Application note

STM32 digital power PID and IIR filters for SMPS control. Design and comparison on B-G414E-DPOW1 discovery kit

Introduction

This document describes the capabilities of digital switch-mode power supplies controlled by STM32 microcontrollers. It shows the empirical design of PI controllers versus the frequency design of IIR filters. These can be implemented easily on STM32-based systems, by using the ST-WDS® tool.

Measured results are then taken from the buck converter projects on the STM32G474 discovery kit (B-G474E-DPOW1). This is a low-cost and easy-to-use development kit that allows quick evaluation and application development with STM32G4 Series microcontrollers.

This document also gives an overview of digital power-controller design methodologies for the STM32, and the advantages of IIR compensators over classic PI control.

For further details, see the documents listed in Table 1.

1 General information

This document is applied to all STM32 devices. All these products are Arm®-based microcontrollers.

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

1.1 Reference documents

Table 1. Document and website references

Reference		Description
[1]	AN5496	Buck voltage mode with the B-G474E-DPOW1 Discovery kit(1)
[2]	AN5497	Buck current mode with the B-G474E-DPOW1 Discovery kit(1)
[3]	Arm® Cortex -M4 Processor RM	http://developer.arm.com/documentation(2)
[4]	The buck converter	C. P. Basso, Switch-Mode Power Supplies, McGraw-Hill Education
[5]	Feedback and Control Loops	
[6]	Step-by-step digital power supply design with STM32	https://www.biricha.com/digital-power-webinar-stm32.html(2)
[7]	How to design a digital power supply using an STM32 microcontroller	http://st-videos.s3.amazonaws.com/BIRICHA%20-%20How%20to%20design%20a%20digital%20power%20supply.mp4(2)
[8]	ST-WDS®	http://www.biricha.com/st-wds(2)
[9]	Measuring the Plant Transfer Function of a Digitally Controlled Converter	https://www.biricha.com/technical-videos.html(2)

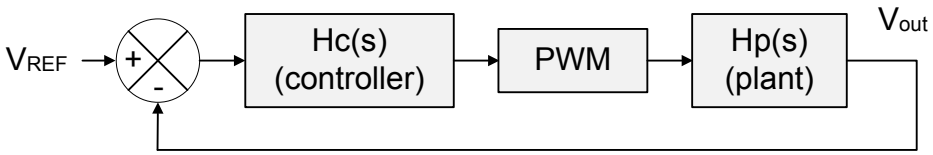
1. *Available in http://www.st.com*
2. *This URL belongs to a third party. It is active at document publication, however STMicroelectronics shall not be liable for any change, move or inactivation of the URL or the referenced material.*

2 Theoretical background

2.1 Switch-mode power supplies (SMPS) control

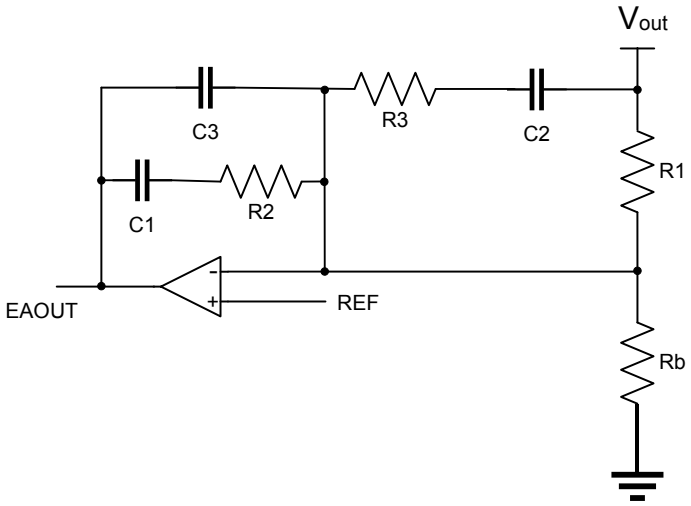
Traditionally, analog methods are used to control SMPSs. The "ON" time of the PWMs driving the switches is controlled by analog circuitry. This is typically an OPAMP with an RC compensation network around the negative feedback path. The Figure 1 represents a general schematic of an analog SMPS control loop. Here, the controller and plant transfer functions are represented in the Laplace domain (continuous domain).

Figure 1. Analog control loop - generic schematic



The Figure 2 shows a typical example of an analog compensator (presented here as $Hc_c(s)$).

Figure 2. Type-III analog compensator for SMPS control



This applies an IIR filter on the difference between reference voltage and output voltage, generating the control signal for the PWM module. Note that the frequency behavior of the closed-loop system is determined by the capacitance and resistance magnitudes of the design.

The analog control presents several challenges (see [7]):

- hardware based: product redesign is required to modify it
- bill of materials (BOM) may be consequent
- component drift/aging/tolerances directly impact control performance
- limited to classical control theory only
- no intelligent control over performance: adaptive control is not possible

These controllers have nevertheless been used in a wide range of applications. This is because analog compensation provides high bandwidth/resolution, which enhances power efficiency and output regulation.

With the STM32G474xx Arm® Cortex®-M4 microcontrollers, combining high integration and performance, high-bandwidth applications can now be targeted with digital control methods.

This addresses the above challenges, and also presents all the benefits of digital control (see [7]):

- can perform multiple loops and other functions
- reduces BOM
- reduces PCB area, thus increasing power density (W/inch^3)
- predictive maintenance algorithms.
- data logging
- software programmable: one hardware design for multiple topologies. This is enabled by the HRTIM1 high configurability
- advanced control becomes possible: nonlinear, multivariable

The STM32G474 also has features that accelerate the computation of control loops. This maximizes CPU bandwidth for the rest of the application; for instance, the FMAC filter computation accelerator, or the CCM-SRAM 0 wait-states access. The analog conversion performance is also optimized for greater bandwidth.

It is therefore a perfect platform for power-conversion applications with digital controllers. The following section covers the theory behind digital controller design, and easy to implement possibilities for such controllers on STM32 MCUs.

2.2 Digital controller design

Several approaches exist to design a digital controller. There is both the selection of the type of controller (PID, Type-III filter, Type-II filter, MPC, …), and the method to tune the parameters of such controller.

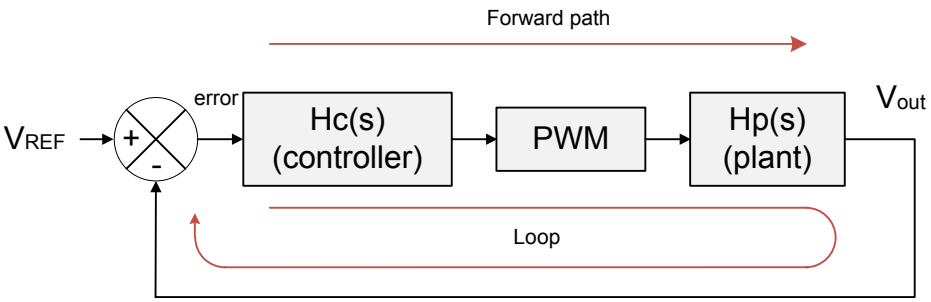
In digitally controlled systems, the most common technique is the heuristically tuned PID controller. However, this presents the following counterparts:

- PID design is based on temporal analysis. This does not give a precise control over the system stability. A frequency-based design provides better control of system stability.
- Heuristic tuning methods are valid for a specific hardware and the process of tuning every unit is not suitable for large productions. A model-based tuning could save effort and production time.

Generally, a finer control of transient and steady state responses can be achieved with frequency analysis than with time response analysis. In the context of power conversion applications, very tight constraints can be specified upon the reference tracking performance. It is therefore considered better to design SMPS control in the frequency domain (see [5], [7]).

A specification can be written in terms of frequency characteristics of the open loop transfer function (often called *forward path* in a controlled system).

Figure 3. Forward path transfer function in a voltage regulation system



2.2.1 Frequency-domain design process

Stability is achieved by guaranteeing the phase margin, crossover frequency, and gain margin as follows:

- Cross-over frequency: (i.e., the frequency at which the gain plot crosses 0 dB) must not exceed 1/20th of the sampling frequency ($F_x < F_s/20$)
  - The higher the F_x the faster the transient response in the time domain
- Phase margin: (i.e., the amount of phase above -180° at the cross-over frequency). This must be greater than 45° ($P_m > 45°$)
  - The lower the phase margin, the more oscillatory the transient response in the time domain. Note that increased response ringing could lead to system instability, and possibly to system damage.
- Gain margin: (i.e., the amount of gain below 0 dB when phase reaches -180°). This must be greater than 10 dB ($G_m > 10$ dB).
  - It is also representative of the level of rejection of high-frequency perturbation (such as load-stepping).
- Slope: at the crossover frequency this must be around -20 dB/decade @(F_x), to ensure system stability.

The process to design a digital controller in the frequency domain is as follows:

1. Model or measure the frequency response of the plant. Note that mathematical models of SMPS systems can be relatively complex. Most of them are at least second order.
2. Choose the best compensator for the system response. This is based on the impact that the *poles* and *zeroes* in the compensator's transfer function can have on the Bode diagram of the *forward path*.
3. Position the poles and zeroes of the controller to respect the criteria mentioned above.
4. Transform the values computed in the continuous domain, into the discrete domain.
5. Adapt the resulting values to the STM32 resources used to perform the control loop. For instance, numeric representation, ADC, and PWM scaling.
6. Simulate/measure the loop and check the resulting behavior. Iterate until the design objectives are met.

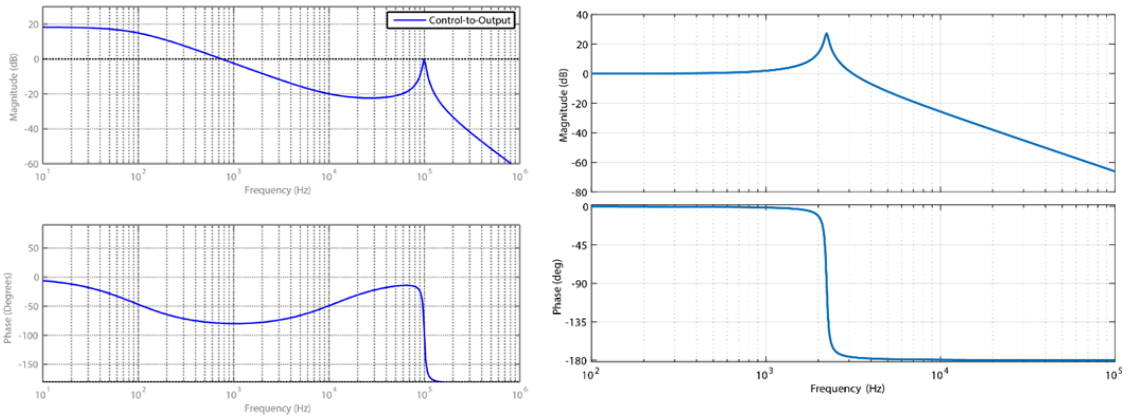
ST-WDS® is a digital power-supply design tool provided by Biricha. With it, these steps are easily performed on STM32 products, as is observed in Section 4.1 IIR compensator frequency design. Note that STM32 product families enable free use of this digital controller design software.

2.2.2 ST-WDS® tool description

Once the system response has been measured or modeled, different compensators can be proposed to obtain a proper forward path Bode plot. The controller that has the best compatibility to the measured plant can cancel the plant's behavior and it can shape the forward path plot as specified , thanks to its transfer function.

For instance, the figure below shows the typical Bode plot of the control-to-output transfer function of a buck converter under different control modes.

Figure 4. Bode plots of buck plant transfer functions



Simulations under peak current mode control

Simulations under voltage mode control

Note: The control mode has a clear impact on the system behavior.

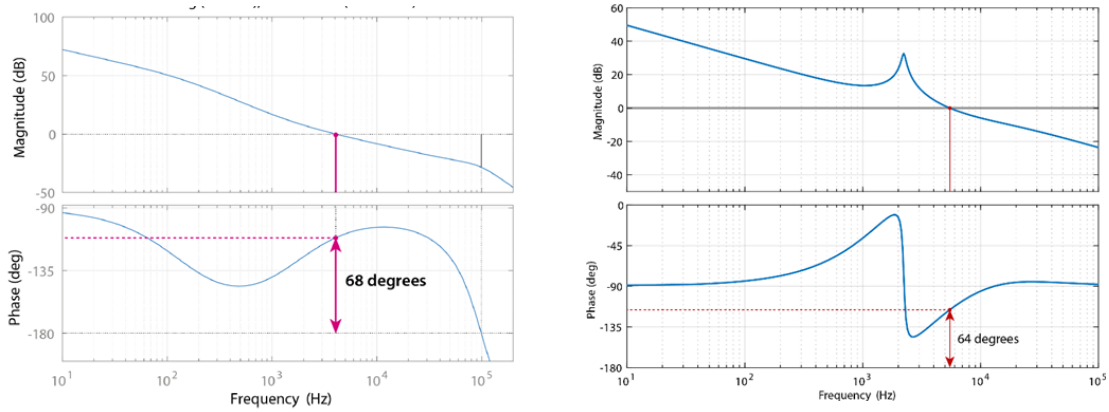
Similarly, a different SMPS topology (for instance boost) has a different frequency behaviour.

On the right side (voltage mode), a double pole can be observed. The controller for voltage mode needs to have two zeroes to compensate for such behaviour, and an additional 'zero' to shape the forward path once the pole pair of the plant has been canceled down.

On the left side (peak current mode), the inductor's current is controlled on a cycle-by-cycle basis, and therefore the inductor acts as a constant current source. This means that the plant power stage no longer contains the double pole present when operating under voltage mode control. A different compensator thus is used for this control mode.

Therefore, based on the system behaviour, an adequate type of compensator needs to be selected, and the compensator needs to be parameterized in such a way that the above mentioned frequency specification is respected. The figure below shows the simulation resulting from the design of compensators for the buck plant.

Figure 5. Open-loop Bode plot: plant, PWM and compensator



Simulations under peak current mode control

Simulations under voltage mode control

There are typically two types of compensators, which are used for stabilizing power supplies, given the poles and zeroes present in their transfer functions and the behavioral model of most of the SMPS topologies. These are universally referred to as the Type-II and Type-III compensators, which consist of IIR filters.

For instance, the z-domain (discrete) representation of a Type-III compensator is the following:

Figure 6. Equation 1: Type-III compensator represented as a z-domain transfer function

$$H_C[z] = \frac{B_3 z^{-3} + B_2 z^{-2} + B_1 z^{-1} + B_0}{-A_3 z^{-3} - A_2 z^{-2} - A_1 z^{-1} + 1} \quad (1)$$

This can be converted into a linear differential equation, which can be easily implemented on the MCU using discrete samples of the controller input and output signals:

Figure 7. Equation 2: Type-III compensator represented as a differential equation

$$y[z] = A_1 y[n-1] + A_2 y[n-2] + A_3 y[n-3] + B_0 x[n] + B_1 x[n-1] + B_2 x[n-2] + B_3 x[n-3] \quad (2)$$

Where *y[*]* are the samples of the controller output after an execution of the algorithm. For instance, the new duty cycle value in voltage mode, or the new current limit in peak current mode.

Here, *x[*]* represent the samples of the controller input. In the case of an SMPS, it is the difference between the output voltage and the tracked reference, scaled to the output values of the ADC.

ST-WDS® contains the mathematical models of buck plants (both voltage and peak current mode-controlled), and the models of the compensators.

By entering the characteristics of the hardware design, the tool automatically selects the appropriate compensator and computes the coefficient values with the right formatting according to the hardware resources of STM32G4 used to implement the loop.

Note that ST-WDS® also permits to represent measured frequency behaviours, as well as fine-tuning the automatically computed parameters to adapt the compensator to the real system. Therefore, even when the mathematical model of the topology is not well known, ST-WDS®can be used to represent a measured plant and to perform the poles/zeros placement manually.

On STM32G4 platforms, the IIR filter can be directly implemented on the FMAC, thus reducing the CPU loading for the digital power application.

Alternatively, the digital approach can implement any type of calculations on the CPU:

- PID controllers
- non-linear control techniques (for instance, anti-windup mechanisms)
- model predictive control

2.2.3 PID usage

PID controllers have not generally been widely used in analog SMPS control for several reasons:

- they require a greater BOM and PCB area
- their design is most of the time based on the time domain response

For these reasons, they give inferior stability control to that obtained with the frequency design of Type-II and Type-III compensators.

Nevertheless, proven heuristic methods exist for the tuning of a PID controller (for instance, Ziegler-Nichols).

The digital approach also permits easier tuning of the PID. This is the case whether a model of the target system, or a heuristic approach, is used.

A reminder of the basic theory behind PID control, and a proposal for an empirical tuning method are given in the following sections.

PID operating principle

The proportional component of the PID regulator ensures fast reaction, while the integral component gives speed regulation and error compensation. (The steady-state error can only be nulled with an integral effect.)

In practice, the differential correction term increases the regulation noise in the event of high frequency disturbance (high-pass function). The system may thus become unstable or difficult to tune. Unless the transient response is unsatisfactory after K_p and K_i tuning, introducing K_d is not recommended.

The regulation loop adjusts either the current limit (in peak current mode), or the PWM duty cycle (in voltage mode). The command, (u), is dependent on the time evolution of the difference between the output voltage and the desired reference, (e).

Figure 8. Equation 3: PI controller digital implementation (discrete equation)

$$u[n] = u_{constant} + K_p \cdot e[n] + K_I * \sum e[n] \quad (3)$$

Here, $u_{constant}$ corresponds to the initial conditions of the command. This is null for a digital SMPS because the initial current limit (in peak current mode), or duty cycle (in voltage mode), output by the controller is null.

Note: For simplicity, nonlinear saturations to min./max. values are not considered.

This constant term is nevertheless used to tune the K_p term around the expected operating point of the system. This is explained in the following section.

Tuning procedure

To tune the PI regulator parameters, it is advisable to load the power stage at 50% of the maximum load that it feeds. Once this is done, the following steps can be performed:

1. Identify u_{steady} value: this is the command value that the controller outputs to stabilize the system at the specified reference output voltage.

   In the case of a peak current mode controller, it is a value between 0 and 4096 written into the DAC, while in voltage mode it is a duty cycle value between 0 and the number of ticks configured as period value on the HRTIM channel.

   This value can be found by setting the system in open loop operation and slowly increasing the command value until the output-voltage reference is reached.

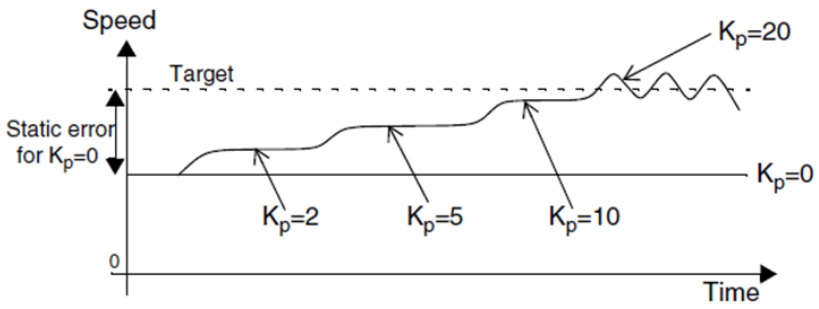
   For this stage, the controller function shall output a constant configurable value, regardless of the difference between the output voltage and the target reference.

Figure 9. Equation 4: Output of the PI controller to identify the steady-state term before K_p tuning

$$u[n] = u_{steady} \quad (4)$$

2. Identify the limit K_P value. With the u_{steady} value implemented, set the system in closed loop operation. Then slowly increase the K_P value until consistent oscillations are clearly observed in the output voltage.

Figure 10. Heuristic tuning of k_p based on steady-state temporal behavior



In this case, the controller should add the proportional effect to the command.

Figure 11. Equation 5: Output of the PI controller for K_p tuning

$$u[n] = u_{steady} + K_p * e[n] \quad (5)$$

3. Fine-tune *K* value: halve the obtained K_P and check if there are any remaining oscillations. If so, keep lowering the K_P value until no oscillations are observed.
4. Approximate the K_I value: once the K_P value has been identified, suppress the u_{steady} term and observe the voltage on 100% to 50% load steps. Then slowly increase K_I until the steady state error is completely suppressed.

   The controller should now perform the following operation:

Figure 12. Equation 6: Output of the PI controller after K_p tuning

$$u[n] = K_p * e[n] + K_1 * \sum e \quad (6)$$

5. Fine-tune K_I: observe the transient response of 100% to 50% load steps. Adjust K_I to find the adequate overshoot/ringing/rise time trade-off.

In any case, this process may give different results depending on the initial state (for instance, output capacitor charge), and its component conditions. It is also very subjective and imprecise. The following section gives a mathematical frequency analysis of the potential PID controller behavior in digital SMPS applications.

Frequency analysis of the PID

The PID controller can also be represented as a filter with a pole at the origin. This could be used to perform frequency design. However, there would no longer be direct parameterization of proportional, integral, and derivative effects, as these are time-domain concepts. This can also be observed in the Laplace transfer function of a PID controller:

Figure 13. Equation 7: Laplace domain representation of a PID controller

$$H_C[s] = \frac{K_D s^2 + K_P s + K_i}{s} = \frac{\left(s + \frac{-K_P + \sqrt{K_P^2 - 4\,K_D K_i}}{2K_i}\right) + \left(s - \frac{-K_P + \sqrt{K_P^2 - 4\,K_D K_i}}{2K_i}\right)}{s} \quad (7)$$

This frequency analysis indicates that PID controllers can barely reach the performance of Type-II and Type-III filters in SMPS applications. Such systems have a resonant peak and phase-boosting zeroes (see Figure 4) around which the gain plot should not cross 0 dB, to avoid instability. This is hard to do just with K_p, K_i, and K_d tuning; although without K_d it is possible.

Therefore, the risk of instability can be high with PID controllers. As the forward-path behavior is hardware-dependent, this is not a recommended characteristic for a product subject to component drift/aging/tolerances. Practically, PID tuning is sub-optimal to accommodate unit-to-unit manufacturing.

3 Buck plant control examples

The results are evaluated with the buck-control example projects, provided for the B-G474E-DPOW1 board. These are accessible via the STM32CubeMX example selector.

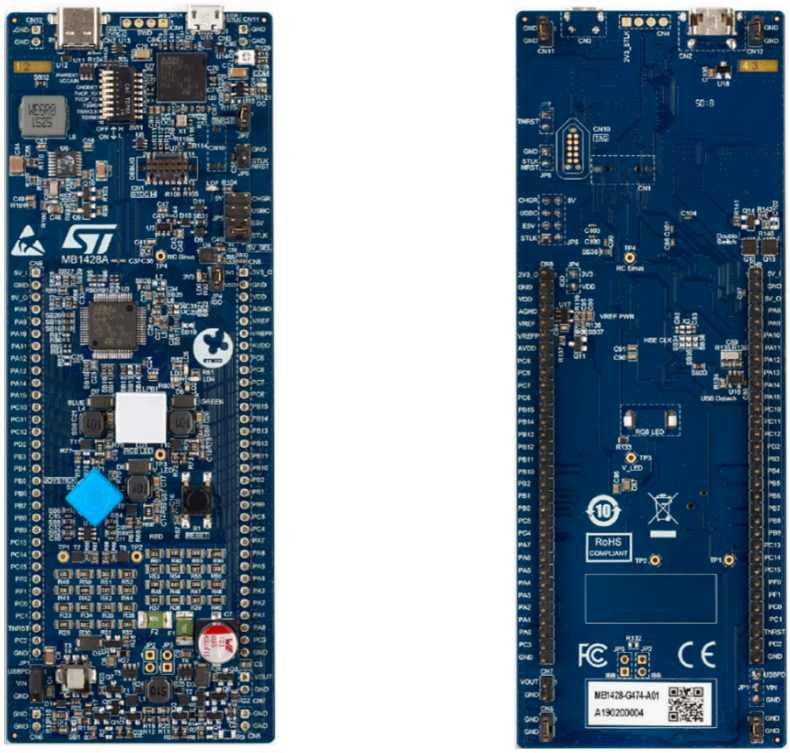
Note: Detailed descriptions of the plant mathematical-modeling, MCU peripheral configuration, and PCB design process can be found in [1] and [2].

3.1 Hardware overview

The STM32G474 discovery kit is a complete digital power starter kit controlled by the STMicroelectronics Arm® Cortex®-M4 core-based STM32G474RET6 microcontroller. The kit shows the features of digital power including:

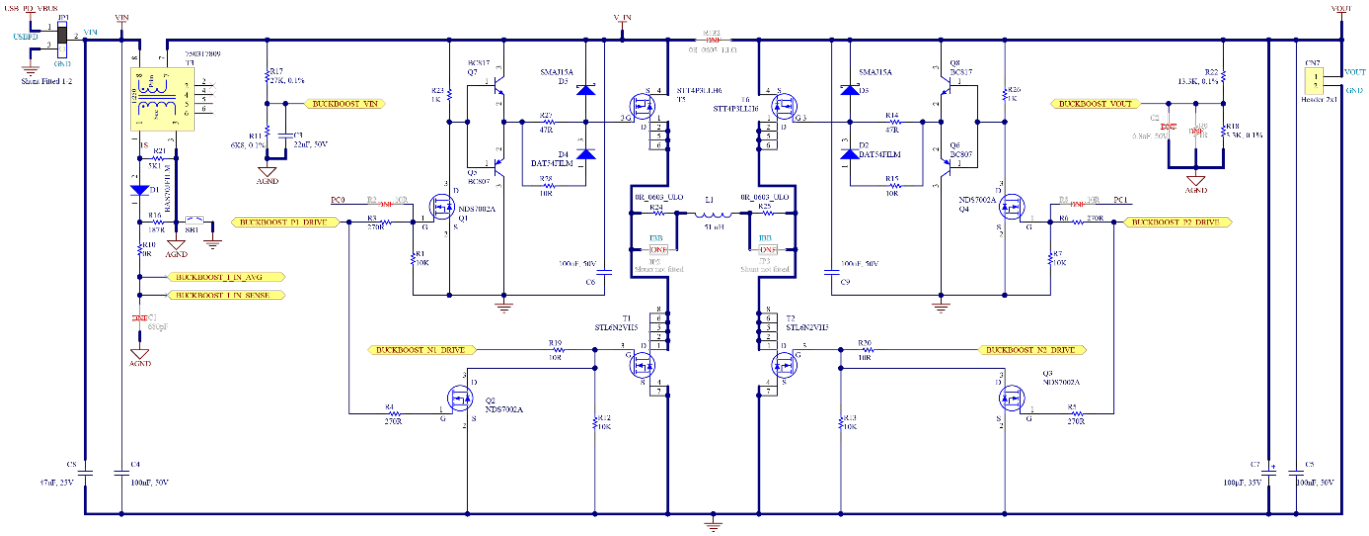
- LED dimming
- Buck/Boost with variable load
- Power delivery (USB type-CTM)
- Audio Class-D amplification

Figure 14. STM32 discovery kit top and bottom sides



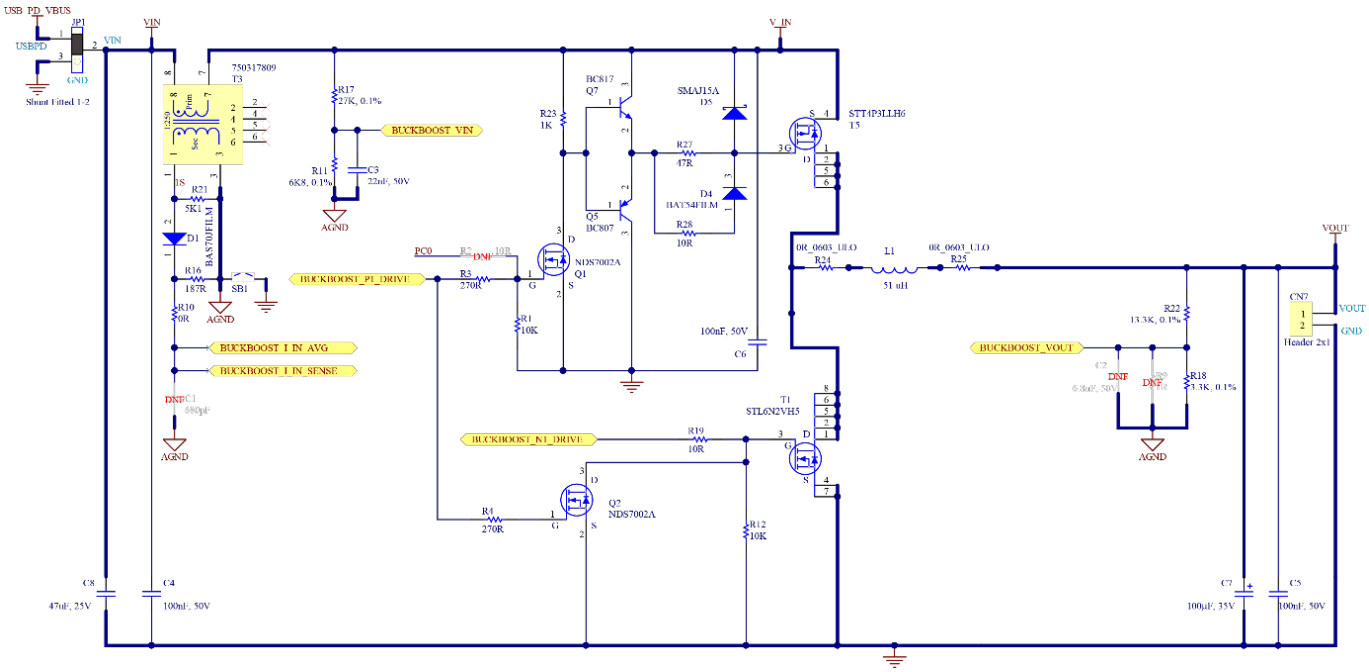
This discovery kit embeds a buck-boost converter. There are both boost and buck switching FETs at each side of the converter inductor. The extract from the schematic shown in Figure 15 identifies the relevant switches. The full schematic can be downloaded from http://www.st.com.

Figure 15. Discovery kit schematic – buck-boost power stage



When using the converter in buck mode the boost switches are not driven (the high side switch is on, the low side switch is off) and the simplified circuit shown in the Figure 16 can be used to describe the power stage.

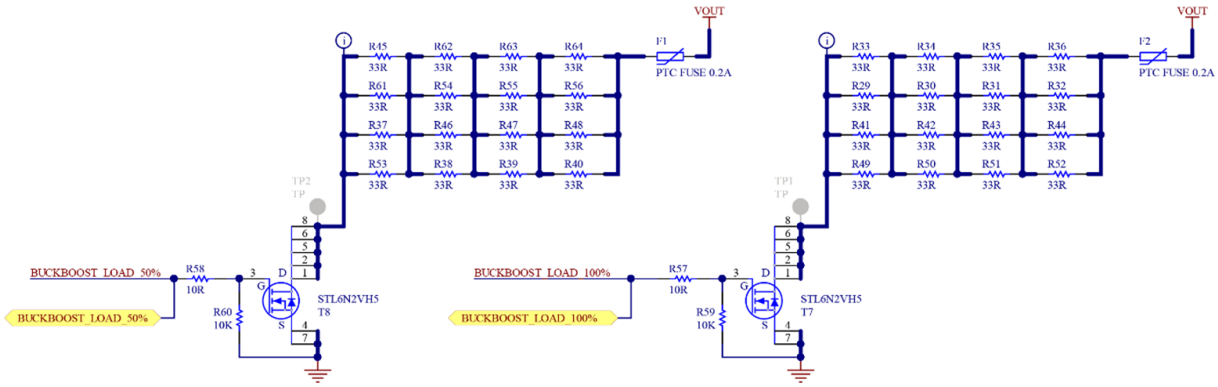
Figure 16. Simplified power stage of the buck converter



3.2 On-board load

This board includes two parallel load banks at the output of the converter. This is shown in Figure 17.

Figure 17. On-board load banks controlled via MOSFETs



The MCU can control the load banks by toggling the PC14 and PC15 outputs. These drive the BUCKBOOST_LOAD_50% and BUCKBOOST_LOAD_100% signals respectively. The MOSFET driven by each of these signals, switches the resistive load bank in-and-out of the circuit.

Therefore, when PC14 is high, load bank 1 is on. When PC15 is high, load bank 2 is on. Test pins TP1 and TP0 may also be used to check for the load activation status, although these are not populated. The load banks have the total resistances shown in Table 2.

Table 2. On board load steps

Load (%)	0% Load 1 OFF Load 2 OFF	50% Load 1 ON Load 2 OFF	100% Load 1 ON Load 2 ON
Load (Ω)	∞	33	16.5
Led status	All off	Green	Green + orange

Note: The load status can also be observed in the example projects via the toggling green and orange LEDs. It can be changed by using the joystick on the board.

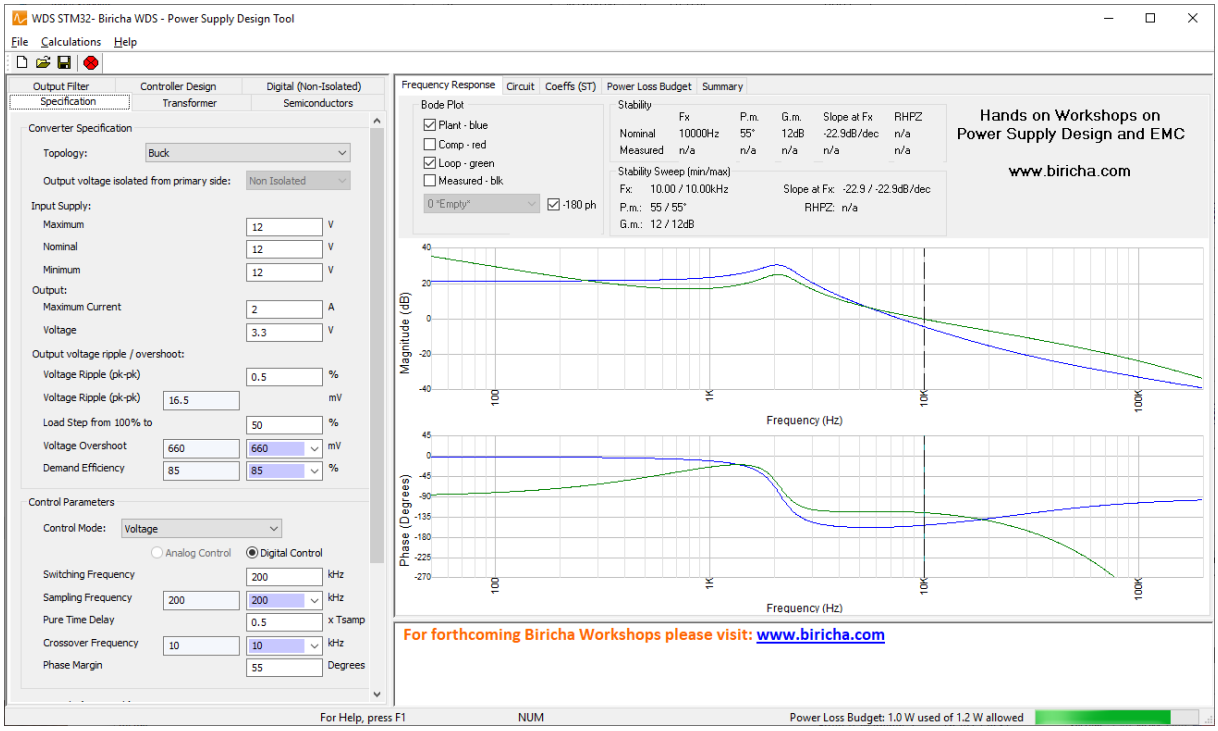
4 Controller design and results on STM32G4 discovery kit

4.1 IIR compensator frequency design

The power-supply design tool, ST-WDS® from Biricha, generates the coefficients needed to implement this controller on an STM32 platform. ST-WDS® is free-to-use, and can be downloaded for free from the Biricha website. Visit [8] for more information.

Note: The .wds file associated for the buck control examples is included in each of the project root directories.

Figure 18. Biricha ST-WDS®



4.1.1 Inputs to ST-WDS®

The initial screen of ST-WDS® is shown in Figure 18.

The left-hand pane is used for entering the specification of the power supply along with the other pertinent design parameters.

The right-hand pane displays the control loop bode plots, a generic schematic for the selected plant, and the outputs required for digital controller coefficients.

To calculate the digital controller coefficients, the power supply specification must first be entered. In the “Specification” tab, enter the specification shown in Table 3.

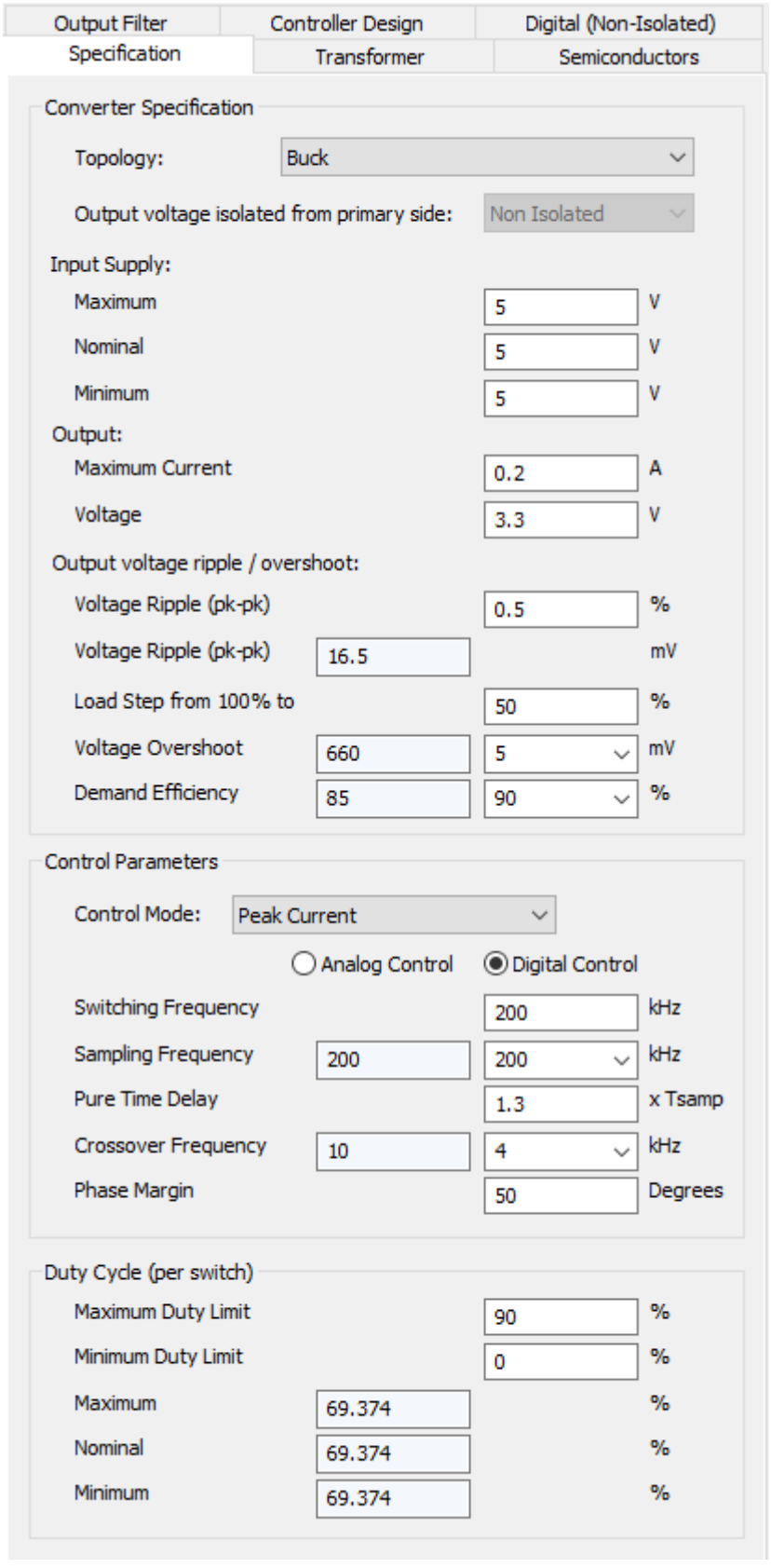
Table 3. Discovery kit specification

Specification tab parameter	Value
Topology	Buck
Input supply max	5 V
Input supply nom	5 V
Input supply min	5 V
Output maximum Current	0.2 A
Output voltage	3.3 V
Voltage ripple (peak-to-peak)	0.5%
Load step 100% to…	50%
Voltage overshoot	5 mV
Demand efficiency	92%
Control mode	Voltage, digital control/peak current, digital control
Switching frequency	200 kHz
Pure time delay	1
Crossover frequency	8 kHz (voltage mode) / 4 kHz (peak current mode)
Phase margin	50 degrees
Maximum duty limit	90%
Minimum duty limit	0%

Note: The control mode is dependent on whether the user is designing a controller for the voltage, or the peak current, mode control.

Figure 19 shows the ST-WDS® specification tab in case of peak current mode control.

Figure 19. **ST-WDS® specification tab for peak current mode control**



The “Transformer” tab is not used, as this is a buck converter. This tab is only applicable for topologies that include a power stage transformer. Next, click on the “Semiconductors” tab and enter the characteristics of the target hardware, as shown in Table 4.

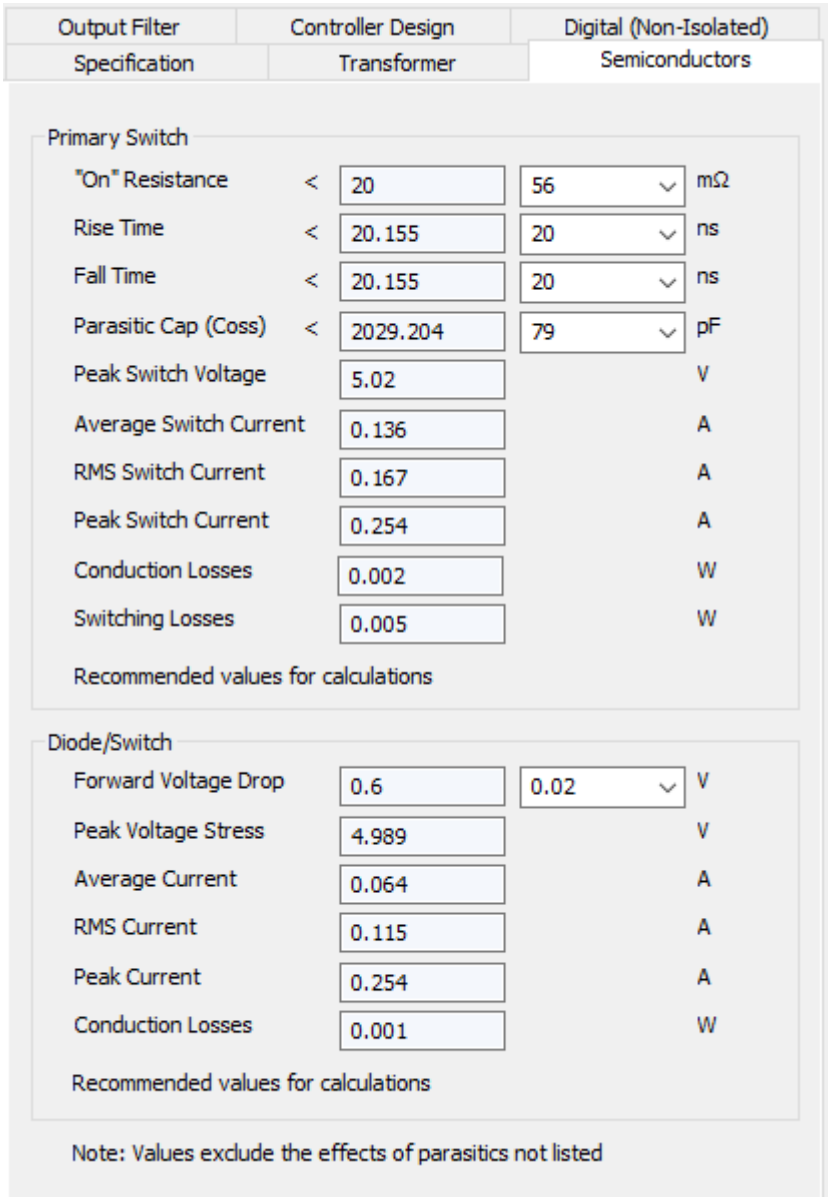
Table 4. Semiconductor parameters

Semiconductors tab parameter	Value
“On” resistance	56 mΩ
Rise time	20 ns
Fall time	20 ns
Parasitic Cap (Coss)	79 pF
Forward voltage drop	0.02 V

Note: In this case, forward voltage drop is particularly low because the buck converter is synchronous (the diode is replaced by second transistor).

The “Semiconductors” tab should now look like the screenshot below, regardless of the control mode.

Figure 20. **ST-WDS® specification tab for buck plant**



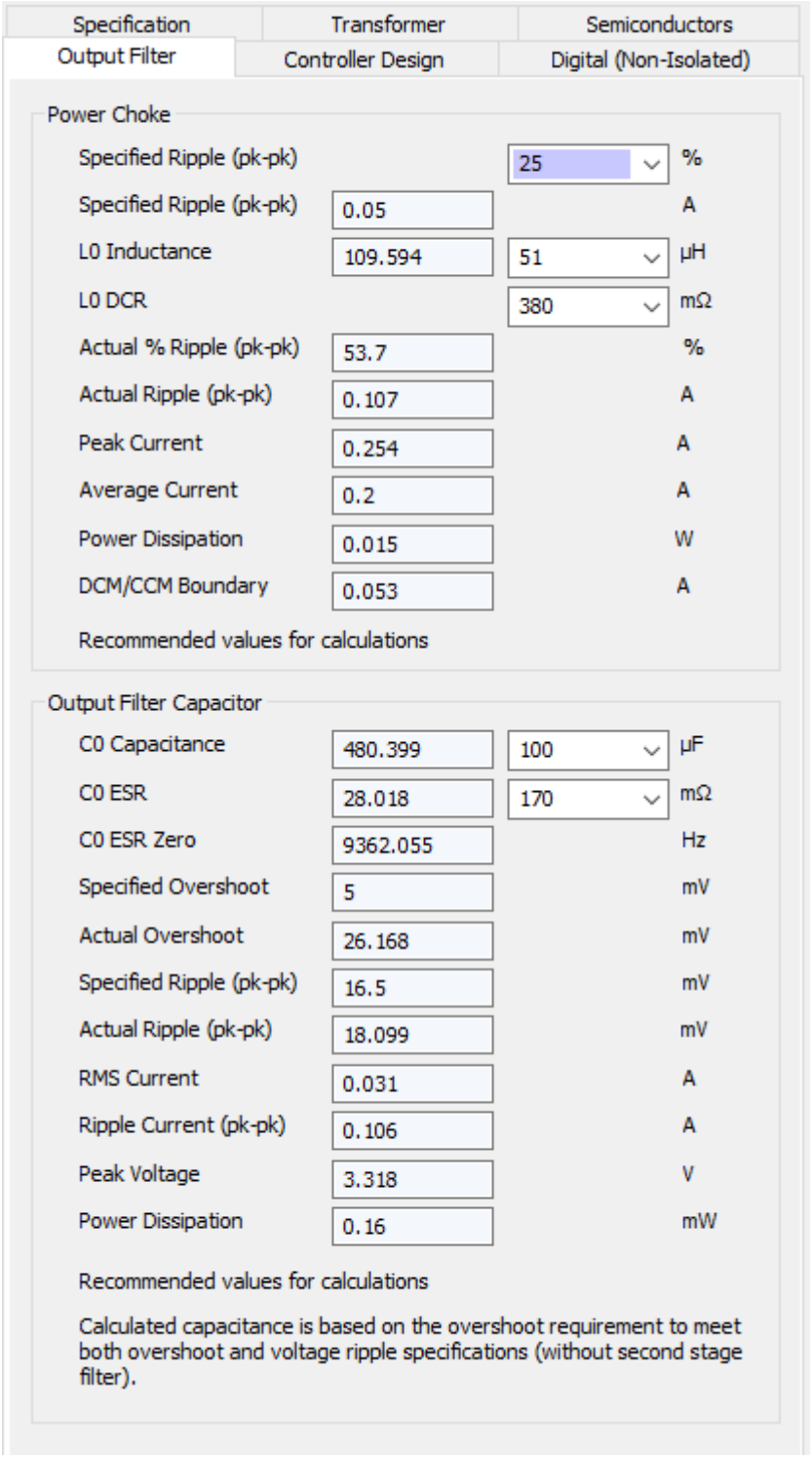
The “Output Filter” tab has the following characteristics.

Table 5. **Output-filter parameters**

Output-filter tab parameter	Value
Specified ripple (pk-pk)	25%
L0 inductance	51 µH
L0 DCR	380 mΩ
C0 capacitance	100 µF
C0 ESR	170 mΩ

The “Output Filter” tab is shown in Figure 21.

Figure 21. **ST-WDS® output filter tab**



In the “Controller Design” tab, ST-WDS® automatically selects the type of compensator, based on the specified plant and control modes. For instance, it selects a Type-III compensator for the voltage mode control, or a Type-II compensator for the peak current mode controller.

The tool also places the poles/zeros of the selected compensator so as to meet the crossover frequency and phase margin specifications, among other criteria. Thus, there is no need to enter any parameters into this tab.

The contents of the tab is the following for each control mode:

Figure 22. **ST-WDS® controller design tab for voltage-mode buck control**

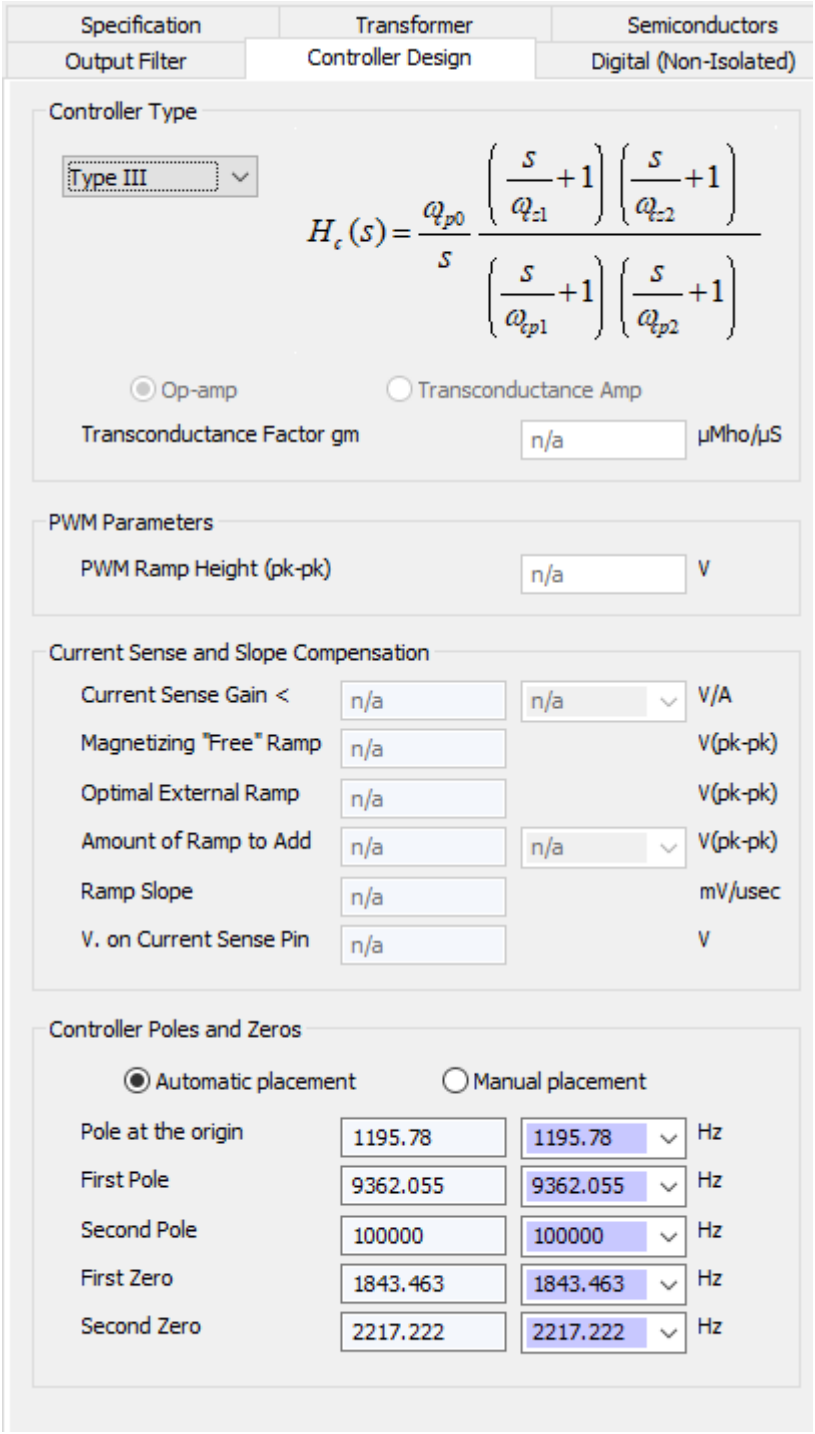
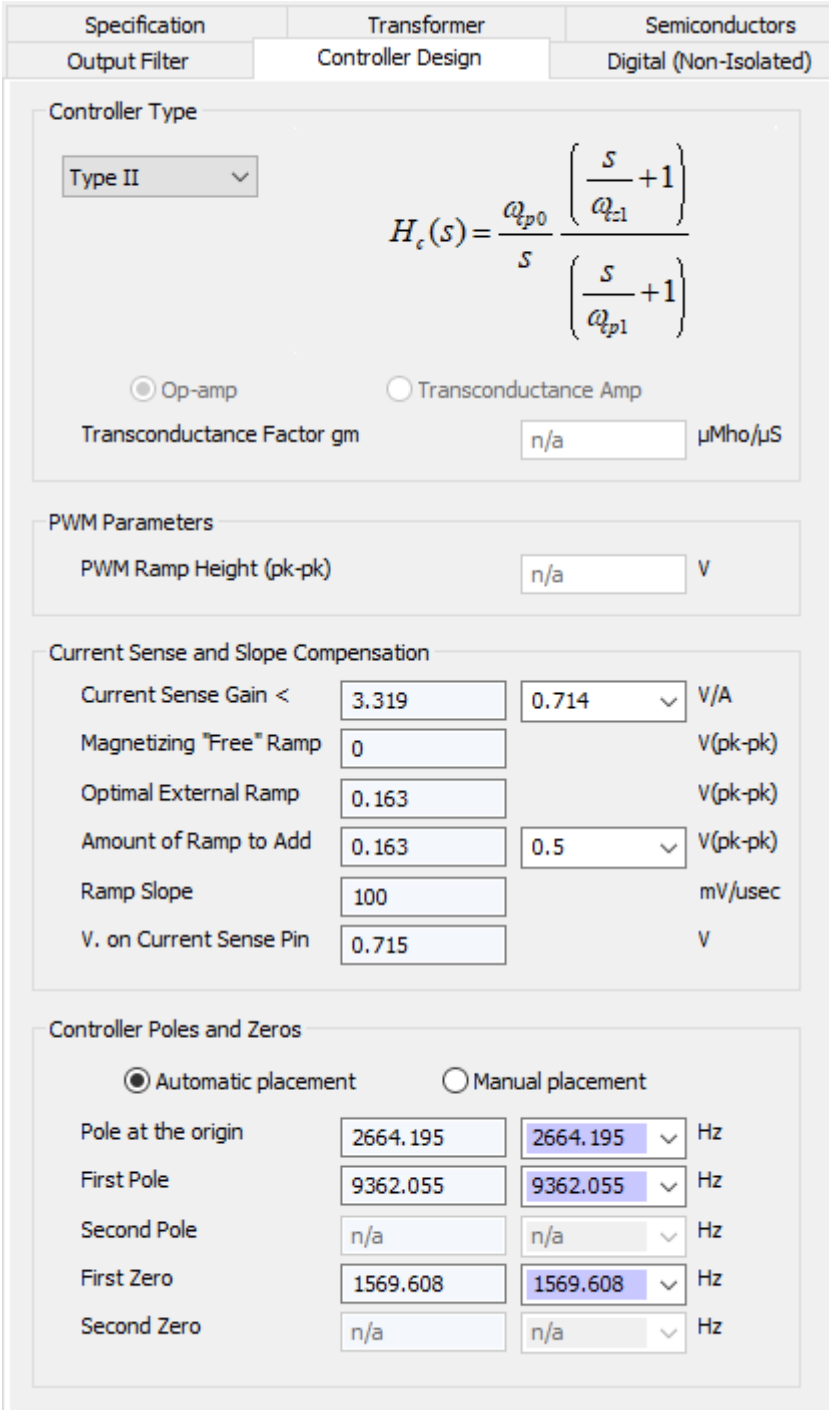


Figure 23. **ST-WDS® controller design tab for peak current mode buck control**



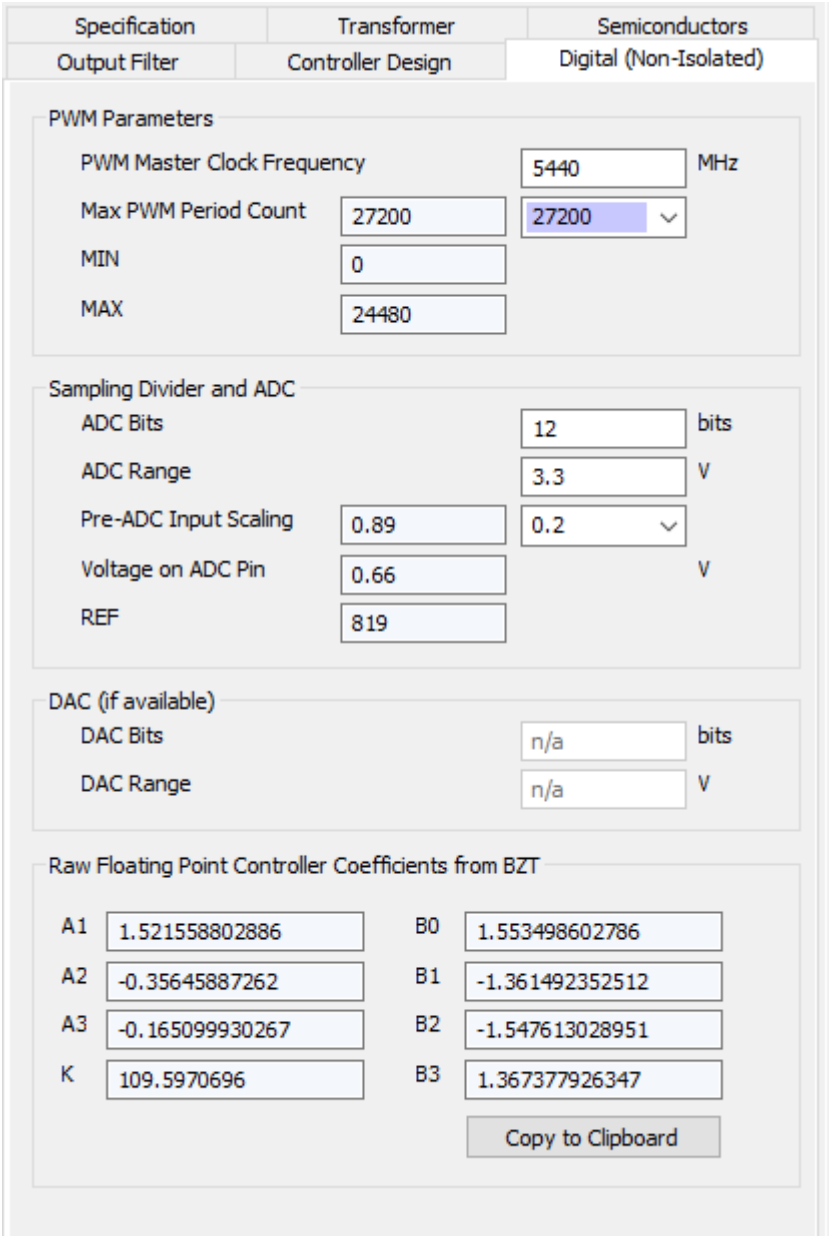
The last configuration tab is “Digital (Non-Isolated)”, its specifications are listed below.

Table 6. **Digital (non-isolated) parameter**

Digital (non-isolated) tab parameter	Value
PWM master clock frequency	5440 MHz
Max PWM period count	27200
ADC bits	12

Digital (non-isolated) tab parameter	Value
ADC range	3.3 V
Pre-ADC input scaling	0.2

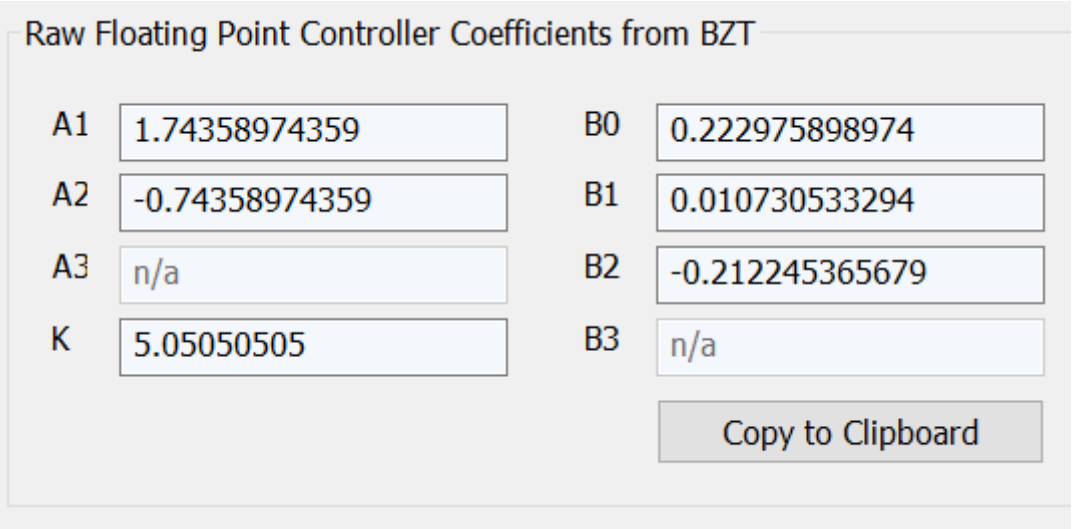
Figure 24. ST-WDS® digital (non-isolated) tab for voltage-mode buck control



Note: The "Raw Floating Point Controller Coefficients" in this tab are dependent on the compensator type, and therefore on the control mode.

The following screenshot depicts its contents for peak current mode control.

Figure 25. ST-WDS® “Raw Floating Point Coefficients” for peak current mode buck control



4.1.2 Coefficients for execution in the STM32G4 CPU

STM32G4 microcontrollers are based on an Arm® Cortex®-M4 core, which contains a floating-point unit. Thus, the user can already implement the IIR filter by copying the resulting coefficients from the tab exposed above and applying 3. This can be observed in the *Buck_CurrentMode_SW* and *Buck_VoltageMode_SW* projects.

The resulting value is casted into an integer as it is entered either to the DAC (peak current mode control) or to the HRTIM1 compare registers (voltage mode control). The new value is configured in the DAC/HRTIM as the CPU exits the above function, as can be seen in the ADC IRQ handler located in *stm32g4xx_it.c*.

A 3p3z compensator is identical to a 2p2z when its A3 and B3 coefficients are null. In any case, the application codes, provided for the buck plant, are such that just by copying the coefficients computed by ST-WDS® into *main.h*, the obtained compensator is applied.

For instance, the following is copied when clicking “Copy to Clipboard” button at the bottom of the “Digital (non-isolated)” tab for the peak current mode controller:

```
#define B0 (+0.222975898974)
#define B1 (+0.010730533294)
#define B2 (-0.212245365679)
#define A1 (+1.743589743590)
#define A2 (-0.743589743590)
#define K (+5.050505050505)
#define REF (811)
#define DUTY_TICKS_MIN (0)
#define DUTY_TICKS_MAX (3686)
#define SLOPE_VPP (0.5000)
#define DECVAL (0.3650)
```

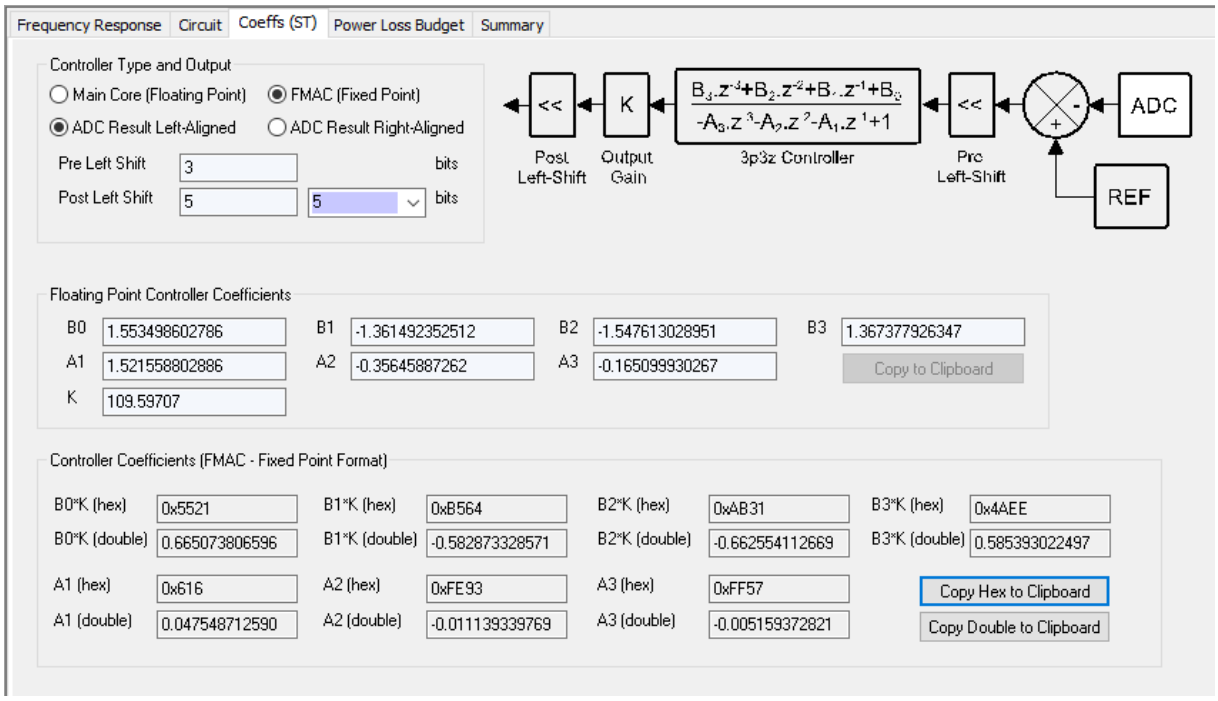
This way, even for a different hardware, the users can enter their own specifications in ST-WDS® but still use this application code to apply a controller on their buck plant.

4.1.3 Coefficients for execution in the STM32G4 FMAC

With STM32G474, IIR filters can also be computed with the fixed-point FMAC accelerator, thus minimizing the CPU-loading caused by the execution of this control loop. The ST-WDS® also offers the possibility to convert the coefficients to the formatting required by FMAC, given a configured ADC conversion format.

To do this, click on the “Coeffs (ST)” tab on the right-hand pane. Here, set the “Controller Type and Output” setting to “FMAC (Fixed Point)”. In these examples, the ADC result was configured to be left-aligned. Therefore, click on the “ADC Result Left-Aligned” radio button within ST-WDS®. The pane looks as in for the voltage mode controller.

Figure 26. Fixed point controller coefficient calculation in ST-WDS®



The fixed-point coefficients is displayed at the bottom of the window. It is possible to copy these to the clipboard for use within our code. Click the “Copy Hex to Clipboard” button. The following coefficients are copied in the case of the voltage mode control:

```
#define B0 (0x5521)
#define B1 (0xB564)
#define B2 (0xAB31)
#define B3 (0x4AEE)
#define A1 (0x616)
#define A2 (0xFE93)
#define A3 (0xFF57)
#define pre_shift (+3)
#define post_shift (+5)
#define REF (819)
#define DUTY_TICKS_MIN (0)
#define DUTY_TICKS_MAX (24480)
```

The FMAC is then configured in *main.c* to execute the filter with the adequate numeric formatting and the above coefficients, as can be seen in the *main.c* file of the *Buck_CurrentMode_HW* and *Buck_VoltageMode_HW* projects.

Again, this application code is such that the user can enter their own hardware specification into ST-WDS®, extract the computed coefficient values and execute the controller on a custom buck plant.

4.2 PI heuristic design

Alternatively, there is also the PI heuristic design technique. Note that this needs less system parameters. It consists of a try-and-error method until the controller specification is met as best as possible. Moreover, the controller tunning needs to be performed for every hardware, as opposed to the automated coefficients calculation of ST-WDS®.

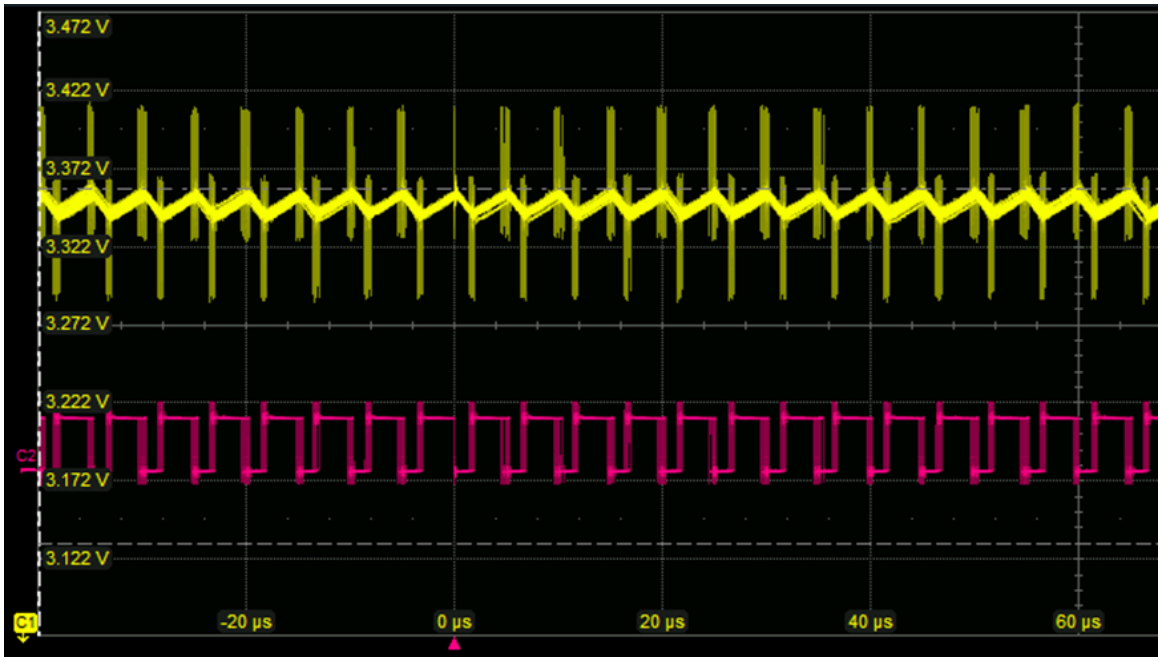
Below is exposed the observed process for the PI tunning on peak current mode and voltage mode controllers. As mentioned earlier, the differential term is not included as it increases the regulation noise in the event of high frequency disturbance (high-pass function).

4.2.1 Peak current mode PI tuning

1. Identify u_{steady} value: in the case of peak current mode control, the output voltage at 50% stabilizes at 3.3 V when the output of the controller is set to u_{steady} = 512.

2. Identify the limit K_P value: the oscilloscope trace below shows the observed behaviour for K_p=30 (normal mode acquisition triggered on the output PWM rising edge):

Figure 27. Steady-state oscillations of peak current mode control with K_p=30

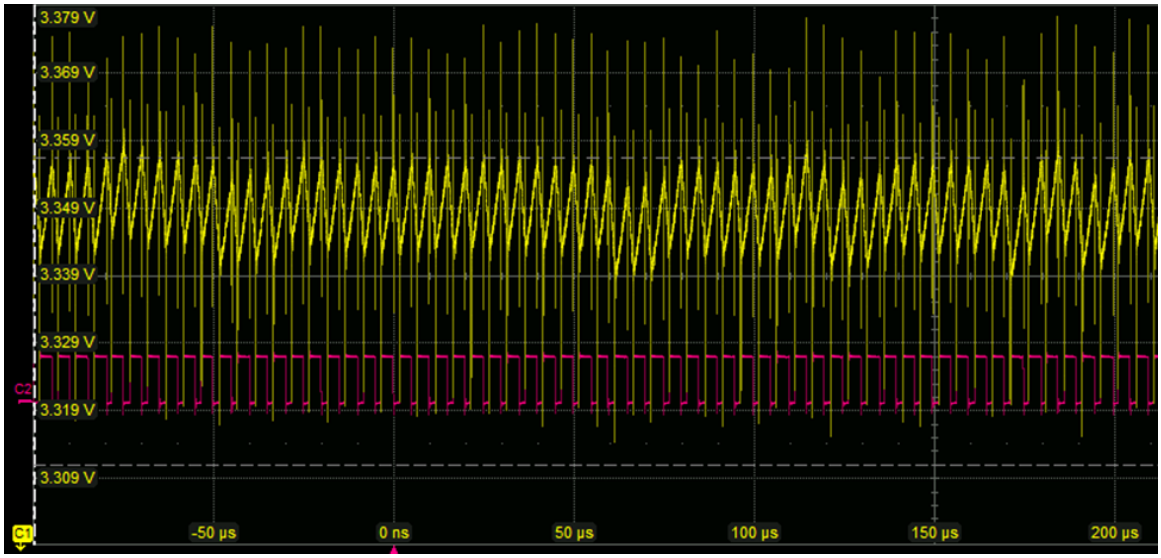


Here, the yellow curve corresponds to the output voltage and the pink to the PWM driving the top switch of the buck plant. Clearly the DAC value computed by the controller oscillates, as can be deduced by the variance of the PWM's duty cycle, and so does the output voltage.

Then the user reduces the K_p value until the oscillations are less perceptible. The single-shot acquisition can be used to get a finer view of the presence of oscillations.

Below is a single shot triggered on the PWM rising edge with K_p = 7.

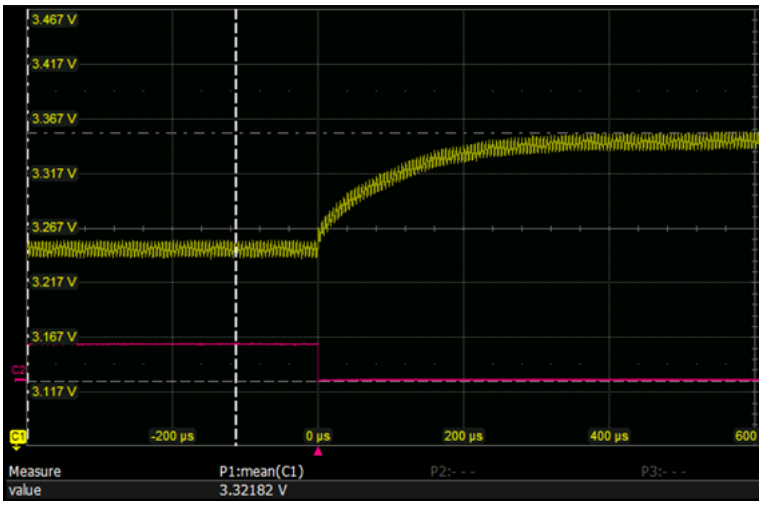
Figure 28. Steady-state oscillations of peak current mode control with Kp =7



Oscillations are not much attenuated below this K_p value; user therefore takes K_p = 7 as the limit for the proportional effect.

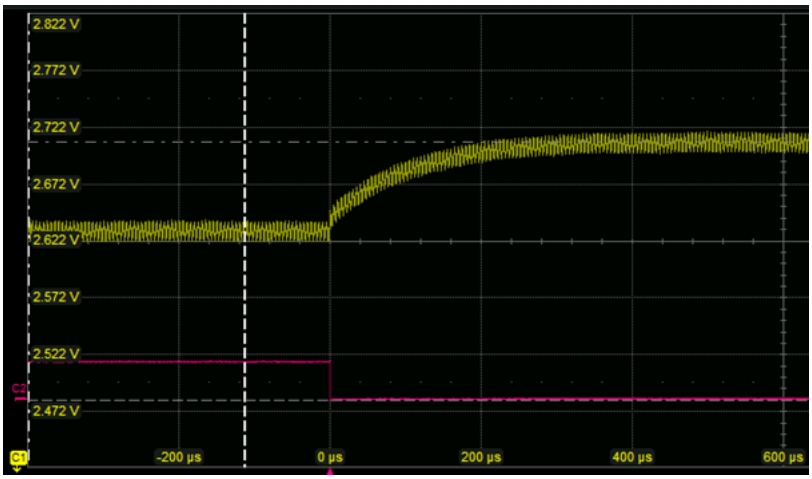
3. Fine-tune K_P value: if user now halves the value of K_P to 3.5 and pay closer attention to the oscillations, some can still be observed from time to time. By reducing it to three the oscillations completely disappear. Note that, only with this K_p value (without integral effect), a 100% to 50% load clearly does not track the 3.3 V reference

Figure 29. 100% to 50% load-step on peak current load with final K_p value and steady state term, without integral effect



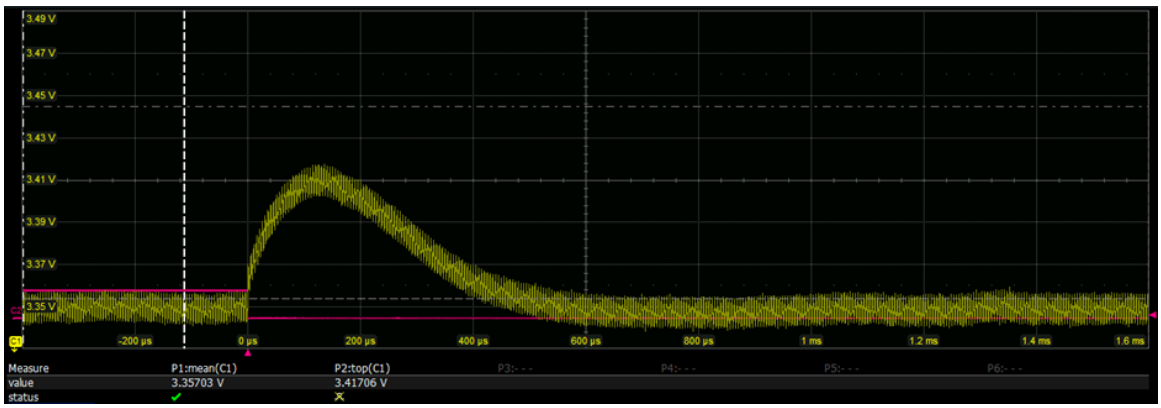
Here, the pink trace corresponds to a signal commanding the switches that enables/disables 100% load. Note that when the constant term is suppressed (which is needed to tune the K_i value), the output voltage is lower and therefore further from the 3.3 V reference.

Figure 30. 100% to 50% load-step on peak current load with final Kp value, without integral effect



4. Approximate the K_I value: by configuring a small increase of K_i the steady state regulation can already be observed. Below the resulting load-step transient with K_i=0.1.

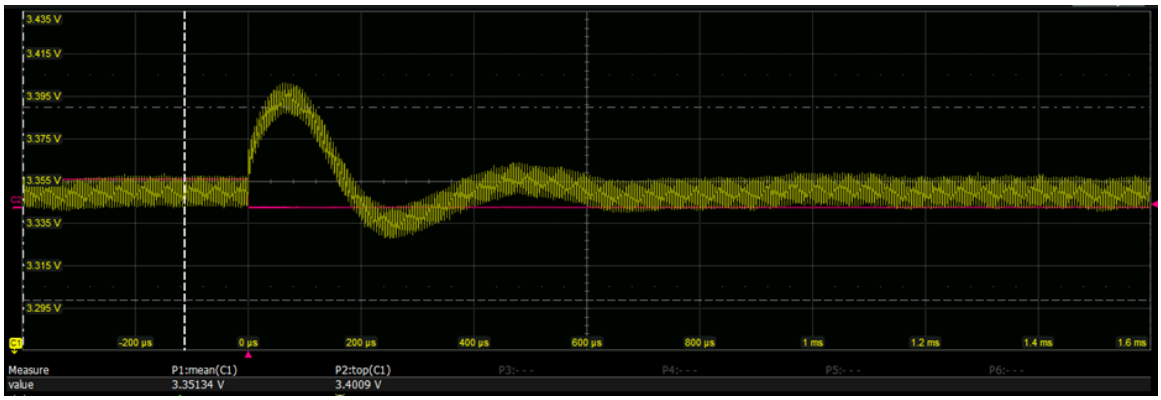
Figure 31. K_i tunning step for peak current mode control, 100% to 50% load step when K_p=3 and K_i=0.1



5. Fine-tune K_I: The trace above shows a close-to-spec setup time. Moreover, the overshoot value is out of spec.

   User can try making the response faster and lesser overshoot by increasing K_i, at the expense of some additional ringing. Below the trace obtained with K_i = 0.4

Figure 32. K_i tunning step for peak current mode control, 100% to 50% load step when K_p=3 and K_i=0.4



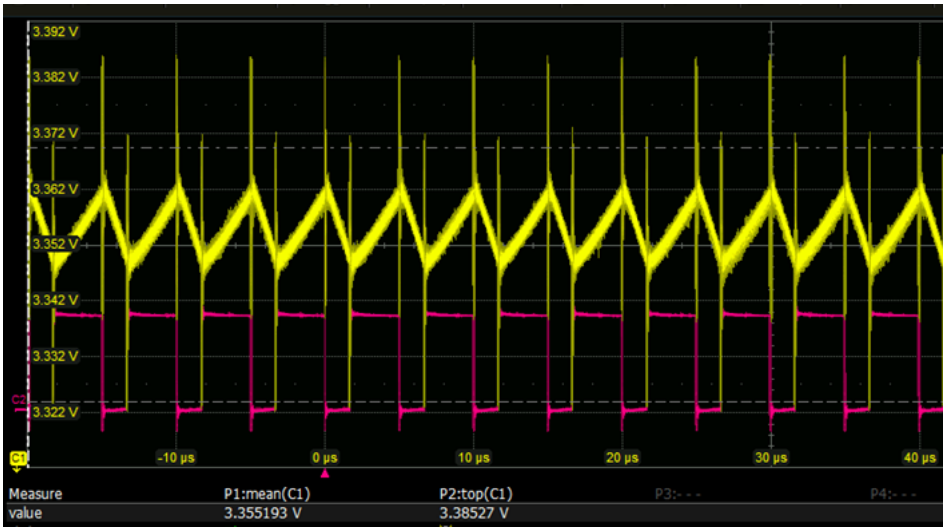
4.2.2 Voltage mode PI tuning

As the dynamics of this control mode are different, so it is necessary to perform the tuning process again to obtain adequate PID parameters.

1. Identify the u_{steady} value. In the case of peak current mode control, the output voltage at 50% stabilizes at 3.3 V when the output of our controller is set to u_{steady} = 18430.

2. Identify the limiting K_P value. The oscilloscope trace below shows the observed behavior for K_p = 200 (normal mode acquisition triggered on the output PWM rising edge):

Figure 33. Steady-state oscillations of peak current mode control with K_p=200

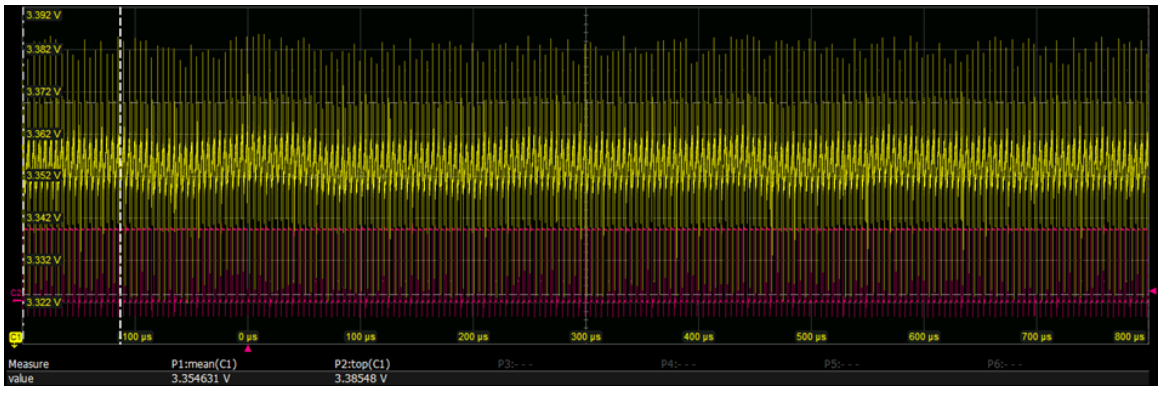


Here, the yellow curve corresponds to the output voltage, and the pink to the PWM driving the top switch of the buck plant. The thickness of the yellow trace indicates a maintained oscillation on the output voltage.

Then the user reduces the K_p value until the oscillations are less perceptible. The single-shot acquisition can be used for a finer view of the presence of oscillations.

Below is a single shot triggered on the PWM rising edge, with K_p = 150.

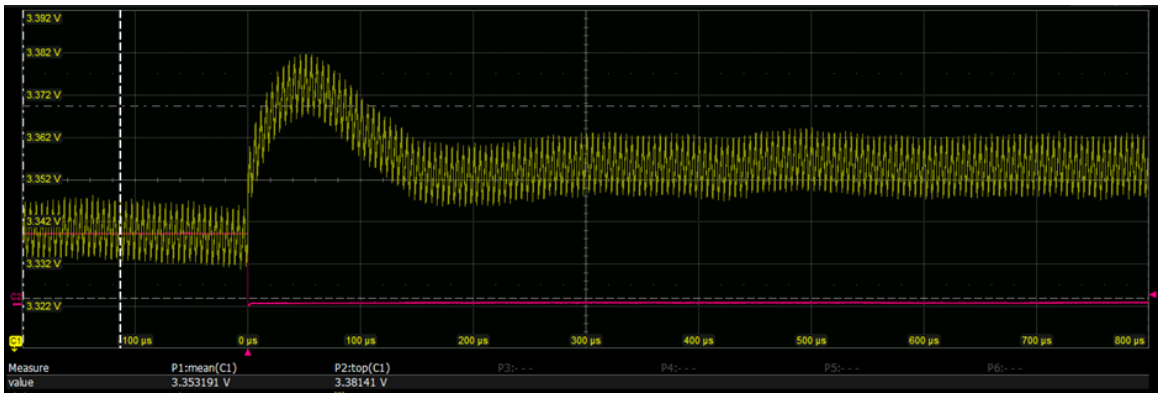
Figure 34. Steady-state oscillations of peak current mode control with K_p=150



Oscillations are not much attenuated below this K_p value. The user therefore takes K_p = 150 as the limit for the proportional effect. Note that the observed ripple as expected in an SMPS of this type. Here, the focus is on the average value oscillation

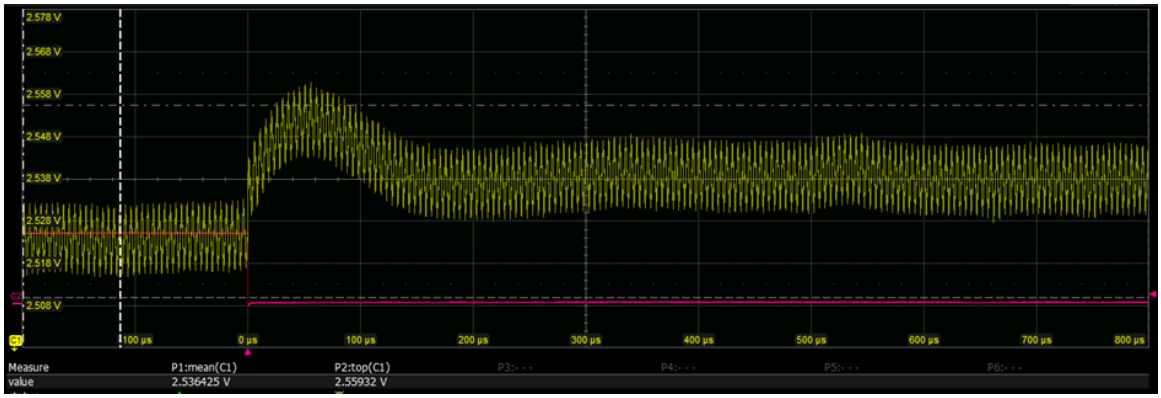
3. Fine-tune the $K_P$value. If we now halve the value of K_P to 75, no oscillations are observed. Note that only with this K_p value (without integral effect), a 100% to 50% load clearly does not track the 3.3 V reference:

Figure 35. 100% to 50% load-step on peak current load with final K_p value and steady state term, without integral effect



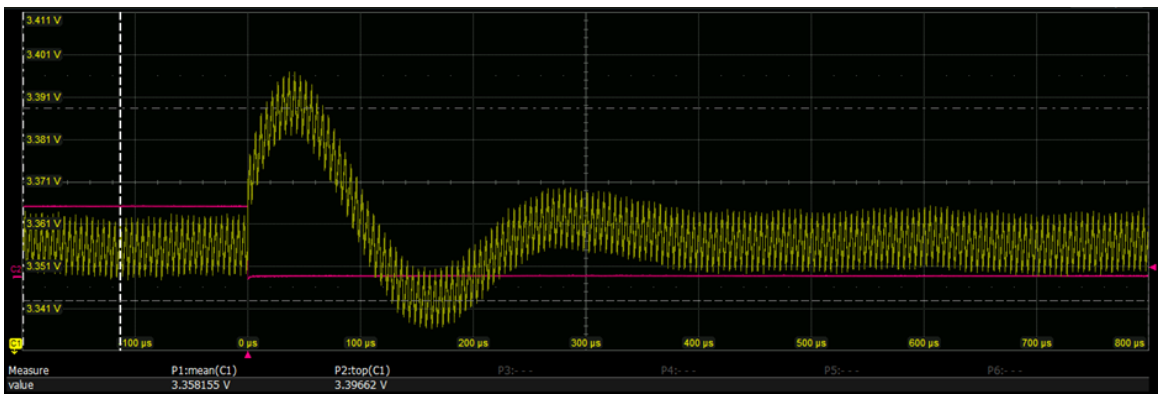
Here, the pink trace corresponds to the signal commanding the switches that enables/disables 100% load. Note that when the constant term is suppressed (which is necessary to tune the K_i value), the output voltage is lower. It is therefore further from the 3.3 V reference.

Figure 36. 100% to 50% load-step on peak current load with final K_p value, without integral effect



4. Approximate the $K_I$value. By configuring a small increase of K_i, the steady state regulation can already be observed. Below, the resulting load-step transient with K_i = 5.

Figure 37. Figure 30: Ki tuning step for peak current mode control. 100% to 50% load step when K_p=75 and K_i=5



5. Fine-tune K_I. Higher values seem to extend the second ring, and smaller values make the response slower with a high overshoot also. The user therefore keeps $K_i = 5$.

4.3 Perfomance comparison

This section shows a comparison between the results obtained with the finely designed IIR compensators with respect to the empirically tuned PI controllers, on both voltage and peak current mode plants.

Three criteria are observed:

- Transient response
- Frequency response
- CPU-loading

4.3.1 Transient response

Figure 38 and Figure 39 show the 100% to 50% load step transient response obtained with the 2p2z compensator on peak current mode, and the 3p3z compensator on the voltage mode

Figure 38. 100% to 50% load step transient response of 2p2z compensator controlling the peak current mode plant

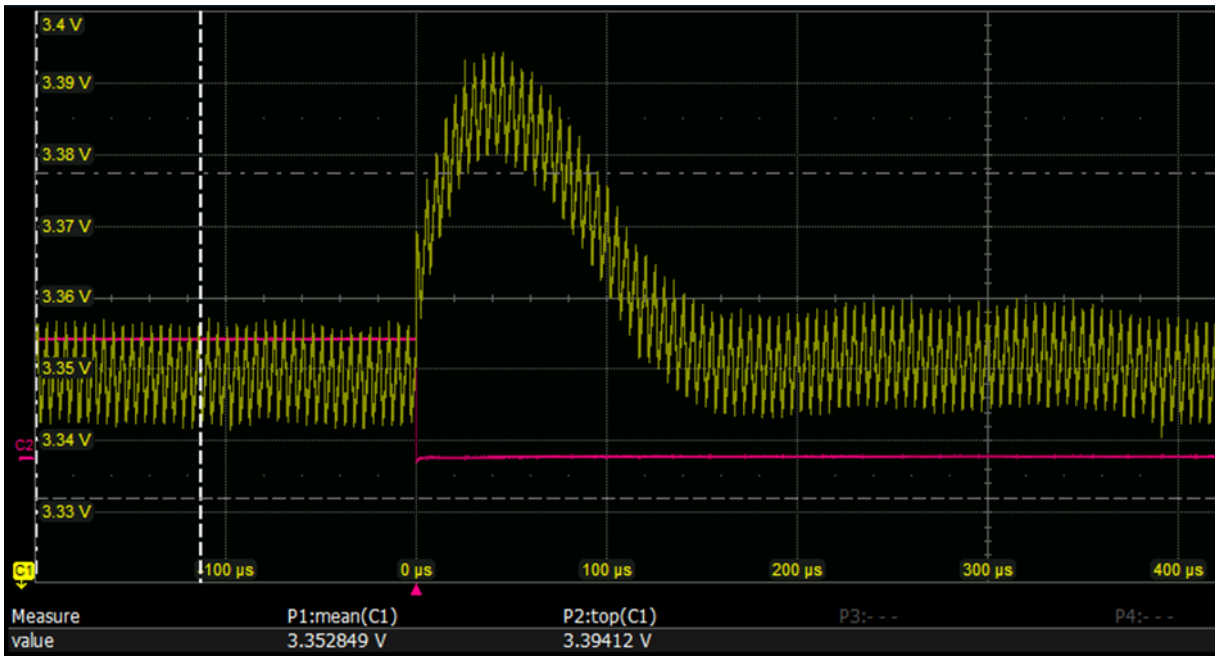
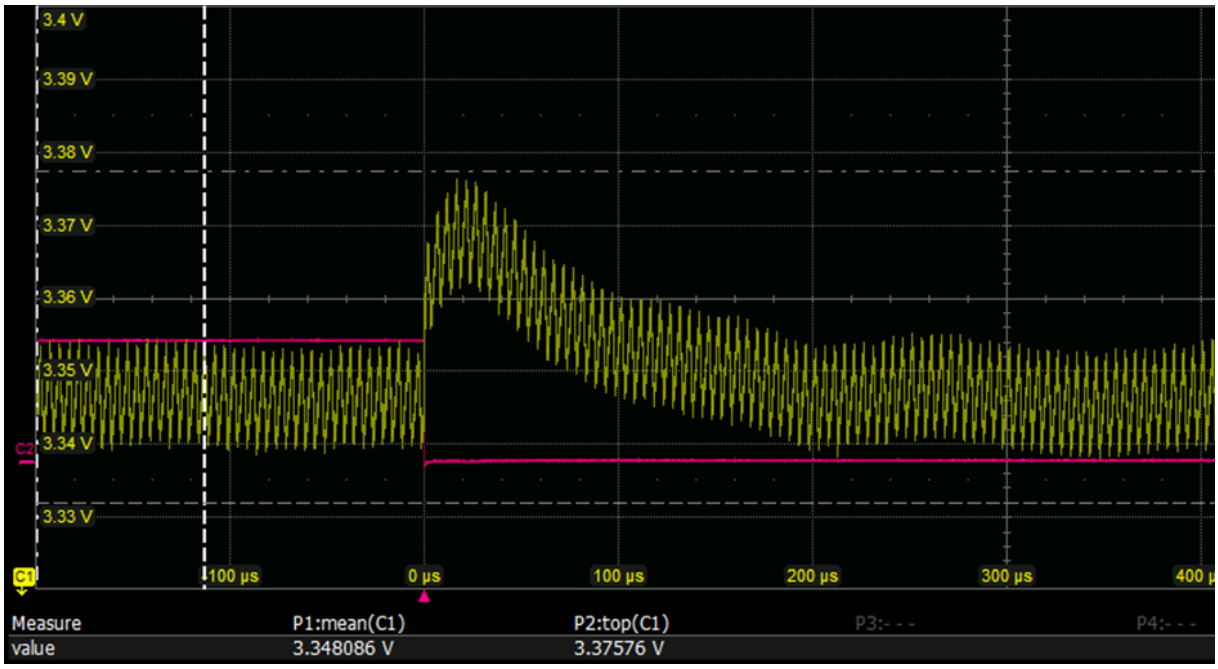


Figure 39. **100% to 50% load step transient response of 3p3z compensator controlling the voltage mode plant**



With these results, the user can now complete Table 7, summarizing the transient behaviour of the different controllers.

Table 7. **Summary of 100% to 50% load-step transient response with each controller**

Controller	Average output voltage (V)	Overshoot (mV)	Setup time (µs)	Number of rings
PI on voltage mode	3.36	39	400	1
3p3z on voltage mode	3.35	28	200	0
PI on current mode	3.35	50	650	1
2p2z on current mode	3.35	41	220	0

As expected, it is observed for both control modes that the finely tuned IIR filter gives better transient response than the PI controller. The overshoot values obtained with the heuristically tuned PI are nevertheless reasonable, although this might not be easily achievable on any platform.

4.3.2 Frequency response

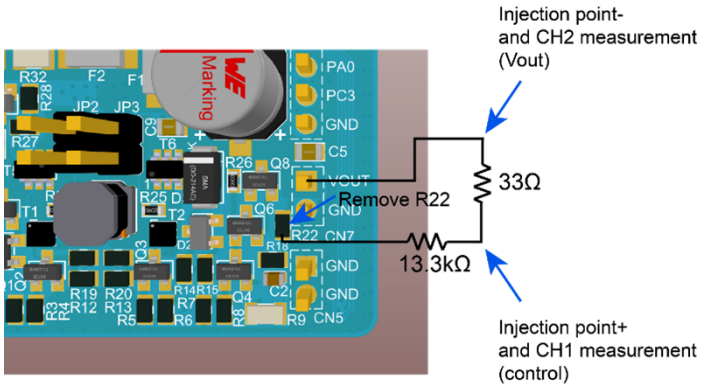
The real-life loop response of the buck converter can be verified through measurement using a frequency response analyzer. In this application note, the Bode 100 vector network analyzer from Omicron lab makes the measurement. The Bode 100 injects a sinusoidal signal into the feedback loop of the power supply. It measures how this signal changes as it passes through the controller and the plant power stage.

A small modification is required to the discovery kit to measure the loop.

The feedback path of the output voltage must be broken, and an injection resistor needs to be inserted. The injection transformer is then connected across this resistor.

The injection transformer overlaps the sinusoidal signal from the Bode 100 onto the feedback voltage, which is being used to close the loop. The schematic for this modification is shown in Figure 40.

Figure 40. Connection setup for control-to-output transfer function measurement



Biricha describes this process in [9]. Note that probe grounds must be connected to the closest header marked GND. Also, JP2 and JP3 are not populated by default on this board.

A frequency sweep from 100 Hz to 100 kHz is recommended (up to half the switching frequency). The signal injection level is adjusted to give a continuous smooth measurement without affecting the steady-state response of the loop.

The following figures show the measured frequency behaviors of the voltage and peak-current modes with each type of controller:

Figure 41. Gain magnitude of the voltage-mode controlled buck at 100% output load

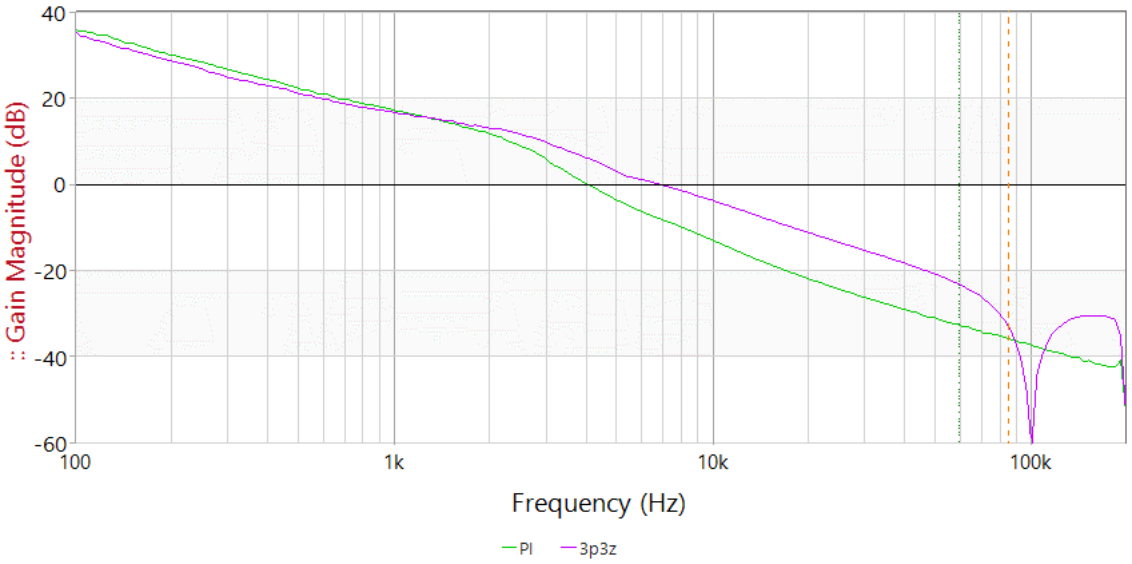


Figure 42. **Gain phase of the voltage-mode controlled buck at 100% output load**

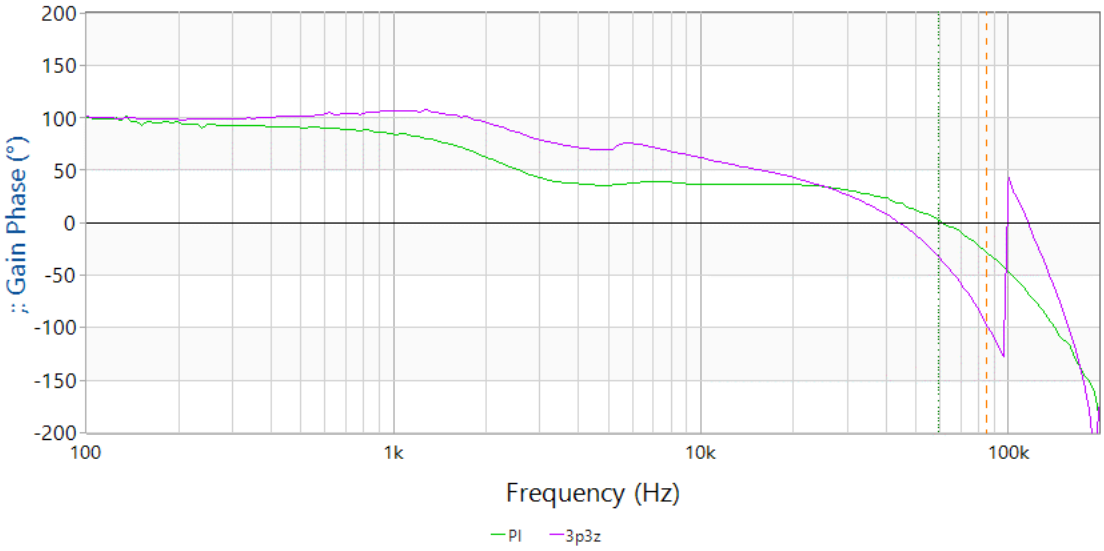


Figure 43. **Gain magnitude of the peak current-mode controlled buck at 100% output load**

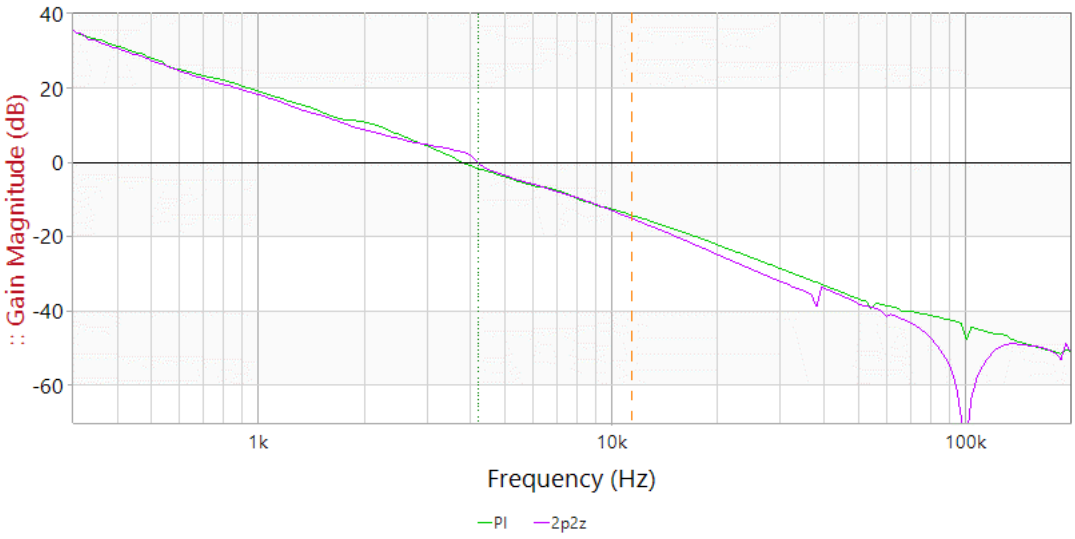
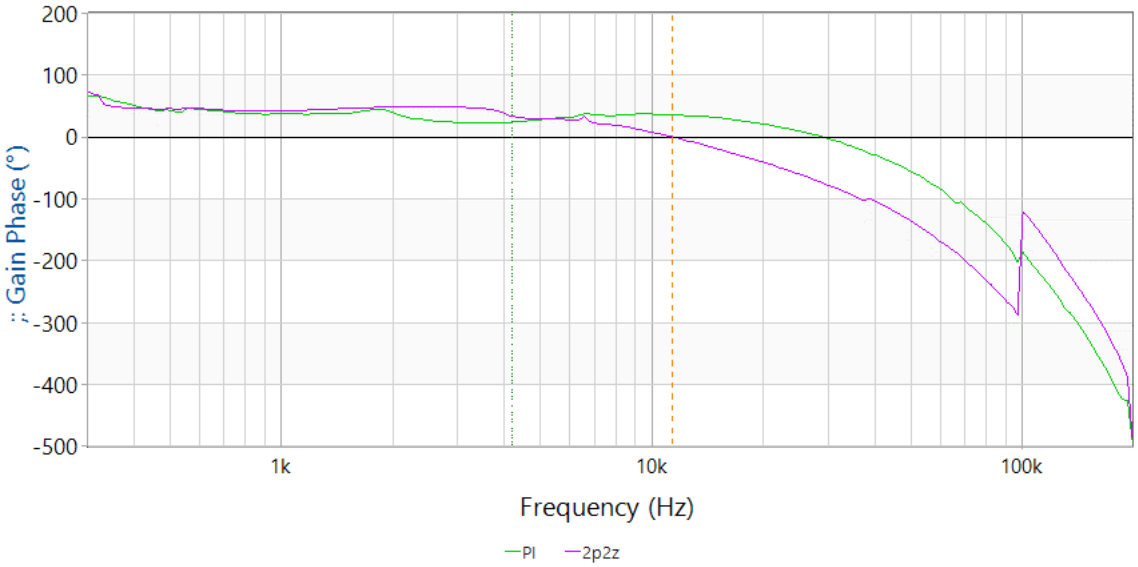


Figure 44. **Gain phase of the peak current-mode controlled buck at 100% output load**



The frequency characteristics of each controller are summarized in Table 8.

Table 8. **Measured frequency characteristics of each type of controller**

Controller	Cross-over frequency (kHz)	Phase margin (°)	Gain margin (dB)
Voltage mode PI	4 kHz	35.9°	32.2 dB
Voltage mode 3p3z	7.1 kHz	72.3°	19.4 dB
Current mode PI	3.74 kHz	22.3°	27.7 dB
Current mode 2p2z	4.18 kHz	32.9°	15.1 dB

The cross-over frequencies obtained are closer to those specified with IIR filters, under both voltage-mode and current-mode control. This ensures a better transient response.

The phase margin results with the IIR filters are better than those with the PI. This is the case for both voltage mode and peak-current mode controllers.

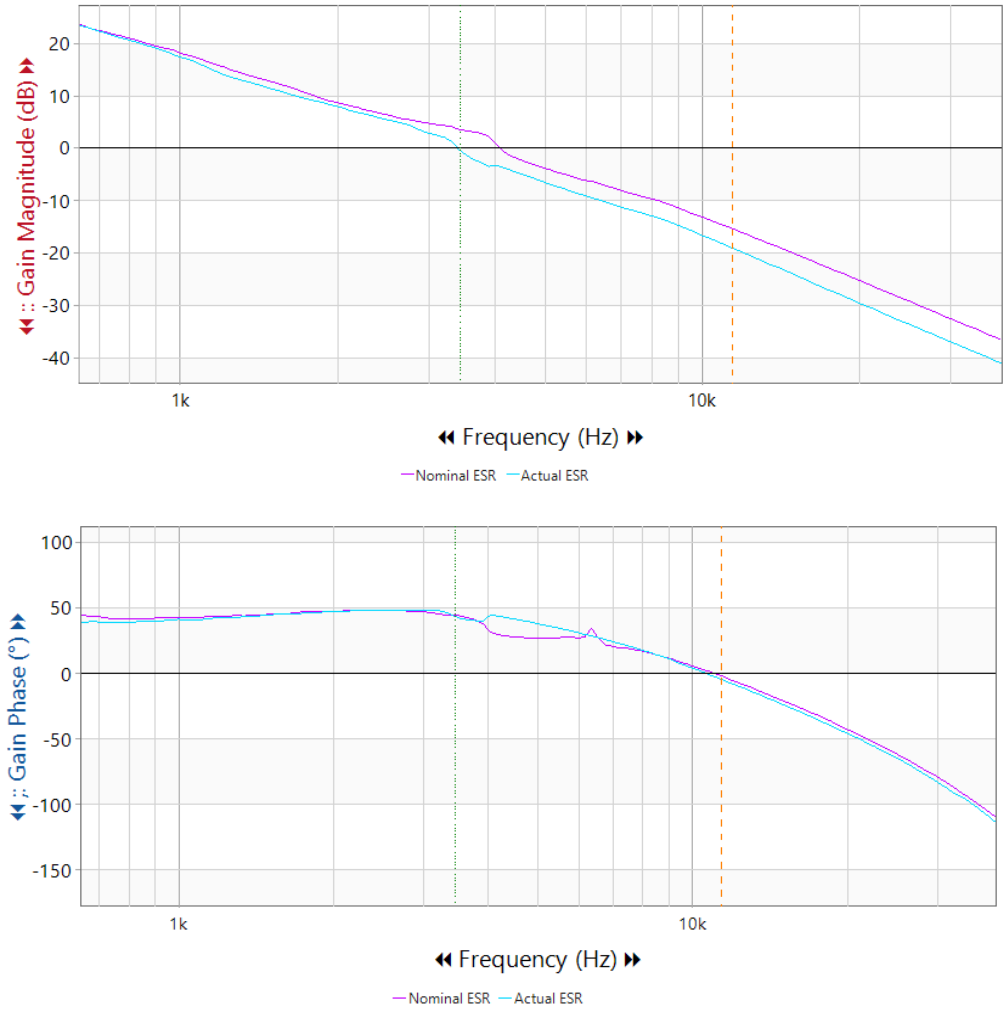
The obtained gain margins are nevertheless better on the PI than on the IIR compensators.

Note that the hardware parameters used to compute the IIR filter coefficients were those corresponding to the nominal behavior of the components. Thus, the possible component drift/aging/tolerances have not been considered. Using ST-WDS®, the poles/zeroes can nevertheless be readjusted until the specified frequency characteristics are obtained.

This was done to improve the phase margin results on the current mode 2p2z controller. The parasitic equivalent series resistance (ESR) of the output capacitor introduces a zero. This has a significant impact on the plant transfer function for the buck converter. The location of this zero is dependent on the capacitance and ESR values.

Therefore, the computed compensator's efficiency is reliant on the accurate determination of the actual ESR and capacitor values. Thus, the capacitor on the example board was measured and the coefficients were recomputed for this value. The figure below represents the obtained bode plot with the first coefficients and the recomputed ones.

Figure 45. Measured frequency behavior with 2p2z compensators computed with nominal capacitor values and actual capacitor values



By readjusting the 2p2z compensator coefficients, the phase margin was improved from 32.9°C to 44.6°C.

4.3.3 CPU loading

The CPU loading can be characterized by the time spent in the control loop IRQ. This IRQ is triggered by the end of the ADC conversion when the controller is executed in the CPU, while it is triggered at the end of the compensator execution when executed in the FMAC.

To measure this time, a GPIO is set high when entering the IRQ and cleared before exiting. Therefore, the measured time does not consider the maximum 29 clocks required for the Cortex®-M4 to start executing the IRQ and at most 27 cycles to return to the previous context. [9]

With a CPU clock of 170 MHz, this corresponds to approximately 329 ns for IRQ entering and exiting.

Note that this is a worst-case scenario. More optimal timing can be reached by using CCM-SRAM (0-wait-states access memory) for the IRQ, as is discussed in [1] and [2]. In any case, the benchmark here discussed focuses on the controller execution, thus ignoring interrupt latency.

The figures below show the time spent in the IRQ (after entering and before exiting), with each type of controller (both with CPU and FMAC in the case of the IIR filter).

Where channel 1(in yellow color) is probing the output toggled when entering and exiting the IRQ, and channel 2 (in pink color)is probing the PWM driving the top switch of the buck plant.

Figure 46. **Time spent in the IRQ when executing a PID with anti-windup**

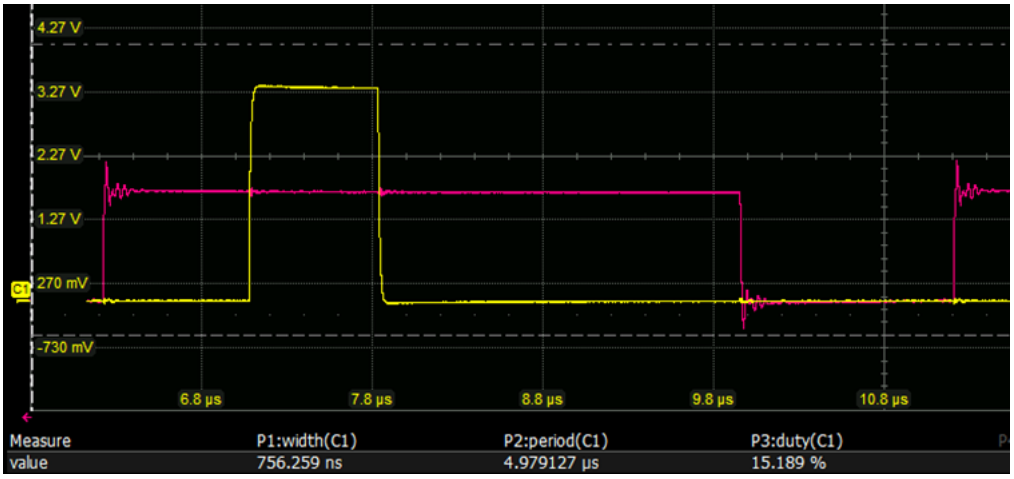


Figure 47. **Time spent in the IRQ when executing a PI without anti-windup**

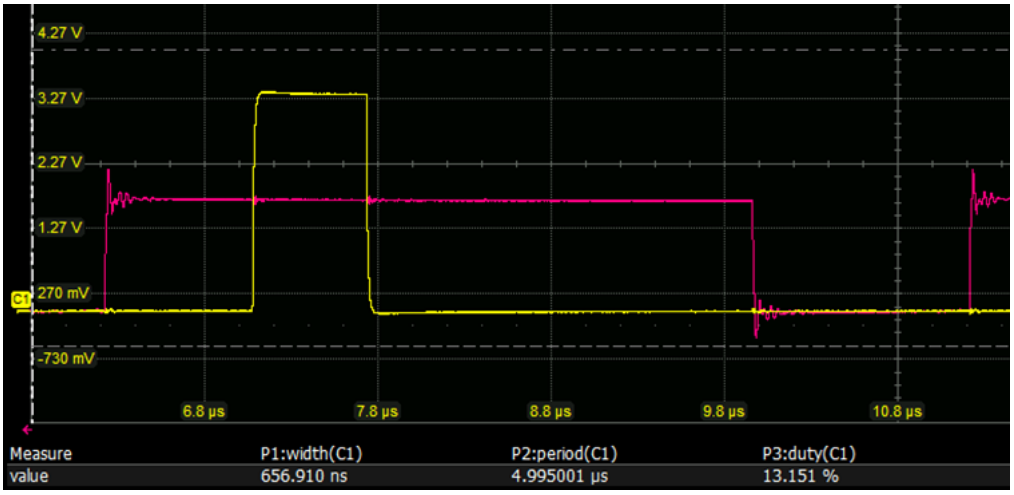


Figure 48. **Time spent in the IRQ when executing the IIR filter in the CPU**

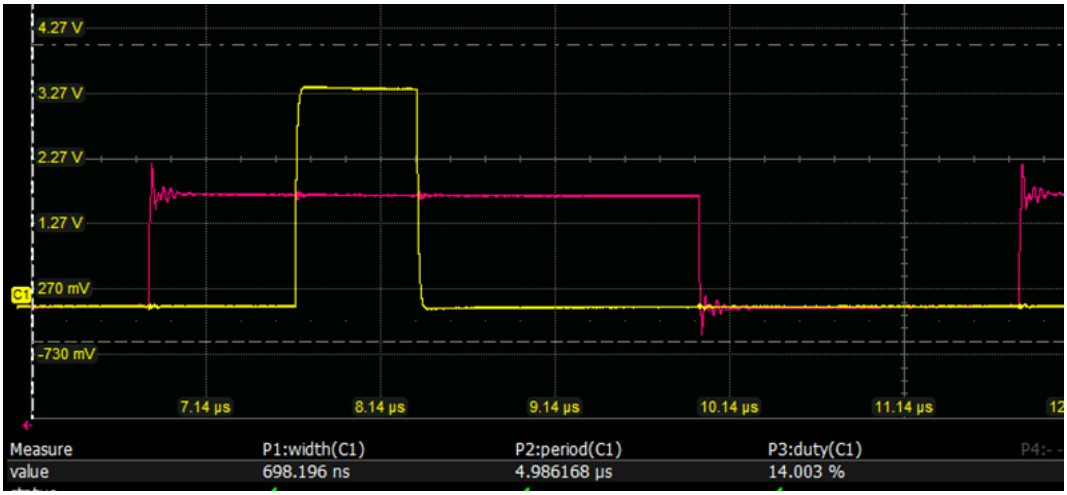
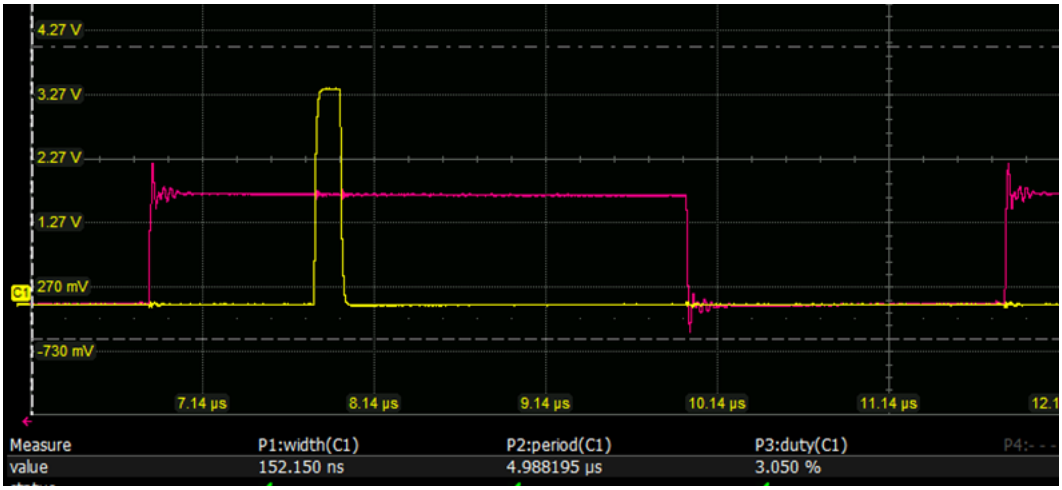


Figure 49. **Time spent in the IRQ when executing the IIR filter in the FMAC**



These results are summarized in the following table:

Table 9. **Summary of the CPU loading with each type of controller**

CONTROLLER	CPU-time (ns)	% of PWM period
PID with anti-windup	756	15.2
PI without anti-windup	657	13.1
IIR on CPU	698	14
IIR on FMAC	152	3

The first observation is that CPU load is going to be dependent on the type of controller. As an example, the differential term and non-linear anti-windup mechanisms were applied on the PI controller to see the impact on the execution time. The change is not huge, and PI/PID measures are comparable to that taken to execute the 3p3z function on the CPU.

However, note that a big advantage of IIR filtering is that most of the CPU bandwidth used for the loop can be saved by implementing the execution on the FMAC IP. This is not possible with the anti-windup PID controller, and can only be done for the PI if its temporal coefficients were translated to discrete frequency domain first.

5 Conclusion

Several methods exist to control SMPS plants. The digital approach allows the user to reduce the hardware dependencies of analog control, and makes it much easier to configure multiple types of microcontroller. Such controllers can be designed in the frequency or the time domain.

Frequency analysis has the advantage that the stability characteristics can be directly targeted. This leads to better robustness against component drift, aging, and tolerances. Moreover, IIR filters offer the possibility to reduce CPU loading via the FMAC IP.

On the other hand, PID controllers can also be implemented without the high BOM that they imply in analog SMPS control. It is possible to use a heuristic method to tune the proportional and integral terms of such controllers. This results in adequate system behavior, as observed in the buck converter examples.

It is the designer's choice to select the most suitable control method for the application. STM32G4 microcontrollers have all of the features needed for digital SMPS control (IP wise). They also have sufficient processing power to manage multiple controllers in parallel, as well as housekeeping functions.

The implementation is made easy by a world-class ecosystem, including: Cube software libraries, a code generation tool (STM32CubeMX), and a free digital power-control design tool (ST-WDS®). These support many topologies and control techniques.

Revision history

Table 10. **Document revision history**

Date	Version	Changes
09-May-2022	1	Initial release.

Contents

List of tables

List of figures