Optimized Implementation of a Current Control Algorithm for Multiphase Interleaved Power Converters

Pablo D. Antoszczuk, *Student Member, IEEE*, Rogelio Garcia Retegui, Marcos Funes, *Member, IEEE*, and Daniel Carrica, *Senior Member, IEEE*

***Abstract*—Multiphase converters have become an attractive alternative for high-current power converters due to their inherent reduction of semiconductor stress. Additionally, total current ripple frequency can be increased and its amplitude decreased by the phases ripple interleaving. These converters require a different number of phases and control specifications depending on the application. A wide range of applications imposes challenging requirements in the control algorithm and its implementation, such as digital platforms and resources optimization. A previous proposal presented a current control algorithm developed to provide a solution to the highly demanding constraints present in high-power applications, where short settling times are required when fast transients in the current reference or the load voltage are present. This work presents the implementation of the above-mentioned algorithm and its optimizations, aimed to obtain a modular and efficient design. The proposed implementation and system scalability are evaluated by means of an experimental setup.**

***Index Terms*—Current control, current ripple, field programmable gate array (FPGA) implementation, interleaved power converters, power conversion.**

I. Introduction

In the field of power current sources, such as in grid-connected inverters, automotive applications, power factor correction (PFC) converters, and voltage regulator modules (VRM) among others [1]–[6], multiphase buck converters (Fig. 1) have become an attractive alternative to deliver high currents. The use of N phases allows the distribution of high current among different paths, thereby reducing the conduction and commutation losses of switching devices. Additionally, the control of these converters allows to interleave the phases ripple so as to reduce the total ripple amplitude and increase its frequency. This feature is depicted in Fig. 2 for a multiphase converter operating in continuous conduction mode (CCM), where the relationship between the steady-state phase current ripple and the total current ripple for a two-, three- and six-phase converter is shown [7]. The number of phases in multiphase power converters can then be selected according to each particular application, in order to improve the output ripple characteristics, reduce the stress of the semiconductor devices, or improve fault tolerance by increasing the parallel stages [8]–[11].

The control of multiphase converters must ensure the even distribution of the current among phases, and the correct interleaving of the current ripple. Numerous control techniques have been published. Peak current mode control (PCMC) and its modifications provide intrinsic current protection, and are simple to implement [12], [13]. On the other hand, average current mode control (ACMC) and its modifications have good stability, noise immunity, and fast response [14]–[16]. These techniques are capable of correcting perturbations in the current reference and disturbances in the load voltage along several switching periods [17], [18].

In high power applications, such as high current-pulsed power converters and the internal current-controlled loops of high voltage power converters, the switching frequency is limited due to the semiconductor devices technological limitations [18]–[20]. Additionally, in these applications, disturbances such as major changes in the current reference and load voltage are present. Load voltage perturbations, which are a consequence of current reference or load variations, modify ripple slopes and amplitude. The use of the aforementioned current control techniques in the applications where the described disturbances are present, produces nonacceptable transitory times. In order to solve these settling-time limitations, Garcia Retegui *et al.* [21] presented a current control based on synchronism reference signals in order to adjust the zero-crossing of the phase current error. This algorithm is capable of recovering the interleaving among phases with a reduced-transitory time and error.

Even though the calculations required by [21] are simple and not highly demanding, requirements and system complexity increase with the number of phases. These calculations may, therefore, limit the maximum number of phases and switching frequency for a given digital platform. Resources optimization and digital platform selection are consequently important issues that must be addressed for implementation.

Regarding the digital platform, different choices to perform the algorithm computation are available, such as digital signal processors (DSPs) or programmable logic devices (PLDs). The main drawback of DSPs is the difficulty in taking advantage of

Manuscript received March 07, 2014; revised August 06, 2014 and June 03, 2014; accepted October 03, 2014. Date of publication October 08, 2014; date of current version November 04, 2014. This work was supported in part by the Universidad Nacional de Mar del Plata (UNMDP); in part by the National Scientific and Technical Research Council (CONICET); in part by the Ministry of Science; in part by Technology and Productive Innovation (MINCYT); in part by the National Agency of Scientific and Technological Promotion, Argentina; in part by the European Organization for Nuclear Research (CERN), Switzerland; and in part by the European Particle Physics Latin American Network (EPLANET). Paper no. TII-14-0290.

The authors are with the Instrumentation and Control Laboratory, Universidad Nacional de Mar del Plata, Mar del Plata 7600, Argentina (e-mail: pablo_ant@fi.mdp.edu.ar; rgarcia@fi.mdp.edu.ar; mfunes@fi.mdp.edu.ar; carrica@fi.mdp.edu.ar).

Digital Object Identifier 10.1109/TII.2014.2362071

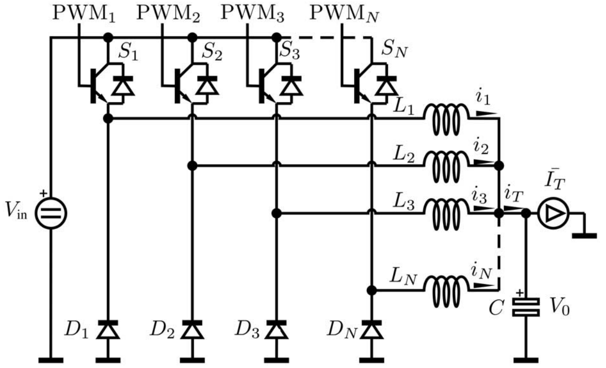


Fig. 1. Parallel buck converter topology.

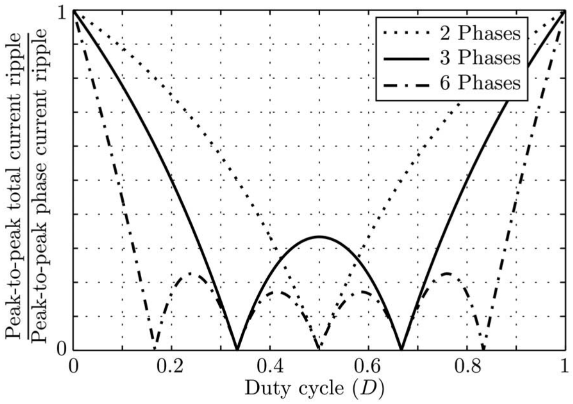


Fig. 2. Total ripple attenuation in CCM as a function of the duty cycle, for different phase numbers.

the potential parallelism of the current control algorithm [22]. This drawback may limit the system flexibility when different number of phases are required. On the other hand, the parallel processing capabilities of programmable logic devices allow to implement the control in a modular way, increasing the flexibility and easing optimization. Particularly, field programmable gate array (FPGA) is an attractive solution given its simplicity and availability as a standard component. The use of this technology to develop and improve digital control systems has been well documented and exemplified in the literature [23]–[26].

In this work, practical implementation aspects for the algorithm proposed in [21] are presented. The optimizations required for the efficient utilization of the available resources are dealt with. These optimizations focus on algorithm implementation for converters with different number of phases, without degrading the switching frequency. The optimizations and practical implementation approaches described may be extended to different control algorithms. The proposed control algorithm implementation and scalability are evaluated by means of an experimental setup.

II. Current Control Algorithm

The current control algorithm proposed in [21] is based on the independent control of each phase switching instants, for converters operating in CCM. Switching instants are calculated to match each phase current-error zero-crossings with a synchronism signal, generated internally by the current control.

TABLE I
Current Control Nomenclature

i_{Ref}	Current reference
i_i	Phase-i phase current
i_{ei}	Phase-i current error $i_{ei} = i_{\text{Ref}} - i_i$
PWM$_i$	Phase-i switch driving signal
V_{in} ,V_0	Input and output voltage
V_{ind} ,$V_{0\text{d}}$	Digital input and output voltage
Sync$i$	Phase-i synchronization signal
T_{Sync}	Synchronization signal period
T_S	Switching period
$t_e(k)$	Time error between i_e zero crossing and syncronization signal at instant k
$t_{\text{hp}}(k+1)$	Next semi-period time: $t_{\text{hp}}(k{+}1) = \frac{T_{\text{Sync}}}{2} - t_e(k)$
$t_{\text{sw}}(k+1)$	Interval between i_e zero crossing and PWM$_i$ switching instant

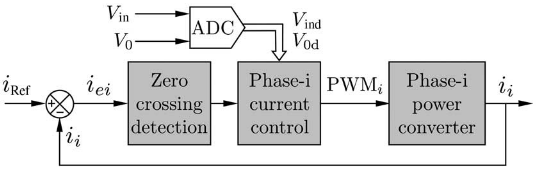


Fig. 3. Phase i control block diagram.

For the sake of clarity, Table I shows the nomenclature of [21]. Additionally, in order to illustrate the current control input signals, Fig. 3 shows the block diagram corresponding to phase-i current control. Current error i_{ei} is generated by substracting the phase current from the current reference, i.e., $i_{ei} = i_{\text{Ref}} - i_i$. The zero-crossing instants of i_{ei} are detected by the zero-crossing circuitry. Current control block uses this zero-crossing information, and the acquired input and output voltages V_{ind} and $V_{0\text{d}}$ for the algorithm calculations. It is worth noting that, by using the above-mentioned definitions, steady-state total mean current is $\bar{I_T} = N i_{\text{Ref}}$.

The control algorithm calculates, for each phase, the time that the switch must remain in its current state before commutation, measured from i_e zero-crossings. This time, defined as switching time t_{sw}, is calculated so as to adjust i_e zero-crossings with its corresponding synchronism signal, in such a way that the steady-state switching period is equal to the synchronism period $T_S = T_{\text{Sync}}$. Additionally, in the synchronized condition, the phase-shift between phases ripple is determined by the phase-shift of the synchronism signals. As an example, Fig. 4 shows the current error ($i_{e1,2,3}$) and the corresponding synchronism signals (Sync1, Sync2, Sync3) for a three-phase converter operating in steady-state condition. It should be noted that the synchronism signals are not only a time reference for the i_e zero-crossing instants, but also indicate the correct i_e slope sign in those instants.

Assuming that the time constant associated with the inductors and their resistive component is much higher than the switching period, the current ripple can be approximated by linear segments. Fig. 5 illustrates i_e and the synchronism signal for a single phase, where the instants $(k-1)$, (k), and $(k+1)$ are defined at the same instants as the synchronism signal. As it can be seen, at $(k-1)$, the zero-crossing with positive slope occurs simultaneously with a positive synchronism signal, thus, the synchronization error $t_e(k-1) = 0$. At time (k), however,

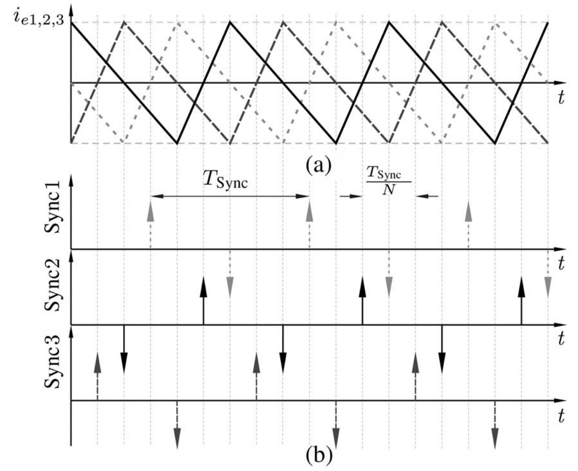


Fig. 4. (a) $i_{\rm ei}$ in ideal interleaved operation. (b) Synchronization signals.

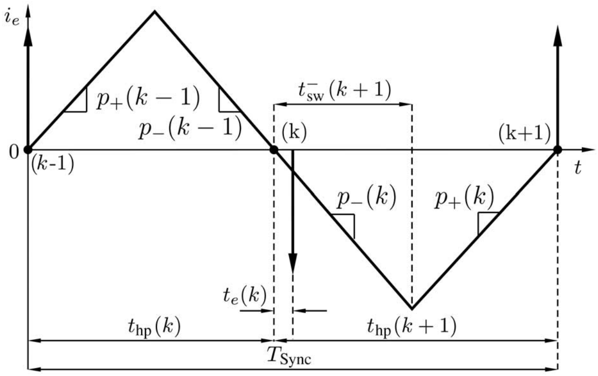


Fig. 5. Switching time calculation for small i_e perturbations.

due to a small perturbation in i_e, the zero-crossing with negative slope occurs at a different instant than the corresponding synchronism signal, i.e., $t_e(k) \neq 0$. Then, the switching time, which adjusts the zero-crossing at $(k+1)$, measured from the negative-slope zero-crossing at (k), is calculated as

$$t^-_{\rm sw}(k+1) = \frac{p_+(k)}{p_+(k) - p_-(k)} \cdot t_{hp}(k+1) \tag{1}$$

where $p_+(k)$ and $p_-(k)$ are the positive and negative slopes, and $t_{\rm hp}(k+1)$ is the duration of the next semiperiod, defined as

$$t_{\rm hp}(k+1) = \frac{T_{\rm Sync}}{2} - t_e(k). \tag{2}$$

The synchronization error $t_e(k)$ can be negative or positive depending on whether the zero-crossing occurs before or after the synchronism signal.

Assuming that the voltage drop in the semiconductor devices and parasitic components is negligible with respect to the input and output voltages, slopes $p_+(k)$, and $p_-(k)$ are approximated as a function of the input and output voltages, measured in the i_e zero-crossing point, as

$$p_+(k) = \frac{V_{\rm in}(k) - V_0(k)}{L_1} \tag{3}$$

$$p_-(k) = -\frac{V_0(k)}{L_1} \tag{4}$$

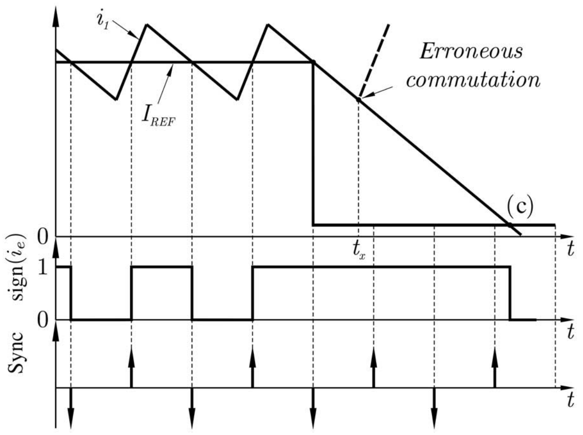


Fig. 6. Current reference tracking.

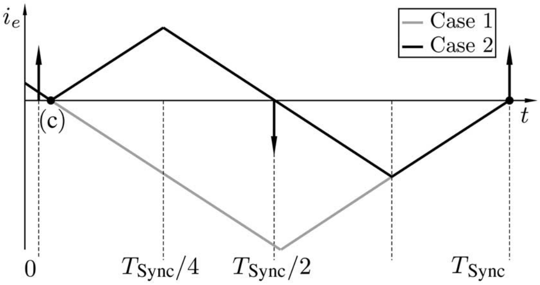


Fig. 7. Interleaving recovery after major synchronization error.

then, (1) becomes

$$t^-_{\rm sw}(k+1) = \left(1 - \frac{V_0(k)}{V_{\rm in}(k)}\right) \cdot t_{\rm hp}(k+1). \tag{5}$$

Analogously, the calculation of the switching time when the i_e zero-crossing occurs with positive slope $t^+_{\rm sw}(k+1)$ is calculated as

$$t^+_{\rm sw}(k+1) = \frac{V_0(k)}{V_{\rm in}(k)} \cdot t_{\rm hp}(k+1). \tag{6}$$

Expressions (5) and (6) allow to synchronize the i_e zero-crossings with the synchronism signals. Additionally, the control algorithm takes into account situations that may arise under large $i_{\rm Ref}$ variations, or disturbances in the load voltage produced by said i_{Ref} modification, in order to determine the correct action to: 1) ensure the i_{Ref} tracking; and 2) optimize the recovery of the synchronism after transitory conditions.

These situations are illustrated in Fig. 6, where a negative $i_{\rm Ref}$ step, which occurs immediately after the i_e zero-crossing with negative slope, is shown. As it can be seen, by using (5), the system calculates the next commutation instant t_x leading to a commutation opposite to $i_{\rm Ref}$ tracking. These erroneous commutations can be detected by checking the coincidence between the slope of the current and the sign of the error, so as to inhibit the commutation and repeat the calculation at (c).

At (c), however, the i_e zero-crossing with negative slope occurs close to a synchronism corresponding to the opposite slope. By calculating the switching time using (5), the system recovers the synchronized condition in the next positive synchronism signal, as illustrated in Fig. 7 (Case 1). As it can

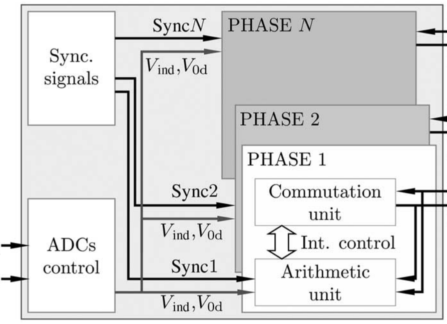


Fig. 8. Current control implementation block diagram.

be seen, this condition generates a bigger transient amplitude and setup time than Case 2, in which an anticipated commutation is performed in (c), and (6) is used for the computation of the switching time. The anticipated commutation is performed when $t_e(k) > T_{\mathrm{Sync}}/4$, reducing the transient time and amplitude.

III. Digital Implementation

The advantage of implementing the current control algorithm in a programmable logic device lies in the fact that it allows the control tasks parallelization. Additionally, and given its formulation, the current control algorithm implementation can be designed for one phase and then easily replicated to an N-phase system, providing modularity to the control system.

Fig. 8 shows a simplified-block diagram of the proposed algorithm implementation. Each phase is composed of a commutation unit, which commands the pulse width modulation (PWM) signal as a function of the current state of the system; and an arithmetic unit, which is controlled by the commutation unit to compute expressions (5) and (6). The commutation and arithmetic units share information via internal control signals. The synchronism signal for each phase is generated by the Sync signals generator block. Input and output voltages, V_{ind} and V_{0d}, required for i_e slope calculation, are provided by the analog to digital converters (ADCs) control block.

A. Synchronism Signals Generation

The Sync. signals block generates N synchronism signals, one for each phase. The time-shift between each signal is used to interleave the steady-state phases ripples in order to obtain optimal total ripple attenuation. Fig. 9 illustrates the generation of phases 1 and i synchronism signals, Sync1 and Synci for an N-phase system. As it can be seen, the synchronism signals are generated by taking the most significant bit (MSB) of each b-bits Ramp counter, which generates a square wave with period T_{Sync}. The rising and falling edges of Sync signals represent the positive and negative synchronization signals, respectively. The correct time-shift between the N binary counters, therefore, the synchronism signals, is performed by loading an initial value at system start-up

$$\mathrm{Ramp}i(0) = 2^b \frac{i-1}{N} \tag{7}$$

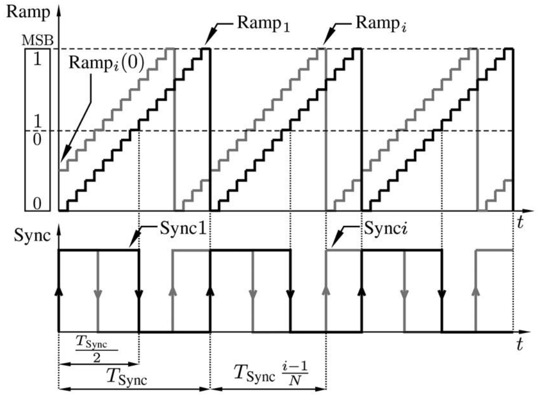


Fig. 9. Synchronism signals generation.

where $\mathrm{Ramp}i(0)$ is the phase-i Ramp counter initial value and i is the phase number $1 < i < N$.

It should be noted that in case the counter module is not divisible by N, there exists an error in the phase shift between the synchronism signals. In this case, the maximum phase shift error e_ϕ is the error corresponding to the least significant bit relative to the counter module, as depicted in (8)

$$e_\phi = 2\pi \frac{1}{2^b}. \tag{8}$$

$\mathrm{Ramp}i$ counters further provide the time-base reference for all the computations required by the control of each phase. The precision for the calculation of $\mathrm{t_{sw}}$, and therefore the PWM_i output, is then defined by the counter module. Additionally, the steady-state switching frequency $f_S = 1/T_S$ is determined both by the counter module and clock frequency, as shown

$$f_S = \frac{f_{\mathrm{CLK}}}{2^b} \tag{9}$$

Therefore, for a given switching frequency, if the precision is to be increased, the clock frequency must be increased accordingly, i.e., f_{CLK} must be doubled for each additional bit in the counters. In this way, as maximum f_{CLK} is determined by the efficiency of the design, maximum f_S is directly affected by the algorithm optimizations.

B. Arithmetic Unit

Equation (5) or (6) must be computed at every i_e zero-crossing for the determination of the next switching time. Solving these equations requires one product, one division and one additional subtraction in (5). The practical implementation, however, must take into consideration the particular capabilities of the digital platform in order to optimize the available resources.

The algorithm computation can be modified to profit from the parallelization capabilities of the FPGA. By analyzing (5) and (6), a generic t_{sw} calculation can be defined as

$$t_{\mathrm{sw}}(k+1) = \left(\frac{K_2(k)}{V_{\mathrm{ind}}(k)}\right) \cdot t_{\mathrm{hp}}(k+1) \tag{10}$$

where $K_2(k)$ is $(V_{\mathrm{ind}}(k) - V_{\mathrm{0d}}(k))$ or $V_{\mathrm{0d}}(k)$ depending on i_e slope sign at the zero-crossing instant, and $V_{\mathrm{ind}}(k)$ and $V_{\mathrm{0d}}(k)$

are the input and output voltages provided by the ADCs control block at every i_e zero-crossing instant.

The computation of (10) can be performed by increasing t_{sw} in every clock pulse and comparing this variable with the right-side terms of the equation. Variable t_{sw} is then implemented by means of a rising counter, t_{sw} counter, which restarts at every i_e zero-crossing points. The instant at which t_{sw} is equal to (or greater than) the right side of (10) indicates that the state of the PWM_i output can be changed. It is worth noting that, as t_{sw} counters have the same module as $\text{Ramp}i$ counters, precision in the PWM_i output is also determined by the counters module 2^b.

The computation can be further optimized for the implementation in programmable logic devices by avoiding the quotient $K_2(k)/V_{\text{ind}}(k)$ calculation. Rewriting (10) as (11), and performing the procedure described, the quotient is transformed into two multiplications. Since division is a very expensive operation and most programmable logic devices include embedded multipliers, this modification leads to resources optimization

$$V_{\text{ind}}(k) \cdot t_{\text{sw}}(k+1) \geq K_2(k) \cdot t_{\text{hp}}(k+1). \tag{11}$$

The computation of $V_{\text{ind}}(k) \cdot t_{\text{sw}}(k+1)$ is performed at each clock instant, and $K_2(k) \cdot t_{\text{hp}}(k+1)$ is calculated in each i_e zero-crossing. $V_{\text{ind}}(k)$ and $V_{0\text{d}}(k)$ are updated when the i_e zero-crossing points are detected. Voltage acquisition period should be $T_{\text{sample}} \leq T_{\text{Sync}}/(2\text{N})$, in order to provide updated information for each phase zero-crossing points that span over one synchronism period.

Equation (11) requires the t_{hp} value calculation when the i_e zero-crossing point is detected. This value is computed using (2) and the time reference provided by $\text{Ramp}i$ counter. Considering that $t_e(k)$ can be positive or negative and the two possible slope sign at the i_e zero-crossing instant, four different cases for the computation of $t_{\text{hp}}(k+1)$ arise.

1) Case 1: i_e slope $> 0, t_e > 0 \Rightarrow t_{\text{hp}}(k+1) = \frac{M}{2} - r_1$.
2) Case 2: i_e slope $> 0, t_e < 0 \Rightarrow t_{\text{hp}}(k+1) = \frac{3M}{2} - r_2$.
3) Case 3: i_e slope $< 0, t_e < 0 \Rightarrow t_{\text{hp}}(k+1) = M - r_3$.
4) Case 4: i_e slope $< 0, t_e > 0 \Rightarrow t_{\text{hp}}(k+1) = M - r_4$.

where r_1, r_2, r_3, and r_4 are the values of the $\text{Ramp}i$ counter when the i_e zero-crossing is detected, for each case, and $M = 2^b - 1$ is the counter final count. These cases are illustrated in Fig. 10, where C_i is the zero-crossing signal of i_{ei}. It can be noted that t_{hp} value is calculated by subtracting the $\text{Ramp}i$ counter value from a constant, when i_e zero-crossing is detected.

Fig. 11 summarizes the arithmetic unit used to compute the commutation instant for a single phase of the modulator. In this figure, signals $\text{START\_TSW}i$ and $\text{END\_TSW}i$ are internal control signals shared between the commutation and arithmetic units, indicated as Int. Control in Fig 8. $\text{START\_TSW}i$ is generated by the commutation unit to initiate the tsw counter, and $\text{END\_TSW}i$ is generated by the arithmetic unit when (11) is satisfied, which indicates the commutation instant.

C. Commutation Unit

The commutation unit is composed of two subunits: *commutation rules* and *commutation state machine*.

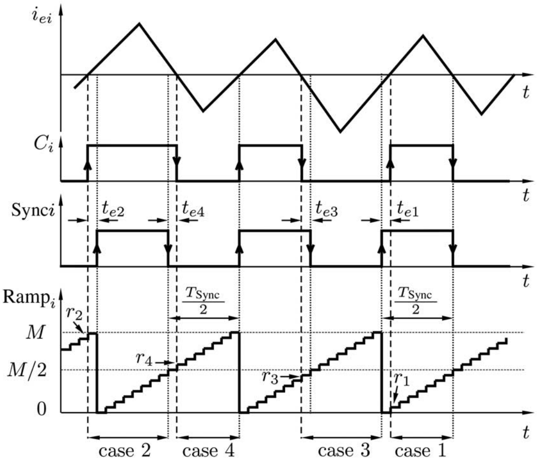


Fig. 10. Possible combinations of t_e sign and i_e slope sign for t_{hp} computation.

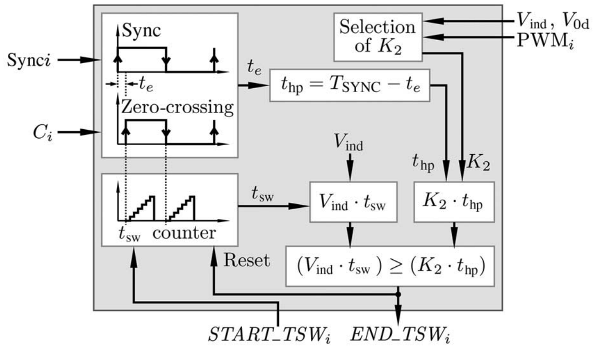

Sync
Synci
t_e
t_e
Zero-crossing
C_i
Selection of K_2
V_{ind}, $V_{0\text{d}}$
PWM$_i$
$t_{\text{hp}} = T_{\text{SYNC}} - t_e$
V_{ind}
t_{hp}
K_2
t_{sw}
t_{sw} counter
$V_{\text{ind}} \cdot t_{\text{sw}}$
$K_2 \cdot t_{\text{hp}}$
Reset
$(V_{\text{ind}} \cdot t_{\text{sw}}) \geq (K_2 \cdot t_{\text{hp}})$
$START\_TSW_i$
END_TSW_i

Fig. 11. Phase-i arithmetic unit detailed block diagram.

1) Commutation Rules: Commutation rules provide information about the optimal action that the control should perform when the i_e zero-crossing is detected, or when t_{sw} has elapsed.

As shown in Fig. 6, the commutation is enabled only when there is a coincidence between the sign and the slope of i_e, in order to ensure the correct i_{Ref} tracking. As the sign of the i_{ei} slope is determined by the PWM_i signal, and C_i signal indicate the i_{ei} sign, they can be combined to determine whether the commutation is enabled or not. The switching enable flag for phase $i\text{SW\_EN}_i$ is, therefore, implemented by performing the exclusive NOR (XNOR) function between PWM_i and C_i signals

$$\text{SW\_EN}_i = \overline{\text{PWM}_i \oplus C_i}. \tag{12}$$

SW_EN_i is, therefore, used by the commutation state machine to decide whether the PWM_i state must be changed or not when (11) is satisfied.

As previously stated, an anticipated commutation should be performed in order to optimize the transient response if the synchronization error $t_e(k) > T_{\text{Sync}}/4$. This condition can be determined by performing the comparison between the synchronization error and the constant $T_{\text{Sync}}/4$ by means of a digital comparator, when a zero-crossing in i_e is detected.

The detection of the aforementioned condition can be optimized by avoiding the implementation of the digital

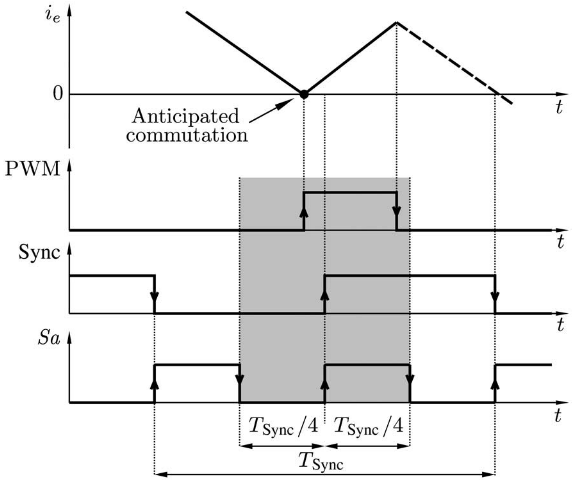


Fig. 12. Determination of the anticipated commutation condition.

comparator. $\mathrm{Ramp}i$ counter provides the synchronism signal by taking its MSB bit, the next bit will, therefore, generate a signal S_{a_i} with half the period of Sync$i$, as shown in Fig. 12. As it can be seen, as S_{a_i} level is constant at $T_{\text{Sync}}/4$ intervals, it can be combined with $\mathrm{Sync}i$ and PWM_i signals in order to determine whether the synchronization error is greater than $T_{\text{Sync}}/4$. If this scenario is detected, $\text{ANT\_COMM}_i = 1$ indicates that an anticipated commutation should be performed in order to optimize the transient response. All possible combinations among these three signals are summarized in Table II, which is implemented as the three-input XNOR function shown in (13). ANT_COMM_i state is verified by the commutation state machine at every i_e zero-crossing, in order to determine if the anticipated commutation must be performed.

This methodology allows to reduce the required resources with respect to the implementation of a comparator and it is independent from the counter module

$$\text{ANT\_COMM}_i = \overline{\mathrm{PWM}_i \oplus \mathrm{Sync}i \oplus S_{a_i}}. \qquad (13)$$

2) Commutation State Machine: This block controls the state of the PWM_i output and starts the t_{sw} computation when necessary. Fig. 13 depicts the phase-i state machine model, based on the following input signals.

1) $X_0 = \text{END\_TSW}_i$: Indicates the instant in which (11) is satisfied.
2) $X_1 = \text{SW\_EN}_i$: Indicates if the PWM_i switching is enabled.
3) $X_2 = C_i$: Sign of i_{ei}.
4) $X_3 = \text{ANT\_COMM}_i$: Indicates if an anticipated commutation is necessary when the i_{ei} zero-crossing is detected.

Fig. 13 also depicts the phase-*i* state machine model, based on the following outputs.

1) PWM_i: Phase-i switch driving signal.
2) START_TSW_i: Indicates to the arithmetic unit that t_{sw} calculation must be initiated.

The transition between the four states is determined by the input signals and the current state. The state machine remains in the states S_0 and S_2 until a change in the input X_2 is detected, indicating a zero-crossing in i_e. Once the zero-crossing is detected, the state machine changes the state to S_1 or S_3, depending on X_3 input, which indicates if an anticipated commutation is necessary. The state machine stays on S_1 or S_3 until the end of t_{sw}, which is indicated by the input $X_0 = 1$. The system shifts to states S_0 or S_2 depending on whether the switching is enabled or not (X_1 input state). Table III lists the state transitions corresponding to Fig. 13.

TABLE II
ANT_COMM_i TRUTH TABLE

PWM_i	Sync$i$	S_{a_i}	ANT_COMM_i
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

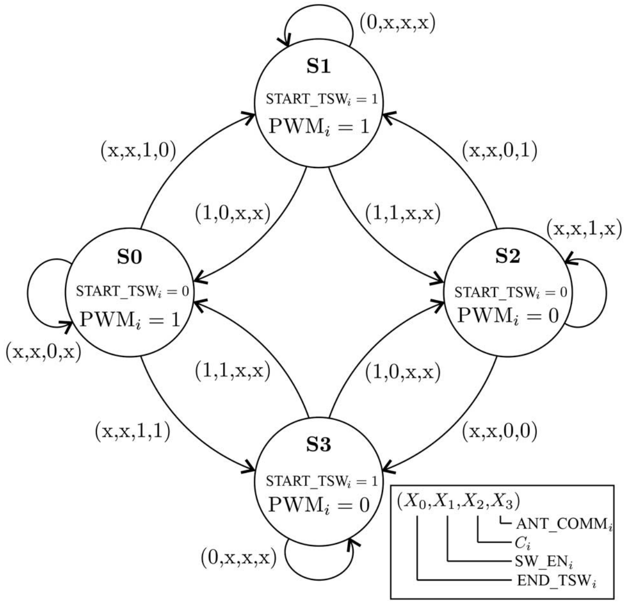


Fig. 13. Phase-i commutation state machine diagram.

It should be noted that the phase control structure described is independent of the control of the remaining phases, as no information or resources are shared among them. By making use of this characteristic and taking advantage of FPGA parallelization capacity, it is possible to obtain a modular and scalar multiphase control.

IV. Design Evaluation

In order to evaluate the control optimization, the design was implemented on an Xilinx FPGA Spartan 3E XC3S1600E. This device can run at synchronous system clock rates of up to 200 MHz and supports 14 752 slices. This FPGA contains dedicated resources such as 8 digital clock managers (DCMs) and 36 18-bit embedded multipliers.

TABLE III
DESCRIPTION OF STATE TRANSITIONS

Present sate	Inputs $X_{[0\cdots3]}$	Future state
S_0	XX0X	S_0 does not detect zero crossing
PWM$_i$= 1	XX10	S_1 zero crossing with correct slope
START_TSW$_i$ = 0	XX11	S_3 zero crossing with incorrect slope
S_1	0XXX	S_0 t_{sw} does not end
PWM$_i$= 1	10XX	S_1 end of t_{sw}. Switching disabled
START_TSW$_i$ = 1	11XX	S_3 end of t_{sw}. Switching enabled
S_2	XX1X	S_0 does not detect zero crossing
PWM$_i$= 0	XX00	S_1 zero crossing with correct slope
START_TSW$_i$ = 0	XX01	S_3 zero crossing with incorrect slope
S_3	0XXX	S_0 t_{sw} does not end
PWM$_i$= 0	10XX	S_1 end of t_{sw}. Switching disabled
START_TSW$_i$ = 1	11XX	S_3 end of t_{sw}. Switching enabled

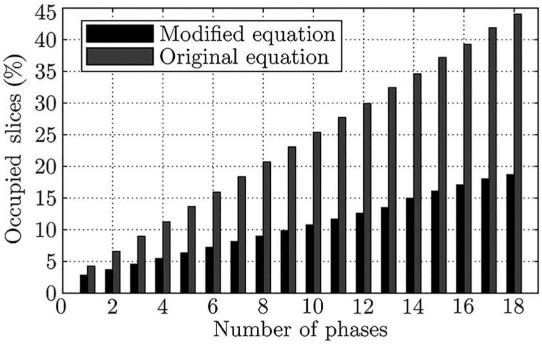


Fig. 14. Spartan 3E resources utilization with and without optimization.

The design evaluation consists of two experimental tests.

1) The design optimization is evaluated by comparing the optimized and nonoptimized implementations.
2) The behavior of the optimized current control implementation is evaluated on a three-phase interleaved buck converter.

The optimizations described in the several control tasks, such as the optimization of (10) and the commutation rules depicted in (12) and (13), were implemented and compared to the nonoptimized version of the control implementation. Fig. 14 shows the percentage of occupied slices, when different number of phases are implemented with the original equation (10), and the modified equation (11) using $b = 10$ bits for both cases. As shown, by avoiding the implementation of the division in (10), the device utilization is greatly improved. On the other hand, the maximum clock frequency that meets the timing constraints is approximately $\mathrm{f}_{\mathrm{CLK}_{\mathrm{MAX}}} = 60$ MHz for both implementations. This frequency can, nevertheless, be significantly increased by switching to faster families of FPGAs. Xilinx Spartan 6, for instance, yields a maximum clock frequency of $\mathrm{f}_{\mathrm{CLK}_{\mathrm{MAX}}} =$ 250 MHz when the same project is implemented.

In order to evaluate the implemented algorithm behavior on a practical buck converter, experimental tests were carried out on a system whose main parameters are listed in Table IV. The experimental tests are intended to verify the proposed current control implementation.

TABLE IV
POWER CONVERTER AND IMPLEMENTATION CHARACTERISTICS

Parameters	
Number of phases (N)	3
Phase inductance (L_f)	250 µH
Input voltage (V_{in})	30 V
Maximum output current (I_{MAX})	10 A
Load resistance (R_L)	2 Ω
Load capacitance (C_L)	40 µF
Resolution bits (b)	10 bits
Switching frequency (f_S)	12.2 KHz
V_0 and V_{in} sample period (T_{Sample})	13.6 µs

By using the parameters listed in Table IV, 669 out of 14 752 slices, 6 embedded multipliers, and 1 DCM are used. Additionally, the phase shift error can be calculated for this implementation using (7), as the counter module is not divisible by N (14). This result should be added to the error produced by turn-ON and turn-OFF delays in the switching devices and zero-crossing detection delays

$$e_\phi = 2\pi\frac{1}{2^{10}} = 6.14 \times 10^{-3} = 0.35^\circ. \tag{14}$$

Phase current is measured with LEM LA-25NP hall-effect current transducers. Transducer output current i_s is sensed by using $R_s = 100\ \Omega$ shunt resistor. Voltage drop in the shunt resistor $V_s = i_s R_s$ is measured with the AD8250 instrumentation amplifier. This amplifier features high common-mode voltage rejection up to high frequency, thus, reducing modules crosstalk and improving noise margin. Current error i_e is generated by substracting V_s from $V_{i_{\mathrm{Ref}}}$, which represents the current reference in the same units as V_s, using the same type of instrumentation amplifier. Current error zero-crossing detection is performed using analog voltage comparators. The hysteresis band present in said comparators is small enough to prevent unwanted switching, without adding significant delay to the zero-crossing detection.

In order to evaluate the different commutation rules, generated in the commutation unit, and the state machine operation, one phase current response to an i_{Ref} step is evaluated. Fig. 15 depicts the current evolution of phase 1, along with internal control signals such as Sync, S_a, C, and the PWM output. As it can be seen, before the i_{Ref} step, C and Sync signals are synchronized. At time t_0, the system calculates the next switching time for instant t_1 using (11). Equation (12) is not satisfied at t_1 as $C \neq \mathrm{PWM}$; the commutation is, therefore, disabled and the system remains in the current state until the zero-crossing at t_2 is detected. As the zero-crossing at t_2 occurs close to a synchronism edge of opposite sign, which can be determined by using (13), the system performs an anticipated commutation in order to recover the synchronized condition at t_3. The ripple amplitude and duty cycle modification that can be noticed in Fig. 15 are produced by the load voltage variation produced by the i_{Ref} modification.

In order to verify the multiphase system performance, a three-phase current control was implemented. This test is aimed to evaluate the correct current control of each phase and its

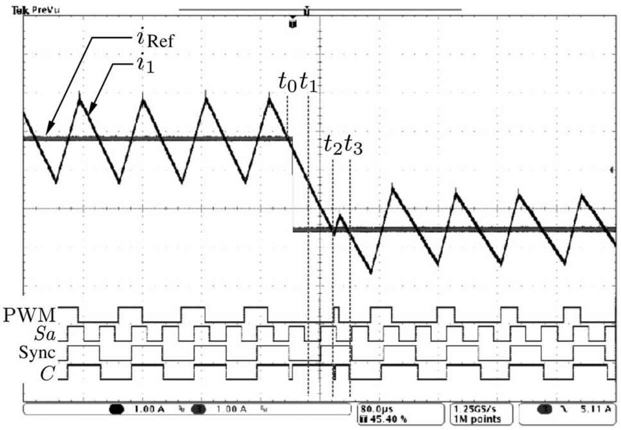


Fig. 15. Single-phase i_{Ref} step tracking. Phase current, i_{Ref} and digital control signals.

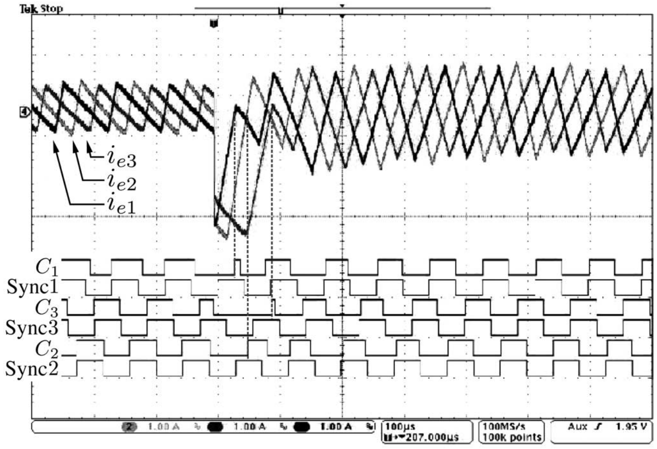


Fig. 16. Three-phase i_e transient response to i_{Ref} step change. Current errors, zero-crossing and synchronism signals.

independence from the remaining phases. Additionally, a step change in i_{Ref} is produced and the multiphase transient behavior is evaluated. Fig. 16 shows the current errors $i_{e1,2,3}$, the synchronism and the zero-crossing signals for the converter described. The phase current error is evaluated instead of the phase current, since it provides better insight of the transient error and recovery. It should be noted that, due to the operation of the converter in current mode, i_{Ref} perturbation produces variations in V_0. Duty cycle and ripple amplitude change in agreement with the output capacitance charge after the current step is produced. As it can be seen, the commutation instants of a given phase are close to the zero-crossing instants of the remaining phases. The sense circuit, however, minimizes crosstalk between phases, which avoids the commutation of one phase to affect the zero-crossing detection of the remaining ones.

With respect to the multiphase transient behavior, the response of each phase current error to an i_{Ref} step is evaluated. Initially, in Fig. 16, the edges of each phase i_e zero-crossing signals are synchronized with the edges of the corresponding synchronization signal. This synchronized condition produces the correct phase interleaving in order to obtain optimal total ripple attenuation. After the i_{Ref} step change occurs, however, the synchronized condition is temporarily lost. When the first i_e zero-crossing after the i_{Ref} step is detected, the control of each phase determines whether an anticipated commutation is needed. As it can be seen, phases 1 and 3 perform anticipated commutation as i_{e1} and i_{e3} zero-crossing points are close to the synchronism signal corresponding to the opposite sign. The control of each phase recovers the synchronized condition in the next zero-crossing point by calculating the switching time using (10). As it can be seen, the small synchronization error, present after the interleaved condition is recovered, is corrected in each zero-crossing until the output capacitance is fully charged and the voltage reaches the steady-state condition.

V. Conclusion

The digital control of power converters poses challenging issues when efficient implementations are required. Particularly, the complexity of multiphase converters increases with the number of required phases. Additionally, the wide range of applications of this type of converters adds flexibility requirements to the current control algorithm implementation. This work presented the FPGA implementation of a previously developed current control algorithm for multiphase converters. A modular approach with independent control for each phase, easily applicable to a particular number of phases, was developed with emphasis on the resource optimization of several control tasks. Arithmetic operations were adapted to profit from the FPGA architecture. Additionally, the implementation of complex digital devices, such as magnitude comparators, were avoided by performing different cases detection with logic functions. Although designed for a particular control algorithm, the proposed implementation and its optimizations could be extended and applied to different control algorithms. The proposed design and optimizations were validated on a three-phase buck converter.

References

[1] M. Abusara and S. Sharkh, “Design and control of a grid-connected interleaved inverter,” *IEEE Trans. Power Electron.*, vol. 28, pp. 748–764, Feb. 2013.

[2] L. Ni, D. Patterson, and J. Hudgins, “High power current sensorless bidirectional 16-phase interleaved dc–dc converter for hybrid vehicle application,” *IEEE Trans. Power Electron.*, vol. 27, pp. 1141–1151, Mar. 2012.

[3] O. Hegazy, J. Van Mierlo, and P. Lataire, “Analysis, modeling, and implementation of a multidevice interleaved dc/dc converter for fuel cell hybrid electric vehicles,” *IEEE Trans. Power Electron.*, vol. 27, pp. 4445–4458, Nov. 2012.

[4] H. Choi, “Interleaved boundary conduction mode (BCM) buck power factor correction (PFC) converter,” *IEEE Trans. Power Electron.*, vol. 28, pp. 2629–2634, Jun. 2013.

[5] J. Quintero, A. Barrado, M. Sanz, C. Fernandez, and P. Zumel, “Impact of linear nonlinear control in multiphase VRM design,” *IEEE Trans. Power Electron.*, vol. 26, pp. 1826–1831, Jul. 2011.

[6] B. Akin, M. Bhardwaj, and S. Choudhury, “An integrated implementation of two-phase interleaved PFC and dual motor drive using single MCU with CLA,” *IEEE Trans. Ind. Informat.*, vol. 9, pp. 2082–2091, Nov. 2013.

[7] P. Antoszczuk *et al.*, “Characterization of steady-state current ripple in interleaved power converters under inductance mismatches,” *IEEE Trans. Power Electron.*, vol. 29, no. 4, pp. 1840–1849, 2014.

[8] T. Ohata, S. Kirikawa, and T. Saito, “Fault tolerance of simplified parallel power converters with current sharing function,” in *Proc. IEEE Asia Pacific Conf. Circuits Syst. (APCCAS)*, 2012, pp. 104–107.

[9] S. Zhang and X. Yu, "A unified analytical modeling of the interleaved pulse width modulation (PWM) dc–dc converter and its applications," *IEEE Trans. Power Electron.*, vol. 28, pp. 5147–5158, Nov. 2013.

[10] O. Garcia, P. Zumel, A. De Castro, and J. Cobos, "Automotive dc–dc bidirectional converter made with many interleaved buck stages," *IEEE Trans. Power Electron.*, vol. 21, pp. 578–586, May 2006.

[11] O. Garcia, A. de Castro, P. Zumelis, and J. Cobos, "Digital-control-based solution to the effect of nonidealities of the inductors in multiphase converters," *IEEE Trans. Power Electron.*, vol. 22, pp. 2155–2163, Nov. 2007.

[12] T. Saito, S. Tasaki, and H. Torikai, "Interleaved buck converters based on winner-take-all switching," *IEEE Trans. Circuits Syst. I*, vol. 52, pp. 1666–1672, Aug. 2005.

[13] Y. Qiu, J. Sun, M. Xu, K. Lee, and F. C. Lee, "Bandwidth improvements for peak-current controlled voltage regulators," *IEEE Trans. Power Electron.*, vol. 22, no. 4, pp. 1253–1260, Jul. 2007.

[14] R. Foley, R. Kavanagh, and M. Egan, "Sensorless current estimation and sharing in multiphase buck converters," *IEEE Trans. Power Electron.*, vol. 27, pp. 2936–2946, Jun. 2012.

[15] R. Li, "Modeling average-current-mode-controlled multi-phase buck converters," in *Proc. IEEE Power Electron. Spec. Conf. (PESC)*, 2008, pp. 3299–3305.

[16] Y. Zhang, M. Yu, F. Liu, and Y. Kang, "Instantaneous current-sharing control strategy for parallel operation of UPS modules using virtual impedance," *IEEE Trans. Power Electron.*, vol. 28, pp. 432–440, Jan. 2013.

[17] W. Guo and P. K. Jain, "Analysis and modeling of voltage mode controlled phase current balancing technique for multiphase voltage regulator to power high frequency dynamic load," in *Proc. 24th Annu. IEEE Appl. Power Electron. Conf. Expo. (APEC'09)*, 2009, pp. 1190–1196.

[18] S. Maestri *et al.*, "Improved control strategy for active bouncers used in klystron modulators," in *Proc. 2011–14th Eur. Conf. Power Electron. Appl. (EPE'11)*, 2011, pp. 1–7.

[19] N. Wassinger *et al.*, "Multiple-stage converter topology for high-precision high-current pulsed sources," *IEEE Trans. Power Electron.*, vol. 26, pp. 1316–1321, May 2011.

[20] E. Dallago *et al.*, "The power supply for the beam chopper magnets of a medical synchrotron," in *Proc. 37th IEEE Power Electron. Spec. Conf. (PESC'06)*, 2006, pp. 1–5.

[21] R. Garcia Retegui, M. Benedetti, M. Funes, P. Antoszczuk, and D. Carrica, "Current control for high-dynamic high-power multiphase buck converters," *IEEE Trans. Power Electron.*, vol. 27, pp. 614–618, Feb. 2012.

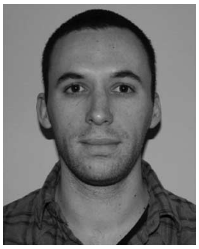
[22] E. Monmasson *et al.* "FPGAs in industrial control applications," *IEEE Trans. Ind. Informat.*, vol. 7, pp. 224–243, May 2011.

[23] M. P. Aguirre, L. Calvino, and M. I. Valla, "Multilevel current-source inverter with FPGA control," *IEEE Trans. Ind. Electron.*, vol. 60, pp. 3–10, Jan. 2013.

[24] C. Buccella, C. Cecati, and H. Latafat, "Digital control of power converters—A survey," *IEEE Trans. Ind. Informat.*, vol. 8, pp. 437–447, Aug. 2012.

[25] N. Wassinger, R. Retegui, M. Funes, and M. Benedetti, "Digital control for a multiple-stage pulsed current source," *IEEE Trans. Ind. Informat.*, vol. 9, pp. 1122–1129, May 2013.

[26] J. Juarez-Abad, J. Linares-Flores, E. Guzman-Ramirez, and H. Sira-Ramirez, "Generalized proportional integral tracking controller for a single-phase multilevel cascade inverter: An fpga implementation," *IEEE Trans. Ind. Informat.*, vol. 10, pp. 256–266, Feb. 2014.



Pablo D. Antoszczuk (S'11) was born in Argentina, in 1985. He received the Electronics Engineering degree from the Universidad Nacional de Mar del Plata (UNMdP), Mar del Plata, Argentina, in 2010, where he is currently working toward the Ph.D. degree in electronics engineering.

He is a Graduate Student Researcher with the National Scientific and Technical Research Council (CONICET), Buenos Aires, Argentina. He is a Researcher with the Instrumentation and Control Laboratory (LIC), UNMdP. His research interests include power converters, current control, and high-precision measurements.

Rogelio Garcia Retegui was born in Argentina, in 1977. He received the Ph.D. degree in electronics engineering from the Universidad Nacional de Mar del Plata (UNMdP), Mar del Plata, Argentina, in 2002 and 2009, respectively.

He is currently working as a Researcher with the Instrumentation and Control Laboratory (LIC), UNMdP, and he is an Assistant Researcher with the National Scientific and Technical Research Council (CONICET), Buenos Aires, Argentina. Since 2003, he has been an Assistant Professor with the Control Theory and Control Systems Course, UNMdP. His research interests include power electronics, current control, and pulsed-power converters for particle accelerators.

Marcos Funes (M'12) was born in Argentina, in 1974. He received the Ph.D. degree in electronics engineering from the Universidad Nacional de Mar del Plata (UNMdP), Mar del Plata, Argentina, in 1999 and 2007, respectively.

He is an Assistant Researcher with the Instrumentation and Control Laboratory (LIC), UNMdP, since 1999, and with the National Scientific and Technical Research Council (CONICET), Buenos Aires, Argentina, since 2009. His research interests include high-density programmable logic devices, power converters control, power line communication, and digital signal processing.

Daniel Carrica (S'85–M'95–SM'08) received the B.S. degree in engineering from the Universidad Nacional de Mar del Plata (UNMdP), Mar del Plata, Argentina, in 1984; the M.Sc. degree in electronics from the Universidad Politcnica de Madrid, Madrid, Spain, in 1992; and the Ph.D. degree in electronics engineering from the UNMdP, in 2006.

He is currently the Director of the Instrumentation and Control Laboratory (LIC), UNMdP. He is a Researcher with the National Scientific and Technical Research Council (CONICET), Buenos Aires, Argentina. His research interests include smart grids and power electronics.