POLITECNICO DI TORINO
Repository ISTITUZIONALE

Software-Defined DDPM Modulators for D/A Conversion by General-Purpose Microcontrollers

Original
Software-Defined DDPM Modulators for D/A Conversion by General-Purpose Microcontrollers / Abdullah, Ahmed; Musolino, Francesco; Crovetti, PAOLO STEFANO. - In: IEEE ACCESS. - ISSN 2169-3536. - ELETTRONICO. - 10:(2022), pp. 17515-17525. [10.1109/ACCESS.2022.3150865]

Availability:
This version is available at: 11583/2955122 since: 2022-04-25T18:38:13Z

Publisher:
IEEE

Published
DOI:10.1109/ACCESS.2022.3150865

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

29 March 2024

IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 21, 2021, accepted February 7, 2022, date of publication February 10, 2022, date of current version February 17, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3150865

Software-Defined DDPM Modulators for D/A Conversion by General-Purpose Microcontrollers

AHMED ABDULLAH, FRANCESCO MUSOLINO, (Member, IEEE), AND PAOLO CROVETTI, (Senior Member, IEEE)
Department of Electronics and Telecommunication (DET), Politecnico di Torino, 10129 Turin, Italy
Corresponding author: Paolo Crovetti (paolo.crovetti@polito.it)

ABSTRACT The software implementation of Dyadic Digital Pulse Modulators (DDPMs) for Digital to Analog (D/A) conversion is addressed in this paper. In particular, an enhanced software DDPM implementation is proposed and compared with a plain, iterative software transposition of the basic DDPM hardware architecture. Experimental results on an 8-bit software-defined DDPM D/A converter implemented on a Texas Instrument c2000 microcontroller platform validate the approach, revealing for the novel optimized software DDPM a 6X maximum sample rate compared to the simple iterative implementation on the same microcontroller and at the same system clock frequency. Based on measurements, an 8-bit DDPM DAC featuring the proposed optimized implementation operates at 7.8kS/s with a maximum INL of 1.64LSB, a maximum DNL of 1.79LSB, an SFDR of 47.02dB and a SNDR of 45.27dB, corresponding to 7.23 ENOB, demonstrating the effectiveness and the applicability of the proposed approach to implement a low cost, software-defined D/A converters in microcontroller-based embedded systems.

INDEX TERMS Dyadic digital pulse modulation (DDPM), digital to analog converter (DAC), software-defined data converters, software-defined DAC, software implementation, c2000 microcontroller.

I. INTRODUCTION

The dyadic digital pulse modulation (DDPM) has been introduced in [1] to generate deterministic bitstreams with a pulse density proportional to a binary input code, in which the most relevant spurious energy content is pushed at high frequency. The spectral characteristics of DDPM streams have been exploited to release the constraints of the reconstruction filter in baseband digital-to-analog (D/A) conversion [1]–[3], to perform dynamic offset calibration in digital-based operational transconductance amplifiers [4], [5], and have also been applied in analog-to-digital (A/D) conversion [6], [7], in digital RF amplitude modulation [8] and in digitally-controlled power converters [9], [10], aiming to increase the effective resolution of digital pulse width modulators (DPWM) as demanded to avoid the onset of limit cycle oscillations.

Compared to other modulations and dithering techniques, DDPM is amenable of an extremely area- and power-efficient all-digital hardware (HW) implementation, which makes it attractive in tightly power- and cost-constrained Internet of Things (IoT) sensor nodes [2], [3], [11]–[18]. Along with the theoretical assessment of the modulation and its spectral properties, indeed, the efficient HW implementation of a DDPM modulator has been addressed starting from [1], in which two different DDPM modulator architectures have been proposed targeting an FPGA DAC implementation alternative to DWPM [19] and Sigma-Delta [20]. One of the DDPM modulator HW architectures proposed in [1] was also adopted in the standard-cell based synthesized DDPM modulators integrated in 40nm CMOS presented in [6], [7] and in [4], and is also the basis of the DDPM modulator included in the Dyadic Digital Pulse Width Modulator (DDPWM) proposed in [9], [10] and of the DDPM-based RF modulator implementation in [8]. A different DDPM modulator HW implementation has been finally proposed in [3], to achieve graceful performance degradation under frequency and supply voltage overscaling.

While a significant attention has been paid so far to the digital HW implementation of a DDPM modulator, the software (SW) implementation of a DDPM modulator in a traditional microprocessor/microcontroller HW has not been

The associate editor coordinating the review of this manuscript and approving it for publication was Vivek Kumar Sehgal.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

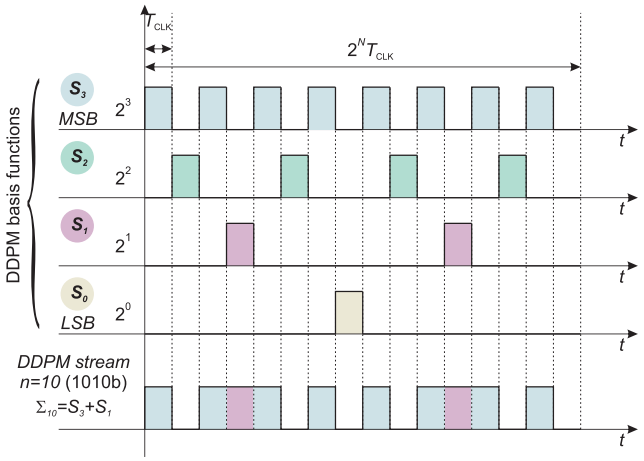


FIGURE 1. Dyadic Digital Pulse Modulation [1]: 4-bit DDPM basis functions and DDPM pattern for $n = 10$ (1010b).

specifically addressed, thus limiting in many practical cases the applicability and the advantages of the DDPM modulation either to application specific integrated circuits (ASIC) or FPGA implementations.

In this paper, different DDPM modulator architectures are discussed and compared with regards to their HW and SW implementation. In particular, a new DDPM modulator architecture, which is amenable to an efficient SW implementation is proposed and compared to a SW architecture resulting from the straightforward translation of the HW implementation. The effectiveness and the efficiency of the proposed SW DDPM modulator is finally demonstrated by the implementation of a software-defined 8-bit DDPM DAC on an off-the-shelf Texas Instruments c2000 microcontroller platform [21].

The rest of the manuscript is organized as follows. In Sect.II, the DDPM modulation and its spectral properties are shortly revised, in Sect.III, the DDPM modulator HW implementations proposed so far are revised and compared in terms of HW complexity and performance, also in view of their possible translation in SW. Moreover, a new optimized DDPM modulator architecture specifically intended for SW implementation is proposed and compared to the other solutions. The effectiveness of the proposed solution is then verified in Sect.IV by the SW implementation of an 8-bit DDPM DAC in a general purpose microcontroller unit, whose operation is experimentally verified in the same Section. Some concluding remarks are finally drawn in Sect.V.

II. THE DDPM MODULATION

Before discussing the HW and SW implementations of a DDPM modulator, the definition and the basic spectral properties of a DDPM stream, as introduced in [1], are revised in this Section.

A. DDPM STREAM DEFINITION

The DDPM modulation, as originally proposed in [1], is a digital modulation technique which associates to an integer number

$$n = \sum_{i=0}^{N-1} b_i 2^i, \tag{1}$$

with a binary representation on N bits

$$B_n[N-1:0] = (b_{N-1}, b_{N-2}, \ldots, , b_1, b_0), \tag{2}$$

the periodic digital stream

$$\Sigma_n(t) = \sum_{i=0}^{N-1} b_i S_i(t) \tag{3}$$

obtained by superposition of orthogonal dyadic basis functions $S_i(t)$ $(i = 0, \ldots, N-1)$ defined on the fundamental period $(0, 2^N T_{\rm clk})$ as:

$$S_i(t) = V_{\rm DD} \sum_{h=0}^{N-1} \Pi\left(\frac{t}{T_{\rm clk}} - h \cdot 2^{N-i} - 2^{N-i-1} + 1\right), \tag{4}$$

where $T_{\rm clk}$ is the clock cycle and $\Pi(x)$ is the unit pulse ($\Pi(x) = 1$ for $0 \le x \le 1$ and $\Pi(x) = 0$ elsewhere).

Orthogonal dyadic basis functions $S_i(t)$ are N non-overlapping, periodically repeated digital streams of 2^N clock cycles, organized so that S_{N-1} is high (i.e. at $V_{\rm DD}$) every other clock cycle (i.e., in 2^{N-1} cycles per period), S_{N-2} is high every other cycle in which S_{N-1} is low (i.e., in 2^{N-2} cycles), S_{N-3} is high every other cycle in which both S_{N-1} and S_{N-2} are low (i.e., in 2^{N-3} cycles per period) and so on, till S_0, which is high just in one cycle per period, as shown in Fig.1.

Since the basis functions $S_i(t)$ are non-overlapping, i.e.

$$S_i(t) \cdot S_j(t) = 0 \quad \forall i \neq j \in \mathbb{N}$$

and high in exactly 2^i clock cycles per period, DDPM streams Σ_n defined in (3) are high for exactly n clock cycles per period and their pulse density is therefore $n/2^N$ as observed in the same Fig.1, where the construction of a DDPM stream by superposition of dyadic basis functions $S_i(t)$ is illustrated for $n = 10$.

The definition of the DDPM modulation given in (3) can be shown to be equivalent to the expression provided in [3], according to which the DDPM bitstream Σ_n associated to the integer n represented in binary as in (1) can be expressed as $\Sigma_n = [\Theta_N, 0]$, where $\Theta_0 = 0$ and Θ_i with $i = 1 \ldots N$ are recursively defined as

$$\Theta_i = [\Theta_{i-1}, b_{N-i}, \Theta_{i-1}] \tag{5}$$

in which $[\cdot, \cdot]$ is the binary string concatenation operator.

B. SPECTRAL PROPERTIES OF DDPM STREAMS AND APPLICATIONS

The spectra of a DDPM stream have been evaluated in [1] by the Fourier transform of (3) as:

$$\Sigma_n(f) = \frac{V_{\rm DD}}{2^N} \sum_{k=-\infty}^{+\infty} c_{k,n} \,\mathrm{sinc}\left(\frac{k}{2^N}\right) \delta\left(f - k f_0\right) \tag{6}$$

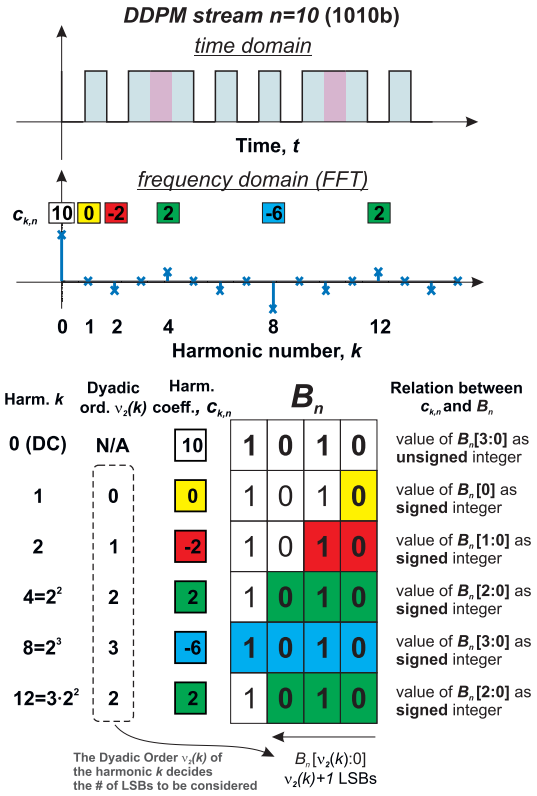


FIGURE 2. Illustration of the relation between the harmonic spectral coefficients $c_{k,n}$ and the binary representation B_n of the digital input code n, as presented in [8], for the digital input $n = 10$ (i.e. $B_n = (1, 0, 1, 0)$).

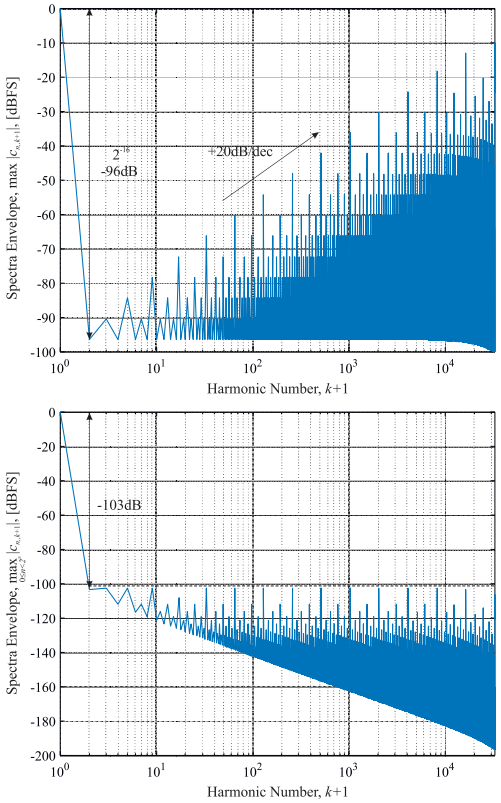


FIGURE 3. Envelope of the magnitude spectra of $N = 16$ bit DDPM streams before (top) and after (bottom) first-order low pass filtering.

where $f_0 = 1/T_0 = f_{\rm clk}/2^N$, sinc$(\cdot)$ is the cardinal sine function, and

$$c_{k,n} = \sum_{i=0}^{N-1} b_{i,n} 2^i \sum_{m=0}^{2^{N-i}-1} \delta\left[k - 2^i m\right] \mathrm{e}^{-j\pi m} \tag{7}$$

in which $\delta[\cdot]$ is the Kroenecker function defined as

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0. \end{cases}$$

Moreover, in [8] it has been shown that, very interestingly, the spectral coefficients $c_{k,n}$ in (6) can be equivalently expressed as

$$c_{k,n} = \begin{cases} n & k = 0 \\ -b_{\nu_2(k)} 2^{\nu_2(k)} + \sum_{i=0}^{\nu_2(k)-1} b_i 2^i & k \neq 0 \end{cases} \tag{8}$$

where, from number theory [22],

$$\nu_2(k) = \max\left\{\nu \in \mathbb{N} : k | 2^\nu\right\} \tag{9}$$

is the *dyadic order* of the integer k, i.e., the largest exponent ν such that 2^ν divides k.

In other words, as illustrated in Fig.2, the DC component ($k = 0$) of the DDPM stream is expectedly equal to the value of the binary input n, whereas higher order spectral coefficients related to the amplitude of the k-th harmonic component of the DDPM spectrum, take the value of the binary string[1]

$$B_n\left[\nu_2(k) : 0\right] = (b_{\nu_2(k)}, \ldots, b_0)$$

consisting of the last $\nu_2(k)+1$ least significant bits (LSBs) of the binary representation of the input code n, interpreted as a signed integer in two's complement notation.

C. APPLICATIONS OF DDPM

The interest of DDPM in D/A conversion [1]–[3] is related to the envelope of the spectra of DDPM signals evaluated as in (6) over different input codes n, i.e.

$$S(kf_0) = \max_n \left| X_n^{\rm DDPM}(kf_0) \right| \tag{10}$$

which is reported in Fig.3 (top) for $N = 16$ bit. Indeed, it can be observed that, while the DC component of DDPM streams is proportional to the digital input n, their AC spectral components are concentrated at high frequencies and can be easily filtered out. In detail, based on (8), the energy of the spectral components is related to $2^{\nu_2(k)}$, and increases with k with a slope of 6dB/oct or 20dB/dec. It follows that in a DAC based on DDPM, a first-order filter with a cutoff frequency of $f_{\rm c} = f_{\rm clk}/2^N\sqrt{3}$ is sufficient to keep all the spurious DDPM spectral components $-6(N+1)$dB below the DC [1],

[1] In this expression it is assumed $b_i = 0$ for $i > N-1$.

as illustrated in Fig.3 (bottom). The same properties have also been exploited to design DDPM-based voltage-mode and current A/D converters, as described in [6], [7].

In view of the low-frequency characteristics of DDPM streams highlighted above, DDPM has also been adopted in power electronics [9], [10], as a dithering technique to increase the effective resolution of DPWM modulators in digitally controlled switching mode power converters, so that to suppress quantization-induced limit cycle oscillations with minimum output voltage ripple degradation and without impairing the dynamic performance, and in dynamic offset calibration of analog integrated circuits [4].

Finally, leveraging (8), in [8] it has been observed that the same DDPM can be conveniently adopted for mixer-less RF digital amplitude modulation in SW-defined radio transmitters.

III. DDPM MODULATOR IMPLEMENTATIONS

In all the above applications, DDPM modulators have been efficiently synthesized starting from an HW description language (HDL) behavioral description either on an FPGA [1], [8], [9] or in an application-specific integrated circuit (ASIC) [2], [3], [7].

Although previously proposed FPGA and ASIC DDPM modulators are very compact and energy efficient, the implementation of a DDPM modulator in low-cost electronic systems based on off-the-shelf components can be easily not convenient since commercial microcontroller units do not include at present a DDPM modulator peripheral, and the implementation of the DDPM modulator by a separate FPGA, ASIC or spare logic is impractical and possibly expensive.

Addressing the above limitation, the SW implementation of a DDPM modulator is tackled in this paper. For this purpose, previously proposed HW architectures are reviewed, and their possible SW implementation is discussed. Moreover, new DDPM modulator architectures, suitable to an efficient SW implementation on standard digital HW (microprocessor/microcontreller unit) are introduced and compared.

A. PARALLEL DDPM MODULATOR

The first DDPM modulator implementation proposed in [1] is based on a single parallel-input, serial-output (PISO) digital register. In this architecture, the bits of the register are loaded in parallel with the bits of the input code according to the DDPM pattern, as illustrated in Fig.4, and are then streamed-out serially to the output.

Such a solution does not require any combinational logic and is therefore suitable to operate at a very high clock frequency, but its complexity (number of D flip-flops) increases exponentially with the number of bits N of the modulator, so that its area and power efficiency is rather low for N exceeding 4-5. Due to this limitation, the parallel architecture in Fig.4 has not been adopted so far in FPGA and ASIC implementations. Since the DDPM function is hardwired in the interconnections of the different memory elements of the circuit in Fig.4, such a solution is clearly not suitable to be implemented in SW.

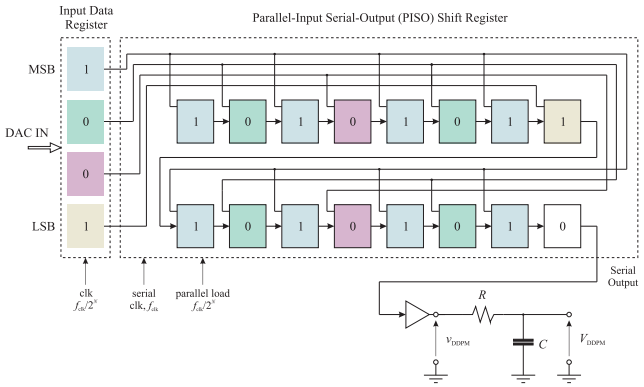


FIGURE 4. Parallel Hardware implementation of a 4-bit DDPM modulator [1].

B. PRIORITY MUX-BASED DDPM MODULATOR

A different DDPM modulator implementation, which was also proposed in [1], is shown in Fig.5 and its behavior is described in synthesizable behavioral VHDL in Fig.6. Its operation is based on a *priority MUX*, i.e. a combinational network with $2 \cdot N$ inputs, among which N data inputs $D_{N-1} \cdots D_0$ and N selection inputs $S_{N-1} \cdots S_0$, and one output X, described by the Boolean function

$$X = \sum_{i=0}^{N-1} D_{N-i-1} \cdot S_i \cdot \prod_{k=0}^{i-1} \overline{S_k}, \tag{11}$$

where sums and products are intended as Boolean OR and AND operators, respectively. The output X takes the value of D_{N-k}, i.e. the k-th bit from the MSB, of the data input, being k the index of the first "one" in the selection inputs starting from the LSB, i.e. the index for which $S_k = 1$ and $S_i = 0$ for[2] $i < k$.

The data inputs of the priority MUX are fed by the DAC input data register, while the selection inputs are connected to a free-running 2^N-bit binary counter, so that every other clock period, $S_0 = 1$ and the output X takes the value D_{N-1} of the MSB of the input data. Moreover, in the remaining counting periods in which $S_0 = 0$, in one half of the cases (i.e. 2^{N-2} times in a full count), $S_1 = 1$ and the output X takes the value of the second MSB D_{N-2} of the input and so on, so that the output X is driven in a full counting period to the logical value of the bit D_i of the input exactly 2^i times, arranged according with the DDPM pattern in Fig.1.

The priority MUX-based DDPM modulator HW architecture achieves a good complexity-performance tradeoff and has been therefore widely adopted so far both in ASIC and in FPGA implementations [1]–[3], [7]–[9]. A modified version of the same DDPM modulator architecture, which achieves graceful performance degradation under clock frequency and

[2]it is assumed that $k = 0$ if all selection inputs are at zero.

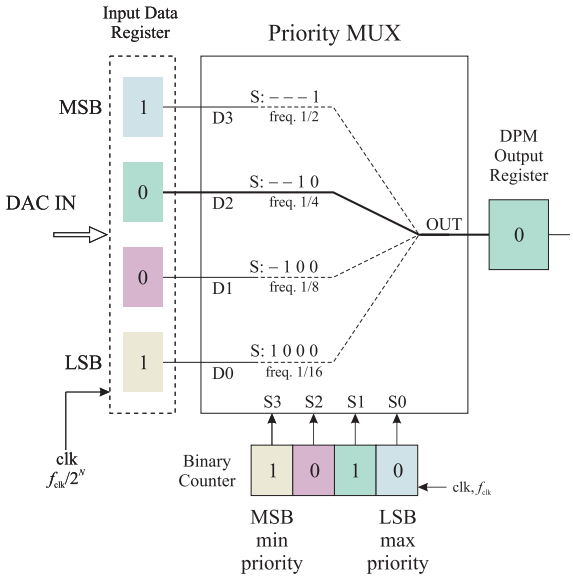


FIGURE 5. Priority MUX-based Hardware implementation of a 4-bit DDPM modulator [1].

```
process (clk)
variable INDEX: integer:= N;
variable DATA: std_logic_vector(N downto 0);
begin
 if (clk'event AND clk='1') then
  DATA:=DAC_IN(N–1 downto 0)&'0';
   for i in N–1 downto 0 loop
    if CNT(i)='1' then
     INDEX:=i;
    end if;
  end loop;
 DDMP_OUT<=DATA(N–INDEX);
 CNT<=CNT+'1';
 end if
end process
```

FIGURE 6. Simplified VHDL Description of the DDPM Modulator in Fig.5.

supply voltage overscaling, thanks to the special custom implementation of the priority MUX shown in Fig.7 has been also proposed in [2].

C. ITERATIVE DDPM MODULATOR

Since the DDPM modulator presented in Sect.III-B requires a priority MUX combinational network, which is normally not included in arithmetic logic units (ALUs) nor is implemented by a dedicated opcode in general purpose microcontrollers, it is not immediately suitable to SW implementation. The same logical behavior described by the VHDL code in Fig.6 can be however translated almost straightforwardly in a procedural form, and implemented iteratively by the architecture shown in Fig.8.

In this architecture, the LSB of the binary counter is checked first: if $COUNT_0 = 1$, the value of the MSB of the input data register (i.e., $OUT = IN_{N-1}$) is assigned to

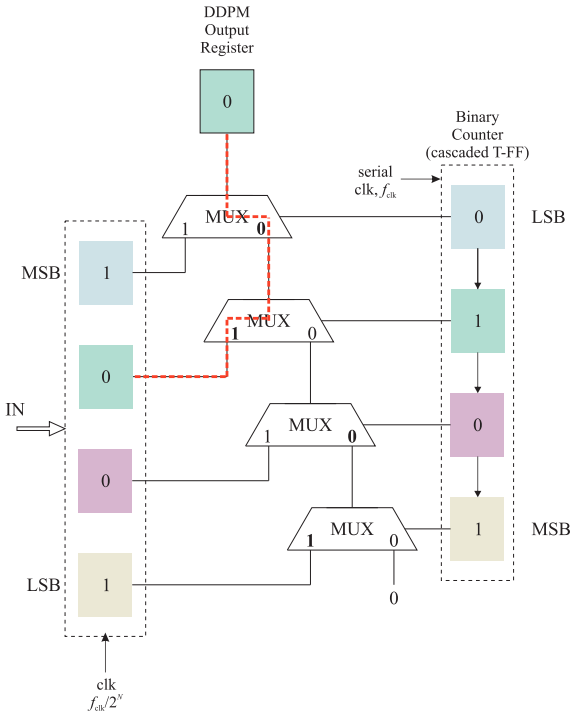


FIGURE 7. 4-bit DDPM modulator implementation featuring graceful performance degradation and dynamic power-resolution scaling [3].

the DDPM output and the procedure is terminated. Otherwise, the second LSB of the binary counter is checked: if $COUNT_1 = 1$, then the DDPM output is updated with the second MSB of input data register (i.e., $OUT = IN_{N-2}$) and the iteration ends. Applying the same approach up to MSB of the counter, when $COUNT_{N-1} = 1$, the DDPM output takes the value of IN_0 and then the iteration is terminated.

An HW implementation of an $N = 4$ iterative DDPM modulator is shown in Fig.8. It includes two shift registers used as "one hot" counters: a right shift register (RSR), and a left shift register (LSR), which are initialized with the binary values 1, 0, 0 . . . , 0 and 0, 0, . . . , 0, 1, respectively. For each iteration, the LSR output is ANDed with the binary counter value. If the result of the AND operation is true, the DDPM output bit is updated according to the result of the AND of the RSR and the input data register, and the iteration is terminated. Otherwise, the iteration continues until the "hot one" in the RSR is discarded from the MSB position and the RSR register content becomes zero.

Unlike the architectures discussed in Section IIIa and IIIb, the iterative architecture in Fig.8 is suitable to SW implementation in a general purpose microcontroller and can be translated into the C-code in Fig.9. Here, the LSR (RSR) are mapped into the `mask` (`cmask`) variables, which are right- (left-) shifted by one position at each iteration of the `while` loop. The content of `mask` is put in AND with the binary counter value `COUNT`, and the result is checked by an `if-else` construct. If the condition is true, the output variable `bit` returned by the function is either 1 or 0, depending

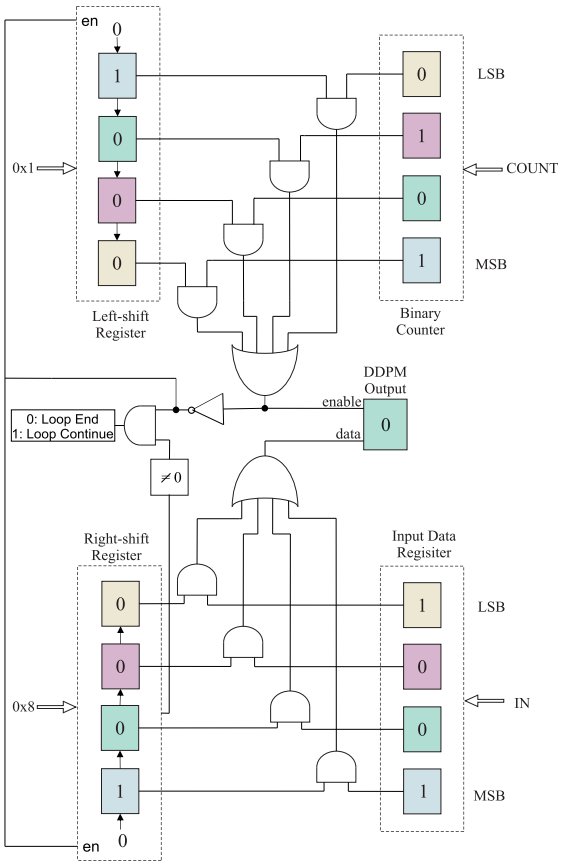


FIGURE 8. Hardware architecture of a 4-bit iterative DDPM modulator.

```
int ddpm_eval (int IN, int COUNT)
{
 int mask = 0x01;
 int cmask = 0x80;
 int loop = 1;
 int bit = 0;

 while ((loop != 0) & (cmask != 0))
    {if ((mask & COUNT) != 0)
          {bit = (cmask & IN) != 0;
           loop = 0;}
       else
          {cmask >>= 1;
           mask <<= 1;};
    }
  return bit;
}
```

FIGURE 9. C code of an iterative 8-bit DDPM Modulator in Fig.8.

on the result of the AND operation among `cmask` and input data value, and the `while` loop iteration is terminated. Otherwise, the `while` loop proceeds with one more iteration, and the same operations are performed until the `mask` varialbe becomes zero.

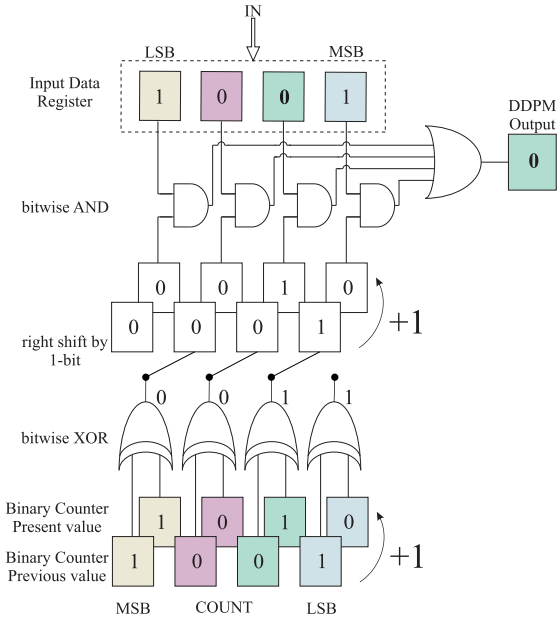


FIGURE 10. Architecture of a 4-bit DDPM modulator based on the proposed enhanced implementation.

```
int ddpm_eval(IN)
{
static COUNT = 0;
if (((((COUNT^(COUNT++))>>1)+1)&IN))
            bit= 1;
          else
            bit= 0;
return bit;
}
```

FIGURE 11. C code of the proposed optimized 8-bit DDPM Modulator in Fig.10.

The iterative SW implementation presented above is effective but not efficient in terms of execution time, since N iterations of the `while` loop are needed in the worst case, i.e. when just the MSB of `COUNT` is one and all the other bits are zero, to get one bit of the DDPM output stream. Since the sample rate needs to be the same for any input code, the worst-case execution time limits in practice the maximum sample rate at which the DDPM can be operated, resulting in modest performance.

D. PROPOSED OPTIMIZED DDPM MODULATOR

In order to address the limitations of the iterative DDPM modulator presented above, a new DDPM architecture suitable to SW implementation is proposed in Fig.10. In such an architecture, the position of the first 'one'' starting from the LSB in the content of the binary counter `COUNT` is obtained by bitwise XOR of the present value of `COUNT` and of the previous value of the counter, i.e. `COUNT-1`. The result of the XOR is a thermometric-encoded binary word in which all the bits starting from the LSB and up to the first ''one'' of `COUNT` are high, whereas the remaining bits, up to the MSB,

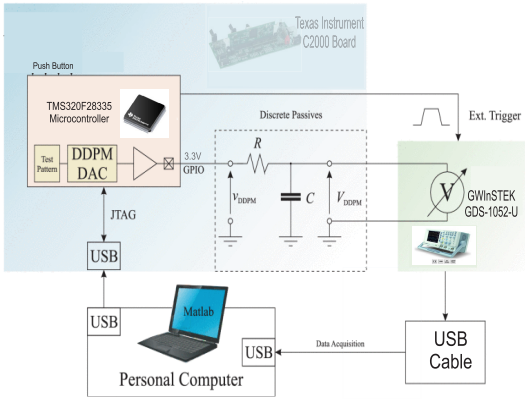


FIGURE 12. Experimental test setup.

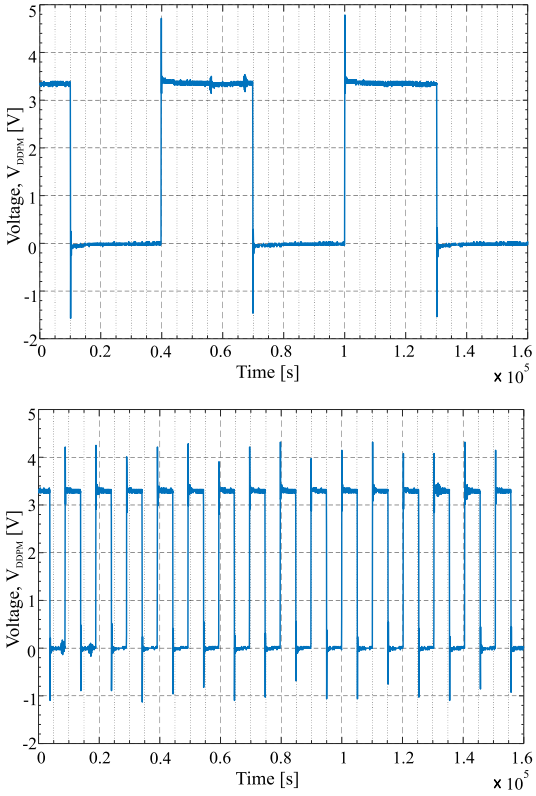


FIGURE 13. DDPM bitstream output for input code n= 127 (0 × 7F): iterative 8-bit DDPM (top) and proposed 8-bit DDPM (bottom) vs. time.

are low. Such a word is then right-shifted by one position and incremented by one unit, so that to get a word which has a single bit at "one", which turns out to be exactly in the position of the first "one" of the binary counter starting from the LSB. This word, when put in AND with the bit-reversed DDPM input gives the bit of the DDPM stream corresponding to `COUNT` and is the output of the DDPM procedure.

This architecture is amenable to SW implementation and can be translated into the C-code shown in Fig.11. The bit-wise XOR of `COUNT` (before incrementing) and `COUNT++` (after incrementing, which gives the present counter value) provides information on the position of the first "one" starting from LSB, in a thermometric-encoded fashion. In the example illustrated in Fig.10, since the first "one" lies at second LSB position, i.e.,`COUNT=1010`, the result contains two ones, i.e., `0011`. In order to retrieve the position of the first "one", the result is right-shifted by one position and once more incremented by one unit. The result (i.e., `0010` in the example) is then bit-wise ANDed with the bit-reversed input data. The input data must be bit-reversed to align respective bit positions and it is worth being observed that the input bit-reversal operation does not need to be performed at each DDPM clock period, but only once in a DDPM pattern consisting of 2^N clock periods and DDPM function evaluation, so it has a negligible impact on the execution time. The DDPM output is then eventually updated based on the result of the AND (*if*-condition in the code). For `IN=1001` in the example, the DDPM output takes the value "zero", since the second MSB of the input data, corresponding to the second LSB of the counter value is, "zero".

In this architecture, the whole N-bits of binary counter are evaluated in parallel to decide the DDPM output code, with a significant reduction of the execution time, which is now independent of the number of bits N of the modulator, thus significantly enhancing the achievable sample rate, especially for large values of N.

IV. HARDWARE TEST SETUP AND EXPERIMENTAL RESULTS

To validate the proposed DDPM modulators, two 8-bit software-defined DDPM DAC prototypes have been implemented on a c2000 microcontroller [21] using the simple iterative approach presented in Sect.III-C and the optimized SW implementation proposed in Sect.III-D, and their static and dynamic performance have been experimentally tested and compared.

A. MICROCONTROLLER-BASED DDPM DAC AND EXPERIMENTAL TEST SETUP

To highlight the effectiveness of the proposed DDPM technique, both the proposed and the iterative 8-bit software-defined DDPM modulator based on the architectures in Fig.10 and Fig. 8, respectively, have been implemented on a Texas Instruments c2000 microcontroller in C language, so that to drive a 3.3 V general-purpose digital output with a new DDPM pulse, evaluated as in Fig.9 for the iterative implementation and as in Fig.11 for proposed optimized implementation, in correspondence of a periodically-generated interrupt signal from the c2000 microcontroller timer peripheral [21]. The interrupt period has been tuned, both for the iterative and for the optimized DDPM modulator, to the minimum compatible with the correct DDPM DACs operation. The software-defined DDPM DACs operate at a system clock frequency of 150MHz and includes an on-board output RC filter with $R = 100$ kΩ and $C = 1$ nF.

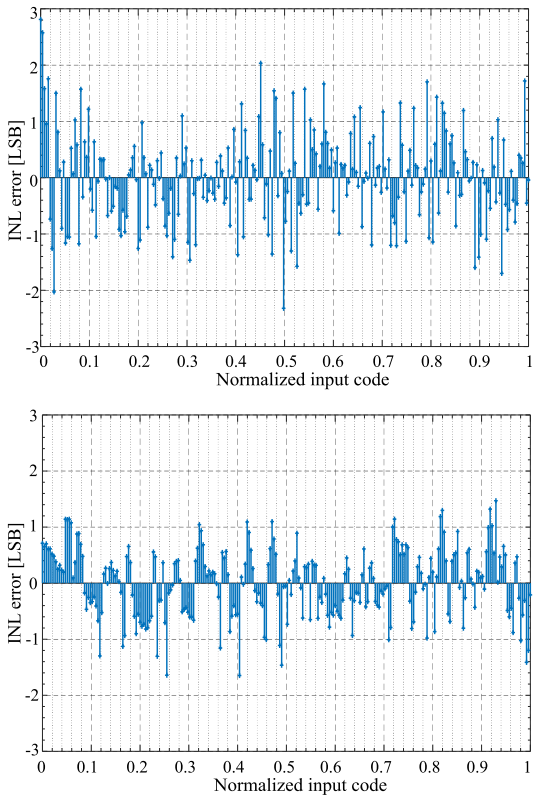


FIGURE 14. Integral non-linearity (INL) error: iterative 8-bit DDPM DAC (top) and proposed 8-bit DDPM DAC (bottom) vs. normalized input code.

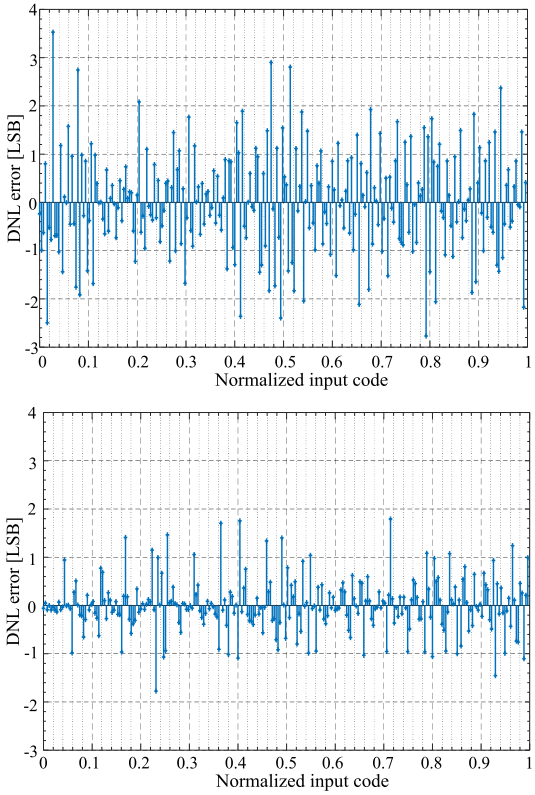


FIGURE 15. Differential non-linearity (DNL) error for the iterative (top) and for the proposed optimized (bottom) 8-bit DDPM DAC, vs. normalized input code.

The DAC output voltage $V_{\mathrm{DDPM}}(n)$ has been measured under static conditions and under sine-wave input with the test setup in Fig.12, applying the double-slope error digital compensation technique described in [1] and also adopted in [2], [3], [6], [7], which consists in applying to the DDPM DAC a digitally-predistorted input code n' evaluated from the integer n to be converted as:

$$n' = \begin{cases} \lceil \dfrac{n}{1+\alpha} \rfloor & \text{for } 0 \le n < 2^{N-1}(1+\alpha) \\ \lceil \dfrac{n-(2^N-1)\alpha}{1-\alpha} \rfloor & \text{for } 2^{N-1}(1+\alpha) \le n < 2^N \end{cases} \tag{12}$$

where $\lceil \cdot \rfloor$ is the rounding operator to the closest integer, α is a compensation factor evaluated based on one-time calibration so that to compensate the error due to the unbalance in the rise/fall times in the digital pulses generated by the microcontroller output drivers [1].

B. EXPERIMENTAL RESULTS AND DISCUSSION

For both the iterative and the proposed optimized DDPM modulators, the digital bitstream outputs for the input code $n = 127$ ($0 \times 7\mathrm{F}$) have been acquired by a digital oscilloscope and are shown in Fig.13.

For that specific input code, the output bitstream sequence is 0,1,0,1,..., having $2^N/2$ pulses are ones and remaining $2^N/2$ pulses are zeros. Considering the sequence and timing information of single pulse, the minimum DDPM unit pulse duration $T_{\mathrm{DDPM}} = 1/f_{\mathrm{DDPM}}$ set by the internal timer interrupts in our microcontroller platform, is found to be 500ns for the optimized DDPM implementation and 3μs for the iterative implementation, which correspond, for an $N = 8$-bit DDPM DAC, to a maximum sample rate $f_S = f_{\mathrm{DDPM}}/2^N$ of 7.81 kS/s and of 1.30kS/s, respectively, for the optimized and iterative implementation, as reported in Tab.1.

Based on the measured static input-output transcharacteristic of the two DDPM DACs, their integral non-linearity (INL) error

$$INL[n] = \frac{V_{\mathrm{DDPM}}[n] - \frac{V_{\mathrm{DD}}}{2^N} \cdot n}{LSB_8}, \tag{13}$$

where $LSB_8 = V_{\mathrm{DD}}/2^8 = 12.9$ mV is the least significant bit at 8-bit resolution. and the differential nonlinearity, error evaluated as:

$$DNL[n] = INL[n+1] - INL[n] \tag{14}$$

are reported in Fig.14 and Fig.15, respectively. Based on the figures, it can be observed that the maximum INL is below 2.80 (1.64) LSBs for the iterative (optimized) DDPM DAC, whereas the maximum DNL is below 3.53 (1.79) LSBs for the iterative (optimized) DDPM DAC, thus revealing a better static linearity (1.19 LSB better maximum INL and 1.74 LSB

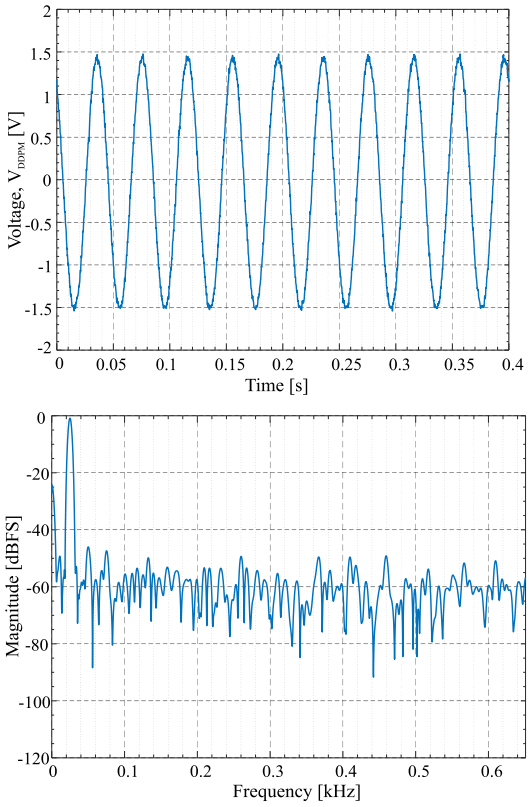


FIGURE 16. Dynamic characterization of the iterative 8-bit DDPM DAC under 25 Hz sine wave input: time-domain output waveform (top) and power spectral density (bottom).

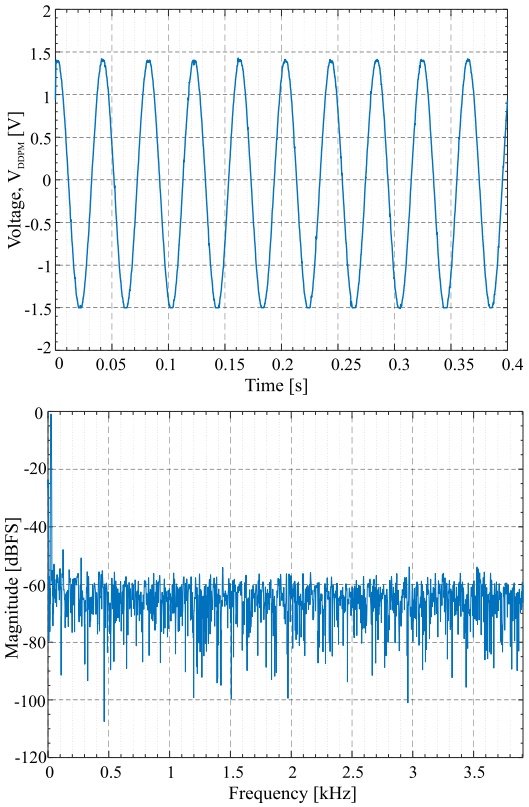


FIGURE 17. Dynamic characterization of the proposed optimized 8-bit DDPM DAC under 25 Hz sine wave input: time-domain output waveform (top) and power spectral density (bottom).

better maximum DNL) for the optimized converter, possibly related to the lower internally generated switching noise due to the simpler implementation.

For dynamic characterization, the input codes $x[n]$ corresponding to a 25Hz sine wave with 90% full-swing amplitude are generated as

$$x[n] = 2^{N-1} + 0.9 \cdot 2^{N-1} \cdot \sin(2\pi n f_o/f_S) \tag{15}$$

where, $f_o = 25$ Hz and n is the discrete-time index, using the sin library function in the C-code.

The time-domain and the frequency-domain DAC outputs under sine wave input are shown in Fig.16 and in Fig.17 for the iterative and for the proposed optimized DDPM DAC. The power spectra (PS) of the measured DAC waveforms are obtained using the `pwelch` MATLAB function. Based on the measured PS, the signal to noise and distortion ratio (SNDR), spurious free dynamic range (SFDR) and effective number of bits (ENOB) are determined by (16), (17) and (18) in what follows. The *SNDR*, in particular, is the ratio of the fundamental signal power to the power of all other spectral components excluding DC, evaluated as

$$SNDR_{\text{dB}} = 10\log_{10}\frac{PS(f_0)}{\sum PS(f)_{|f\neq f_0 \vee f\neq 0}} \tag{16}$$

The SFDR is defined as

$$SFDR_{\text{dB}} = PS_{\text{dBFS}}(f_0) - PS_{\text{dBFS}}(f_n) \tag{17}$$

where $PS_{\text{dBFS}}(f_0)$ is the power of the fundamental and $PS_{\text{dBFS}}(f_n)$ is the power of the largest spurious expressed in decibel with respect to the full swing of the converter (dBFS). Based on the SNDR, the effective number of bit (ENOB) of the converter is finally evaluated as

$$ENOB = \frac{SNDR - 1.76\text{dB}}{6.02\text{dB/bit}} \tag{18}$$

Based on measurements, the SNDR, SFDR and ENOB are evaluated by the above equations for both the proposed and iterative DDPMs are calculated and reported in Table.1. In particular, the SFDR of the iterative and of the proposed optimized SW DDPM DACs are 45.16dB and 47.02dB, respectively, while the measured SNDR is found to be 41.61dB and 45.27dB, corresponding to 6.62 and 7.23 effective bits (ENOB) for the two converters and are therefore comparable. Based on measured results, the optimized DDPM DAC achieves a 6× higher sample rate at 0.61 bit higher effective resolution.

C. DISCUSSION AND COMPARISON

The results of the measurements performed on the microcontroller reported so far are summarized in Table.1, where they are also compared to DDPM and non-DDPM FPGA DAC implementations proposed in the last years.

TABLE 1. Comparison table based on experimental results.

Param.	Unit	Prop.SW DDPM	Iter.SW DDPM	FPGA DDPM	FPGA DPWM	FPGA $\Sigma\Delta$
Pubbl.		This Work	This Work	[1]	[20]	[19]
Res.	bit	8	8	16	8	16
Syst. Clk. f_{CLK}	MHz	150	150	100	25	10
Mod.Clk. f_{DDPM}	MHz	2	0.33	100	N/A	N/A
Sample Rate	kS/s	7.812	1.302	1.525	20	20
Supply	V	3.3	3.3	1.8	± 9	3.3
INL	LSB	1.64	2.80	13	N/A	N/A
DNL	LSB	1.79	3.53	1	N/A	N/A
SNDR	dB	45.27	41.61	N/A	N/A	57.3
SFDR	dB	47.02	45.16	N/A	78	37
ENOB	bit	7.23	6.62	12.1	N/A	9.2
Memory	kB	1638	1720	N/A	N/A	N/A
Res.	kΩ	100	100	180	N/A	0.1
Cap.	nF	1	1	1	N/A	80

Looking at the table, the enhanced SW DDPM DAC proposed in this paper is found to be better in all aspects than the simple iterative SW implementation. The maximum achievable sample rate, in particular, is 7.812 kS/s, i.e. 6X higher than in the iterative SW DDPM DAC implementation which is just 1.302 kS/s, thus significantly widening its application domain. Considering the static characteristics results, the INL peak error of proposed converter is 1.64 LSBs compared to 2.8 LSBs of iterative converter. The better behavior of the proposed DAC is further confirmed by the measured DNL, which is 1.79 LSBs compared to 3.53 LSBs for the iterative implementation, and by dynamic characterization (i.e., SNDR, SFDR, ENOB) which give a clear picture of the improvement. Moreover, the proposed SW DDPM takes less memory and is more efficient than the iterative one.

Compared to FPGA implementations of a DDPM DAC [1] and non-DDPM DACs, like the Digital PWM FPGA DAC in [20] and the FPGA $\Sigma\Delta$ DAC in [19], whose main parameters are reported in Tab.1 for comparison, the proposed DDPM DAC achieves comparable performance in terms of sample rate (5X more than [1], 2.5X less than [19], [20]) and slightly worse effective resolution (2.0-4.8 effective bits less than [1]–[19]). Unlike the other solutions, however, it does not require expensive programmable logic devices and can be easily implemented in software in any general purpose microcontroller unit at a very low cost.

V. CONCLUSION

The hardware and software implementation of Dyadic Digital Pulse Modulators (DDPMs) for Digital to Analog (D/A) conversion has been addressed and an enhanced software DDPM implementation has been proposed and compared with a plain, iterative software transposition of the DDPM hardware architecture. The effectiveness of proposed technique has been demonstrated by software defined 8-bit DDPM DACs described and programmed in C and implemented on a c2000 microcontroller platform.

Based on experimental results, the software-defined DDPM DAC featuring the proposed enhanced DDPM modulator achieves 6X higher maximum sample rate and better static and dynamic performance (+0.61 ENOB) compared to the plain iterative software implementation.

The reported performance are comparable with previously proposed DDPM and non-DDPM DACs implemented on FPGA at a significantly higher cost and system complexity and are compatible with the requirements of many practical applications [23].

REFERENCES

[1] P. S. Crovetti, "All-digital high resolution D/A conversion by dyadic digital pulse modulation," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 64, no. 3, pp. 573–584, Oct. 2017.

[2] O. Aiello, P. Crovetti, and M. Alioto, "Standard cell-based ultra-compact dacs in 40-nm CMOS," *IEEE Access*, vol. 7, pp. 126479–126488, 2019.

[3] O. Aiello, P. Crovetti, and M. Alioto, "Fully synthesizable low-area digital-to-analog converter with graceful degradation and dynamic power-resolution scaling," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 8, pp. 2865–2875, Aug. 2019.

[4] P. Toledo, P. Crovetti, H. Klimach, and S. Bampi, "Dynamic and static calibration of ultra-low-voltage, digital-based operational transconductance amplifiers," *Electronics*, vol. 9, no. 6, p. 983, 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/6/983

[5] P. Toledo, P. Crovetti, O. Aiello, and M. Alioto, "Fully digital rail-to-rail OTA with sub-1000-μm area, 250-mV minimum supply, and nW power at 150-pF load in 180 nm," *IEEE Solid-State Circuits Lett.*, vol. 3, pp. 474–477, 2020.

[6] O. Aiello, P. Crovetti, A. Sharma, and M. Alioto, "Fully-synthesizable current-input ADCs for ultra-low area and minimal design effort," in *Proc. 26th IEEE Int. Conf. Electron., Circuits Syst. (ICECS)*, Nov. 2019, pp. 715–718.

[7] O. Aiello, P. Crovetti, and M. Alioto, "Fully synthesizable low-area analogue-to-digital converters with minimal design effort based on the dyadic digital pulse modulation," *IEEE Access*, vol. 8, pp. 70890–70899, 2020.

[8] P. S. Crovetti, "Spectral characteristics of DDPM streams and their application to all-digital amplitude modulation," *Electron. Lett.*, vol. 57, no. 5, pp. 212–215, Mar. 2021. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/ell2.12050

[9] P. S. Crovetti, M. Usmonov, F. Musolino, and F. Gregoretti, "Limit-cycle-free digitally controlled DC–DC converters based on dyadic digital PWM," *IEEE Trans. Power Electron.*, vol. 35, no. 10, pp. 11155–11166, Oct. 2020.

[10] M. Usmonov, P. S. Crovetti, F. Gregoretti, and F. Musolino, "Suppression of quantization-induced limit cycles in digitally controlled DC–DC converters by dyadic digital pulse width modulation," in *Proc. Energy Convers. Congr. Expos. (ECCE)*, 2019, pp. 2224–2231.

[11] *Circuit Realization at Faster Timescales (Craft) Project, Defense Advanced Research Projects Agency (DARPA)*. Accessed: Sep. 30, 2010. [Online]. Available: https://www.darpa.mil/program/circuit-realization-at-faster-timescales

[12] P. Toledo, R. Rubino, F. Musolino, and P. Crovetti, "Re-thinking analog integrated circuits in digital terms: A new design concept for the IoT era," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 3, pp. 816–822, Mar. 2021.

[13] P. Crovetti, F. Musolino, O. Aiello, P. Toledo, and R. Rubino, "Breaking the boundaries between analogue and digital," *Electron. Lett.*, vol. 55, no. 12, pp. 672–673, 2019. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/el.2019.1622

[14] P. S. Crovetti, R. Rubino, and F. Musolino, "Relaxation digital-to-analog converter with foreground digital self-calibration," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, Oct. 2020, pp. 1–5.

[15] R. Rubino, P. S. Crovetti, and F. Musolino, "FPGA-based relaxation D/A converters with parasitics-induced error suppression and digital self-calibration," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 6, pp. 2494–2507, Jun. 2021.

[16] O. Aiello, P. Crovetti, and M. Alioto, "A sub-leakage pw-power hz-range relaxation oscillator operating with 0.3 v-1.8 v unregulated supply," in *Proc. IEEE Symp. VLSI Circuits*, Jun. 2018, pp. 119–120. [Online]. Available: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85056811140&doi=10.1109%2fVLSIC.2018.8502413&partnerID=40&md5=bafbe6c61e20828c91bf177d52ec7aa1
[17] P. S. Crovetti, "A digital-based analog differential circuit," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 12, pp. 3107–3116, Dec. 2013.
[18] P. S. Crovetti, "A digital-based virtual voltage reference," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 62, no. 5, pp. 1315–1324, May 2015.
[19] B. L. Moffo and J. Mbihi, "A novel digital duty-cycle modulation scheme for FPGA-based digital-to-analog conversion," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 62, no. 6, pp. 543–547, Jun. 2015.
[20] *Integrated DAC for Altera Cyclone-IV Devices*, Missing Link Electronics, 2011. [Online]. Available: https://www.missinglinkelectronics.com/devzone/files/papers/MLE-TB20110426.pdf
[21] Texas Instruments. (Jun. 2010.) *Peripheral Explorer Kit Overview*. [Online]. Available: https://www.ti.com/lit/ml/sprugm2a
[22] K. Ireland and M. Rosen, *A Classical Introduction to Modern Number Theory*. New York, NY, USA: Springer, 2000.
[23] M. Alioto, *Enabling the Internet of Things—From Integrated Circuits to Integrated Systems*, 1st ed. Cham, Switzerland: Springer, 2017.



AHMED ABDULLAH was born in Abbottabad, Pakistan, in 1993. He received the Laurea degree in electronic engineering from the Politecnico di Torino, Torino, Italy, in 2020, where he is currently pursuing the Ph.D. degree with the Department of Electronics and Telecommunications (DET). He is also a Researcher at the DET, Politecnico di Torino. His research interests include electronics for power conversion, digital control of power converters, and analysis and experimental characterization of various high-speed digital to analog converters used as digital pulse width modulators.



FRANCESCO MUSOLINO (Member, IEEE) was born in Torino, Italy, in 1972. He received the Laurea and Ph.D. degrees in electronic engineering from the Politecnico di Torino, Torino, in 1999 and 2003, respectively. He is currently a Researcher with the Department of Electronics and Telecommunications (DET), Politecnico di Torino, where he teaches courses on power electronics and electronics for electric drives. His research interests include electronics for power conversion and motor drive applications, mixed-signal circuits, electromagnetic compatibility at the system levels, and the analysis, modeling, and experimental characterization of electromagnetic compatibility problems at the printed circuit board and package level.



PAOLO CROVETTI (Senior Member, IEEE) was born in Turin, Italy, in 1976. He received the Laurea *(summa cum laude)* and Ph.D. degrees in electronic engineering from the Politecnico di Torino, Turin, in 2000 and 2003, respectively. He is currently an Associate Professor with the Department of Electronics and Telecommunications (DET), Politecnico di Torino, where he leads a research group in analog, mixed-signal and power microelectronics with several international collaborations and teaches courses on basic and automotive electronics. He has coauthored more than 75 papers appearing in journals and international conference proceedings. His main research interests include integrated circuit design and electromagnetic compatibility. His recent research activities are focused on non-conventional digital-based information processing techniques and ultra-low-voltage, ultra-low-power IC design for the Internet of Things. In 2009 and 2019, he was a co-recipient of the Excellent Paper Award of the EMC 2009 Kyoto Symposium and the Best Student Paper Award of the International Symposium of Circuits and Systems ICECS 2019. He is the Subject Editor-in-Chief of *IET Electronics Letters* in the area of circuits and systems. He is an Associate Editor of the IEEE TRANSACTIONS ON VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS.