electronics

MDPI

Article

Automatic Word Length Selection with Boundary Conditions for HIL of Power Converters

Mariano Alberto García-Vellisca *, Carlos Quiterio Gómez Muñoz, María Sofía Martínez-García and Angel de Castro

HCTLab Research Group, Universidad Autónoma de Madrid, 28049 Madrid, Spain; carlosq.gomez@uam.es (C.Q.G.M.); sofia.martinez@uam.es (M.S.M.-G.); angel.decastro@uam.es (A.d.C.)
* Correspondence: marianoa.garcia@uam.es; Tel.: +34-914-973-556

Abstract: Hardware-in-the-loop (HIL) is a common technique used for testing in power electronics. It draws upon FPGAs (field-programmable gate arrays) because they allow for reaching real-time simulation for mid-high switching frequencies. FPGA area and delay are keys to reaching a compromise between performance and accuracy. To minimize area and delay, signal word length (WL) is critical. Furthermore, the input and output's WL should be carefully chosen because these signals come from ADCs (analog-to-digital converters) or go to DACs (digital-to-analog converters). In other words, the role of ADCs and DACs is the boundary condition when assigning all the signal WLs in an HIL model. This research presents an automatic method for computing the signal WLs in the corresponding model by considering input/output boundary conditions. This automatic method needs a single simulation to decide both the integer and fractional width of every signal. Our method accelerates the process, showing an advantage over manual methods and those requiring multiple simulations. The proposed method is applied to create all the WL assignments to the signals involved in a fixed-point coded buck converter model, which shows its feasibility.

Keywords: FPGA; HIL; word length (WL); ADC; DAC; boundary conditions; fixed-point; floating-point; buck converter

Citation: García-Vellisca, M.A.; Gómez Muñoz, C.Q.; Martínez-García, M.S.; de Castro, A. Automatic Word Length Selection with Boundary Conditions for HIL of Power Converters. *Electronics* **2023**, *12*, 3488. https://doi.org/10.3390/electronics12163488

Academic Editor: Raffaele Giordano

Received: 8 July 2023
Revised: 2 August 2023
Accepted: 14 August 2023
Published: 17 August 2023

1. Introduction

Electric and electronic devices evolve at breakneck speed, making the hardware-in-the-loop (HIL) simulation technique a unique partner for keeping up with this fast development. This technique allows for the emulation of system behavior in real time with accurate results [1]. One of the most recently published research studies on wind turbine certification shows small enough deviations between field-based and HIL-based simulation results [2].

HIL is commonly used as a testing process in different areas of power electronics [3–9]. Within this field, there are different studies on automatic control for electric drives [10,11] or power converters [12–14] implemented with FPGAs. In fact, there are many research studies on power converters using FPGAs because of their performance for mid-high switching frequencies [15,16]. To drive signals in and out from an HIL simulation, DACs (digital-to-analog converters) and ADCs (analog-to-digital converters) must be used. The number of bits used in these DACs and ADCs is limited and cannot be freely chosen. Therefore, the hardware (HW) imposes strong restrictions that play the role of boundary conditions (BC).

When designing any system in an FPGA, the signals that represent numerical quantities will be represented as floating-point or fixed-point, where there are different bits devoted to the integer and fractional parts. For modeling, algorithm designers prefer floating-point due to its flexibility and ease of use. However, floating-point implies more hardware resources, which increase the cost. Therefore, fixed-point arithmetic is the final

choice for synthesizable models where area or time are critical, even though it also requires more effort to find a good enough number of bits for properly representing every signal; that is, to find the right word length (WL) [17–21].

Industry is interested in making smaller and faster hardware. Nonetheless, in order to program an FPGA, a synthesizable code is needed, in which FPGA area and delay are the keys to efficiency without jeopardizing accuracy [22–25]. The number of lookup tables (LUTs) and flip-flops (FFs) are typical parameters for assessing the size of the logic circuit. A total delay comes from the logic and routing within the FPGA: the shorter the WL, the fewer the number of LUTs, despite reducing accuracy. Nevertheless, the area and delay shrink with lower LUTs. This leads to finding a trade-off between the FPGA area and delay and the accuracy. One of the strategies applied for optimizing the WL is signal grouping based on different criteria [26–28].

This paper proposes a method for choosing the number of bits of both the integer and fractional part of all signals in an HIL model of a fixed-point switched converter. The paper presents two main contributions in the field of WL optimization for HIL applied to power electronics. The first contribution is an automatic method that achieves optimal results through a single simulation of the system, obtaining the range of different signal values in both transient and steady-state regimes. From this simulation, the necessary values are automatically derived to adjust the WL to an optimal level, avoiding wasted space and ensuring appropriate resolution. This automated approach eliminates the need for calculations or formula applications by hand to determine bandwidths, overcoming limitations in previous research approaches [26,29].

The second contribution focuses on boundary conditions. The proposed system allows for the customization of optimization across numerous cases by imposing specific boundary conditions. These boundary conditions are determined by the bit width of the ADCs and DACs, i.e., the inputs and outputs of the system. It is crucial to adapt the resolution of internal signals based on these boundary conditions to prevent bit wastage and ensure efficient resource utilization. The paper demonstrates how to identify the signals that determine the resolution of other signals, thereby enabling the optimal adjustment of bit widths in all internal signals of the system.

In summary, this paper addresses the calculation of the optimal number of bits in internal signals. The proposed approach is the only one that takes into account the resolution of ADCs and DACs, which are the inputs and outputs of the HIL system, ensuring efficient resource utilization and adaptation to specific conditions. This automatic and adaptable methodology represents a substantial advancement in the field of bit width optimization for HIL systems.

The rest of the paper is organized as follows: Section 2 presents fixed-point basic concepts and explains the application example and simulation details. Section 3 shows the results. Finally, Section 4 provides the discussion and conclusions.

2. Word Length Selection with Boundary Conditions

2.1. Case Study

In this paper, a buck converter is selected to show in detail how to make a WL selection with BC. The buck converter topology is shown in Figure 1. For simplicity, a simple buck converter without losses is presented, although the same conclusions for other topologies can also be achieved. For this model, the output voltage (v_{out}) and the inductor current (i_L) are the state variables. The selected parameters used for the proposed method are shown in Table 1.

Table 1. Buck converter parameters.

Parameter	Buck Converter
f_{sw}	200 kHz
L	22 µH
C	220 µF
P	10 W
v_{out}	5 V
v_g	12 V
Δt	20 ns

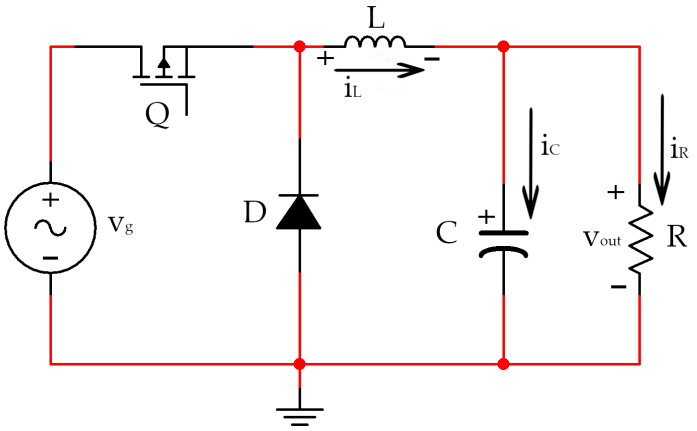

Figure 1. Buck converter schematic.

The inductor voltage is defined by Equation (1):

$$v_L = L\frac{di_L}{dt}. \tag{1}$$

Using the explicit Euler method, from Equation (1), the inductor current for each time step k is as follows:

$$i_L(k) = i_L(k-1) + \frac{\Delta t}{L} v_L(k-1), \tag{2}$$

where:

$$v_L(k-1) = \begin{cases} v_g(k-1) - v_{out}(k-1), & \text{when } Q = 1 \\ -v_{out}(k-1), & \text{when } Q = 0 \text{ and } i_L > 0 \\ 0, & \text{when } Q = 0 \text{ and } i_L = 0 \end{cases}. \tag{3}$$

Analogously, the capacitor current is defined by Equation (4):

$$i_C(k-1) = i_L(k-1) - i_R(k-1). \tag{4}$$

Again, using the explicit Euler method, the output voltage for each time step k is:

$$v_{out}(k) = v_{out}(k-1) + \frac{\Delta t}{C} i_C(k-1), \tag{5}$$

where Δt is the simulation time step, L is the inductance, and C is the capacitance, which are all constants, while the MOSFET state (Q), the inductor voltage value (v_L), the load current i_R, and the capacitor current i_C value are variables. The MOSFET may be closed ($Q = 1$) or open ($Q = 0$). In the former case, v_L is equal to $v_g - v_{out}$, when in the second case v_L is equal to $-v_{out}$ when $i_L > 0$ (continuous conduction mode or CCM), but is zero when $i_L = 0$ (discontinuous conduction mode or DCM). On the other hand, i_c is always equal to $i_L - i_R$.

2.2. Initial Assignment of Bits Based on Simulation Data

To represent a fixed-point signal, several bits (X) for the integer part and some other bits (Y) for the fractional part are needed. The total number of bits is $X + Y + 1$ to include the sign, which is coded in two's complement. The number of bits is carefully chosen for each part and presented in the format $X.Y$ for simplicity. A 64-bit floating-point simulation from the case study is used as a golden model to obtain actual values from every signal. This simulation must include a transient regime in which the extreme values of every signal are reached, but also a steady-state regime which usually includes the smallest increments. The simulation can be run by any tool, as long as the set of values of every signal along the simulation can be extracted. At this point, this work follows [26] to find the minimum number of bits for the integer and fractional part of each signal as described below. For that purpose, model signals are separated into three main groups: constants, accumulative, and non-accumulative (see Figure 2). Constants have a fixed value. State variables depend on their previous values, so they fall into the accumulative category. However, non-accumulative signals, i.e., inputs and outputs, feedback signals, and multiplexer outputs, may or may not depend on another system variable.

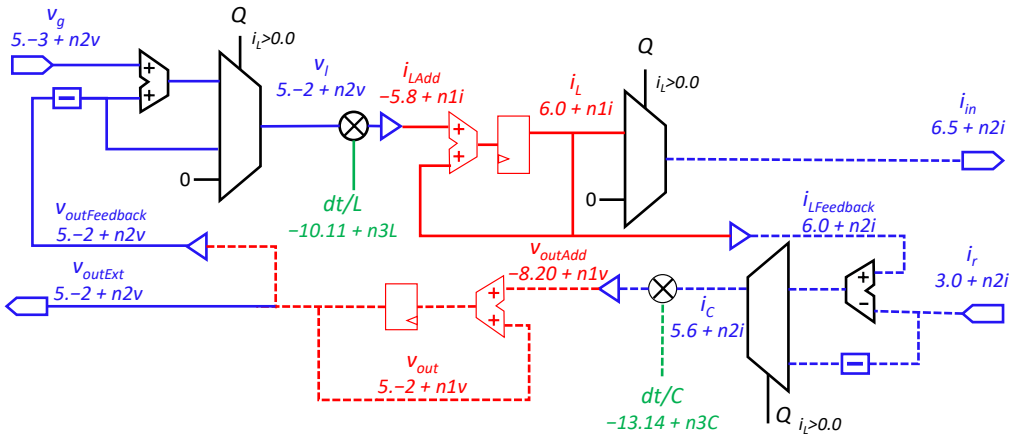

Figure 2. Schematic of the case study. Non-accumulative signals (blue), accumulative signals (red), and constants (green).

Following Equation (6), the minimum number of bits for the integer part X of every signal (b) is obtained. The maximum absolute value of the signal is used in this case, which usually takes place during the transient behaviour. An extra bit is added for avoiding possible overflows.

$$X = \lceil log_2(max(|b|)) \rceil + 1 \tag{6}$$

As a general rule for the fractional side, the minimum value in steady-state for every signal is taken. Thus, the number of bits following Equation (6) is:

$$Y = -\lfloor log_2(min(|b|)) \rfloor \tag{7}$$

However, if the signal in a steady-state regime crosses zero, a colossal amount of fractional bits is required to represent numbers close to zero. In this scenario, the criteria in [29] are followed, where the authors suggest not computing the smaller values to a highly accurate level if those points do not have enough impact over the whole signal. Thus, the algorithm first detects the maximum and minimum in a steady-state regime. The next step is discarding 5% of the time with smaller values, considering the rest for an accurate level of representation. The mathematical expression is shown in (8):

$$Y = -\lfloor log_2(|\frac{2.5}{100}(Max_{ss} - min_{ss})|) \rfloor, \tag{8}$$

where Max_{ss} is the maximum of the signal, and min_{ss} is the minimum of the signal, both over a steady-state regime.

Equations (9) and (10), as in (6) and (7), return the minimum number of bits to begin with for representing a constant value:

$$X = \lceil log_2(const) \rceil, \tag{9}$$

$$Y = -\lfloor log_2(|const|) \rfloor, \tag{10}$$

In fact, $X + Y$, as stated in (9) and (10), will be always 1, so the total number of bits would be $X + Y + 1 = 2$ bits, including the sign. Such a minimum number of bits represents a signal with low resolution, so to obtain a reasonable resolution, more fractional bits have to be added. The following sub-section presents detailed rules to figure out the right amount of additional bits to be added to the fractional part.

2.3. Final Assignment of Bits Based on Boundary Conditions

This section shows why boundary conditions are key to finding out the final number of bits for every signal involved. The extra bits to be added to the fractional part of every signal are represented as $n1i, n1v, n2i, n2v, n3L$, and $n3C$ for the accumulative (i_L and v_{out}), non-accumulative ($i_{in}, i_r, i_{LFeedback}, i_C, v_g, v_{outFeedback}, v_l$ and v_{outExt}), and constant (dt/L and dt/C) groups, respectively. For the sake of clarity, the proposed automatic method is divided into three steps:

1. **Accumulative signals' word length.**
   There are two different sub-groups within those three main groups, as can be seen in Figure 2. The solid red line represents the current, and the dashed red line represents the voltage. Every signal is linked within the same sub-group through arithmetical operations. When adding, if one signal has fewer bits than another, the resolution of the result is defined by the one with fewer fractional bits. Therefore, the signal with fewer fractional bits should be raised to have as many fractional bits as the others in the same sub-group in order to maintain the required accuracy.
   For instance, taking the solid red line sub-group, the fractional sides of all accumulative signals are adjusted, assigning them the highest value. The fractional part of i_L is $Y = 0$, and the fractional part of i_{Ladd} is $Y = 8$ at the initial assignment (see Figure 2). Taking the highest value within this upper sub-group, the fractional part of i_L is eight as well. This is analogous to the sub-group at the bottom. Initially, v_{outadd} is $Y = 20$, and v_{out} is $Y = -2$. Their fractional part becomes $Y = 20$, which is the highest between them.
   Once the fractional bits in each sub-group are adjusted, it is time for WL equalization. In [26], the authors state that if the number of bits in a group (such as accumulative signals) are different, the one with fewer bits acts as a bottleneck. Thus, the difference between both WLs will be the number of extra bits to be added to the sub-group with fewer bits. The extra bits $n1i$ and $n1v$ (for the above sub-group and below sub-group, respectively) are added to the fractional part for improving resolution. For the accumulative group, i_L and v_{out} are the signals inside the accumulative group, so they need to be compared with each other. Their WLs are 15 ($Q6.8$) and 26 ($Q5.20$), respectively. The difference between them is 11 bits, which represents the number of bits to add to the smallest accumulative signal sub-group. Thus, the extra bits for adding to each group are $n1i = 11$ and $n1v = 0$, as shown in Figure 3.
2. **Input–output boundary conditions.**
   This second step presents the adjustment of non-accumulative signals (blue signals). Input–output signals fall in this group and have a number of bits imposed by the bits of the DACs and ADCs used. The same adjustment as in the previous step is carried out, but for equalizing both sub-groups: the current (solid blue line) and voltage sub-group (dashed blue line). The input signals from ADCs must have the

same number of bits as the ADCs used. They become the BC because having fewer bits degrades the model accuracy and filling with zeros to the right does not include any advantage. Taking the upper sub-group, we adjust the fractional sides of all non-accumulative signals, assigning them the highest value.

Taking the current sub-group, the fractional sides of all non-accumulative signals are adjusted, assigning them the highest value. For instance, the fractional parts of v_g and v_l are $Y = -3$ and $Y = -2$, respectively, at the initial assignment (see Figure 3). Taking the highest value within this upper sub-group, the fractional part of each one will be $Y = -2$. This is analogous for the voltage sub-group. The initial fractional parts for v_{outExt} and $v_{outFeedback}$ are incidentally the same, $Y = -2$. In this example, there is no need to find the highest fractional part for imposing the most restrictive amount to the fractional part of v_l.

Next is for WL equalization, as shown before. Following input–outputs, the fractional parts of v_g, v_{outExt}, i_{in}, and i_r need to be compared among each other. Their WLs are 3 $(5.-2)$, 3 $(5.-2)$, 12 (6.6), and 9 (3.6), respectively. In this work, 12 bits are imposed as a BC, since this is a common value of ADCs. Therefore, the extra bits ($n2i$ and $n2v$) that end up with the same WL for every signal within the same main group are $n2v = 9$ and $n2i = 3$, respectively, to fit the BC. Additionally, $v_{outFeedback}$ and $i_{LFeedback}$ receive the same value as v_g and i_r, respectively.

3. **Adjust constants' word length.**

That is the easiest step because constants ($\Delta t/L$ and $\Delta t/C$) need only one bit to start. The WL without the sign bit is $(X + Y = 1)$, so there is no need to find the highest value between constants. For WL equalization, as pointed out in [26], constants use the largest n value of the rest of the sub-groups. The highest value among the four sub-groups mentioned before is 11. Thus, the extra bits needed to result in the same WL for each constant is $n3L = n3C = 11$. The final values are summarized in Figure 3.

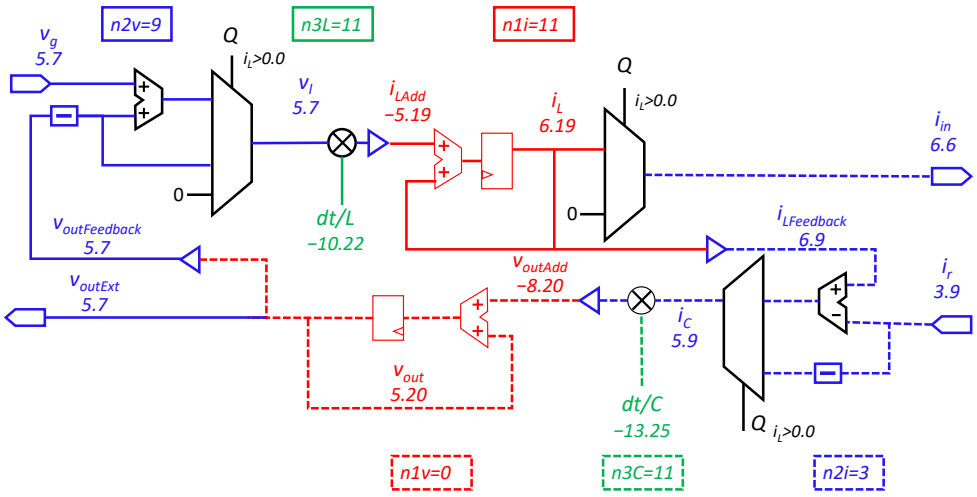

Figure 3. Schematic of the case study. Non-accumulative signals (blue), accumulative signals (red), and constants (green). Values after applying BC.

This method is able to be automated as an algorithm. A novelty of this work is its automation by coding it in a MATLAB script. The algorithm is based on the BC and the parameters addressed in Table 1. It creates a single simulation to find the minimum values for the integer (X) and fractional (Y) parts. At that point, the algorithm executes the previous steps to find the number of bits for both parts of all model signals, obtaining the WL for every model signal. A flow chart is depicted in Figure 4 to clarify what the algorithm does.

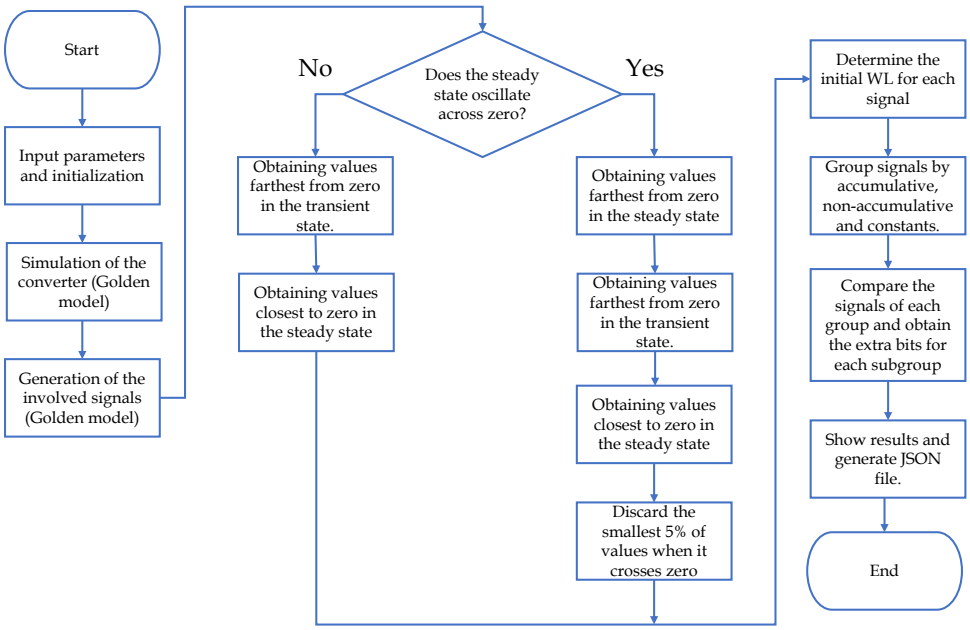

Figure 4. Flow chart of MATLAB script.

3. Results

The proposed method supplies a WL for every signal with only one simulation and boundary conditions. However, to verify the method works, this study uses relative error (RE) to analyze whether the number of bits chosen is other than the ones given back by the proposed method. Therefore, all simulations shown in this section are used for method validation. However, they would not be necessary when simply applying the method.

First, a golden model as a reference is established by simulating the case study using 64-bit floating-point number representation. Figure 5 shows how state variables v_{out} and i_L from that model behave until both reach a steady-state regime. Parameter values are shown in Table 1.

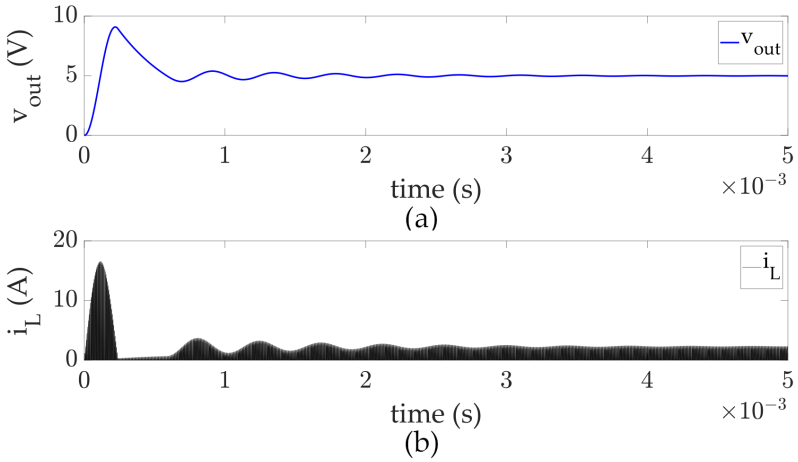

Figure 5. (**a**) v_{out} signal and (**b**) i_L signal, both based on a 64-bit floating-point simulation.

Second, a fixed-point buck converter model is simulated to assess the number of bits given by the proposed method for every signal. The RE of the fixed-point model with respect to the golden model is computed by the mean absolute error (MAE), which is expressed in the numerator of Equation (11), divided by the typical value of each signal. Of course, the typical value is different for each signal. These values are 5 V and 2 A for v_{out} and i_L, respectively, which correspond to the nominal values, as can be deduced in Table 1.

The RE parameter provides information about how efficient the proposed method is in comparison with a supposedly almost no error system, the golden model. The same time step (20 ns) is defined for both simulations, so there are no differences due to sampling. A comparison of every sample between the golden model (Gm_i) and the respective sample (Fpm_i) from the fixed-point model is completed. The number of samples, $Nsamp$, is also the same for both simulations, and $Tvalue$ is the typical value for each signal, as stated before. The mathematical expression for the RE is shown in Equation (11):

$$\mathrm{RE} = \frac{MAE}{Tvalue} = \frac{\sum_{i=1}^{Nsamp} |Fpm_i - Gm_i| / Nsamp}{Tvalue}. \tag{11}$$

The resolution problem in fixed-point implemented systems usually comes from a bottleneck, i.e., a signal that produces most of the system error. Thus, several simulations are carried out by increasing every signal WL (or signal group in this study) to identify such a bottleneck. Then, a sweep over different values is accomplished for the state variables WL (continuously and independently for both state variables), constants, and non-accumulative variables, restricted in principle by the BC. The procedure is based on increasing the WL of every group individually and analyzing whether the RE decreases.

Figure 6 shows how the RE evolves versus the number of bits (in increments of two) for the fractional part of one state variable, in this case v_{out}. The number of bits for the state variable i_L remains with the value ($X.Y = 6.19$) given by the method. Both curves have similar tendencies and point to $Y = 20$ as the maximum number of bits for the fractional part without wasting bits. From that value onward, the RE does not diminish, so additional bits would be wasted. Thus, although similar results can be achieved by using i_L, v_{out} is used for computing RE for the following comparisons.

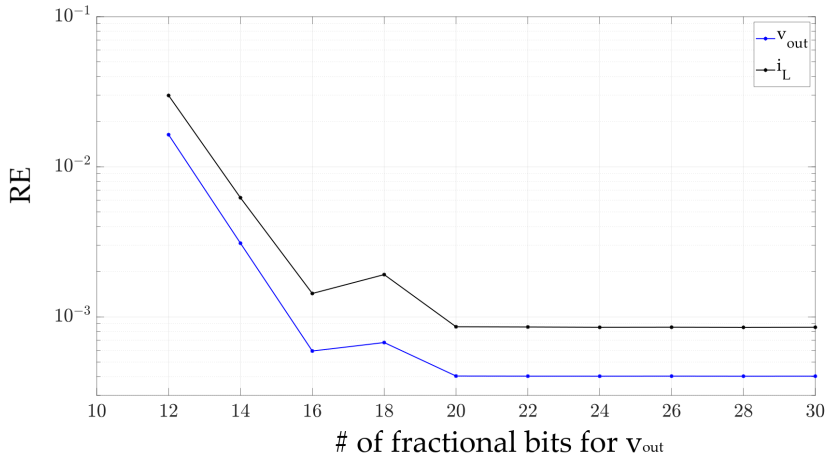

Figure 6. RE used as a reference signal for computing v_{out} (blue) and i_L (black) .

A biparametric simulation study is made to clarify candidates for the bottleneck. A sweep over the number of bits for the fractional part of v_{out} and another sweep over the number of bits for the fractional part of i_L are made for four different scenarios. The results from all of them—(a), (b), (c), and (d)—are presented in Figure 7. Every chart shows the RE with a different number of bits for the fractional part of i_L versus a range of bits for the fractional part of v_{out}. They also highlight the optimum number of bits for the fractional part of v_{out}, given back by the automated algorithm.

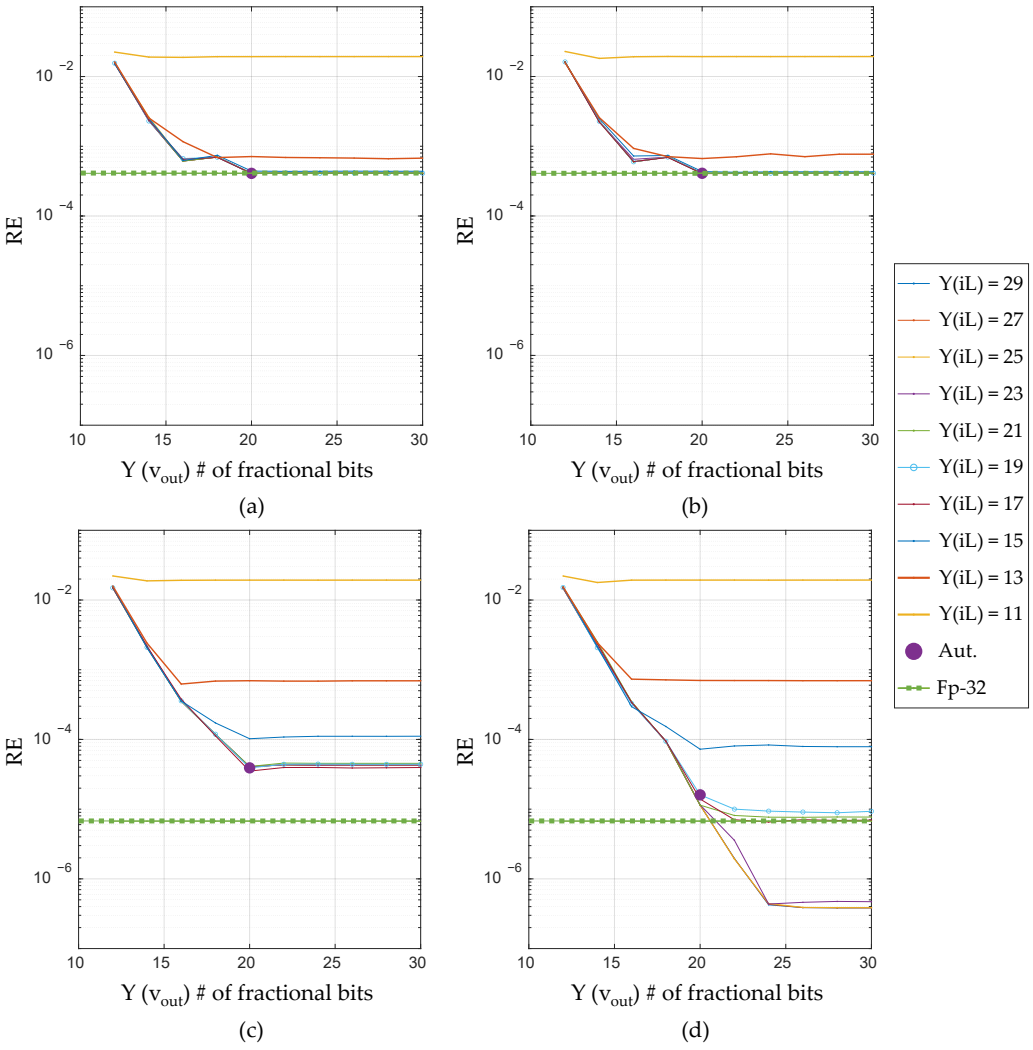

Figure 7. RE for four scenarios: (**a**) WL = 12 for input–outputs and constants; (**b**) WL = 12 bits for input–outputs and WL = 22 for constants; (**c**) WL = 12 for input–outputs and WL = 22 for constants; and (**d**) WL = 22 for every group.

An additional simulation based on a 32-bit floating-point number representation (instead of a fixed-point one) is included in the study. A 32-bit floating-point number is a very common choice in the industry because it has the advantage of floating-point (not choosing the integer and fractional bits for each signal), while also keeping the necessary HW resources within reasonable limits compared to 64-bit numbers. The additional simulation also meets the 12-bit restriction for the inputs and outputs, and it is addressed by the green horizontal line to help clarify the error results in Figure 7. Additionally, it is meaningful in the synthesis results.

On the other hand, such a simulation acts as a baseline for comparison in every scenario in Figure 7 coming from the proposed method.

The setup for every scenario is as follows:

(a) Theoriginal WL defined by the proposed method for the input–outputs and constants. In this case, both are set to WL = 12. The minimum RE is around 4×10^{-4}. A similar result can be seen in Figure 6. In fact, the v_{out} simulation in Figure 6 is the same as $Y(iL) = 19$ in Figure 7a, which is the result coming from the proposed method. However, in Figure 7a, there are different simulations with other fractional values for i_L to verify that the automatic selection comes out with the optimum

number of bits. The difference between the RE coming from the 32-bit floating-point and the RE from the automatic selection ($Y_{v_{out}} = 20$, $Y_{i_L} = 19$) is smaller than 1% (0.64%).

(b) Keeping WL = 12 bits for input–outputs and increasing the constants by 10 bits, for a total WL = 22 bits. As seen in Figure 7b, all the results for this case are very similar to those of case (a). Thus, the signals that increased their WL for (b) do not have any additional contribution from case (a). The minimum RE is also around 4×10^{-4}, so there is no improvement. Thus, constants are not the bottleneck.

(c) Keeping WL = 12 bits for constants and increasing the input–outputs by 10 bits, for a total WL = 22 bits. If the WL increases one bit for every signal, the error should be divided by two. Simulations from $Y = 17$ to $Y = 29$ show that RE drops to be around 4×10^{-5}, a lower order of magnitude than in previous scenarios. That means that increasing 10 bits reduces the RE by only a factor of 10, when it should be reduced by $2^{10} = 1024$ if this were the only bottleneck. Therefore, the input–outputs that increased their WL in this case were part of the bottleneck, but not the only ones responsible for the system error.
Regarding the 32-bit floating-point (green horizontal line), the input–outputs have WL = 22 bits for the analysis coherence. This chart shows that increasing input-outputs by 10 bits is not enough to achieve the results shown by the baseline green horizontal line, despite RE being smaller than in previous scenarios. Thus, input–outputs are not exclusively the bottleneck.

(d) Increasing input–outputs by 10 bits and raising constants to a total of WL = 22 bits in both groups. The results shown in Figure 7d fall further, to 4×10^{-7}, which is approximately 2^{10} lower than in Figure 7a. This is in good accordance with the expected results because, for obtaining a decrease in the error of 2^{10}, 10 bits have been added to both groups. This means that both input–outputs and constants were acting as bottlenecks in the original scenario (a) obtained by the proposed method, showing its validity.

Looking at the automatic selection, although its RE is a bit higher than the RE from the 32-bit floating-point, its RE remains around 4×10^{-5}. Both results come from synthesizable models and keep the same WL restriction for the input–output signals. However, hardware resources are different, as shown by the synthesis results.

Apart from the error results, it is also important to check the synthesis results. The proposed method's advantage is in obtaining a reasonable error, which has been shown to be as good as 32-bit floating-point, but it also decreases the necessary HW resources. Let us examine the HW resources for both methods. Tests have been conducted on an ad hoc HIL system implemented in an FPGA Xilinx Zynq Zybo (xc7z010cllg400-1) with the Vivado 2021.2 tool. It has not been tested on a commercial HIL system because these systems do not allow the user to configure the format of the internal signals, so their width cannot be tuned. The setup is shown in Figure 8.

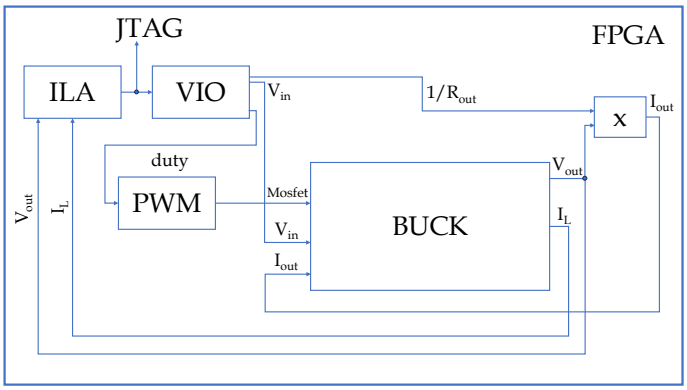

Figure 8. Block diagram setup for the experimental tests.

Figure 8 shows the block diagram setup for the experimental tests. It is important to notice that the experimental setup is entirely inside an FPGA. The central block is called BUCK, which is the HIL system explained in Section 2. The internal signal width of this block is updated for each experiment. The rest of the blocks are used for testing the HIL system. For instance, the PWM block (pulse-width modulator) generates a gate signal (MOSFET), which is the main input of the HIL system. Both blocks, PWM and BUCK, are synchronized using the same clock, so no error is due to synchronization issues. The other inputs of the HIL system are V_{in} and I_{out} (input voltage and load current, respectively). In order to change them easily, they are defined by the user using the VIO block, which is a virtual input/output controlled by a computer. The user defines the values of these signals in the computer interface, and the new values are sent to the FPGA via the JTAG (joint test action group) cable, which is the four-pin interface commonly used in FPGAs for programming and debugging. The outputs of the HIL system, mainly V_{out} and I_L (output voltage and inductor current, which are the state variables), are sent to the ILA (integrated logic analyzer) block. The ILA is a customizable logic analyzer core capable of registering the values of internal FPGA signals in order to send them later to the computer through the JTAG cable. The values are stored in the internal RAM memory of the FPGA and later sent to the computer visualization tool through the JTAG cable, so no additional wires are needed for debugging, just the JTAG cable, which is also used for programming. Finally, the load current (I_{out}) is generated by multiplying V_{out} by $1/R_{out}$. This value, $1/R_{out}$, is also sent through the VIO, so the resistor load can be changed by the user.

The synthesis results are summarized in Table 2. Fx-p is the fixed-point model of the proposed method, Fx-p + 10 is the model of the proposed method with 10 bits added to all signals, and 32-bit Fl-p is the 32-bit floating-point model. In the design, the number of LUTs and DSPs point to the size of the combinational part, while the number of FFs represents the sequential part, respectively. The buck converter model used in this work only registers two points (the two state variables), so almost the entire design is combinational. Thus, the number of LUTs is higher than the number of FFS. A DSP (digital signal processor) block consists of computing arithmetic operations (addition and or multiplication). Therefore, DSPs are also part of the combinational part of the circuit.

Table 2. Synthesis results.

Model	LUTs	FFs	DSPs	$T_{clk,min}$
Fx-p	338	61	4	6.326
Fx-p + 10	455	61	6	7.619
32-bit Fl-p	1029	73	4	16.885

The 32-bit floating-point model has a noticeably higher number of LUTs (1029) than the two fixed-point models (338) and (455), correspondingly. Additionally, the number of FFs, as well as the delay, are also higher. Therefore, the 32-bit floating-point model requires more than double the area and computing delay, leading to the method proposed as the best choice for saving the design area and improving the delay without jeopardizing accuracy.

As a comparison, the work presented in [26] proposes a manual adjustment of the WL. In that study, all the signals involved in the system must be individually analyzed to set the WL. The novelty of the current paper, compared to the approach in [26], lies in two fundamental aspects. Firstly, the proposed method automatically obtains the data by extracting them from the simulation, automatically distinguishing the key values in both transient and steady-state regimes. Another novelty concerning this work is that, given the wide variety of ADCs and DACs available on the market, each one with different resolutions in their inputs and outputs, it was not very practical to develop a system valid for only one resolution. This paper proposes integrating the resolution constraints for both the output and input of the converter as input variables of the system. The advantage of this method compared to others presented in the bibliography is that it can now be

used with any ADC or DAC converter available on the market, making it applicable in countless scenarios.

4. Conclusions

This paper proposes an automatic method to select the number of bits for all the signals of the HIL model in a fixed-point switched converter. It only requires a single 64-bit floating-point simulation: steady state and transients, in which extreme values that are reached are included. From the values reached in this simulation, minimum starting bits are chosen for the integer and fractional parts. However, the fractional parts are subsequently increased, as detailed in the proposed method, in order to spread the weight of the error among them and to ensure that no signal becomes the bottleneck alone, i.e., no wasted bits.

In addition, the proposed method takes into account the boundary conditions imposed by the inputs and outputs of the model via ADCs and DACs. It is the only method that includes these boundary conditions. Additional simulations show that the number of bits chosen by the method from a single simulation is adequate.

Compared to other works of WL determination, the method shown in this work allows the user to use a single simulation instead of multiple ones, and also to take into account the available resolution of its inputs and outputs determined by ADCs and DACs. The proposed method provides the optimal WL for all signals involved in that specific system. On the other hand, the synthesis results indicate a significant improvement over classical minimum design effort methods, such as representing all signals in a 32-bit floating-point. The results in both area and time are more than twice as good in the proposed method compared to the classical 32-bit floating-point method.

Supplementary Materials: The following supporting information can be downloaded at: https://github.com/carlosqgomez/Resol_Buck_Boundary_Conditions.git.

Author Contributions: M.A.G.-V.: Conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing—original draft preparation and visualization; C.Q.G.M.: Conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing—review and editing and visualization; M.S.M.-G.: Conceptualization, methodology, validation, formal analysis, investigation, writing—review and editing, visualization, resources and funding acquisition; A.d.C.: Conceptualization, methodology, validation, formal analysis, investigation, writing—review and editing, visualization and resources. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data may be accepted in Supplementary Materials.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Estrada, L.; Vaquero, J.; Rodriguez-Lorente, A.; Arau, J.; de Castro, A.; Sanchez, A.; Vazquez, N. Asynchronous and Decoupled HIL Simulation of a DC Nanogrid. *Electronics* **2022**, *11*, 2045. [CrossRef]
2. Kaven, L.; Frehn, A.; Basler, M.; Jassmann, U.; Roettgers, H.; Konrad, T.; Abel, D.; Monti, A. Impact of Multi-Physics HiL Test Benches on Wind Turbine Certification. *Energies* **2022**, *15*, 1336. [CrossRef]
3. Zamiri, E.; Sanchez, A.; Yushkova, M.; Martínez-García, M.S.; de Castro, A. Comparison of Different Design Alternatives for Hardware-in-the-Loop of Power Converters. *Electronics* **2021**, *10*, 926. [CrossRef]
4. Tumasov, A.V.; Vashurin, A.S.; Trusov, Y.P.; Toropov, E.I.; Moshkov, P.S.; Kryaskov, V.S.; Vasilyev, A.S. The Application of Hardware-in-the-Loop (HIL) Simulation for Evaluation of Active Safety of Vehicles Equipped with Electronic Stability Control (ESC) Systems. *Procedia Comput. Sci.* **2019**, *150*, 309–315. [CrossRef]
5. Mayet, C.; Idkhajine, L. Introduction to Hardware-In-the-Loop (HIL) simulations of electrical power systems. In *Encyclopedia of Electrical and Electronic Power Engineering*; García, J., Ed.; Elsevier: Oxford, UK, 2023; pp. 590–599. [CrossRef]
6. DaneshvarDehnavi, S.; Negri, C.; Bayne, S.; Giesselmann, M. Dynamic Voltage Restorer (DVR) with a novel robust control strategy. *ISA Trans.* **2022**, *121*, 316–326. [CrossRef] [PubMed]
7. Glumac, S.; Varga, N.; Raos, F.; Kovacic, Z. Co-simulation perspective on evaluating the simulation with the engine test bench in the loop. *Automatika* **2022**, *63*, 275–287. [CrossRef]

8. Xie, B.; Wang, S.; Wu, X.; Wen, C.; Zhang, S.; Zhao, X. Design and hardware-in-the-loop test of a coupled drive system for electric tractor. *Biosyst. Eng.* **2022**, *216*, 165–185. [CrossRef]
9. Belhaouane, M.M.; Almaksour, K.; Papangelis, L.; Gomozov, O.; Colas, F.; Prevost, T.; Guillaud, X.; Van Cutsem, T. Implementation and Validation of a Model Predictive Controller on a Lab-Scale Three-Terminal MTDC Grid. *IEEE Trans. Power Deliv.* **2022**, *37*, 2209–2219. [CrossRef]
10. Rataj, D.; Slawik, D.; Wrobel, K.; Tomczewski, K. A fast switched reluctance motor controller based on FPGA. *ITM Web Conf.* **2018**, *19*, 01028. [CrossRef]
11. Reddy, B.P.; Murali, A.; Kumar, K.S. A low cost sense coil based position sensing system for SRM implemented in a SoC FPGA. In Proceedings of the IECON 2017—43rd Annual Conference of the IEEE Industrial Electronics Society, Beijing, China, 29 October–1 November 2017; pp. 6603–6606. [CrossRef]
12. Lamo, P.; de Castro, A.; Sanchez, A.; Ruiz, G.A.; Azcondo, F.J.; Pigazo, A. Hardware-in-the-Loop and Digital Control Techniques Applied to Single-Phase PFC Converters. *Electronics* **2021**, *10*, 1563. [CrossRef]
13. Shahbazi, M.; Poure, P.; Saadate, S.; Zolghadri, M.R. FPGA-Based Reconfigurable Control for Fault-Tolerant Back-to-Back Converter Without Redundancy. *IEEE Trans. Ind. Electron.* **2013**, *60*, 3360–3371. [CrossRef]
14. Robert, S. Ac/ac multicell converter analysis on the basis of FPGA-based model of the converter. *Prz. Elektrotechniczny* **2007**, *83*, 28–36.
15. Zamiri, E.; Sanchez, A.; Martínez-García, M.S.; de Castro, A. Analysis of the aliasing effect caused in hardware-in-the-loop when reading PWM inputs of power converters. *Int. J. Electr. Power Energy Syst.* **2022**, *136*, 107678. [CrossRef]
16. Anurag, N.; Nath, S. Effect of Commutation Control on Switching Frequency of SiC based Four Quadrant Switch. In Proceedings of the 2021 National Power Electronics Conference (NPEC), Bhubaneswar, India, 15–17 December 2021; pp. 1–6. [CrossRef]
17. Riché, T.L.; Nagle, J.; Xu, J.; Hubbard, D. Converting Executable Floating-Point Models to Executable and Synthesizable Fixed-Point Models. In Proceedings of the 22nd International Conference on Model Driven Engineering Languages and Systems, MODELS '19, Munich, Germany, 15–20 September 2019; IEEE Press: Piscataway, NJ, USA, 2019; pp. 354–361. [CrossRef]
18. Xu, Y.; Shuang, K.; Jiang, S.; Wu, X. FPGA Implementation of a Best-Precision Fixed-Point Digital PID Controller. In Proceedings of the 2009 International Conference on Measuring Technology and Mechatronics Automation, Zhangjiajie, China, 11–12 April 2009; Volume 3, pp. 384–387. [CrossRef]
19. Lee, D.U.; Gaffar, A.; Mencer, O.; Luk, W. MiniBit: Bit-width optimization via affine arithmetic. In Proceedings of the Proceedings. 42nd Design Automation Conference, Anaheim, CA, USA, 13–17 June 2005; pp. 837–840. [CrossRef]
20. Kulisz, J.; Chmiel, M.; Krzyzyk, A.; Rosół, M. A Hardware Implementation of Arithmetic Operations for an FPGA-based Programmable Logic Controller. *IFAC-PapersOnLine* **2015**, *48*, 460–465. [CrossRef]
21. Cui, Y.; Chen, B.; Zhang, S. Design of Floating-point Operation Based on FPGA and it's Application. In Proceedings of the 2006 8th international Conference on Signal Processing, Guilin, China, 16–20 November 2006; Volume 4. [CrossRef]
22. Channappanavar, R.; Mishra, S. Impact of Non-Linear Commutation Delay on the Performance of Inductor Current Estimation Techniques. In Proceedings of the 2018 IEEE Energy Conversion Congress and Exposition (ECCE), Portland, OR, USA, 23–27 September 2018; pp. 4226–4231. [CrossRef]
23. Chuong, L.M.; Lam, S.K.; Srikanthan, T. Area-Time Estimation of Controller for Porting C-Based Functions onto FPGA. In Proceedings of the 2009 IEEE/IFIP International Symposium on Rapid System Prototyping, Paris, France, 23–26 June 2009; pp. 145–151. [CrossRef]
24. Liu, Q.; Qian, H. FPGA Delay Model Considering Logic-Level and Transistor-Level Parameters. In Proceedings of the 2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), Napa, CA, USA, 30 April–2 May 2017; p. 29. [CrossRef]
25. Grady, B.; Anderson, J.H. Synthesizable Heterogeneous FPGA Fabrics. In Proceedings of the 2018 International Conference on Field-Programmable Technology (FPT), Naha, Japan, 10–14 December 2018; pp. 222–229. [CrossRef]
26. Martínez-García, M.S.; de Castro, A.; Sanchez, A.; Garrido, J. Word length selection method for HIL power converter models. *Int. J. Electr. Power Energy Syst.* **2021**, *129*, 106721. [CrossRef]
27. Sung, W.; Kum, K.I. Simulation-based word-length optimization method for fixed-point digital signal processing systems. *IEEE Trans. Signal Process.* **1995**, *43*, 3087–3090. [CrossRef]
28. Kum, K.I.; Sung, W. Combined word-length optimization and high-level synthesis of digital signal processing systems. *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **2001**, *20*, 921–930. [CrossRef]
29. Goni, O.; Sanchez, A.; Todorovich, E.; de Castro, A. Resolution Analysis of Switching Converter Models for Hardware-in-the-Loop. *IEEE Trans. Ind. Inform.* **2014**, *10*, 1162–1170. [CrossRef]