A VHDL-based Modelling Approach for Rapid Functional Simulation and Verification of Adiabatic Circuits

Sachin Maheshwari, Viv. A. Bartlett, and Izzet Kale, *Member, IEEE*

Abstract— **Adiabatic logic is an energy-efficient technique compared to conventional CMOS design. However, the time required in the design, validation and debugging increases manifold for large-scale adiabatic system designs. In this endeavor, we present a Hardware Description Language (HDL) based modelling approach for dual-rail 4-phase adiabatic logic design. The paper highlights the drawbacks of the existing approaches and proposes a new approach that captures the timing errors and detects the circuit's invalid operation due to mutually exclusive inputs being violated. We develop a model library containing the function of the four periods used in the trapezoidal power-clock and the adiabatic logic gates. The validation and verification of the approach was done on the ISO-14443 standard benchmark circuit, a 16-bit Cyclic Redundancy Check circuit. The system modelled using HDL shows the timing agreement with the transistor-level SPICE simulations. The novel use of the four periods of a power-clock improves the robustness and reliability of the proposed modelling approach for the design and verification of large adiabatic circuits and systems.**

Index Terms—**adiabatic, modelling, power-clock, timing verification, VHDL**

I. Introduction And Motivation

Adiabatic circuits were introduced to reduce the dynamic energy consumption, CV^2 that is dissipated in conventional CMOS. Over the past 25 years, many energy-efficient fully adiabatic or quasi-adiabatic logic families have been proposed as an alternative low-power circuit technique where speed is of secondary concern [1] - [5]. They all are based on the same adiabatic principle, but the structures and complexity, such as 1) the power-clocking scheme; 2) evaluation network connection; 3) adiabatic losses and non-adiabatic losses; 4) their transistor counts, differ from each other.

Nevertheless, the verification of the functionality and the low energy traits of adiabatic logic in comparison to the non-adiabatic logic is conventionally performed using transistor-level SPICE simulations. But designing a large complex adiabatic system increases the design time and its validation. Additionally, due to the complexity of synchronizing the power-clock phases [6], the debugging of errors become difficult and time-consuming. This gives rise to a need for a versatile modelling approach that can be used, across the technology, to describe the adiabatic logic behaviour at a higher level of abstraction before SPICE simulations are performed for close to realistic energy measurements. Such a model would allow the functional errors to be detected and corrected early in the system design process, decreasing the overall simulation and verification time.

To the best knowledge of the authors and the literature review undertaken, the first HDL modelling of adiabatic logic was done by M. Vollmer and J. Gotze in 2005 [7]. They described a CORDIC systolic array with precise timing using VHDL but did not model the dual-rail encoding of input and output signals and used only one global power-clock. A year later, Laszlo Varga et.al. [8] described two-level pipelining and scheduling of adiabatic logic. This approach mainly focussed on producing a pipeline schedule of the power-clock behaviour of the adiabatic logic only for a single-rail scheme. In 2010, David John Willingham in his PhD thesis [9] reported *Asynchrobatic Logic* modelling in Verilog. The author first demonstrated the idea on a single-rail scheme and then extended it to dual-rail. Though the dual-rail implementation proved to be advantageous in detecting invalid circuit operations, this author like the others did not model the power-clock in HDL and instead used a square waveform. Though they have all successfully demonstrated the behavioural aspects of the adiabatic logic circuits using HDL, none have calibrated their approach in the presence of invalid input cases, i.e. violation of the adiabatic principle.

A. Contributions of this paper

The work reported in this paper builds on the work done in the author's previous publications [10], [11]. To give a clear objective and the focus, the authors in this paper highlights the differences between their earlier publications as follows;

1) The authors demonstrate the errors associated with using a square waveform as a power-clock and calibrated and compared this with the proposed approach for the chain of cascade NOT/BUF gates.

2) The adiabatic primitives 'Aand' and 'Aor' logic gate truth tables were modelled and validated through the approach.

3) The NOT/BUF VHDL code was further enhanced by removing an unnecessary elsif condition and appending more validation steps for checking invalid inputs in each of the four power-clock periods.

4) The proposed approach has been tested for larger fan-in gates (10-input XOR) and compatibility with Bennett clocking [12].

5) The reliability, correctness and robustness of the proposed modelling approach were verified for the ISO 14443 benchmark circuit, 16-bit Cyclic Redundancy Check (CRC) [13].

B. Structure of the paper

Section II of this paper presents the proposed approach deploying adiabatic logic. This section demonstrates the encoding of power-clock and dual-rail signals, gate-level modelling for 'Aand' and 'Aor' logic gates, encoding of invalid inputs and compatibility with Bennett clocking. Section III shows how the functionality is affected in the existing approaches by introducing intentional timing violations. The simulation result for the 16-bit CRC circuit using 4-phase adiabatic logic is presented in section IV. The paper is concluded in section V.

II. Digital Simulation Approach for Adiabatic Logic

VHDL is used to model the 4-phase adiabatic logic [13] to capture the circuit description. One of the advantages of the proposed

Manucript revision submitted on June 28, 2020. This work was supported in part by the University of Westminster under the Cavendish Research Scholarship.

Sachin Maheshwari is currently a Research Fellow at the Centre for Electronics Frontiers in Zepler Institute, University of Southampton, SO17 1BJ, UK (e-mail: s.maheshwari@soton.ac.uk).

Viv A. Bartlett & Izzet Kale both are with the School of Computer Science and Engineering, University of Westminster, London, W1W6UW, UK (e-mail: {v.bartlett,kalei}@westminster.ac.uk).

approach is that the entire system design can be rapidly simulated with a logic simulator and can be interfaced and mixed with the non-adiabatic logic designs for effective design solutions.

A. Trapezoidal waveform and dual-rail inputs using VHDL

To realize the trapezoidal power-clock in standard logic, a multi-level approach is proposed as depicted in Fig. 1. In the proposed approach, the Hold (H) and the Idle (I) periods of the power-clock are represented as a logic ‘1’ and logic ‘0’ respectively, whereas, the Evaluation (E) and the Recovery (R) period are encoded with an intermediate state marked as ‘X’, for the duration of the ramp period.

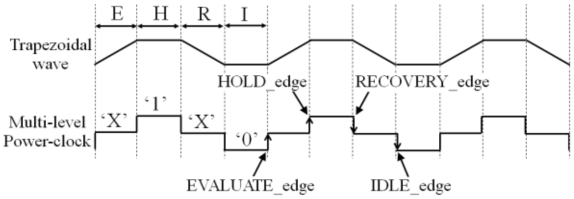


Fig. 1. Proposed multi-level encoding of the trapezoidal power-clock.

The encoding of the four power-clock periods in standard logic requires four states, where each state is encoded based on the logic levels required. The four states are generated using two standard D flip-flops counting from “00” to “11”. Also, the four periods of the power-clock are defined as an edge function which is aggregated into a package named ‘Adiabatic_signal’. As an example, Fig. 2 shows the function defining the EVALUATE_edge. The HOLD_edge is defined as a transition from ‘X’ to ‘1’ state, RECOVERY_edge from ‘1’ to ‘X’ state and finally IDLE_edge from ‘X’ to ‘0’ state. Here the signal type ‘std_ulogic’ is used as the proposed logic uses ‘X’ for the intermediate state. The package is shared in the NOT/BUF adiabatic logic VHDL model which is used to develop the cell library of the basic 2-input, 3-input and 4-input adiabatic logic gates.

```
Function EVALUATE_edge (Signal s : std_ulogic)
Return Boolean is
  Begin
     Return (s'event AND (To_X01(s) = 'X') AND
     (To_X01(s'last_value)= '0'));
End Function;
```

Fig. 2. A user-defined function declaration of EVALUATE_edge.

After the power-clock is modelled in VHDL, the next step is to generate the adiabatic inputs using the multi-level approach. Like the modelling of the power-clock, the pulse input to the adiabatic conversion also requires four states. For simplicity, we forced the D flip-flop outputs externally using the clock signal ‘CLK’ as a two-bit counter generating four states as depicted in Fig.3.

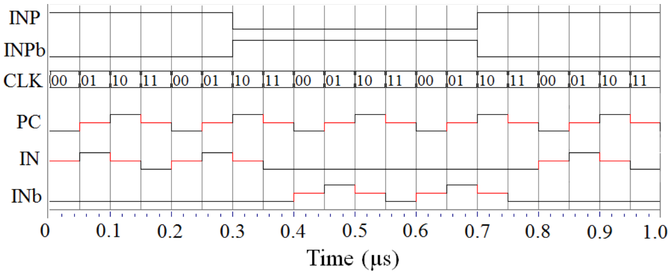


Fig. 3. Pulse input to multi-level adiabatic signals. Generation of mutually exclusive adiabatic input signals (IN, INb) and the power-cock (PC).

B. Gate-level Modelling

To model the adiabatic logic gates, the HDL primitives are compared to an equivalent adiabatic gate based on the multi-level encoding approach. Fig. 4 (a) shows the truth table of the primitive AND and OR logic gates. Since our approach represents ‘z’ and ‘x’ as an invalid and intermediate state respectively, the primitive gates are modelled as shown in Fig. 4 (b). In Fig. 4 (a), the outputs in red/italic indicate the ones that are modified for the adiabatic logic. In Fig. 4 (b), the operation involving either of ‘x’ and ‘z’ with ‘1’ and ‘z’ produces an invalid output ‘z’. Also, the operation involving ‘z’ with ‘0’ produces an invalid output marked with ‘z’. The tables in Fig. 4 (b) are used to write a user-defined primitive for AND and OR as a function in VHDL. The functions utilize the case statement control structure and are named ‘Aand’ and ‘Aor’ in the Adiabatic_GATES package body.

AND	**0**	**1**	**x**	**z**
0	0	0	0	0
1	0	1	*x*	*x*
x	0	*x*	x	*x*
z	0	*x*	*x*	*x*

OR	**0**	**1**	**x**	**z**
0	0	1	x	*x*
1	1	1	*1*	*1*
x	x	*1*	x	*x*
z	*x*	*1*	*x*	*x*

(a)

Aand	**0**	**1**	**x**	**z**
0	0	0	0	z
1	0	1	z	z
x	0	z	x	z
z	z	z	z	z

Aor	**0**	**1**	**x**	**z**
0	0	1	x	z
1	1	1	z	z
x	x	z	x	z
z	z	z	z	z

(b)

Fig. 4. Basic logic AND and OR gate truth-table (a) primitive (b) adiabatic modelled. The outputs in red indicate the amended for adiabatic logic.

A fragment of the VHDL description of the NOT/BUF adiabatic gate is shown in Fig. 5. The code shows the behaviour of the four periods explicitly. Under the evaluation period, the only valid condition is when PC is in state ‘X’ and input is transiting from state ‘X’ to ‘1’, (i.e. HOLD_edge) and the rest are invalid conditions. During the hold period, the only valid condition is when PC is in state ‘1’ and the input is transiting from state ‘1’ to ‘X’, (i.e. RECOVERY_edge) and the rest are invalid conditions, similarly for the recovery period. Apart from checking the invalid input condition in each of the four periods, an invalid state is also checked for in cascade designs. Here the PC and the inputs are 90^o out of phase and hence we use an ‘edge’ sensitive check for the input and a ‘level’ sensitive check for the PC in the HDL model. The SPICE and VHDL output waveforms for the 4-phase PFAL [2] NOT/BUF gate are shown in Fig. 6 (a) and (b) respectively. The VHDL simulation clearly shows a timing agreement of the proposed approach with that of the transistor-level SPICE simulation.

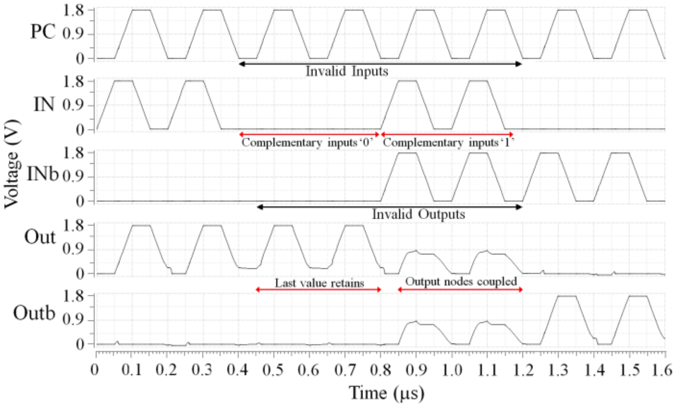
C. Modelling invalid inputs

The operation of the adiabatic logic gates, although conceptually simple, can be somewhat complex to model accurately. This is due to the two cross-coupled inverters forming a latch [13], which retains the last value stored on the two output nodes. For example: if the mutually exclusive adiabatic inputs are both at logic ‘0’ (indicating an invalid state), the adiabatic outputs will retain the last value stored. This, invalid input condition in a large-scale circuits and systems will be difficult to debug, especially in the case when functionally, logic ‘1’and ‘0’ is expected on the two output nodes. On the other hand, if the inputs are at logic ‘1’ (again an invalid state), the output nodes will try to charge via the input transistors, and at the same time, the cross-coupled nMOS transistors will discharge it to the ground. Therefore, the output nodes get capacitively coupled and will settle at some intermediate value. The above two invalid conditions can be seen in the SPICE simulation of Fig. 6 (a).

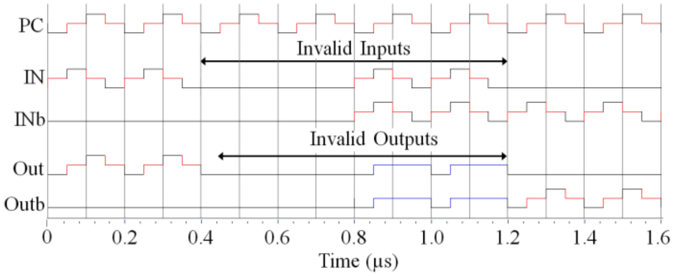
```
   Process (PC, IN, INb) is
     Begin
                          // Idle Period //
1    if PC=’0’ then
2          Out <= PC;     Outb<= PC;
                        // Evaluation Period //
3      elsif PC=’X’ and HOLD_edge (IN) and
       HOLD_edge (INb) then
4          Out<= ‘Z’;     Outb<=’Z’;
5      elsif PC=’X’ and HOLD_edge (IN) then
6          Out <= PC;     Outb<='0';
7      elsif PC=’X’ and HOLD_edge (INb) then
8          Out <= '0';    Outb<=PC;
9      elsif PC='X' and RECOVERY_edge (IN) then
10         Out<= 'Z';     Outb<='Z';
11     elsif PC='X' and RECOVERY_edge (INb) then
12         Out<= 'Z';     Outb<='Z';
                        // Hold Period //
13     elsif PC=’1’and RECOVERY_edge (IN) and
       RECOVERY_edge (INb) then
14         Out <= ‘Z’;    Outb<=’Z’;
15     elsif PC=’1’ and RECOVERY_edge (IN) then
16         Out <= PC;     Outb<='0';
17     elsif PC=’1’ and RECOVERY_edge (INb) then
18         Out <= '0';    Outb<=PC;
19     elsif PC=’1’ and IDLE_edge (IN) then
20         Out <= ‘Z’;    Outb<='Z';
21     elsif PC=’1’ and IDLE_edge (INb) then
22         Out <='Z';     Outb<=‘Z’;
                        // Recovery Period //
23     elsif PC=’X’ and IDLE_edge (IN) and
       IDLE_edge (INb) then
24         Out <= ‘Z’;    Outb<= ‘Z’;
25     elsif PC=’X’ and IDLE_edge (IN) then
26         Out <= PC;     Outb<='0';
27     elsif PC=’X’ and IDLE_edge (INb) then
28         Out <= '0';    Outb<=PC;
29     elsif PC='X' and EVALUATE_edge (IN) then
30         Out<= 'Z';     Outb<='Z';
31     elsif PC='X' and EVALUATE_edge (INb) then
32         Out<= 'Z';     Outb<='Z';
                        // Invalid State //
33     elsif IN = 'Z' and INb = 'Z' then
34         Out<= 'Z';     Outb<='Z'
     End if;
   End Process;
```

Fig. 5. VHDL description of the NOT/BUF adiabatic gate.

The above invalid conditions modelled using our proposed approach are shown in Fig. 6 (b). When the mutually exclusive inputs are at logic ‘1’, the output nodes will be capacitively coupled to an invalid state denoted by ‘z’, and when at logic ‘0’ the output nodes remain at logic ‘0’. As a result, our approach not only models the invalid dual-rail inputs but also helps in identifying the invalid inputs.



(a)



(b)

Fig. 6. Simulation of PFAL NOT/BUF gate for invalid conditions (a) SPICE (b) VHDL model differentiating between logic ‘0’ and logic ‘1’.

D. *Scalability to higher fan-in gates*

The proposed approach also has the capability to model logic gates that scale to large fan-in. Having large fan-in gates allows a significant reduction in latency, energy and area, however not all adiabatic logic families exhibit the same benefit [6]. The author in [6] (section 6.2) has demonstrated the overhead reduction by using large fan-in complex gates at transistor level simulations. Since our approach is applicable for all 4-phase adiabatic logic families where the complexity of designing adiabatic systems increases with an increase in the system size. Here the authors are more concerned about functional and timing verification. hence, to demonstrate the validity of our proposed approach scaling to large fan-in gates whilst maintaining the same latency with that of the SPICE simulations, a 10-input XOR logic gate was constructed and simulated. The complementary inputs are not shown in Fig. 7 but are coded and represented as Fig. 3. The simulation result shows that the proposed approach can be easily used for large fan-in gates.

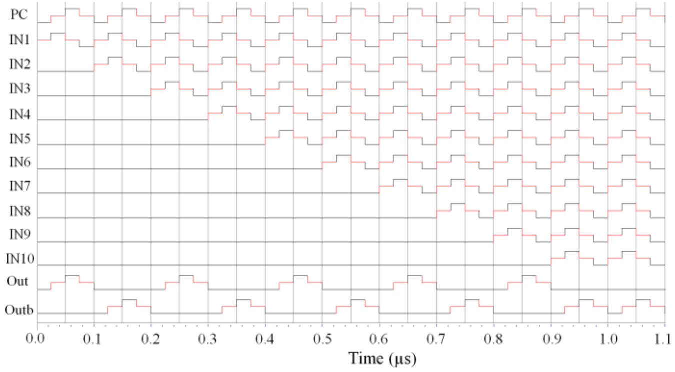


Fig. 7. Simulated waveforms for the10-input XOR/XNOR gates

E. *Bennett Clocking Compatibility and Other Effects*

The versatility of our approach is the compatibility with a different clocking strategy known as Bennett clocking [12] unlike our 4-phase clocking scheme. Here the evaluation and recovery period exist only when the inputs are at the same logic level. The results shown in Fig. 8 corresponds to the result of the first three stages of the 4-stage cascade buffer chain depicted in Fig. 9 (a). The complementary input ‘INb’ is at logic ‘0’, producing the complementary outputs (Q01b - Q03b) as logic ‘0’, hence they have been omitted in Fig. 8. The encoding of the logic states in HDL is like the trapezoidal PC shown in Fig. 1. However, due to the variable hold and idle periods, here, the Power-Clocks (PCs) and the adiabatic input are generated using a BCD counter. The VHDL code in Fig. 5 was amended with the elseif conditions in all the 3 periods (evaluation, hold, recovery) on lines 5, 7, 15, 17, 25 and 27. For example, in the evaluation period, when PC=X and IN=’1’ one of the outputs will follow PC: and the other will be logic ‘0’ and vice-versa for the complementary inputs, IN=’0’

& INb='1'. The rest of the user-defined adiabatic signals as well as the adiabatic 'Aand' and 'Aor' gates remain unchanged.

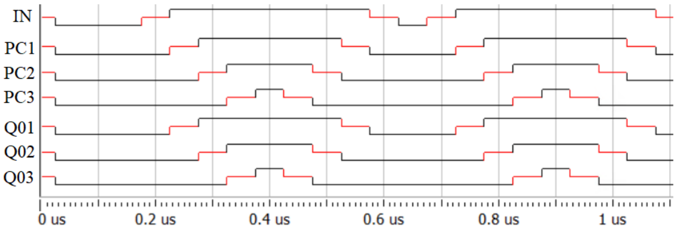


Fig. 8. Bennett clocking waveform with 3-stage cascade buffer chain outputs

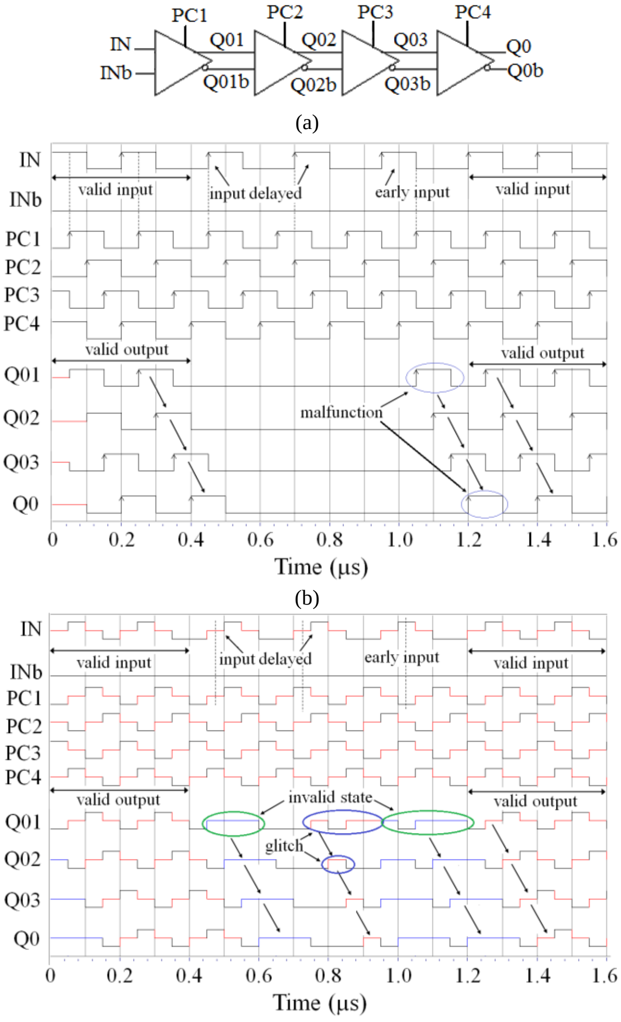
The proposed modelling is only for functional and timing verification. To provide a solution related to issues such as the floating node problem, transistor count, sensitivity to process variations and low voltage operation, an adiabatic circuit designer is tied into undertaking low-level circuit simulations in SPICE. In the same line of thinking the leakage power is another concern for deep sub-micron technologies. Moreover, the Non-Adiabatic Losses (NAL) arising due to the threshold voltage degradation causes energy to increase and is different for different adiabatic logic families [6], [13]. At the functional level, all the above effects will not alter the operation of the circuit unless wrong inputs/connections are provided.

III. Error in Encoding of Existing Modelling Approach

The most difficult part in modelling adiabatic logic using the conventional HDLs is that these languages are made entirely for encoding two logic levels ('0' and '1') and are either 'level' or 'edge' sensitive. The existing modelling approach uses voltage-level encoding for adiabatic logic, similar to the non-adiabatic logic designs. Here, the logic '1' corresponds to the hold period and logic '0' corresponds to the idle period. The remaining two periods, evaluation and recovery (which are both ramps) one changing from logic '0' to '1' and the other from logic '1' to '0' respectively, have been merged with the hold and idle periods respectively.

Thus, to calibrate our proposed approach, in case, if either the input or the power-clock arrives early or gets delayed the two output nodes should discharge to ground, identifying an invalid input that has occurred and the modelling follows the adiabatic principle. Fig 9 (a) shows the 4-stage cascade NOT/BUF chain designed using PFAL. As the complementary input 'INb' is at logic '0', all the complementary outputs will be at logic '0', hence they have been omitted in Fig. 9 (b) and (c). The gate working in phase 1 of PC (PC1) produces the first stage output denoted as 'Q01' and 'Q01b'. The fourth stage works in phase 4 of PC (PC4) produces the final stage outputs denoted by 'Q0' and 'Q0b'. It can be seen in Fig. 9 (b) that for the delayed input condition, the outputs follow the adiabatic principle by generating logic '0' for the existing approach. However, when the input arrives early, the output follows the PC, thus violating the adiabatic principle. Therefore, in the existing approach, a timing window exists between the input and the PC for the correct circuit and timing operation. The same condition can occur if the PC is either delayed or arrives early.

It can be seen from Fig. 9 (c) that the proposed approach will fail if the wrong input signal or the PC (delayed or arrived early) is supplied. This gate generation failure will be similar to that of the SPICE simulation. The proposed approach is much more accurate, however, it generates a glitch for the delayed input condition, which reduces as it is passed through a cascade gate. The glitch arises due to the signal 'X' being used for encoding both the ramps (evaluation and recovery period). It can however be removed if two different signals such as 'U' and 'X' are used for encoding the two ramps. However, this glitch is insufficient to cause any functionality and timing error which the existing logic exhibits.



(a)

(b)

(c)

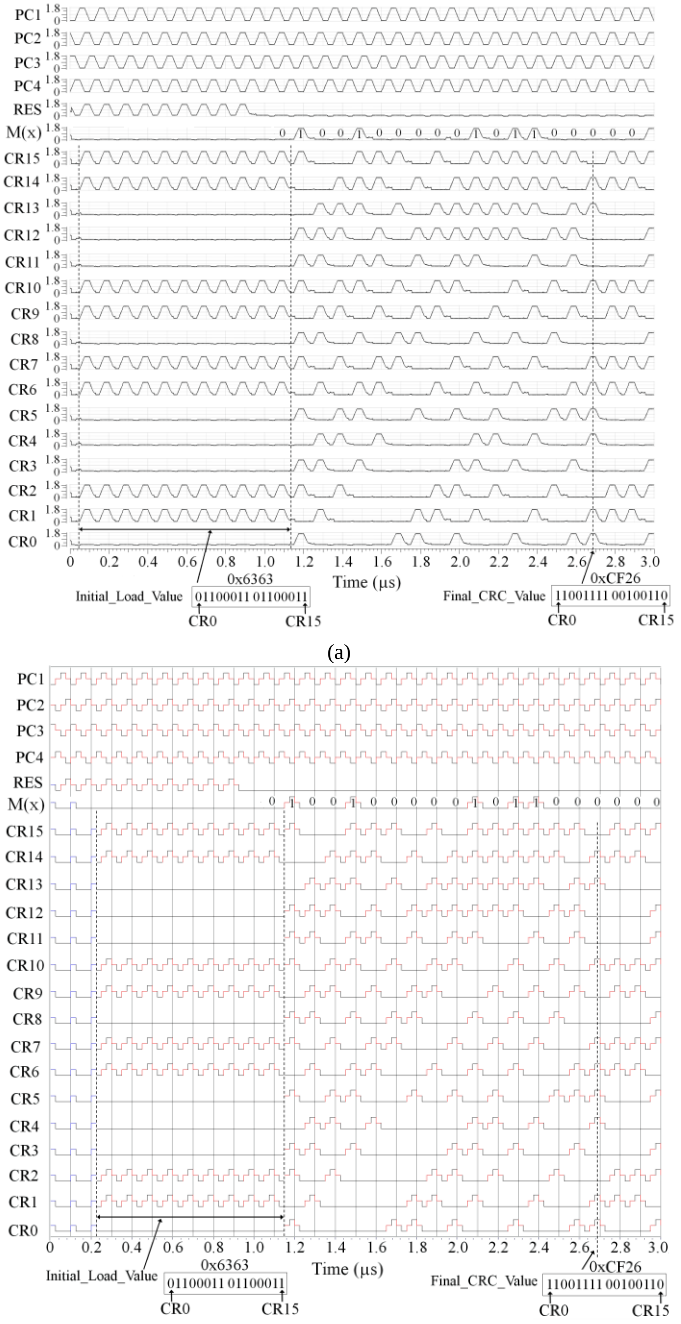
Fig. 9. (a) Schematic of the 4-satge cascade buffer chain. (b) Simulated waveforms of input timing variations for the existing approach using square-waveform. (c) Simulated waveform using the proposed approach.

IV. Simulation Results

The 4-phase adiabatic logic family used for the SPICE simulation is PFAL [2]. SPICE simulations were performed with the high-performance Spectre XPS MS simulator using the Cadence EDA tool in a 'typical-typical', TT process corner for a TSMC 180nm CMOS process at 1.8V power supply.

For all the other adiabatic gates such as AND/NAND, OR/NOR, XOR/XNOR and MUX/DeMUX the VHDL behaviour is described by combining the functional part and the adiabatic NOT/BUF gate for timing validation. The collection of all the logic gates described in VHDL formed the cell library. Using our home-grown cell library, the structural model of a 16-bit CRC circuit for a 16-bit message word was successfully verified. The block diagram and the working of the 16-bit CRC circuit are given in [13]. The time-period of the PC was taken as 100ns. The simulation setup for the SPICE analysis is similar to that of the VHDL so that the uniformity and comparability are maintained across both the simulations. The CRC is initialized using the reset input 'RES' which resets the counter to the "0000" state and load the pre-set value of "0x6363" to the CRC datapath. The load value is per the NFC application compliant with the

ISO/IEC-14443 standard for contactless smart cards [14]. The 16-bit message, M(x) is sent serially. When the 'RES' signal is set false (logic '0'), the CRC starts the computation.



(a)

(b)

Fig. 10. Simulation results for 16-bit CRC for 16-bit message length (a) SPICE (b) proposed VHDL approach.

Fig. 10 (a) and (b) show the SPICE and the VHDL simulation waveform respectively. The SPICE simulation takes 117% longer than the VHDL ModelSim simulator. Also, the VHDL results show the precise timing model when compared to the SPICE results. However, the VHDL implementation shows a larger delay at the start of the simulation compared to the SPICE. This is because the pulse inputs are converted to the adiabatic inputs, whereas, in the transistor level design the inputs are given based on the requirement of the PC input phase. If the inputs in the transistor level design are processed similar to the VHDL design, then both the simulations will have the same initial delay, however, this was deemed unnecessary as it would not add to our findings, as this is a mere setup latency.

V. Conclusion

To overcome the problem of functional validation and error detection arising due to the complexity of the 4-phase power-clocking scheme in large-scale adiabatic circuits and systems, a new modelling approach using VHDL has been proposed. The versatility of the proposed approach is the compatibility with the Bennett clocking and its applicability to the single-phase and 2-phase adiabatic logic families with the prerequisite of more states required to encrypt the variable idle and hold periods. The simulation results for the chain of buffer circuits demonstrate that the proposed approach works correctly at the functional and timing levels and obeys the adiabatic principle. With the simulation results of the ISO 14443 benchmark circuit, 16-bit CRC, the proposed approach exhibits precise timing and validates the functional performance with the SPICE simulations. Thus, our proposed modelling approach shows the possibility of an efficient design approach for painless and accurate functional and timing characterization of a high-end complex adiabatic system.

References

[1] Saed G. Younis, "Asymptotically Zero Energy Computing Using Split-Level Charge Recovery Logic," PhD. Dissertation, MIT, USA, 1994.

[2] A. Vetuli, S. D. Pascoli and L. M. Reyneri, "Positive feedback in adiabatic logic", *Elec. Lett*, vol. 32, no. 20, pp. 1867-1869, 1996.

[3] Y. Moon, D.K. Jeong, "An efficient charge recovery logic circuit", *IEEE J. Solid-State Circuits*, vol. 31, no. 4, pp. 514-522, 1995.

[4] W. C. Athas, L. J. Svesson, J. G. Koller, N. Traztzanis and E. Y.-C. Chuo,"Low power digital system based on adiabatic-switching principles," *IEEE Trans. VLSI Sys.*, vol. 2, no. 4, pp. 398–406, 1994.

[5] D. Maksimovic, V. G. Oklobdzija, B. Nikolic and K. W. Current, "Clocked CMOS adiabatic logic with integrated single-phase power-clock supply," *IEEE Trans. VLSI Sys.*, vol. 8, no. 4, pp. 460-463, 2000.

[6] P. Teichmann, "Adiabatic logic: future trend and system level perspective," vol. 34. Springer Science & Business Media, pp. 65-81, 128-136, 2011.

[7] M. Vollmer, and J. Götze, "An adiabatic architecture for linear signal processing", *Adv. in Radio Sci.*, pp. 325–329, 2005.

[8] L. Varga, G. Hosszu and F. Kovacs, "Two-level Pipeline Scheduling of Adiabatic Logic," *29th Int. Spring Semi. Electro. Technol.*, pp. 390-394, 2006.

[9] David John WILLINGHAM, "Asynchrobatic logic for low-power VLSI design," PhD. dissertation, School of Computer Science & Engineering, University of Westminster, London, UK, 2010.

[10] Sachin Maheshwari, V. A. Bartlett and I. Kale, "VHDL-Based Modelling Approach for the Digital Simulation of 4-Phase Adiabatic Logic Design," *28th Int. Sym. on PATMOS*, pp. 111-117, 2018.

[11] Sachin Maheshwari, V. A. Bartlett, Izzet Kale "Modelling, Simulation and Verification of 4-phase Adiabatic Logic Design: A VHDL-Based Approach" *Integra., the VLSI J., Elsevier*, vol. 67, pp. 144-154, 2019.

[12] C.H. Bennett, "Logical reversibility of computation," *IBM J. Res. Dev*, vol. 17, no. 6, pp. 525–532, 1973.

[13] Sachin Maheshwari, V. A. Bartlett and Izzet Kale "Energy efficient implementation of multi-phase quasi-adiabatic Cyclic Redundancy Check in near field communication", *Integra., the VLSI J., Elsevier*, vol. 62, pp. 341-352, 2018.

[14] Identification cards - Contactless integrated circuit(s) cards – Proximity cards – Part 3: Initialization and anticollision, ISO/IEC Std. FCD 14443–3, 2001. [Online].