



Repositorio Institucional de la Universidad Autónoma de Madrid

<https://repositorio.uam.es>

Esta es la **versión de autor** de la comunicación de congreso publicada en:
This is an **author produced version** of a paper published in:

3rd Southern Conference on Programmable Logic, SPL 2007, IEEE, 2007. 137-142.

DOI: <http://dx.doi.org/10.1109/SPL.2007.371737>

Copyright: © 2007 IEEE

El acceso a la versión del editor puede requerir la suscripción del recurso
Access to the published version may require subscription

HIGH RESOLUTION PULSE WIDTH MODULATORS IN FPGA

Angel de Castro, Gustavo Sutter

Digital Systems Lab
Universidad Autónoma de Madrid
email: {angel.decastro, gustavo.sutter}@uam.es

Santa C. Huerta, José A. Cobos

División de Ingeniería Electrónica
Universidad Politécnica de Madrid
email: conihuerta@etsii.upm.es, ja.cobos@upm.es

ABSTRACT

Pulse Width Modulation (PWM) is a very common technique used in different applications, from the control of motors, switching power converters (power supplies), audio amplifiers or illumination systems. In some of those applications, the pulse frequency has increased so much in the last years that the resolution obtained with classical (counter) techniques is not enough. This paper explains some methods used for increasing the resolution of PWMs and proposes a new method based on the resources available in almost every FPGA nowadays.

1. INTRODUCTION

Pulse Width Modulators (PWMs) are very usual and spread modules, as many applications need them. In fact, they are so usual that, for instance, almost every microcontroller includes one or more PWMs nowadays. The digital version of these blocks, or Digital Pulse Width Modulators (DPWMs), allow obtaining a range of mean output voltages (or currents) using a digital output that only has two possible values: high or low. In order to do so, they use a digital word-to-time conversion. In the most common version, the output is a pulse of fixed frequency in which the on-time (width) is changed (modulated) according to the control value. In this case (see Fig. 1), the duty cycle, d , is defined as:

$$d = \frac{t_{on}}{T}, \quad (1)$$

where t_{on} is the on-time and T is the pulse frequency. The mean value in the output is:

$$V_{out} = d \cdot V_{on}, \quad (2)$$

where V_{on} is the voltage in the output during the on-time and supposing that the output is at 0 V during the off-time. If the off-time voltage is different from 0 V, another expression must be used:

$$V_{out} = d \cdot V_{on} + (1 - d) \cdot V_{off}, \quad (3)$$

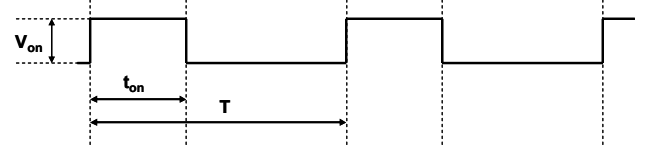


Fig. 1. PWM output.

However, (2) is the most common case and it will be used from here on for the sake of simplicity.

In the case of DPWMs, not all d values between 0 and 1 are possible, but a finite number of them. The number of possible d solutions gives its resolution, which can be measured in percentage steps or in on-time steps. In some cases, the resolution is measured in number of bits, n , assuming that there are 2^n possible d values. A high resolution implies small percentage steps, small on-time steps or high number of bits.

It is obvious that the design objective is to obtain small on-time steps (Δt_{on}), as the percentage steps (Δd) are not a comparison criterion, as they depend on the pulse frequency (or period) according to:

$$\Delta d = \frac{\Delta t_{on}}{T}, \quad (4)$$

Nowadays, many applications demand high PWM frequencies. For instance, many switching power supplies are now working at frequencies in the range of 1 to 10 MHz [1-3], while frequencies in the order of hundreds of MHz are starting to be considered [4]. As the frequency raises, T decreases and, taking (4) into account, Δd becomes coarser (lower resolution). As an example, for a 10 MHz application and 8 bits of resolution, using a counter-based DPWM (see next section) would need a 2.56 GHz clock. This is unthinkable for low-cost applications. A number of solutions have appeared to increase the DPWM resolution, which are summarized in section 3.

This paper proposes a very simple method that achieves resolutions comparable or even higher than state-of-the-art methods. This method is based on the FPGA resources, and is proposed in section 4. Experimental results are shown in section 5.

2. COUNTER-BASED DPWM

The most common DPWMs are based on a counter and comparison block. The duty cycle command, d , received at the DPWM is compared to the value of a cyclic counter (equivalent to the saw-tooth signal of analog PWMs). Both d and the counter have the same number of possible values, N . If d is over the counter value, the output is in the on-state, turning off when the counter reaches d . This is shown in Fig. 2. If the counter is upwards, as in Fig. 2, it is called a *trailing-edge* PWM, in which the rising edges of the output are in fixed instants while the falling edges depend on the value of d . If the counter is downwards, it would be a *leading-edge* PWM (fixed falling edges and variable rising edges). The third option is the *triangle-wave* PWM, in which the counter goes up and then down. A period then includes both the upwards and downwards cycle of the counter, and both the rising and falling edges are variable. Anyhow, the three options lead to the same duty cycle and all of them are fixed-frequency methods. From here on, the *trailing-edge* method will be used, but the results are applicable to the three methods.

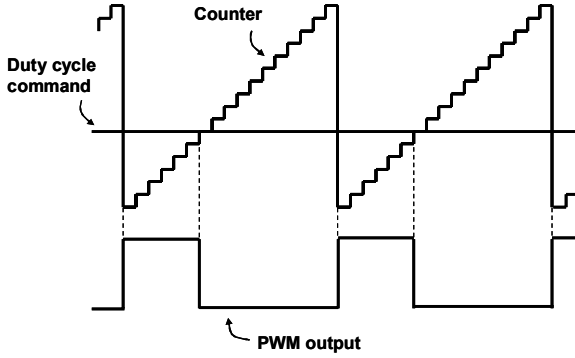


Fig. 2. Counter-based DPWM (*trailing-edge* method).

The output period of the PWM output is:

$$T = N \cdot T_{clk}, \quad (5)$$

where N is the counter range and T_{clk} the clock period.

As T_{clk} is also Δt_{on} , using (4) we obtain that $\Delta d = 1/N$. Therefore, the resolution is higher (smaller Δd) with higher values of N .

However, T is usually imposed by the application, so in order to increase N (and the resolution) we have to use a faster clock (lower T_{clk}). In fact, this is clear if we measure the resolution in terms of on-time steps, as $\Delta t_{on} = T_{clk}$. As a conclusion, the faster the clock, the higher the resolution.

As a counter-based DPWM is a very simple hardware block (counter and comparator, see Fig. 3) it can work at high clock frequencies. In fact, it can usually work at a higher frequency than the rest of the modules. We can take advantage of the FPGA resources in order to feed the DPWM with a high clock frequency while maintaining a lower clock frequency for the rest of the design.

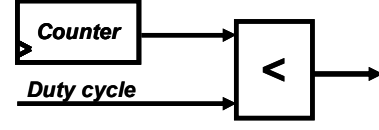


Fig. 3. DPWM hardware structure.

In order to do so, the internal FPGA DLLs can be used. These blocks allow multiplying or dividing the clock frequency in order to obtain a second clock frequency (see Fig. 4). For instance, in the experimental results (section 5) an external 32 MHz clock has been used. This relatively slow clock is also used for the rest of the blocks in order to decrease power consumption and to facilitate their design. However, it is multiplied by 4 (128 MHz) in order to feed the DPWM, so a higher resolution can be obtained.

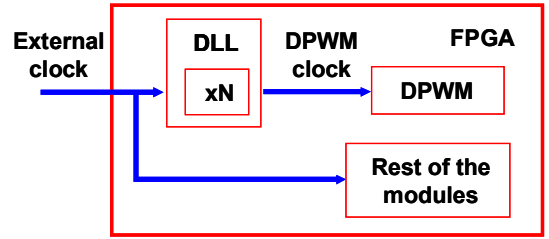


Fig. 4. Using DLLs for multiplying the clock frequency sent to the DPWM and allow a slower clock in the rest of the modules.

Finally, there are some cases in which even a fast clock is not enough because the maximum counter-based DPWM frequency does not achieve the required resolution. In those cases, resolution can be further increased using asynchronous design techniques based on delays. These are described in the following section.

3. DELAY-BASED AND HYBRID DPWM

In order to obtain higher resolution, delay-based DPWMs have been developed. The idea is that the minimum Δt_{on} obtained in counter-based DPWMs is equal to T_{clk} . However, a smaller Δt_{on} can be obtained using a delay line (see Fig. 5) [5-9]. In that case, Δt_{on} is equal to the delay of a single delay element, which is usually implemented with two inverters. If the DPWM has n bits of resolution, 2^n delay elements are necessary, and also a $2^n:1$ multiplexer. The first obvious disadvantage of these DPWMs is that they need much more area than a counter-based DPWM, especially for a high number of bits. The counter-based DPWM needs an n -bits counter (n FFs) and an n -bits comparator, while the delay-based DPWM needs 2^n delay elements and a $2^n:1$ multiplexer. However, power consumption is lower in delay-based DPWMs because there is a single delay element switching in each moment, while a high frequency clock must be used in counter-based DPWMs.

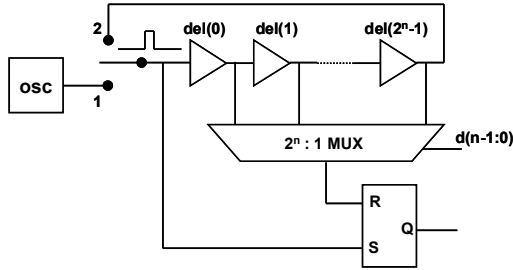


Fig. 5. Delay line-based DPWM.

Other disadvantages of delay-based DPWMs are decreased linearity and even non-monotonic behavior. Clock signals have very accurate periods, so counter-based DPWMs have very similar increments from one duty cycle value to the next one (high linearity). However, these increments can be more variable from one delay element to other one due to small physical differences and, even more important, due to unbalanced route paths to the multiplexer. If a large number of delay elements are used (1024 for a 10-bits DPWM), these routing delay differences can be even higher than the delay of a delay element. In that case, non-monotonic behavior can be found, that is, an increment in the duty cycle command can produce a decrease in the pulse width output (see Fig. 6).

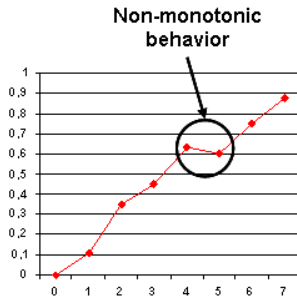


Fig. 6. Non-monotonic behavior in a DPWM.

Other important concern about delay-based DPWMs is the output frequency. If an oscillator is used in order to create the pulse that is propagated through the delay line (option 1 in Fig. 5), the output frequency is constant. In that case, the total delay of the delay line should be equal to the oscillator period. However, due to fabrication mismatches and/or operation mismatches (i.e. temperature or supply voltage variations), the total delay can be different from the oscillator period. If it is shorter, not all the range of duty cycle commands is covered. If it is longer, high duty cycle commands produce totally wrong outputs (small duty cycles). In order to avoid these disadvantages, the delay of each element should be adjustable (see Fig. 7) or option 2 of Fig. 5 can be used. In the latter case, the problem is that the output frequency is not constant.

Taking into account all the delay-based DPWMs disadvantages, hybrid counter-delay DPWMs have appeared [10-11] trying to obtain the advantages of both, mainly high resolution of delay-based DPWMs and higher

linearity and monotonic behavior of counter-based DPWMs. Regarding area and power consumption, a trade-off can be reached moving the number of bits in each part of the DPWM. The basic structure of hybrid DPWMs is shown in Fig. 7. The total number of bits ($n=l+m$) is divided in l bits for a counter and m bits for the delay line. Linearity is almost perfect in the counter, so it is only decreased by the delay part. However, as a smaller number of bits is used in the delay part, there are only 2^m delay elements. For instance, for a 5+5 bits hybrid DPWM, 32 delay elements instead of 1024 are necessary. In that way, the routing differences are much smaller, increasing linearity and avoiding non-monotonic behavior.

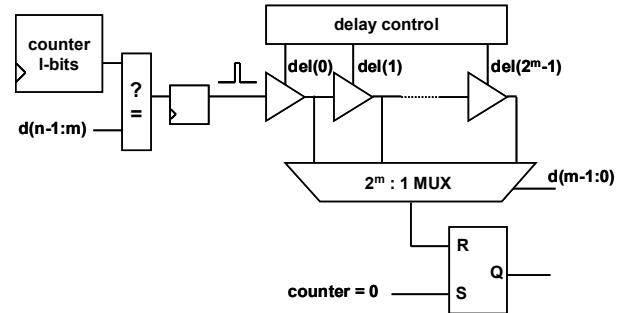


Fig. 7. Hybrid DPWM.

In hybrid DPWMs, the total delay of the delay line should be equal to one clock cycle. If not, the same problems of case 1 in Fig. 5 take place. The solution is to use delay-adjustable elements, controlling the individual delay of each element with an additional block (delay control of Fig. 7). This can be implemented using variable number of pairs of inverters (i.e. each delay element can include the delay of 2, 4 or 6 inverters). Given that in the general case, not all the delay elements will use the same number of inverters, linearity is somehow decreased, but monotonic behavior is still obtained. In ASIC solutions, it is also possible to control the supply voltage of the delay elements to control its delay, but this is not possible in FPGA solutions.

Finally, a trade-off between area and power consumption exists, depending on the number of bits of the counter and delay parts. More bits in the counter leads to less area but more power, and also to higher linearity.

4. PROPOSED HYBRID DPWM

Delay-based and hybrid DPWMs shown in the previous section are valid for both ASIC and FPGA implementations. However, using an FPGA makes the minimum delay element to be a LUT, whose delay is higher than that of two inverters, so lower resolution can be achieved in an FPGA than in an ASIC. Furthermore, routing differences can be considerably higher in FPGAs because distances are longer.

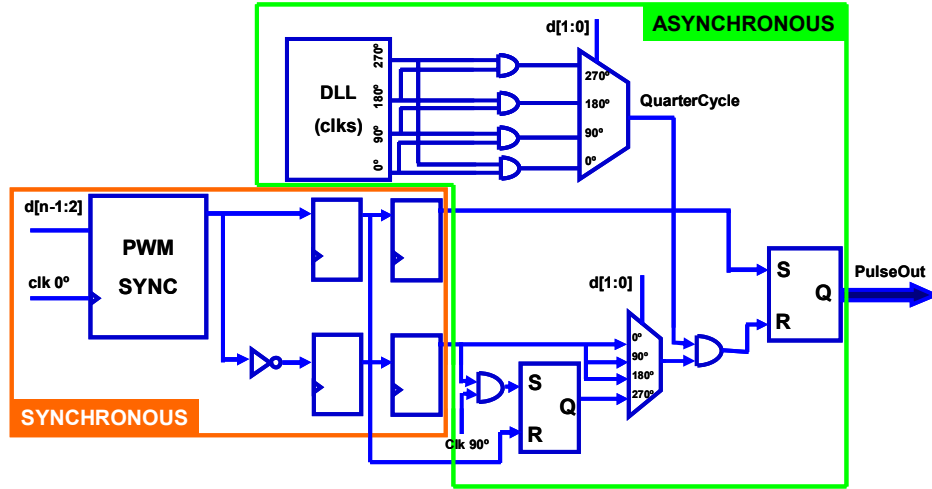


Fig. 8. Proposed hybrid DPWM.

Avoiding the delay elements would also avoid these problems, but a counter DPWM achieves lower resolution. This paper proposed a new hybrid DPWM architecture without delay elements, but with increased resolution. The proposed DPWM is suitable for FPGA implementation, as it uses resources available in almost all FPGAs.

The key element for the proposed DPWM is the DLL (or PLL) available in almost all FPGAs. Many of the available DLLs generate 4 phase-shifted clocks (shifted 0°, 90°, 180° and 270°) directly (Spartan and Virtex families of Xilinx) or allow to generate phase-shifted versions of the clock (Cyclone and Stratix families of Altera, ProAsic3, Fusion or Axcelerator families of Actel). Using 4 phase-shifted clocks allows multiplying the DPWM resolution by 4 (2 bits) without using delay elements. The idea is to use a counter DPWM with the fastest possible clock and then increase the resolution 2 additional bits using the 4 phase-shifted clocks.

The proposed DPWM architecture is shown in Fig. 8. As it can be seen, the synchronous block is a counter-based DPWM that uses the 0° shifted clock. It receives the $n-2$ MSBs of the duty cycle command and its output is then used in the asynchronous block, which uses the four phase-shifted clocks. The asynchronous block generates a signal named *QuarterCycle* that is active a quarter of the clock cycle starting in the rising edge of one of the four clocks, depending on the value of the 2 LSBs of the duty cycle command (see Fig. 9).

The working of the proposed DPWM is the following. The output of the synchronous block sets the output of the proposed hybrid DPWM, so it always rises in a rising edge of the 0° clock, lasting for at least the number of clock cycles determined by the MSBs of d . The output resets when the synchronous output is off and *QuarterCycle* arrives. This would be enough if *QuarterCycle* had no

delay. However, its value corresponding to the 270° clock also has a short on-time during the 0° quarter because of the delay (see Fig. 9). This would lead to non-monotonic behavior (duty cycle commands ending in “11” would have similar values to those ending in “00”). In order to avoid this problem, the reset of the output when d ends in “11” is only allowed after the arrival of the 90° clock. That is the reason for including an additional RS register and multiplexer, as shown in Fig. 8.

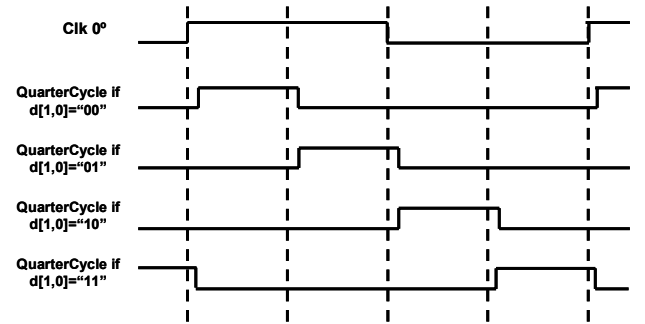


Fig. 9. *QuarterCycle* signal depending on the value of $d[1,0]$.

As a conclusion, the proposed DPWM output lasts for a number of cycles indicated by the $n-2$ MSBs of d plus the number of quarter cycles indicated by the 2 LSBs. In this way, the resolution is 4 times higher than that of only counter-based DPWMs, while no delay line is necessary. Therefore, high resolution is achieved, but also with high linearity and monotonic behavior.

5. EXPERIMENTAL RESULTS

The proposed DPWM architecture has been synthesized in a Spartan 3 FPGA [12], from Xilinx. The exact device has been a XC3S200-4FT256. A Spartan 3 has been chosen in order to show that high resolution is possible with low cost FPGAs.

An external 32 MHz clock has been used, multiplying it by 4 (128 MHz) with the internal DLLs (called Digital Clock Managers or DCMs in this family). These DCMs also generate the four phase-shifted clocks. The obtained resolution is one quarter of the 128 MHz clock, that is, 1.95 ns. However, the resolution could have been somewhat higher. The maximum clock frequency using XST (Xilinx Synthesis Tool) [13] and Xilinx ISE v8.1 [14] was 171.2 MHz, which would lead to a resolution of 1.46 ns. Of course, higher resolutions can be achieved using faster devices, such as the -5 version of the same device or Virtex 4 devices [15]. A summary of the resolutions that can be achieved with different devices is shown in Table 1.

Table 1. Resolution of the proposed DPWM for different FPGAs.

Device	Max. freq. (MHz)	Resolution (ns)
Spartan 3 (slow) XC3S200-4FT256	171.2	1.460
Spartan 3 (fast) XC3S200-5FT256	196.3	1.274
Virtex 2 (slow) XC2V40-4CS144	147.2	1.698
Virtex 2 (fast) XC2V40-6CS144	206.7	1.209
Virtex 4 LX (slow) XC4VLX15-10SF363	253.5	0.986
Virtex 4 LX (fast) XC4VLX15-12SF363	331.3	0.754

The obtained resolution is comparable or even higher than those of other state-of-the-art DPWMs based on delay lines [5-11]. Three other DPWMs have been tested in FPGAs. In [8], a 2.08 ns resolution is obtained (the exact FPGA is not mentioned), in [10] they obtain 1.25 ns of resolution with a Virtex 4 LX25 device and in [11] the resolution is 0.97 ns using a Virtex 2-3000 device, but without controlling the delay in each delay element (see section 3).

Apart from the high resolution of the proposed method, other important advantage is its high linearity and monotonic behavior. This has been tested in post-route simulations, but also with experimental results that are shown below. For these experiments, an 11-bits DPWM has been built (9 synchronous bits and 2 asynchronous bits). Given that the internal clock frequency was 128 MHz, the output pulses frequency is $128/2^9$ MHz, that is, 250 kHz. Small duty cycles values have been measured in order to set the oscilloscope (Tektronix TDS 5054, 500 MHz bandwidth, 5 GS/s) at its maximum time resolution, so the

complete on-time can be measured (see Fig. 10 and 11). Even doing so, the limited sampling speed of the oscilloscope causes differences in the on-time value from measurement to measurement. In order to decrease this effect, 10 measurements have been taken for each duty value. The mean values of the 10 measurements are presented in Table 2.

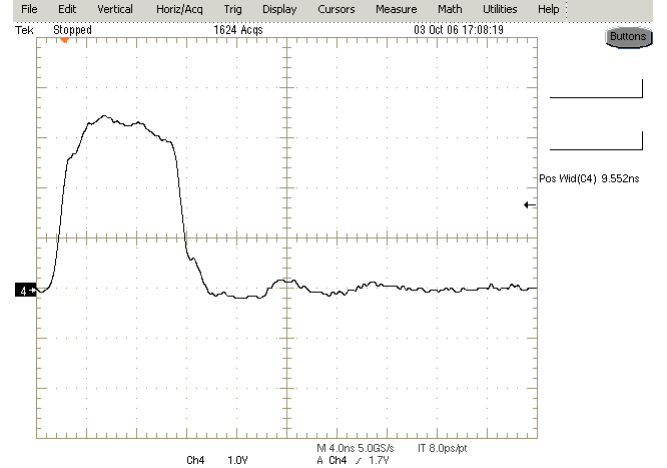


Fig. 10. DPWM output when $d='00000000100'$.

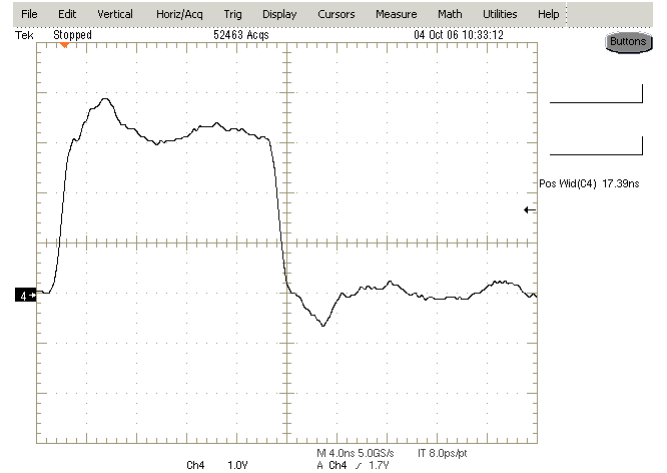


Fig. 11. DPWM output when $d='00000001000'$.

Table 2. Experimental on-time measurements for different d values.

Duty value	Mean t_{on} (ns)	Δt_{on} (ns)
00000000100	9.524	
00000000101	11.178	1.654
00000000110	13.123	1.945
00000000111	15.123	2.000
00000001000	17.439	2.316
00000001001	18.941	1.502
00000001010	20.866	1.925
00000001011	22.868	2.002
00000001100	25.226	2.358

The expected Δt_{on} was 1.953 ns. Small differences around this value have been found in the experimental results (and post-route simulations) due to differences in the delay of the path for each value of $d[1:0]$ (four different asynchronous paths). However, great repeatability is found every four duty values, when $d[1:0]$ is repeated. This is a consequence of the great accuracy of the synchronous part of the design.

6. CONCLUSIONS

A novel high resolution DPWM architecture has been proposed. This DPWM is thought for FPGA implementation, as it takes advantage of the DLLs available in almost every FPGA nowadays, which generate 4 phase-shifted clocks. Using these resources, the resolution of counter-based DPWMs can be multiplied by 4.

The proposed DPWM has been explained in detail, and experimental results are presented. A resolution under 2 ns has been demonstrated using a Spartan 3 FPGA, while it can be increased under 1 ns without changes using faster devices such as Virtex 4. High linearity and monotonic behavior are other advantages of the proposed DPWM, apart from its simplicity.

7. REFERENCES

- [1] B.J. Patella, A. Prodic, A. Zirger, D. Maksimovic, "High-frequency digital PWM controller IC for DC-DC converters", *IEEE Trans. Power Electronics*, vol. 18, no. 1, pp. 438–446, Jan. 2003.
- [2] A.V. Peterchev, J. Xiao, S.R. Sanders, "Architecture and IC implementation of a digital VRM controller", *IEEE Trans. Power Electronics*, vol. 18, no. 1, pp. 356–364, Jan. 2003.
- [3] T. Takayama, D. Maksimovic, "Digitally controlled 10 MHz monolithic buck converter", *Proc. 10th IEEE Computers in Power Electronics Workshop, COMPEL*, July 2006.
- [4] G. Schrom, P. Hazucha, J. Hahn, D.S. Gardner, B.A. Bloechel, G. Dermer, S.G. Narendra, T. Karnik, V. De, "A 480-MHz, multi-phase interleaved buck DC-DC converter with hysteretic control", *Proc. IEEE 35th Annual Power Electronics Specialists Confereren, PESC*, vol. 6, June 2004, pp. 4702–4707.
- [5] A.P. Dancy, A.P. Chandrakasan, "Ultra low power control circuits for PWM converters", *Proc. IEEE 28th Annual Power Electronics Specialists Confereren, PESC*, vol. 1, June 1997, pp. 21–27.
- [6] A. Syed, E. Ahmed, D. Maksimovic, E. Alarcon, "Digital pulse width modulator architectures", *Proc. IEEE 35th Annual Power Electronics Specialists Confereren, PESC*, vol. 6, June 2004, pp. 4689–4695.
- [7] E. O'Malley, K. Rinne, "A programmable digital pulse width modulator providing versatile pulse patterns and supporting switching frequencies beyond 15 MHz", *Proc. 19th Annual IEEE Applied Power Electronics Conference and Exposition, APEC*, vol. 1, Feb. 2004, pp. 53–59.
- [8] Z. Lukic, K. Wang, A. Prodic, "High-frequency digital controller for dc-dc converters based on multi-bit SigmaDelta pulse-width modulation", *Proc. 20th Annual IEEE Applied Power Electronics Conference and Exposition, APEC*, vol. 1, March 2005, pp. 35–40.
- [9] K. Wang, N. Rahman, Z. Lukic, A. Prodic, "All-digital DPWM/DPFM controller for low-power DC-DC converters", *Proc. 21st Annual IEEE Applied Power Electronics Conference and Exposition, APEC*, March 2006, pp. 719–723.
- [10] V. Yousefzadeh, T. Takayama, D. Maksimovic, "Hybrid DPWM with Digital Delay-Locked Loop", *Proc. 10th IEEE Computers in Power Electronics Workshop, COMPEL*, July 2006.
- [11] R.F. Foley, R.C. Kavanagh, W.P. Marnane, M.G. Egan, "An area-efficient digital pulsewidth modulation architecture suitable for FPGA implementation", *Proc. 20th Annual IEEE Applied Power Electronics Conference and Exposition, APEC*, vol. 3, March 2005, pp. 1412–1418.
- [12] Xilinx Inc, DS099 Product Specification, Spartan-3 FPGA Family: Complete Data Sheet, available at www.xilinx.com, April 2006.
- [13] Xilinx Inc, XST User Guide 8.1i, available at www.xilinx.com.
- [14] Xilinx Inc, Xilinx ISE 8 Software Manuals and Help - PDF Collection, available at www.xilinx.com, 2005.
- [15] Xilinx Inc, DS112 Preliminary Product Specification, Virtex-4 Family Overview, available at www.xilinx.com, Oct. 2006.