Step-by-Step Design Guide for Digital Peak Current Mode Control: A Single-Chip Solution

tags: digital, peak current mode, slope compensation

Dr. Ali Shirsavar, Dr. Michael Hallworth

Abstract

This application note investigates the implementation of peak current mode control with slope compensation using an single TMS320F28027 (Piccolo A) MCU from Texas Instruments. This MCU is ideal for peak current mode implementation due to its dedicated internal circuitry which enables a fully digital slope compensation scheme. The theory of operation, mathematical modeling and all relevant equations are presented along with a detailed step-by-step design procedure in both analog and digital domains.

A design example and associated experimental results are also presented with two methods of implementation; one using TI's ControlSUITE and one using Biricha's Chip Support Library (CSL).

Further information with regards to the CSL and digital power design workshops can be found at www.biricha.com/workshops/ (http://www.biricha.com/workshops/)

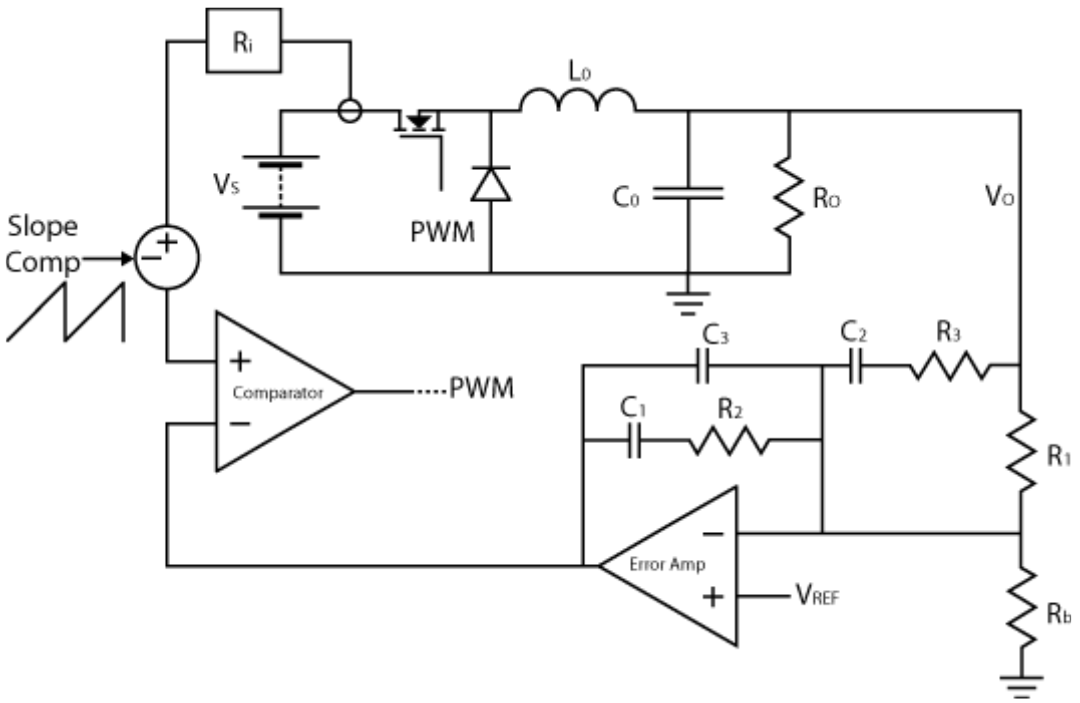

Figure 1. Analog Peak Current Mode Buck Converter

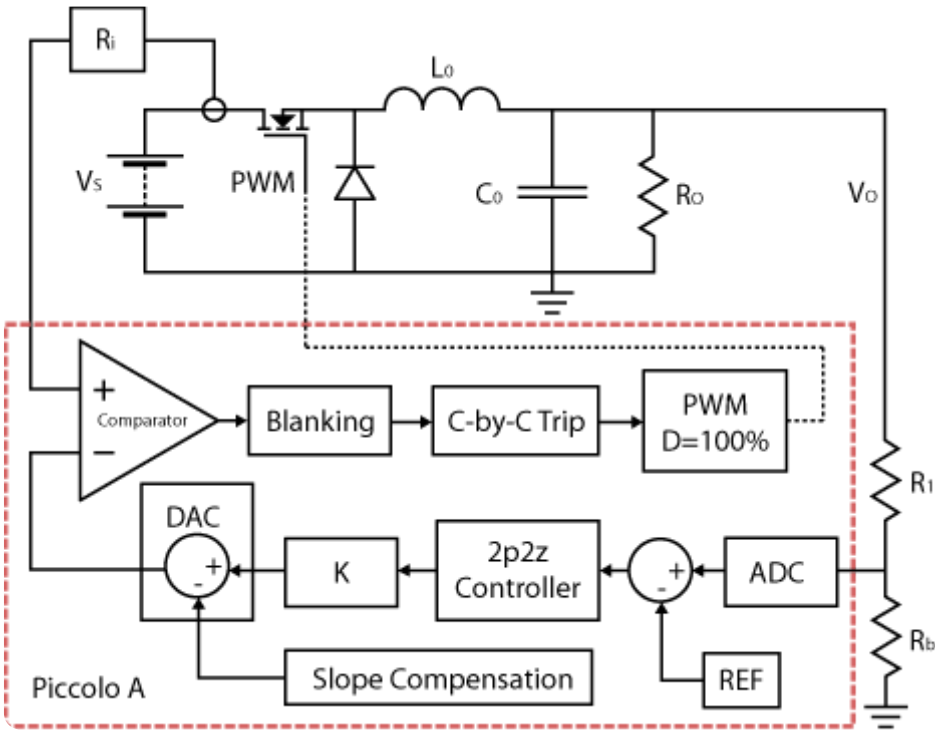

Figure 2. Digital Peak Current Mode Buck Converter

Introduction

The operation of a digital peak current mode converter is similar to its analog counterpart as shown in Figure 1. However, the compensation network, error amplifier, slope compensation and PWM generator are all replaced by a single microprocessor in the digital converter shown in Figure 2. This application note describes the processes involved in setting up a Piccolo MCU for use within a digital peak current mode power supply.

A complete design example and two complete implementations are given; the first method uses TI's ControlSUITE whilst the second method uses Biricha's Chip Support Library (CSL).

The Biricha Digital Chip Support Library provides a fast and simple method of configuring Texas Instruments C2000 MCUs for use in digital power applications. In-depth knowledge of the MCU's internal registers and associated configuration bits are not required; in place of this, simple function calls are used.

The Biricha Digital CSL documentation contains full descriptions and examples of all of the functions used in the implementation presented here. The user guides and an evaluation copy of CSL can be downloaded from www.biricha.com/resources/ (http://www.biricha.com/resources/)

Referring to Figure 2, the operation of the peak current mode controlled power supply is as follows. Initially, the duty is set to 100% and the PWM is driven high. The output voltage of the converter is applied to a sampling divider network which is connected to the Piccolo's ADC. The voltage is sampled and converted to a digital value. A digital reference (REF) is subtracted from the digital value and the resulting error value is used as an input to the digital controller (2p2z Controller). This represents the error amplifier and compensation network of the analog equivalent. A full design procedure for this controller will be given later on in this paper.

The output of the controller is then multiplied by a gain term K. This gain scales the output of the controller to a digital value that is suitable for use with the DAC of the comparator module, counteracting the effects of the various gains within the closed loop system. The value of this gain term can be calculated to obtain the correct crossover frequency; details are given in the design example.

This scaled output is then used as an input to the DAC connected to the comparator's inverting input. The non-inverting input is connected to the current sense transformer; the gain of this is represented by the Ri block. The current spike associated with turning the MOSFET switch on is ignored through the use of leading edge blanking within the Piccolo's blanking block. The output of the comparator will change state when the inductor current reaches the level of the voltage on the DAC output. This causes a cycle-by-cycle trip event to occur within the digital compare sub-module of the PWM module. The PWM signal will be low for the remainder of the switching period. Therefore, as with the analog equivalent, the duty is determined by the peak of the inductor current in the power stage of the converter. The digital implementation of peak current mode control achieves the desirable cycle-by-cycle peak current limiting effect of the analog equivalent. Full experimental results will be given shortly.

Peak Current Mode Model

The Buck converter in Figure 3 is used to describe the peak current mode model used in this application note. However the same procedure can be applied to other topologies.

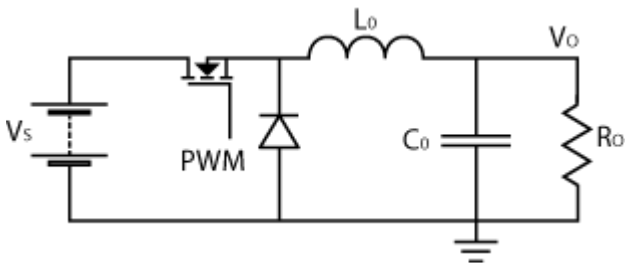

Figure 3 - A typical Buck converter

In order to design a stable compensator, we first need a mathematical model of the Buck converter plant. According to [3], this can be described by three terms:

1. $F_H(s)$, a high frequency transfer function.
2. H_{DC}, a DC gain.
3. $H_E(s)$, a power stage small signal model.

The complete control-to-output transfer function for a Buck converter under peak current mode control, as described in [1], is a combination of these three terms:

$$H_{CO}(s) = \frac{V_O(s)}{V_{IN}(s)} = F_H(s) \,.\, H_E(s) \,.\, H_{DC}$$

Equation 1

High Frequency Transfer Function

The high frequency transfer function, $F_H(s)$, has a double pole at half the switching frequency, $F_S/2$. Inevitably this will result in a resonant peak occurring at this frequency. Therefore this peak needs to be damped in order to avoid the gain Bode plot crossing the 0dB axis at resonance and causing instability.

A compensating ramp is added to the system to effectively damp resonant peak; this called “slope compensation”. This is achieved by setting the Q of this double pole system to 1. A low Q forces a damped this resonant peak and reduces the gain at $F_S/2$.

For a Buck converter, the required peak-to-peak value of the external compensation ramp has been calculated in [2] and is given in Equation 2. This compensating ramp reduces the Q of the high frequency transfer function to 1.

$$V_{PP} = -\frac{(0.18 - D)\, R_i T_S V_{IN}}{L_0}$$

Equation 2

Where:

Current-sense transformer gain: R_i

Switching period: T_S

Input voltage: V_{IN}

Output inductor: L_0

The duty, D, is: $D=V_O/V_{IN}$

Where V_O is the output voltage.

With Q set to 1 $F_H(s)$ simplifies to that given in Equation 3.

$$F_H(s) = \frac{1}{1 + \frac{s}{\omega_N} + \frac{s^2}{\omega_N^2}}$$

Equation 3

Where:

$$\omega_N = \pi F_S \text{ in rad/s, i.e. } \frac{1}{2} F_S \text{ in } Hz$$

DC Gain

Using Ridley's model in [1] and critically damping the resonance peak by setting it's Q to 1, the DC gain of the system simplifies to:

$$H_{DC} = \frac{R_O}{R_i} \cdot \frac{1}{1 + \frac{R_O T_S}{\pi L_0}}$$

Where the load resistance is R_O.

Power Stage Small Signal Model

With current mode control, the inductor of the Buck converter in Figure 3 becomes a current controlled source. In [3] the small-signal model of the Buck power stage is given as:

$$H_E(s) = \frac{1 + \frac{s}{\omega_{ESR}}}{1 + \frac{s}{\omega_{OP}}}$$

Equation 4

The pole, ω_{OP}, is formed from the output capacitance and load resistance. With Q set to 1, this pole can be calculated in Equation 5.

$$\omega_{OP} = \frac{1}{R_O C_0} + \frac{T_S}{\pi L_0 C_0}$$

Equation 5

Furthermore, the zero, ω_{ESR}, formed from the output capacitance and its equivalent series resistance is:

$$\omega_{ESR} = \frac{1}{R_{ESR} C_0}$$

Equation 6

Please note that these are in rad/s and not Hz.

Compensator Poles and Zeros

The poles and zeros of the compensation network can now be placed according to the analysis of the control-to-output transfer function in order to set the desired crossover frequency and phase margin of the closed loop system. A Type II compensator is used to control the Buck converter under peak current mode. The transfer function of a Type II compensation network is:

$$H_C(s) = \left(\frac{\omega_{CP0}}{s}\right) . \frac{\left(1 + \frac{s}{\omega_{CZ1}}\right)}{\left(1 + \frac{s}{\omega_{CP1}}\right)}$$

Equation 7

The pole, ω_{CPI}, of the compensator is set to the frequency of the ESR zero in the control-to-output transfer function in order to approximately cancel out its effects.

$$\omega_{CP1} = \frac{1}{R_{ESR} C_0}$$

Equation 8

The zero, ω_{CZI}, is set to achieve a suitable phase margin and the pole at origin, ω_{CP0}, is set to achieve the desired crossover frequency. The frequency of the compensator zero should be set to 20% of the required crossover frequency. Under most circumstances this will give a reasonable phase margin.

$$\omega_{CZ1} = \frac{1}{5} . 2\pi f_X$$

Equation 9

Where f_X is the crossover frequency in Hertz.

Finally, the pole at origin (or gain of the compensator) is calculated. This is the frequency at which the gain solely due to the pole at origin would be unity. This value sets the desired crossover frequency, f_X. After analyzing the Buck converter's control-to-output transfer function, Equation 10 has been derived for directly calculating ω_{CP0} of the compensator [2].

$$\omega_{CP0} = \frac{1.23 f_X R_i (L_0 + 0.32 R_O T_S) R_1 R_2}{L_0 R_O}$$

Equation 10

Where:

$$R_1 = \sqrt{1 - 4 f_X^2 T_S^2 + 16 f_X^4 T_S^4}$$

$$R_2 = \sqrt{1 + \frac{39.48 C_0^2 f_X^2 L_0^2 R_O^2}{(L_0 + 0.32 R_O T_S)^2}}$$

Please note that these values are calculated in rad/s and not Hz.

A complete design example using these equations will be given later in this application note.

Please be aware that in this application note only an approximate solution is given. Both the zero and pole at the origin can be calculated analytically and full details of the analytical method is taught in the Biricha Digital workshops.

See www.biricha.com/workshops/ (http://www.biricha.com/workshops/) for more information.

Digital Controller Design

The poles and zeros of the analog compensation network have been calculated based on the model given in Section 3. These poles and zeros must be converted in to the digital domain. This involves converting from the continuous time s-domain to the discrete time z-domain. There are various methods to achieve this. The Bilinear or Tustin transform is a relatively simple and effective method.

There is no need for this transform to be calculated by hand as an automated tool exists on the Biricha Digital website to convert from s-domain poles and zeros to the coefficients required by the discrete time digital controller.

Please visit www.biricha.com/resources/ (http://www.biricha.com/resources/) to access these tools free of charge.

However, the process is described in detail here for completeness. The transfer function of a Type II compensation network can be converted in to the z-domain by replacing the 's' terms with the approximation:

$$s \approx \frac{2}{T_S}\frac{z-1}{z+1}$$

Equation 11

Where T_S is the sampling period which is equal to the switching period. Equation 11 is then substituted in to the Type II controller transfer function given in Equation 7. The result is given in Equation 12.

$$H_C[z] = \left(\frac{\omega_{CP0}}{\frac{2}{T_S}\frac{z-1}{z+1}}\right).\frac{\left(1+\frac{\frac{2}{T_S}\frac{z-1}{z+1}}{\omega_{CZ1}}\right)}{\left(1+\frac{\frac{2}{T_S}\frac{z-1}{z+1}}{\omega_{CP1}}\right)}$$

Equation 12

After simplification, the two-pole two-zero discrete controller transfer function is found:

$$H_C[z] = \frac{Y[z]}{X[z]} = \frac{B_2 z^{-2} + B_1 z^{-1} + B_0}{-A_2 z^{-2} - A_1 z^{-1} + 1}$$

Equation 13

Where:

$$B_0 = \frac{T_S . \omega_{CP0} . \omega_{CP1} \times (2 + T_S . \omega_{CZ1})}{2 \times (2 + T_S . \omega_{CP1}) \times \omega_{CZ1}}$$

$$B_1 = \frac{T_S^2 . \omega_{CP0} . \omega_{CP1}}{2 + T_S . \omega_{CP1}}$$

$$B_2 = \frac{T_S . \omega_{CP0} . \omega_{CP1} \times (-2 + T_S . \omega_{CZ1})}{2 \times (2 + T_S . \omega_{CP1}) \times \omega_{CZ1}}$$

$$A_1 = \frac{4}{2 + T_S . \omega_{CP1}}$$

$$A_2 = \frac{-2 + T_S . \omega_{CP1}}{2 + T_S . \omega_{CP1}}$$

Equation 13 can now be rearranged to find the linear difference equation (LDE):

$$y[n] = B_2 x[n-2] + B_1 x[n-1] + B_0 x[n] + A_2 y[n-2] + A_1 y[n-1]$$

Equation 14

Where $x[n]$ is the error input to the controller for this sampling period and $y[n]$ is the controller output for this sampling period. $[n-1]$ denotes the previous sampling period and $[n-2]$ is two sampling periods in the past.

The coefficients of the discrete time controller are used with this linear difference equation.

Please note that it is now possible to calculate all of these controller coefficients analytically as all the variables within the coefficients are known; we will give a design example shortly.

Digital Slope Compensation

As mentioned earlier, sufficient ramp needs to be added such that no subharmonic oscillations occur; this is called slope compensation. The oscillations are caused by the current feedback loop which has a double pole in the high frequency term, $F_H(s)$, in the control-to-output transfer function. The resonant peak of the double pole (at half the switching frequency) was damped by setting the Q to 1 through calculating the compensation ramp in Equation 2. Now we will discuss the digital implementation of this compensation ramp.

The Piccolo range of MCUs from TI are ideally suited for this purpose due to the presence of dedicated ramp generating modules.

The DAC module of the Piccolo includes a ramp sub-module which is used to implement slope compensation. This slope compensation method uses a digital staircase to remove the subharmonic oscillations [6].

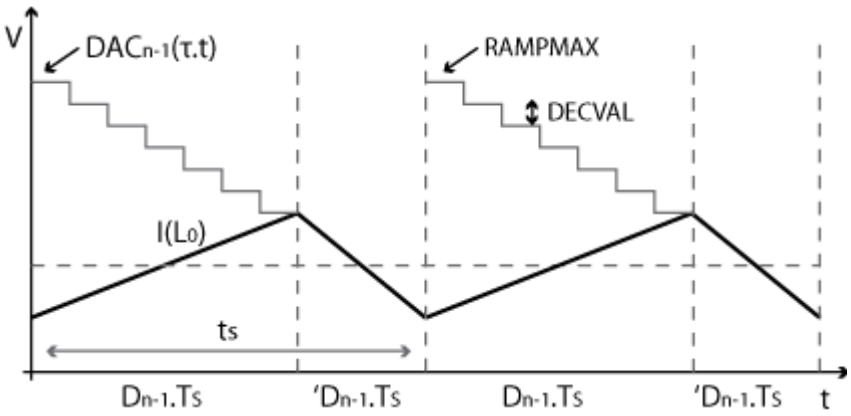

Figure 4. Slope compensation using a digital staircase

In Figure 4, the DAC is initially loaded with the output of the controller after being scaled by K. This sets the initial value of our demand current reference signal before slope compensation. The sensed current is compared to this reference current. The counter sub-module then decrements the DAC by a set value at each system clock tick. This generates a negative ramp on the current reference signal, just as slope compensation would in analog, and effectively damps the oscillations.

The required compensation ramp height must first be converted to a digital value using the gain of the DAC:

$$\text{DigitalRampHeight} = V_{PP}.\frac{2^{DAC_{bits}}-1}{V_{DAC}}.64$$

Equation 15

The value to be decremented at each system clock tick can therefore be found using the digital ramp height, switching frequency and system clock frequency:

$$\text{DecVal} = \frac{\text{DigitalRampHeight}.F_S}{F_{CLK}}$$

Equation 16

If using the CotnrolSuite these equations can be used to set up the correct values in the relevant registers; please see the appendix. However, if using the CSL dedicated CSL functions can be called within the initialization in order to automatically set these values.

A selection of Biricha CSL functions needed for slope compensation are described below. Further practical examples as well as hands-on exercises are provided during the workshops run by Biricha.

The ramp sub-module is first configured using the function `CMP_rampConfig()`. This connects the specified ramp sub-module with a PWM module. The PWM module synchronization pulse is used to reset the ramp back to the initial value at the beginning of every period.

```
CMP_rampConfig( CMP_MOD_2,
                PWM_MOD_1 );
```

The value DECVAL is decremented from the DAC at each system clock tick. This value is set using `CMP_setRampDec()` after the ramp module has been configured. The peak-to-peak value of the analog compensation ramp was calculated in Equation 2. The Biricha CSL function `CMP_calcRampDec()` takes this analog voltage value and returns the correct digital decrement value.

This function requires the digital equivalent of the ramp height along with the period of the PWM output. The digital equivalent of the ramp height is calculated using `CMP_mVtoRampValue()`. For example, if a 1500mV ramp is required to compensate a 200kHz PWM output the following functions would be called during initialization:

```
period = PWM_freqToTicks(200000);
decval = CMP_calcRampDec( CMP_mVtoRampValue(1500),
                          period );
CMP_setRampDec( CMP_MOD_2, decval );
```

The controller output, or reference current, is set as the initial value of the ramp at the beginning of the period using the function `CMP_setRampMax()`. This is called in the ADC interrupt routine after the controller output has been calculated.

Leading Edge Blanking

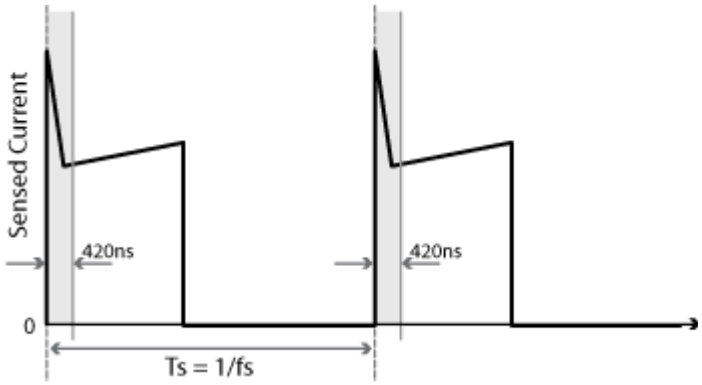

Figure 5. The required amount of leading edge blanking

The sensed inductor current signal contains switching noise. During turn on there is a large spike which would potentially cause the comparator to trigger and a trip event to occur. To prevent this, a leading edge blanking block is provided within the Piccolo's comparator module. For a specified window period the comparator's output is ignored by the digital compare sub-module within the Piccolo.

The designer must specify the number of nanoseconds required for the duration of the blanking window. This can either be set using the TI's ControlSUITE or by calling CLS's `PWM_setBlankingWindow()` function. An initial value should be set and its suitability can be confirmed using an oscilloscope. In the example shown in Figure 5 a window of 420ns is shown. This can be set up using the functions described below.

```
PWM_configBlanking( PWM_MOD_1,
                    PWM_CMP_COMP2,
                    GPIO_NON_INVERT,
                    true );

PWM_setBlankingWindow( PWM_MOD_1,
                       PWM_nsToTicks(420) );
```

The `PWM_configBlanking()` function configures the digital compare sub-module of the PWM module to use the output of the comparator as an event trigger.

Design Example

In this example a 12V to 3.3V digital peak current mode Buck converter is designed, however, the design principles used here can be applied to most other topologies with slight modifications.

The converter specification is as follows:

Parameter	Value
V_{IN}	12V
V_O	3.3V
I_O	2A
R_L	1.65Ω
L_0	22µH
C_0	440µF
R_{ESR}	31mΩ
R_i	0.48
D	0.275
f_s	200kHz
f_x	10kHz

The switching frequency is chosen as $F_S = 200kHz$ and the desired crossover frequency is $f_X = 10kHz$.

Step 1: Calculate the size of the compensating ramp

Calculate the peak-to-peak value of the external compensation ramp required to achieve $Q_C = 1$ using Equation 2:

$$V_{PP} = -\frac{(0.18 - D)\, R_i T_S V_{IN}}{L_0}$$

$$V_{PP} = -\frac{(0.18 - 0.275) \times 0.48 \times 5\mu s \times 12}{22\mu H}$$

$$V_{PP} = 0.124V$$

Equation 17

Step 2: Calculate the positions the poles and zeros of our “analog” compensator

In the case of our Buck converter a Type II compensator is used. The transfer function is given in Equation 7. The pole, ω_{CP1}, is used to cancel out the ESR zero of the output capacitor and equivalent series resistance:

$$\omega_{CP1} = \omega_{ESR}$$

$$\omega_{CP1} = 73,314 rads^{-1}$$

$$(11,668 Hz)$$

The zero of the compensator is used to set the phase margin of the open loop system at the crossover frequency. An approximate solution which gives reasonable results is to set the zero to one fifth of the crossover frequency.

$$\omega_{CZ1} = 12,566 rads^{-1}$$

$$(2,000 Hz)$$

Finally, the pole at the origin (or gain of the compensator) is calculated to achieve the desired crossover frequency using Equation 10:

$$\omega_{CP0} = 162,464 rads^{-1}$$

$$(25,857 Hz)$$

Figure 6 shows the frequency response of the plant (i.e. control-to-output transfer function). This clearly shows the double-pole at half the switching frequency. The choice of compensation ramp has effectively damped the resonant peak of this double pole.

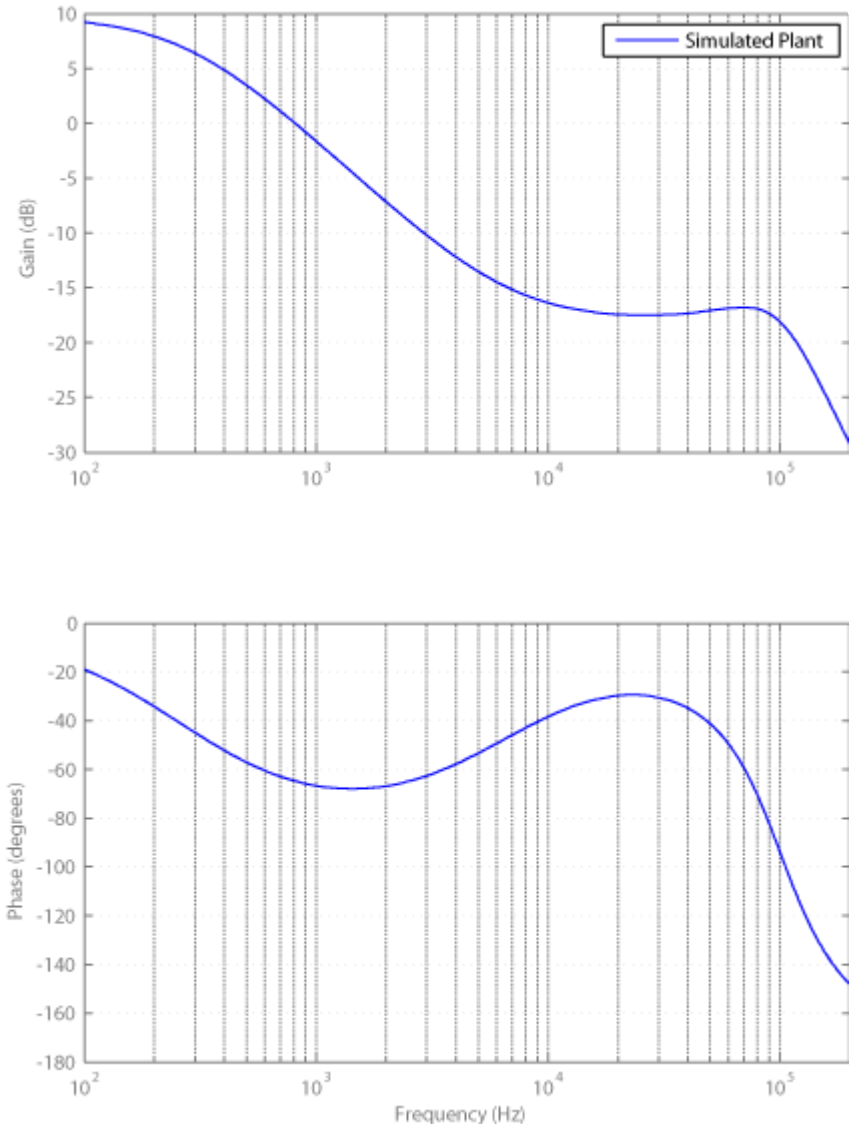

Figure 6. Bode plot of the plant (control-to-output transfer function)

The frequency response of the compensator is represented by the dotted trace in Figure 7 whilst the solid trace on this plot represents the combined plant and controller transfer function; this is the open loop frequency response of our system. The gain and phase margins of our system and hence its relative stability are determined from this trace.

The controller has been designed to achieve large phase margin at the crossover frequency of 10kHz. The Bode diagram of the open loop system indicates that the poles and zeros of the controller achieve a phase margin of 74.5° at the crossover frequency.

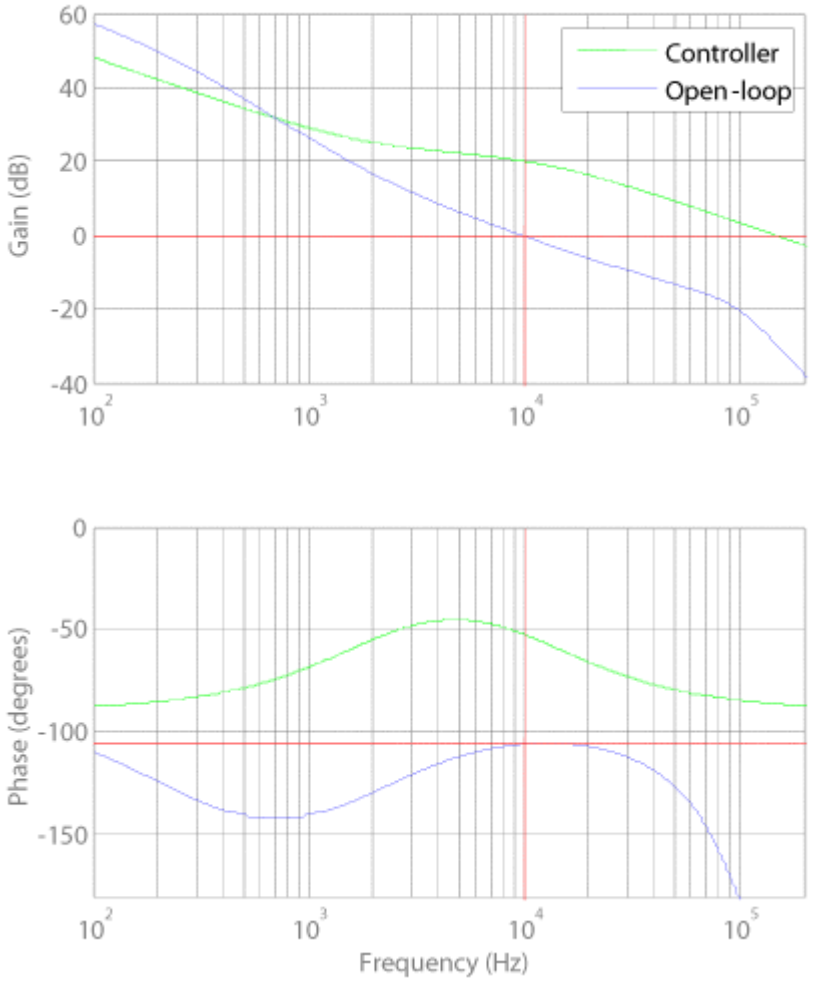

Figure 7. Bode plots of Controller and Open-loop system (GM = 20.1dB, PM=74.5°)

Step 3: Convert the analog compensator to digital

In the previous step we designed an analog compensator to stabilize our power supply. We now convert this into its digital equivalent.

We use the bilinear transform for this purpose using the automated tool available at www.biricha.com/resources/ (http://www.biricha.com/resources/converter.php?type=2) .

The following information is entered into the online form:

Crossover frequency of analog pole at origin: $f_{CP0} = \frac{\omega_{CP0}}{2\pi} = 25857Hz$

Frequency of first pole: $f_{CP1} = \frac{\omega_{CP1}}{2\pi} = 11668Hz$

Frequency of first zero: $f_{CZ1} = \frac{\omega_{CZ1}}{2\pi} = 2000Hz$

Switching frequency: $F_S = 200kHz$

The result is automatically calculated as:

$$H[z] = \frac{U[z]}{E[z]} = \frac{B_2 z^{-2} + B_1 z^{-1} + B_0}{-A_2 z^{-2} - A_1 z^{-1} + 1}$$

Equation 18

Where:

$\mathbf{A_1}$ = 1.6902106568
$\mathbf{A_2}$ = -0.6902106568
$\mathbf{B_0}$ = 2.065467832
$\mathbf{B_1}$ = 0.1258242849
$\mathbf{B_2}$ = -1.9396435478

These are the coefficients that the designer should use with the two-pole two-zero controller equation. The CSL of course provides a dedicated function of this type of controller.

Step 4: Initialize the controller

When using either the Biricha CSL or controlSUITE the digital current mode controller parameters are entered at the top of the C file using `#define` statements.

A reference is used to calculate the digital error value by subtracting the ADC output from the reference. The digital error value is then used as an input to the controller. Therefore the reference value must be equal to the digital equivalent of the output voltage multiplied by the sampling divider gain.

$$REF = V_O . \frac{R_B}{R_1 + R_B} . \frac{2^{ADC_{bits}} - 1}{V_{DAC}}$$

Equation 19

For the 3.3V output, with a sampling divider gain of 0.5 and 12 bit ADC:

$$REF = 3.3 \times 0.5 \times \frac{4095}{3.3}$$
$$REF = 2048$$

Equation 20

The 2p2z controller can now be configured by defining the following constants at the top of the main C file.

```
/* Set up the coefficients for the 2p2z controller */
#define REF (_IQ15toF(2048))
#define MIN_DUTY 0
#define MAX_DUTY 65535
#define A1  +1.6902106568
#define A2  -0.6902106568
#define B0  +2.0654678327
#define B1  +0.1258242849
#define B2  -1.9396435478
#define PERIOD_NS  5000 /* Period in ns for fs = 200kHz */
```

The final term to be calculated is that of the scaling factor K. This is used to negate the effects of the gains within the microcontroller and thus achieve the correct crossover frequency. The value of K can be calculated accurately; however this is beyond the scope of this text and is discussed in detail in Biricha's digital power design workshops.

For the purpose of this application note we start with an initial value of K of say 10, implement the software and then measure the loop using a Bode 100 network analyzer from OMICRON Lab. K is then adjusted in software to achieve the correct cross over frequency. In Section 9 we describe how the code is implemented, we measure the loop and adjust K for a perfect crossover.

Real Life Implementation

Figure 8, Figure 9 and Figure 10 represent the flowcharts for our system. Both ControlSUITE and CSL implementations follow the same software structure and hence the flowcharts are valid for both. Complete listings for both implementations are given in the appendix.

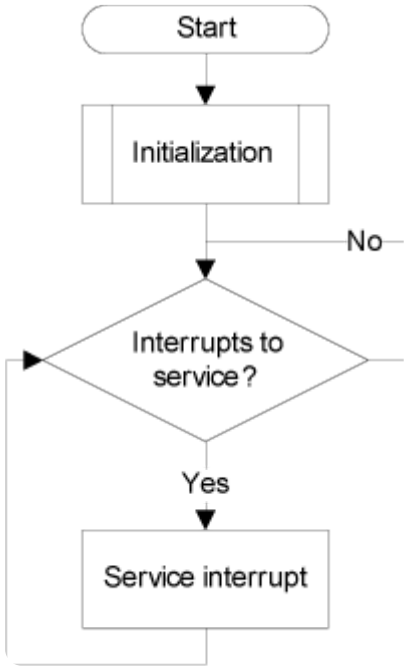

Figure 8. Main function program flow

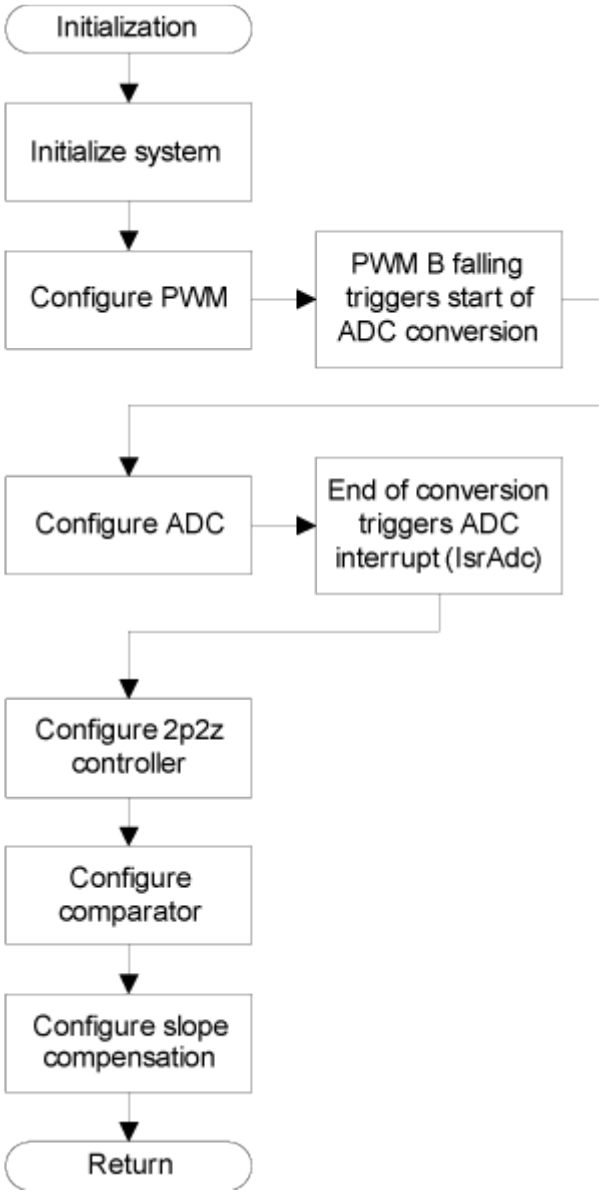

Figure 9. Main function initialization routine

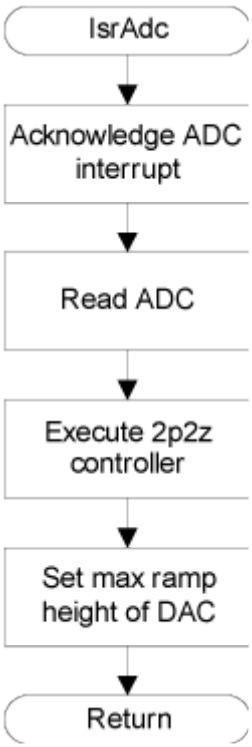

Figure 10. ADC and CLA interrupts

An analysis of the code within the main() function will now be presented. First of all, the system and peripherals must be initialized before they can be used.

```
/* Initialize the MCU and ADC */
SYS_init();
ADC_init();
```

The PWM module is configured to operate at the switching frequency of 200kHz. Channel A is connected to the MOSFET Driver IC and controls the switching of the MOSFET in the Buck power stage.

The duty of channel A is set to 100% duty as with peak current mode control the effective duty is determined when the current through the switch reaches the output of the controller. At this point the PWM signal goes low and the MOSFET turns off.

The counter register resetting to zero is used to trigger the sampling and conversion of the output voltage. At the end of conversion an ADC interrupt is automatically triggered and the 2p2z algorithm is executed.

The initialization code within the `main()` function is discussed below.

```
/* Setup PWM_MOD_1 for fs = 200kHz. PWM1 Ch A is used for switching
 * the MOSFET.
 */
PWM_config( PWM_MOD_1, PWM_nsToTicks(PERIOD_NS), PWM_COUNT_UP );
PWM_pin( PWM_MOD_1, PWM_CH_A, GPIO_NON_INVERT );
PWM_pin( PWM_MOD_1, PWM_CH_B, GPIO_NON_INVERT );

/* Set the maximum duty to 100%. The trip zones (configured later)
 * will end the high output of the PWM when the current reaches
 * the slope level.
 */
PWM_setDutyA( PWM_MOD_1, PWM_nsToTicks(PERIOD_NS) );

/* Sets the PWM1 Ch B such that the calculations are complete
 * just before the rising edge PWM A.
 */
PWM_setDutyB( PWM_MOD_1, PWM_nsToTicks( PERIOD_NS-2450+0 ) );

/* This sets up the PWM Mod1 to start the ADC conversion whenever
 * PWM1 Channel B timebase counter matches Ch B's duty.
 */
PWM_setAdcSoc( PWM_MOD_1, PWM_CH_B, PWM_INT_CMPB_UP );

/* This sets up PWM Mod1 to generate an interrupt every PWM
 * cycle whenever timebase counter = 0.
 */
PWM_setCallback( PWM_MOD_1, 0, PWM_INT_ZERO, PWM_INT_PRD_1 );
```

The following functions set up the cycle-by-cycle trip of the PWM output triggered by the comparator output. The `PWM_configBlanking()` function effectively connects the comparator output to the PWM module using the digital compare sub-module. The blanking window size is set within the digital compare sub-module and the digital compare event is used for the trip zones configured within PWM module.

```
/* This effectively feeds the output of comparator Mod2 into
 * PWM Mod1 and activates the blanking by setting the digital
 * compare event PWM_DCEVT at the correct time.
 */
PWM_configBlanking( PWM_MOD_1, PWM_CMP_COMP2, GPIO_NON_INVERT,
                        true );

/* Sets the size of the blanking window to 420ns */
PWM_setBlankingWindow( PWM_MOD_1, PWM_nsToTicks(420) );

/* Sets the relevant trip zones: i.e. when PWM_DCEVT occurs clear
 * PWM1 Ch A on a cycle by cycle basis but takes no action on
 * PWM1 Ch B
 */
PWM_setTripZone(  PWM_MOD_1, PWM_DCEVT, PWM_TPZ_CYCLE_BY_CYCLE );
PWM_setTripState( PWM_MOD_1, PWM_CH_A,  GPIO_CLR );
PWM_setTripState( PWM_MOD_1, PWM_CH_B,  GPIO_NO_ACTION );
```

The ADC Module 1 is configured to read and convert the output voltage from Channel B2. The conversion is triggered from the start of conversion event of PWM module 1. When the conversion is complete an interrupt is called and the interrupt service routine `IsrAdc()` is entered. This interrupt service routine is included in the appendix.

```
/* Configures ADC to sample Vo when triggered by PWM1 Ch B's
 * falling edge
 */
ADC_config( ADC_MOD_1, ADC_SH_WIDTH_7, ADC_CH_B2, ADC_TRIG_EPWM1_SOCB );

/* When conversion is finished, cause interrupt and jump to IsrAdc
 */
ADC_setCallback( ADC_MOD_1, IsrAdc, ADC_INT_1 );
```

The control structure is initialized with the values determined from the bilinear transform of the compensator transfer function. A soft start can also be configured.

```
/* Initalise the 2p2z control structure */
CNTRL_2p2zInit(&MyCntrl
               ,_IQ15(REF)
               ,_IQ26(A1),_IQ26(A2)
               ,_IQ26(B0),_IQ26(B1),_IQ26(B2)
               ,_IQ23(K),MIN_DUTY,MAX_DUTY
               );

/* Set up a 500ms soft-start */
CNTRL_2p2zSoftStartConfig(&MyCntrl, 500, PERIOD_NS );
```

The comparator is configured in asynchronous mode with a non-inverted output. The inverting input of the comparator is tied to the internal DAC. The DAC value is set by the control algorithm.

```
/* Configures the comparator Mod2 */
CMP_config( CMP_MOD_2, CMP_ASYNC, GPIO_NON_INVERT, CMP_DAC );
```

Slope compensation is achieved using the ramp sub-module of the comparator. First, the sub-module must be synchronized with the PWM period using `CMP_rampConfig()`. The required ramp height must be converted to a digital value before being passed as an argument to `CMP_setRampDec()`.

```
CMP_rampConfig( CMP_MOD_2,  PWM_MOD_1 );
decval = CMP_calcRampDec( CMP_mVtoRampValue(124), PWM_freqToTicks(200000) );
CMP_setRampDec( CMP_MOD_2, decval );
```

Global interrupts must be enabled before any of the interrupts can be serviced. After this, the execution waits in an idle loop as all of the events will now occur using interrupts.

```
/* Enables global interrupts and wait in idle loop */
INT_enableGlobal(true);

while(1) {}
```

The complete code listing, including the interrupt functions, can be found in the appendix.

The code entry is now complete. The final value to be determined is that of the gain term, K, which determines the correct crossover. This scales the output of the controller to a digital value that is suitable for use with the DAC of the comparator module and negates the effects of the gains within the system. Initially this scaling factor was set to 10. The most effective way to determine the correct scaling factor is to measure the frequency response of the complete system using a network analyzer.

The Bode 100 Vector Network Analyzer from OMICRON Lab is used to measure the frequency response of the converter by injecting a small-signal perturbation on to the output voltage and measuring the response of the system.

The code is compiled and downloaded to the microcontroller. The working converter is connected to the Bode 100 by means of a small injection resistor (9.1&) in the output voltage feedback path to the ADC of the microcontroller.

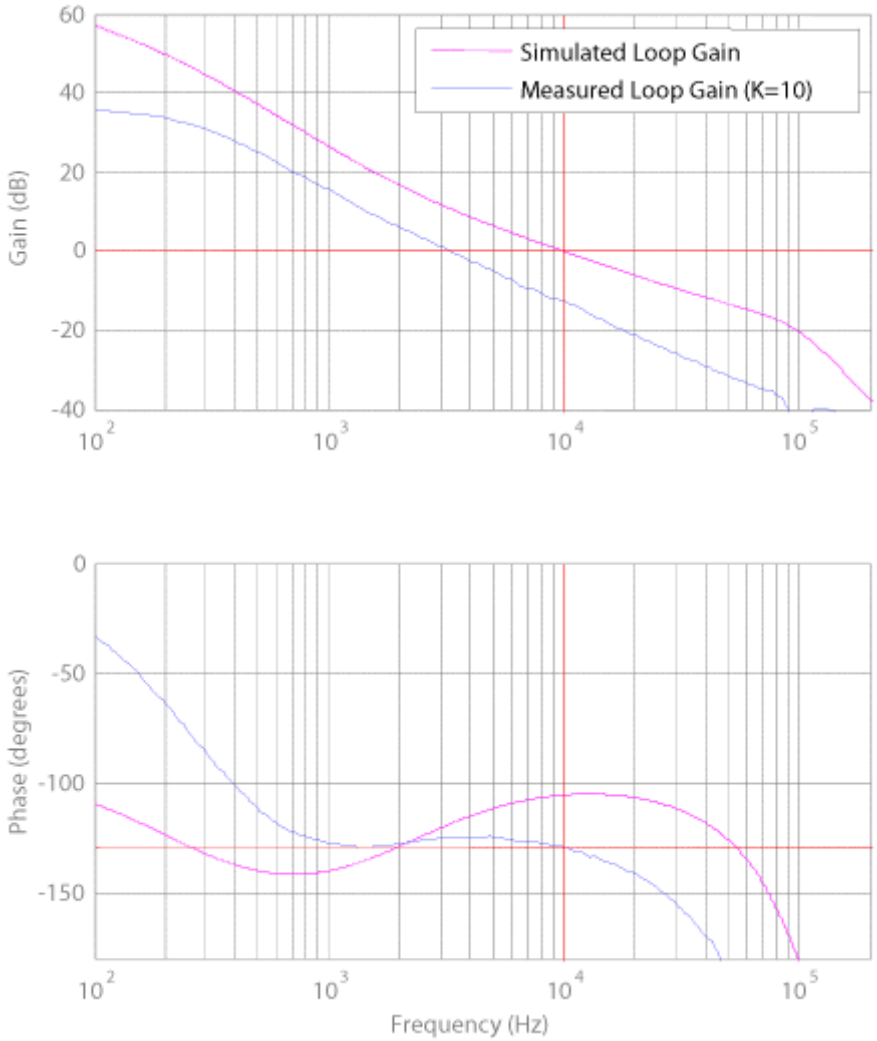

Figure 11. Frequency response of open loop system. Measured using Bode 100 from OMICRON Lab

The correct value for K can be determined from the frequency response data shown in Figure 11 as follows. Record the gain, in decibels, at the desired crossover frequency. Using Equation 21 the required value of K can be calculated using the initial value for K, K_{INIT}.

$$K = K_{INIT} \times \frac{1}{10^{\frac{dB}{20}}}$$

Equation 21

In this case, at the required crossover frequency of 10kHz, the magnitude response of the open loop system is recorded as -12.47dB. Therefore, for this system, K is calculated in Equation 22.

$$K = 10 \times \frac{1}{10^{\frac{-12.47}{20}}} = 42.0243$$

Equation 22

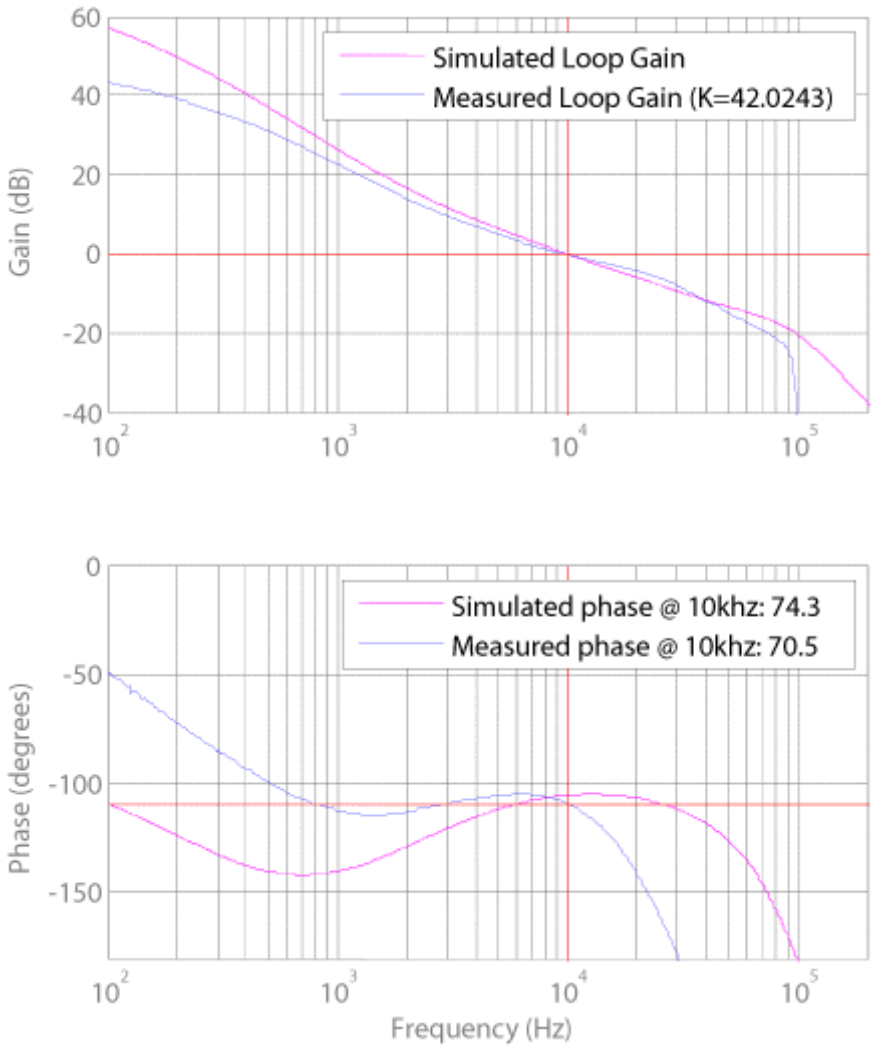

Figure 12. Frequency response of open loop system. Measured using Bode 100 from OMICRON Lab

The code is updated with the new calculated value for K. After recompiling and updating the microcontroller the frequency response sweep is performed again to confirm the correct crossover frequency. The result in Figure 12 shows a crossover of 10kHz with 70.5 degrees of phase margin. The system is stable and has a fast transient response.

An accurate analytical method for calculating the scaling gain K is taught in the workshops presented by Biricha Digital. Visit www.biricha.com/workshops/ (http://www.biricha.com/workshops/) for more information.

In Figure 12 the simulated open loop model (dashed line) is compared with the measured results (solid line). The low frequency discontinuities are to be expected.

The top magnitude plot of Figure 12 shows a particular characteristic of digital converters that is apparent at low frequencies. The gain is less than the simulated model. This is due to a combination of the quantization effects of the ADC and the precision of the fixed point arithmetic. This result draws parallels to the gain-bandwidth-product limitation of analog operational amplifiers and a similar result also found in the analog domain [6].

The lower phase plot of Figure 12 shows good agreement with the predicted model around the crossover frequency. Phase roll-off becomes apparent as the perturbation frequency approaches the switching frequency. At lower frequencies the quantization and fixed point arithmetic again mask the effects of the pole at origin.

Overall the measured results show a good correlation to the results of the model. The measured open loop gain crosses the 0dB axis at the desired crossover frequency with approximately 70.5 degrees of phase margin and a 10dB gain margin.

Conclusion

This application note has described a method for designing a digital peak current mode power supply. An example of a Buck converter is used to illustrate this process. The measured small-signal frequency response of the digital power supply matches with the predicted response from the current mode model.

The digital power supply offers advantages over the analog equivalent. This is a software solution which can easily be modified to meet end user requirements. One Piccolo can be used to control multiple power supplies and advanced or non-linear control methods are possible. The designer can use the remaining microprocessor bandwidth to perform other tasks such as predicting possible failures.

This application note has explained one possible implementation of a digital power supply. Further applications of digital power are explored from a hardware designer's perspective in the Digital Power Workshops available from Biricha Digital. Visit www.biricha.com (http://www.biricha.com/) for more information.

References

[1] Ridley, R.B., A new, continuous-time model for current-mode control [power convertors], Power Electronics, IEEE Transactions on (1991), pp. 271-280

[2] Biricha Digital Power Ltd. Digital Power Multi-Day Workshop (2010), pp. 168

[3] Ridley, R.B., A New Small-Signal Model for Current-Mode Control, Virginia Polytechnic Institute and State University, PhD. Thesis (1990)

[4] Cooke, P., Modeling average current mode control [of power convertors] (2000), pp. 256-262 vol.1

[5] Tang, W. and Lee, F.C. and Ridley, R.B., Small-signal modeling of average current-mode control (1992), pp. 747-755

[6] Hallworth, M. and Shirsavar, S.A., Microcontroller Based Peak Current Mode Control Using Digital Slope Compensation, Power Electronics, IEEE Transactions on (2012)

Downloads

Click here to download the article and appendix in PDF format. (http://www.biricha.com/download_resource/id/22)

Important Notice

All material presented in this application note, associated software, exercises, appendices and any oral presentations made by or on behalf of Biricha Digital Power Ltd are for educational purposes only, are not intended for production or commercial applications and may contain irregularities and defects or errors. Furthermore, none of the material has been tested or certified by any government agency or industry regulatory organization or any other third party organization.

Biricha Digital Power Ltd makes no warranties or representations, express, implied or statutory, including any implied warranties of merchantability, fitness for a particular purpose, lack of viruses, lack of patent infringements, accuracy or completeness of responses and lack of

negligence. Prior to using, incorporating or distributing any of the material that has been included in this application note in any commercial product, you must thoroughly test the product and the functionality in that product and be solely responsible for any problems or failures.

In no event shall Biricha Digital Power Ltd, or any applicable licensor, be liable for any special, indirect, incidental, or punitive damages, however caused, arising in any way out of your use of the material presented in the DPS Workshop. These excluded damages include, but are not limited to, cost of removal or reinstallation, procurement of substitute goods or services, outside computer time, labour costs, loss of data, loss of goodwill, loss of profits, loss of savings, loss of use or interruption of business or other economic loss.