An Evaluation of a High-Level Synthesis Approach to the FPGA-Based Submicrosecond Real-Time Simulation of Power Converters

Federico Montano, Tarek Ould-Bachir, *Member, IEEE*, and Jean Pierre David, *Member, IEEE*

***Abstract*—This paper evaluates the benefits of using a high-level synthesis (HLS) tool to develop field-programmable gate array (FPGA)-based real-time simulators for power electronics systems. The investigated workflow generates a synthesizable hardware description from a system level C-code along with a set of directives that specify performance criteria such as area utilization and timing closure requirements. The performance of the HLS approach is evaluated for different circuit sizes and target clock frequencies. Results show that HLS can be used for hardware-in-the-loop (HIL) applications when the circuit to be simulated is small and the target clock frequency is not too high (up to 100 MHz). For larger circuits and higher clock frequencies, HLS will either require a simulation time-step that is too large for real-time simulation purposes, or will tend to use almost all of the FPGA resources.**

***Index Terms*—Field-programmable gate array (FPGA), high-level synthesis (HLS), hardware-in-the-loop (HIL), power electronics, real-time simulation.**

I. Introduction

HARDWARE-IN-THE-LOOP (HIL) simulation is recognized as an economic and efficient industrial prototyping approach for the design of power system controllers [1]. The high switching frequency requirements of modern power converters (>20 kHz) [2]–[6] are however very challenging for real-time simulators because the simulation time-steps for such systems must be in the submicrosecond range to meet criteria such as a fine time resolution on the gate signals and accurate simulation results [7]. In the recent years, filed-programmable gate array (FPGA) devices have been used in HIL simulators to fulfill these requirements, to enhance their computing capabilities and to reduce the total latency of the simulation loop [8].

However, FPGA programming is a difficult task that requires specialized skills in hardware design. FPGA designs are usually based on register transfer level (RTL) abstraction to describe a digital circuit. At that level, registers contain the current state of the system. At each clock cycle, the state is updated with a combinatorial function of itself and inputs. The functions involved are typically implemented with arithmetic and logic units or dedicated resources [9]. The low abstraction level of RTL increases the development cost and the time to market of FPGA-based simulators for HIL systems, thus, restricting their use.

A useful design guideline to develop FPGA-based real-time simulators is presented in [10] and summarized in Fig. 1. Passing from system specification to HIL experimental validation involves several steps including modeling, partitioning, coding and validation. Many constraints must be satisfied including libraries and resources availability, timing, and accuracy.

The development of FPGA-based systems is in itself a multistep process as depicted in Fig. 2 [11]. Passing from system

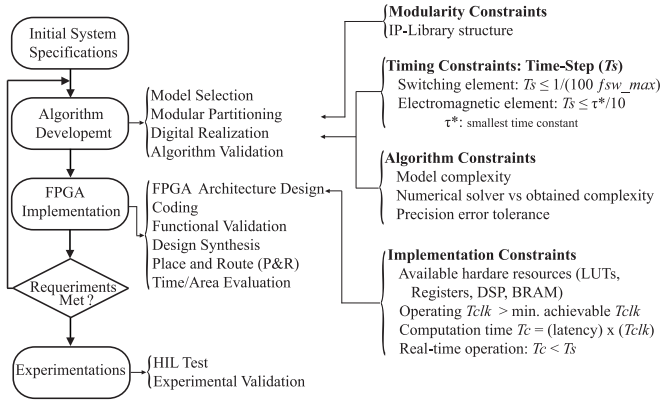

Fig. 1. Design guideline for FPGA-based real-time simulator design. Adapted from [10].

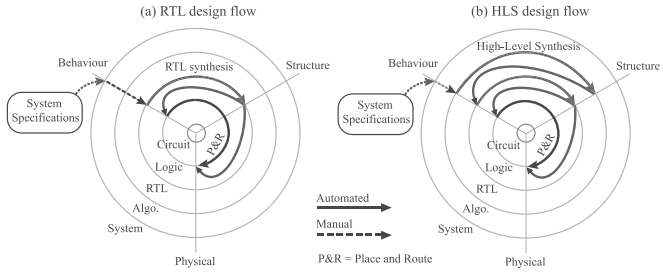

Fig. 2. Y-chart for (a) RTL and (b) HLS. Adapted from [11].

Manuscript received October 31, 2016; revised February 22, 2017 and April 21, 2017; accepted May 24, 2017. Date of publication June 16, 2017; date of current version November 16, 2017. This work was supported in part by Mitacs and in part by OPAL-RT Technologies. *(Corresponding author: Federico Montano.)*

F. Montano and J. P. David are with the Department of Electrical Engineering, École Polytechnique de Montréal, Montreal, QC H3T 1J4, Canada (e-mail: federico.montano@polymtl.ca; jpdavid@polymtl.ca).

T. Ould-Bachir is with Opal-RT Technologies, Montreal, QC H3K 1G6, Canada. (e-mail: ouldbachir@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2017.2716880

description to a circuit description requires several steps on three different axes (behavior, structure, physical). Most error prone conditions are present in algorithmic development, hardware description and their verification (typically done by human beings) while logic and circuit transforms are automatically computed by tools, which are supposed to reduce the presence of errors.

Working at an abstraction level higher than RTL, with dedicated languages and tools, is believed to reduce the need for hardware specialists and to increase productivity [12], [13]. Important research efforts have been invested in the recent years to allow efficient exploitation of the FPGA potential with high-level programming languages [14].

Nowadays, high-level synthesis (HLS) tools available in the market such as Vivado HLS (VHLS) from Xilinx [15], and OpenCL SDK from Altera [16] allow the use of high-level languages to ease design and verification of hardware. Since hardware under development must meet certain performance criteria, HLS tools offer the use of configurations and directives to support pipelining, duplication, and other digital design techniques used to improve hardware performance [17]. Recent works suggest that such tools could be employed for developing high-performance computing units, designated hereafter as hardware solvers (HS), for real-time simulation applications [14], [18].

The main goal of this paper is to evaluate the benefits of using VHLS to design FPGA-based HS with reduced simulation time-steps for HIL applications. The evaluation takes into account the design operation frequency, the resources requirements as well as the accuracy of the simulation.

The remainder of this article is organized as follows: Section II reviews the works related to this research. Section III presents a design exploration comparing the performances of VHLS against custom design. Section IV presents case studies where actual power converters are considered. A conclusion is given in Section V.

II. Related Work

FPGA devices are used in HIL simulators to model and simulate power converters with small time-steps. Reported time-steps vary from 12.5 ns [7], [19] to $0.5 - 1\ \mu$s [20], [21]. Most of the time, a low-level design approach is used to describe the FPGA design, whether using hardware description languages (HDLs) such as VHDL or Verilog or specialized blockset toolboxes such as Xilinx's System Generator [22]. Additional examples can be found in [6], [23]–[25].

The number format adopted for these implementations are either fixed-point arithmetic [7], [20], [26] or floating-point arithmetic [6], [21], [24]. Fixed-point designs are characterized by smaller footprints and time-steps, whereas floating-point designs offer better accuracy and larger dynamic range.

The HLS approach to FPGA-based real-time simulation in the field of power electronic systems is at its early stage of exploration. Very few works have been reported to date. This includes [27], an FPGA-based feedback controller; as well as [14], [18], a controller for a resonant half-bridge power converter, and its HIL emulator for testing heating appliances. In each case, high-level description was demonstrated to fulfill the expected performances and the small simulation time-step was highlighted. However, the small size of the simulated circuits does not exploit the full potential of current FPGA and the related works do not explore the possible pitfalls associated with larger circuits.

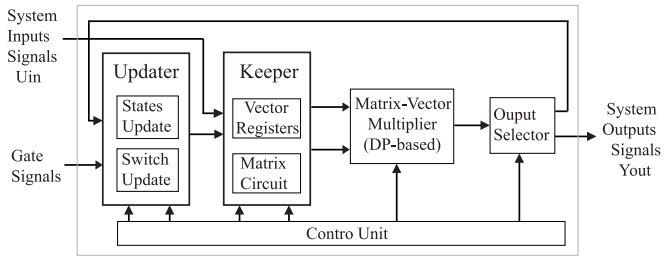

Fig. 3. Custom design HS architecture.

More generally, none of the aforementioned works apprehended the evaluation of the HLS approach in broader terms for the HIL simulation context. Such evaluations do exist for other application fields: Inggs *et al.* [28] offer a comparison between using HLS tools and other technologies in financial engineering; Homsirikamol and Gaj [29] compare HLS to hand-written code in cryptography; Reiche *et al.* [30] evaluate HLS for image processing; Cong *et al.* [12] evaluate the use of HLS in telecommunications. However, these studies are concerned with using HLS for developing FPGA-accelerated computing engines, whereas the objective of this study is to conduct a thorough evaluation of the benefits of HLS for HS with low calculation time-steps as the primary target result.

III. Design Exploration

A HS performs mainly a matrix-vector multiplication—refer to Appendix A for details about power circuit modeling. This operation has a reduced data dependency, which allows a high degree of parallelism. To evaluate the effectiveness of VHLS for the development of HSs, we conduct a design exploration study investigating the relationship between the size of the electric circuit, the achievable time-step, the working frequency of the FPGA design, and the cost of the implementation. All the HSs are designed using the following:

1) custom design approach;
2) VHLS approach constrained by resources;
3) VHLS approach constrained by latency.

A. Custom Design

In this approach, the HS is implemented using System Generator v.14.4 from Xilinx. Fig. 3 presents the architecture of the custom designs, which consists of five fundamental modules:

1) The updater is formed of two logic blocks that analyze gating signals and state variables and dynamically update the vector registers file.
2) The keeper contains the vector registers and the matrix representing network equations. The matrix is saved in RAM blocks arranged to allow simultaneous access to multiple elements at a time. RAM blocks are filled using

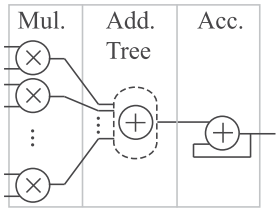

Fig. 4. DP operator structure.

TABLE I
DP LATENCY FOR THE VARIOUS OPERATION FREQUENCIES

Frequency (MHz)	40–50	100	200	250	320
Latency (Clk Ticks)	2	5	13	14	16

a MATLAB m-script. The circuit inputs are directly sent to the input vector registers.

3) The matrix-vector multiplier is formed by eight dot-product (DP) units working in parallel. The DP operator is a custom design optimized for low-latency floating-point operation [31]. Each DP has eight multipliers connected to an adder tree and terminated by an accumulator as depicted in Fig. 4. Each DP uses 16 DSP blocks. Table I shows the latency of the DPs for each working frequency.
4) The output selector routes matrix-vector multiplication results. History terms (referred to as states in the following) are fed back to the updater module for the next computing cycle, while the remainder of the results form the outputs of the HS.
5) The control unit schedules all operations for vector updating, memory data reading, DP operation, data feedback, and outputs activation. It is a state machine formed by a counter and memory blocks. Since the scheduling is determined during design, the content of the memory blocks is precomputed by an m-script with respect to circuit specifications, and the target simulation time-step.

B. VHLS Approaches

For the two VHLS approaches discussed in Sections III-B1 and III-B2, the HSs are described by the C-code of Fig. 5 and implemented using Vivado HLS 2015.2. Even though directives can be introduced directly in the source code as pragmas, we have grouped them in reusable attachable directive files to facilitate design explorations without changing the source code. The whole design of the solver is implemented using the code structure of Fig. 5.

1) `switch_update(...)` updates switch statuses according to the gating signals, previous switch statuses, and the currents and voltages associated with the switches (`swhist`). This function is not used in this design exploration but is used in Section IV.
2) The `vector_forming` loop groups circuit inputs, states, and switches conditions to update the state/input vector.
3) `mvmult(...)` takes the vector composed by `vector_forming` and performs matrix-vector multiplication. This is the most time and resource consuming function. It is composed of two nested loops with a multiply accumulate operation at the inner loop level.
4) The `assign_vects` loop separates the matrix-vector multiplication results and builds the state vector (`x`), switch voltages and currents (`swhist`), and the output vector (`y`).

```
void hw_solver(
  float y[nb_outputs],
  float u[nb_inputs],
  int gates){

  static int sw = 0;
  static float x[nb_states] = {...};
  static float swhist[2*nb_switches] = {...};
  const float A[nb_rows][nb_columns] = {...};
  float b[nb_columns];
  int i, j;

  // updates switch status
  sw = switch_update(gates, sw, swhist);

  vector_forming: for(i=0; i<nb_columns; ++i){
    // puts u, x and
    // sub values from swhist into b
  }

  // Matrix-vector multiplication
  float * results = mvmult(A, b);

  assign_vects: for(i=0; i<nb_rows; ++i){
    // puts sub values from results into
    // swhist, x and y
  }
}
```

Fig. 5. Excerpt from the C-code used for the design of the HS in VHLS.

VHLS automatically generates inputs/outputs, control signals, internal control logic, and performs scheduling and binding processes. Voltages and currents use floating-point variables while an integer variable is used for gating signals. VHLS maps the variables at the top level function to input/output ports automatically. Matrix values are internally saved in a constant matrix that can be mapped into memory blocks or registers.

1) VHLS Approach Constrained by Resources: In our first attempt, and in order to provide a fair comparison between the custom designs and HLS results, VHLS is constrained by resource utilization in a way that allows the tool to match the computing power of the custom designs. Hence, allocation directives are applied to limit the number of arithmetic operators to 64 adders and 64 multipliers. The single precision multipliers and adders instantiated by VHLS require, respectively, 3 and 2 DSP blocks, which leads to a consumption of $(2+3)\times 64 = 320$ DSP blocks. Fig. 6 shows the directives used when VHLS is constrained by resources.

2) VHLS Approach Constrained by Latency: In a second round of HLS implementations, pipeline and latency directives are set to force VHLS to generate a design with the lowest time-step possible for each solver design. In this case, the

```
set_directive_array_partition -type complete
        -dim 1 "hw_solver" yout
set_directive_array_partition -type complete
        -dim 1 "hw_solver" uin
set_directive_top "hw_solver"
set_directive_pipeline "hw_solver"
set_directive_allocation -limit 64
        -type operation "multiplication" fmul
set_directive_allocation -limit 64
        -type operation "multiplication" fadd
```

Fig. 6. Directives for the VHLS approach constrained by resources.

```
set_directive_array_partition -type complete
        -dim 1 "hw_solver" yout
set_directive_array_partition -type complete
        -dim 1 "hw_solver" uin
set_directive_pipeline -II $II "hw_solver"
set_directive_latency -max $lat "hw_solver"
set_directive_top "hw_solver"
```

Fig. 7. Directives for VHLS approach constrained by latency.

tool is free to use as much resources as needed. Fig. 7 shows the directives used when the tool is constrained by latency, aiming for the minimum achievable time-step.

C. Implementation Results

The design exploration targets Xilinx's Kintex XC7K325T-FFG676-1 FPGA [32]. A wide range of clock frequencies are investigated, namely 40, 50, 100, 200, 250, and 320 MHz (resp. 25, 20, 10, 5, 4, and 3.125 ns). The custom designs are capable of handling all frequencies, whereas VHLS designs were only successful handling frequencies up to 200 MHz. The designs are tested for various number of states ranging between 2 and 60, and their performance evaluated by assessing the achievable minimum time-step and the FPGA area occupation (cost). Results are presented in Figs. 8 and 9. Dashed lines are synthesis estimates resulting from situations where either VHLS encounters timings issues or the resources utilization exceeds the capacity of the target FPGA.

Fig. 8 gives the minimum time-step for the three design approaches as a function of the FPGA clock period and the number of states in the circuit. For a given clock frequency, it is observed that the time-step grows quadratically with respect to the circuit size. The growth rate is more pronounced for lower frequencies. Hence, achieving high clock frequencies is crucial when designing HSs, more so when large circuits are considered, a goal that VHLS presently fails to fulfill for clock frequencies exceeding 100 MHz (respectively, a clock period of less than 10 ns). However, the clock frequency has little impact when the circuit size is small.

Fig. 9 gives the cost for the three design approaches as a function of the FPGA clock period and the number of states in the circuit. Typically, the cost of an FPGA implementation is a composite metric given by the number of look-up tables (LUTs), the number of registers, and the number of DSP and RAM blocks. Here, a global cost is considered, computed as the maximum percentage of the resources used among LUTs, registers, DSP, and RAM blocks. Such a metric is a good indicator of the number of HSs that can be implemented on a single target FPGA.

From Fig. 9(a), one observes that custom designs show a flat 15% area occupation for all clock frequencies and circuit sizes. The area occupation is dominated by the DSP consumption. When VHLS is constrained by resources [see Fig. 9(b)], the area occupation is also dominated by the DSP consumption, at least for circuits with up to 30 states. For circuits with more than 30 states, the resource consumption is dominated by reconfigurable logic and grows quickly with respect to the circuit size. Finally, when VHLS is constrained by latency [see Fig. 9(c)], the area occupation is very high (close to 100%) for both DSP blocks and reconfigurable logic.

IV. Case Study: Power Converter

This section considers a three-level neutral-point clamped (NPC) converter connected to an RLE load, as illustrated in Fig. 10. This is equivalent to a 41-state circuit.

The switches in the circuit are modeled using the associated discrete circuit (ADC) model [33] to keep network equations fixed regardless of switch statuses—refer to Appendix A for a presentation of the switch model. Table II gives the parameters for the power converter and for its control.

A. Implementation Results

The HS for the power converter is conceived using the three aforementioned design approaches, i.e.,

1) custom design;
2) VHLS constrained by resources;
3) VHLS constrained by latency.

All implementations target a 100 MHz clock frequency (10 ns clock period), a design tradeoff motivated by the results of Section III-C and aiming to guarantee fairness in the comparison of VHLS results against our custom implementations. Table III gives synthesis estimates for the three implementation approaches.

The custom design clearly outperforms VHLS results in terms of resources and speed. The minimum time-step is smaller by more than two folds, and the reconfigurable resource consumption by 1.5 to 2 folds. DSP block utilization for VHLS is increased by 150–450% due to the use of three DSP blocks per floating-point multiplier and two DSP blocks per floating-point adder, whereas the custom designs consumes only two DSP blocks per floating-point multiplication. Even if VHLS appears to be a very good choice because of its ease of use, when it comes to implementation of real power converters, VHLS is presently not cost-effective for real-time simulation requirements.

B. Offline Simulation Results

In this section, we evaluate the correctness of the computations executed by the proposed implementations. This is done from within MATLAB, in which test vectors are generated and

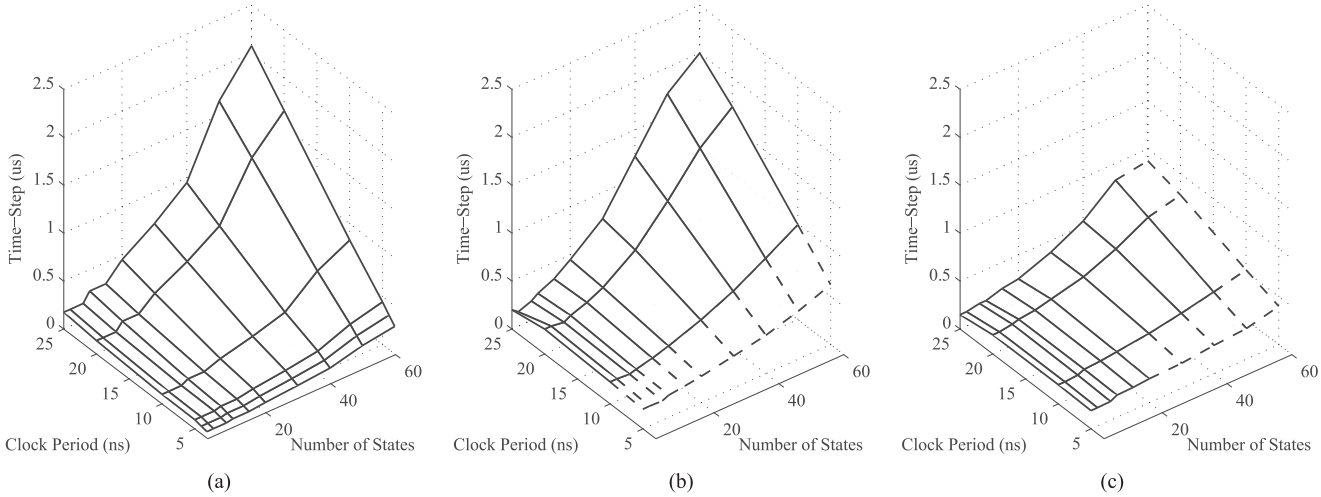

Fig. 8. Minimum time-step: (a) Custom design; (b) VHLS constrained by resources; (c) VHLS constrained by latency.

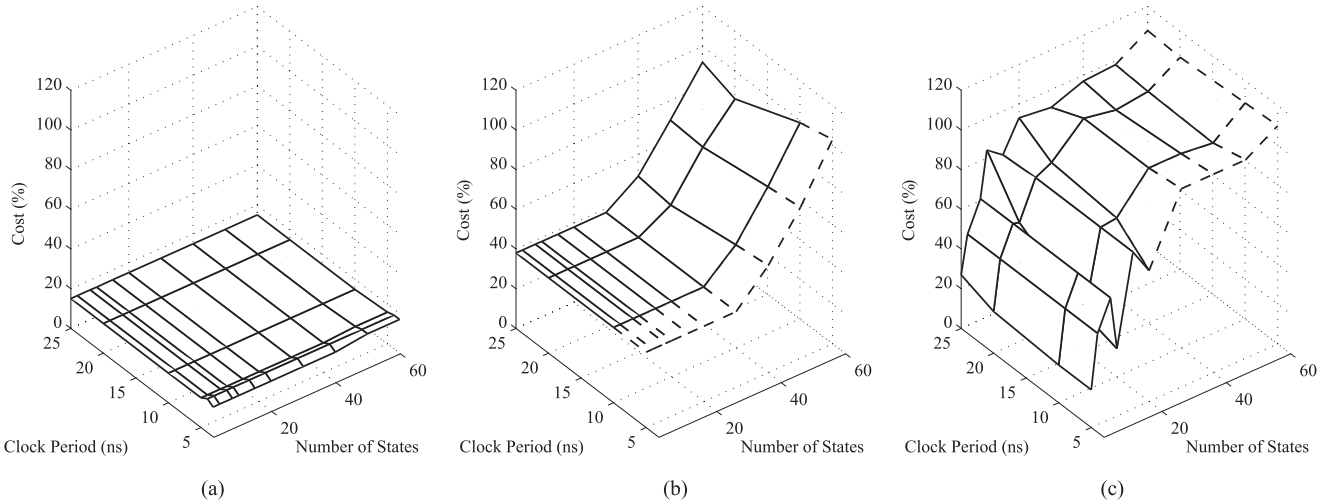

Fig. 9. Implementation cost: (a) Custom design; (b) VHLS constrained by resources; (c) VHLS constrained by latency.

simulation results compared against reference outputs. Expected simulation results are generated by running a m-script file using single and double precision arithmetics. Double precision results serve as a reference to all implementations, whereas single precision results are used to make sure that FPGA-based simulations compare favorably against IEEE-754 complying code.

The results presented hereafter consider the NPC converter with a time-step of 630 ns. The simulation is executed for a runtime of 10 ms during which the inverter is controlled in open loop by a sine pulse width modulation (PWM): the modulation sine wave has a frequency of 400 Hz, the carrier frequency is 25 kHz. A modulation index of 0.8 is used. For this evaluation, we only use the results from the implementation where VHLS is constrained by latency.

Fig. 11(a) presents the load current on phase a: i_a. Fig. 11(b) shows superimposed relative errors for single precision computations (m-code), the custom design as well as for VHLS. It demonstrates that both designs are similar in their accuracy

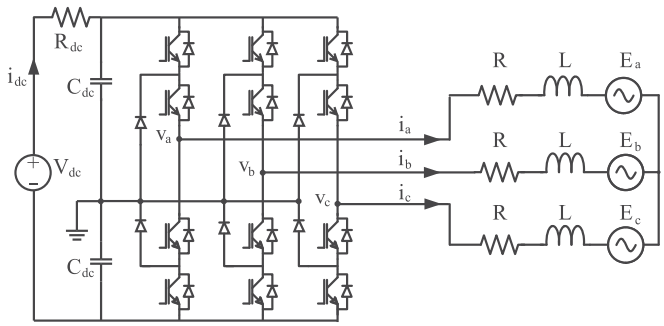

Fig. 10. Case study: NPC converter connected to an RLE load.

performance, and that the overall relative error is close to what is expected from a single precision IEEE-754 compliant code ["Single" in Fig. 11(b)], which is a relative error of less than 0.01%. The relative error is higher when i_a crosses zero, but this is to be expected as confirmed by the "Single" curve in

TABLE II
CIRCUIT PARAMETERS

Parameter	Value
V_{dc}	100 V
R_{dc}	0.4 Ω
C_{dc}	1 mF
R	0.4 Ω
L	0.1 mH
PWM Modulation Freq.	400 Hz
Modulation Index	0.8
PWM Carrier Freq.	25 kHz

TABLE III
SYNTHESIS RESULTS FOR THE KINTEX XC7K325T

	NPC Converter			
Metrics	Custom design	VHLS (*DSP)	VHLS (*Ts)	Available
Registers	21 136	50 635	83 272	407 600
LUTs	35 726	52 249	73 161	203 800
DSP	128	320	703	840
Clock (ns)	10	10	10	N/A
Min. Δt (ns)	310	680	630	N/A

(*DSP) = Constrained by resources,
(*Ts) = Constrained by latency.

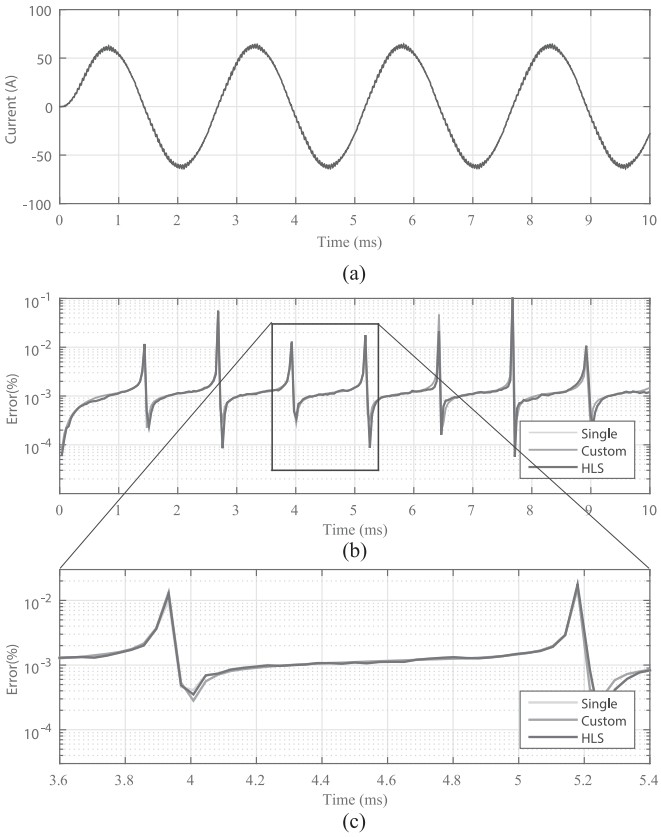

Fig. 11. Simulation results: (a) load current i_a; (b) relative errors; and (c) close-up view on relative errors.

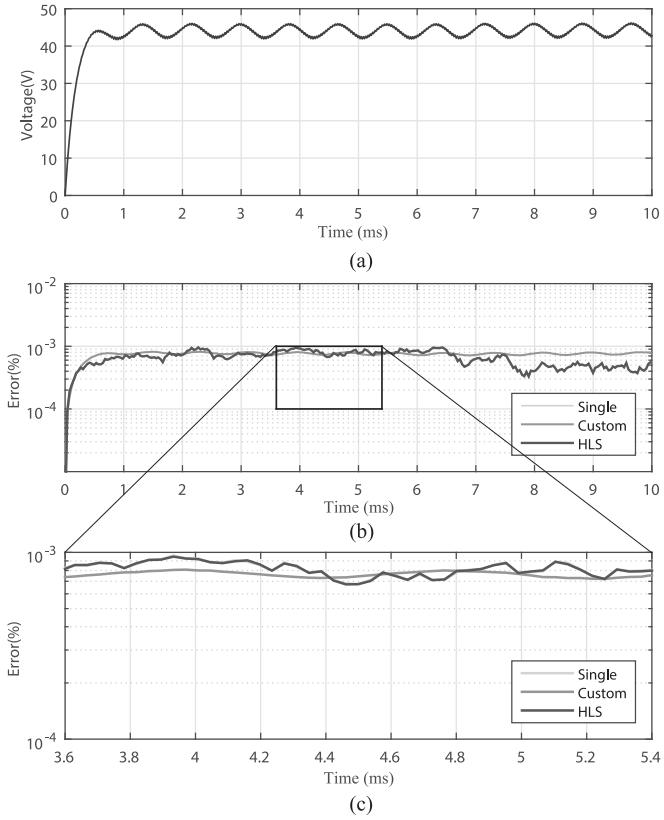

Fig. 12. Simulation results: (a) capacitor voltage V_{CAP}; (b) relative errors; and (c) close-up view on relative errors.

Fig. 11(b). Fig. 11(c) shows a close-up view from the relative error and shows that all three curves are almost identical.

In a similar way, Fig. 12(a) presents the voltage on capacitor C_{dc}: V_{CAP}. Fig. 12(b) show the superimposed relative errors for single precision computations, custom design, and VHLS. The error remains always inferior to 0.01%.

It is worth mentioning that the custom design is able to run the NPC at 310 ns (see Table III). The 630-ns time-step was used instead in order to keep a certain level of fairness toward VHLS. It is well known that reducing the simulation time-step increases accuracy. This observation simply confirms our conclusion that presently RTL design outperforms VHLS for larger circuits.

C. Online Simulation Results

This section presents the implementation and real-time simulation of the NPC converter. The FPGA design is composed of three main blocks:

1) A VHLS block constrained by latency for the NPC HS.
2) A PWM block for generating gates and E_{abc} signals.
3) An I/O interface controller.

Table IV gives the resource consumption for the real-time implementation. The complete system was implemented on a OP5707 [34] target, equipped with a Xilinx Virtex 7 7VX485TFFG1761-2.

TABLE IV
IMPLEMENTATION RESULTS FOR THE VIRTEX 7VX485T

Resource	Used	Available
LUTs	88 403	303 600
Registers	94187	607 200
DSP	754	2800
Block RAM	45	1030

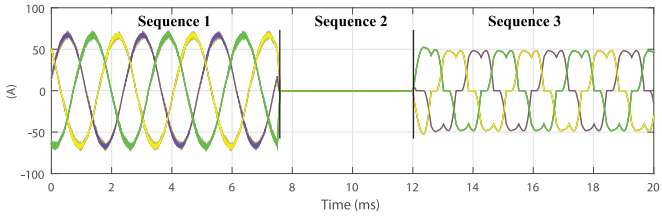

Fig. 13. Load currents from an SPS offline simulation with three sequences: (1) inversion mode; (2) gates turned-off; and (3) rectification mode.

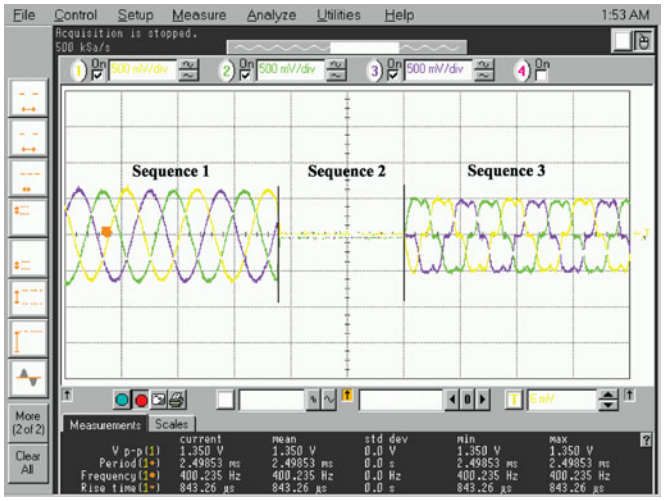

Fig. 14. Load currents from an online simulation with three sequences: (1) inversion mode; (2) gates turned-off; and (3) rectification mode.

Offline simulation results for a 20-ms simulation period have been generated using SimPowerSystems (SPS) to validate real-time results. The simulation encompasses three sequences:

1) Sequence 1: The PWM is active during 7.5 ms, putting the NPC in inversion mode.
2) Sequence 2: Gate signals are turned off during 4.5 ms to force currents to zero;
3) Sequence 3: Sinusoidal E_{abc} voltages are applied to put the converter in rectification mode for a 8-ms period.

Fig. 13 shows the currents on the load for offline simulation. Fig. 14 shows the load currents from the real-time simulation, captured by a Infiniium 54845A Oscilloscope. A gain of 0.01 is applied on the analog outputs, yielding a 50 V/div. Similarly, Figs. 15 and 16 show the voltage on capacitor during inversion mode for offline and real-time simulation, respectively. The channel is set to 20 V/div. Figs. 13–16 show a perfect match between SPS and the real-time simulation.

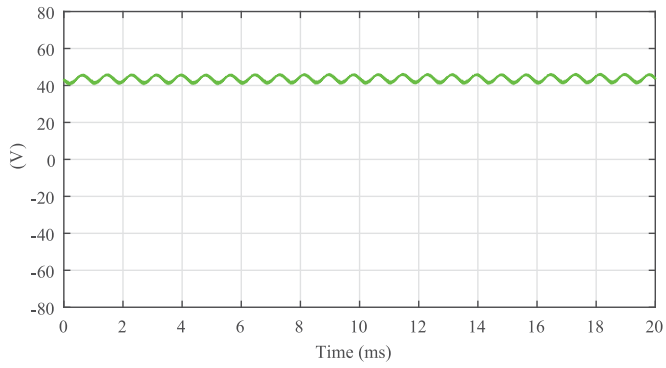

Fig. 15. Capacitor voltage from an SPS offline simulation in inversion mode.

Fig. 16. Capacitor voltage from an online simulation in inversion mode.

V. CONCLUSION

This paper presented an evaluation for the use of Vivado HLS as a design approach for the FPGA-based real-time simulation of power electronic systems. This work is the first of its kind to investigate in broad terms the use of Vivado HLS for HIL applications. The area occupation and minimum time-step of the VHLS design obtained during design exploration show that, for small circuits, VHLS produces HS with submicrosecond time-steps and a footprint comparable to what an experienced hardware designer can achieve. The simulation accuracy was also evaluated and was found to be very acceptable with a relative error inferior to 0.01%. However, for larger circuits, VHLS presently fails to fulfill HIL requirements in a cost-effective manner. Nevertheless, the simplicity of using higher level hardware description tools and the good performance of VHLS for the real-time simulation of small power converters suggests a future trend in the HIL practice where HLS will play an important role.

APPENDIX
POWER CIRCUIT MODELING

Network equations are obtained using a fixed time-step discretization. All components are replaced by their companion

circuit, a Norton equivalent resulting from an implicit integration rule in which the current source is a history term. Switches are modeled as small inductor when closed, and small capacitor when open, which is known in the literature as the ADC model of the switch [33]. In this paper, network equations are assembled using the modified-augmented node analysis (MANA) [35] and using backward Euler integration rule, yielding.

$$\mathbf{Ax} = \mathbf{b} \tag{1}$$

where $\mathbf{A}$ is the MANA matrix, $\mathbf{x}$ is a vector containing unknown node voltages and currents entering the independent voltage sources, and $\mathbf{b}$ is a vector composed of known sources and history terms.

By choosing appropriate values for the L_{sw} and C_{sw} associated each switch, such that $C_{\text{sw}}/\Delta t = \Delta t/L_{\text{sw}}$, network equations become fixed irrespective of the switch statuses. The inverse of the invariant $\mathbf{A}$ matrix is then precomputed $\mathbf{A^{-1}}$ and its values used to build a state-space alike computing scheme, as suggested in [36],

$$\begin{bmatrix} \mathbf{v}^{n+1} \\ \mathbf{y}^{n} \end{bmatrix} = \begin{bmatrix} \mathbf{W_{vu}} & \mathbf{W_{vx}} \\ \mathbf{W_{yu}} & \mathbf{W_{yx}} \end{bmatrix} \begin{bmatrix} \mathbf{u}^{n} \\ \mathbf{x}^{n} \end{bmatrix} \tag{2}$$

where $\mathbf{x}^n$ is a subvector of $\mathbf{v}^n$ that is consistent with switch statuses, $\mathbf{u}^n$ is the input vector, $\mathbf{x}^n$ is a state (history term) vector, and $\mathbf{y}^n$ is the output vector.

The switch status of an insulated-gate bipolar transistor–diode pair is updated according to

$$s^{n+1} = c^{n+1} + s^n(i_s^n \leq 0) + \overline{s^n}(v_s^n < 0) \tag{3}$$

where c^{n+1} is the current command at the IGBT gate, s^n, v_s^n, and i_s^n are, respectively, the switch status, the voltage, and the current associated with the switch.

The switch status of a diode is updated according to

$$s^{n+1} = s^n(i_s^n \geq 0) + \overline{s^n}(v_s^n \geq 0). \tag{4}$$

REFERENCES

[1] O. Faruque *et al.*, "Real-time simulation technologies for power systems design, testing, and analysis," *IEEE Power Energy Technol. Syst. J.*, vol. 2, no. 2, pp. 63–73, Jun. 2015.

[2] L. Huang, A. Hu, A. P. Swain, and X. Dai, "Comparison of two high frequency converters for capacitive power transfer," in *Proc. IEEE Energy Convers. Congr. Expo.*, 2014, pp. 5437–5443.

[3] D. Stepins, "An improved control technique of switching-frequency-modulated power factor correctors for low THD and high power factor," *IEEE Trans. Power Electron.*, vol. 31, no. 7, pp. 5201–5214, Jul. 2016.

[4] W. Zhu, K. Zhou, M. Cheng, and F. Peng, "A high-frequency-link single-phase PWM rectifier," *IEEE Trans. Ind. Electron.*, vol. 62, no. 1, pp. 289–298, Jan. 2015.

[5] T. Wang, S. Sen, and M. Amirabadi, "Soft switching high frequency AC-link DC-DC buck-boost converters," in *Proc. IEEE Appl. Power Electron. Conf. Expo.*, 2015, pp. 57–64.

[6] T. Ould-Bachir, H. F. Blanchette, and K. Al-Haddad , "A network tearing technique for FPGA-based real-time simulation of power converters," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3409–3418, Jun. 2015.

[7] G. G. Parma and V. Dinavahi, "Real-time digital hardware simulation of power electronics and drives," *IEEE Trans. Power Del.*, vol. 22, no. 2, pp. 1235–1246, Apr. 2007.

[8] J. Rodriguez-Andina, M. Valdes-Pena, and M. Moure, "Advanced features and industrial applications of FPGAs: A review," *IEEE Trans. Ind. Electron.*, vol. 11, no. 4, pp. 853–864, Aug. 2015.

[9] F. Vahid, *Digital Design With RTL Design, Verilog and VHDL*. Hoboken, NJ, USA: Wiley, 2010.

[10] M. Dagbagi, A. Hemdani, L. Idkhajine, M. W. Naouar, E. Monmasson, and I. Slama-Belkhodja, "ADC-based embedded real-time simulator of a power converter implemented in a low-cost FPGA: Application to a fault-tolerant control of a grid-connected voltage-source rectifier," *IEEE Trans. Ind. Electron.*, vol. 63, no. 2, pp. 1179–1190, Feb. 2016.

[11] W. Meeus, K. Van Beeck, T. Goedemé, J. Meel, and D. Stroobandt, "An overview of today's high-level synthesis tools," *Des. Autom. Embedded Syst.*, vol. 16, no. 3, pp. 31–51, 2012.

[12] J. Cong, B. Liu, S. Neuendorffer, J. Noguera, K. Vissers, and Z. Zhang, "High-level synthesis for FPGAs: From prototyping to deployment," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 30, no. 4, pp. 473–491, Apr. 2011.

[13] S. Sinha and T. Srikanthan, "High-level synthesis: Boosting designer productivity and reducing time to market," *IEEE Potentials*, vol. 34, no. 4, pp. 31–35, Jul./Aug. 2015.

[14] D. Navarro, O. Lucia, L. A. Barragan, I. Urriza, and O. Jimenez, "High-level synthesis for accelerating the FPGA implementation of computationally demanding control algorithms for power converters," *IEEE Trans. Ind. Informat.*, vol. 9, no. 3, pp. 1371–1379, Aug. 2013.

[15] *UG902 v2016.1: Vivado Design Suite User Guide—High Level Synthesis*, Xilinx, San Jose, CA, USA, 2016.

[16] *UG-OCL002 v2015.11.02: Altera SDK for OpenCL—Programming Guide*, Altera, San Jose, CA, USA, 2015.

[17] P. Li, P. Zhang, L. N. Pouchet, and J. Cong, "Resource-aware throughput optimization for high-level synthesis," in *Proc. ACM/SIGDA Int. Symp. Field-Programmable Gate Arrays*, 2015, pp. 200–209.

[18] Ó. Jiménez, Ó. Lucía, I. Urriza, L. A. Barragan, D. Navarro, and V. Dinavahi, "Implementation of an FPGA-based online hardware-in-the-loop emulator using high-level synthesis tools for resonant power converters applied to induction heating appliances," *IEEE Trans. Ind. Electron.*, vol. 62, no. 4, pp. 2206–2214, Apr. 2015.

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

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

[21] D. Majstorovic, I. Celanovic, N. D. Teslic, N. Celanovic, and V. A. Katic, "Ultralow-latency hardware-in-the-loop platform for rapid validation of power electronics designs," *IEEE Trans. Ind. Electron.*, vol. 58, no. 10, pp. 4708–4716, Oct. 2011.

[22] *UG640 v11.4: System Generator for DSP—User Guide*, Xilinx, San Jose, CA, USA, 2009.

[23] L. Herrera, C. Li, X. Yao, and J. Wang, "FPGA-based detailed real-time simulation of power converters and electric machines for EV HIL applications," *IEEE Trans. Ind. Appl.*, vol. 51, no. 2, pp. 1702–1712, Mar./Apr. 2015.

[24] M. Dagbagi, L. Idkhajine, E. Monmasson, and I. Slama-Belkhodja , "FPGA implementation of power electronic converter real-time model," in *Proc. Int. Symp. Power Electron. Electr. Drives, Autom. Motion*, 2012, pp. 658–663.

[25] O. Lucía, I. Urriza, L. A. Barragán, D. Navarro, O. Jiménez, and J. M. Burdío, "Real-time FPGA-based hardware-in-the-loop simulation test bench applied to multiple-output power converters," *IEEE Trans. Ind. Appl.*, vol. 47, no. 2, pp. 853–860, Mar./Apr. 2011.

[26] H. F. Blanchette, T. Ould-Bachir, and J. P. David, "A state-space modeling approach for the FPGA-based real-time simulation of high switching frequency power converters," *IEEE Trans. Ind. Electron.*, vol. 59, no. 12, pp. 4555–4567, Dec. 2012.

[27] R. Velazquez, O. Lucia, D. Navarro, L. Barragan, J. Artigas, and C. Sagues, "Design of an FPGA-based full-state feedback controller using high level synthesis tools," in *Proc. 15th IEEE Workshop Control Model. Power Electron.*, 2014, pp. 1–6.

[28] G. Inggs, S. Fleming, D. Thomas, and W. Luk, "Is high level synthesis ready for business? A computational finance case study," in *Proc. Int. Conf. Field-Programmable Technol.*, 2014, pp. 12–19.

[29] E. Homsirikamol and K. Gaj, "Can high-level synthesis compete against a hand-written code in the cryptographic domain? A case study," in *Proc. 14th Int. Conf. Reconfigurable Comput. FPGAs*, 2014, pp. 1–8.

[30] O. Reiche *et al.*, "Synthesis and optimization of image processing accelerators using domain knowledge," *J. Syst. Archit.*, vol. 61, no. 10, pp. 646–658, 2015.

[31] T. Ould-Bachir and J. David, "Self-alignment schemes for the implementation of addition-related floating-point operators," *ACM Trans. Reconfig. Technol. Syst.*, vol. 6, no. 1, pp. 1–21, 2013.
[32] *UG470 v1.1: 7 Series FPGAs Configuration, User Guide*, Xilinx, San Jose, CA, USA, 2016.
[33] P. Pejovic and D. Maksimovic, "A method for fast time-domain simulation of networks with switches," *IEEE Trans. Power Electron.*, vol. 9, no. 4, pp. 449–456, Jul. 1994.
[34] OPAL-RT, "OP5707 Simulator," 2017. [Online]. Available: www.opal-rt.com/simulator-platform-op5707/
[35] J. Mahseredjian, S. Dennetière, L. Dubé, B. Khodabakhchian, and L. Gérin-Lajoie, "On a new approach for the simulation of transients in power systems," *Elect. Power Syst. Res.*, vol. 77, no. 11, pp. 1514–1520, 2007.
[36] T. Ould-Bachir, C. Dufour, J. Bélanger, J. Mahseredjian, and J.-P. David, "Effective floating-point calculation engines intended for the FPGA-based HIL simulation," in *Proc. Int. Symp. Ind. Electron.*, 2012, pp. 1363–1368.

Federico Montano received the B.Sc. degree in electronics engineering from the Universidad Del Valle, Valle del Cauca, Colombia, in 2000, and the Ms.Ing. degree in electrical engineering from the École Polytechnique de Montréal, Montreal, QC, Canada, in 2013, where he is currently working toward the Ph.D. degree in electrical engineering.

His research interests include FPGA-based real-time simulation, hardware-in-the-loop simulation, and high-level synthesis tools for FPGA development.

Tarek Ould-Bachir (M'08) received the M.A.Sc. and Ph.D. degrees in electrical engineering from the École Polytechnique de Montréal, Montreal, QC, Canada, in 2008 and 2013, respectively.

Since 2014, he has been with OPAL-RT Technologies, Montreal, where he leads the Solvers and the eMEGASIM Teams. His research interests include real-time simulation, power electronics, power systems, and FPGA-based computing.

Jean Pierre David (M'05) received the Ph.D. degree in electrical engineering from the Université Catholique de Louvain, Louvain-la-Neuve, Belgium, in 2002.

He was an Assistant Professor with Université de Montréal, Montréal, QC, Canada, and then with the Polytechnique Montréal, where he has been an Associate Professor since 2013. His research interests include digital system design, reconfigurable computing, high-level synthesis, and their applications.