Parametrizable fixed-point arithmetic for HIL with small simulation steps

Alberto Sanchez*, Angel de Castro, *Member, IEEE,* and Javier Garrido, *Member, IEEE*
Universidad Autónoma de Madrid, Madrid, Spain
*alberto.sanchezgonzalez@uam.es

Paper contains original research and has not been submitted/published earlier in any journal and it is not being considered for publication elsewhere.

***Abstract*—HIL techniques are increasingly used for test purposes because of their advantages over classical simulations. FPGAs are becoming popular in HIL systems because of their parallel computing capabilities. In most cases, FPGAs are mainly used for signal processing like input PWM sampling and conditioning while there are also processors to model the system. However, there are other HIL systems that implement the model in the FPGA. For FPGA implementation and regarding the arithmetics, there are two main possibilities: fixed-point and floating-point. Fixed-point is the best choice only when real-time simulations with small simulation steps are needed, while floating-point is the common choice because of its flexibility and ease of use. This paper presents a novel hybrid arithmetic for FPGAs called parametrizable fixed-point which takes advantage of both arithmetics as the internal operations are accomplished using simple signed integers while the point location of the variables can be adjusted as necessary without redesigning the model of the plant. Experimental results show that a buck converter can be modeled using this novel arithmetic with a simulation step below $20\ ns$. Besides, the experiments prove that the proposed model can be adjusted to any set of values (voltages, currents, capacitances, etc.) keeping its accuracy without resynthesizing, showing the big advantage over fixed-point arithmetic.**

***Index Terms*—emulation, fixed-point arithmetic, floating-point arithmetic, field programmable gate array**

I. Introduction

Digital control for power electronics has grown during the past few decades [1]–[8], so the need of new test techniques has also increased. Meanwhile, the advances in HIL (Hardware-in-the-Loop) have made it viable as a test technique for power electronics. HIL allows the designer to test, even in real-time, the controller along with a model of the plant. Besides, HIL simulators offer the opportunity to debug the controller in its final implementation. This is possible because the HIL system can be a different device and its inputs and outputs can be driven to the final controller, offering a reliable simulation. HIL systems have presented a significant growth in the academic but also in the commercial world. [9] offers an extensive review of the simulation alternatives for microgrids and it manifests the consolidated use of HIL in power electronics.

First HIL systems were based on computers, reaching simulations with an integration step of $50\ \mu s$ [10]. However, some current HIL systems use FPGAs (Field Programmable Gate Array) because their parallel nature makes it possible to make numerous calculations concurrently, so the integration step can be decreased dramatically [11]–[18]. Another common choice is using FPGAs along with processors to create the HIL systems, as shown in [19]. In these cases, FPGAs are often used for tasks that can be executed in parallel, like input/output sampling and conditioning, while the system also uses real-time processors. For instance, commercial tools like Opal-RT and dSPACE use both devices in their systems [19].

The main drawback of the HIL systems that use FPGAs to implement the model is the complexity of the model design. The implementation of HDL (Hardware Description Language) code is not trivial and the arithmetic type used inside the model determines many factors such as the simulation step, the ease of design and the resources used in the FPGA.

One possibility is to design the model with high-level tools such as Matlab models, and translate it into HDL code using automatic tools [12], [13]. There are also commercial tools such as Opal-RT, Typhoon HIL and dSPACE which let the designer to create graphical models and use them for simulation with the help of FPGAs. Commercial FPGA-based HIL systems can handle complex models, defined by the user, with an integration step of hundreds of nanoseconds, almost without requiring user optimization. However, commercial tools are expensive (generally over 10,000 USD) and, then, their use is not justified in many low cost applications, especially if the test scenarios can be easily reproduced in a laboratory. On the other hand, commercial tools are almost mandatory when the system under test is a complex one [20], [21]. As examples of complex system tested with commercial HIL tools, in [20], the Typhoon HIL system is used to test a system which mitigates grid background harmonics for photovoltaic inverters. Also in [21], OPAL-RT platform is used to verify the controller of a fast charger for electric vehicles.

Commercial HIL systems facilitate the model implementation but the performance results are not optimal, both in resources and simulation step. The main reason is that commercial tools use floating-point arithmetic. Floating-point is easy to use because the point location of every variable is dynamically shifted when necessary changing the exponent field of the number, so it is not necessary to control the point location, the numerical representation limits, etc. However, the overhead of floating-point internal operations is noteworthy.

When it is necessary to improve the simulation step to increase the simulation accuracy, it is possible to hand-code the model of the plant. In this case, the election of the arithmetic

becomes even more important. Again, floating-point arithmetic is easier to use and the translation from the equations of the model to the code is straightforward in most cases. For instance, in [22] a HIL system is presented which models a multi-inductor domestic induction heating platform using floating-point arithmetic. However, if the simulation step is critical, fixed-point models should be used. Authors of [23] showed a comparison between several arithmetics to model a power converter, and they showed that fixed-point was ten times faster while it required much less resources than floating-point using the synthesis tools of that moment. The experimental results of Section IV show that present synthesis tools have decreased the gap between both arithmetics, but there are still important differences, ranging between three and five times in terms of both speed and resources.

Even with these advantages, fixed-point may not be viable if the model is not simple because of the effort of designing optimized fixed-point models. The main reason is that the designer should decide where the point of every variable should be located, as the point location cannot be shifted after the implementation of the model. Therefore, once implemented, the model only works inside a range of values and any further change will need a redesign of the model. Considering this limitation, fixed-point models only can be used in specific applications.

This paper presents how to model a power converter using parametrizable fixed-point arithmetic. This arithmetic is based on fixed-point arithmetic, taking advantage of the speed of fixed-point, but it allows the model to shift the point location without resynthesizing. The point location shifting gets rid of the numerical limitations of fixed-point, so the model can be adapted to any simulation range of values. Although the design effort is not negligible compared with floating-point, the advantages of this type of arithmetic are significative, so it is a viable choice to implement models with small simulation steps.

The rest of the article is organized as follows: Section II shows how to model a power converter using difference equations. Section III describes the proposed parametrizable fixed-point arithmetic and how the equations are translated to a digital model coded in HDL. Experimental results are shown in Section IV making a comparison between traditional and parametrizable fixed-point. Finally, conclusions are given in Section V.

II. Modeling a power converter

In this section, it is described how to create a simple high-speed model of a buck converter (see Fig. 1), but any converter can be modeled using this methodology. The model must calculate the values of the state variables every simulation step. In this case, the state variables are the capacitor voltage and the inductor current. The converter can be modeled analyzing the equations of the capacitor and the inductor, which are differential equations:

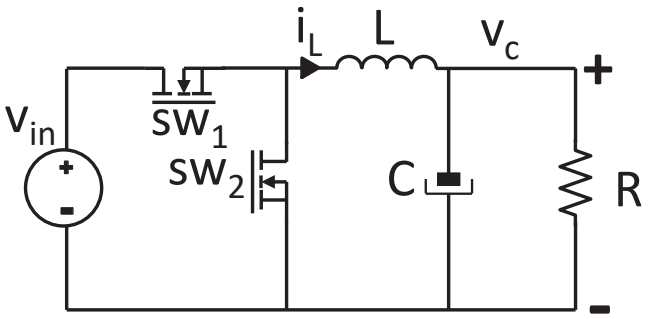

Fig. 1: Buck converter topology used as application example

$$\begin{aligned} v_L &= L \cdot \frac{di_L}{dt} \\ i_C &= C \cdot \frac{dv_c}{dt} \end{aligned} \tag{1}$$

where L is the inductance, v_L and i_L are the inductor voltage and current respectively, C is the capacitance and i_C and v_C are the capacitor current and voltage respectively. In order to create a simple but real-time model, Explicit Euler method can be used so these difference equations can be extracted:

$$\begin{aligned} i_L(k) &= i_L(k-1) + \frac{\Delta t}{L} \cdot v_L(k-1) \\ v_C(k) &= v_C(k-1) + \frac{\Delta t}{C} \cdot i_C(k-1) \end{aligned} \tag{2}$$

where Δt is the simulation step, which is fixed so the system is less complex but faster. As it can be seen, the inductor current depends on the inductor voltage, and the capacitor voltage depends on the capacitor current, but both are determined by the state of the switches. Therefore, there are two pairs of equations that must be solved by the model every simulation step; when the upper switch is closed (3) and when the lower switch is closed (4):

$$\begin{aligned} v_L(k-1) &= v_{in}(k-1) - v_C(k-1) \\ i_C(k-1) &= i_L(k-1) - i_R(k-1) \end{aligned} \tag{3}$$

$$\begin{aligned} v_L(k-1) &= -v_C(k-1) \\ i_C(k-1) &= i_L(k-1) - i_R(k-1) \end{aligned} \tag{4}$$

If both switches are open, commonly used to add dead-times, and the inductor current is positive, equation (4) will be used and (3) otherwise. If the inductor current is identically zero when both switches are open, there is a third case in which the inductor current remains zero (no increment nor decrement). Electrical losses have not been included for the sake of clarity but the methodology proposed in this paper is applicable to them.

A. Arithmetic possibilities

The equations seen in the previous section will be implemented in a real-time model using an FPGA. The election of the arithmetic will determine several factors such as the minimum simulation step achievable by the model, the design effort, the size of the design, etc. The first election lies in choosing between floating and fixed-point arithmetics, but a specific data type should be chosen. For instance, in VHDL some of the possibilities are:

- Real arithmetic. It is a standard numeric type of VHDL which uses double-precision floating-point format. It cannot be synthesized but it is useful to create a first approximation and it can be used as a reference model because its numerical error should be negligible.
- Float arithmetic. This type, implemented in the VHDL2008 *float_pkg* package [24], provides synthesizable floating-point arithmetic. It has the versatility of the floating-point, but it uses many hardware resources. Also, if the simulation step is small, 32-bit floating-point, which is the most common floating-point format, may not have enough resolution, as it can be seen in [23]. Besides, floating-point arithmetic presents poor time performance, reaching simulation steps several times higher than using fixed-point as it will be shown in the experimental results of Section IV.
- Fixed arithmetic. An alternative to float arithmetic is fixed-point arithmetic. Using this notation, both the point location and the data width are fixed when the model is implemented so the designer must decide the number of bits to the integer and fractional parts of every variable. Besides, as some mathematical operations require point alignment, the designer must reformat the variables prior to that operation. Therefore, this arithmetic requires much more design effort.

Due to its versatility, floating-point may be the best arithmetic format to use in an HIL system but only if the simulation step can be above a threshold that may vary between $100\ ns$ and $1\ \mu s$ depending on the technology and model complexity. Below that threshold, two problems may arise. On the one hand, the design implemented in the FPGA may not be executed in real time due to the complexity of floating-point. On the other hand, small simulation steps imply also small increments in the equation (2). Those small increments may produce resolution issues, obtaining non-functional simulations, as it was seen in [23]. Nevertheless, obtaining a small input PWM (Pulse Width Modulation) sampling step is crucial because it allows the model to be accurate even with high switching-frequency converters. It is important to note that the PWM sampling and simulations steps do not have to be numerically equal. In fact, present commercial tools have drastically decreased the PWM sampling step down to nanoseconds or tens of nanoseconds while the simulation step remains about hundreds of nanoseconds. For instance, in Typhoon HIL602 and HIL402, the PWM sampling step is $20\ ns$ and the simulation step is down to $500\ ns$ in the case of HIL602. It should be noted that the PWM sampling step must be around one hundred times (resolution around 1%) below the switching period in order to detect accurately the switching inputs. In the literature, examples can be found from a minimum of 20 samples per input PWM period [25], to hundreds [26] or even thousands [27] of samples per period. Those times cannot be reached for the simulation step using floating-point arithmetic, and that is why both steps are very different in commercial tools. Anyway, simulation steps need to remain well below the switching period, so they should be also decreased to be able to emulate converters with switching periods of $1\ \mu s$ or less. This will become more important with the increasing use of GaN SiC devices, as the switching periods have a tendency to decrease [28], [29].

Taking all into account, fixed-point arithmetic should be the election if the simulation step must be minimized. However, the need of determining the numerical limits of every variable to calculate the width and point location is a big drawback. Besides, another shortcoming is that the model must be redesigned every time the simulation conditions are changed, because the numerical limits may be surpassed. One strategy is to transfer fractional bits to the integer part so even the worst case can be simulated, but that entails resolution losses and the model probably becomes unusable. Therefore, redesigning the model is the only way to maximize both time step and resolution. All these drawbacks usually make fixed-point HIL models unfeasible to be commercialized.

This paper proposes a parametrizable fixed-point notation which takes simultaneously the advantages of floating and fixed-point. The aim is to obtain the small simulation step and hardware utilization of fixed-point arithmetic but also the flexibility of floating-point, which does not require the redesign of the model when the simulation conditions change.

III. Parametrizable fixed point arithmetic

As it was explained in the previous section, the main drawback of the standard fixed-point arithmetic is its inability to shift the point location when needed. It implies resolution issues when there are less fractional bits than necessary and saturation when the variables exceed the numerical limits determined by the format. Therefore, when the simulation conditions change (changing the inductance/capacitance values, the input voltage range, or any other value), it is necessary to redesign the model, calculating the number of bits for the integer and fractional parts.

In this paper a novel technique which uses parametrizable fixed point arithmetic is proposed. It consists on fixed-point notation but the point location is shifted without resynthesizing the code. In order to maximize the performance of the model, simple signed integers operations are implemented. The widths of the variables are constant and they can be chosen taking into account the DSPs (Digital Signal Processors) located in the FPGA, to obtain the minimum critical path — minimum delay between two simulation steps.

Although the arithmetic operators process the numbers as integers, the variables actually are in fixed-point format, so a number of integer and fractional bits must be allocated for every signal. The point location of every signal must be calculated using this equation:

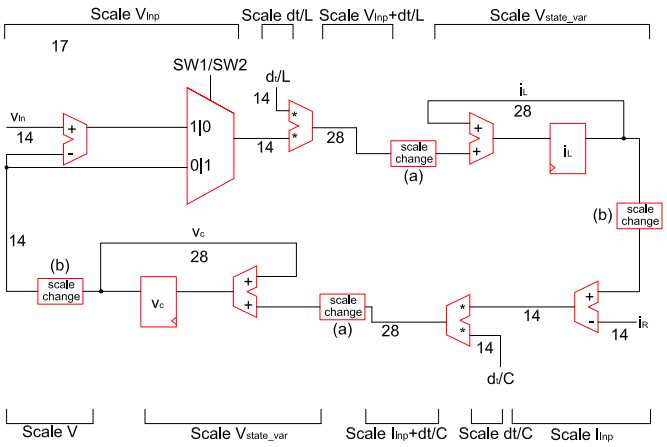

Fig. 2: Model schematic implemented using parametrizable fixed-point arithmetic

$$scale \quad = \quad width - \lceil \log_2 max\_value \rceil \tag{5}$$

where $scale$ is the number of bits for the fractional part. The previous equation determines that the scale is equal to the total width minus the number of integer bits needed to store the maximum expected value (worst case for the present simulation execution). For instance, if the user knows that the maximum output voltage is $400\ V$ and the state variable has 27 bits (28 including the sign bit), the scale of the output voltage variable will be 18 — because 9 bits are needed for the integer part — so the variable has 18 fractional bits with a resolution of $3.81 \cdot 10^{-6}\ V$. However, if another simulation should be done with lower voltage values like $5\ V$, the variable will have 24 decimal digits, so the resolution will be $5.96 \cdot 10^{-8}\ V$. Therefore, the system is adapted to the expected values in order to maximize the simulation accuracy. It is important to note that maximizing the numerical resolution is crucial as the increments of equation (2) are very small because the simulation step (Δt) will be decreased down to tens of nanoseconds using the proposed arithmetic.

As it can be seen in Fig. 2, the widths of every variable are fixed during the simulation. The scale of every variable should be calculated using (5), but this can be accomplished without the need of resynthesizing. This task can be done automatically as the user only has to define the maximum values of the main model variables (input and output voltages and currents) and the values of the converter (inductance, capacitance, actual input voltage), etc. This information should be known by the simulation user, because the control which will be tested should have been designed using this information. The actual values can vary during the simulation but they should not exceed the numerical limits defined before the simulation. Only when the parameters of the converter are changed, the scales should be calculated again, but without any resynthesis, which is the main advantage of the parametrizable approach.

Regarding the implementation of equations in VHDL code, it must be taken into account that two variables should have the same scale (number of fractional bits) before being added or subtracted, and the result will have the same scale. However, two signals can be multiplied no matter the scale of them, and the result will have as scale the sum of both scales. In order to fulfill the first requirement, four scale changer blocks are added as it will be explained later. The schematic of the model using parametrizable fixed-point arithmetic can be seen in Fig. 2. The scale changers labeled as *(a)* allow the model to calculate with maximum resolution the increments of the state variables. In this way, the $\frac{\Delta t}{L}$ and $\frac{\Delta t}{C}$ can be defined using the maximum number of fractional bits. However, if the scale of the increments is different from the scale of the state variables, the scale changer will translate the increments before the addition.

On the other hand, the accumulators of the state variable are implemented with high resolution but the model feedback (the current depends on the voltage and *vice versa*) is accomplished using less fractional bits. For this task, two more scale changers labeled as *(b)* are included in the design. It can be thought that this truncation would affect severely the whole system accuracy but it should be noted that, while the integration is highly sensitive to resolution because the incremental values are very small, the important information in the feedback is how big the current and voltage variables are. This is because, the feedback information is multiplied again for $\frac{\Delta t}{C}$ or $\frac{\Delta t}{L}$. For example, considering that the resolution of both state variables is 10^{-15}, there is no need to feedback the current with high resolution (for example $4.0 + 1 \cdot 10^{-15}\ A$) because that value, after being subtracted by the load current ($1\ A$ for instance), will be multiplied by $\frac{\Delta t}{C}$ (for example $\frac{20\ ns}{100\ \mu F}$), so the capacitor incremental value will be $6.000000000000002 \cdot 10^{-4}\ V$ and it will be truncated to $6.0 \cdot 10^{-4}\ V$ because the resolution, as it was said was 10^{-15}. Therefore, there are no significant resolution losses when the variables are truncated before the model feedback.

As the scale of a variable represents the number of fractional bits, the scale can be changed simply by a left/right shifting of the variable. The left shifting increments the scale (adds one fractional bit) while the right shifting decrements it (subtract one fractional bit). The scale changer blocks are implemented using barrel shifters so the amount of bits shifted is adjustable at run time. The barrel shifter block adds low latency as it is based on multiplexers and static shifters. Therefore, the slowest elements of the model are the multipliers. However, those elements can be mapped into the DSPs of the FPGA, achieving global time steps of only tens of nanoseconds. In the proposed schematic of Fig. 2, the widths of the signals have been chosen in order to fit the operations into the DSPs [30], optimizing the performance of the model. The methodology proposed in this section offers real-time simulations with small simulation steps (around tens of nanoseconds) along with the flexibility of not having to redesign and resynthesize the model. As it can be seen in Fig. 2 the proposed implementation does not use any pipelining technique. With the proposed schematic, the results of each integration step are fedback for the next integration step, so using a pipeline strategy would not be an advantage. In order to sample the PWM inputs of the model, the FPGA registers them twice in order to

TABLE I: Buck parameters used in the Results section

Case	C	L	V_{in}	V_{out}	P	F_{sw}
1 [31]	$100\ \mu F$	$22\ \mu H$	$60\ V$	$5\ V$	$120\ W$	$200\ kHz$
2 [32]	$150\ \mu F$	$100\ \mu H$	$16\ V$	$12\ V$	$48\ W$	$200\ kHz$
3 [33]	$100\ \mu F$	$10\ \mu H$	$3.3\ V$	$2.7\ V$	$0.2\ W$	$600\ kHz$

synchronize them with the internal clock domain. This is a common practice in FPGA designs and it only adds a delay of two clock cycles — around 30 ns — taking into account the results of Section IV.

IV. Experimental results

This section presents a thorough comparison between the proposed model and two other models: standard fixed-point, and a reference model implemented with double-precision floating-point arithmetic. Despite of the logic resources needed to implement the models, the election of arithmetic determines whether the model can be adapted to simulations with different parameters (inductance, capacitance, load, etc.). Besides, that choice also determines the minimum simulation step that can be reached due to the implemented logic. In order to get a wide view of situations, three different configurations of the buck converter are considered in this section, as Table I shows. These configurations are recommended in the application notes of the following commercial buck controllers: LT3430-1 and LTC3892-1 [31], [32] from Linear Technologies, and MAX1685 from Maxim [33].

It is important to note that the proposed models have to be tested in open loop, without using a closed loop regulator. If closed loop were used, the regulator would compensate the numerical errors of the model, so the whole system would probably get the desired current and voltage values at steady state even if the model did not have the appropriate numerical resolution.

Every configuration has been implemented using parametrizable and standard fixed-point models and also the reference model. The reference model implements the same equations but using double-precision floating-point (variables of 64 bits) in order to avoid resolution issues. This reference model has been implemented using *real* floating-point data type. This data type cannot be synthesized but it is easy to use and provides a wide dynamic range along with high numerical accuracy. Likewise, this reference model implemented by hand has been previously validated comparing it with a Simulink model.

The traditional fixed-point should be implemented taking into account a set of variables that depends on the application (voltages, currents, capacitances, etc.), so the point location should be fixed before designing the model. Other sets can be used afterward as long as the new values fit in the original variables which cannot change their point locations without redesigning the model. For that reason, the buck converter implemented with standard fixed-point has been designed to optimize the simulation of Case 1 but allocating two more bits to the integer parts so bigger transients can be simulated. For the remaining cases, only the inputs of the models have been changed (inductance, capacitance, load, etc.). Case 1 is a halfway case, while case 2 and case 3 will present saturation and resolution errors respectively. Regarding the parametrizable fixed-point model, it has been reconfigured with the new scales and constants in order to optimize the model for every case. Table II shows the number of bits used (excluding the sign bit) in the state variables in every case, which are constant in the fixed-point model (as the change of the format would require redesign and resynthesize the model), but they vary in the parametrizable fixed-point model in order to optimize the accuracy of the emulation.

Fig. 3 shows the waveforms of the state variables for the fixed-point, parametrizable fixed-point models and the reference model (real-type). The simulation step for the simulations has been set to 20 ns. As it will be shown later, this simulation step can be reached with both models, so real-time simulations can be executed. All the simulations present a transient in the output voltage from 80% to 100%. The length of each simulation is equal to the settling time. These simulations show the mean values each switching cycle as the ripple makes it harder to follow the figures.

Both models obtain the same results in Case 1 (Fig. 3a), as the fixed-point model has been optimized to run this simulation. This figure has been included to show the transient.

Case 2 (Fig. 3b) presents a transient of a higher-current buck converter. As the inductor current exceeds 16 A —the highest value that can be stored in the state variable of the fixed-point model— that current saturates in the fixed-point model. Due to the saturation produced in the fixed-point model, the capacitor voltage behavior is affected, producing also a phase difference between the expected results and the calculated ones. This is the most obvious problem of traditional fixed-point notation. If a signal saturates, the simulation gets completely wrong results. On the other side, the parametrizable fixed-point model generates the expected results, as all the variables have been rescaled before the simulation.

Finally, case 3 (Fig. 3c) shows a low-power buck converter. This system presents high oscillations and needs much time to be settled. For that reason, Fig. 3d presents the detail of the last millisecond of the transient. It can be seen that the voltage of the fixed-point model is around 1% higher than the one of the parametrizable model, its oscillation is also 4% higher and there is a small phase difference between them. There are also mismatches in the current, but this effect is less important. These problems are present because this buck converter is intended for low power applications. Lower currents imply lower increments in the voltage state variable and, therefore, the resolution impact is higher. It is important to note that the resolution issues are mitigated during a transient because the increments of the state variables become higher, letting the increments to be considerably higher than the resolution of the variables. However, during steady state the increments can be considerably lower and these issues have more importance.

Table III shows numerical results of the previous experiments. The mean absolute error has been calculated comparing the results of both models with the high-resolution reference model. Case 1 presents an error below 0.7% for both state variables. If needed, this result can be improved by using more bits for the calculations. As it was explained before, the fixed-

TABLE II: Fixed-point format for the state variables

	Case 1		Case 2		Case 3	
	integer bits	fractional bits	integer bits	fractional bits	integer bits	fractional bits
Fixed-point	4	23	4	23	4	23
Par. Fixed-point	4	23	5	22	1	26

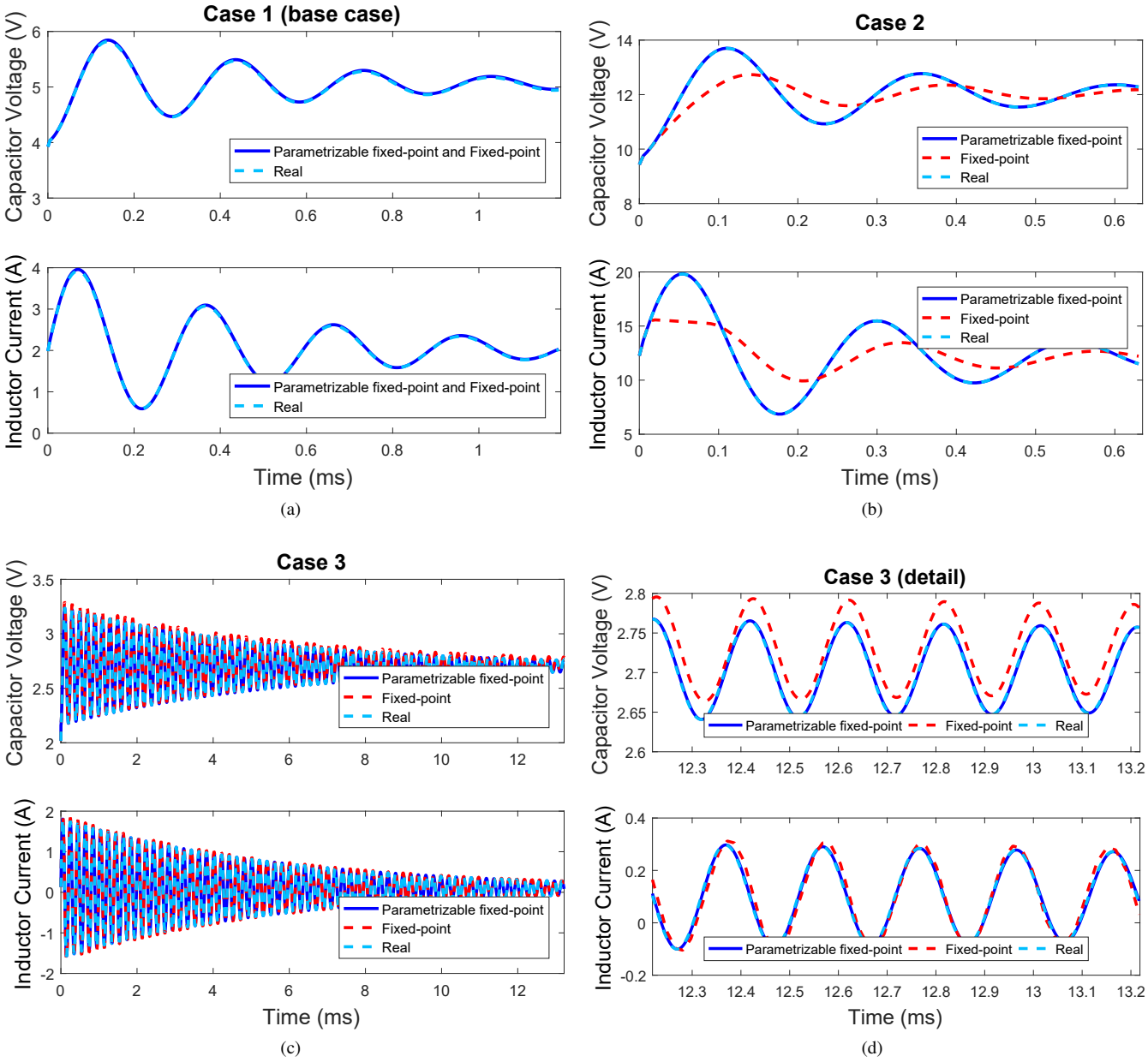

Fig. 3: Comparison between fixed-point and parametrizable fixed-point and *real*

TABLE III: Mean absolute error during a transient of 20% in the output voltage. Until settling time (2%)

	Case 1		Case 2		Case 3	
	i_L	v_c	i_L	v_c	i_L	v_c
Fixed-point	0.658%	0.304%	12.795%	3.509%	6.377%	0.956%
Par. Fixed-point	0.658%	0.304%	0.035%	0.015%	1.360%	0.069%

point model was designed to simulated Case 1, so it yields the same results as the parametrizable model. However, in Case 2, the fixed model presents an error over 12% in the inductor current because of its saturation, while parametrizable fixed-point presents errors around 0.03%. Finally, Case 3 presents resolution issues for both models, but in the case of the fixed-point model, the error is much bigger. The problem is that the increments for the current in this simulation are usually around

TABLE IV: Mean absolute error during steady state

	Case 1		Case 2		Case 3	
	i_L	v_c	i_L	v_c	i_L	v_c
Fixed-point	0.636%	0.410%	0.281%	0.223%	24.422%	1.034%
Par. Fixed-point	0.636%	0.410%	0.023%	0.014%	1.745%	0.021%

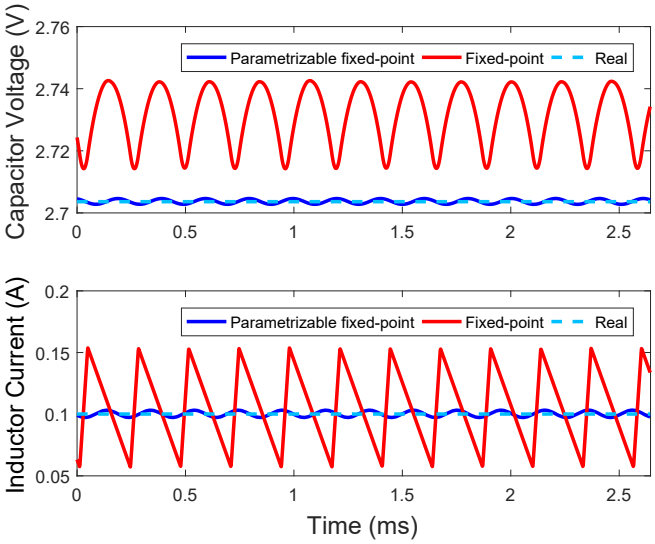

Fig. 4: Waveforms of case 3 starting already in steady state.

10^{-5} A while the actual current is around 0.1 A (around 10^{-5} V and 2.7 V in the case of the voltage, respectively), so many bits should be used to store the state variable and also to feedback the results to the other state variables. This resolution problem affects noticeably the fixed-point model, as its resolution is lower because it was designed for Case 1.

Another set of experiments has been accomplished to analyze the behavior of the models during steady state. Using the same cases of Table I, the simulations have been started already in steady state. These simulations let us analyze whether the models reproduce accurately the natural frequency of the systems without any perturbation. Table IV shows the mean absolute error compared with the reference model, as well as the previous table. Case 1 and case 2 for parametrizable fixed-point present similar errors than in the transient as there were no significant resolution issues. For case 2 with fixed-point, the error is much smaller because in steady state saturation is not reached.

However, Case 3 presents a different behavior because the resolution problem is aggravated during steady states, as the increments for the state variables are even smaller. In this case, the error of the inductor current of the fixed-point model grows up to 24%, while the parametrizable model is kept under 1.8%. Fig. 4 shows the results for Case 3 during steady state. The reference waveforms have also been included in this case. It can be seen that the reference model presents almost no oscillations in both state variables. Parametrizable model produces some oscillations but with low amplitude, while fixed-point model presents a noticeably oscillation. The switching ripple has been filtered, so the inductor current oscillation that can be seen is produced by the resolution problems of the model. Basically, for the standard fixed-point model, the increments of the current variable are numerically around the resolution of the model. Because of this, the increments cannot be represented with enough resolution and a sawtooth wave is obtained. The mean value of this wave is near the ideal value, but its oscillation is noticeably wrong. This oscillation affects the calculations of the capacitor voltage, even when the voltage does not present these resolution issues.

Taking into account both sets of experiments, some conclusions can be extracted. The standard fixed-point model is easy to implement but its utility is constrained to simulations close to its base case. Even when the converter characteristics are near the base case, some transients can saturate momentarily the variables of the model, preventing from simulating accurately those transients. On the other side, simulations with values much lower than the base case also present problems when using the fixed-point model, because of resolution problems. The resolution problems in this case can be always present, but they are especially noticeable during simulations in steady state. All these problems can be avoided redesigning and resynthesizing the model but it is not affordable in real applications because the user may not have the knowledge to do that task and the effort would not be negligible. However, parametrizable fixed-point is only designed once, and it is reconfigured with the parameters of the present simulation. This configuration consists in calculating the scales of every variable, but this process can be automated with a simple software.

Once the accuracy results have been analyzed, implementation results in the FPGA will be shown. Fixed-point and parametrizable fixed-point models have been implemented in a Xilinx FPGA Zynq 7 (XC7Z020-1CLG400C) in order to get the utilization of the device and the minimum simulation step. All models have been implemented using Vivado 2017.3 and using the standard Xilinx synthesizer: XST. The architecture of the selected FPGA has two main parts: PLS (Programmable Logic Structure) and APU (Application Processing Unit). The former is where the whole model is implemented, while the latter is only used to implement the communications between the computer and the hardware-based model. These communications only consist on the configuration of the scales of the variable of the model and the converter parameters.

It can be seen in Table V that both models can reach simulations steps under 16 ns (around 62 MHz). It is worth noting that in these simple models, which are a direct implementation of equations (2 - 4), the simulation step is equal to the FPGA clock period. Parametrizable fixed-point requires more FPGA resources and its minimum simulation step is slightly greater than in the case of standard fixed-point model. That was expected as the parametrizable model includes the barrel-shifter modules which change the scale of the different

TABLE V: FPGA (Xilinx XC7Z020-1CLG400C) resources used by the design

System	Min simulation step	4 input LUTs	FFs	DSP
Parametrizable fixed-point	15.13 ns	290	56	2
Parametrizable fixed-point (no DSPs)	18.72 ns	699	56	0
Fixed-point	11.79 ns	110	56	2
Fixed-point (no DSPs)	16.99 ns	664	56	0
32-bit Floating-point	45.44 ns	1516	64	4
32 bit Floating-point (no DSPs)	49.62 ns	3365	64	0
64-bit Floating-point	64.35 ns	3540	128	18
64 bit Floating-point (no DSPs)	66.92 ns	10626	128	0

variables. Comparing the number of LUTs (Look Up Table) it may seem that the barrel-shifters imply a large overhead, but most of the logic is in fact dedicated to the multipliers, implemented in the DSP blocks. Just to make an idea of the barrel-shifters global overhead, both models were also synthesized without using DSP blocks showing similar results in terms of LUTs and simulation step. Of course, this is not recommended because the simulation step greatly increases.

Table V also shows a comparison with 32-bit and 64-bit floating-point implementations. The table shows that the 32-bit floating-point model is about three times slower and it also requires around five times more resources. Likewise, the 64-bit floating-point model is about four times slower and it also needs around fifteen times more resources. A first drawback is the simulation step because, as it was explained in Section II-A, small simulation steps are needed to emulate high switching-frequency converters with accuracy. But the increase in necessary resources is also a important point to be taken into account because it has a direct impact in the final price of the HIL system.

Acknowledgment

This work has been supported by the Spanish Ministerio de Economía y Competitividad under project TEC2013-43017-R

V. Conclusion

This paper has proposed a variation of fixed-point arithmetic called parametrizable fixed-point intended to be implemented in FPGAs. This arithmetic takes the advantages of both fixed and floating-point, as it gets the best performance of the former while keeping the flexibility of the latter. The speed requirement has been reached using simple signed integer mathematical operations as they are quite faster than the floating-point alternative. Besides, point location should be changed without the need of resynthesizing so several barrel shifters have been included to get that purpose. The proposed arithmetic has been tested along with a model of a buck converter and it has been shown that it offers real-time simulations with a simulation step under 20 ns with negligible numeric errors. This noticeably small simulation step enables modeling high switching-frequency converters, as the simulation step limits the applications that can be modeled accurately. Besides, the experiments show that this arithmetic is adapted to the simulation characteristics, avoiding saturation or low-resolution issues.

References

[1] B. Patella, A. Prodic, A. Zirger, and D. Maksimovic, "High-frequency digital PWM controller IC for DC-DC converters," *Power Electronics, IEEE Transactions on*, vol. 18, no. 1, pp. 438–446, Jan 2003.

[2] A. Peterchev, J. Xiao, and S. Sanders, "Architecture and IC implementation of a digital VRM controller," *Power Electronics, IEEE Transactions on*, vol. 18, no. 1, pp. 356–364, Jan. 2003.

[3] D. Maksimovic, R. Zane, and R. Erickson, "Impact of digital control in power electronics," in *Power Semiconductor Devices and ICs, 2004. Proceedings. ISPSD '04. The 16th International Symposium on*, May 2004, pp. 13–22.

[4] S. Buso, L. Malesani, and P. Mattavelli, "Comparison of current control techniques for active filter applications," *IEEE Transactions on Industrial Electronics*, vol. 45, no. 5, pp. 722–729, Oct 1998.

[5] D. M. V. de Sype, K. D. Gusseme, A. P. M. V. den Bossche, and J. A. Melkebeek, "Duty-ratio feedforward for digitally controlled boost PFC converters," *IEEE Transactions on Industrial Electronics*, vol. 52, no. 1, pp. 108–115, Feb 2005.

[6] M. Salehifar, M. Moreno-Eguilaz, G. Putrus, and P. Barras, "Simplified fault tolerant finite control set model predictive control of a five-phase inverter supplying BLDC motor in electric vehicle drive," *Electric Power Systems Research*, vol. 132, pp. 56 – 66, 2016.

[7] H. Berriri, W. Naouar, I. Bahri, I. Slama-Belkhodja, and E. Monmasson, "Field programmable gate array-based fault-tolerant hysteresis current control for ac machine drives," *IET Electric Power Applications*, vol. 6, no. 3, pp. 181–189, March 2012.

[8] Z. Tir, O. P. Malik, and A. M. Eltamaly, "Fuzzy logic based speed control of indirect field oriented controlled double star induction motors connected in parallel to a single six-phase inverter supply," *Electric Power Systems Research*, vol. 134, pp. 126 – 133, 2016.

[9] A. S. Vijay, S. Doolla, and M. C. Chandorkar, "Real-time testing approaches for microgrids," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 5, no. 3, pp. 1356–1376, Sept 2017.

[10] B. Lu, X. Wu, H. Figueroa, and A. Monti, "A low-cost real-time hardware-in-the-loop testing approach of power electronics controls," *Industrial Electronics, IEEE Transactions on*, vol. 54, no. 2, pp. 919–931, April 2007.

[11] M. Matar and R. Iravani, "FPGA implementation of the power electronic converter model for real-time simulation of electromagnetic transients," *Power Delivery, IEEE Transactions on*, vol. 25, no. 2, pp. 852–860, April 2010.

[12] G. Parma and V. Dinavahi, "Real-time digital hardware simulation of power electronics and drives," *Power Delivery, IEEE Transactions on*, vol. 22, no. 2, pp. 1235–1246, April 2007.

[13] A. Myaing and V. Dinavahi, "FPGA-based real-time emulation of power electronic systems with detailed representation of device characteristics," *Industrial Electronics, IEEE Transactions on*, vol. 58, no. 1, pp. 358–368, Jan. 2011.

[14] S. Karimi, P. Poure, and S. Saadate, "An HIL-Based reconfigurable platform for design, implementation, and verification of electrical system digital controllers," *Industrial Electronics, IEEE Transactions on*, vol. 57, no. 4, pp. 1226–1236, April 2010.

[15] T. Liang and V. Dinavahi, "Real-time system-on-chip emulation of electrothermal models for power electronic devices via hammerstein configuration," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 6, no. 1, pp. 203–218, March 2018.

[16] Y. Chen and V. Dinavahi, "Digital hardware emulation of universal machine and universal line models for real-time electromagnetic transient simulation," *Industrial Electronics, IEEE Transactions on*, vol. 59, no. 2, pp. 1300–1309, Feb. 2012.

[17] A. Fernández-Álvarez, M. Portela-García, M. García-Valderas, J. López, and M. Sanz, "Hw/sw co-simulation system for enhancing hardware-in-the-loop of power converter digital controllers," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 5, no. 4, pp. 1779–1786, Dec 2017.

[18] X. Yang, C. Yang, T. Peng, Z. Chen, B. Liu, and W. Gui, "Hardware-in-the-loop fault injection for traction control system," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 6, no. 2, pp. 696–706, June 2018.

[19] L. Ibarra, A. Rosales, P. Ponce, A. Molina, and R. Ayyanar, "Overview of real-time simulation as a supporting effort to smart-grid attainment," *Energies*, vol. 10, no. 6, 2017. [Online]. Available: http://www.mdpi.com/1996-1073/10/6/817

[20] H. Jafarian, N. Kim, and B. Parkhideh, "Decentralized control strategy for ac-stacked pv inverter architecture under grid background harmonics," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 6, no. 1, pp. 84–93, March 2018.

[21] S. Srdic, X. Liang, C. Zhang, W. Yu, and S. Lukic, "A sic-based high-performance medium-voltage fast charger for plug-in electric vehicles," in *2016 IEEE Energy Conversion Congress and Exposition (ECCE)*, Sept 2016, pp. 1–6.

[22] O. Lucia, I. Urriza, L. Barragan, D. Navarro, O. Jimenez, and J. Burdio, "Real-time FPGA-based hardware-in-the-loop simulation test bench applied to multiple-output power converters," *Industry Applications, IEEE Transactions on*, vol. 47, no. 2, pp. 853–860, March-April 2011.

[23] A. Sanchez, A. de Castro, and J. Garrido, "A comparison of simulation and hardware-in-the-loop alternatives for digital control of power converters," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 3, pp. 491–500, Aug 2012.

[24] IEEE, "1076-2008 - IEEE standard VHDL language reference manual."

[25] C. Dufour, S. Cense, and J. Bélanger, "Fpga-based switched reluctance motor drive and dc-dc converter models for high-bandwidth hil real-time simulator," in *2013 15th European Conference on Power Electronics and Applications (EPE)*, Sept 2013, pp. 1–8.

[26] E. Adzic, S. Grabic, M. Vekic, V. Porobic, and N. F. Celanovic, "Hardware-in-the-loop optimization of the 3-phase grid connected converter controller," in *IECON 2013 - 39th Annual Conference of the IEEE Industrial Electronics Society*, Nov 2013, pp. 5392–5397.

[27] R. Kang, S. Kim, I. Yang, K. Jeong, C. Kang, and G. Kim, "The use of fpga in hil simulation of three phase interleaved dc-dc converter," in *2012 IEEE Vehicle Power and Propulsion Conference*, Oct 2012, pp. 772–776.

[28] J. G. Kassakian and T. M. Jahns, "Evolving and emerging applications of power electronics in systems," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 1, no. 2, pp. 47–58, June 2013.

[29] M. Rodríguez, Y. Zhang, and D. Maksimović, "High-frequency pwm buck converters using gan-on-sic hemts," *IEEE Transactions on Power Electronics*, vol. 29, no. 5, pp. 2462–2473, May 2014.

[30] *UG479. User Guide of DSP48E1 Slice*, Xilinx. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf

[31] *High Voltage, 3A, 200kHz/100kHz Step-Down Switching Regulators LT3430-1*, Linear Technology. [Online]. Available: http://www.analog.com/media/en/technical-documentation/data-sheets/34301fa.pdf

[32] *60V Low IQ, Dual, 2-Phase Synchronous Step-Down DC/DC Controller LTC3892-1*, Linear Technology. [Online]. Available: http://www.analog.com/media/en/technical-documentation/data-sheets/38921fc.pdf

[33] *Low-Noise, 14V Input, 1A, PWM Step-Down Converters MAX1685*, Maxim. [Online]. Available: https://datasheets.maximintegrated.com/en/ds/MAX1684-MAX1685.pdf

Alberto Sanchez was born in Madrid, Spain, in 1986. He received the M.Sc. and Ph.D. degrees in computer science and telecommunication engineering from the Universidad Autónoma de Madrid, Madrid, in 2010 and 2013, respectively. He is currently an Assistant Professor in the Technology for Electronics and Communications Department, Universidad Autónoma de Madrid. His research interests include simulation of mixed digital and analog systems and digital control of switching mode power supplies.

Angel de Castro (M'08) was born in Madrid, Spain, in 1975. He received the M.Sc. and the Ph.D. degrees in electrical engineering from the Universidad Politécnica de Madrid, Madrid, Spain, in 1999 and 2004, respectively. He has been an Associate Professor in the Universidad Autónoma de Madrid since 2010, and as Assistant Professor from 2006 to 2010. Previously, he was an Assistant Professor in the Universidad Politécnica de Madrid since 2003. His research interests include digital control of switching mode power supplies, field programmable gate arrays and mobile nodes in wireless sensor networks.

Javier Garrido (M'97) was born in Madrid, Spain, in 1954. He received the B.Sc. degree in 1974, the M.Sc. degree in 1976 and the Ph.D. degree in 1984 in Physics from the Universidad Autónoma de Madrid, UAM (Spain). Since 1992 he has been participated, in the implementation of the Computer Science (1992) and Telecommunication (2002) engineering studies at the Polytechnic School (EPS-UAM), and he got his current position as Full Professor at 2010. From his incorporation to the EPS, he has extended his research interests to topics related with HW/SW applications on embedded systems (microcontrollers, microprocessors, FPGAs and SOC devices) as platforms for wireless sensor nets (WSN) or robotic sensor agents (RSA). In 2003 he co-founded the HCTLab group, (Hardware and Control Technology Laboratory) and now he is its director. Dr. Garrido has been participated in several R&D projects and has published more than 40 articles in peer-review journals and 60 papers in archived conference proceedings.