International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/CSEIT206620

Development of Jump Markovian Dynamic Models of Low Cost Digital Duty-Cycle Modulation Drivers For Target System-on-Chip Devices

Jean Mbihi*, Léandre Nneme Nneme
Research Laboratory of Computer Science Engineering and Automation, University of Douala/ENSET, Douala, Cameroon

Article Info
Volume 6, Issue 6
Page Number: 364-372
Publication Issue :
November-December-2020

Article History
Accepted : 12 Dec 2020
Published : 30 Dec 2020

ABSTRACT

In this paper, a novel building *algorithmic scheme* of DDCM (*duty-cycle modulation*) *drivers* is presented. It is modelled in the analog domain as a continuous time *jump Markovian dynamic model*, with a deterministic two-state Markov chain. An equivalent discrete jump dynamic model is computed using *pole-zero matching transform*. Then, the resulting digital iterative algorithm, consists of simple digital operators and structures. The proposed DDCM algorithm is simulated under Matlab framework, and implemented using Arduino IDE-C++ with uploading into an ESP32 system-on-chip (SoC) device. The monitoring device connected to the ESP32 via an USB communication cable is an Arduino/IDE virtual monitor. It is configured for 230400 bauds communication. A low cost *ESP32-based DAC* (digital-to-analog converter), is virtually implemented and well tested as a case study of the proposed new generation of DDCM drivers. Matlab digital simulation results and ESP32 processing and virtual monitoring results are presented and discussed, in order to show the realistic nature and the great challenge the proposed DDCM algorithmic scheme for SOC devices.

Keywords: Algorithmic scheme, DDCM (Digital duty-cycle modulation) drivers, Markovian dynamic model, pole-zero matching transform, ESP32-based DAC.

I. INTRODUCTION

Most switching modulation techniques used in instrumentation and industrial electronics, have been developed and improved using analog electronic modules, e.g., PWM (Pulse Width Modulation [1]), SDM (Sigma-delta-modulation [2])), and DCM (Duty-cycle modulation [3]). However, they are widely implemented nowadays, on DSP (digital signal processing) devices, with the hope to achieve better performance, easier extension and more flexible distribution. Therefore, given fixed technical specifications of an analog modulation circuit, the general problem is to synthesize an equivalent digital

modulation scheme, to be implemented into DSP devices. As an implication, ADC (digital-to-analog) and DAC (digital-to-analog) converters, built from a high frequency switching modulating interface are known as oversampling converters. It is worth noting that, the equivalence between analog and digital modulation is mainly restricted to corresponding input-output waveforms. However, in the analog domain, the output wave $x_m(t)$ of most switching digital modulation schemes with input x as modulating input, is illustrated in Fig. 1.

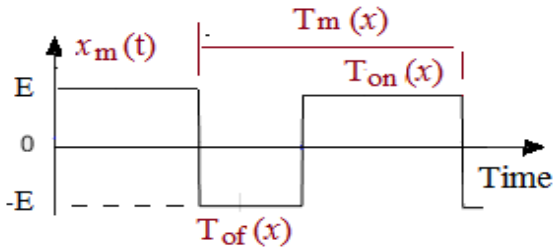


Figure. 1. Output waveform $x_m(t)$, of most switching modulation schemes

In PWM schemes, the modulation period $T_m(x) = T$ is a constant, in which case the input x propagates through $T_{on}(x)$ (i.e., width of the high pulse). In SDM structure, both Ton (x) and T_{of} (x) also vary as a function of input x, according to a heuristic law to be determined. In DCM schemes, however, both T_{on} (x) and T_{of} (x) (i.e., widths of High and Low pulses respectively), simultaneously vary as known functions of x. As an implication, the modulating input x is encapsulated into the modulation duty-cycle $R_m(x) = T_{on}/(x)/T_m(x)$ with known analytical model, where $T_m(x) = T_{on}(x) + T_{of}(x)$ stands for the duty-cycle modulation period.

In application areas using PWM-based, SDM-based and DCM-based ADCs (analog-to-digital converters), analog switching interfacing circuits are frequently used to reduce the overall size and cost of a whole ADC. However, for DACs (digital-to-analog converters), existing topologies such as DPWM (Digital PWM) DACs [4], DSDM (Digital SDM) DACs [5] and DDCM (Digital DCM) DACs [6], require upstream m-bits counting processes for $T_{on}(x)$ and T_m (x) quantities, followed by chains of greedy sequential logic circuitries, including interrupt service routine (ISR).

From the preceding literature analysis, it is worth noting that, existing oversampling modulation topologies, remains architecturally complex and notoriously difficult to build. Indeed, they involve pulse counting processes, interrupt service routine (ISR), and chains of sequential and combinatory logic devices Even though the pioneering DDCM-based DAC algorithm developed in [6] and well tested on a FPGA- target, remains less costly and more efficient than existing solutions, its greedy overall complexity, remains unrealistic for possible implemented into standard low cost SOC (system-on-chip) devices, found in today market. The relevant goal of this paper is to overcome that relevant weakness, by a new synthesis approach of low cost DDCM schemes.

The remaining sections of this paper are organized as follows: The research methodology and tools related to the development of a new algorithmic scheme for DDCM drivers is outlined in Section II with application to a virtual ESP32-based DDCM-DAC. Then, the experimental setup and relevant results obtained are presented in Section III, followed in Section IV by the conclusion of the paper.

II. RESEARCH METHODOLOGY AND TOOLS

A. Similarity Principles Between Analog and DDCM Schemes

For the sake of easy understanding, three similarity principles will be outlined later, from both analog and numerical analysis of a simple DCM circuit shown in Fig. 2. That elementary circuit was originally studied in [3] for electronic instrumentation applications, and has increasingly

become a versatile inspiration source for the development of numerous types of signal processing devices, including DCM-based ADC ([7], [8]), DCM and DDCM drivers for power electronic systems [9], and DDCM-based DAC [6].

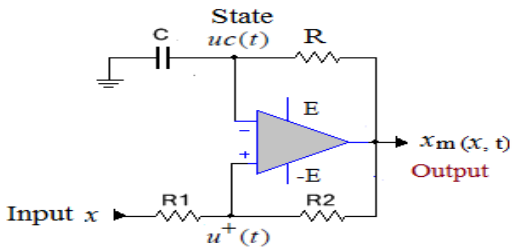


Figure 2. Analog DCM circuit studied in [3]

It is worth noting that the dynamic behaviour of the DCM circuit depicted on Fig. 2, is given in continuous time as in [10], by the set of equations (1) and (2).

$$\begin{cases} u^{+}(t) = a_1 x_m(t) + (1-a_1)x(t) & \text{(a)} \\ \varepsilon(t) = u^{+}(t) - u_C(t) & \text{(b)} \\ x_m(t) = \begin{cases} E, & \text{if } \varepsilon(t) \geq 0 \\ -E, & \text{otherwise} \end{cases} & \text{(c)} \\ \dfrac{duc(t)}{dt} = -\dfrac{1}{\tau} uc(t) + \dfrac{1}{\tau} x_m(t) & \text{(d)} \end{cases} \quad (1)$$

with,

$$\begin{cases} a_1 = \dfrac{R1}{R1+R2} \\ \tau = R\,C \end{cases} \quad (2)$$

In addition, Equation (1) structurally falls into the class of *continuous time and deterministic jump Markovian dynamic models*, with state uc(t) and *jump Markov chain* $\{x_m(t) = \{E, -E\}$. It differs from stochastic a jump Markovian dynamic process, since in this case, both Markov chain and dynamic process are deterministic phenomena. Indeed, given arbitrary initial conditions uc(0), x(0) and $x_m(0)$, the next switching time and state of the Markov chain $\{x_m(t), t\}$, as well as the further trajectory of uc(t), can be analytically predicted.

Furthermore, equations (1a) (1b) and (1c) involve two predictable switching upper and lower thresholds $uc_M(t)$ and $uc_m(t)$ respectively, given as follows:

$$\begin{cases} u_{CM}(t) = a_1 E + (1-a_1)x(t) & \text{(a)} \\ u_{Cm}(t) = -a_1 E + (1-a_1)x(t) & \text{(b)} \\ \text{with } u_{Cm}(t) \leq u_C(t) \leq u_{CM}(t) & \text{(c)} \end{cases} \quad (3)$$

As a relevant advantage, solving (1), given (2), (3) and admissible initial conditions u(0), x(0) and $x_m(0)$, just requires a few analog operators, e.g., +, -, *, /, = =, ≥, ≤, d/dt, if, etc. On the other hand, reproducing the dynamic behaviour of (1) within a DSP device, requires a suitable transform (with sampling period T), into a lossless discrete dynamic model. Obviously, the application to (1) of the *pole-zero matching discretization technique*, leads to a *discrete jump Markovian dynamic model* described by similar equations (4), (5) and (6), where the *k* index stands for discrete time k T, with k = 0, 1, …, N (total number of sample), T being a suitable sampling period, witch would be chosen according to Nyquist's sampling theorem, applied to the DDCM frequency spectrum.

$$\begin{cases} u^{+}(k) = a_1 x_m(k) + (1-a_1)x(k) & \text{(a)} \\ \varepsilon(k) = u^{+}(k) - u_C(k) & \text{(b)} \\ x_m(k) = \begin{cases} E, & \text{if } \varepsilon(k) \geq 0 \\ -E, & \text{otherwise} \end{cases} & \text{(c)} \\ u_c((k+1)) = a\ u_c(k) + (1\text{-}a)\ x_m(k) & \text{(d)} \end{cases} \quad (4)$$

$$k \equiv k\,T,\ a = e^{-T/\tau} = e^{-T/(Rs\,Cs)} \quad (5)$$

$$\begin{cases} u_{CM}(k) = a_1 E + (1-a_1)x(k) & \text{(a)} \\ u_{Cm}(k) = -a_1 E + (1-a_1)x(k) & \text{(b)} \\ \text{with } u_{Cm}(k) \leq u_C(k) \leq u_{CM}(k) & \text{(c)} \end{cases} \quad (6)$$

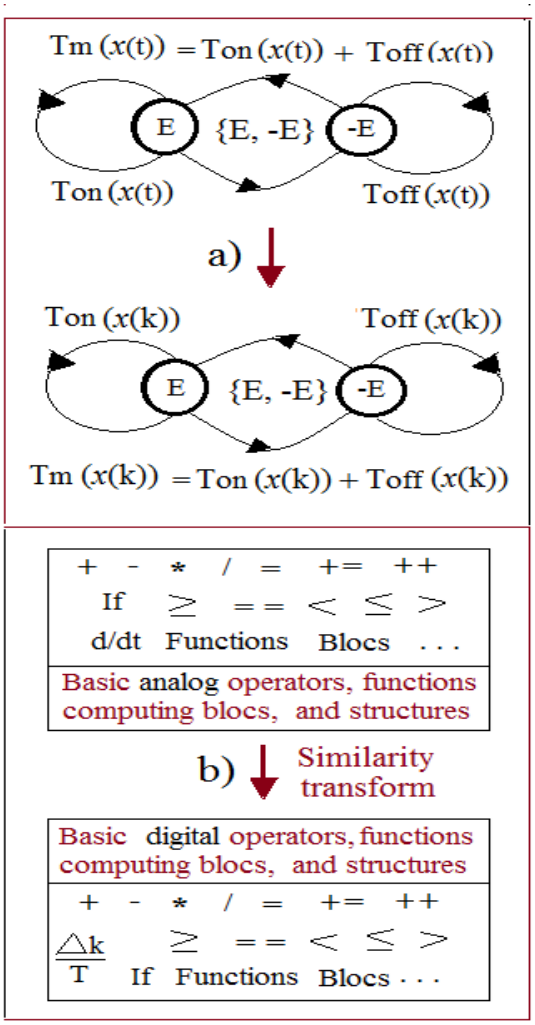
In [4], the z-transform was applied to (4), in order to build Simulink-based graphical solver. Then, the key operator required for emulating *jumps instants of the two-state Markov chain* $x_m(k) = \{+E, E\}$, was a piecewise smith trigger relay. However, although simulations under Simulink-based of virtual discrete

model were successfully conducted, the ambition in further research works, to migrate the related Simulink scheme into Xillinx BlocAdd editor for possible implementation into FPGA targets, was and remains unfeasible. Of course, a pure smith trigger relay model, is unavailable in Xillinx System generator, in latest versions available today in 2020.

Given that lack in *Xilinx System generator* of a suitable digital solver for *jump Markovian dynamic models* {(4), (5), (6)}, we resort in this paper to three relevant similarity principles between {(1), (2), (3)} and {(4), (5), (6)}, i.e. {*Structural, operational, Waveform}* principles. The first two similarity principles {structural, operational} are schematically summarized in Fig. 3, whereas the *waveform similarity principle* is illustrated graphically in Fig. 4.

These three principles are outlined as follows:

- *Structural similarity principle*. It results from the fact that, the continuous time jump Markovian dynamic models defined by {(1), (2) (3)}, is *structurally similar* to that of the discrete time jump dynamic model (1) given (3) (see Fig. 3a).
- *Operational similarity principle.* It is founded on the fact that, solving {(4), (5), (6)}, in the digital world, requires similar digital operators (e. g. +, -, *, /, ==, ≥, ≤,, Δk/T/,) as shown in Fig. 3b.
- *Waveform similarity principles*. It is visually obvious since the waveforms topology shown in Fig. 4a as the solution of (1) given (3) in the analog domain, are the discrete versions of corresponding waveforms displayed in Fig. 4b.



a) Structural principle b) Operational principle

Figure 3. First two similarity principles

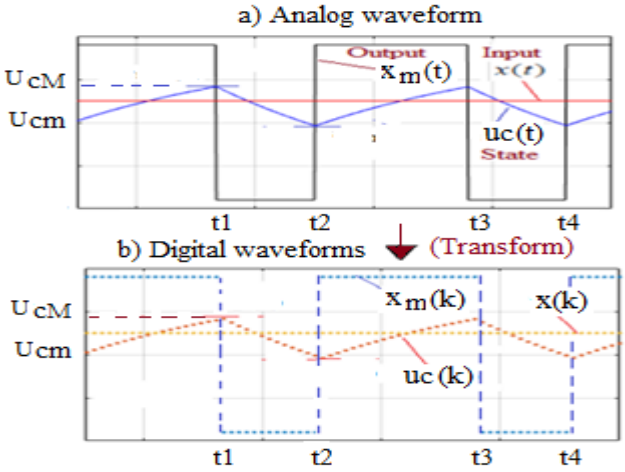
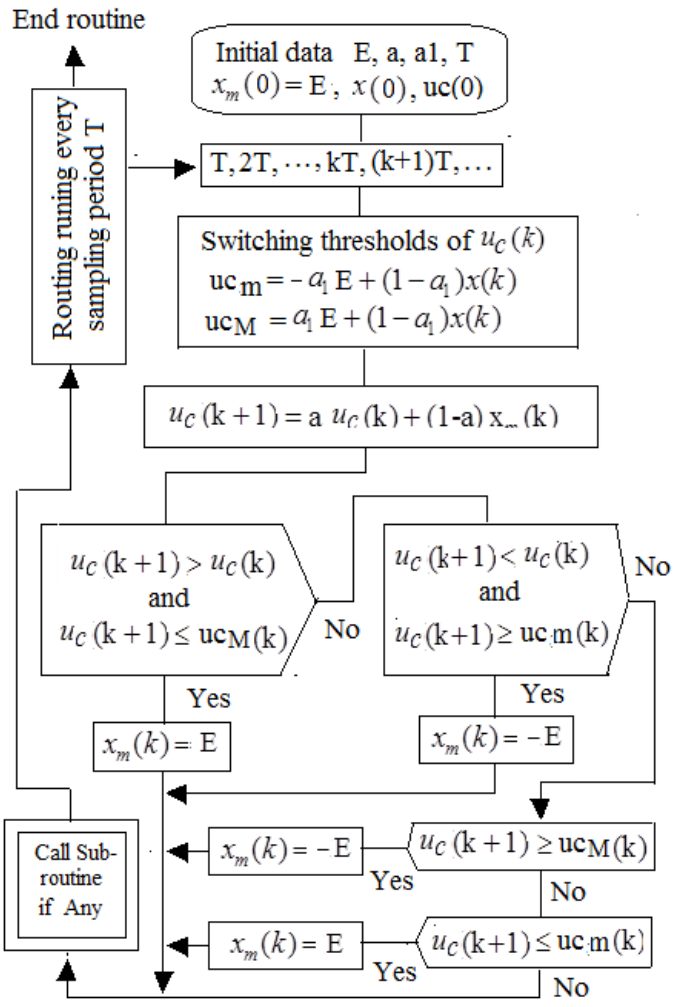


Figure 4. Waveform similarity principle

B. Proposed DDCM algorithmic scheme

Given a set of parameters {E, a1, a} in equations {(4), (5), (6)}, and an admissible set of initial conditions, e.g., {$u_c(0) = 0$, $x(0) = x0$, $x_m(0) = E$}, the proposed algorithmic scheme presented in Fig. 5, shows how {(4), (5), (6)}, can be recursively solved for each sampling period T, over a chosen discrete time horizon, with the help of either a numerical simulation tool, or a given programming language within any IDE (integrated development environment).



Subroutine Call: e.g., 1st order IIR filter

Figure 5. Proposed DDCM algorithmic scheme

It is worth noting in Fig. 4 that a call to a personalized subroutine block could be useful for decisions making on the basis of known values of {x(k), $x_m(k)$, uc(k)} at discrete times k T, with k = 0, 1, …, N. As a relevant example, the subroutine can be a called to a *digtalWrite* command, with $x_m(k)$ as argument value, to a digital output pin of the processing micro-chip. In which case the analog image x(t) of x(k) can be recovered outside using a simple analog low-pass RC filter, and the overall processing scheme becomes a DDCM-Based DAC.

For a case study to be outlined and well tested in this paper, the samples of the DDCM wave {$x_m(k)$}, are digitally processed over time within an ESP32 device, by a low-pass IIR (infinite impulse response) filter, in order to reconstruct the digital image $x_s(k)$ of the modulating control signal x(k). As a result, the overall DDCM scheme including the low-pass IIR filter, acts as an overall virtual DDCM-based DAC.

The low-pass IIR filter to be used is synthesized from an analog first order Rs-Cs structure, with transfer function:

$$D(s) = \frac{Xs(s)}{Xm(s)} = \frac{Bs}{1 + s\ Rs\ Cs} \qquad (7)$$

Where the nominal value of Ks could be computed as in [6] as follows:

$$Kf = \frac{(1+\alpha_1)}{2\ \alpha_1} \log\left(\frac{1+\alpha_1}{1-\alpha_1}\right) \qquad (8)$$

Using the pole-zero discretization technique, (9) leads to a discrete transfer function of the 1st order IIR filter (9) given (10).

$$D(z) = \frac{Xs(z)}{Xm(z)} = \frac{B}{z - As} \qquad (9)$$

$$A = e^{-T/(Rs\ Cs)},\ B = Bs\ (1\text{-}A_s) \qquad (10)$$

As an implication, the recursive equation of that IIR filter with initial conditions $x_s(0) = 0$ and $x_m(0) = E$, can be written as follows:

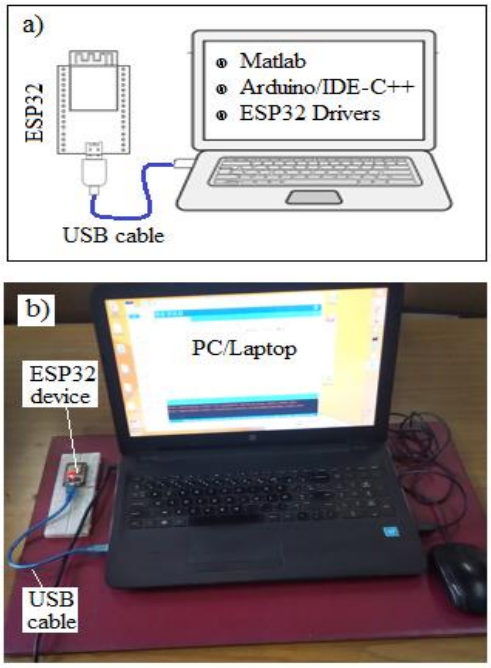
$$x_s(k) = As * x_s(k-1) + Bs * x_m(k-1) \qquad (11)$$

C. Simulation data and tools

The data set of the analog prototyping system to be used in the illustrative example is {E = 9 V, T= 1e-04, E= 9 V, [Am = 2, fs = 20 Hz} for a sine modulating x, {R1 = 2.2 kΩ, R2 = 33 kΩ} (i.e. according to (2)), a1 = 0.625 given (3), {R = 4.4 kΩ, C= 0.22 uF} with RC = 1e-03, and Kf = 0.0625 according to equation (9).

The simulation software tools used for rapidly solving the DDCM-based DAC problem associated with the overall DDCM algorithmic scheme are: Matlab, Arduino/IDE-C++ (version 1.8.12), and ESP32 drivers. Matlab is used as a high level tool for signal processing and numerical analysis. Then, Arduino/IDE-C++ 1.8.12, is a latest development environment of C++ sketches to be embedded for real time application, into numerous types of micro-chip boards, including ESP32 SOC device. On the order hand, ESP32 drivers are required for operating and programming needs of ESP32 devices within Arduino/IDE-C++ platform.

Fig. 6 shows the virtual image (in Fig 6a) and the experimental setup (in Fig. 6b), of a simple workbench used in this research work, for simulation, programming, running and virtual monitoring a relevant application of the pioneering DDCM algorithmic scheme initiated in this paper.



a) Schematic view b) Experimental setup

Figure 6. Workbench for the proposed ESP32-based DDCM drivers with application to virtual DAC.

A relevant part of Matlab code used to implement the ESP32-based DDCM-DAC is provided in Appendix A1. It has been easily translated into Arduino IDE/C++ sketch as shown in Appendix A2. Then, the resulting sketch has been updated and restructured for the aim of recursively solving {(4), (5), (6), (11)} within an unconditional computing loop, while sending numerical results to be graphically visualized on the USB virtual monitoring screen provided by Arduino/IDE-C++. The communication rate between the ESP32 and the virtual monitor is 230400 bauds.

III. RESULTS AND DISCUSSIONS

A. Matlab simulation results and discussions

Recall that, the transfer function given by (7), stands for the model of a 1st order analog Rs-Cs filter, to be discretised, given the set of parameters {Rs = 47 KΩ, Cs = 2.13 uF (i.e. Rs Cs = 0.1) and Ks = 8, Kf = 1.0639}, computed according to (9) as in [6].

Fig. 7 shows the graphs of {x (modulating sine wave in Fig. 7a), x_m (switching DCM signal in Fig. 7b), u_c (voltage across C in Fig. 1), and that of frequency spectrum plot of x_m in Fig. 6c, i.e., a visual waveform of the numerical data associated with 250000-points FFT (Fast Fourier Transform).

In Fig. 7b, it is important to observe in depth the waveforms of $x_m(t)$ and $u_c(t)$ signals. Indeed, as exactly predicted earlier, $\{x_m(t), t\}$ is a *continuous time jump Markovian process, punctuated of two discrete states*. In addition, $u_c(t)$ is visually as predicted, the encapsulation envelope of the modulating input signal x(t). A better understanding of the signal encapsulation principle is illustrated in Fig. 7c, where it is visually clear that the modulating signal with 20 Hz frequency, is an isolated low harmonic component within the remaining higher frequencies of the same spectrum.

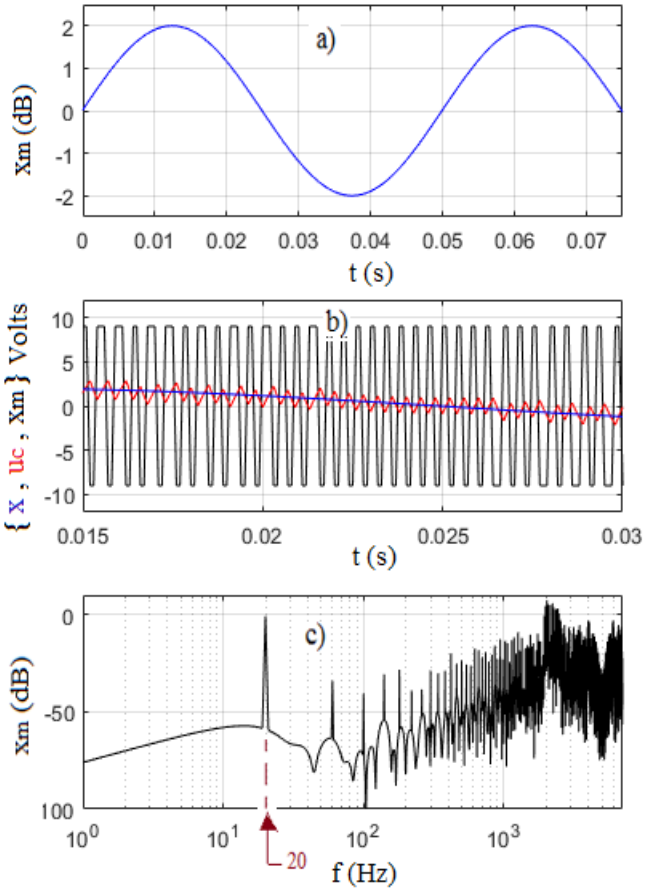


Figure 7. Matlab simulation of the simple DDCM algorithm.

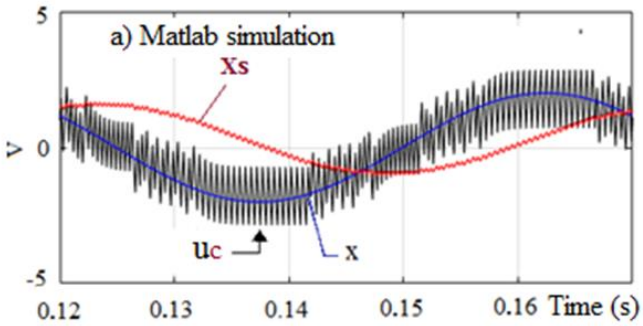
From the anatomy of the frequency spectrum of $x_m(t)$ (modulated wave) observed in Fig. 7c, it is now clearer that the sine-modulating x with its isolated 20 Hz marginal frequency, could be rigorously recovered from Xm(t), using a simple analog low-pass filter.

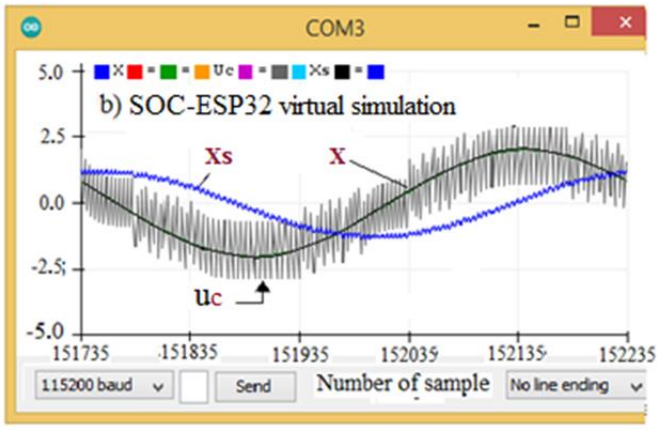
B. Matlab and ESP32 results for DDCM-DAC

The case study considered in this section is an DDCM-based DAC, which has been independently simulated in Matlab, and then embedded for real time executing into an ESP32 device, with possible visualization of digital processing results via Arduino virtual USB monitor.

Fig. 8 shows a graphical comparison of most relevant results, independently obtained under the same operating conditions consisting of:

- Matlab code written for pure digital processing of the DDCM-based DAC case study. Fig. 8a presents corresponding predicted results, obtained and plotted under Matlab.
- ESP32 device with embedded version of a corresponding Arduino C++ sketch. Fig. 8b shows the results, also obtained in this case.





a) Matlab Simulation b) ESP32 processing and virtual monitoring

Figure 8. Comparison of Matlab and ESP32 processing results for DDCM-based DAC

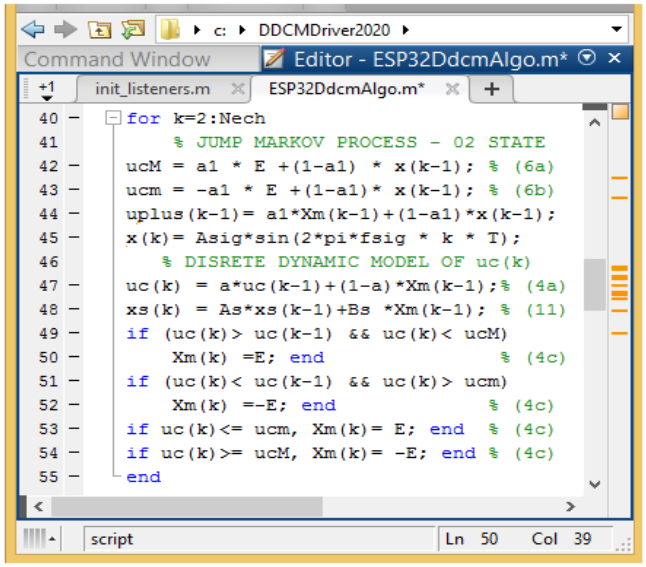
The most relevant finding emerging from Fig.8a and Fig. 8b is that, the whole waveforms in Fig. 8b (i.e., sine modulating input x, intermediary modulated wave uc, digital IIR response xs}, behave as perfect digital photocopies (or images) of their corresponding waves predicted in Fig. 8a from a pure simulation Matlab framework. The high quality of these results, is the insurance of the realistic nature and the great challenge of the synthesis approach initiated and well tested in this paper of low cost DDCM drivers for SOC devices.

IV. CONCLUSION

The most relevant merit of this paper has been a great challenge of showing and successfully proven a new digital synthesis scheme, for rapid and easily implementing DDCM drivers into SOC devices, e.g. ESP32 chip. A relevant future research opportunity of this work is to apply the proposed DDCM algorithmic scheme, to the design and realization of a complete ESP32-based DDCM DAC for digital control of DC-DC and DC-AC power converters. Another research perspective might be to extend the analytical and numerical developments conducted here, to the case of ESP32-based multichannel DDCM drivers. Therefore, resulting new generation of low cost and fast DDCM drivers for SOC devices, will have a growing and relevant impact factor in industrial and digital communication electronics.

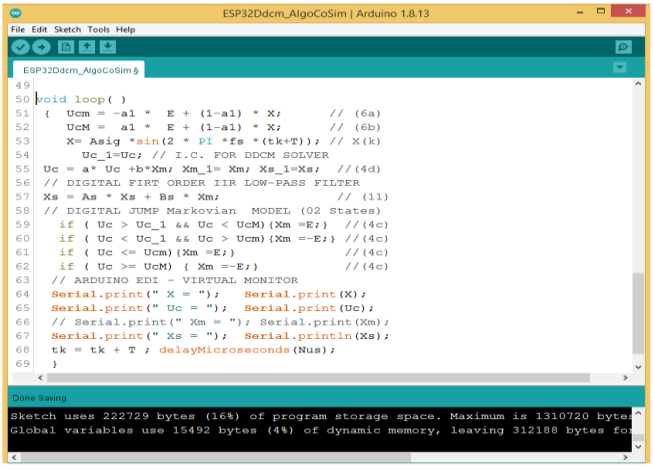
V. APPENDIX

APPENDIX A1: MATLAB CODE FOR DDCM-Based DAC



The content of the above Appendix A1, shows that a few lines of Matlab code, is sufficient for recursively computing the jump Markovian process xm(k), and digital simulating a pure DAC system.

APPENDIX A2. ARDUINO C++ CODE FOR DDCM-Based DAC



Furthermore, the content of the above Arduino sketch, proves that a real time execution loop of both discrete jump Markovian process $x_m(k)$, and

digital processing with virtual monitoring a DAC system, involve a low cost C++ code. It is a great challenge to see that the corresponding sketch only uses 16% the program memory space, and a maximum of 4% dynamic memory of the target ESP32 device.

VI. REFERENCES

[1]. J. Holtz, “Pulse width modulation: A survey”, IEEE Transactions on power Electronics, vol. 39, no. 5, pp. 410-420, 1992.

[2]. Sangil Park, "Motorola digital signal processors – Principles of sigma-delta modulation for analog-digital converters”, February, 2010, pp 1.1-9.6J.

[3]. J. Mbihi, B. Ndjali, and M. Mbouenda, “Modelling and Simulations of a Class of Duty Cycle Modulators for Industrial Instrumentation”, Iranian Journal of Electrical and Computer Engineering, Vol.4, N°2, 2005.

[4]. W. Jung and M. J. Hawksford, “An oversampled digital PWM linearization technique for digital-to-analog conversion”, IEEE Transactions on circuits and systems, Vol. 51, No. 9, pp. 1781-1789, September 2004.

[5]. M. Aboudina, and Behzad Razavi, “A New DAC Mismatch Shaping Technique for Sigma–Delta Modulators”, IEEE Transactions on Circuits and Systems-II: Express briefs, Vol. 57, No. 12, December 2010, pp. 966-970.

[6]. Moffo Lonla B., Mbihi J. A. “Novel Digital Duty–Cycle Modulation Scheme for FPGA-Based Digital-to-Analog

[7]. Conversion,” IEEE Transaction on circuits and system II. 2015; 62(6), pp. 543–547.

[8]. G. Sonfack, J. Mbihi, B. Lonla Moffo, “Optimal Duty-Cycle Modulation Scheme for Analog-To-Digital Conversion Systems”. International Journal of Electronics and Communication Engineering Vol:11, No:3, p.348-354, March 2017.

[9]. L. N. Nneme, B. M. Lonla, G. B. Sonfack, and J. Mbihi, ''Review of a Multipurpose Duty-Cycle Modulation Technology in Electrical and Electronics Engineering'', Journal of Electrical Engineering, Electronics, Control and Computer Science, vol. 4, no 2, p. 9-18, 2018

[10]. L. Nneme Nneme and J. Mbihi, “Virual simulation and comparison of sine pulse width and sine duty-cycle modulation drivers for single phase power inverters”, Journal of electric engineering, Electronics, Control and Computer Science, vol. 6, Issue 21, pp. 31-38, 2020.

[11]. J. Mbihi, “Dynamic modelling and virtual simulation of digital duty-cycle modulation control drivers”, International Journal of Electrical, Computer, Energetic, Electronic and communication Engineering”, vol. 11, no. 4, pp. 472-477, 2017.

Cite this article as :

Jean Mbihi, Léandre Nneme Nneme, "Development of Jump Markovian Dynamic Models of Low Cost Digital Duty-Cycle Modulation Drivers For Target System-on-Chip Devices ", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 6 Issue 6, pp. 364-372, November-December 2020. Available at doi : https://doi.org/10.32628/CSEIT206620
Journal URL : http://ijsrcseit.com/CSEIT206620