rails and does not start until these rails are actively regulating. The sequencing can also depend on an external input signal. Additionally, a regulated voltage rail can be configured to track a second voltage under closed-loop control until the voltage being tracked exceeds the configured voltage for the regulated voltage rail. These features are reviewed in another section of this document.

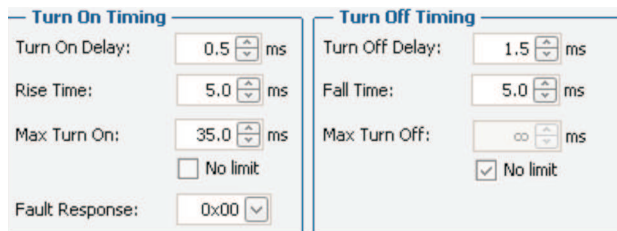
The simplest way to sequence several regulated voltage rails is to set the start delay and soft-start ramp time for each rail. The PMBus commands that set these timings are TON\_DELAY, TON\_RISE, TOFF\_DELAY, and TOFF\_FALL. **Figure 1-5** illustrates these PMBus commands.



**Figure 1-5. Fusion GUI: Start Delay and Soft-Start Ramp Time Configuration to Illustrate PMBus Commands**

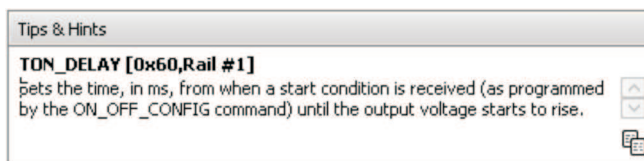


In the Fusion GUI, these values are entered in the *Turn On Timing* and *Turn Off Timing* boxes at the right of the *Vout Config* page, as shown in [Figure 1-6](#).



**Figure 1-6. Fusion GUI: Turn-On and Turn-Off Timing (Voltage Configuration)**

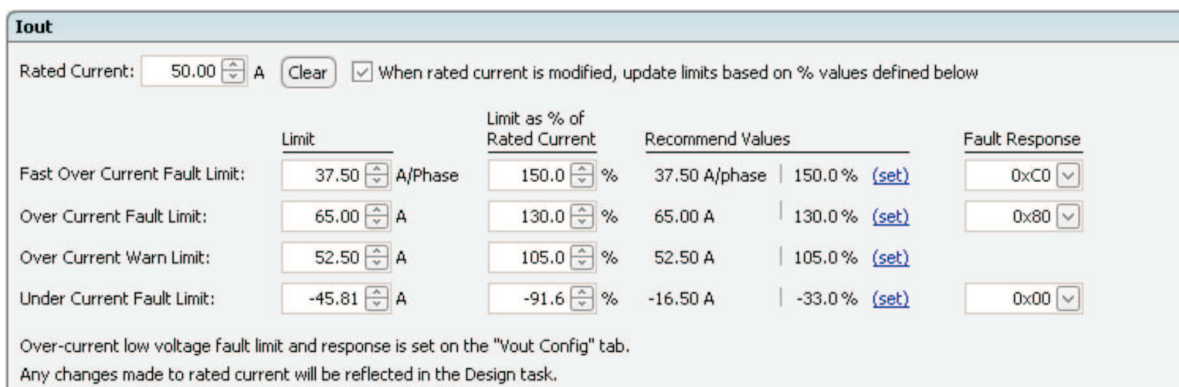
A detailed description of the underlying PMBus command and the range of allowable values for that command are displayed in the *Tips and Hints* window (such as [Figure 1-7](#) illustrates) at the bottom of the screen when you hover the cursor over an editable parameter.



**Figure 1-7. Fusion GUI: Tooltip Example**

**Step 7.**

Next, go to the *Iout*, *Vin*, *Temp Config* tab of the **Configure** page. [Figure 1-8](#) shows this tab. You must define a *rated current* for the output voltage rail. This value is the maximum current for the power stage circuit that you will create later. Rated current is not a recognized PMBus command, but the Fusion GUI uses the concept of rated current to set the current-related PMBus commands such as OC warn level, OC fault level, voltage foldback setting, etc. The Fusion GUI also uses the rated current parameter to select various component values in the current sense circuits.



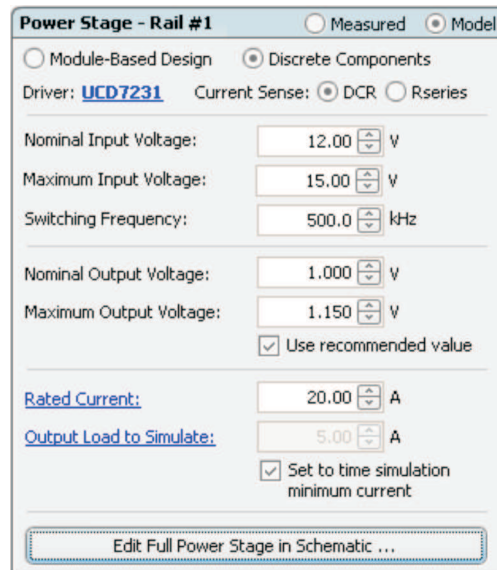
**Figure 1-8. Fusion GUI: I<sub>OUT</sub>, V<sub>IN</sub>, and Temperature Configuration**

A small icon with a **U** next to a given parameter setting indicates that the parameter has not yet been downloaded to the device. (The **U** means *Undo*.) Clicking this icon undoes the change. Pressing the **Write to Hardware** button at this point (or at any point in the process) sends any changes made to the configuration to the project file; if the software is connected to an active UCD controller, the **Write to Hardware** button sends the changes to the device registers as well. All PMBus commands sent when the **Write to Hardware** button is pressed are logged and can be retrieved for subsequent review. Note that when you switch between rails, any pending configurations are automatically written to the project file or device.

**Step 8.**

Go to the *Design* page of the GUI. This page requires that users specify the specific power stage that is to be used (a PTD power module or UCD7xxx-based circuit).

First, select whether the power stage is a module or consists of discrete components, as shown in [Figure 1-9](#). Power modules contain a gate driver IC, MOSFETs, an inductor, input capacitor, and output capacitor.<sup>(1)</sup> If you choose to develop the power stage circuit yourself, select *Discrete Components*, then select the technique you plan to use to measure current. Last, select the *DCR* option, unless your current monitoring requirements are very stringent; this option does not reduce power-supply efficiency.

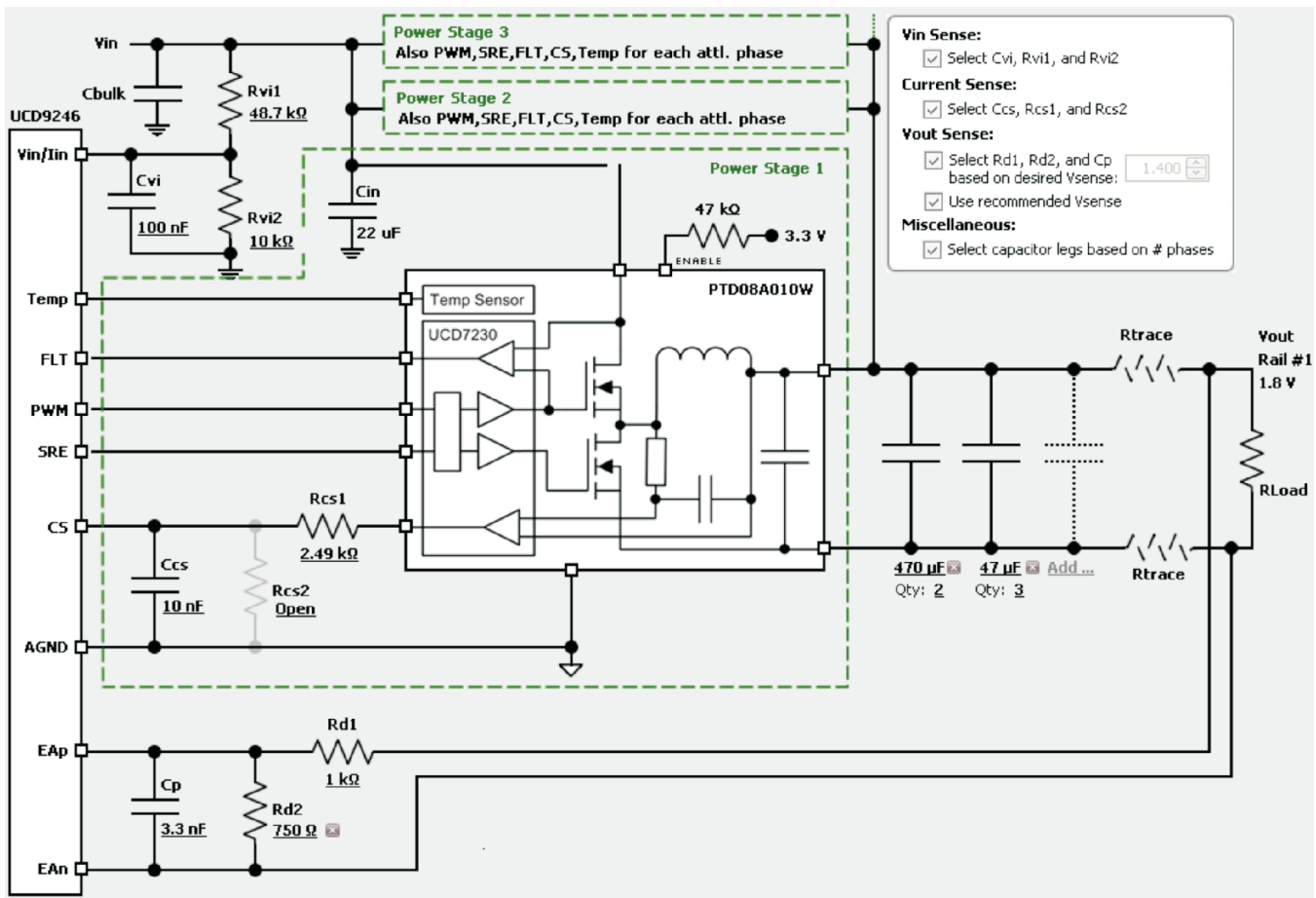


**Figure 1-9. Fusion GUI: Design Menu**

<sup>(1)</sup> The benefit of a choosing a power module is that it gives users the option of using an engineered layout of the circuits around the power MOSFETs, including the critical routing of ground currents between the MOSFETs and the input capacitor. However, this type of layout generally has an increased cost compared to a user-designed layout.

**Step 9.**

Now open the schematic by clicking the **Edit Full Power Stage in Schematic...** button. The design schematic then opens, as illustrated in [Figure 1-10](#).



**Figure 1-10. Fusion GUI: Design Schematic Example**

At this point, users can select different inductors, insert additional output capacitance, and so forth. However, if you leave all the boxes in the upper right corner checked, each component is selected for you by the GUI and the design is complete (if you have selected a driver with internal MOSFETs; refer to [Figure 1-9](#)). If you choose a design with external MOSFETS, you must select those components yourself by clicking on the part numbers for Q1 and Q2.

Upon closing the design schematic, the Fusion GUI performs an *autotune* process to select a set of digital filter coefficients that properly compensate the voltage regulation loop for the power stage circuit specified by the schematic. The autotune algorithm attempts to minimize the closed-loop output impedance for the given schematic.

**NOTE:**

The compensation design is not written to the project file or to an attached device until the **Write to Hardware** button is pressed. This architecture allows users to try various loop compensation settings without concern for making the actual power-supply circuit unstable. A **Write to Hardware** operation must be performed for each rail. This requirement is unique to the design page. On the configuration page, when you switch from one rail to the next, any pending configuration changes are automatically applied.

In addition to setting loop compensation, the Fusion GUI also issues the PMBus commands that set the scaling for sensing  $V_{IN}$ ,  $V_{OUT}$ ,  $I_{OUT}$ , temperature, and so forth, based on the circuit defined in the schematic. This software feature is controlled by the *Design→Device Synchronization* checkboxes. A more detailed description of this behavior is presented later in this document.

Now check the Bode plots and time simulation graphs of a load step (25% to 75% of rated current) on the design page in the GUI. If desired, the user can switch to manual compensation mode and make fine adjustments to the loop compensation in order to improve the step response. The compensation filter coefficients are sent to the device using the CLA\_GAINS command. Figure 1-11 shows the example Bode plots and simulation graphs.

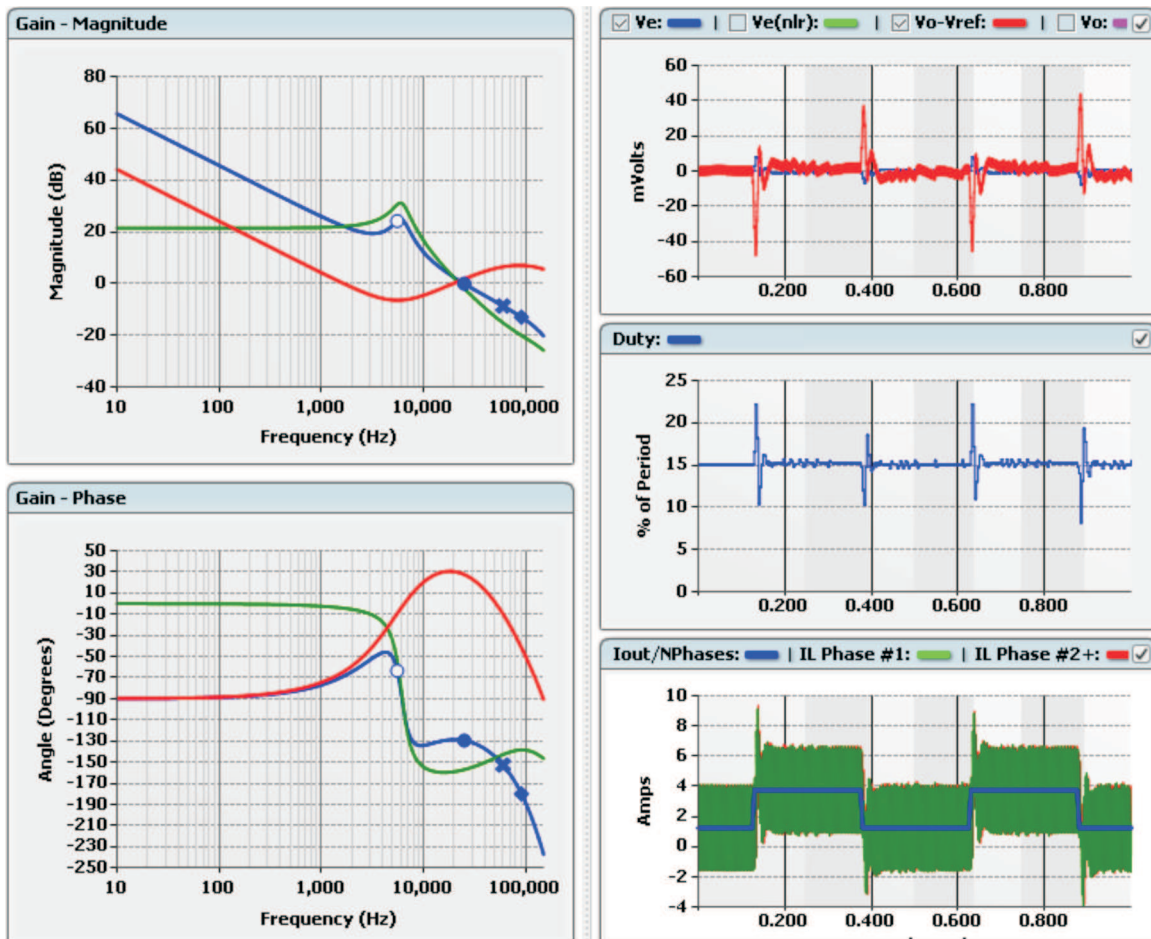


Figure 1-11. Fusion GUI: Bode Plots and Simulation Graphs

### Step 10.

Once you are satisfied with the configuration, save it to a project file; use the *File/Save Project* menu command. If the GUI is connected to an actively powered device, selecting the **Store RAM to Flash** button issues a STORE\_DEFAULT\_ALL PMBus command and saves your configuration to nonvolatile memory on the device.

## 1.2 Summary

The project file created in these 10 steps is the basis for operating your power-supply circuit hardware. Use the components recommended by the GUI to create the schematic and layout drawings that will be used to fabricate and populate your PCB assembly (PCBA). When the completed and assembled PCBA is ready, connect the controller PMBus to a PC using the [TI USB Adapter](#). Apply power to the board and start the Fusion Digital Power Designer GUI. If the controller has never been configured, all the PWM outputs are disabled and no regulation occurs.

Import the project file into the GUI. Depending on the setting of the ON\_OFF\_CONFIG step (see [Step 6](#)), the power supply is now ready to accept a command to start regulating each controlled voltage rail. When you have confirmed that each voltage rail is active and error-free, save any final changes to the configuration to the device flash memory and then save the configuration to the project file.

The balance of this document describes the underlying functions of the Fusion Digital Power Designer software. For most applications, the Fusion GUI does the work of designing a complete power system; it is not necessary to understand all the details presented here. However, if a given power system is exhibiting behavior different from what is expected based on the software, it may be helpful to have a complete description of the design equations.



## ***Circuit Considerations Using the UCD92xx***

---



---

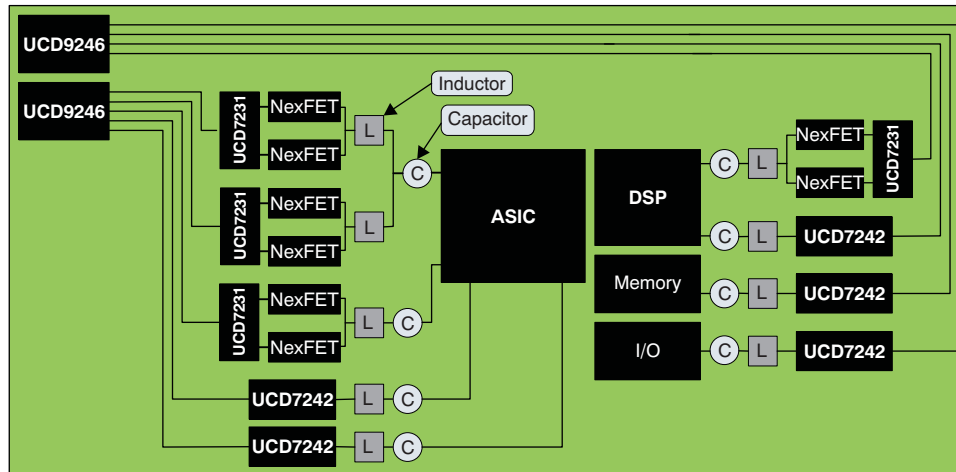
This chapter describes the design of the circuits that the UCD92xx will control, starting with the layout of the voltage sense signals. The power stage that is controlled by the UCD92xx consists of a gate driver, power MOSFETs, and output LC filter circuits. The equations presented in this chapter are the same equations used by the Fusion GUI. In most cases, the easiest method is to allow the Fusion software determine the circuit component values, and refer to this document if a specific application has unique requirements. Additional information regarding the power stage circuits in the product data sheet for the various gate driver ICs is also included.

Topic	Page
<b>2.1 Circuit Board Design .....</b>	<b>24</b>
<b>2.2 Temperature Monitoring .....</b>	<b>28</b>
<b>2.3 Current Monitoring .....</b>	<b>31</b>
<b>2.4 Inductor Derating Used by the Fusion GUI .....</b>	<b>33</b>
<b>2.5 Determining IOUT_CAL_GAIN and IOUT_CAL_OFFSET .....</b>	<b>34</b>
<b>2.6 Calibration of IOUT_CAL_GAIN and IOUT_CAL_OFFSET .....</b>	<b>34</b>
<b>2.7 Filtering Requirements on the UCD723x CS Outputs .....</b>	<b>36</b>
<b>2.8 Filtering Requirements on the UCD7242 and UCD74106 CS Outputs .....</b>	<b>36</b>
<b>2.9 Calculation of UCD7230A ILIM Voltage .....</b>	<b>37</b>
<b>2.10 Calculation of UCD7231 ILIM Voltage .....</b>	<b>38</b>

## 2.1 Circuit Board Design

### 2.1.1 Power Stage Schematics

A typical power system controlled by a UCD92xx device has multiple voltage rails with specific start-up and shutdown sequencing requirements. For the purpose of this document, we will assume a system that requires power has eight voltage rails and fits in a 19-inch (48.26-cm) rack. For a large board similar to this one, we want the generation of power to be located close to the IC that requires the power; the regulation controller, on the other hand, can be located anywhere on the PCB that is convenient. Thus, we want the output capacitors, inductor, MOSFETs, and gate drive circuits placed in close proximity to the powered IC. [Figure 2-1](#) illustrates this configuration.



**Figure 2-1. Typical Multi-Voltage-Rail Board**

In addition, we also want the circuits that sense current to be close to the load. Because the currents may be large, in order to keep the impact on power conversion efficiency small, the voltage across the sense element must also be kept small. For this reason, the UCD92xx controllers do not include a current sense amplifier in the controller. Instead, each UCD72xx gate drive IC has a current sense amplifier to condition this small current sense voltage and send it to the digital controller.

When you select a power stage circuit to be controlled by the UCD92xx, you must make three significant design decisions:

1. Whether to lay out the gate drive, MOSFETs, inductor, and capacitors locations yourself, or purchase a power module with an engineered and tested circuit layout;
2. Whether to use external power MOSFETs or select a gate drive IC with power MOSFETs integrated into the device; and
3. Which type of circuit is to be used to detect load current.

For the first question, selecting a power module has a significant advantage: the layout of the input capacitor(s) relative to the MOSFETs has been tested and proven to be sufficiently tight, thereby ensuring that ringing on the ground and circulating currents has been minimized. The cost of this engineered design is reflected in the price of a power module compared to the cost of the individual components purchased separately.

As a response to the second question, consider whether the design uses voltage rails that must supply 10 A or less, or whether board space is at a premium. In either case, a gate driver with internal MOSFETs is a good choice. The UCD7242 provides two 10-A power stages and the UCD74106 provides a single 6-A power stage. Above 10 A, the power stage circuit must either gang multiple UCD7242 stages together, or users must select a design with external MOSFETs.



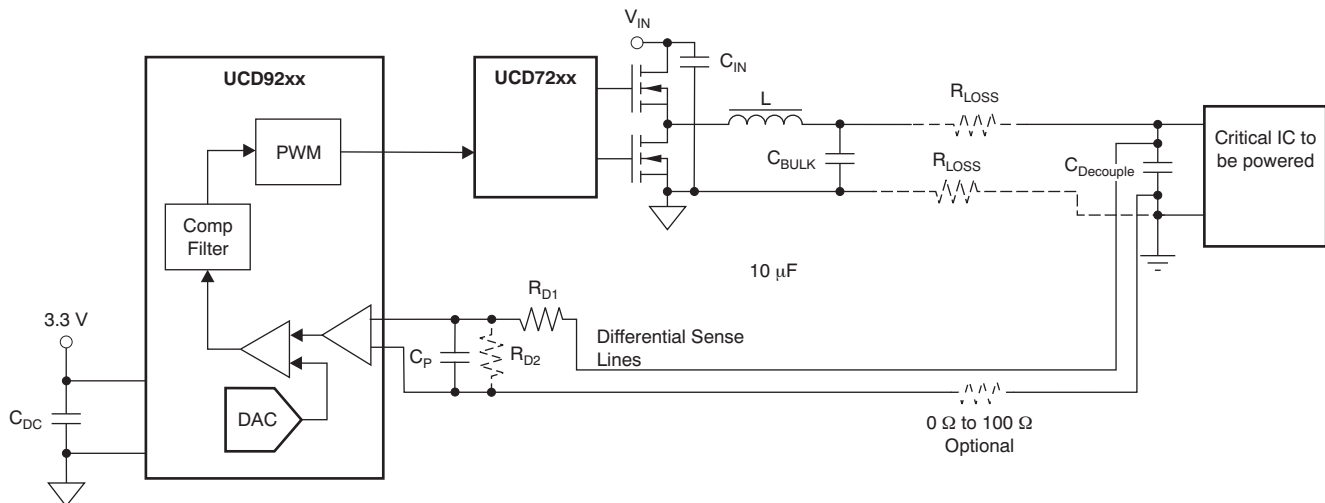
To answer the third question in a broad sense, there are three commonly-used circuits available to measure load current:

- If the MOSFETs are internal to the gate drive IC, current mirroring techniques can be used to sense the voltage across the  $R_{DS(ON)}$  segment of the MOSFETs; these techniques then amplify this voltage and make it available to the controller. The UCD7242 and UCD74106 devices follow this approach.
- A second option is to add a sense resistor in series with the load. The differential voltage across the sense resistor can then be applied to the input of the current sense amplifier.
- The third approach is to measure the voltage across the resistance of the copper windings in the inductor. This method has the advantage of providing a current sense voltage without inserting any additional losses into the system.

The description of this third method is detailed in [Section 2.3.1](#).

### 2.1.2 Differential Sense Lines

The UCD92xx controller has up to four differential error amplifier inputs to sense the voltage that is to be regulated. This architecture allows the controller to be placed some distance away from the point where precise voltage regulation is required. [Figure 2-2](#) shows the case where the UCD92xx is located on the same board as the load that receives the regulated voltage. Typically, the gate driver and MOSFETs are located in fairly close proximity to the load to minimize wiring losses. The voltage sense signals should be laid out as a differential pair of traces, terminating at a decoupling capacitor located at the IC that is being powered. To ensure that the PCB layout tools properly feed the voltage sense back to the controller as a differential pair, a small resistor can be placed near the decoupling capacitor.

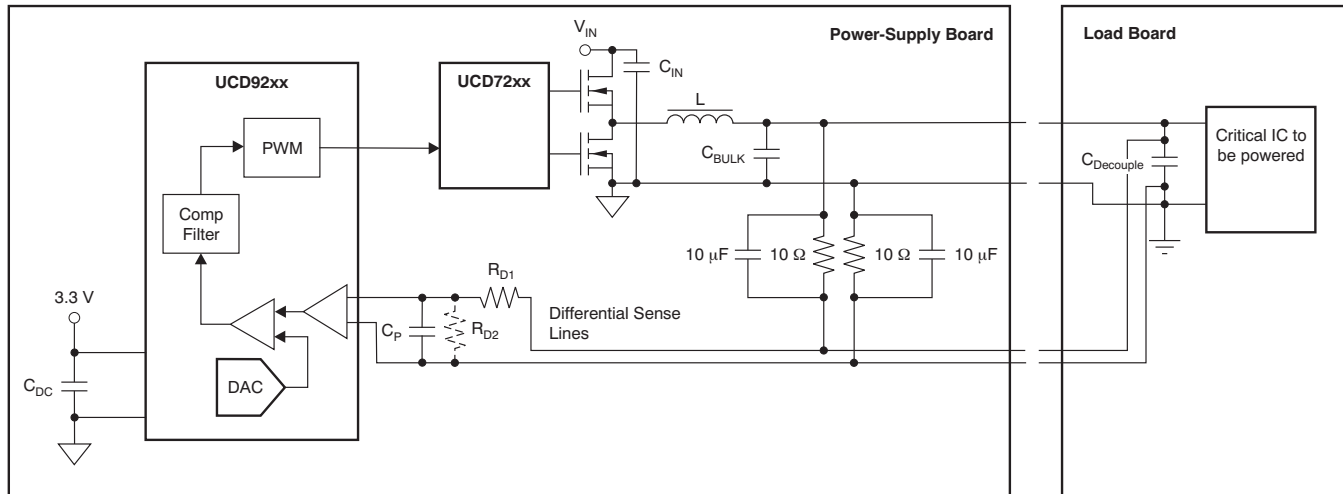


**Figure 2-2. Single Board  $V_{OUT}$  Sense**

It is recommended that an RC network be placed across the differential sense lines near the UCD92xx to act as an antialias filter. This filter should have a corner frequency that is 35% of the switching frequency. The recommended value of  $R_{D1}$  is 1.0 k $\Omega$ . Making  $R_{D1}$  larger than this value may cause an error in the sensed voltage because of the non-zero offset currents present at the input to the error amplifier. [Equation 1](#) shows the calculation of the input filter capacitor.

$$C_P = \frac{\sqrt{2}}{\pi f_{SW} R_{D1}} \left[ 1 + \frac{R_{D1}}{R_{D2}} \right] \quad (1)$$

When the power supply controlled by the UCD92xx is on a separate board from the load, as shown in [Figure 2-3](#), the differential sense leads can be brought out separately; this technique enables the regulated voltage to not droop as a result of the wires from the power supply to the load. In this case, a pair of 10- $\Omega$  resistors connect the output of the power stage with the UCD92xx error amplifier inputs. This configuration allows the output voltage to be controlled if the remote sense leads are not attached. When the remote sense signals are connected, the 10- $\Omega$  resistors are shorted out and the controller sees the actual voltage at the load. Texas Instruments' EVM boards use this remote sense circuit configuration because the EVM is typically separate from the load.



**Figure 2-3. 10- $\Omega$  Backup Circuit for Remote Sense**

### 2.1.3 3.3-V and 1.8-V Core Voltage Decoupling

The UCD92xx controller can be powered from a 3.3-V supply, or from  $V_{IN}$  through a series pass transistor controlled by the V33FB pin. The core logic in the controller operates at 1.8 V. The digital PWM circuits are clocked by an internal 250-MHz oscillator. The microcontroller that manages the PMBus interface and responds to faults is clocked at 31 MHz. These clocks are fast enough that good decoupling of the 3.3-V and 1.8-V power is required for proper device operation.

The [product data sheet](#) states that 0.1- $\mu$ F and 4.7- $\mu$ F bypass capacitors should be connected from V33A and V33D to ground near the device. In addition, a 0.1- $\mu$ F to 1- $\mu$ F bypass capacitor must be connected from the BPCap pin to ground for the internal 1.8-V supply to the device logic circuits. As with any digital device with fast clock edges, decoupling capacitors should be placed so that they are physically close to the device. This requirement means that vias should be avoided between the decoupling capacitors and the power and ground pins. Placing additional decoupling capacitors (0.01  $\mu$ F to 0.1  $\mu$ F) close to each power pin is also recommended.

Figure 2-4 illustrates the recommended decoupling capacitor layout.

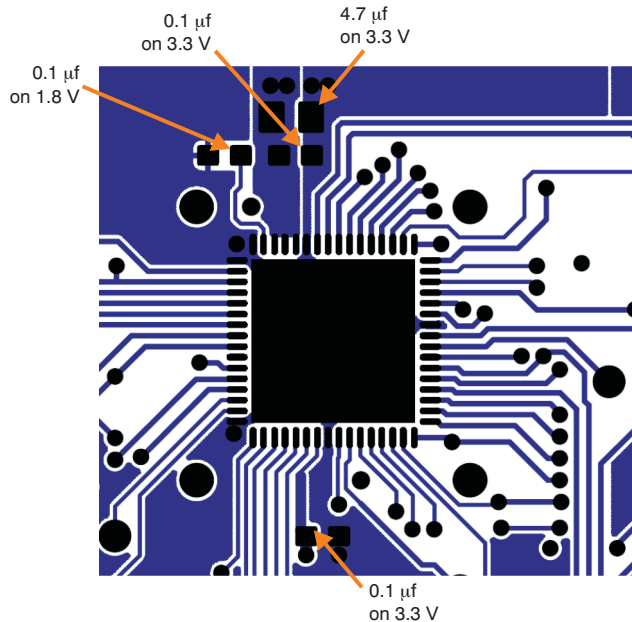


Figure 2-4. Recommended Decoupling Capacitor Layout

The UCD92xx device has separate analog and digital ground pins, and separate analog, digital, and I/O power pins. Tying the analog and digital ground together to a ground plane under the controller has been shown to produce good results. The V33A pin requires very good decoupling. If desired, this pin can be separated from the V33D and V33IO pins with a ferrite bead; in most cases, this bead is not necessary.

Figure 2-5 shows a typical application using an external transistor. The base of the transistor is driven by resistor  $R_1$  to  $V_{IN}$  and a transconductance amplifier with the output on the V33FB pin. The NPN emitter becomes the 3.3-V supply for the chip. Figure 2-5(a) illustrates an external transistor with a shunt regulator controller; a configuration with the 3.3-V provided from a low-dropout (LDO) regulator is shown in Figure 2-5(b).

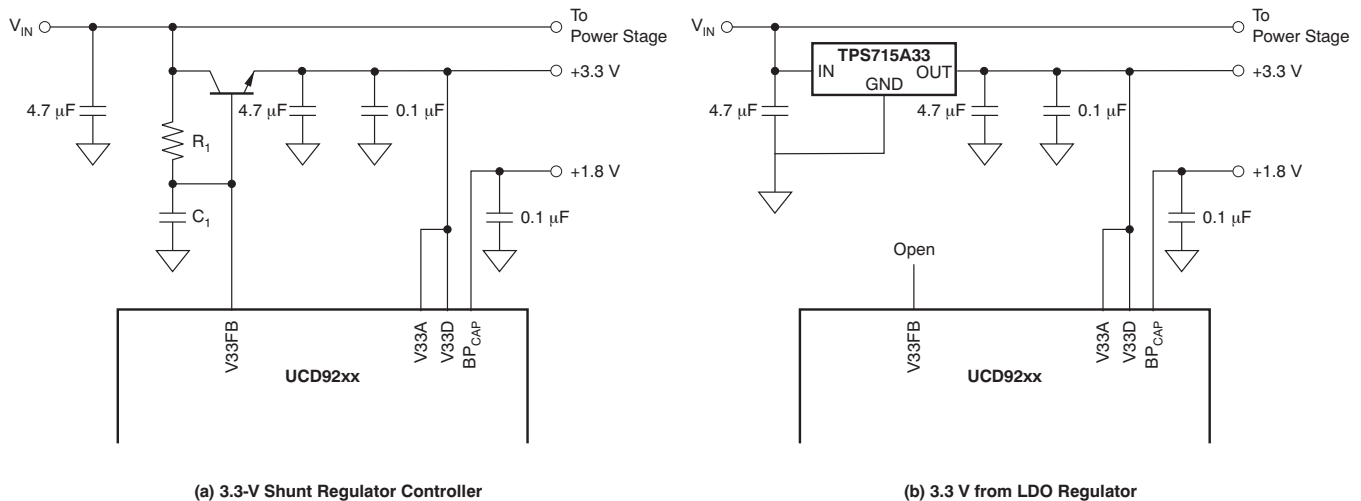


Figure 2-5. Providing 3.3-V Power for the Controller

In order to generate the correct voltage on the base of the external pass transistor, the internal transconductance amplifier sinks current into the V33FB pin and a voltage is produced across  $R_1$ . This resistor value should be chosen so that  $I_{\text{SINK}}$  is in the range of 0.2 mA to 0.4 mA.  $R_1$  is defined by [Equation 2](#).

$$R_1 = \frac{V_{\text{IN}} - 3.3 \text{ V} - V_{\text{BE}}}{\frac{I_{\text{E}}}{(\beta + 1)} + I_{\text{SINK}}} \quad (2)$$

where:

- $I_{\text{SINK}}$  is the current into the V33FB pin;
- $V_{\text{IN}}$  is the power-supply input voltage, typically 12V;
- $I_{\text{E}}$  is the current draw of the device and any pull-up resistors tied to the 3.3-V supply;
- $\beta$  is the beta of the pass transistor.

For  $I_{\text{SINK}} = 0.3 \text{ mA}$ ,  $V_{\text{IN}} = 12 \text{ V}$ ,  $\beta = 99$ ,  $V_{\text{BE}} = 0.7 \text{ V}$ , and  $I_{\text{E}} = 50 \text{ mA}$ , this formula selects  $R_1 = 10 \text{ k}\Omega$ . Weaker transistors or larger current loads require a lower resistance value to maintain the desired  $I_{\text{SINK}}$  current. For example, lowering  $\beta$  to 40 would require  $R_1 = 5.23 \text{ k}\Omega$ ; likewise, an input voltage of 5 V requires a value of 1.24 k $\Omega$  for  $R_1$  with the standard  $\beta = 99$  transistor.

The capacitor  $C_1$  in [Figure 2-5\(a\)](#) acts to compensate the loop that regulates the internal 3.3 V. As  $R_1$  changes, a proportional change to  $C_1$  must be made. [Equation 3](#) shows this relationship.

$$C_1 = \frac{0.001}{R_1} \quad (3)$$

## 2.2 Temperature Monitoring

### 2.2.1 Internal Temperature Sense

The internal temperature of the controller die is periodically sampled by the 12-bit analog-to-digital converter (ADC). The temperature can be monitored by issuing a READ\_TEMPERATURE\_1 PMBus command. The expected relationship of the sensed temperature to the discrete value at the output of the ADC is given by [Equation 4](#).

$$T_{\text{LSB}} = \text{round} \left[ \frac{10.44 \text{ LSB}}{^{\circ}\text{C}} T_{\text{Internal}} \right] + 2559 \text{ LSB} \quad (4)$$

This value is scaled by the firmware in the ARM7™ core processor and returned on the PMBus using [Equation 5](#).

$$T_{\text{C}} = T_{\text{LSB}} \frac{1.0^{\circ}\text{C}}{10.44 \text{ LSB}} - 245^{\circ}\text{C} \quad (5)$$

Therefore, the maximum temperature that can be returned is approximately +150°C.

### 2.2.2 External Temperature Sense

The UCD92xx is also able to measure the temperature of the inductor (and/or MOSFETS) of each controlled power stage. This temperature measurement is used to correct the monitored current when the dc resistance of the inductor is used to sense current. External temperature measurement is also used to detect an overtemperature fault. The device provides either one analog external temperature input pin and three select lines that can be used to drive an external analog multiplexer, or dedicated temperature sense inputs, so that up to eight external temperatures can be monitored.

One example multiplexer that can be used is the [CD74HC4051](#). The three mux select lines TMUX0, TMUX1, and TMUX2 continuously cycle through all eight mux channel addresses. However, the CS signals are logically assigned to the mux input channels in the same order that they are defined by the PHASE\_INFO command. Given the following case (summarized in [Table 2-1](#)), where voltage Rail #1 is driven by two phases and Rail #3 uses PWM4A (perhaps because it lays out more efficiently than using PWM3A), the temperature sensor outputs are routed to the mux as shown in [Table 2-2](#).

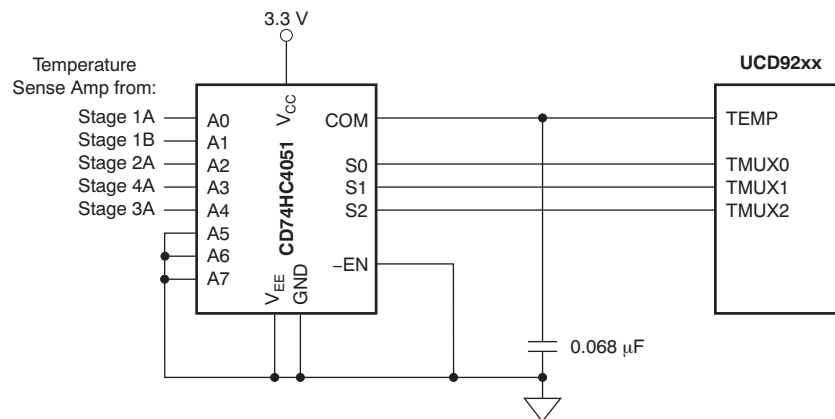
**Table 2-1. Example Assignment of PWM Outputs**

Voltage Rail	PWM Output
1	PWM1A
	PWM1B
2	PWM2A
3	PWM4A
4	PWM3A

**Table 2-2. Resulting Power Stage to Multiplexer Address Assignment**

Mux Address	Power Stage
0	1A
1	1B
2	2A
3	4A
4	3A
5	n/a
6	n/a
7	n/a

[Figure 2-6](#) illustrates the example outlined in [Table 2-1](#) and [Table 2-2](#). The 68-nF capacitor rolls off the temperature sense signal at 12 kHz, given the nominal 200-Ω on-resistance of the multiplexer.



**Figure 2-6. External Temperature Sense Multiplexer**

Various circuits can be used to sense temperature. A dedicated temperature sensor IC has the best accuracy and linearity, but typically are the most expensive to purchase. Thermistors, on the other hand, are generally available at a lower cost, but have poor linearity compared to temperature sensors.

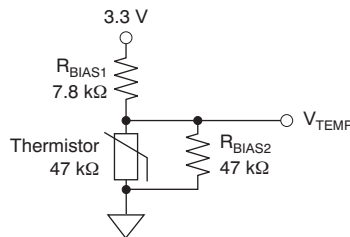
### 2.2.3 Thermistor Temperature Sense

The resistance of a thermistor changes exponentially with temperature. Equation 6 shows this relationship, where  $B$  and  $R_{REF}$  are specified in the data sheet for a given device.

$$R_T = R_{REF} \exp \left[ B \left( \frac{1}{T} - \frac{1}{T_{REF}} \right) \right] \quad (6)$$

For instance, consider a Murata NCP15WB473J03 NTC thermistor:  $B = 4050$  kelvins;  $T_{REF} = 298$  kelvins, and  $R_{REF} = 47 \text{ k}\Omega$ .

The temperature response of a thermistor can be linearized by placing it in a resistive voltage divider. Figure 2-7 uses the 47-k $\Omega$  Murata thermistor to illustrate this configuration.



**Figure 2-7. Thermistor Circuit**

The thermistor itself has very nonlinear characteristics. However, if it is used as the bottom half of a resistor divider, the nonlinearity is virtually canceled out, and the voltage becomes linear over a wide range. In the circuit of Figure 2-7, the second bias resistor pushes the linear region toward the high end of the temperature range.

From Equation 6 and the bias circuit in Figure 2-7, we can determine the slope of the temperature sense output voltage in Equation 7.

$$\frac{dv_T}{dT} = V_{REF} \frac{B}{T^2} \left[ \frac{R_T R_{B1}}{R_T R_{B1} + R_T R_{B2} + R_{B1} R_{B2}} + \frac{R_T^2 R_{B2} (R_{B1} + R_{B2})}{(R_T R_{B1} + R_T R_{B2} + R_{B1} R_{B2})^2} \right] \quad (7)$$

If we evaluate this formula for a given operating temperature (for example, +55°C, or 328 degrees Kelvin), then the values for the PMBus commands TEMPERATURE\_CAL\_GAIN and TEMPERATURE\_CAL\_OFFSET can be determined as shown in Equation 8.

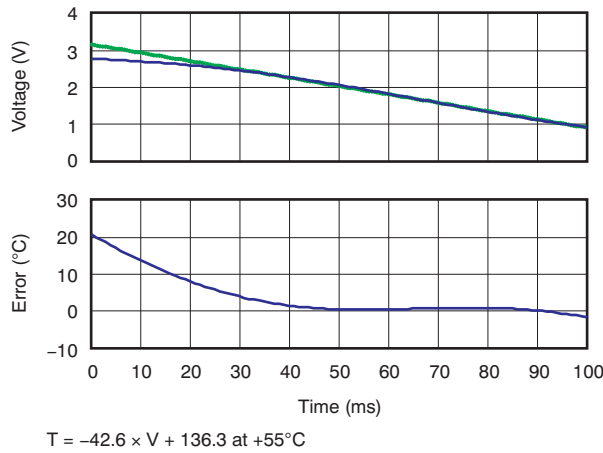
$$\text{TEMPERATURE\_CAL\_GAIN} = \frac{1}{\frac{dv_T(T_{OP})}{dT}}$$

$$\text{TEMPERATURE\_CAL\_OFFSET} = T_{OP} - \frac{V_{OP}}{\frac{dv_T(T_{OP})}{dT}} \quad (8)$$

where

$$V_{OP} = \frac{R_{B2}}{R_{B1} + R_{B2} + \frac{R_{B1} R_{B2}}{R_T(T_{OP})}} V_{REF} \quad (9)$$

A plot of the temperature sense voltage and the expected error, given the linear gain and offset configuration in the UCD92xx, is shown in Figure 2-8.



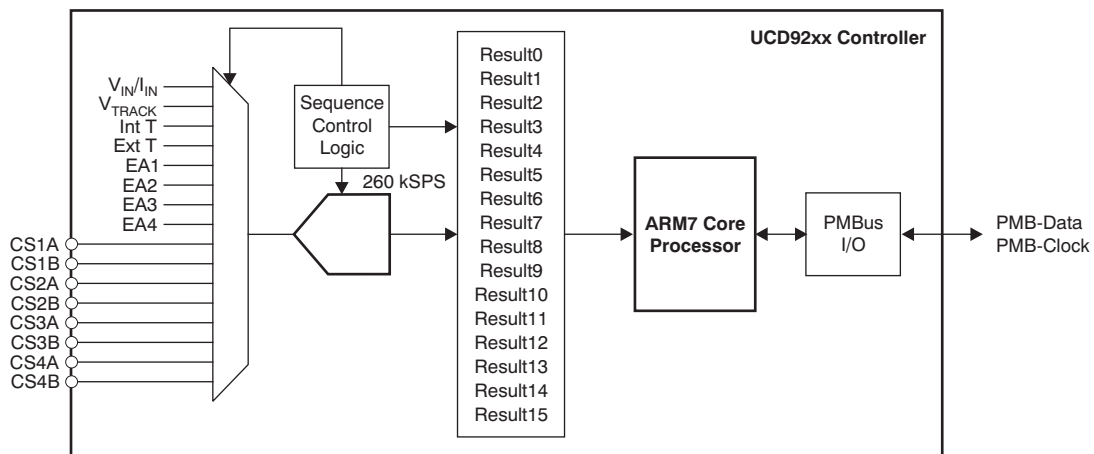
**Figure 2-8. Thermistor Circuit Response**

### 2.3 Current Monitoring

There are several techniques available to measure the output current of a power supply controlled by a UCD92xx device. In each case, the voltage across a series element feeds to an amplifier that generates a voltage; that voltage, in turn, is routed to the CS inputs of the UCD92xx.

Within the UCD92xx, the current sense signal is routed to a 12-bit ADC, shown in Figure 2-9. This voltage is converted to a current based on the configuration defined in the PMBus commands IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET. In the case of a multi-phase regulated output (that is, multiple MOSFET/inductor power stages driving one voltage rail), each power stage is configured by a separate IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET command.

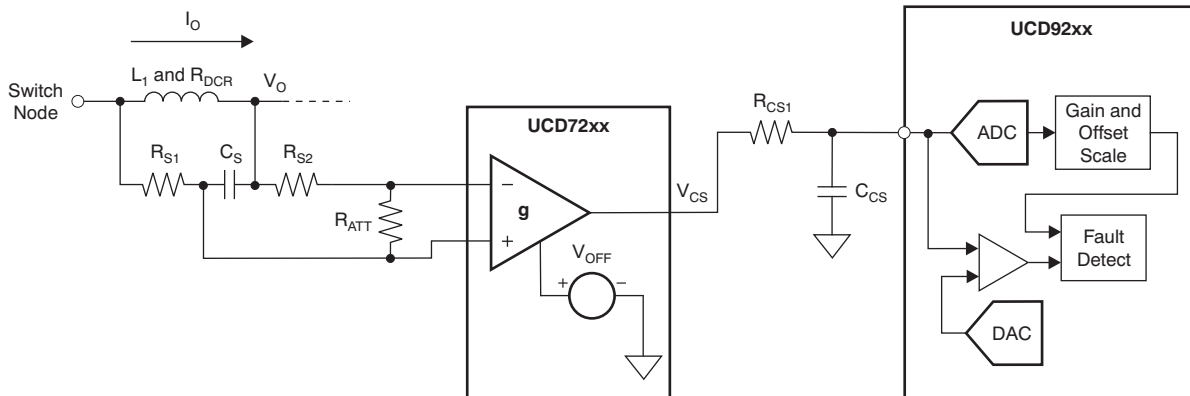
In addition, the current sense signal input on the A CS inputs is routed to an analog comparator for fast overcurrent protection. The threshold for this fast comparator is set by the FAST\_OC\_FAULT\_LIMIT command. The units for this command are amps (A), and the controller uses the values defined in the IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET commands to set the actual comparator voltage threshold.



**Figure 2-9. Block Diagram of the 12-Bit ADC Used for Monitoring**

### 2.3.1 Calculation of the Current Sense Voltage Using Inductor DCR

Many modern dc/dc power supplies measure output current by sensing the voltage across the winding resistance of the power stage inductor. The sensing circuit consists of a differential amplifier and an RC network that cancels the R/L zero of the inductor. Figure 2-10 shows the important circuit elements.



**Figure 2-10. Current Sense Schematic: Direct Current Resistance (DCR) Current Sense**

In addition, there is an optional low-pass network on the output of the current sense amplifier to provide antialias filtering for the ADC in the controller. The Fusion GUI determines the required component values to correctly measure the current across the dc resistance of the inductor. For reference, the following equations show the calculations used by the design tool.

The transfer function from sensed current to the output of the current sense amplifier is given in Equation 10.

$$V_{CSA} = \frac{G_{CSA} \frac{R_{ATT}}{(R_{S2} + R_{ATT})C_S} \left( s + \frac{R_{DCR}}{L} \right) i_O}{s^2 + \frac{L + (R_{S1} + R_{DCR})(R_{S2} + R_{ATT})C_S}{(R_{S2} + R_{ATT})LC_S} s + \frac{R_{DCR} + R_{S1} + R_{S2} + R_{ATT}}{(R_{S2} + R_{ATT})LC_S}} \quad (10)$$

We would like to cancel the  $R_{DCR}/L$  zero in the transfer function of Equation 10. This result occurs when Equation 11 is valid.

$$\frac{R_{S1}(R_{S2} + R_{ATT})}{R_{S1} + (R_{S2} + R_{ATT})} C_S = \frac{L}{R_{DCR}} \quad (11)$$

From Equation 11, therefore, we can define the value for  $R_{S1}$  as Equation 12.

$$R_{S1} = \frac{(R_{S2} + R_{ATT})L}{R_{DCR}C_S(R_{S2} + R_{ATT}) - L} \quad (12)$$

However,  $R_{S1}$  is a function of  $R_{ATT}$ . Consequently, before we can determine  $R_{S1}$ , we need to determine if the voltage across the inductor winding resistance must be attenuated by  $R_{ATT}$ .



### 2.3.2 Calculation of Attenuation Resistor $R_{ATT}$

At dc, the output voltage of the current sense amplifier (as calculated in Equation 10) becomes that shown by Equation 13.

$$V_{CSA} = G_{CSA} \frac{R_{ATT}}{R_{S1} + R_{S2} + R_{ATT}} R_{DCR} i_L + V_{OFF} \quad (13)$$

When is the attenuation resistor  $R_{ATT}$  necessary? It is needed to ensure that the current sense amplifier voltage does not saturate.

The maximum voltage for the fast overcurrent comparator in the UCD92xx is 2.0 V. Therefore, we want the current sense output voltage to be 2.0 V, when the current is at a level that generates a fault condition. We can find the required attenuation by solving Equation 6 and Equation 7 for  $R_{ATT}$  and  $R_{S1}$ . Unfortunately, this solution is rather messy. However, there are two cases that we are most interested in:

1. For the UCD7230A gate driver, the recommended circuit has  $R_{S2} = R_{S1}$ .
2. For the UCD7231 and UCD7232, the recommended circuit has  $R_{S2} = 0 \Omega$ .

For the UCD7230A, where  $R_{S2} = R_{S1}$  and  $\Delta v_{CS} = 2.0 - 0.6 = 1.4$ , this result simplifies to Equation 14.

$$R_{ATT} = \frac{L}{C_S} \frac{4G_{CSA}\Delta v_{CS}i_{L\_Max}}{(G_{CSA}R_{DCR}i_{L\_Max})^2 - \Delta v_{CS}i^2}$$

$$R_{S1} = \frac{L}{C_S} \frac{2G_{CSA}i_{L\_Max}}{(G_{CSA}R_{DCR}i_{L\_Max} + \Delta v_{CS}i)} \quad (14)$$

For the UCD7231, where  $R_{S2} = 0$  and  $\Delta v_{CS} = 2.0 - 0.5 = 1.5$ , the calculation is shown in Equation 15.

$$R_{ATT} = \frac{L}{C_S} \frac{G_{CSA}i_{L\_Max}}{G_{CSA}R_{DCR}i_{L\_Max} - \Delta v_{CS}}$$

$$R_{S1} = \frac{L}{C_S} \frac{G_{CSA}i_{L\_Max}}{\Delta v_{CS}} \quad (15)$$

In either Case (1) or Case (2), if the equation for  $R_{ATT}$  results in a very large (greater than 100 k $\Omega$ ) or a negative value, the attenuation resistor is not necessary;  $R_{S1}$  is thus calculated as shown in Equation 16.

$$R_{S1} = \frac{L}{R_{DCR}C_S} \quad (16)$$

The Fusion GUI software uses these equations to recommend the DCR current sense circuit component values.

## 2.4 Inductor Derating Used by the Fusion GUI

When calculating the RC network values that sense the voltage across the inductor DCR, the *inductance* is derated by 90% to account for the effect of B-H saturation at elevated current and temperature. The *resistance* (DCR) of the inductor is derated as shown in Equation 17.

$$R_{DCR(Warm)} = R_{DCR} \left[ 1 + \text{THERMAL\_COEFF}(55^\circ - 25^\circ) \right]$$

$$R_{DCR(Hot)} = R_{DCR} \left[ 1 + \text{THERMAL\_COEFF}(100^\circ - 25^\circ) \right] \quad (17)$$

To determine if  $R_{ATT}$  is necessary,  $R_{DCR(HOT)}$  calculation is used to determine the maximum voltage expected out of the current sense amplifier, as Equation 18 shows.

$$V_{CS} = G_{CSamp} R_{DCR(Hot)} I_{MAX} \quad (18)$$

Where  $I_{MAX}$  is the specified FAST\_OC\_FAULT\_LIMIT. If this value is greater than 2.0 V (the maximum threshold at which the UCD92xx trips the Fast OC analog comparator), the GUI applies Equation 8 or Equation 9 to calculate  $R_{ATT}$ . The value for  $R_{DCR(WARM)}$  is used in these equations to determine  $R_{S1}$  and  $R_{ATT}$ .

## 2.5 Determining IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET

The current sense voltage for the various methods of monitoring current can be described as shown in Equation 19: *current times sense resistance times current sense amplifier gain*, plus an offset (to allow for the measurement of negative current).

$$V_{CS} = G_{AMP} R_{SENSE} I_{OUT} + V_{OFF} \quad (19)$$

Then the controller uses the PMBus commands IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET to report the current as shown in Equation 20.

$$\begin{aligned} I_{MON} &= (V_{CS} \text{ Volts}) \left[ \frac{1 \text{ Amp}}{IOUT\_GAIN \text{ mV}} \right] \left[ \frac{1000 \text{ mV}}{\text{Volt}} \right] + IOUT\_OFFSET \\ &= (G_{AMP} R_{SENSE} I_{OUT}) \left[ \frac{1000}{IOUT\_GAIN} \right] + V_{OFF} \left[ \frac{1000}{IOUT\_GAIN} \right] + IOUT\_OFFSET \end{aligned} \quad (20)$$

Therefore, the OFFSET and GAIN parameters are calculated as Equation 21.

$$IOUT\_CAL\_GAIN = G_{AMP} R_{SENSE} 1000$$

$$IOUT\_CAL\_OFFSET = \frac{-V_{OFF}}{G_{AMP} R_{SENSE}} \quad (21)$$

## 2.6 Calibration of IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET

As a result of effects of board trace resistance and tolerances in the current sense circuit, the values for IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET for the physical PCBA may be somewhat different than the calculations discussed in this document. If the existing load current can be estimated, an additional load can then be applied in parallel with the application load and the gain and offset values calibrated for the actual physical PCB layout. Where high accuracy is required, this calibration can be done for each phase of every rail of every board as part of a manufacturing calibration process. More typically, users may sample a series of early prototype boards in the lab and then tweak the values of IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET. These resulting values, based on real measurements, can be stored in the project file for the device instead of those calculated by the GUI.

Figure 2-11 shows an example system board with one UCD9224 controller regulating two voltage rails. Here, an additional load is connected in parallel with the load on the board in order to calibrate the IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET values.

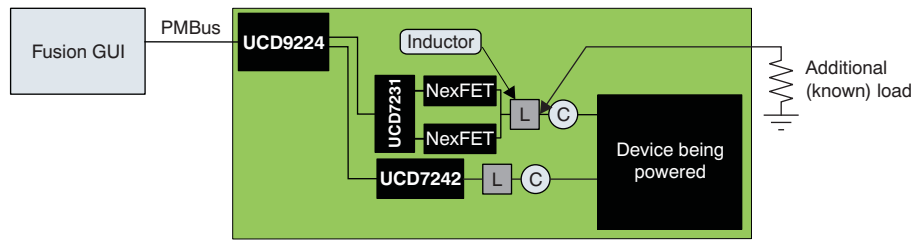


Figure 2-11. Adding an Additional Load to Check/Calibrate IOUT\_CAL\_GAIN

The load side of the inductor is typically an easy location to insert an additional load.

If we measure two or more points of estimated load current and note the value of the current monitored by the UCD92xx, we can calculate a correction to the first-pass estimates of IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET. From these two or more points, we can then calculate a slope,  $k$ , and offset,  $I_{OFF}$ , that relate the monitored current to the actual current as shown in Equation 22. One common method of calculating the slope and offset for more than two data points is to use the linear regression functions in a spreadsheet.

$$I_{MON}(n) = k \cdot I_{OUT}(n) + I_{OFF} \quad (22)$$

If the initial values of IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET are perfectly accurate, then  $k$  should be 1.0 and  $I_{OFF}$  should be 0.0. If this condition is not the case, then we can use  $k$  and  $I_{OFF}$  to correct the IOUT\_CAL values. From Equation 12, we see that Equation 23 is valid.

$$I_{MON}(n) = (G_{AMP}R_{SENSE}) \left[ \frac{1000}{IOUT\_GAIN_1} \right] I_{OUT}(n) + V_{OFF} \left[ \frac{1000}{IOUT\_GAIN_1} \right] + IOUT\_OFFSET_1 \quad (23)$$

Then the slope and offset are given by Equation 24:

$$k = (G_{AMP}R_{SENSE}) \left[ \frac{1000}{IOUT\_GAIN_1} \right]$$

$$I_{OFF} = V_{OFF} \left[ \frac{1000}{IOUT\_GAIN_1} \right] + IOUT\_OFFSET_1 \quad (24)$$

The actual current sense amplifier gain and offset are shown by Equation 25:

$$(G_{AMP}R_{SENSE}) = k \left[ \frac{IOUT\_GAIN_1}{1000} \right]$$

$$V_{OFF} = (I_{OFF} - IOUT\_OFFSET_1) \left[ \frac{IOUT\_GAIN_1}{1000} \right] \quad (25)$$

Substituting into Equation 13, we find that the IOUT\_CAL correction is calculated by Equation 26.

$$IOUT\_CAL\_GAIN_2 = k \cdot IOUT\_GAIN_1$$

$$IOUT\_CAL\_OFFSET_2 = \frac{IOUT\_CAL\_OFFSET_1 - I_{OFF}}{k} \quad (26)$$

If there are only two measured values of load current and UCD92xx monitored, then the formula for  $k$  and  $I_{OFF}$  are relatively simple and can be incorporated directly into the correction formula, shown in Equation 27:

$$IOUT\_CAL\_GAIN_2 = IOUT\_GAIN_1 \left( \frac{I_{MON1} - I_{MON2}}{I_{LOAD1} - I_{LOAD2}} \right)$$

$$IOUT\_OFFSET_2 = \frac{I_{MON1}I_{LOAD2} - I_{MON2}I_{LOAD1} + (I_{LOAD1} - I_{LOAD2})IOUT\_OFFSET_1}{I_{LOAD1} - I_{LOAD2}}$$
(27)

## 2.7 Filtering Requirements on the UCD723x CS Outputs

As noted above, the UCD92xx devices sample current with a 12-bit ADC. This ADC can sample up to 16 channels at 260 kS/s. These readings are scaled and processed by the core processor within the device at a sample period of 200  $\mu$ s / number of rails. Therefore, if the device is configured for one rail, the sample rate of measuring current is 5 kHz with each power stage current for that rail sampled consecutively at 260 kS/s. If four rails are configured, the sample rate for the current of any one rail is 1.25 kHz.

In addition to the measurement of current with the 12-bit ADC, the A CS inputs to the UCD92xx also have a programmable fast analog comparator that can be configured to terminate PWM action in case of an overcurrent condition. Assume that the fast OC threshold is set to the maximum limit of 2.0 V; further assume that the nominal operating current corresponds to a current sense voltage of 1.5 V. If we set a criteria that the current is to suddenly double, we would want to detect the OC event and shut down the device in 10  $\mu$ s. Thus, we want to set the filter time constant so that there is a 0.5-V change in voltage because of a 1.5-V change in the output of the current sense amplifier in 10  $\mu$ s. Equation 28 expresses this mathematically.

$$RC = \frac{\Delta t}{\ln \left[ 1 - \frac{\Delta V_{CS}}{\Delta V_1} \right]} = \frac{10 \mu s}{\ln \left[ 1 - \frac{0.5}{1.5} \right]} = 24.7 \mu s$$
(28)

Therefore, the RC filter on each of the current sense inputs should have a bandwidth of 6.45 kHz. The Fusion GUI software recommends values of  $R_{CS} = 2.49 \text{ k}\Omega$  and  $C_{CS} = 10 \text{ nF}$ , respectively.

## 2.8 Filtering Requirements on the UCD7242 and UCD74106 CS Outputs

Given the sampling rate in the UCD92xx, we must apply the same 6.5-kHz low-pass rolloff to the current sense output for the drivers with internal MOSFETs, as illustrated in Figure 2-12. These devices measure the current in the MOSFETs directly, multiplexing the current sense depending on which FET is active. The current sense output is a current proportional to the sensed current. In this way, the transresistance gain ( $V_{CS}/I_{OUT}$ ) can be selected by adjusting  $R_{CS1}$ .

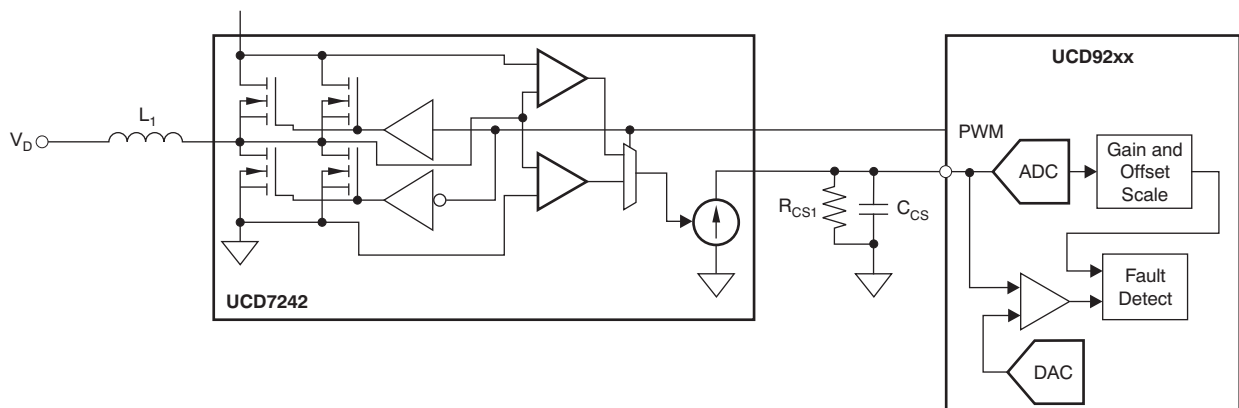


Figure 2-12. Current Sense for Drivers with Internal MOSFETS

For the UCD7242 driver, the current sense gain is 20  $\mu\text{A}/\text{A}$ . A 10-k $\Omega$  resistor then produces a transresistance gain of 0.2 V/A; the current sense voltage is 2.0 V at the rated current of 10 A. In this case, we want the filter capacitor,  $C_{CS}$ , to be 2.7 nF. Equation 29 shows the calculation for  $C_{CS}$ .

(Note: Several reference designs with  $C_{CS} = 10$  nF have been defined with good results during lab investigations.)

$$C_{CS} = \frac{25 \mu\text{s}}{R_{CS1}} \quad (29)$$

## 2.9 Calculation of UCD7230A ILIM Voltage

Figure 2-13 illustrates a circuit that is ideal for setting a CLF/FLT threshold for the UCD7230A.

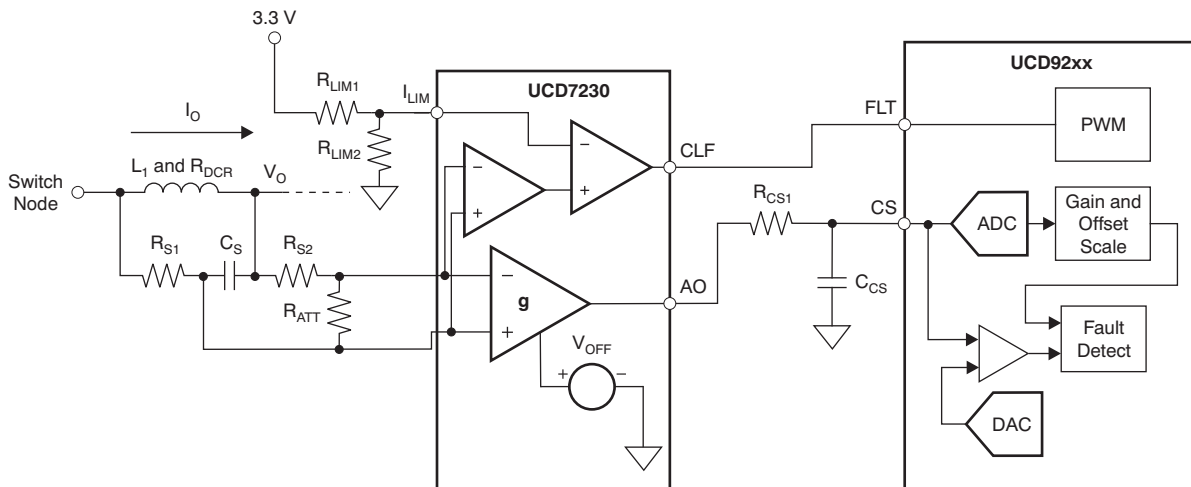


Figure 2-13. Setting UCD7230A CLF/FLT Threshold

The internal overcurrent threshold on the inductor current is set by the ILIM pin. The voltage on this pin is set to 10 times the voltage representing the maximum current at the POS and NEG pins. Therefore, the voltage on the ILIM pin should be equivalent to that shown by Equation 30.

$$V_{ILIM} = 10 \frac{R_{ATT}}{R_{S1} + R_{S2} + R_{ATT}} R_{DCR} I_{MAX} \quad (30)$$

There is also an internal divider on the ILIM pin; the equivalent circuit for this internal divider is a 0.5-V source with a 42-k $\Omega$  resistor. If we define this internal resistance as  $R_{LIM3}$  and set  $R_{LIM1} = 24.9$  k $\Omega$ , we can then determine the desired value for  $R_{LIM2}$  using Equation 31.

$$R_{LIM2} = \frac{V_{ILIM}}{\frac{3.3}{R_{LIM1}} + \frac{0.5}{R_{LIM3}} - V_{ILIM} \left[ \frac{1}{R_{LIM1}} + \frac{1}{R_{LIM3}} \right]} = 15.6k \frac{V_{ILIM}}{2.26 - V_{ILIM}} \quad (31)$$

Leaving  $R_{LIM2}$  open sets the internal threshold at 0.100 V, and in this case, Equation 32 is valid.

$$I_{MAX} = \frac{100m}{R_{DCR}} \left[ \frac{R_{S1} + R_{S2}}{R_{ATT}} + 1 \right] \quad (32)$$

## 2.10 Calculation of UCD7231 ILIM Voltage

For the UCD7231, the overcurrent threshold that generates an active signal on the CLF pin is compared directly to the current sense amplifier output voltage. [Equation 33](#) shows this relationship.

$$V_{ILIM} = \frac{R_{ATT}}{R_{S1} + R_{S2} + R_{ATT}} R_{DCR} I_{MAX} + 0.5 \quad (33)$$

Normally, the ILIM voltage is set by a resistor divider connected between 3.3 V and ground. The Fusion GUI recommends a 24.9-kΩ resistor for  $R_{LIM1}$ ; therefore,  $R_{LIM2}$  is calculated as shown in [Equation 34](#).

$$R_{LIM2} = R_{LIM1} \frac{V_{ILIM}}{1 - V_{ILIM}} = 24.9k \frac{V_{ILIM}}{1 - V_{ILIM}} \quad (34)$$

## Voltage Regulation

This chapter gives a detailed description of how the UCD92xx controller regulates the supply voltage using digital feedback control. As noted in [Chapter 2](#), the formula and equations presented here are incorporated into the Fusion Digital Power Designer software and in most cases, allowing the design tool to recommend the loop compensation gives the best results. However, the recommended compensation is only as good as the information provided to the Fusion GUI, which can be uncertain as a result of board layout issues or individual component tolerances. Therefore, this information is included to help users understand what is going on when the regulation behavior is different from what is expected.

Topic	Page
<b>3.1 Loop Compensation .....</b>	<b>40</b>
<b>3.2 Implementation of the Compensating Filter <math>H(s)</math> .....</b>	<b>41</b>
<b>3.3 Modeling Delay in the Control Loop .....</b>	<b>45</b>
<b>3.4 Phase Loss as a Result of On-Time and Multiple Power Stages .....</b>	<b>47</b>
<b>3.5 Quantization Effects .....</b>	<b>48</b>
<b>3.6 Anti-Saturation Feature .....</b>	<b>48</b>
<b>3.7 Autotune Operation .....</b>	<b>49</b>
<b>3.8 How to Tweak the Compensation .....</b>	<b>51</b>
<b>3.9 Making Loop Transfer Function Measurements .....</b>	<b>54</b>

### 3.1 Loop Compensation

The UCD9xxx family of point-of-load (POL) voltage regulation controllers use digital signal processing techniques to perform voltage regulation. This architecture enables programmable control of the regulation setpoint voltage, the switching frequency, the phase of the switching action for each power stage controlled, and the loop compensation. It also enables digital monitoring of the regulation control loop. Ultimately, through the use of a multiplexed, 12-bit ADC (not shown in Figure 3-1), voltage, current, and temperature fault limits can be digitally programmed and detected.

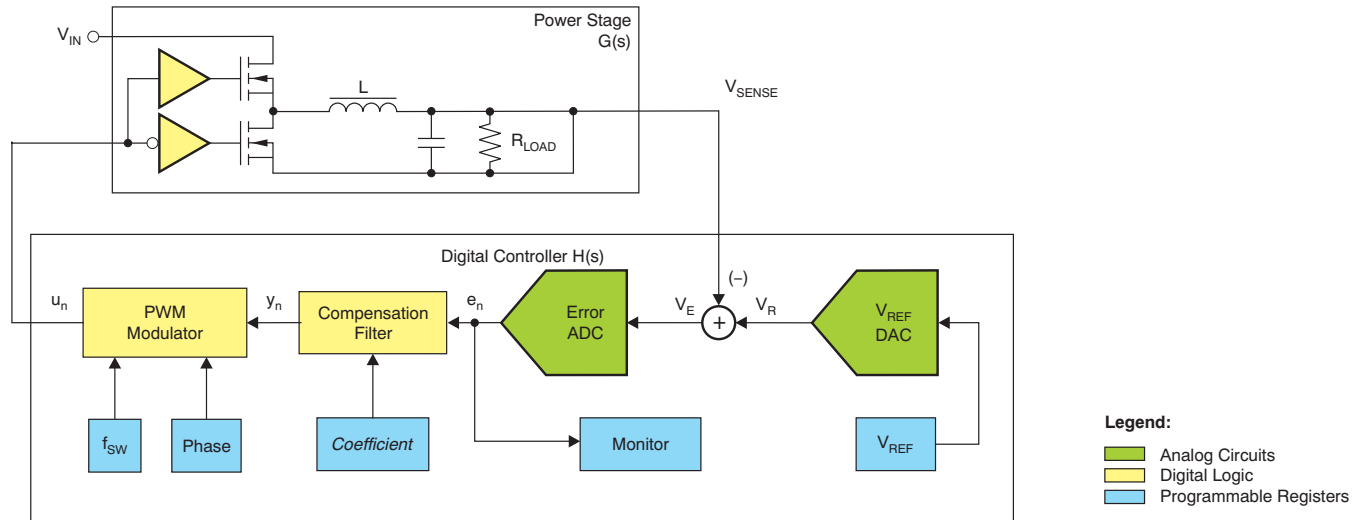


Figure 3-1. UCD92xx Fusion Peripheral Block Diagram

#### 3.1.1 Total Loop Gain

In Figure 3-1, we can define the transfer function for each block in this manner:

- **Power Stage:** The transfer function for the power stage is given in Equation 35:

$$G(s) = dV_{IN} \left( \frac{R_L}{R_S + R_L} \right) \frac{R_C C s + 1}{\frac{R_C + R_L}{R_S + R_L} L C s^2 + \left( \frac{L}{R_S + R_L} + \frac{R_C R_L + R_S R_L + R_C R_S}{R_S + R_L} C \right) s + 1}$$

where

- $d$  is the PWM signal duty cycle;
- $R_S$  is the series resistance of the MOSFETs and the inductor;
- $R_C$  is the effective series resistance of the output capacitor;
- $R_L$  is the load resistance;
- $L$  is the inductance; and
- $C$  is the output capacitance.

(35)

Note that this transfer function has a second-order pole at approximately  $\frac{1}{\sqrt{LC}}$ , with a no-load Q of  $\frac{1}{R_S + R_C} \sqrt{\frac{L}{C}}$  and a zero at  $\frac{1}{R_C C}$ . The units of the power stage gain are volts/cycle.



- **Error Amplifier Gain:** The analog front end circuits of the UCD92xx have a programmable gain that can be set to 1x, 2x, 4x, or 8x.
- **Error ADC:** The ADC that samples the error between the setpoint reference and the voltage to be regulated has a gain of 125 LSB/V, or a resolution of 8 mV/LSB.
- **Compensation Filter:** See [Section 3.2](#).
- **Pulse Width Modulator:** The pulse width modulator has an input-to-output gain of  $2^{-15}$  cycles/LSB.
- **Computational Delay:** There is a delay between sampling the error voltage and calculating the latest control effort. This delay ( $T_{\text{DELAY}}$ ) causes a phase lag in the open-loop transfer function that is expressed as a phase rotation,  $e^{-sT_{\text{delay}}}$ .

The total open-loop gain that is displayed as the system Bode plot is calculated in [Equation 36](#):

$$T(s) = G_{\text{PLANT}}(s) \cdot K_{\text{AFE}} \cdot K_{\text{ADC}} \cdot H(s) \cdot K_{\text{PWM}} \cdot G_{\text{DELAY}}(s) \quad (36)$$

From this equation, then, we can estimate the closed-loop bandwidth, phase margin, and gain margin by noting where the magnitude of  $T(s)$  crosses 0 dB and where the phase of  $T(s)$  crosses  $-180$  degrees.

### 3.2 Implementation of the Compensating Filter $H(s)$

The UCD9224/46/48 device uses a direct form digital filter to determine the compensated error signal. This filter has the form shown in [Equation 38](#). To determine the numerator coefficients [b0, b1, and b2] and the denominator coefficients [a0, a1, and a2], it is most convenient to start with the required second-order, continuous-time polynomial in  $s$ . Given the requirement for Type 3 compensation (that is, a compensating filter with a two-zero, two-pole transfer function), we start with the polynomial transfer function in  $s$  shown in [Equation 37](#).

$$H(s) = \frac{\left[ \frac{s}{\omega_{z1}} + 1 \right] \left[ \frac{s}{\omega_{z2}} + 1 \right]}{\left[ \frac{s}{\omega_0} \right] \left[ \frac{s}{\omega_{p2}} + 1 \right]} = K_{\text{DC}} \frac{\frac{s^2}{\omega_z^2} + \frac{s}{\omega_z Q_z} + 1}{\frac{s^2}{\omega_{p2}^2} + s + 0} \quad (37)$$

By using feedback loop analysis (discussed in the [Autotune](#) section), we can determine the desired values of the compensator gain  $K_{\text{DC}}$ ; the center frequency of the zeroes  $F_z$ , the spreading of the zeroes  $Q_z$ , and the second-pole frequency  $F_{p2}$ . These constants define the compensator transfer function. Then, we can find an equivalent discrete-time, two-zero, two-pole compensating filter of the form shown by applying the bilinear transformation to [Equation 37](#) and then equating the resulting terms to the coefficients in [Equation 38](#).

$$H(z) = \frac{y(z)}{x(z)} = \frac{b_0 z^2 + b_1 z + b_2}{a_0 z^2 + a_1 z + a_2} \quad (38)$$

To do this algebraic process, we substitute the approximation given in [Equation 39](#) for  $s$  into [Equation 37](#). Then, collect the terms.

$$s = 2F_s \frac{z - 1}{z + 1} \quad (39)$$

The resulting discrete-time filter coefficients  $[b_0 \ b_1 \ b_2]$  and  $[a_0 \ a_1 \ a_2]$  are shown in [Equation 40](#) and [Equation 41](#).

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix} = \frac{K_{DC}}{2\pi} \begin{bmatrix} 1 & 1 & 1 \\ -2 & 0 & 2 \\ 1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \left(\frac{F_s}{F_z^2}\right) \\ \left(\frac{\pi}{F_z Q}\right) \\ \left(\frac{\pi^2}{F_s}\right) \end{bmatrix} \cdot \frac{F_{P2}}{F_s + \pi F_{P2}} \quad (40)$$

$$\begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ -2 & 0 & 2 \\ 1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \left(\frac{F_s}{F_p^2}\right) \\ \left(\frac{\pi}{1}\right) \\ \left(\frac{0}{F_s}\right) \end{bmatrix} \cdot \frac{F_{P2}}{F_s + \pi F_{P2}} \quad (41)$$

To construct the compensation filter from [Equation 38](#), we multiply the denominator by both sides of the equation, multiply through by  $z^{-2}$ , then replace  $(x \cdot z^{-1})$  with  $x_{n-1}$  and  $(x \cdot z^{-2})$  with  $x_{n-2}$ . We do likewise with the  $y$  expressions. This process results in [Equation 42](#).

$$y_n = b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} - a_1 y_{n-1} + a_2 y_{n-2} \quad (42)$$

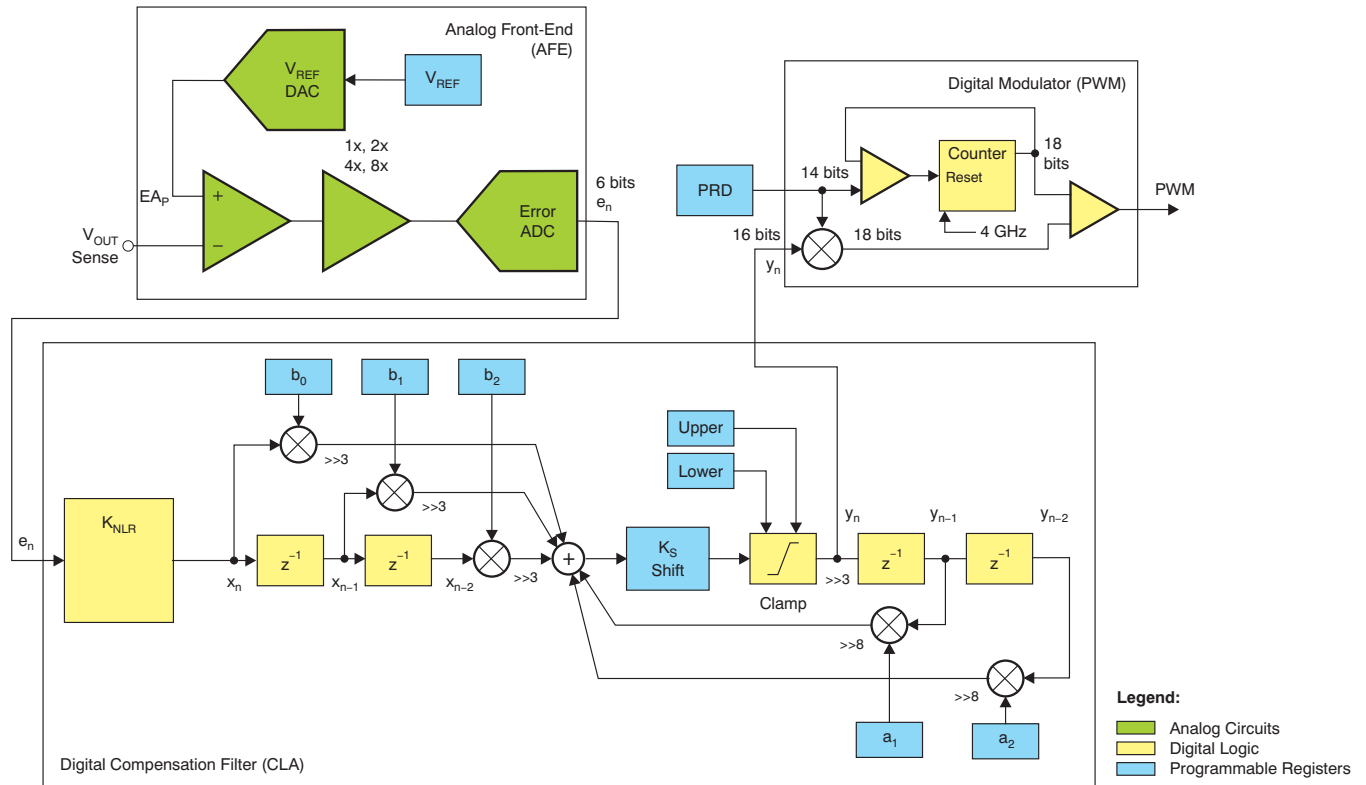
Finally, to reduce the bit length of the discrete filter coefficients, a binary gain (left shift) is applied to the filter, as shown in [Equation 43](#):

$$y_n = 2^{K_{scale}} (b_0 x_n + b_1 x_{n-1} + b_2 x_{n-2} - a_1 y_{n-1} + a_2 y_{n-2}) \quad (43)$$

This result is the filter that is implemented in the UCD9224/46/48; the filter is shown in [Figure 3-2](#).

### 3.2.1 Compensating Filter Architecture

Figure 3-2 illustrates the compensating filter architecture.



**Figure 3-2. UCD92xx Fusion Peripheral Block Diagram: Compensating Filter Overview**

There are three major segments of the voltage regulation signal path in the UCD92xx device:

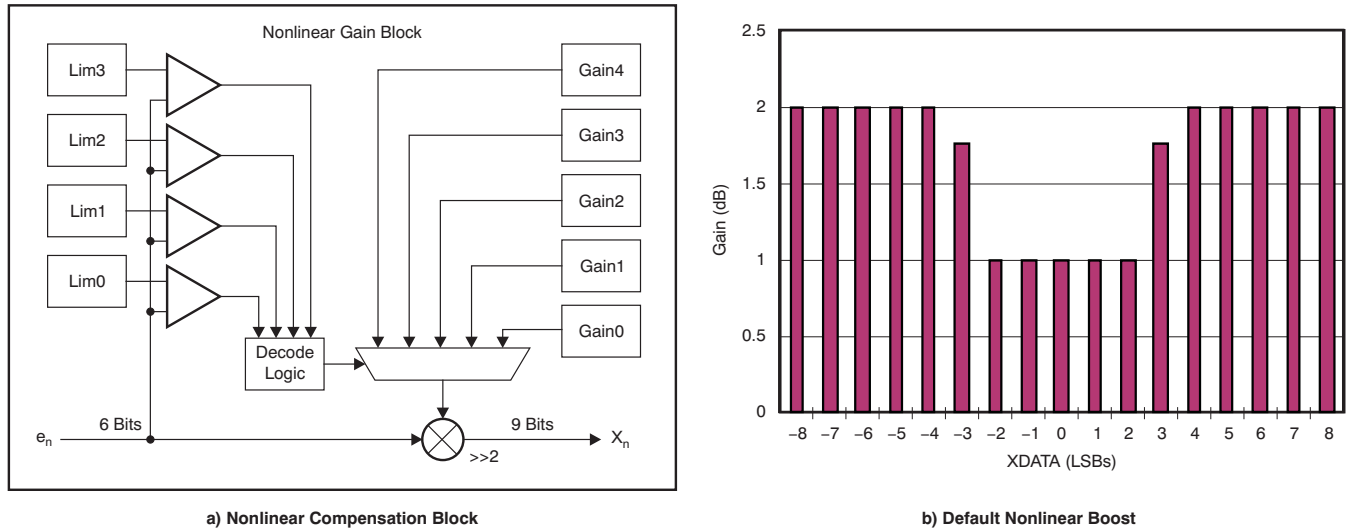
1. The analog front-end (AFE) that converts the error voltage to a discrete-time sequence;
2. The digital compensation filter; and
3. The digital pulse width modulator (dPWM).

The AFE subtracts the sensed regulated voltage from a programmed reference voltage using a switched capacitor difference amplifier. The reference voltage is set using a 10-bit, 0-V to 1.6-V DAC. The DAC is written to by the ARM7 processor core. (In fact, writing to the DAC register is the primary method by which the firmware in the ARM core controls the voltage regulation process.) The output of the AFE is a 6-bit value. This 6-bit error signal is applied to a nonlinear gain block, with a 9-bit output value that is fed into the digital compensation filter.

The compensated error signal out of the digital filter is applied to the pulse width modulator. At the PWM, it is multiplied by a value that represents the switching period (the PRD register). The result is a value that is compared to the PWM ramp counter. The modulator runs on a 250-MHz clock; there are 16 phases of the output, giving the modulator an effective 4-GHz clock rate. This configuration sets the pulse width resolution at 250 ps.

A nonlinear gain block (NLR) was added to the design of the UCD92xx device. This block sits between the AFE and the digital compensation filter. It boosts the gain when the error signal is large, and reduces the gain when the error signal is near zero. In practice, the primary benefit of the NLR circuit is to reduce the gain when the load is quiescent, thereby reducing jitter as a result of quantization effects in the signal path.

Figure 3-3 shows the NLR block and default boost.



**Figure 3-3. Nonlinear Compensation Block and Default Boost Characteristic**

### 3.2.2 Fixed-Point Math

Each regulated voltage rail has a dedicated compensation filter implemented in digital logic, separate from the core processor that manages the PMBus interface. (It is not necessary for the core processor to run in order for the device to regulate the voltage.) The filter performs each multiply and summing operation in fixed-point math. The error signal is a 6-bit signal that becomes a 9-bit signal after passing through the nonlinear gain block. The filter coefficients are 12-bit integers; the result of each multiply operation is a 20-bit value; and the output that feeds the PWM modulator is a 16-bit value. All of the signals are signed integers.

To determine the fixed-point integer filter coefficients, the floating-point pole frequencies, zero frequencies, and filter gain are applied to Equation 40 and Equation 41, respectively. The resulting floating-point coefficients  $[b_{01}, b_{11}, b_{21}]$  and  $[a_{01}, a_{11}, a_{21}]$  are then converted to integer values and a binary gain scaler as shown in Equation 44:

$$K_{\text{Scale}} = \text{ceiling} \left( \frac{\log \left[ \max \left( [b_{01}, b_{11}, b_{21}, a_{11}, a_{21}] \right) \right]}{\log(2)} \right) \quad (44)$$

Then we apply Equation 45:

$$\begin{bmatrix} B_{01} \\ B_{11} \\ B_{21} \\ A_{11} \\ A_{21} \end{bmatrix} = \text{round} \left( \begin{bmatrix} b_{01} \\ b_{11} \\ b_{21} \\ -a_{11} \\ -a_{21} \end{bmatrix} \cdot 2^{(11 - K_{\text{Scale}})} \right) \quad (45)$$

The minus sign on  $a_{11}$  and  $a_{21}$  is a result of the fact that each of the multiply accumulate circuits in the controller perform a sum and the  $a$  coefficients in Equation 43 are subtracted.

Each multiply operation in the compensation filter discards the least significant bits and rounds the result.

The numerator multiply operation drops the bottom 3 bits; the binary  $K_{\text{Scale}}$  gain drops 3 bits; and the denominator multiply operations drop 8 bits. Therefore, the forward (numerator) gain through the fixed-point filter is  $2^{(12-3-3)} = 32$  times larger than the floating-point gains. This gain of 32 (30.1 dB) must be accounted for when calculating the total loop gain for the Bode plots. That is, the true gain of the compensation filter is the  $K$  value entered in the Fusion GUI, plus 30.1 dB.

### 3.2.3 Cascaded First-Order Filter

The control law accelerator (CLA) consists of a second-order filter cascaded with a first-order filter. So far, we have only discussed the second-order filter section. The cascaded first-order filter allows an additional zero and pole to be applied to the loop transfer function. The transfer function of this filter section is given in Equation 46.

$$G_3(z) = \frac{z + B_{12}}{z - A_{12}} \quad (46)$$

Because  $z = e^{sT}$ ,  $z = 1$  at dc, and the dc gain of the cascaded filter section can be shown by Equation 47.

$$G_3(\text{DC}) = \frac{1 + B_{12}}{1 - A_{12}} \quad (47)$$

Thus, when  $B_{12}$  and  $A_{12}$  are non-zero, the cascaded first-order filter section has a gain other than unity, and the gain of the second-order filter section must be adjusted to account for this condition if the overall loop gain is to be maintained.

The adjustment is handled automatically in the Fusion GUI. However, when the UCD92xx applies the fixed, minimum pulse-width kick-start period at the beginning of the soft-start ramp, it holds the pulse width constant with the clamps shown in Figure 3-2. Because the hardware uses the same clamp values for both the second-order filter section and the first-order filter section, there is an offset in the pulse width equal to the dc gain of the cascaded first-order filter. This offset causes a fairly large transient in the soft-start ramp. Therefore, for most point-of-load voltage regulation applications, the cascaded first-order filter should not be used. This restriction is applied by making the discrete-time filter coefficients zero. This condition, in turn, is accomplished by making the continuous-time pole and zero frequency equal to  $f_{\text{sw}}/\pi$ , because the Fusion GUI uses the bilinear transformation to convert between continuous-time and discrete-time.

### 3.2.4 Type II versus Type III Compensation

The default loop compensation implemented in the UCD92xx uses two zeroes and two poles. This approach is commonly referred to as *Type III compensation* when implemented in an analog PWM controller. (This approach is also called *PID compensation*.) In some cases, where the inductor or capacitors exhibit significant loss, Type II compensation may be appropriate. Type II compensation (or PI compensation) consists of one zero and a pole at the origin. Type II compensation can be implemented in the UCD92xx by making the zeroes real and setting the second pole,  $F_{p2}$ , equal to one of the real zeroes.

## 3.3 Modeling Delay in the Control Loop

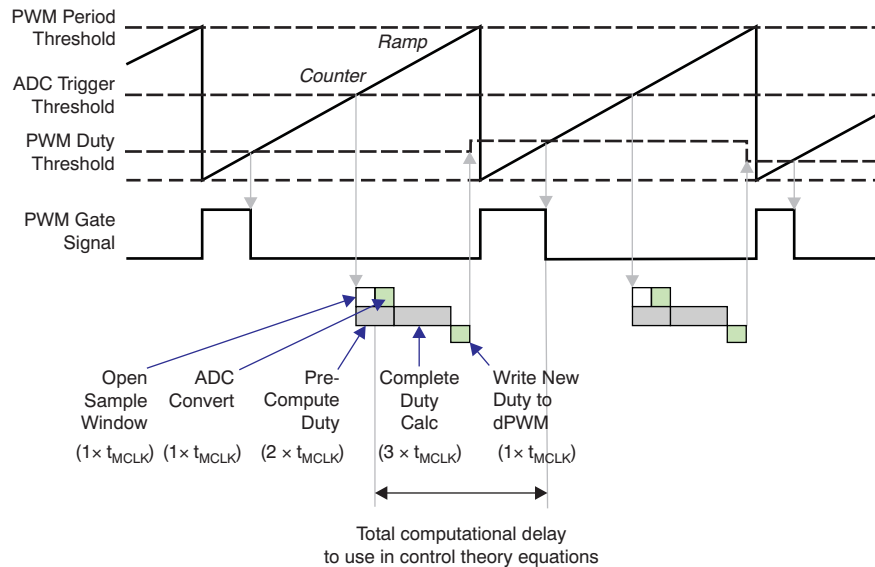
There is a delay between the time that the output voltage is sampled and the time that a new control effort is applied to the power stage. There are three contributors to the total computation delay:

- Error ADC sample and conversion time.
- Control effort computation time.
- Average time to the falling edge of the power stage switch voltage.

For the UCD92xx, the total delay is the time from the error ADC sample trigger (less the ADC sample window time) to the nominal falling edge of the MOSFET switch node. This delay period can be split into two time intervals:

- The time from the sample point to the end of the PWM switch period.
- The time from the beginning of the PWM switch period to the falling edge of the MOSFET switch node.

Figure 3-4 shows the PWM computational delay sequence.



**Figure 3-4. PWM Computational Delay**

In order to allow flexibility with regard to the point in time where the output voltage is sampled, the UCD92xx provides a PMBus command: EADC\_SAMPLE\_TRIGGER (located on the *Advanced Config* tab in the Fusion GUI). Normally, this command is set to the smallest possible value so that the delay between sampling the voltage and applying the control effort causes the minimum amount of phase lag in the control loop. However, there may be times where there is noise in the system that occurs right at the sample time. In this case, the user can increase the EADC\_SAMPLE\_TRIGGER value to sample the regulated voltage earlier to trade off lower noise for more phase lag. The minimum EADC\_SAMPLE\_TRIGGER value is between 224 ns and 236 ns, depending on the switching frequency. This range is a result of the fact that the ADC and control law accelerator (CLA) both run on a 31-MHz clock, but the switching period comparator is based on a 250-MHz clock.

To sample the error voltage, the error ADC applies the voltage to a sample-and-hold capacitor for 32 ns and then opens the input switch to hold that value. Because the specified time is for the beginning of the sample-and-hold window, the actual delay occurs at the end of the window to the time when the new calculated CLA control effort is available. So the phase shift that this delay applies to the open-loop gain is the result of [Equation 48](#).

$$G_{\text{Delay1}}(s) = e^{-s \cdot T_{\text{Delay(CLA)}}$$

$$\text{where } T_{\text{Delay(CLA)}} = T_{\text{SAMPTRIG}} - 32 \text{ ns}$$

(48)

### 3.4 Phase Loss as a Result of On-Time and Multiple Power Stages

From the point of view of the analog power stage, the time at which the control effort is applied to the system occurs at the falling edge of the on-time pulse. The falling edge of the on-time is the sum of the value in the EV1 register (or EV3 register for the B PWM outputs) and the calculated duty cycle times the switching period. For the UCD9224/46/48 devices, the value in EV1 is set by the PHASE\_OFFSET command. Figure 3-5 shows the latency periods for a three-phase power supply.

When there are multiple power stages driving the output voltage rail, we must find the average time to the falling edge of the PWM pulses. This calculation is shown in Equation 49.

$$T_{\text{Phase(Ave)}} = \frac{1}{2} \frac{N_{\text{Phases}} - 1}{N_{\text{Phases}}} T_{\text{Period}} \quad (49)$$

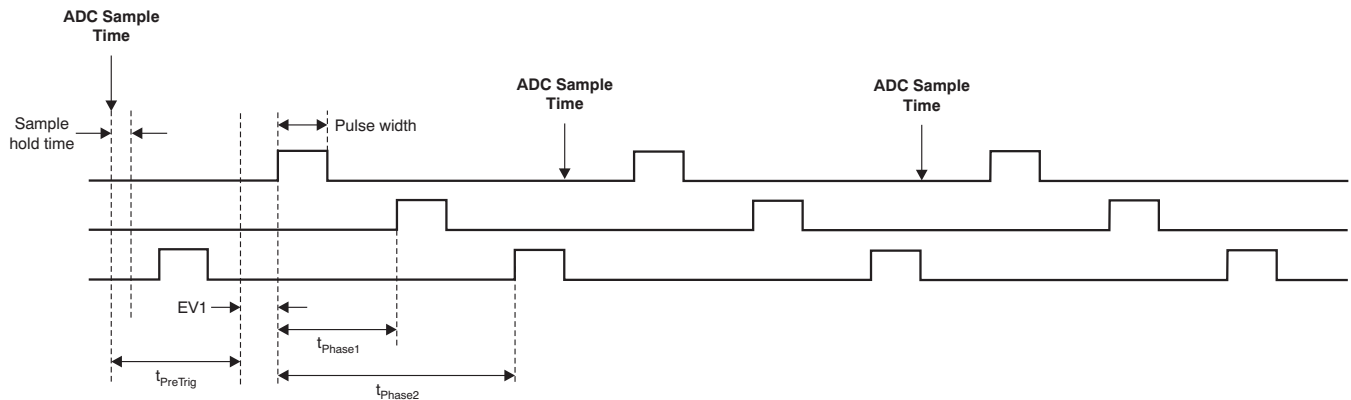


Figure 3-5. Latency for an Example Three-Phase Power Supply

The computational latency is then the period from the end of the sample window to the average end of the on-time pulse, as shown in Equation 50.

$$T_{\text{Delay(PS)}} = T_{\text{Phase(Ave)}} + (T_{\text{EV1}} + \text{duty} \cdot T_{\text{Period}}) \quad (50)$$

The Bode gain associated with computational latency is given by Equation 51.

$$G_{\text{Delay2}}(s) = e^{-s \cdot T_{\text{Delay(PS)}}} \quad (51)$$

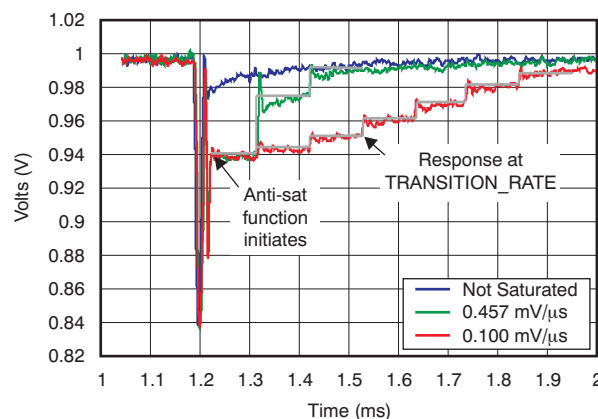
### 3.5 Quantization Effects

The primary concern with regard to quantization error is that it causes the pulse width of the PWM signal to be modulated in discrete increments. As long as this timing jitter is sufficiently high in frequency, it should not be a problem. Furthermore, the quantization-induced variation in the PWM pulse width does not create any new harmonics. The frequency content of the PWM signal above the switching frequency consists of the odd and even harmonics of the switching frequency, regardless of the amount of timing jitter.<sup>(1)</sup>

However, the quantization-induced timing jitter also creates sub-harmonics. If these sub-harmonics are low enough in frequency, they are not attenuated by the power stage LC output filter and can generate disturbances in the regulated output voltage. Therefore, we want to see what sub-harmonics are generated at different AFE gains as a result of the differences in error ADC resolution.

### 3.6 Anti-Saturation Feature

The UCD92xx device has a 6-bit ADC that samples the error between the setpoint reference and the sensed output voltage. This result is normally enough dynamic range to sense disturbances in the regulated voltage because of changes in load. However, if a very large load transient occurs, the 6-bit error ADC can saturate. At this point, the actual rate of change of sensed voltage is likely to be quite large, but the rate of change as seen by the compensation filter is zero, so it does not respond as quickly as it should. To allow the compensation filter to be able to see the true rate of change of the sensed voltage, when the UCD92xx detects that the error ADC is saturated, it moves the setpoint reference DAC 1/2 the dynamic range of the error ADC in an attempt to bring the error ADC out of saturation. This effect is illustrated in [Figure 3-6](#).



**Figure 3-6. Anti-Saturation Feature Activation and Recovery**

The scope traces shown in [Figure 3-6](#) show the setpoint reference being dropped by 64 mV when it detects that the error ADC is saturated. It then steps the setpoint reference back up to the desired voltage defined by `VOUT_COMMAND` in increments as defined by the PMBus command `TRANSITION_RATE`. The setpoint reference DAC is updated every 100 μs.

<sup>(1)</sup> Because there are always two edges in any switching cycle, the frequency content consists of only integer multiples of the fundamental switching frequency.



### 3.7 Autotune Operation

The goal of any power supply is to make the regulated voltage behave in the same way as an ideal voltage source. In other words, it should accurately hold the desired voltage and have a low output impedance. To extend the frequency at which the power supply can hold the output voltage steady, there are several steps we must take:

1. We must cancel the second-order pole in the LC output filter with zeroes in the feedback compensation filter;
2. To eliminate steady-state (or dc) error, we must include an integrator, which consists of a pole at the origin;
3. We must set the overall open-loop gain so that the closed-loop bandwidth and stability margin criteria are satisfied; and
4. A second pole should be added to the feedback transfer function to attenuate high-frequency noise that arises from the voltage quantization that results from the digital signal processing.

The autotune routine in the Fusion GUI sets the nonlinear gain to a moderate amount of gain boost. It also sets the frequency of the two poles of the compensation filter such that one pole is at the origin (creating an integrator) and the other pole is at a fixed percentage of the switching frequency (20%). These parameters do not change between designs. The parameters that are optimized by the autotune routine are the location of the compensating filter zeroes and the dc gain of the filter, which sets the gain of the overall loop transfer function. Figure 3-7 shows the default nonlinear boost controls.

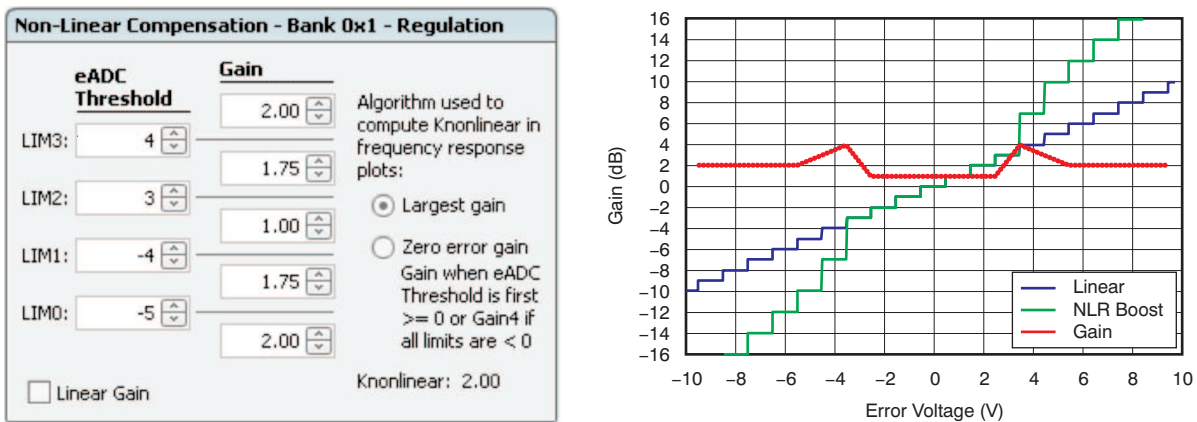
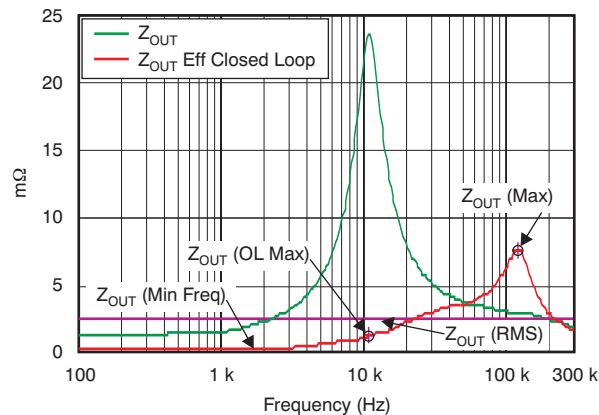


Figure 3-7. Default Non-Linear Boost Controls

To optimize the location of the compensating zeroes and the compensation gain, the autotune routine attempts to minimize the closed-loop output impedance of the power supply. Output impedance varies with frequency; therefore, a performance metric of output impedance is defined. The cost metric is the weighted sum of four measures of the output impedance:

- Closed-loop  $Z_{OUT}$  at the minimum frequency measured (100 Hz). Low output impedance at dc ensures that the power supply has minimum steady-state error (low dc offset).
- Closed-loop  $Z_{OUT}$  at the frequency where the open-loop  $Z_{OUT}$  is a maximum. The frequency where the open-loop output impedance is a maximum is the frequency at which the open-loop output voltage starts to ring because of a disturbance. Minimizing the closed-loop  $Z_{OUT}$  at this frequency is important for good transient performance.
- Maximum closed-loop  $Z_{OUT}$ . If the loop gain is high enough, the closed-loop  $Z_{OUT}$  can peak up at a frequency other than the frequency at which the open-loop  $Z_{OUT}$  is a maximum, and the system starts to ring at this frequency instead. Putting a cost on the maximum closed-loop impedance helps to flatten the closed-loop  $Z_{OUT}$ , which is also important for good transient performance.
- The RMS of the closed-loop  $Z_{OUT}$ . The RMS measure of output impedance covers all the frequencies not represented in the previous cost metric measures.

The autotune performance criteria are illustrated in [Figure 3-8](#).



**Figure 3-8. Autotune Criteria**

A cost metric consisting of the weighted sum of these four measures is calculated for each trial location of the zeroes. The compensation coefficients for the trial with the lowest cost metric are returned as the results of the autotune algorithm. The default weights of the cost metric are set so that each of the measures of output impedance contribute more or less equally to the result.

### 3.7.1 Determining the Compensating Zeros and Compensating Gain

The autotune routine first determines the transfer function for the power stage plant. This process can be done by calculating the transfer function based on the components specified in the power stage schematic, or it can be measured using the built-in transfer function measurement capability of the UCD92xx. The goal is to have the compensating zeroes approximately cancel the poles in the plant. However, it turns out that the closed-loop impedance, and therefore a load transient disturbance, is not minimized by an exact cancellation of the poles in the plant.

To set the location of the compensating zeroes, the Q of the plant poles is determined and the Q of the compensating zeroes is set to be a fraction of the plant value. Specifically, the parameter  $Q_z$  in [Equation 49](#) is set in this manner:

```
if( Qplant > 2*0.333)
    Qz = Qplant/2;
else
    Qz = 0.333;
```

Then the center frequency of the compensating zeroes,  $F_z$ , is swept over a range of frequencies that includes the corner frequency of the plant LC output filter.

Once the Q and center frequency of the zeroes have been selected for a given trial, the phase of the open-loop transfer function is completely defined. Therefore, in order to determine the compensator gain,  $K_{DC}$ , the autotune routine traverses the phase of the open-loop transfer function and finds the first frequency where the phase first satisfies the phase margin criteria. [Equation 52](#) shows this calculation.

$$\theta = -(180 - PM) = \text{angle}(H(s)) \tag{52}$$

By default, the phase margin is set to 50 degrees, so the routine looks for the frequency where the phase of the open-loop transfer function is  $-130$  degrees. If the frequency that satisfies the phase margin is higher than  $1/10$ th the switching frequency, this frequency is limited to  $f_{SW}/10$ .

Once the frequency that satisfies the phase margin criteria is determined, the dc gain of the compensator ( $K_{DC}$ ) is adjusted so that the magnitude of the total open-loop gain is 1.0, or 0 dB, at this frequency. This process sets the loop bandwidth and completes the definition of the floating-point compensation coefficients for this trial. From the floating-point coefficients, the fixed-point integer values that would be downloaded to the device are then calculated. Lastly, these fixed-point coefficients are used to calculate the closed-loop output impedance, from which the impedance cost metric is calculated.

### 3.7.1.1 Caveats

There are several tests performed by the autotune routine that can reject a trial set of compensating zeroes and gain.

- If the phase does not pass through the value ( $-180^\circ + \text{PM}$ ), abort the trial filter coefficients.
- The phase should be relatively flat at frequencies below the bandwidth frequency set by the phase margin. If there is a local minimum in the shape of the phase versus frequency curve, abort the trial.
- If the round-off error as a result of the fixed-point math performed in the compensation filter using the trial integer filter coefficients produces a zero amplitude response to a 4-LSB impulse, the trial is aborted.
- Check that coefficients B01 and B21 are not equal; this condition results in an infinite  $Q_z$ . If so, abort the trial.
- Abort the trial coefficients if the loop gain magnitude dips below  $-1$  dB at frequencies below the phase margin-defined bandwidth.

## 3.8 How to Tweak the Compensation

Normally, the autotune routine will find the best set of loop compensation coefficients for voltage regulation. There may be some circuits, however, where a better load step response can be obtained by manually adjusting the compensation. In such a case, this procedure can be used to tweak the compensation filter coefficients.

### 3.8.1 Tuning by Pole/Zero Placement

In general, the loop compensation required to stabilize voltage regulation for a synchronous buck converter is a technique called *lead/lag compensation*. In common power-supply terms, this technique is known as a *type-III compensator*. We want to make the closed-loop output impedance of the regulator as low as possible so that it appears to the load as an ideal voltage source. The closed-loop output impedance is related to the open-loop output impedance as shown in [Equation 53](#).

$$Z_{\text{OUT(CL)}}(s) = \frac{Z_{\text{OUT(OL)}}(s)}{1 + G(s)H(s)} \quad (53)$$

Where  $G(s)H(s)$  is the open-loop transfer function. This evaluation means that the higher we make  $GH$ , the lower (or better) the closed-loop output impedance becomes.

We want the output impedance to be very low at dc, so an integrator is incorporated into the compensation. We also want the loop gain to be large (greater than 1) for as much of the frequency band as possible. This goal is what drives the desire for a higher loop bandwidth.

To achieve a higher bandwidth than the power stage transfer function has by itself, we employ lead/lag compensation. That is, the compensator has two zeroes that cancel the second-order pole of the LC output filter of the power stage. Therefore, to a first approximation, the zeroes of the compensator should match the poles of the power stage. The power stage poles and compensator zeroes then cancel each other, and the overall loop bandwidth is set by the pole of the compensator, along with the dc gain of the compensator.

We can use the Fusion GUI to demonstrate this type of pole/zero cancellation. The power stage pole frequency can be estimated by observing the roll-off frequency of the power stage; the  $Q$  of the power stage pole is the amount of peaking in the magnitude response of the power stage before the response rolls off. For instance, if the power stage input voltage is 10 V, the dc gain is  $20\log_{10}(10) = 20$  dB. If the response peaks at 25 dB before rolling off, then  $Q = 10^{(5/20)} = 1.78$ . If we manually set the center frequency of the compensating zeroes ( $F_z$ ) and the  $Q$  of the compensating zeroes to match the frequency and  $Q$  of the plant, the plot of total loop gain is a flat line with a 20-dB/decade slope.

However, perfectly matching the compensating zeroes to the second-order pole of the power stage does not generate the best transient performance. A review of the time simulation plots in the Fusion GUI bears this out. The response to a load step exhibits a great deal of ringing. The reason that perfect pole/zero cancellation is not the ideal way to set up the loop compensation can be seen in the graphs of output impedance. The open-loop impedance is typically very large at the resonance of the power stage and a flat  $GH$  loop transfer function does little to attenuate the resonance.

Instead of perfectly matching the plant poles, a better strategy is to place the compensating zeroes just above and below the plant resonant frequency. This approach causes the compensator transfer function to peak at the resonant frequency. Because the compensator transfer function  $H(s)$  is in the denominator of Equation 57, the resonance in the output impedance is attenuated.

Our question now becomes this: how far should the compensating zeroes be spread on either side of the peak in the open-loop output impedance? The answer depends on the  $Q$  of the power stage output filter. A good rule of thumb is that the  $Q$  of the compensating zeroes should be 50% of the  $Q$  of the second-order pole of the output filter. This guideline is what the autotune routine approximates in the Fusion GUI.

The load transient response can be improved further by moving the center frequency of the compensating zeroes down in frequency. This shift helps because each zero increases the gain; lowering the zeroes generate more gain at the output impedance resonance.

So far, we have only discussed the compensating zeroes. What about the poles of the compensating filter? The first pole must be at the origin to make the impedance very low at dc and eliminate steady-state error. The second pole has a smaller effect on the behavior of the feedback loop, but has some value. This pole smooths the compensated error; in the case of a digital loop such as in the UCD92xx, it also attenuates the quantization-induced variation in the commanded duty cycle.

Because the Fusion GUI uses the bilinear transform to move between the continuous-time design parameters of  $K$ ,  $Q$ ,  $F_z$ , and  $F_{p2}$ , the values for  $F_{p2}$  range from approximately 15% to 31.8% of  $f_{sw}$ . At 31.8% (or  $1/\pi$ ), the bilinear transformation places the pole at the origin of the unit circle, where it has no influence on the compensator transfer function. This process is performed by the Fusion GUI when the user toggles the *Simple Integrator* checkbox.

When  $F_{p2}$  is set to frequencies below 31.8% of  $f_{sw}$  this pole adds some roll-off of high frequency variation in the error signal. Typically, we can achieve a loop bandwidth of ~10% of  $f_{sw}$ , so  $F_{p2}$  cannot be set much below 15% of  $f_{sw}$  without using too much phase margin. For this reason, the default recommended setting of  $F_{p2}$  is 20% of  $f_{sw}$ . This calculation is performed by the autotune routine.

Finally, the nonlinear boost (NLR) applies loop gain when the error signal is away from zero. The best way to understand this feature is that it allows the loop gain to be reduced when the load is quiescent and the error is near zero. In fact, the default setting for the Bode plots in the Fusion GUI calculates the loop gain with the maximum gain in the NLR circuit. By default, the NLR gain is 2.0 when the error is non-zero, and has a gain of 1.0 when it is near zero. These parameters give a 6-dB reduction in gain when the load is quiet and improve noise-induced disturbance in the voltage regulation control effort.

From this discussion, a set of rules for doing pole/zero placement of the compensation can now be established:

1. Identify the  $Q$  and resonant frequency of the power stage LC output filter. Note that these values change with the load current: the lighter the load, the more peaking can be observed in the resonance (that is, a higher  $Q$ ). The Fusion GUI calculates the plant transfer function at 25% of rated current by default.
2. Set the Compensation mode to *Manual*, and set the Linear Compensation mode to *Complex Zeros*. (This setting does not mean that the zeroes must be complex; it means only that the zeroes are described by a center frequency and a  $Q$ .)
3. Set the  $Q_z$  of the compensating zeroes to be approximately 1/2 of the power stage  $Q$ . With electrolytic or tantalum capacitors,  $Q_z$  is generally less than 0.5, which results in real zeroes. An all-ceramic capacitor design (because ceramic capacitors have very low ESR) typically has a higher  $Q$ , and the compensating zeroes will be complex.
4. Set the center frequency of the zeroes  $F_z$  to be 60% to 90% of the plant resonant frequency.
5. Set  $F_{p2}$  to 20% of the switching frequency  $f_{sw}$ .
6. At this point, the phase of the total loop gain (blue trace) is completely defined. Find the frequency where the phase has the desired amount of phase margin. If 50 degrees is the goal for the phase margin, find the frequency at which the phase is  $(-180 \text{ degrees} + 50 \text{ degrees}) = -130 \text{ degrees}$ . Now use the magnitude plot and find the gain at this frequency.

7. Adjust the dc gain  $K$  so that the loop gain magnitude is 1.0, or 0 dB, at the frequency where the phase margin is achieved.
8. Now check the time simulation waveforms to view the load step response to the compensation selected above.

This routine is essentially the same process used by the autotune routine. In most cases, then, it is simply easier to permit the Fusion GUI to determine the compensation, and focus on the effects of changes to the power stage output filter inductor and capacitor values. Of course, the capacitors and inductors generally cost more than the controller as well. In this way, the user can explore the effects of adding or removing capacitance and the effects of a larger or smaller inductor.

This discussion is based on setting the compensation during normal voltage regulation. These gains are stored in the UCD92xx as the Bank-1 coefficients. The Bank-0 coefficients are used during the soft-start and soft-stop ramps. The output voltage is lower during the start ramp, and as a result, not all of the circuits in the powered load are on; thus, the effective load on the power supply is less. This condition causes the plant  $Q$  to go up, which reduces the stability margin of the feedback loop in turn. Therefore, it is often desirable to have a lower loop gain during the soft-start ramp.

One easy way to achieve a lower gain during the soft-start ramp is to use the same  $K$ ,  $Q$ ,  $F_z$ , and  $F_{p2}$  values for both Bank-1 and Bank-0, but set the NLR gains to be flat for Bank-0. This configuration gives a 6-dB reduction in loop gain and produces half the loop bandwidth during the start ramp.

### 3.8.2 Tuning by PID Gains

An alternative to setting the compensation by adjusting the location of the compensating poles and zeroes is to set the compensator transfer function by adjusting the proportional, integral, and derivative gains. These adjustments are done by setting the Linear Compensation mode to *PID*. Note that this approach does not involve a different set of compensation hardware; it is only a different way of describing the same second-order compensator transfer function.

Figure 3-9 shows the PID gains control option in the software.

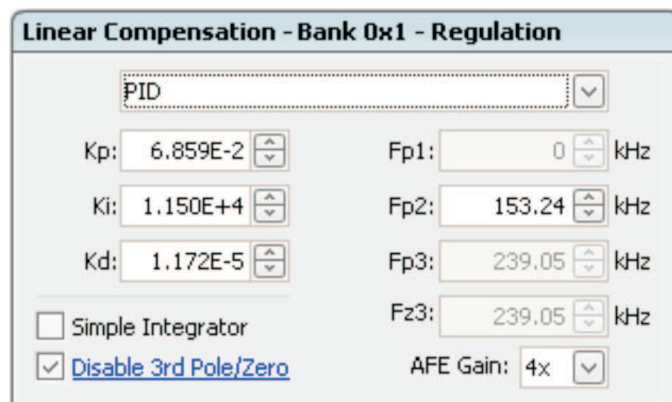


Figure 3-9. Fusion GUI: PID Gains Control

To tune the compensator by using PID mode, start with the gains closest to the desired setting using the above technique. Then switch to PID mode and perform the following procedure:

- Step 1. The integrator gain  $K_i$  sets the settling envelope. Increase this value until the envelope does not improve. It is acceptable if the ringing overshoot increases.
- Step 2. Now increase the derivative gain  $K_d$ ; this change should reduce any ringing. At some point, additional  $K_d$  gain increases the ringing, so back off from that point if you are experiencing the additional ringing.
- Step 3. Now increase the proportional gain  $K_p$  to further reduce the ringing.

There is a clear interaction between the three PID gains, so tuning by this method is often iterative. This technique, however, can sometimes identify a compensation transfer function that was elusive using the pole/zero placement method.

### 3.9 Making Loop Transfer Function Measurements

It is possible to perform a transfer function analysis (TFA) of the voltage regulation system while it is operating. The UCD92xx controller enables this option by generating a sinusoidal excitation signal that is injected into the loop at the output of the compensation filter. Then, the response to the excitation is captured at the output of the error ADC. If this process is repeated for a range of frequencies, the Bode plot for the system can be created. Figure 3-10 shows the system gain blocks active in the TFA calculation process.

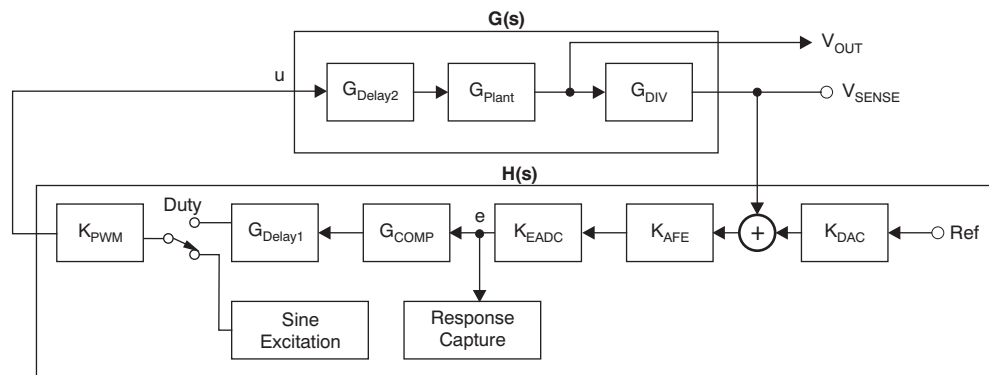


Figure 3-10. Transfer Function Analysis Block Diagram

#### 3.9.1 Types of Excitation

Several types of signals can be used to excite the system. Three of the most common are sine waves, white noise, and an impulse. White noise and an impulse are both attractive options because one signal can force a response at all frequencies. However, because all frequencies are present in the excitation signal, the signal power at any one frequency can be very small. This factor means that in order to obtain a reasonable measurement of signal-to-noise ratio (SNR), very long sequences must be stored. Sine wave excitation requires that separate measurements are made for each frequency, but the dynamic range required to excite the system to obtain a sufficiently accurate measurement for each frequency measurement is hundreds of times smaller than that required for a broadband excitation signal. Additionally, as we shall see, capturing the response to sine wave excitation can be done by storing just two words.

#### 3.9.2 Generation of Sinewave Excitation

An accurate and efficient method of generating a sine wave is by using a look-up table.<sup>(2)</sup>

The UCD92xx has a table in ROM that holds one period of a sine wave. It can generate a sine wave by stepping through the table with a step size that is inversely proportional to the desired generated signal frequency. At each sample time a new value is found from the table by adding the step offset to the previous index into the table. When the new table index would point to a value beyond the range of the table, the index wraps around to the beginning of the table by subtracting the table length from the calculated index value. Very high frequency resolution can be obtained by maintaining the fractional part of the table index and then rounding or truncating to generate each looked-up sample value. The step offset for a given frequency  $F_{Meas}$  is defined as shown in Equation 54:

$$\text{step} = N_{\text{Table}} \frac{F_{\text{Meas}}}{F_{\text{Sample\_Rate}}}$$

(54)

<sup>(2)</sup> See Figoli, D. (1999). *Creating a two channel sine wave generator*. Texas Instruments application note [SPRA412](#).

The following C code snippet illustrates one technique of generating the table look-up sine values. Note that because sine and cosine values always differ by 90 degrees, the sine values are produced by adding a fixed offset to the table index equal to 1/4 the table length.

```
#define QTR_TABLE_OFFSET (512*2/4) // *2 for long words
int32 phase = 0; // Define phase as 16 bits integer and 16 bits fraction
int32 step; // Step offset is set in the command handler
int16 index, bcos, bsin

phase += step; // Get next sine sample index
phase &= 0x03FFFFFF; // Force phase to wrap around above 1023.9999
// since table contains 512 32b words.
index = phase >> 16; // Use upper 16 bit word for table index.
bcos = sinTable[index]; //
bsin = sinTable[index + QTR_TABLE_OFFSET];
```

When the index is stepped past the end of the table, the MSB bits are discarded and the index now points to a sine value at the beginning of the table. In this way, very fine frequency resolution can be obtained.

### 3.9.3 Response Measurement Implementation

The definition of a discrete Fourier transform (DFT) is given by Equation 55.<sup>(3)</sup>

$$K_k \equiv \sum_{n=0}^{N-1} v_n e^{j2\pi nk/N} = \sum_{n=0}^{N-1} (v_n) \cdot \left[ \cos\left(2\pi \frac{k}{N} n\right) + j \sin\left(2\pi \frac{k}{N} n\right) \right] \quad (55)$$

Equation 55 postulates that we can calculate the real and imaginary magnitudes of the  $k$ th harmonic of any signal by multiplying that signal by a cosine and sine at that harmonic frequency. We have already generated the sine and cosine sequences to inject the excitation into the system; thus, the measurement calculation becomes very simple.

```
eCosSum += e*bcos; // accumulate cosine sum for node u
eSinSum += e*bsin; // accumulate sine sum for node u
```

One of the characteristics of the DFT formula is that only harmonics of the measurement interval are calculated. By limiting the measurement frequencies to harmonics, there is always an integer number of cycles over the measurement interval. Equation 56 shows the harmonic frequencies that produce an integer number of cycles.

$$F_{\text{Meas}} = \frac{k}{N} F_S \quad (56)$$

When a frequency is chosen that generates a non-integer number of cycles over the measurement interval, the DFT algorithm spreads the signal energy over several frequencies. This diffusion results in an error in the calculated magnitude called *leakage*. Leakage can be compensated for by applying a window function to the measurement signal before multiplying by the sine and cosine reference sequences. There are several popular window functions<sup>(4)</sup> that can be applied. One window function that is a good compromise between complexity and sensitivity is the triangular window shown in Equation 57.

$$w(n) = \frac{2}{N} \left[ \frac{N}{2} - \left| n - \frac{N-1}{2} \right| \right] \quad (57)$$

Therefore, the algorithm for measurement frequencies that are not harmonics is shown in Equation 58. This is not technically a DFT and is more correctly called a matched filter.

$$K_{\text{Meas}} = \sum_{n=0}^{N-1} \left( w(n) \cdot v_n \right) \cdot \left[ \cos\left(2\pi \frac{F_{\text{Meas}}}{F_S} n\right) + j \sin\left(2\pi \frac{F_{\text{Meas}}}{F_S} n\right) \right] \quad (58)$$

<sup>(3)</sup> Zwillinger, D. (Ed.). (1996.) Standard mathematical tables and formulas, 30th Edition, CRC Press.

<sup>(4)</sup> Source: [Wikipedia](http://en.wikipedia.org)

An alternative to windowing is to adjust the measurement interval to ensure an integer number of sine cycles. In this case, we use Equation 59:

$$K_{\text{Meas}} = \sum_{n=0}^{N-1} (v_n) \cdot \left[ \cos\left(2\pi \frac{F_{\text{Meas}}}{F_s} n\right) + j\sin\left(2\pi \frac{F_{\text{Meas}}}{F_s} n\right) \right]$$

$$N = \text{round}\left(k \frac{F_s}{F_{\text{Meas}}}\right)$$

(59)

Where  $K$  is the number of cycles desired in the measurement interval. This technique is used by the UCD92xx devices.

The default TFA measurement performed by the Fusion GUI software measures 50 frequencies from 1.0 kHz to  $f_{\text{SW}}/2$ . For each measurement, it makes the DFT measurement of the response for ~50000 cycles (100-ms interval). In making the measurement, the routine attempts to hold the amplitude of the response to the injected excitation to 30 mV, as shown in Figure 3-11.

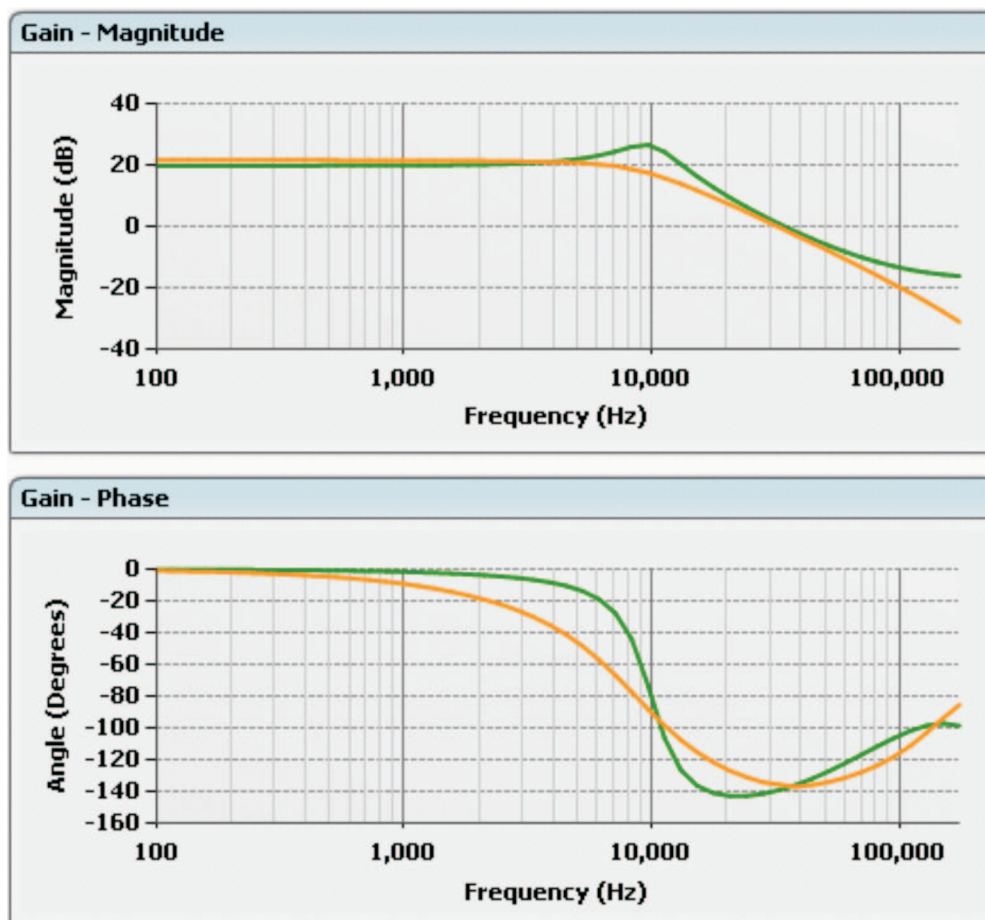


Figure 3-11. Modeled versus Measured Power Stage Transfer Function



---

---

## Start-Up Sequencing

---

---

This chapter reviews the features of the UCD92xx that can be used to set up sophisticated start/stop sequencing between several controlled voltage rails.

Topic	Page
4.1 Sequencing .....	58
4.2 Start Indications .....	58
4.3 TON_DELAY and TON_RISE .....	60
4.4 Group Commands .....	61
4.5 Daisy-Chaining CTRL and PowerGood Pins .....	62
4.6 Voltage Tracking .....	62
4.7 GPIO_SEQ_CONFIG .....	64
4.8 Alternative/Additional PowerGood Outputs .....	71

## 4.1 Sequencing

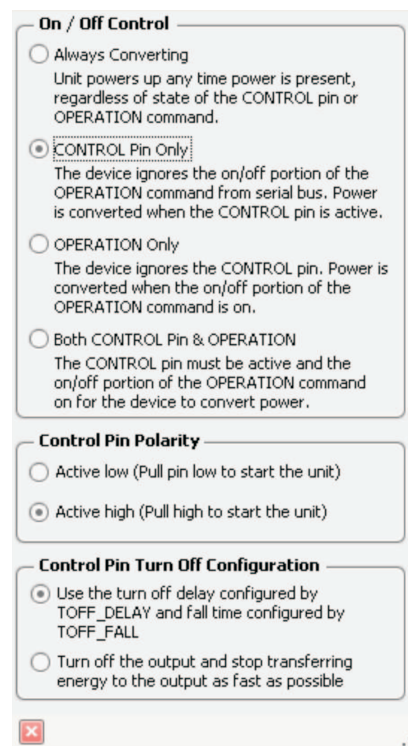
The UCD92xx controller provides multiple techniques to sequence the start-up of each voltage rail that it controls. In addition, the device can generate enable signals to start non-PMBus power supplies as well. The control of the sequencing of voltage rails falls into three broad categories:

- Timing the start of a voltage rail based on the TON\_DELAY and TON\_RISE PMBus commands.
- Configuring one voltage rail to track another voltage rail.
- Time the start of one voltage rail based on an internal or external *enable* signal. This enable signal can be issued by an external sequencer or microcontroller, or be based on a PowerGood indication for one of the UCD92xx controlled voltage rails. These enable signals are configured to be *turn-on dependencies* and *stay-on dependencies* for each rail.

These same techniques can be used to turn off a voltage rail. In addition, a fault indication on one voltage rail can be used to turn off other voltage rails. When enabled, this feature is called a *fault shutdown slave*.

## 4.2 Start Indications

The PMBus Standard defines four methods of defining the start indication for the sequencing process. The specific indication to use is configured by the ON\_OFF\_CONFIG command. The Fusion GUI displays the four options when you edit the ON\_OFF\_CONFIG value in either the *Vout Config* tab of the Configuration page, or on the *System Dashboard* window. Figure 4-1 shows the four options as displayed in the Fusion GUI.



**Figure 4-1. Fusion GUI: ON\_OFF\_CONFIG Options**

When the *Always Converting* option is selected, the start indication is defined by the time that the sensed voltage on the  $V_{IN}$  pin rises above the threshold set by the PMBus command VIN\_ON. This command accepts a value in volts and scales that value to the voltage at  $V_{IN}$  based on the PMBus command VIN\_SCALE. (The Fusion GUI sets the VIN\_SCALE value based on the resistors entered into the schematic on the Design page.) The UCD92xx continually monitors the  $V_{IN}$  pin at the sample rate specified in the data sheet. If the scaled voltage on this pin drops below the value defined by the VIN\_OFF command, the outputs are shut down.

When the *CONTROL Pin Only* option is selected, the start indication is defined by the time that the CTRL pin goes active, **and** the time that the input voltage is above the VIN\_ON threshold. The polarity of this pin is set by the ON\_OFF\_CONFIG command.

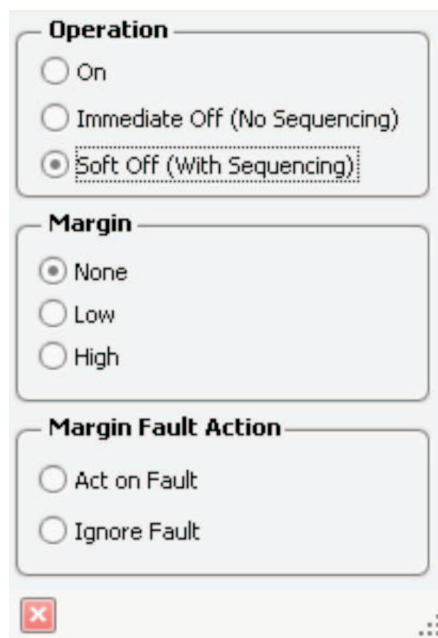
Note that the Fusion GUI uses the Texas Instruments' USB to GPIO adapter to communicate with a UCD92xx device. The adapter has several general-purpose digital I/Os in addition to the I<sup>2</sup>C interface signals. Pin 7 of the 10-pin ribbon cable is defined as the PMBus CTRL signal. When a user sets the Control line high or low in the Fusion GUI, then, the user drives this pin on the ribbon cable connector.

---

**NOTE:** It is up to the user to connect this pin on the PMBus ribbon cable connector to the device.

---

When the *Operation Only* option is selected, the start indication is defined by the time that the device receives an OPERATION command over the PMBus, **and** the time that the input voltage is above the VIN\_ON threshold. The OPERATION command can be used for several actions: to turn on all rails; turn on individual rails; turn on the rail at margin high or margin low; immediately turn off a rail; or turn off a rail with a specified soft-off ramp. If each rail is configured to respond to the operation command and the PMBus PAGE is set to 0xFF, all rails on the device respond to this command. Figure 4-2 illustrates the OPERATION command options.



**Figure 4-2. Fusion GUI: OPERATION Command Options**

A voltage rail can also be configured to start when all three conditions are true:

- The input voltage is above the VIN\_ON threshold;
- The CTRL pins is active; and
- The OPERATION command to turn on is received by the device.

### 4.3 TON\_DELAY and TON\_RISE

The simplest technique to sequence multiple voltage rails is to assign each regulated voltage a delay relative to the start or stop indication. Then, define a time to ramp to the regulated voltage defined by VOUT\_COMMAND. In the same manner, the turn-off sequence of multiple voltage rails can be controlled by defining a delay from the stop indication. The PMBus commands that accomplish this process are:

- **TON\_DELAY:** The time (in ms) to delay after the start indication occurs before beginning the soft-start ramp. ( $V_{IN}$  above VIN\_ON, control pin active, or an OPERATION command directs a rail to turn on.)
- **TON\_RISE:** The time (in ms) to complete the soft-start ramp. This time defines the ramp rate assuming the output voltage is starting from ground. If there is a pre-existing voltage (pre-bias) on the regulated output, the controller waits until the ramp from zero equals the pre-bias value before starting the control action on the PWM and SRE pins. This period ensures that the voltage reaches the final value set by TON\_DELAY + TON\_RISE, regardless of the pre-bias voltage on the output voltage rail.
- **TOFF\_DELAY:** The time (in ms) to delay after the stop indication occurs before beginning the ramp down.
- **TOFF\_FALL:** As with TON\_RISE, this command specifies the time (and therefore, the rate) at which the ramp down occurs.

In all cases, when the on-signal (command or input) is received, the order in which the rails are turned on is set by the timing parameters.

#### **Example 4-1. Rail Turn-On**

Given a two-rail system where the 2.5-V Rail #1 should turn on before the 1.0-V Rail #2, and both rails should ramp at 200 mV/ms.

TON\_DELAY [Rail #1]= 0 ms

TON\_RISE [Rail #1]= 12.5 ms (2.5/0.2)

TON\_DELAY [Rail #2]= 4.0 ms

TON\_RISE [Rail #2]= 5.0 ms (1.0/0.2)

Refer to [Figure 4-3](#) for an example of the Fusion GUI screen that is used to set the TON\_DELAY and TON\_RISE parameters.

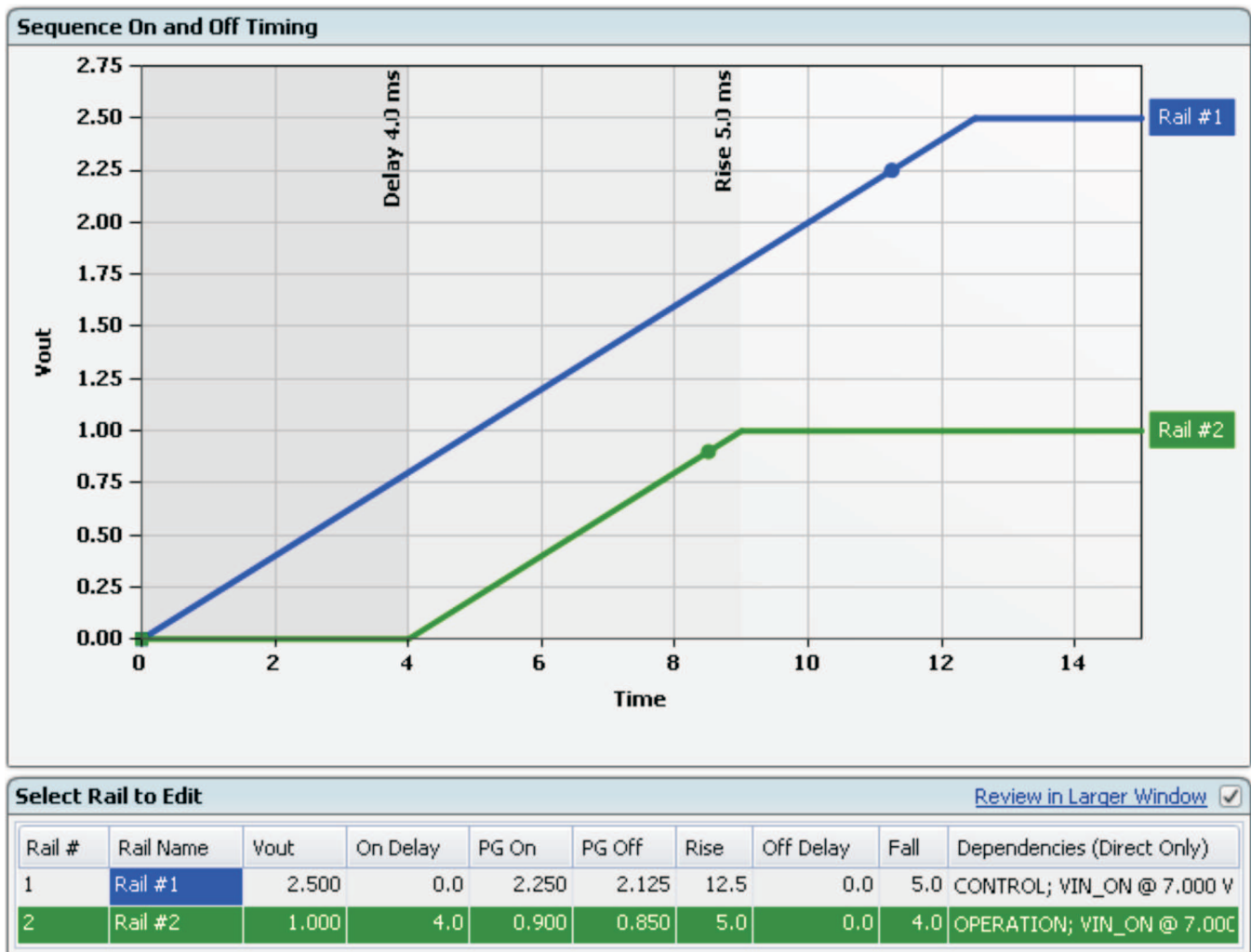


Figure 4-3. Fusion GUI: Setting TON\_DELAY and TON\_RISE

#### 4.4 Group Commands

The PMBus specification (Rev 1.1), section 5.2.3, states:

PMBus devices must support the Group Command Protocol. The Group Command Protocol is used to send commands to more than one PMBus device. The commands are sent in one continuous transmission. When the devices detect the STOP condition that ends the sending of commands, they all begin executing the command they received.

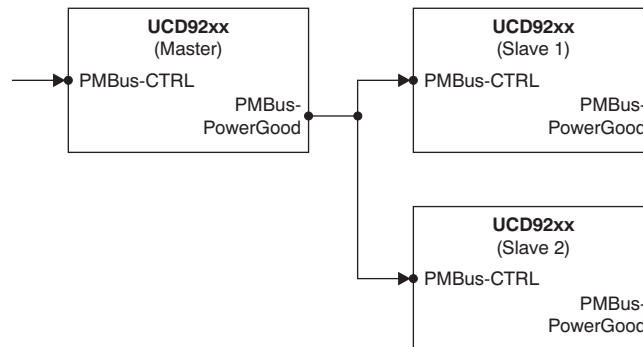
To sequence the start-up of multiple devices over the PMBus, each voltage rail should be configured individually using the PMBus TON Commands as described in Section 4.3. Then, the Group Command Protocol can be used to issue the OPERATION command to turn on (or off) all of the configured devices simultaneously. When the command is received, each device operates independently, but at a common starting time point with other devices. This configuration results in a synchronized turn-on sequence across multiple rails on multiple UCD92xx devices (or across other devices that also support both the group and operation commands).

The Group Command Protocol can be tested in the Fusion GUI by selecting *Group Command Protocol Tester ...* from the GUI **Tools** menu.

## 4.5 Daisy-Chaining CTRL and PowerGood Pins

One straightforward method to establish sequencing between multiple UCD92xx devices is to simply cascade multiple parts by routing the power-good signal from one device into the control pin of the next. This configuration does constrain the system by requiring that all rails on the first (or master) device gate all rails on subsequent (or slave) devices. During start-up, the slave controllers initiate the respective start sequences only after the master has completed its entire start sequence, and after all rails have reached the respective regulation voltage(s). During shutdown, as soon as the master starts its shutdown sequence, it sends the shutdown signal to the slaves.

In the example shown in [Figure 4-4](#), one master device gates the startup of two slave devices. Each of the three devices is configured independently, and a single CTRL input to the master initiates the startup sequence.



**Figure 4-4. Daisy-Chain Sequence Example**

As we discuss further on in this document, the PowerGood signal can be modified so that it goes active for a subset of the all the voltage rails controlled by the master UCD92xx device.

## 4.6 Voltage Tracking

*Tracking* allows one voltage rail to follow a master source. This master source can be an voltage rail controlled by the UCD92xx or an external signal. Each rail can be independently configured to track any other internal rail or the external input pin,  $V_{\text{TRACK}}$ , using the command TRACKING\_SOURCE. [Table 4-1](#) lists the tracking options.

**Table 4-1. Tracking Options**

TRACKING_SOURCE	Master
< 0	All negative values disable tracking.
0	Internal Rail #1
1	Internal Rail #2
2	Internal Rail #3
3	Internal Rail #4
≥ 4	All values ≥ 4 select the external $V_{\text{TRACK}}$ input

The Fusion GUI simplifies this selection process with a control selection screen, shown in Figure 4-5.

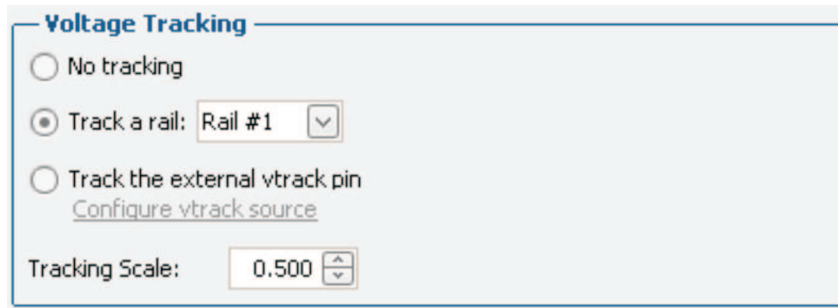


Figure 4-5. Fusion GUI: Voltage Tracking Control Options

The tracking scale can be specified using the command TRACKING\_SCALE\_MONITOR, which defines a linear relationship between a voltage rail and its master. For example, slave Rail #2 can track master Rail #1 at half of its voltage as illustrated in Figure 4-6.

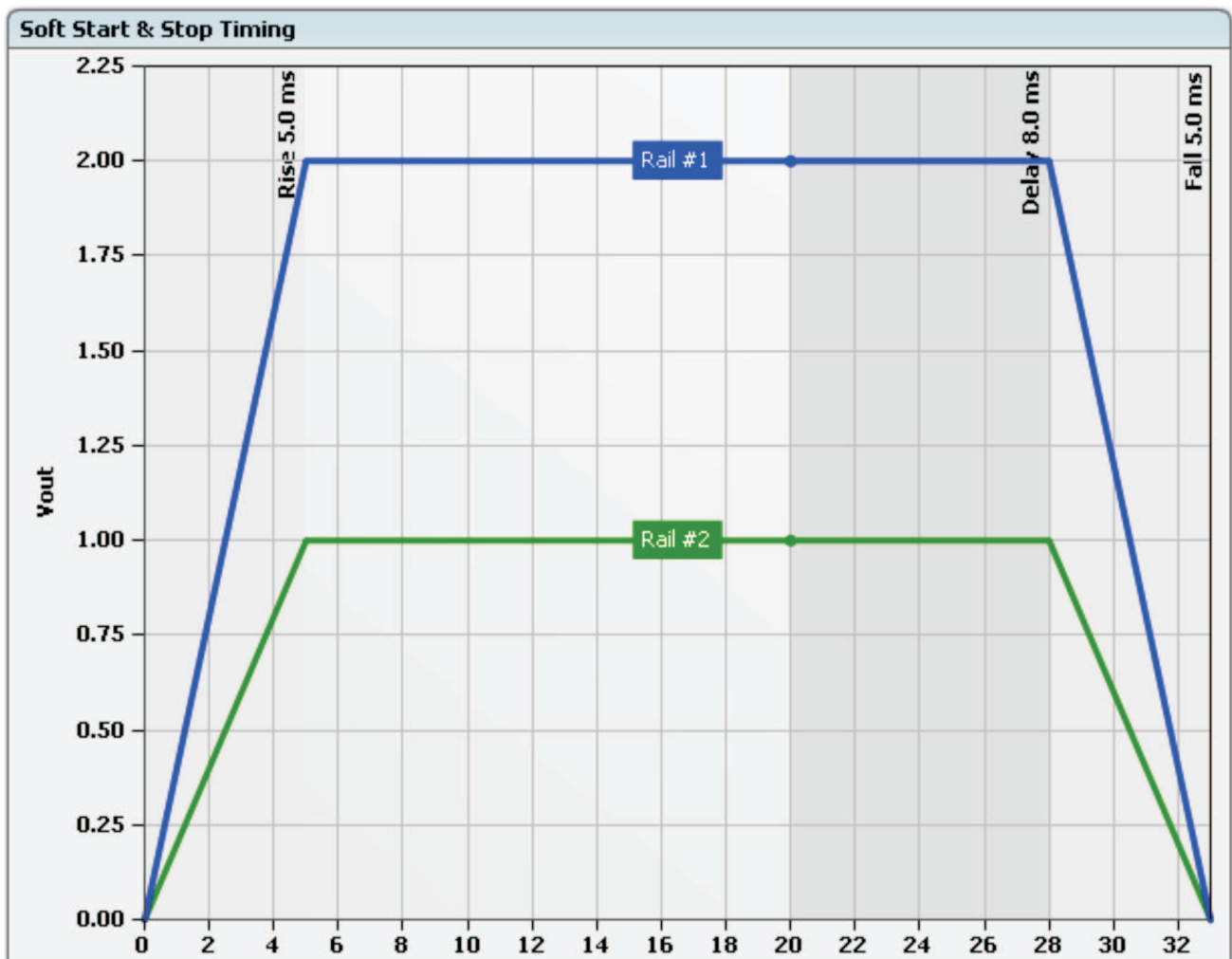


Figure 4-6. Tracking Example

As in standard non-tracking mode, a target  $V_{OUT}$  is specified for the voltage rail. If the master input exceeds this target, the slave rail stops following the master, switches to the Bank-1 regulation compensation, and regulates at the target voltage. When the master drops back below the target (with 20 mV of hysteresis), tracking compensation (Bank-0) restarts and the slave rail again follows the master reference. Note that the target can be set far above the range of the tracking input, forcing the slave rail to always remain in tracking mode.

When configured for tracking, slave rails follow the master device and ignore the PMBus timing parameters (TON\_DELAY, TON\_RISE, etc.).

## 4.7 GPIO\_SEQ\_CONFIG

In addition to the TON and TOFF commands and voltage tracking, any unused digital I/O pin on the UCD92xx can be configured to be either an enable input or a PowerGood output. Once the pins are defined as a sequencing input or output, they can be used to define dependencies between voltage rails.

Several features are combined in a single command: GPIO\_SEQ\_CONFIG. The options are grouped into a single command because they all may depend on the hardware configuration of the UCD92xx digital I/O, and writing these features together prevents order-of-operation dependencies that could result in an invalid configuration if each instruction was processed independently. The [UCD92xx Digital PWM System Controller PMBus Command Reference](#) should be referenced as well for additional information. There are seven parts to the GPIO\_SEQ\_CONFIG command:

- *Turn On Dependencies* define the sequencing requirement (a combination of internal rails and external inputs) to turn on a rail.
- *Stay On Dependencies* define the sequencing requirement (a combination of internal rails and external inputs) to keep a rail on (without these dependencies, the rail turns off).
- *Output Dependencies* set the type of output; if sequencing, these instructions determine the rails that are involved.
- *Input Pin Configuration* selects the pins to use for inputs and specifies the respective polarity of each.
- *Output Pin Configuration* selects the pins to use for outputs and specifies the respective polarity of each, as well as whether the pins are actively driven or open-drain.
- *Fault-Slaves Masks* allow a fault on one rail to shut down multiple rails.
- *Misc Configurations* includes these options:
  - Sequence Timeout Source selects the input (if any) that is used in conjunction with the SEQ\_TIMEOUT command to time a required external event.
  - Shutdown Mode Configuration determines whether the soft-stop parameters (TOFF\_DELAY and TOFF\_FALL) are used when the Stay On Dependencies are not met, or whether the rail shuts down immediately.

[Table 4-2](#) lists all the digital I/O pins for each device. Each pin is available for use as sequencing control, provided it is not being used for its primary purpose. Each item shows the input/output configurability of the pin and the initial state of the pin, before configuration.



**Table 4-2. Available UCD92xx Sequencing Pins**

Pin Name	UCD9248	UCD9246	UCD9224
DPWM-1A	In/Out - low	In/Out - low	In/Out - low
DPWM-1B	In/Out - low	In/Out - low	In/Out - low
DPWM-2A	In/Out - low	In/Out - low	In/Out - low
DPWM-2B	In/Out - low	In/Out - low	—
DPWM-3A	In/Out - low	In/Out - low	In/Out - low
DPWM-3B	In/Out - low	—	—
DPWM-4A	In/Out - low	In/Out - low	—
DPWM-4B	In/Out - low	—	—
<hr/>			
FAULT-1A	In/Out - Hi-Z	In/Out - Hi-Z	In/Out - Hi-Z
FAULT-1B	In/Out - Hi-Z	In/Out - Hi-Z	In/Out - Hi-Z
FAULT-2A	In/Out - Hi-Z	In/Out - Hi-Z	In/Out - Hi-Z
FAULT-2B	In/Out - Hi-Z	In/Out - Hi-Z	—
FAULT-3A	In/Out - Hi-Z	In/Out - Hi-Z	In/Out - Hi-Z
FAULT-3B	In/Out - Hi-Z	—	—
FAULT-4A	In/Out - Hi-Z	In/Out - Hi-Z	—
FAULT-4B	In/Out - Hi-Z	—	—
<hr/>			
SRE-1A	In/Out - Hi-Z	In/Out - low	In/Out - Hi-Z
SRE-1B	In/Out - Hi-Z	In/Out - low	In/Out - Hi-Z
SRE-2A	In/Out - Hi-Z	In/Out - Hi-Z	In/Out - low
SRE-2B	In/Out - Hi-Z	In/Out - Hi-Z	—
SRE-3A	In/Out - Hi-Z	In/Out - Hi-Z	In/Out - low
SRE-3B	In/Out - Hi-Z	—	—
SRE-4A	In/Out - Hi-Z	In/Out - Hi-Z	—
SRE-4B	In/Out - Hi-Z	—	—
<hr/>			
PGOOD	In/Out - Hi-Z	In/Out - Hi-Z	In/Out - Hi-Z
SEQ1	In/Out - Hi-Z	In/Out - Hi-Z	In/Out - Hi-Z
SEQ2	In - Hi-Z	In - Hi-Z	In/Out - Hi-Z
SEQ3	In - Hi-Z	—	In/Out - Hi-Z

Most device pins can be used for either input or output (*In/Out*); note that a few are limited for use as input only (*In*). When the device is powered up there are approximately 15  $\mu$ s when the state of the pin is set by a hardware default to be either driven low (*low*) or 3-stated (*Hi-Z*). If the output state during this time is important, the correct type of pin should be selected; otherwise, its output should be gated with the signal OD\_VALID.

The underlying arrays can be configured manually, but it is far more intuitive to use the GUI to configure the more advanced sequencing capabilities provided with GPIO\_SEQ\_CONFIG. The [UCD92xx Digital PWM System Controller PMBus Command Reference \(SLUU337\)](#) contains the required bit-field definitions to manually set these parameters.

Note that if a voltage rail does not start or shuts down because the turn-on or stay-on dependencies are not met, there is no fault status indication. The dependency (internal or external signal) acts as an OR with the CTRL line or OPERATION command, and is treated by the device as a command to turn on or off.

#### 4.7.1 Configuration for Sequencing Output

In the Fusion GUI, the various sections of the command are broken out into the respective components; this design provides a more intuitive method of configuration. The configuration is set using the GPIO Config tab, shown in [Figure 4-7](#).

Vout Config Other Config Phase/Rail Config **GPIO Config** CLA Coefficient Banks Advanced Config

### IC Pin Configuration

Pin #	Name	Supports	Polarity	Configuration
11	FLT-1A	I/O	---	Rail #1 Phase 1A Fault Input (Primary Function)
12	FLT-1B	I/O	Active Low	GPO: Regulating on Rail #2,3

**Pin Mode**

Rail #1 Phase 1B Fault Input (Primary Function)  
 GPI (Sequencing Input)  
 GPO (Output)  
 Configure under what conditions the pin is active below

**Pin Configuration**

Polarity:  Active Low  Active High  
 Output Mode:  Actively Driven  Open-Drain

**Output Mode Configuration**

Regulating  
 Select which rails that, when regulating at the commanded voltage, will cause the pin to be active. Every rail checked must be regulating to set the output pin active.  
 Rail #1  Rail #2  Rail #3  Rail #4

Power Good  
 The pin is active when the POWER\_GOOD condition is met (all configured rails are on and within their limits).

Open-Drain outputs – status flag  
 This pin is active when other sequencing output pins configured as open-drain have been initialized.  
 Open-Drain Active Low: when the control variable is TRUE, the output pin is actively pulled low by the controller.  
 Open-Drain Active High: when the control variable is TRUE, the output pin is passively pulled high by an external pull-up resistor.

Over-Current Warning  
 Select which rails that, when the IOUT\_OC\_WARN limit is exceeded, will cause the pin to be active. An OCW condition on any of the rails checked will activate this pin.  
 Rail #1  Rail #2  Rail #3  Rail #4

**Sequencing Timeout**

Sequencing Timing Source: No input pins have been defined yet

Sequencing Timeout:

Rail #1	Rail #2	Rail #3	Rail #4
0.0 ms	0.0 ms	0.0 ms	0.0 ms
<input checked="" type="checkbox"/> Disable	<input checked="" type="checkbox"/> Disable	<input checked="" type="checkbox"/> Disable	<input checked="" type="checkbox"/> Disable

After a rail reaches its POWER\_GOOD\_ON threshold, it will monitor the pin specified above for the duration configured. If during that time, the input pin does not reach the polarity configured in the IC Pin Configuration, the rail will be shut down and a Fault will be posted.

Figure 4-7. Fusion GUI: GPIO Config Tab

After selecting an unused pin for configuration as an output and declaring its polarity, its function within the system can be defined. For use as a sequencing output, the radio-button *Regulating* should be selected, and then the rail(s) on which the output depends are marked with checkboxes. The sequencing output is the logical *AND* of the checkboxes selected; that is, all of the selected rails must have exceeded the respective *POWER\_GOOD\_ON* thresholds in order for the output to be enabled.

If, for example, unused FLT-1B is configured as an output to reflect the power-good state of Rail #2 AND Rail #3, that pin is active only when both Rail #2 and Rail #3 have met the individual power-good requirement. As soon as either of these rails drops below the respective *POWER\_GOOD\_OFF* threshold, the output deasserts. The output signal reflected on pin FLT-1B in this case can then be routed to another device either through that device control pin or another pin configured as a sequencing input.

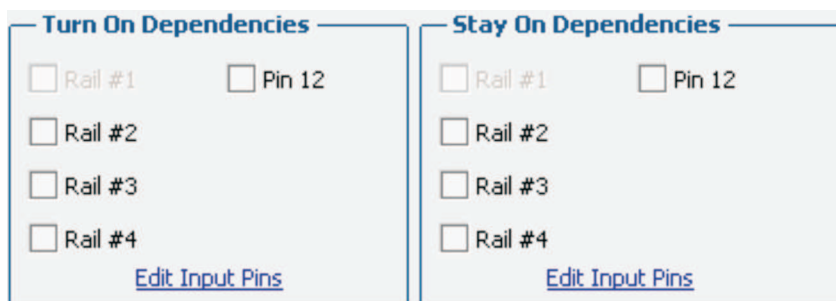
### 4.7.2 Configuration for Sequencing Inputs

Undedicated pins can also be configured to be sequencing inputs. The control of the state of a regulated voltage rail may be influenced by a combination of input pins and/or depend on other internal rails. Furthermore, a single input pin may affect multiple rails, or it may be used as an input for timing rather than sequencing. For these reasons, input pin configuration is separated from the sequencing configuration for a rail. This process is different from how sequencing output pins are configured.

When a pin is configured as an input from the GPIO Config menu, two results occur:

- The Output Mode section (refer to [Figure 4-7](#)) is grayed out; and
- On the *Vout Config* tab, the newly configured pin appears as an option in the **Turn On Dependencies** and **Stay On Dependencies** boxes.

As shown in [Figure 4-8](#), the newly configured input: *Pin-12*, is available. (Rail #1 is grayed out because we are configuring Rail #1, and a dependency on itself is meaningless.)



**Figure 4-8. Fusion GUI: Dependency Control Options**

---

**NOTE:** The ability to have both turn-on dependencies and stay-on dependencies is a new feature of the UCD9224/46/48 devices. The UCD9220/40 controllers have only a turn-on dependency. This document describes the newer UCD9224/46/48 behavior.

---

The turn-on dependencies, as defined by the *GPIO\_SEQ\_CONFIG* command, establish additional gating conditions beyond what are configured in the *ON\_OFF\_CONFIG* command. When enabled, the dependencies must also be satisfied before a rail can be turned on. Once the rail is turned on, these turn-on dependencies become irrelevant and the stay-on dependencies assume influence over the rail behavior. This hand-off of dependency occurs when the relevant voltage rail PowerGood threshold is satisfied. By splitting the dependencies into two groups, it is possible to allow the turn-on sequencing to differ from the turn-off sequencing. For example, this architecture allows the controlled voltage rails to turn on in a given order such as 1, 2, 3, and turn off in the opposite order: 3, 2, 1.

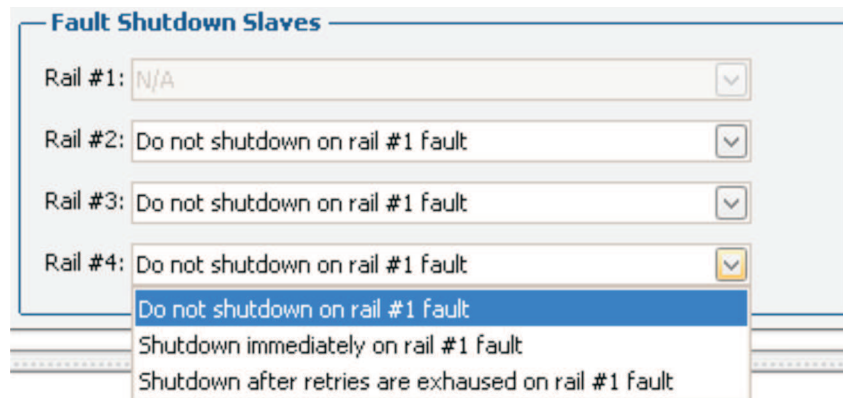
A further important point regarding the stay-on dependencies is that these controls are edge-dependent signals. In order to force a rail to turn off, the stay-on dependent signal (internal or external) must

transition from its active stay-on polarity to its inactive polarity. For example, consider that Rail #1 is configured to have stay-on dependencies based on Rail #2 and Rail #3. With no turn-on dependencies, any one of the start indications causes the voltage rail to turn on, regardless of the fact that neither stay-on dependency is met. However, once Rail #1 is on, either of the stay-on dependencies can cause it to turn off by turning on and then turning off.

When a rail is turned off because of the detected loss of a stay-on dependency, it turns off in the manner directed by the bits in the Shutdown Mode Configuration bit (defined later in this document).

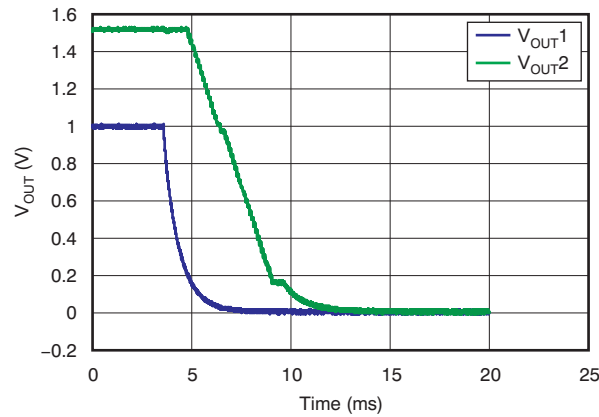
### 4.7.3 Fault Slaves

The GPIO\_SEQ\_CONFIG command allows the configuration of *fault slaves* such that a fault on one voltage rail can result in the shutdown of other voltage rails that are regulated by the same controller. This process is similar to the stay-on dependencies (see [Section 4.7.2](#)) except that the fault slaves are associated with the rail experiencing the fault and stay-on dependencies are associated with the rail to be shut down. The primary difference between these two features is that for fault slaves, the slave voltage rails can be configured to shut down either immediately on a master fault or wait until the master has exhausted all retry attempts. When the slave finally shuts down in either case, it uses the *off-immediately* (as opposed to *soft-off*) mode of shutdown, regardless of its programmed shutdown mode. If both a stay-on dependency and a fault slave are defined for a voltage rail, the stay-on dependency takes precedence. That is, when a fault occurs on a voltage rail that another rail has been defined to depend on, the dependent voltage rail does not wait for retries to be completed, but starts the shutdown sequence based on the soft-off or immediate-off setting of the stay-on dependency. [Figure 4-9](#) illustrates the fault shutdown slave control screen for the Fusion GUI software.



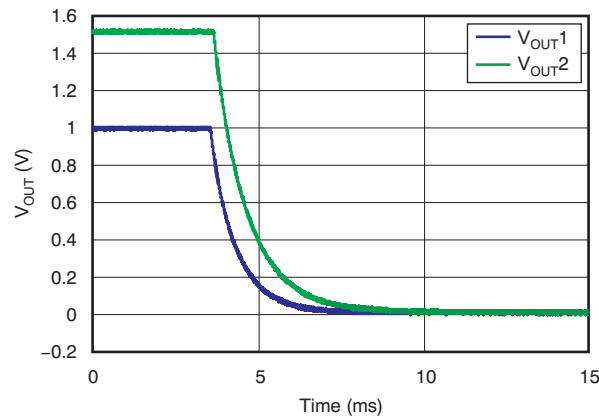
**Figure 4-9. Fusion GUI: Fault Shutdown Slave Control**

The following graphs (presented in [Figure 4-10](#) through [Figure 4-12](#)) show how voltage Rail #2 behaves when a fault occurs on Rail #1 and Rail #2 depends on Rail #1. In the first example, voltage Rail #2 is configured to have a stay-on dependency with Rail #1 and the shutdown mode is set to *soft-off*. To set this configuration with the Fusion GUI, Rail #2 is selected and the box for Rail #1 is checked under **Stay-on dependencies**.



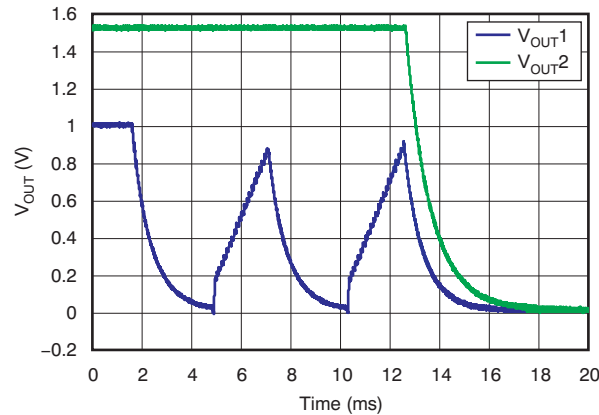
**Figure 4-10. Fault on Rail #1; Rail #2 has a Rail #1 Stay-on Dependency with Soft-Off**

In this case, Rail #1 turns off immediately because of the fault; Rail #2, the dependent rail, goes through the soft-off sequence, which was set for a delay of 1.0 ms and a soft-off ramp time of 4.5 ms. Note that the controller ramps the voltage on Rail #2 down until the pulse width is at the minimum set by DRIVER\_MIN\_PULSE, then turns off the MOSFETs.



**Figure 4-11. Fault on Rail #1; Rail #2 has a Rail #1 Stay-on Dependency with Immediate-Off**

With the shutdown mode set to *immediate-off*, the switching action on Rail #2 shuts down as soon as the stay-on dependency (Rail #1) goes away.



**Figure 4-12. Fault on Rail #1—Rail #2 is a Fault Slave for Rail #1; Shutdown After Retries**

In [Figure 4-12](#), the stay-on dependency set for Rail #2 has been removed and a fault-slave dependency for Rail #2 has been set for Rail #1. In addition, the shutdown mode has been set to wait until retries are exhausted. To create these waveforms, the fault on Rail #1 was generated by setting the OV fault limit to less than the  $V_{OUT}$  Command voltage and the OV fault response was set to shut down immediately with two retries. Here, voltage Rail #1 is set to regulate at 1.0 V and throws an OV fault when the OV fault limit is set to 0.8 V. It then tries to restart, but throws an OV fault again when the regulated voltage exceeds 0.8 V. It proceeds through this sequence for the configured two retries, then shuts down permanently, taking Rail #2 with it. (According to the PMBus standard, at this point in order to restart the voltage rails, the OPERATION or CTRL pin must be commanded to turn the faulting rail off; then the CLEAR\_FAULTS command are sent to the device.)

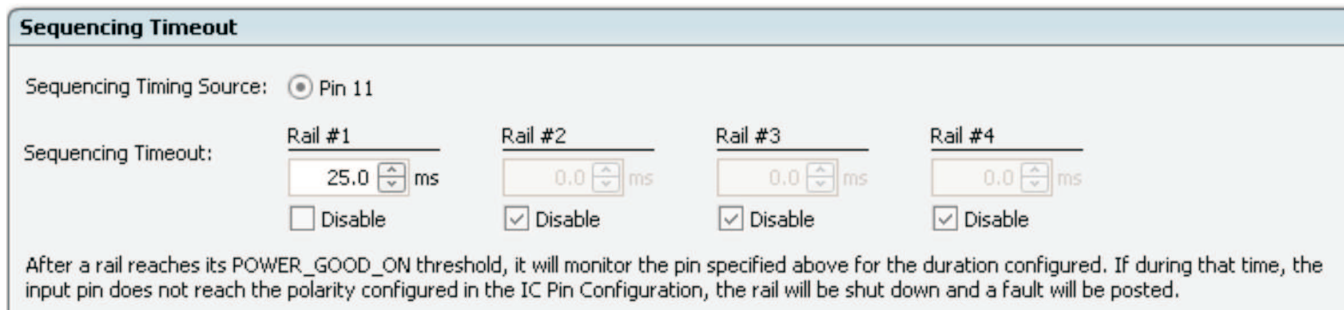
#### 4.7.4 Sequencing Timeout

Each rail can be optionally configured to monitor a sequencing input pin for a specified period of time after it turns on and reaches its power-good threshold. If the programmable timeout is reached before the input pin state matches its defined logic level, the rail shuts down and a status error is posted. This feature could be used, for example, to ensure that an LDO on the board did in fact turn on when the main system voltage came up. Each rail is enabled independently of the other rails and has a unique timeout value; a single input pin is used as the timeout source for all rails.

To enable this feature, two sections of the GPIO\_SEQ\_CONFIG must be initialized. First, at least one pin must be configured as an input in the *Input Pin Configuration* section. Second, it must be identified by the *Sequence Timeout Source Selection*.

Beyond the GPIO\_SEQ\_CONFIG command, each rail has a timeout value defined by the PMBus command SEQ\_TIMEOUT. Setting SEQ\_TIMEOUT to '0' disables the timeout function for that rail.

Configuration of the sequencing timeout function is found on the *GPIO Config* tab below the **IC Pin Configuration** section (again, this process is greatly simplified by the GUI). All pins configured as inputs are listed as potential sequencing timing sources, as shown in [Figure 4-13](#).



**Figure 4-13. Fusion GUI: Sequencing Timeout Control Screen**

Here, Rail #1 is configured to wait 25 ms for pin #11 to be driven to its defined polarity. Each time Rail #1 is turned on, it monitor spin #11 for the specified 25 ms; if it does not match its active polarity within that period, then Rail #1 shuts down and posts an error.

## 4.8 Alternative/Additional PowerGood Outputs

In addition to functioning as sequencing input signals, any of the pins in [Table 4-2](#) that can operate as an output can also be configured to function as a digital output status pin, conveying one of three possible signals. These options are set by the four bits in the Event Type field in the Output Dependency Configuration bytes, as shown in [Table 4-3](#).

**Table 4-3. Output Dependency Configuration Bytes**

Bit	7	6	5	4	3	2	1	0
Purpose	Event Type				PAGE3	PAGE2	PAGE1	PAGE0

- **Event Type = 0001: Regulating.** Used for a sequencing output as discussed earlier with an *AND* of the power-good signal for rails specified by the lower four PAGE bits.
- **Event Type = 0010: Power Good.** Reflects the PMBus-defined power-good signal (all configured rails above POWER\_GOOD\_ON). Note that this event is similar to Event Type 0001 with all rails selected, but not identical; this setting responds faster, especially under fault conditions.
- **Event Type = 0011: Open-Drain Outputs.** Status flag; generates an output used to gate any pins configured as open-drain. It toggles to its active state when the device has come out of reset and executed the hardware configuration routines.
- **Event Type = 0100: Overcurrent Warning.** The output is active when any (logical *OR*) of the rails identified by the lower four PAGE bits exceeds its IOUT\_WARNING level.





## ***Device Operation***

---

---

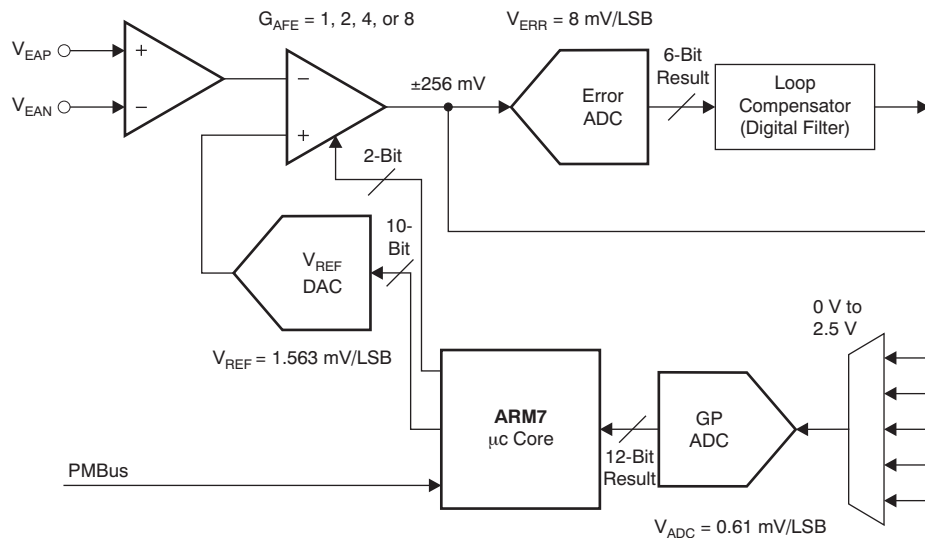
---

This chapter presents a functional description of several key circuits in the UCD92xx. It also describes the behavior of the device during the soft-start ramp.

<b>Topic</b>	<b>Page</b>
<b>5.1 AFE Overview .....</b>	<b>74</b>
<b>5.2 Synchronizing Multiple PWM Signals .....</b>	<b>75</b>
<b>5.3 The Soft-Start Ramp .....</b>	<b>78</b>
<b>5.4 PWM Timing .....</b>	<b>80</b>
<b>5.5 Inrush Current During Kick-Start .....</b>	<b>80</b>
<b>5.6 Current Balance .....</b>	<b>83</b>
<b>5.7 Temperature Balance .....</b>	<b>86</b>

## 5.1 AFE Overview

Figure 5-1 illustrates a block diagram of the analog front-end component.



**Figure 5-1. UCD92xx Analog Front-End**

In the UCD92xx digital controllers, the gain through the AFE can be configured to be one of four values. This flexibility allows the user to trade off effective resolution for dynamic range. There are two conditions that define the required dynamic range:

- Load step response
- Soft-start ramp rate

Table 5-1 summarizes the AFE settings.

**Table 5-1. AFE settings**

AFE Gain	Resolution at Error ADC Out	Gain (LSB/V <sub>EA</sub> )	Dynamic Range	Maximum Soft-Start Ramp Rate
8x	1 mV	1000 LSB/V	-32 mV / +31 mV	240 mV/m
4x	2 mV	500 LSB/V	-64 mV / +62 mV	480 mV/ms
2x	4 mV	250 LSB/V	-128 mV / +124 mV	960 mV/ms
1x	8 mV	125 LSB/V	-256 mV / +248 mV	1.920 V/ms

During a fast change in the load, the output voltage changes by a certain amount before the controller can respond with an increase (or decrease) to the voltage regulation control effort (that is, before there is a change to the MOSFET duty cycle). During this period, it is important that the error ADC does not saturate.<sup>(1)</sup> Therefore, the AFE gain must be selected so that the largest expected disturbance is less than the resulting dynamic range.

To perform a soft-start, the UCD92xx controller steps the V<sub>REF</sub> DAC from the off-state voltage to the desired regulated voltage under control of the microcontroller. The microcontroller can update the DAC at a rate of 10 kHz; consequently, the microcontroller updates the DAC approximately every 100 µs. For a 5-ms soft-start, then, there are 50 discrete steps that make up the soft-start voltage ramp. For each step, there must be sufficient dynamic range such that the error ADC does not saturate. To prevent saturation, the UCD92xx limits the voltage change for any one step to be less than 75% of the configured positive dynamic range. This restriction puts an effective limit on the maximum rate at which the output voltage can be ramped during the soft-start.

<sup>(1)</sup> The voltage regulation loop is compensated by placing zeroes in the compensation filter such that they cancel the poles of the power stage LC output filter. The compensating filter creates the action of the zeroes by comparing the difference between consecutive voltage samples from the error ADC. If the signal is saturated at -32 LSBs or +31 LSBs, the difference is zero and the effect of the compensating zeroes is lost.

Thus far, we have discussed why a too-narrow dynamic range is a problem. Now we will consider the opposite situation: Why not set the AFE gain to the 1x setting and use the largest dynamic range possible? This configuration increases the effective resolution, which introduces quantization error. This report addresses that primary issue: the effect of error ADC resolution on the regulation performance of the power supply.

## 5.2 Synchronizing Multiple PWM Signals

When it is desired to synchronize the PWM timing to an external reference, the reference clock is applied to the Sync\_In pin and the SYNC\_IN\_OUT PMBus command is configured to select the rails that are to be synchronized. In this case, the ramp counter for phase 0 of each rail that is configured to accept the sync clock reference is reset on the rising edge of the Sync\_In signal. Note that because of the way the internal synchronization is configured (where the phase relationship is set up once and the digital counters are then allowed to run freely), in most applications you should enable *all* rails when external synchronization is desired. Figure 5-2 shows the synchronization options that are configured in the SYNC\_IN\_OUT command.

Select zero or more rails to be a slave to the input sync signal. A single rail may optionally be selected to drive the master sync out signal.

	<u>Slave?</u>	<u>Master?</u>
Rail #1 :	<input type="checkbox"/>	<input type="radio"/>
Rail #2 :	<input type="checkbox"/>	<input type="radio"/>
Rail #3 :	<input type="checkbox"/>	<input type="radio"/>
Rail #4 :	<input type="checkbox"/>	<input type="radio"/>
		<input checked="" type="radio"/> No Master

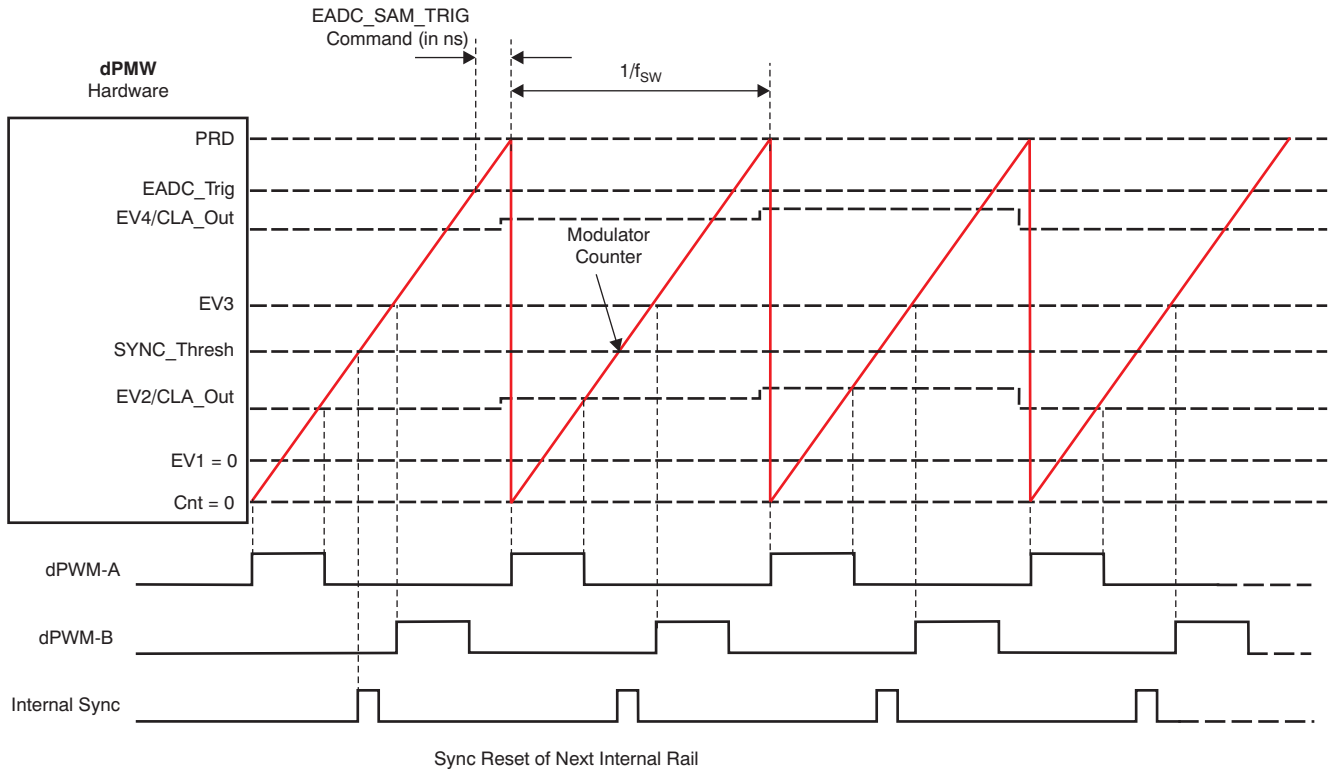
**Figure 5-2. Synchronization Options**

To adjust the phase of the PWM on-time pulse relative to the sync signal, use the PHASE\_OFFSET command. The PHASE\_OFFSET command delays the start of the on-time pulse relative to the modulator ramp counter. Because a sudden change to the PWM timing could cause a large voltage disturbance, the PHASE\_OFFSET command must be issued with the voltage rail turned off (using the OPERATION command or CTRL signal).

To generate a sync clock output, one of the controlled voltage rails is selected as the master in the SYNC\_IN\_OUT command. The SyncOut pin then produces a ~100-ns pulse coincident with the rising edge of the phase 0 PWM signal for that rail.

### 5.2.1 Description of the Digital Demodulator Logic

Figure 5-3 shows a diagram of the timing generated by the pulse width modulator logic using internal synchronization. For the *A* PWM output signals, the turn-on time is defined by the value of the EV1 register. The turn-off time is set by the EV2 register. The EV2 register is normally fed by the product of the PRD register and the output of the compensating digital filter. For the *B* PWM outputs, there are corresponding EV3 and EV4 registers.

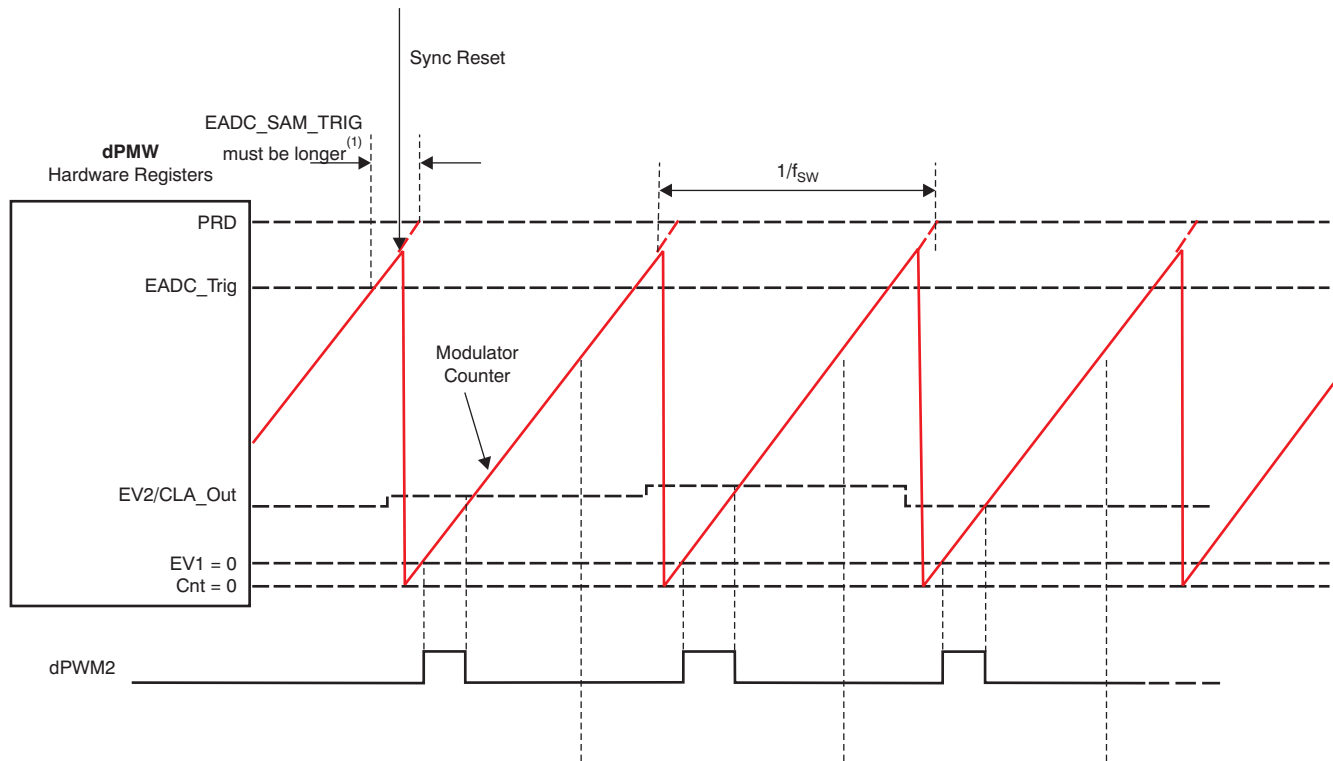


**Figure 5-3. PWM Timing with Internal Synchronization**

Each modulator also generates a sync signal that is timed from the main modulator counter. The phase of this signal is set by the firmware based on the page and phase configuration. The sync signal is then routed to the next higher PWM engine to reset its counter and define the phase between each PWM engine.

When it is desired to synchronization a voltage rail to an external clock signal, on the other hand, that rail is configured as a sync slave with the SYNC\_IN\_OUT command. When a sync clock is received on the SyncIn pin, the dPWM modulation counter for that rail is reset on the rising edge of the external sync clock. It is necessary, therefore, to set the free-running switching period to be longer than the expected period of the sync clock. In other words, the value of FREQUENCY\_SWITCH should be lower than the expedited value of the sync clock. The phase of the PWM signal relative to the sync clock can be adjusted by defining a positive value in the PHASE\_OFFSET command. The PHASE\_OFFSET command sets the counts in the Event 1 and Event 3 threshold registers that define the turn-on time of the PWM-A and PWM-B outputs. Changing these registers causes a sudden change to the duty cycle and thus the regulated voltage; consequently, the UCD92xx requires that the output be off when these registers change.

Figure 5-4 shows a diagram of the timing generated by the pulse width modulator logic using external synchronization.



(1) EADC\_SAM\_TRIG must be longer than in unsynchronized case.

Figure 5-4. PWM Timing with External Synchronization

### 5.2.2 Effect of EV1

Assume that Sync\_In is used to reset the main counter for dPWM1A. Furthermore, assume that the application requires that dPWM1-A be delayed relative to the Sync\_In signal. This condition is defined in the PHASE\_OFFSET command, and the controller accomplishes this action by writing to the EV1 register for dPWM1A. As a consequence, the on-time pulse PWM pulse is delayed relative to the modulation counter ramp. This delay increases the time from when the output voltage is sampled to the falling edge of the on-time pulse. This increased latency adds a phase loss to the feedback loop, and must be taken into account when setting the loop bandwidth. The Fusion GUI makes an automatic adjustment for this delay, resulting in a lower loop bandwidth and lower load transient performance (compared to what would occur without the delay). Therefore, it is best to look for a synchronization arrangement that allows EV1 to be zero, or as small as possible.

### 5.2.3 Setting EADC\_SAMPLE\_TRIGGER Value

The EADC\_SAMPLE\_TRIG PMBus command sets the time required to sample the regulated voltage and calculate the control effort before the start of the next PWM pulse. This period is typically 224 ns to 240 ns; the actual time depends on the modulo of the switching period register and the EADC\_SAMPLE\_TRIGGER register. To implement the command, the controller converts the specified pre-trigger time to a threshold value that is compared to the main counter in the modulator.

This approach works very well when the modulator is set to run freely. However, when the modulator is synchronized to the external signal applied to the Sync\_In pin, the EADC\_SAMPLE\_TRIG value must be increased. To see this effect, consider Figure 5-3. When synchronizing to an external signal, that signal must be at a higher frequency than the internally-configured value. However, the internal register that times when to sample the output voltage is calculated based on the configured switching frequency, *not* the actual sync signal frequency.

Therefore, in order to ensure that the output voltage is not sampled too late, the EADC\_SAMPLE\_TRIGGER value must be increased according to Equation 60:

$$EADC\_SAMPLE\_TRIGGER \geq \frac{1}{F_{SW}} - \frac{0.95}{F_{SYNC}} + 248 \text{ ns} \tag{60}$$

### 5.3 The Soft-Start Ramp

The UCD92xx digital PWM controllers perform a soft-start (SS) ramp-up by using the stored values from the PMBus configuration commands TON\_DELAY, TON\_RISE, and VOUT\_COMMAND to define an ideal SS ramp trajectory. The device then measures  $V_{IN}$  and determines the point along that trajectory where the average pulse width is equal to the minimum pulse width that is defined by the DRIVER\_MIN\_PULSE PMBus command. This calculation is shown in Equation 61. .

$$V_{START} = V_{IN} \cdot DRIVER\_MIN\_PULSE \cdot F_{SW} \tag{61}$$

Figure 5-5 illustrates the SS ramp sequence.

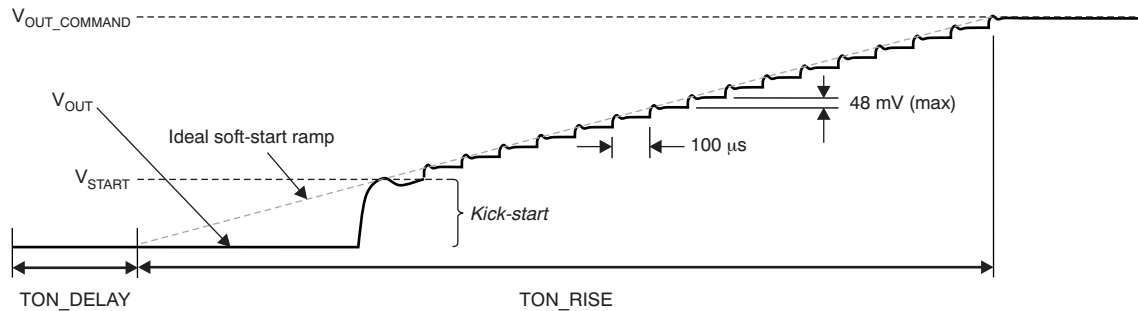


Figure 5-5. Soft-Start Ramp

When the controller determines the SS ramp must start, it sets the setpoint reference digital-to-analog converter (DAC) to  $V_{START}$ . The device waits for the period of time defined by TON\_DELAY; then it waits until the ideal SS ramp voltage has reached  $V_{START}$ . Only then does the PWM switching of the MOSFETS start.

At this point, the controller applies a constant pulse width of DRIVER\_MIN\_PULSE on the PWM pin. This portion of the SS ramp is called the *kick-start*. The PWM signal is held at this constant pulse width until the output voltage has increased to the point where it is within the dynamic range of the error ADC. (This range is  $\pm 64$  mV from the default analog front-end gain setting of 4x.) The controller then switches to closed-loop control, and steps the output voltage to the desired VOUT\_COMMAND voltage by stepping the internal setpoint reference DAC. The number of steps depends on the ramp rate defined by TON\_RISE and VOUT\_COMMAND. The setpoint reference DAC is updated at a rate of 10 kHz and has a resolution of 1.6 mV.

The PWM signal is held inactive until the start of the kick-start period (usually a low logic level) as well as the synchronous rectifier enable (SRE) signal. At the beginning of the kick-start period, the SRE signal goes high and the PWM output is enabled. To generate the constant pulse width signal on the PWM pin, the output of the compensation filter is clamped at the value defined by DRIVER\_MIN\_PULSE. At the end of the kick-start period, the clamps are released.

Releasing the clamps has two effects. First, it allows closed-loop control of the PWM signal based on the error between  $V_{OUT}$  and the  $V_{REF}$  DAC. Second, it precharges the integrator state in the compensation filter to the correct average duty cycle for the output voltage at this point in the SS ramp sequence.

### 5.3.1 Starting into a Prebias State

When there is a pre-existing voltage at  $V_{OUT}$  before the regulator starts, the soft-start algorithm must be modified. There are two cases that must be considered:

1. When the pre-existing voltage ( $V_{PREBIAS}$ ) is less than the kick-start voltage; and
2. When  $V_{PREBIAS}$  is greater than  $V_{KICKSTART}$

In the first case, the SS ramp operation proceeds as described in Section 5.3, except that the length of the kick-start period is less. This effect is shown in Figure 5-6.

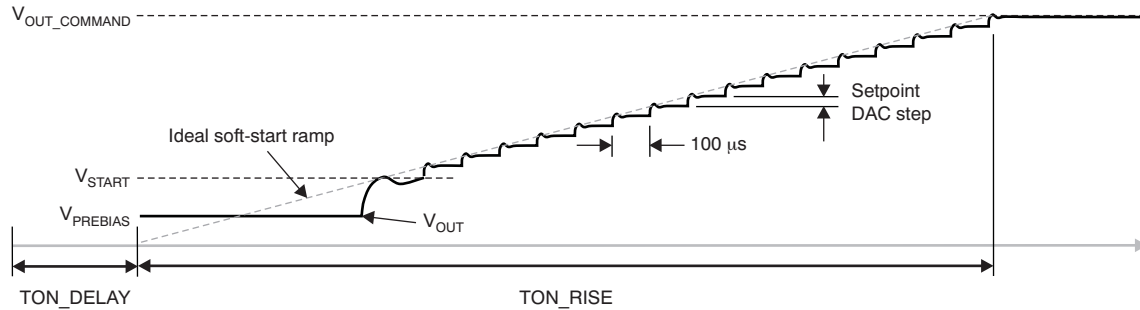


Figure 5-6. Soft-Start Ramp:  $V_{PREBIAS}$  Less Than  $V_{START}$

In the second case, the controller waits for an additional amount of time until the ideal ramp has reached the prebias voltage level. Then the PWM and SRE signals are enabled, and the SS ramp operation performed. In this case there is no kick-start period at all, and the controller moves immediately into closed-loop regulation. Because the controller had digitally measured the input and output voltages, the states of the compensating filter can be preset. This approach results in a smooth turn-on with very little disturbance when the prebias is above  $V_{START}$ . Figure 5-7 illustrates this case.

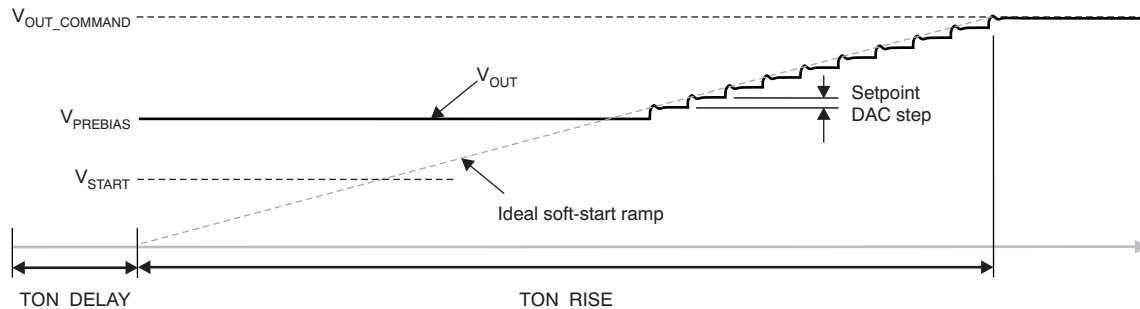


Figure 5-7. Soft-Start Ramp:  $V_{PREBIAS}$  Greater Than  $V_{START}$

In each case, the goal is to reach the final regulated voltage level at the end of the configured  $TON\_DELAY$  and  $TON\_RISE$  time.

### 5.3.2 Effect of a Large DRIVER\_MIN\_PULSE

When the DRIVER\_MIN\_PULSE value is large relative to the average duty cycle during normal voltage regulation, the kick-start period does not initiate until late in the SS ramp, as shown in Figure 5-8. This condition results in a large current draw during the kick-start period (exactly what the SS ramp is intended to avoid). Therefore, drivers with a relatively large minimum pulse width, such as the UCD7230, should not be used when the ratio of  $V_{OUT}/V_{IN}$  is small (less than 10%) or when the switching frequency is high (greater than 400 kHz), unless special precautions are taken to address the large inrush current

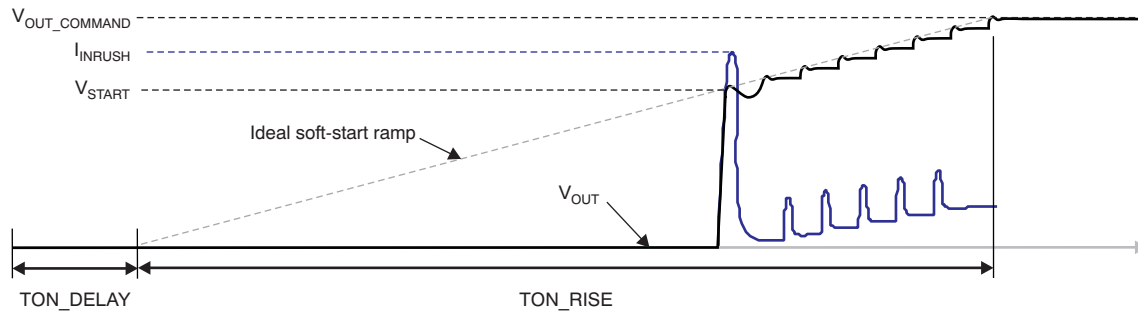


Figure 5-8. Soft-Start with Large DRIVER\_MIN\_PULSE

## 5.4 PWM Timing

By default, the UCD92xx controller spreads the phase on each PWM output to minimize the input current ripple and the output voltage ripple. This architecture is enabled by setting the phase between the first power stage on each voltage rails to be 3/13th of the fastest switching period. If, for example, the controller is configured to regulate four single phase voltage rails, each switching at 500 kHz, then Rail #2 is delayed 3/13th or 462 ns from Rail #1; Rail #3 is delayed 923 ns from Rail #1; and Rail #4 is delayed 1.385  $\mu$ s from Rail #1. A phase delay of 3/13th was selected so that if different switching frequencies are configured on different voltage rails, the rate at which the PWM pulses line up would be low.

When the controller is configured to drive multiple power stages to supply current to a regulated voltage rail, the phase of each power stage is set to evenly distribute the ripple current into the output capacitors. Thus, if Rail #2 is set to switch at 500 kHz and consists of three power stages, then phase 0 is delayed 3/13 of the switching period from the first phase of Rail #1; phase 1 is delayed 1/3 of the switching period from phase 0; and phase 2 is delayed 2/3 of the switching period from phase 0.

To accomplish the phase relationships outlined above, each PWM engine in the UCD92xx generates a delayed reset pulse that resets the ramp counter in the next higher PWM engine. This reset event happens at device power-up or at any time the device receives a command to change the assignment of PWM outputs to voltage rails or to change the switching frequency. Once the reset event is applied, the delayed reset pulse is disabled, and each counter is allowed to run from the master clock. In this way, each voltage rail can have a different frequency.

## 5.5 Inrush Current During Kick-Start

The UCD92xx device holds off switching the MOSFETs until the soft-start ramp voltage equals the steady state voltage that corresponds the specified minimum pulse width for the gate driver. The ramp voltage at which voltage conversion starts is given by Equation 62.

$$V_{START} = T_{MIN} F_{SW} V_{IN} \tag{62}$$

Where  $T_{MIN}$  is the minimum pulse width for the gate driver, which is specified by the PMBus command DRIVER\_MIN\_PULSE, and  $f_{SW}$  is the switching frequency.

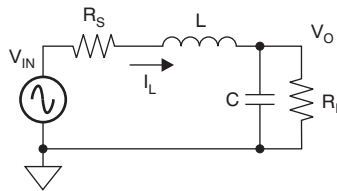
As noted earlier, when the internal soft-start ramp voltage reaches this value, the device begins the kick-start period. During the kick-start time, the controller drives the PWM signal with a fixed pulse width of  $T_{MIN}$ . The device drives the signal by setting the high and low clamp registers on the output of the



compensating filter to the duty cycle that corresponds to  $T_{MIN}$ . At the same time, the setpoint reference DAC is set to  $v_{START}$ . This PWM signal causes the output voltage to rapidly increase to  $v_{START}$ . When the output voltage has risen sufficiently so that the error ADC (that samples  $V_{REF} - V_{OUT}$ ) comes out of saturation, the upper and lower clamps on the compensated error are released and allow the voltage regulation loop to close.

During the kick-start period, the output capacitors must be charged up to  $v_{START}$ . This charging process requires current. Depending on the output filter components, it may take a significant amount of current. Therefore, it is important to know the peak inrush current during the kick-start period so that the OC thresholds can be set accordingly.

A simplified schematic of the power-supply output circuit is shown in [Figure 5-9](#). Here,  $R_s$  is the sum of the series resistances (average  $R_{DS(ON)}$ ) of the MOSFETs and the DCR of the inductor). The capacitance is the sum of all the capacitors on the output of the power supply. (Note that we are ignoring the equivalent series resistance, or ESR, of the capacitors.)  $R_L$  is the load and can be estimated as  $V_{OUT} / I_{OUT(max)}$ .



**Figure 5-9. Simplified Schematic for Estimating Inrush Current**

If we write the mesh equations for the voltage around each loop, or the nodal equations for the currents into each node, we can find an expression in  $s$  for the output voltage and the inductor current. [Equation 63](#) and [Equation 64](#) describe this process.

$$v_o(s) = \frac{v_{sw}}{LC} \frac{1}{s^2 + \left( \frac{R_s}{L} + \frac{1}{R_L C} \right) s + \left( \frac{R_s + R_L}{R_L} \right) \frac{1}{LC}} \quad (63)$$

$$I_L(s) = \frac{v_{sw}}{L} \frac{s + \frac{1}{R_L C}}{s^2 + \left( \frac{R_s}{L} + \frac{1}{R_L C} \right) s + \left( \frac{R_s + R_L}{R_L} \right) \frac{1}{LC}} \quad (64)$$

Where the average switch node voltage,  $v_{sw}$ , is calculated as [Equation 65](#).

$$v_{sw} = V_{IN} D = V_{IN} \frac{T_{PW}}{T_{SW}} = V_{IN} T_{PW} F_{SW} \quad (65)$$

From the frequency-domain expression of the inductor current, we can write the time-domain expression by performing a Laplace transform on [Equation 64](#). To do that, we first multiply by the Laplace equivalent of a unit step,  $1/s$ , and perform partial fraction expansion on this expression. This calculation is given in [Equation 66](#).

$$I_L(s) = \frac{v_{sw}}{R_s + R_L} \left[ \frac{1}{s} - \frac{s - \frac{R_L}{L} + \frac{1}{R_L C}}{s \left( s^2 + \left( \frac{R_s}{L} + \frac{1}{R_L C} \right) s + \left( \frac{R_s + R_L}{R_L} \right) \frac{1}{LC} \right)} \right] \quad (66)$$

Which also has the form shown in [Equation 67](#):

$$I_L(s) = \frac{V_{SW}}{R_S + R_L} \left[ \frac{1}{s} - \frac{s + \lambda}{(s + \alpha)^2 + \omega^2} \right] \quad (67)$$

Where the terms are defined by [Equation 68](#).

$$\alpha = \frac{R_S}{2L} + \frac{1}{2R_L C}$$

$$\omega = \left[ \left( \frac{R_S + R_L}{R_L} \right) \frac{1}{LC} - \alpha^2 \right]^{1/2}$$

$$\lambda = \frac{1}{R_L C} - \frac{R_L}{L} \quad (68)$$

The Laplace transform is then given by [Equation 69](#).

$$I_L(t) = \frac{V_{SW}}{R_S + R_L} \left[ 1 - e^{-\alpha t} \left( \cos(\omega t) + \left[ \frac{\lambda - \alpha}{\omega} \right] \sin(\omega t) \right) \right] \quad (69)$$

[Equation 69](#) has a maximum at the time point where the derivative with respect to time is zero, which is calculated with [Equation 70](#).

$$\frac{dI_L(t)}{dt} = 0 \quad (70)$$

This point occurs at the value calculated by [Equation 71](#):

$$t_{MAX} = \tan^{-1} \left( \frac{\omega(\lambda - 2\alpha)}{\omega^2 + \alpha\lambda - \alpha^2} \right) \frac{1}{\omega} \quad (71)$$

Therefore, we can find the maximum inrush current by calculating  $\alpha$ ,  $\omega$ , and  $\lambda$ . Then, we determine  $t_{MAX}$  and substitute that value into [Equation 69](#).

[Figure 5-10](#) shows an example circuit where the parameters in [Table 5-2](#) are established. For this circuit, the maximum inrush current is 3.8 A.

**Table 5-2. Example Circuit Component Values**

Component	Value
$R_S$	8 m $\Omega$
L	2.2 $\mu$ H
C	611 mF
$R_L$	180 m $\Omega$
$V_{SW}$	12 V • 40 ns • 500 kHz = 240 mV

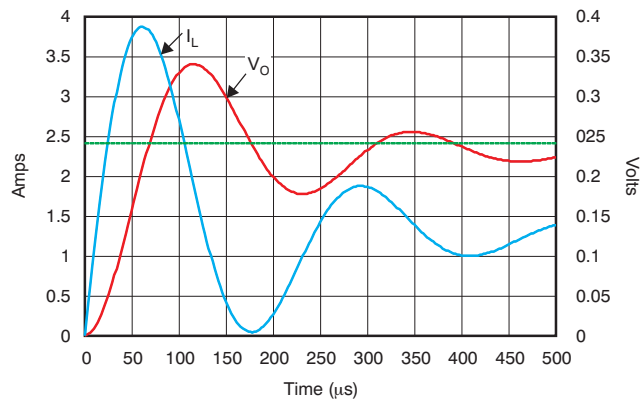


Figure 5-10. Voltage and Current during the *Kick-Start* Period

### 5.5.1 Inductor Ripple Current

The equations discussed in [Section 5.5](#) ignore the switching action of the power supply. Superimposed on the waveform in [Figure 5-10](#) is the ripple current in the inductor because of the ground-to- $V_{IN}$  swings of the switch node. The magnitude of the ripple current is a function of the switching frequency, pulse width, inductance, and input voltage. It is independent of the load current.

The peak ripple current is the integral of the voltage across the inductor during the on-time pulse. The voltage across the inductor at this time is the input voltage minus the output voltage. At the top of the inrush current waveform (from [Equation 69](#)), the output voltage is approximately 1/2 of the steady-state value for the kick-start pulse width. The base-to-peak ripple current at the top of the inrush current curve is then calculated by [Equation 72](#).

$$I_{\text{RIPPLE}} = \frac{1}{2L} \int_0^{T_{\text{PW}}} v_L dt = \frac{(V_{\text{IN}} - V_{\text{O}})}{2L} T_{\text{PW}} = \frac{V_{\text{IN}}}{4L} (2T_{\text{PW}} - T_{\text{PW}}^2 F_{\text{SW}}) \quad (72)$$

This value should be added to the peak small-signal current calculated using [Equation 69](#).

For this example, the ripple current is ~55 mA. However, for a high current power stage with  $L = 250 \text{ nH}$ ,  $V_{\text{IN}} = 12 \text{ V}$ ,  $f_{\text{SW}} = 350 \text{ kHz}$ , and  $\text{DRIVER\_MIN\_PULSE} = 140 \text{ ns}$ , the ripple current amplitude is substantial: 1.6 A (base-to-peak).

## 5.6 Current Balance

When multiple power stages supply power to an output voltage rail, we want the power dissipated in each of the power stages to be balanced. This approach maximizes the efficiency of the voltage converter system. The UCD92xx controller provides two methods of balancing the power dissipation:

- First, the controller monitors the average current in each inductor or MOSFET, adjusts down the commanded pulse width on the power stage with the largest current, and adjusts up the commanded pulse width on the power stage with the lowest current.
- In the second method, the controller monitors both the current and the temperature of each power stage. Here, each PWM output is adjusted to balance the current (as in method 1), but the PWM output is additionally adjusted down on the power stage with the highest temperature and adjusted up on the power stage with the lowest temperature.

The current balance function only becomes active after the soft-start ramp completes and after the criteria for PowerGood have been satisfied. Then, the controller begins to monitor the currents and temperatures, and periodically calculates the proper adjustment to the PWM outputs to balance the power dissipation in the power stages.

The pulse width,  $T_{PW}$ , on each PWM output pin is determined by calculating the appropriate duty cycle based on the compensated error voltage, which is multiplied by the switching period. A current balance adjustment is then added to this product for each PWM output that is driving a phase of the output voltage rail, as [Equation 73](#) explains.

$$T_{PW[Phase]} = \left[ \text{round}(\text{duty} \cdot \text{PRD} \cdot 2^{-11}) + D_{ADJ[Phase]} \right] T_{CLK} \quad (73)$$

Here, *duty* is a signed 16-bit value such that  $2^{15} = 100\%$  on-time,  $T_{CLK} = 250$  ps, and PRD is the period register contents where  $F_{CLK} = 1/T_{CLK}$ , as shown in [Equation 74](#).

$$\text{PRD} = \text{round} \left( \frac{F_{CLK}}{F_{SW} \cdot 2^4} \right) \quad (74)$$

[Table 5-3](#) summarizes the current balance update rates.

**Table 5-3. Current Balance Update Rates**

Function	interval
Raw current measurement	200 $\mu$ s
Raw temperature measurement	8 ms <sup>(1)</sup>
Calculate current balance	2 ms
Calculate temperature balance	100 ms

<sup>(1)</sup> UCD9248 = 8 ms, UCD9246 = 6 ms, and UCD9224 = 4 ms.

Every 2.0 ms (with a 500-Hz update rate), the controller reviews the most recently stored currents for each power stage and determines the largest and smallest currents for each voltage rail. The current adjustment is calculated by first finding the difference between the largest and smallest current, as shown by [Equation 75](#).

$$I_{DIFF} = I_{HIGH} - I_{LOW} \quad (75)$$

A pulse width correction factor,  $I_{ADJ}$ , is then calculated using [Equation 76](#).

$$I_{ADJ} = 0 \leq \text{trunc} \left( \frac{I_{DIFF}}{125 \text{ mA}} \right) \leq 5 \quad (76)$$

In other words,  $I_{ADJ}$  has a value between 0 and 5. Then the duty cycle for the power stage with the lowest current is updated as shown in [Equation 77](#).

$$D_{ADJ[Lowest]} = D_{ADJ[Lowest]} + I_{ADJ} \quad (77)$$

Similarly, the duty cycle for the power stage with the highest current is updated as shown in [Equation 78](#).

$$D_{ADJ[Highest]} = D_{ADJ[Highest]} - I_{ADJ} \quad (78)$$

The average voltage at the switch node is then given by [Equation 79](#).

$$V_{SW} = \frac{\text{round}(\text{duty} \cdot \text{PRD} \cdot 2^{-11}) + D_{ADJ[Phase]}}{(\text{PRD} - 1)2^4} V_{IN} \quad (79)$$

Therefore the approximate change to the average switch node voltage can be found with [Equation 80](#).

$$\Delta V_{SW} = \frac{I_{ADJ} V_{IN}}{(\text{PRD})2^4} = \frac{I_{ADJ} F_{SW} V_{IN}}{F_{CLK}} \quad (80)$$

Now, how much change in actual current does this equation represent? We can determine the change in current by noting that the output voltage is a function of the duty cycle, as described by Equation 81.

$$V_{OUT} = V_{SW} - I_O R_S \tag{81}$$

where  $I_O$  is the dc output current/phase, and  $R_S$  is the series resistance through the MOSFETS and inductor. The change in output current is then calculated by Equation 82:

$$\Delta I_O = \frac{V_{SW}}{R_S} = \frac{I_{ADJ} F_{SW} V_{IN}}{F_{CLK} R_S} \tag{82}$$

For a typical system with  $V_{IN} = 12\text{ V}$ ,  $R_S = 5\text{ m}\Omega$ , and  $f_{SW} = 500\text{ kHz}$ , we calculate results with Equation 83.

$$\begin{aligned} \Delta I_{MIN} &= 300\text{ mA} \quad \text{and} \\ \Delta I_{MAX} &= 1.5\text{ A} \end{aligned} \tag{83}$$

In review, the controller samples the monitored current every 2.0 ms, and determines which phase supplies the largest amount of current and which phase supplies the minimum amount of current. The device then makes the pulse width of the PWM signal that supplies the smallest amount of current greater, usually by the minimum digital increment of 0.25 ns, and reduces the pulse width of the PWM signal that supplies the most current by the same amount.

By constantly adding to one phase and subtracting the same amount from another phase, the net charge on the output capacitors is maintained and the voltage regulation action is unaffected.

Figure 5-11 shows the measured current balance action for a power system that consists of a dual phase voltage rail supplied by 12 V, outputting 1.0 V at 5.5 A through a series resistance of ~8 m $\Omega$ .<sup>(2)</sup> The switching frequency in this example is 350 kHz. In this case, the expected resolution of the current balance from Equation 82 is 130 mA; as a result, we should expect to see peak-peak updates to the current of  $2 \times [0\ 1\ 2\ 3\ 4\ 5]$  times this value, or [0 0.263 0.525 0.788 1.050 1.313] A. This estimate is confirmed in Figure 5-11, where we see peak-to-peak updates in the current of approximately [0, 0.29, 0.53, 0.77, and 1.02] A.

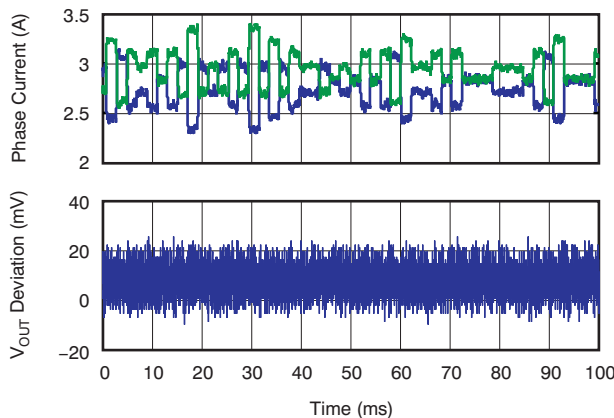


Figure 5-11. Example Current Balance Action for a Dual-Phase Voltage Rail

<sup>(2)</sup> d3 m $\Omega$  in the FETs, 1 m $\Omega$  in the inductor, and 4 m $\Omega$  in the copper board traces

## 5.7 Temperature Balance

At every 100 ms, as shown in [Table 5-3](#), the controller compares all of the monitored temperatures to find the power stage at the highest temperature and the power stage at the lowest temperature for each page (voltage rail). If the difference between these temperatures is greater than a predetermined threshold,  $D_{ADJ}$  is incremented for the minimum temperature power state and decremented for the maximum temperature power stage. This calculation is then summed with the  $D_{ADJ}$  duty cycle correction determined based on the measured current imbalance.

It is necessary to run both the current balance loop and the temperature balance loop because the current balance operating point changes with the load current. For example, consider a dual-phase system with a temperature difference between phases of 4 degrees with a 25-A load. At device turn-on, the temperature rises about 20 degrees over 10 seconds; it then takes about 30 seconds for the temperature balancing function to remove the imbalance. At that point, the currents continue to run with an imbalance between them to maintain even temperatures. The system continues to respond to rapid imbalance transients in the current.

A minimum threshold is necessary to prevent the temperature balance loop from winding up when there is a light load. This threshold is set by the TEMP\_BALANCE\_IMIN command. Without this threshold, under a very light load, it is possible that no amount of *current* balancing affects the temperature sufficiently to compensate for any ambient variations or calibration errors. In such a case, if an insufficient dead-band is specified, the algorithm slowly ramps the correction to its limit. When a load is subsequently enabled, the applied current is directed exclusively to the phase that had been reporting a slightly cooler temperature. This situation may persist for many seconds, during which time damage could be incurred by the overdriven phase. Adding a threshold also provides an easy method to disable the feature; by setting the limit to be very high, the pulse width is never adjusted as a result of temperature.

## Using the Fusion Designer Software

---

---

The Fusion Digital Power Designer software offers several tools to assist in creating a design and then saving that design to a device and a project file. Three of those features are described here.

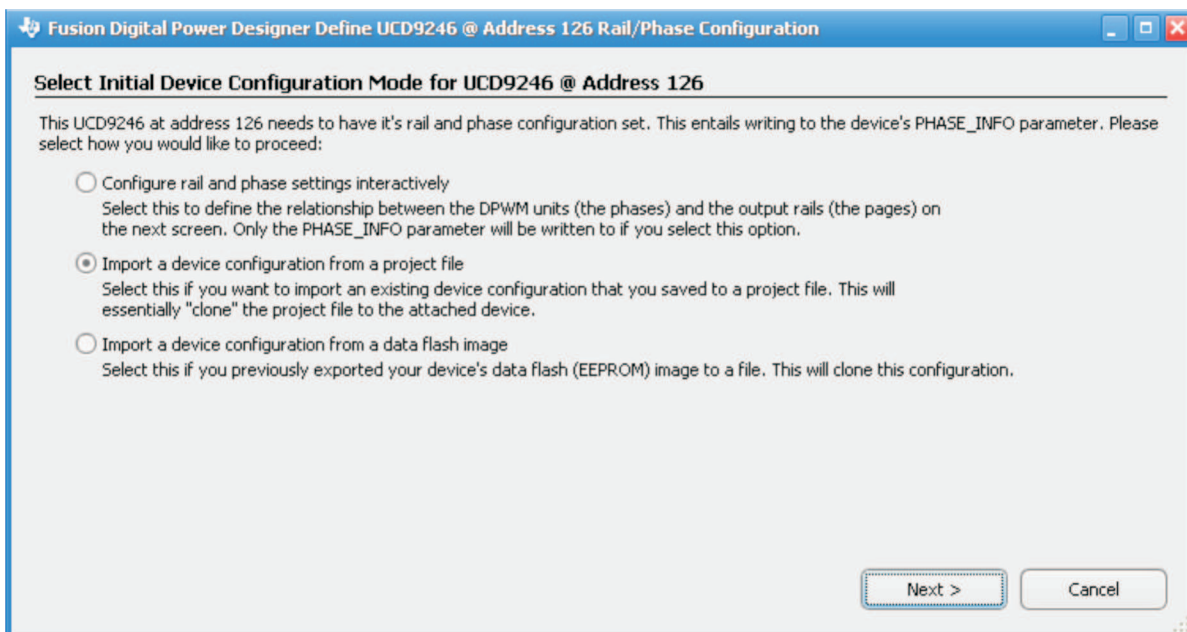
Topic	Page
6.1 Downloading the Project File .....	88
6.2 Copying One Design to Another Voltage Rail .....	90
6.3 Design®Device Synchronization .....	92

## 6.1 Downloading the Project File

Once you have a completed project file that produces the desired circuit behavior for your application, you must download this configuration into every board as part of the manufacturing process. The simplest way to accomplish this step (and the method envisioned by the PMBus standard) is to add a PC-based tester to the board manufacturing assembly process. Connect the tester to a PMBus header on the board and download the configuration in the project file. Texas Instruments offers three options for this process:

1. Use the Texas Instruments' Fusion Digital Power Designer software program (the *Fusion GUI*).
2. Use the Texas Instruments' Fusion Digital Power Manufacturing Tool.
3. Create your own tester using the Fusion Digital Power Designer API.

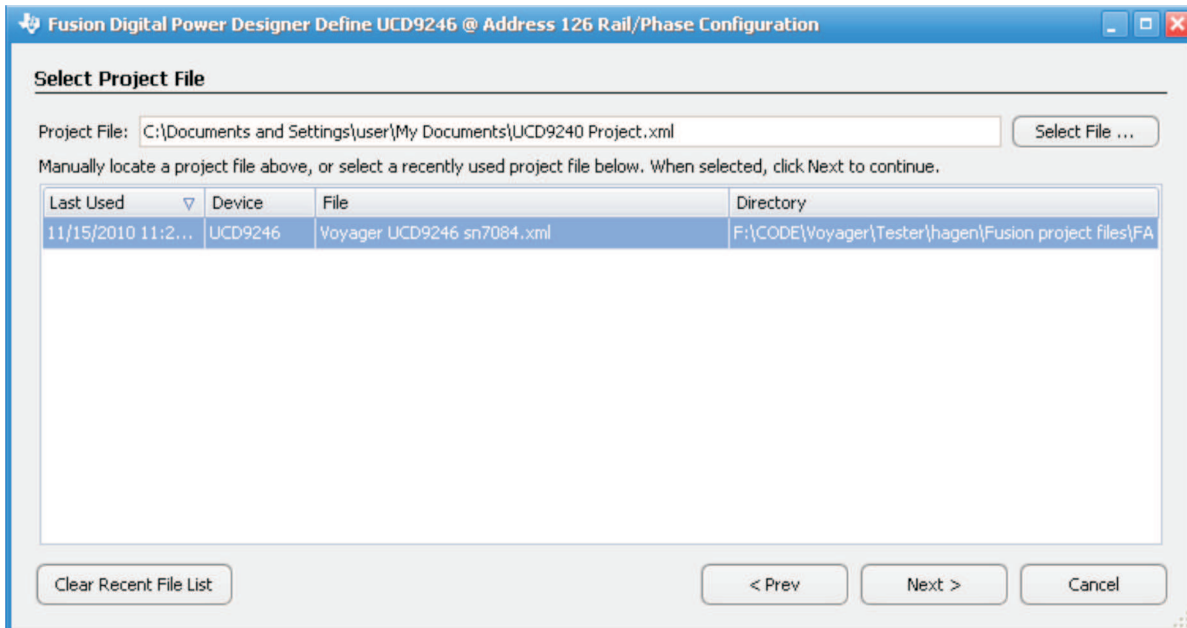
The first time you download the project file, you should use the Fusion GUI. Connect a USB-to-GPIO dongle to the PMBus header on your PCB; apply power; then start the Fusion GUI software. If the controller configuration memory has been cleared (or is new from TI), the Fusion GUI prompts you for a project file, as [Figure 6-1](#) shows.



**Figure 6-1. Fusion GUI: Project File Download Menu**

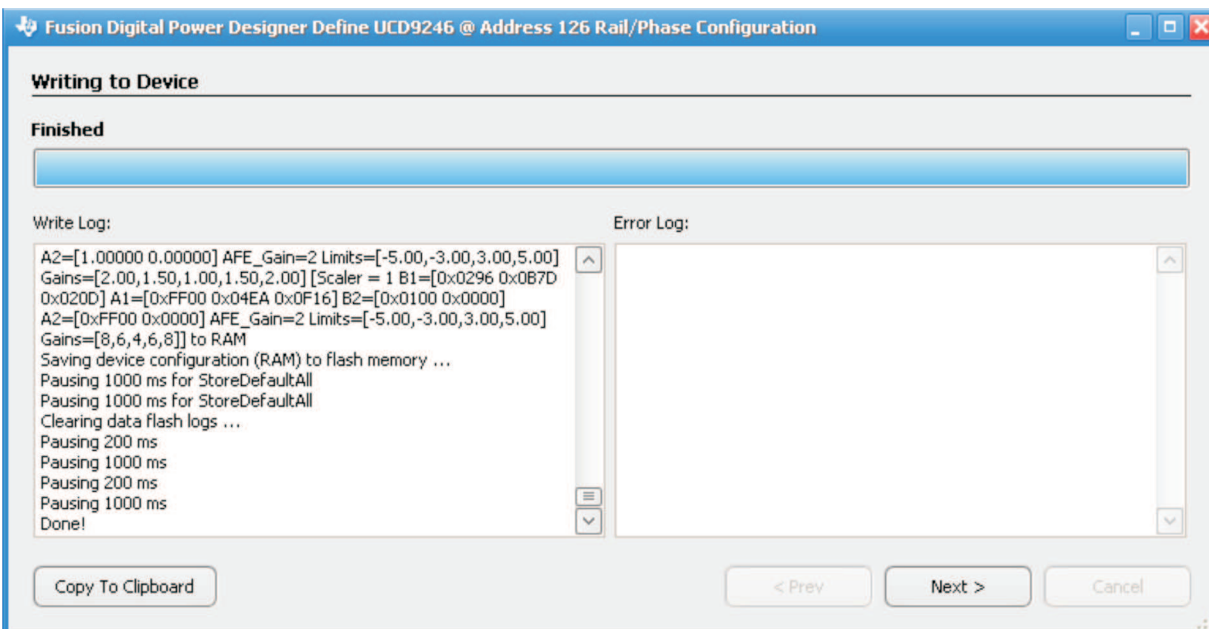


Enter the location of the project file, as [Figure 6-2](#) illustrates.



**Figure 6-2. Fusion GUI: Project File Selection**

The Fusion GUI then loads the selected configuration. [Figure 6-3](#) shows this sequence.



**Figure 6-3. Fusion GUI: Project File Configuration**

The next time that power is applied to the board, the controller starts regulating each rail based on the sequence and voltage settings stored in the project file that has been loaded to the device.

### 6.1.1 Communicating to a Previously Configured UCD92xx Device

Keep in mind that the Fusion GUI does not load a project file in online mode. During the startup sequence, the software loads its own *mini-project* for each unique device/address pair that it has worked with before (such as a UCD9240 device at address 126). This mini-project data—design tool data such as the circuit design, compensation settings, and sequencing simulation information—are stored in a user preferences database. This database is in the file *C:\Program Files\Texas Instruments Fusion Digital Power Designer\data\prefs.xml*.

Consider this example:

- Connect to the device in online mode, such as a UCD9240.
- Jump to the design task; no design is loaded.
- Import a project file. This process loads design data from the project file into the design tool. You now see the power stage circuit and compensation inputs.
- Tweak the design as desired.
- Write the changes to the device memory.
- Quit the GUI program.

*Your original project file is not updated in this case.* because you did not explicitly perform a *File/Save* operation to save updated the project file. However, the Fusion GUI saved the new design data to the global preferences file, and linked it to the device that was connected (a UCD9240 at address 126, in this example).

If you start the GUI again later, the software loads the device configuration information from the active device and uses the cached design for the part/address it saved earlier. This design is not in a specific project file, but in the specific user preferences.

In order to ensure that your project file can be exported to different PCs or testers, or that you can work with the file offline, you must save your changes in online mode. This step is not necessary each time you quit the GUI program; however, if you have made changes to the design tool and wish to keep them, it is recommended that you save these changes to a project file for future use.

## 6.2 Copying One Design to Another Voltage Rail

There are two ways to copy a design from one Bank/Rail to another Bank/Rail. The first method is by going to the *CLA Coefficient Banks* tab of the Configure page of the Fusion GUI. This method only copies the compensation filter coefficients. The second method is found in the drop-down menu on the Design page called *Copy Rail and/or Design*.

### 6.2.1 Method 1: Copy Coefficients

Figure 6-4 shows the CLA Coefficients Banks tab of the Fusion GUI.

Bank / Name	Floating Point								Fixed Point								Non-Linear Limits				Non-Linear Gains				
	B01	B11	B21	A11	A21	B12	A12	AFE	Scalar	B01	B11	B21	A11	A21	B12	A12	L0	L1	L2	L3	G0	G1	G2	G3	G4
<b>Hardware Banks</b>																									
Bank 0xFE - Active	0.000	0.000	0.000	-1.000	0.000	0.000	0.000	1x	0x0	0x0000	0x0000	0x0001	0x0800	0x0000	0x0000	0x0000	-5	-3	3	5	1.00	1.00	1.00	1.00	1.00
Bank 0xFF - Inactive	0.000	0.000	0.000	-1.000	0.000	0.000	0.000	1x	0x0	0x0000	0x0000	0x0001	0x0800	0x0000	0x0000	-5	-3	3	5	1.00	1.00	1.00	1.00	1.00	
<b>RAM Banks</b>																									
Bank 0x0 - Soft Start/Stop	17.375	-33.750	16.406	-1.000	0.000	0.000	0.000	2x	0x6	0x022C	0x08C8	0x020D	0x0020	0x0000	0x0000	-5	-4	3	4	1.00	1.00	1.00	1.00	1.00	
Bank 0x1 - Regulation	6.906	-13.742	6.852	-1.219	0.219	0.000	0.000	4x	0x4	0x0374	0x0921	0x036D	0x009C	0x0FE4	0x0000	-5	-4	3	4	2.00	1.75	1.00	1.75	2.00	
Bank 0x2 - Light Load	0.000	0.000	0.000	-1.000	0.000	0.000	0.000	1x	0x0	0x0000	0x0000	0x0001	0x0800	0x0000	0x0000	-5	-3	3	5	1.00	1.00	1.00	1.00	1.00	
<b>Saved Coefficients</b>																									
Rail #1 Bank 0x1 - Regulation	6.906	-13.742	6.852	-1.219	0.219	0.000	0.000	4x	0x4	0x0374	0x0921	0x036D	0x009C	0x0FE4	0x0000	-5	-4	3	4	2.00	1.75	1.00	1.75	2.00	

Linear Gains		Non-Linear Gains		Actions & Notes	
Floating Point	Fixed Point	eADC Threshold	Gain		
B01: 6.906	B01: 0x0374	LIM3: 4	2.00 0x08	Save Changes Activate Copy to Other Bank Duplicate View as Text Load in Design Tool	
B11: -13.742	B11: 0x0921	LIM2: 3	1.75 0x07	Delete	
B21: 6.852	B21: 0x036D	LIM1: -4	1.00 0x04	Summary: Rail #1 Bank 0x1 - Regulation	
A11: -1.219	A11: 0x009C	LIM0: -5	1.75 0x07	Notes:	
A21: 0.219	A21: 0x0FE4		2.00 0x08		
B12: 0.000	B12: 0x0000				
A12: 0.000	A12: 0x0000				
AFE Gain: 4x	Scaler: 0x0004				

Figure 6-4. Fusion GUI: CLA Coefficients Banks Tab

First, go to the design page. Either allow autotune to determine the compensation coefficients, or manually determine the desired values for one of the coefficient banks. Then go to the CLA Coefficients tab on the Configuration page and select the bank that contains the desired coefficients. Under **Actions**, select *Save to Favorites*. This selection adds a line under Saved Coefficients. Select this line. Now you can copy this set of coefficients to other banks by clicking the **Copy to other Bank** button.

To copy the saved coefficients to other rails, change to the new rail in the drop-down list at the upper corner of the main GUI window.

### 6.2.2 Method 2: Copy a Design

The Copy Rail and/or Design tool copies the actual coefficients that are downloaded into the device. It can also copy the power stage circuit design. In most cases, this method is the preferred means of transferring a design across multiple voltage rails and multiple controller devices. Figure 6-5 illustrates this screen menu.

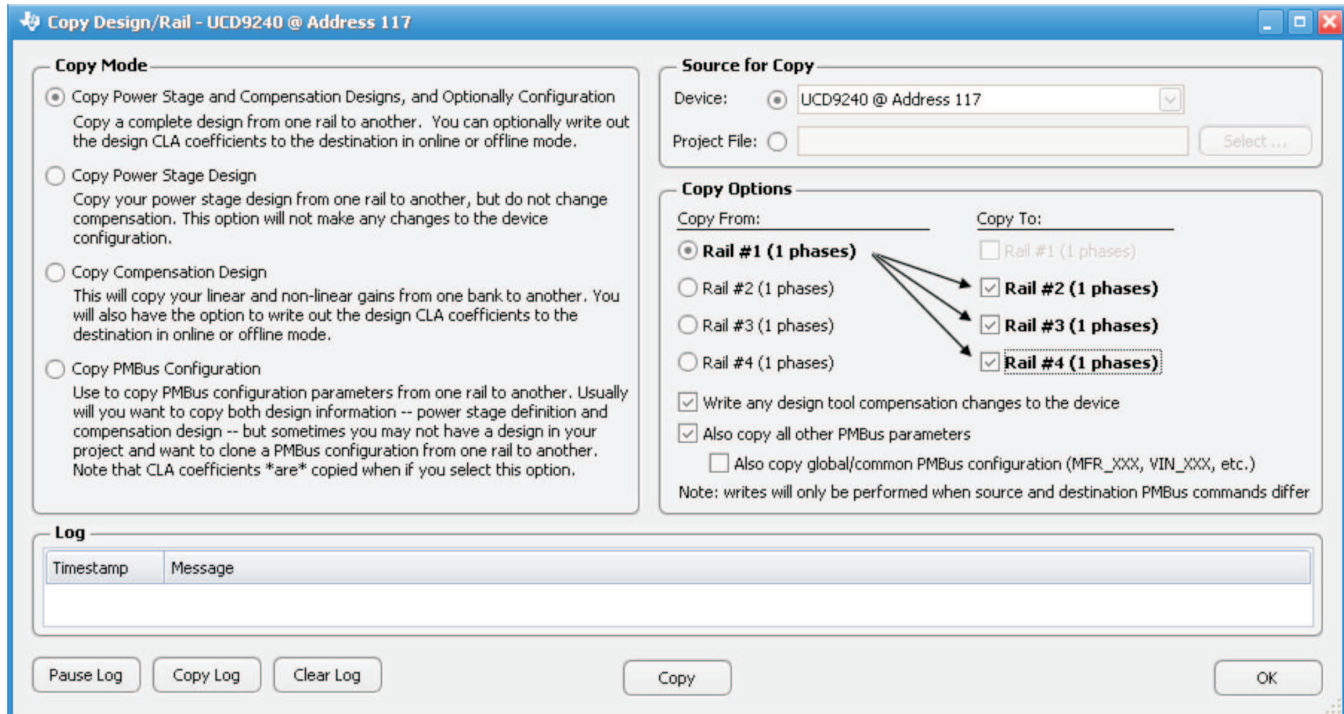


Figure 6-5. Fusion GUI: Copy Design Window

The third option, *Copy Compensation Design*, can be useful once you have a set of compensation filter coefficients that are well-defined in Bank-1. Use this option to copy the coefficients to Bank-0 and Bank-2; then go the design page and make any fine-tune adjustments to the copied Bank. Once this voltage rail is completely configured, you can use it as a source to copy to other voltage rails.

The first copy mode option copies the power stage schematic and the PMBus configuration from one rail to another. Use this option to replicate the first complete voltage rail that you have configured to the other voltage rails in a system. To copy a design from one device to another device in offline mode, select the desired project file as the source.

### 6.3 Design@Device Synchronization

The Design@Device Synchronization window on the Design page determines which PMBus command values to write, based on the information entered here by the user. Normally, for a new design a user should check all the boxes and enter the information about the power stage circuits, so that the GUI does the work of calculating values for commands such as VOUT\_SCALE\_LOOP, IOUT\_CAL\_GAIN and CLA\_BANK.

Once you have a complete design and that design has been saved to a project file and/or an actual device, you may want to fine-tune some of the configuration parameters. For instance, the Fusion GUI calculates the current-to-voltage gain and offset for monitoring current based on the values selected in the power stage schematic. Once the current sense is measured on actual hardware, however, users may find that there is additional resistance in the board wiring that makes the inductor DCR (over which current

is measured in some of the power stage circuits) larger than the data sheet value for the inductor. In this case, uncheck the current sense checkbox and modify the PMBus values for IOUT\_CAL\_GAIN and IOUT\_CAL\_OFFSET on the Iout tab of the Configuration page. (You can also modify these values directly on the *Advanced Configuration* tab of the Configuration page.) Then the calibrated current sense gain and offset are not updated in the project file on subsequent changes to the power stage schematic.

Note that when you edit and write out a change for a PMBus command on the Configuration page that can also be managed on the Design page, the Fusion GUI automatically turns off synchronization for that category of commands; it also sends the user a notification. These synchronization settings are saved in the specific project file (offline) or user preferences (online).

The following list shows the commands that are set (or not set) for each checkbox. Note that writes are only performed when the device value differs from the design tool computed value.

- **Vin Configuration.** Sets PMBus commands relating to input voltage limits. When checked, the GUI writes the following commands when you click the **Write to Hardware** button:
  - VIN\_OV\_FAULT\_LIMIT
  - VIN\_OV\_WARN\_LIMIT
  - VIN\_UV\_WARN\_LIMIT
  - VIN\_UV\_FAULT\_LIMIT
- **Vout Timing.** Commands related to the voltage setpoint of the device that are not set in the Vout Config tab. When checked, the GUI writes the following commands when you click the **Write to Hardware** button:
  - VOUT\_TRANSITION\_RATE
  - TON\_RISE
  - TON\_DELAY
  - TOFF\_FALL
  - TON\_DELAY
  - TOFF\_MAX\_WARN\_LIMIT
  - TON\_MAX\_FAULT\_LIMIT
- **Driver Configuration.** The driver used in the active design, either selected directly or used in the power train module selected by the user. When checked, the GUI writes the following commands when you click the **Write to Hardware** button:
  - DRIVER\_CONFIG
  - DRIVER\_MIN\_PULSE
  - PREBIAS\_GAIN
  - PREBIAS\_OFFSET
  - MAX\_DUTY
  - EADC\_SAMPLE\_TRIGGER
  - EADC\_TRIM

The Fusion GUI sets the  $V_{IN}$ ,  $V_{OUT}$ ,  $I_{OUT}$ , and Temperature-related configuration commands based on the information entered into the design window and schematic. You can perform more precise calibration of these commands using the built-in GUI calibration tool (Tools menu, *Calibrate Device*) or the Manufacturing GUI calibration module. The built-in GUI calibration tool automatically unchecks the related box after the input voltage calibration is run. If you plan on performing the calibration manually or through the Manufacturing GUI, you should uncheck this box yourself.

- **Vin Scale.** Sets the scaling on the monitored input voltage based on the resistors specified in the schematic. When checked, the GUI updates the following command:
  - VIN\_SCALE
- **Vout Scale Monitor/Loop.** Sets the scaling on both the output voltage sense used for voltage regulation and the scaling used for monitoring the regulated voltage. When checked, the GUI updates the following commands based on the resistors specified in the schematic.
  - VOUT\_SCALE\_MONITOR
  - VOUT\_SCALE\_LOOP

Note that the design tool does not write to the following commands, which are also adjusted when performing full  $V_{OUT}$  calibration.

VOUT\_CAL\_OFFSET  
VOUT\_CAL\_MONITOR

- **Current Sense.** Sets both the scaling and offset for the current sense circuit defined in the schematic. In the case of Texas Instruments power modules and select drivers, this network is already defined and the design tool does not need any additional information. When checked, the GUI updates the following commands:
  - IOUT\_CAL\_GAIN
  - IOUT\_CAL\_OFFSET
  - THERMAL\_COEFF
- **Temp Gain/Offset.** Sets the gain and offset for external temperature sensors that are multiplexed onto the TEMP analog input pin. When checked, the GUI updates the following commands:
  - TEMPERATURE\_CAL\_GAIN
  - TEMPERATURE\_CAL\_OFFSET

## PMBus Communication

### A.1 Overview

The UCD92xx PWM controller implements approximately 120 commands as defined in the [PMBus Standard](#). Those commands that are unique or differ in some way from the standard are described in the [UCD92xx Command Reference](#) document. The PMBus Standard describes how the device should behave during start-up and shutdown, and how to handle faults.

The way that a power supply (controlled by a PMBus controller) starts is defined by the ON\_OFF\_COMMAND. There are three events that can be defined to initiate power conversion. Depending on how the bits are set in the command, each event can start conversion or a combination of events are required. The bits are defined in Table 10 of the standard.

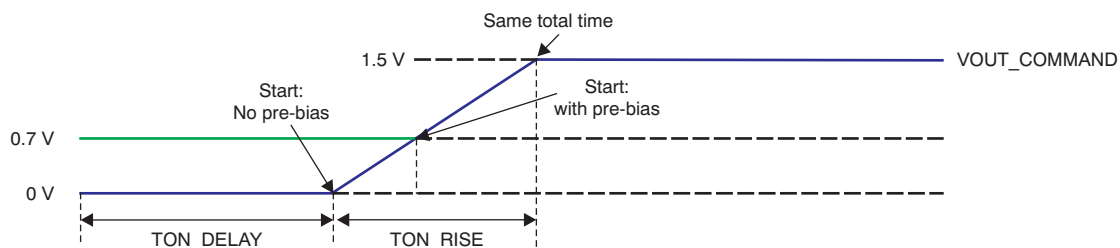
- Turn on with  $V_{IN}$ . The threshold for  $V_{IN}$  is set by the VIN\_ON and VIN\_OFF commands.
- Turn on with the CTRL line. (Polarity is set in the ON\_OFF\_CONFIG command;  $V_{IN} > VIN\_ON$ .)
- Turn on with the OPERATION command. (Optionally gated with the CTRL pin;  $V_{IN} > VIN\_ON$ .) The OPERATION command is also used to set margin high or margin low.

Once the conditions to start conversion have been satisfied, the PMBus standard defines additional commands that control the relative timing of the start-up of each controlled voltage rail. The two most basic commands are:

- TON\_DELAY
- TON\_RISE

TON\_DELAY sets a time (in ms) that the controller waits before starting the soft-start ramp after the start conditions defined in ON\_OFF\_COMMAND have been met. TON\_RISE sets a time (in ms) for the voltage to ramp to the voltage specified by the VOUT\_COMMAND command. Note that the UCD92xx devices adjust the timing and slope of the soft-start ramp such that the regulated voltage reaches the VOUT\_COMMAND value at the time of TON\_DELAY+ TON\_RISE, regardless of whether the output voltage started at zero or some non-zero value. A non-zero starting voltage is termed a *pre-bias* voltage.

Figure A-1 illustrates the timing operation for TON\_DELAY and TON-RISE.



**Figure A-1. TON\_DELAY and TON\_RISE**

## A.2 LINEAR11 Format

Many of the commands defined in the PMBus Standard (particularly the ones related to setting the output voltage) have values that are specified using LINEAR11 format. The standard defines this format as a 16-bit value with an 11-bit mantissa and a 5-bit exponent. The mantissa has 10 significant bits plus one sign bit, allowing a base range from  $-1024$  to  $+1023$ . The exponent has four significant bits plus one sign bit, allowing exponent values from  $-16$  to  $+15$ . When combined, the range of expressible values goes from  $2^{-16}$  (0.000015) to  $1023 \cdot 2^{15}$  (33,521,664). Section 7.1 of the PMBus Standard covers this translation.

Note that not all values can be expressed by this numerical format. Some values, such as 0.5, can be expressed exactly in LINEAR11 format. Other values can only be approximated. For example, 0.4900 would be expressed as  $502 \ll (-10)$ , which is 0.4902. In addition, the underlying hardware DAC or ADC can have lower resolution than the PMBus command variable. The UCD92xx device uses the underlying resolution of the actual hardware when calculating the value to return when reading the value of a command. This architecture provides the host with the ability to get an accurate observation of the command value. Section 7.4 of the PMBus standard describes this condition, and explicitly states that such behavior complies with the standard.

The PMBus command variables (such as `VOUT_TRANSITION_RATE` or `VOUT_COMMAND`) are translated into internal variables to match the scaling of the hardware peripherals such as DACs and ADCs. In many cases, the hardware peripheral has less resolution than the PMBus command that controls it. In such cases, the low-order bits could be truncated during the read-back operation to match the hardware resolution.

As an example, consider the command `VOUT_TRANSITION_RATE`.

- `VOUT_TRANSITION_RATE` is expressed in  $\text{mV}/\mu\text{s}$  (which is equivalent to  $\text{V}/\text{ms}$ ).
- The signal at the EAP/EAN pins is attenuated by an external feedback divider network, so the resolution at the voltage output terminals is even larger. The `VOUT_SCALE_LOOP` command is used to inform the controller about the value of the feedback divider.
- The DAC is updated once every 0.1 ms.

Combining these terms together, the size of each step is given by [Equation 84](#).

$$V_{\text{Step}} = (\text{VOUT\_SCALE\_LOOP}) \left[ \text{VOUT\_TRANSITION\_RATE} \frac{\text{mV}}{\mu\text{s}} \right] \left[ \frac{100 \mu\text{s}}{\text{Step}} \right] \quad (84)$$

Several quantization terms must be included in this calculation.

- The setpoint inside the UCD92xx is controlled by a 10-bit ADC with a range of 1.6 V. The resolution of the DAC hardware is thus 1.5625 mV, relative to the EAP/EAN feedback pins.
- To improve the resolution of the ramp rate, the internal variable that controls the DAC uses 14 bits of precision. When slewing, the 14-bit variable is incremented by the appropriate step amount, and the resulting value is then quantized down to 10 bits before writing to the DAC.
- The internal variables used to store the `VOUT_TRANSITION_RATE` and `VOUT_SCALE_LOOP` variables have limited precision.
- Each of the intermediate terms in the calculation has limited precision.

The DAC transition calculations also include two additional limits:

- To prevent the ramp from stalling when commanded to slew very slowly, the internal variable has a minimum value of one count. At its slowest rate, the DAC moves by 1.5625 mV every 1.6 ms, or approximately 0.001  $\text{V}/\text{ms}$  at the EAP/EAN pins.
- To prevent the EADC from saturating while slewing very quickly, the transition rate has an upper limit. This limit depends on the AFE gain setting. The upper limit is  $192 \text{ mV} / \text{AFE\_Gain} / 0.1 \text{ ms/step}$ . For the most common AFE gain setting, 4x, the upper slew limit becomes 0.48  $\text{V}/\text{ms}$  at the EAP/EAN pins.

The Fusion GUI Numeric Encode/Decode Tester tool (found in the Tools menu) can be used to review how numeric values are encoded (written) and decoded (read) via PMBus.



### A.3 Clock Stretching

The PMBus hardware within the UCD92xx controller issues an acknowledge (ACK) or not-acknowledge (NACK) after each byte is transferred on the PMBus. The ACK handshake immediately follows the device Address byte and is automatically generated by the PMBus handler logic. The Command, Data, and optional packet error check (PEC) bytes are acknowledged after the firmware has determined that the byte is valid. In order for the firmware to validate the received byte, the device holds the clock line low. This condition is called *clock stretching*.

Block Write commands of more than 4 bytes produce clock stretching after receiving each 4 bytes of data, to wait for the firmware to copy those bytes from the hardware buffer to a RAM buffer. For instance, the CLA\_GIANS command is 29 bytes. Figure A-2 illustrates the clock-stretching effect for two traces.

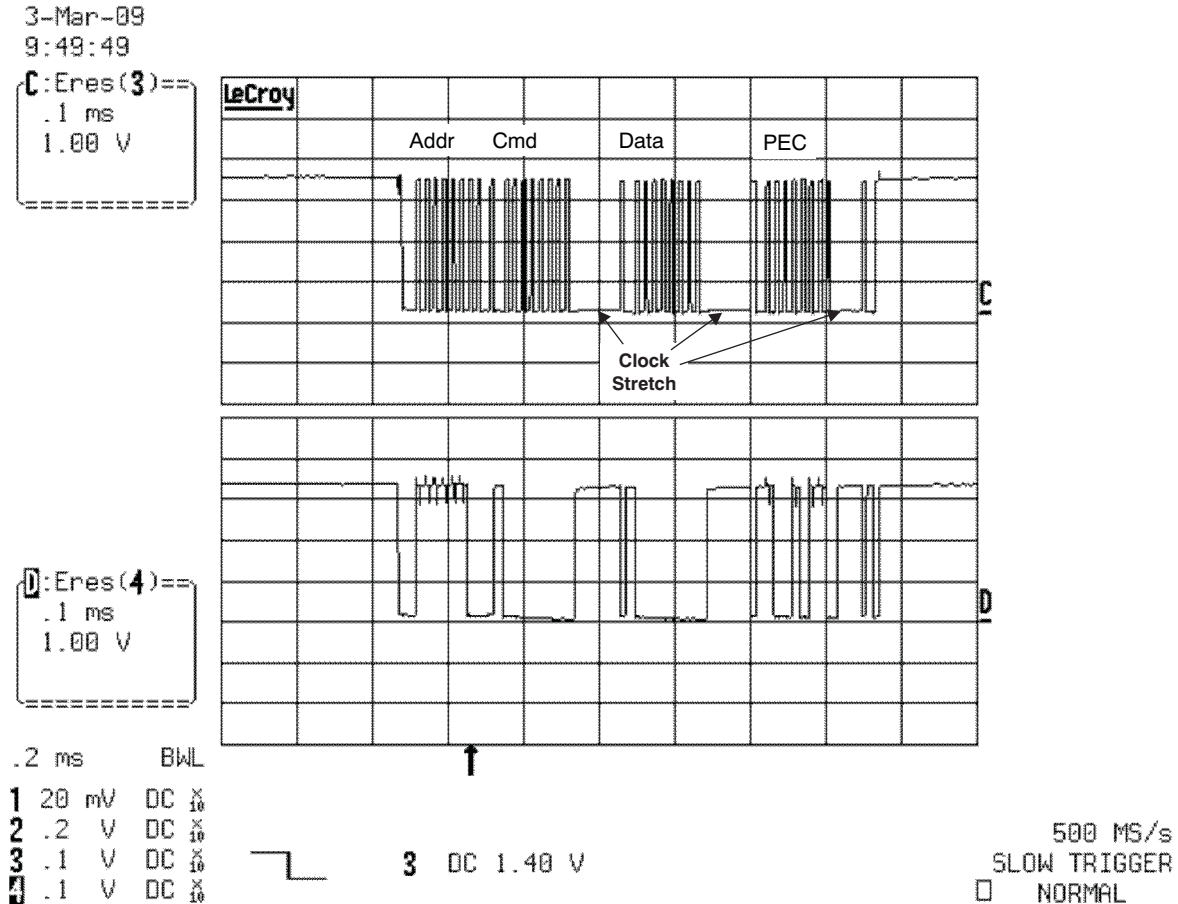


Figure A-2. Trace C: PMBus Clock, Trace D: PMBus Data



## Locating the Compensating Zeros

---



---

### B.1 Overview

Given a polynomial in  $s$ , such as that in [Equation 85](#) :

$$s^2 + \frac{\omega_z}{Q}s + \omega_z^2 \tag{85}$$

We can show that the real roots (zeroes) of the polynomial spread equally on a log plot on either side of a center frequency  $\omega_z$  by a spreading factor  $\beta$ , as calculated by [Equation 86](#).

$$(s + \omega_{z1})(s + \omega_{z2}) = (s + \beta\omega_z)\left(s + \frac{\omega_z}{\beta}\right) = s^2 + \left(\beta + \frac{1}{\beta}\right)\omega_z s + \omega_z^2 \tag{86}$$

where [Equation 87](#) shows the respective values.

$$\omega_{z1} = \frac{\omega_z}{\beta}, \omega_{z2} = \beta\omega_z, \text{ and } \omega_z^2 = \omega_{z1}\omega_{z2} \tag{87}$$

In other words, when  $\beta$  is 2.0, the lower frequency zero is at 1/2 of  $F_z$  and the higher frequency zero is at  $2.0 \times F_z$ , as shown in [Equation 88](#).

$$F_{z1} = \frac{1}{\beta} F_z, \text{ and } F_{z2} = \beta F_z \tag{88}$$

From [Equation 86](#),  $Q$  can be expressed in terms of the spreading factor  $\beta$  as shown by [Equation 89](#).

$$Q_z = \frac{\beta}{\beta^2 + 1} \tag{89}$$

Solving then for  $\beta$ , we find an expression for the spreading factor as a function of  $Q$ , given in [Equation 90](#):

$$\beta = \frac{1}{2Q_z} \left(1 + \sqrt{1 - 4Q_z^2}\right) \tag{90}$$

Because the center frequency  $F_z$  is the squared product of  $F_{z1}$  and  $F_{z2}$ , we can also express the spreading factor  $\beta$  and  $Q$  in terms of the resulting real zeroes of the polynomial shown in [Equation 91](#).

$$\beta = \sqrt{\frac{F_{z2}}{F_{z1}}} \text{ and } Q_z = \frac{\sqrt{F_{z1}F_{z2}}}{F_{z2} + F_{z1}} \tag{91}$$

Some example values for the spreading factor  $\beta$  are listed in [Table B-1](#).

**Table B-1. Example Spreading Factor ( $\beta$ ) Values**

Q	Spreading Factor $\beta$
0.010	100
0.050	20
0.099	10
0.100	9.899
0.123	8.000
0.192	5.000
0.200	4.791
0.300	3.000
0.333	2.618
0.345	2.500
0.400	2.000
0.462	1.500
0.500	1.000 (zeroes coincident)

From [Table B-1](#), if it is desired to place the zeroes at 1/2 and twice a given frequency,  $Q_z$  should be set to 0.400.

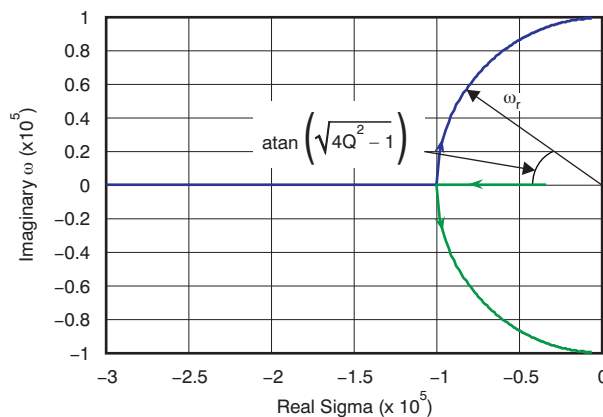
We can obtain these formulas by using the familiar solution to a quadratic equation. If we do some algebra, the relationship between the center, or resonant, frequency of the zeroes and the individual zero locations is shown in [Equation 92](#):

$$\omega_{z1} = \frac{\omega_z}{2Q_z} \left( 1 - \sqrt{1 - 4Q_z^2} \right) = \frac{\omega_z}{\beta}$$

$$\omega_{z2} = \frac{\omega_z}{2Q_z} \left( 1 + \sqrt{1 - 4Q_z^2} \right) = \beta\omega_z$$

(92)

To demonstrate this relationship, choose a  $F_r$  value of 15.91 kHz, which corresponds to  $\omega_r = 10^5$  radians. Then sweep the Q of the zeroes from 0.10 to 3.00. This result is shown in [Figure B-1](#).



**Figure B-1. Locus of Zeroes with Changing Q**

At low values of Q, the zeroes are far apart on either side of the  $F_r$ . When  $Q = 0.5$ , the two zeroes lie on top of each other exactly at  $F_r$ . When the Q is greater than 0.5, the zeroes are complex. These complex zeroes are complex conjugates of each other. That is, [Equation 93](#) shows the complex zeroes:

$$\omega_{Z1} = \omega_{\text{Real}} + j\omega_{\text{Imag}}$$

$$\omega_{Z2} = \omega_{\text{Real}} - j\omega_{\text{Imag}}$$

(93)

where Equation 94 is true:

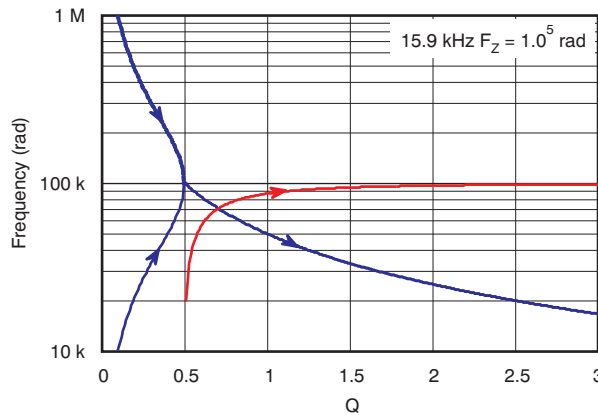
$$\omega_{\text{Real}} = \omega_z \frac{1}{2Q_z}$$

$$\omega_{\text{Imag}} = \omega_z \sqrt{1 - \frac{1}{4Q_z^2}}$$

(94)

Furthermore, the magnitude of each zero (which is the geometric distance from the origin to the zero) is  $\omega_R$ , and the angle of the zero from the origin is the arctangent of  $\sqrt{4Q^2 - 1}$ .

If we plot the real and imaginary frequencies as shown in Figure B-2, we can see that the two zeroes start with a small Q at an equal distance from  $\omega_R$  when plotted on a log scale. Then, as the zeroes become complex with a Q greater than 0.5, the magnitude of the real part decreases and the magnitude of the imaginary part heads toward  $\omega_R$ .



**Figure B-2. Real and Imaginary Frequency vs Q**

## IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

### Products

Audio	<a href="http://www.ti.com/audio">www.ti.com/audio</a>
Amplifiers	<a href="http://amplifier.ti.com">amplifier.ti.com</a>
Data Converters	<a href="http://dataconverter.ti.com">dataconverter.ti.com</a>
DLP® Products	<a href="http://www.dlp.com">www.dlp.com</a>
DSP	<a href="http://dsp.ti.com">dsp.ti.com</a>
Clocks and Timers	<a href="http://www.ti.com/clocks">www.ti.com/clocks</a>
Interface	<a href="http://interface.ti.com">interface.ti.com</a>
Logic	<a href="http://logic.ti.com">logic.ti.com</a>
Power Mgmt	<a href="http://power.ti.com">power.ti.com</a>
Microcontrollers	<a href="http://microcontroller.ti.com">microcontroller.ti.com</a>
RFID	<a href="http://www.ti-rfid.com">www.ti-rfid.com</a>
RF/IF and ZigBee® Solutions	<a href="http://www.ti.com/lprf">www.ti.com/lprf</a>

### Applications

Communications and Telecom	<a href="http://www.ti.com/communications">www.ti.com/communications</a>
Computers and Peripherals	<a href="http://www.ti.com/computers">www.ti.com/computers</a>
Consumer Electronics	<a href="http://www.ti.com/consumer-apps">www.ti.com/consumer-apps</a>
Energy and Lighting	<a href="http://www.ti.com/energy">www.ti.com/energy</a>
Industrial	<a href="http://www.ti.com/industrial">www.ti.com/industrial</a>
Medical	<a href="http://www.ti.com/medical">www.ti.com/medical</a>
Security	<a href="http://www.ti.com/security">www.ti.com/security</a>
Space, Avionics and Defense	<a href="http://www.ti.com/space-avionics-defense">www.ti.com/space-avionics-defense</a>
Transportation and Automotive	<a href="http://www.ti.com/automotive">www.ti.com/automotive</a>
Video and Imaging	<a href="http://www.ti.com/video">www.ti.com/video</a>
Wireless	<a href="http://www.ti.com/wireless-apps">www.ti.com/wireless-apps</a>

TI E2E Community Home Page

[e2e.ti.com](http://e2e.ti.com)

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265  
Copyright © 2011, Texas Instruments Incorporated