FPGA-Based Digital Pulse Width Modulator With Optimized Linearity

Martin Scharrer, Mark Halton and Tony Scanlan
Dept. Electronic and Computer Engineering,
University of Limerick
Limerick, Ireland
Email: martin.scharrer@ul.ie

***Abstract*—This paper proposes a new FPGA based architecture for digital pulse width modulators which takes advantage of dedicated digital clock manager (DCM) blocks present in modern FPGAs and applies manual placement techniques to match internal delays for high linearity.**

The proposed hybrid DPWM uses a synchronous counter-based coarse-resolution block and an DCM based fine-resolution block implementing a synchronous delay line.

The design was successfully implemented on a low-cost Xilinx Spartan-3 FPGA with 9-bit resolution with a switching frequency of 1 MHz. Linearity was manually optimized using the presented technique which reduced the integral non-linearity error by 0.5 LSB.

I. Introduction

Recent years has seen increased interest in digital control of switched mode power supplies (SMPS) [1–3]. A key reason for this is that digital control has a number of advantages over analog control: such as programmability, reduced sensitivity to external influences, the use of adaptive or other advanced control algorithms along with simpler implementations and prototyping.

A digital controller uses a digital pulse width modulator (DPWM) to generate the control signals for the power supply switches. A sufficiently high resolution of the DPWM is critical for the stability of the output voltage. Conversely, a DPWM resolution that is lower than an ADC resolution leads to limit cycling [4, 5].

While DPWMs can be simply implemented using a counter and a comparator, this design leads to unreasonably high clock demands for higher resolutions. In order to implement a N-bit DPWM for a switching frequency of f_{sw}, a clock frequency of $2^N f_{\text{sw}}$ is needed. For modern switching frequencies e.g. 1 MHz, a 10-bit counter-based DPWM would have to be clocked at over 1 GHz which can cause design difficulties and increased power consumption.

To overcome this problem different architectures have been proposed and implemented in ICs [6–8]. A common architecture is the use of an asynchronous delay line in combination with a large multiplexer (MUX). Both can be combined as a hybrid architecture using a counter for the coarse and an additional short delay line for the fine resolution [9]. This architecture suffers from the difficulty of matching the delay times with the counter period. All delay times must be identical and fit exactly between two counter time steps in order to avoid non-linear errors and non-monotonic behavior.

While most DPWM implementations target application specific integrated circuits (ASICs), practicing engineers extensively utilize field programmable gate arrays (FPGAs) to prototype and validate their designs. DPWM implemented on FPGAs [10–14] offer the possibility for easy and fast prototyping of SMPSs but the use of pre-existing logic cells and automatic place & route decreases the layout freedom and makes the accurate implementation of the delay line a challenge.

II. Proposed DPWM Architectures

This paper proposes a new FPGA based architecture for digital pulse width modulators which takes advantage of dedicated digital clock manager (DCM) blocks present in modern FPGAs.

A DCM as implemented in Xilinx FPGAs is a dedicated configurable all-digital block which can be used to multiply or divide an input clock and to produce multiple phase-shifted versions of it. The feature range includes: multiply by 2, multiply by $^M/_N$ where $M = 2..32, N = 1..32$ and four phase output with 0°, 90°, 180° and 270° phase shifts. In addition, DCMs can be concatenated to produce combinations, e.g. multiply by $^3/_2$ with four phases.

The proposed hybrid architecture is shown in Figure 1 and contains a coarse resolution counter-comparator DPWM stage with the bit width of N_{counter}. This uses the most significant bits (MSBs) of the N_{total}-bit wide duty cycle as input where $N_{\text{counter}} = N_{\text{total}} - 2$. These MSBs are compared against the counter value. The output pulse of this first stage is delayed using the three phase-shifted clocks from the DCM. The two least significant bits (LSBs) of the duty cycle are used to select one of these four pulses. The final pulse is generated by a set/reset flip-flop (SR-FF) with the coarse pulse used as dominant *set* input and the selected delayed pulse as negative *reset* input.

A. Coarse DPWM

The coarse DPWM shown in Figure 1 consists of the following blocks:

1) Counter: A simple synchronous counter with N_{counter} bits which counts up at every positive clock edge is implemented using a N_{counter} bit wide register and an adder.

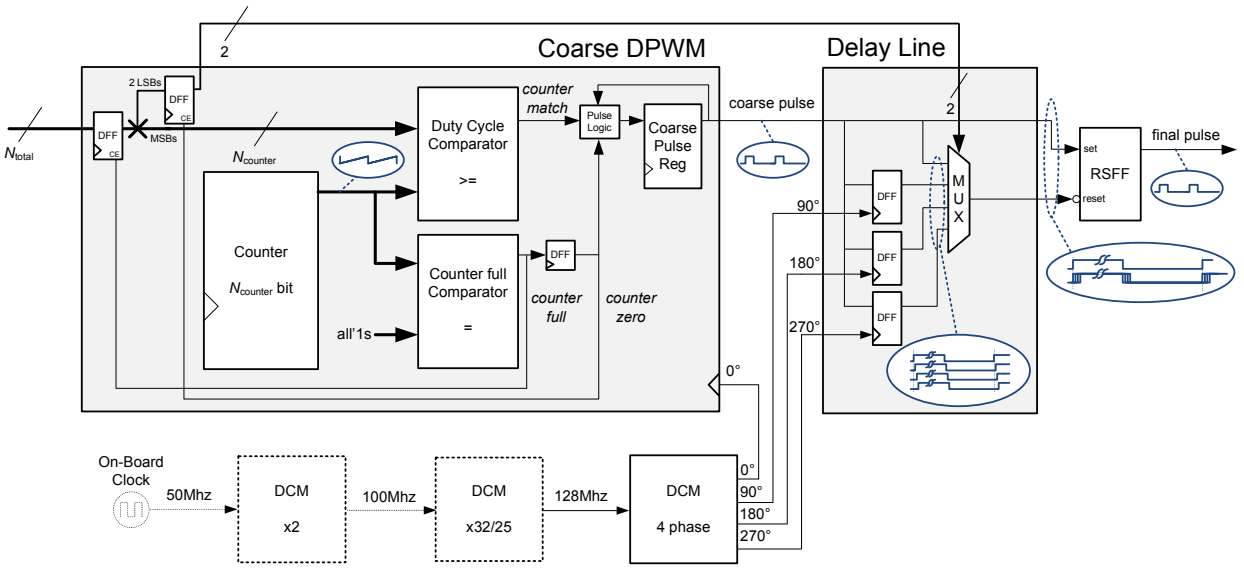

Fig. 1. Proposed FPGA based DPWM architecture.

2) Comparators: Two comparators with the same bit width as the counter are included. The first produces a *counter match* signal which indicates if the counter is greater or equal to the N_{counter} MSBs of the duty cycle. The second comparator produces a *counter full* signal which indicates if the counter is full, i.e. all-1s, and will overrun to zero at the next clock edge. A *counter zero* signal is generated by delaying the *counter full* signal by one clock cycle using a DFF.

3) Interface Registers: The duty cycle which is calculated outside the DPWM by e.g. a digital signal processor must be held constant during the full pulse length and only be updated directly before the pulse start. This is ensured by the use of interface registers which are only clocked when the *counter full* signal is high.

The two LSBs of the duty cycle which are used as select signal of the MUX need to be delayed by another clock edge to synchronise them with the coarse pulse register.

4) Coarse Pulse Logic and Register: The coarse pulse register is controlled by the *counter zero*, *counter match* and the previous register value where the following logic applies:

IF *counter match* == HIGH:
    coarse pulse reg <= LOW
ELSE IF *counter zero* == HIGH:
    coarse pulse reg <= HIGH
ELSE
    coarse pulse reg <= *coarse pulse reg*

This can be implemented with the following assignment:

coarse pulse reg = NOT *counter match*
AND (*counter zero* OR *coarse pulse reg*) .

B. DCM Based Delay Line

The proposed fine resolution block as shown in Figure 1 consist of three data flip-flops (DFF) and a 4-to-1 MUX which is controlled by the two LSBs of the duty cycle. The coarse pulse generated by the coarse DPWM is registered by the DFFs at 90°, 180° and 270° of the counter clock (0°) and is therefore delayed by ¼, ½ and ¾ of the clock period, respectively.

All four resulting pulses are then fed to the MUX, the output of which is taken as the negative reset signal for the final output set/reset FF. The original coarse pulse is used as the dominant set signal. This sets the final DPWM pulse high as long as the coarse pulse is high. It is only reset to low after the selected delayed pulse goes low. This widens the pulse by the appropriate clock period fraction.

For illustration purposes the clocks and pulses for an example duty cycle of $D = 19/512$ are shown in Figure 2. It can be seen that the fine pulse goes high simultaneously with the coarse pulse but is set low simultaneously with the third delayed pulse.

Because the phase relationship between these four clocks is constant, the signals are synchronous to each other, leaving the MUX propagation delay and the routing delay as the only other timing influences. This has many advantages in terms of linearity and monotony over an asynchronous delay line implemented in ASICs using inverter chains.

Monotony of this architecture is guaranteed as long as the routing delays of the pulses do not differ by more than a

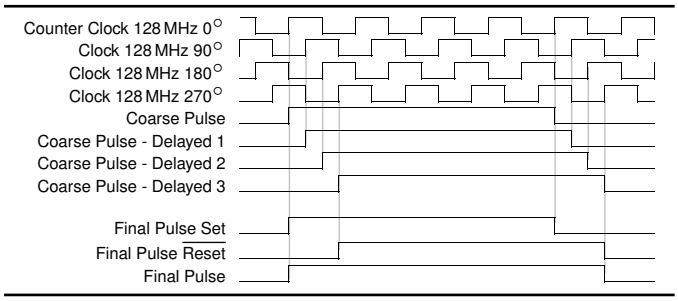

Fig. 2. Pulse signals for D=19/512 (4 coarse steps + 3rd delay tap selected).

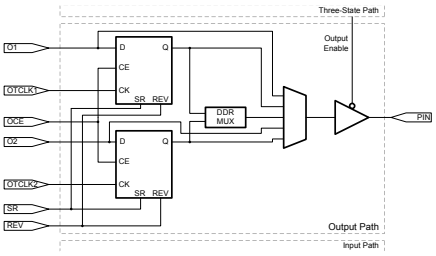

Fig. 3a. Input/Output Block (IOB) of a Xilinx Spartan 3 FPGA [15].

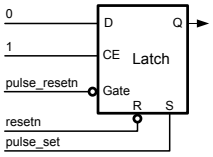

Fig. 3b. Resulting IOB SR-FF configuration.

quarter of a clock period. This can be ensured by constraining the physical location of the coarse pulse and the delay registers close to each other. The linearity is discussed in section III.

C. DCM Configuration

The architecture proposed above needs four clocks with a phase relationship of $\Delta\varphi = 90°$. A single DCM can be used to produce these from a base clock which in turn can be produced out of a lower frequency clock using one or two additional DCMs [Figure 1]. The resulting time resolution is equivalent to a clock with a four times higher frequency.

In this work, a 9-bit DPWM with a frequency of 1 MHz results in a base clock frequency of $f_{clk} = 2^{9-2} = 128$ MHz. The resulting 0° phase is used to clock all registers of the coarse DPWM. The other three phases are fed to the fine resolution block.

D. Output Register

The final output pulse is produced by a level sensitive set/reset-flip-flop (SR-FF), sometimes called a SR-latch, which is not normally available as a dedicated hardware block inside a FPGA. However, Spartan 3 FPGAs have configurable input/output blocks (IOBs) which include two flexible output registers as shown in Figure 3a. Only one of these registers is used and must be configured to act as a SR-FF with the following attributes:

- Global reset must always drive the output low.
- Set is dominant over reset.
- Reset is provided inverted.

This may be achieved with the following Verilog code where `resetn` is the global negative asynchronous reset:

```
always @(resetn or pulse_set or pulse_resetn) begin
    if      (!resetn      ) dpwm_pulse = 1'b0;
    else if ( pulse_set   ) dpwm_pulse = 1'b1;
    else if (!pulse_resetn) dpwm_pulse = 1'b0;
end
```

This code configures the IOB so that the SR (set-or-reset) input is used as positive asynchronous *reset* input for the global reset and therefore REV (reverse of SR) as *set* input for the *pulse set* signal. The *pulse reset* signal is used as gate input together with a constant zero as data input. This equivalent logic symbol of this IOB configuration is shown in Figure 3b.

III. Optimisation of Linearity

The proposed architecture produces satisfactory linearity due to the usage of a synchronous counter, synchronous delay line and the minimal number of combinatorial logic elements.

A. Causes of Non-linear Behaviour

The coarse pulse linearity is only affected by the clock jitter of the coarse counter clock. The DCMs produce a small periodic jitter when used as multipliers causing a periodic pulse width error over the duty cycle range as shown in Figure 4. However, the simulated peak-to-peak value of this error is under 14 ps and can be neglected over other non-linear influences.

Non-linear behaviour can be introduced between the fine steps by different signal routing delays from the four coarse pulse registers (coarse output register plus three delay registers) to the MUX. These delay differences introduce a selective signal jitter to the final reset signal causing a linearity error which repeats periodically for every MUX selection.

B. Improving Linearity

To improve the linearity of the implementation, the internal structure of the FPGA must be known. FPGAs use configurable logic blocks (CLBs) to produce the required logic behavior. These CLBs consist of four slices, two stacked top right and two stacked bottom left, which each include a configurable D-FF, a programmable look-up-table (LUT) with four inputs and additional dedicated logic like 2-to-1 MUX (FiMUX) for signal routing as shown in Figure 5. The CLBs are positioned like the squares of a chess-board over the FPGA separated by the signal wires. This allows the manual positioning of logic in specific slices using a XY coordination system.

The main optimisation goal is to minimise the routing delays between the different delayed pulses, i.e. on the paths between the coarse pulse register and the three delay registers and from these to the MUX. This can be achieved by manual placing of the synthesized logic which is easily done with the Xilinx Floorplanner using drag & drop.

The placement starts with the selection of a pulse output pin which is determined by the FPGA board and application.

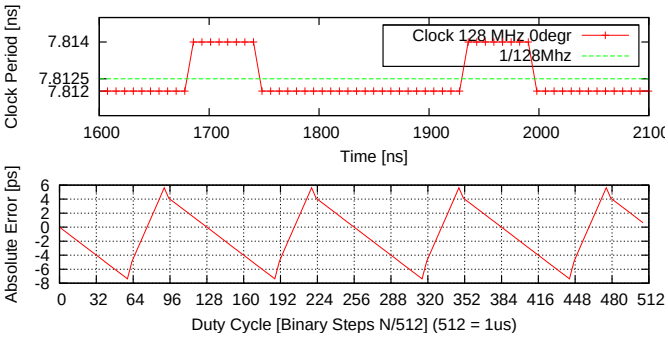

Fig. 4. Periodic clock jitter of generated clock signal ($\times 32/25$ DCM) over time (top) and resulting absolute pulse width error over vs. duty cycle (bottom).

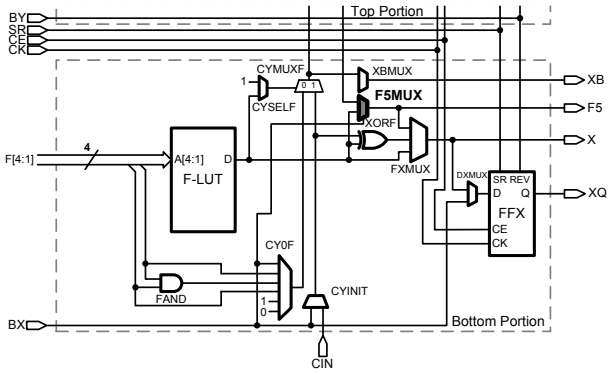

Fig. 5. Bottom portion of Xilinx FPGA Logic Slice [15], top portion only differs marginally.

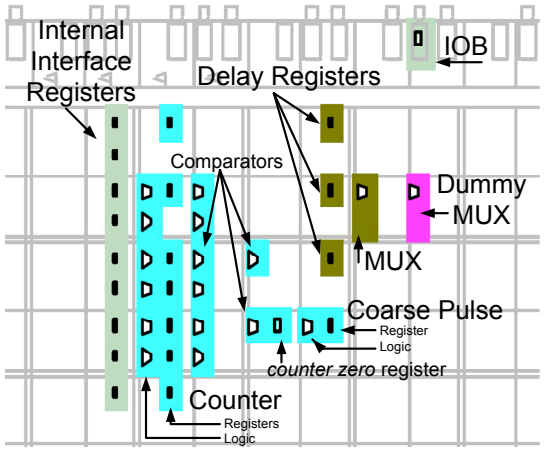

Fig. 6. Manual placed DPWM on Xilinx Spartan 3 FPGA.

The Diligent Xilinx Spartan 3 development board was used for this work which connects several of the FPGA I/O pins to three 40-pin connectors.

The DPWM MUX, implemented by a single slice, is then set into the CLB directly beside the IOB to minimise the routing distance.

Next the four pulse registers are placed into the four slices symmetrically to the left of the MUX slice, stacked after the delay order. Only one FF per slice is used because every slice has only one clock input, forcing both slice FFs to share the same clock. However without this limitation, this placing scheme should be used because routing delays between FFs inside the same slice and different slices would differ.

In addition, the coarse pulse logic is placed into a LUT in the same slice as the coarse pulse register. Also the comparator logic and register, the counter registers, adder logic and the interface registers are placed in slice columns from right to left to minimise used area which also increases the maximum usable clock speed. These parts can either be placed one by one or by defining a rectangle placing range for each type. The latter leaves the stacking order to the synthesiser, which is most useful for the comparator and counter logic.

The DCMs are located in the four edges of the FPGA and are placed clockwise with the four phase DCM in the same quadrant as the DPWM. The final DPWM placement is shown in Figure 6.

While the manual placing adds an extra design step for the DPWM, placing the low number of parts only requires a short amount of time and has a significant influence on the fine resolution linearity as shown in the experimental results. The placing locations can be persistently stored in an user constraint file (.ucf). Some of them can even be directly set in the Verilog source code on register transfer level (RTL) elements using Verilog 2001 attributes or meta comments, which simplifies for later design changes like signal renaming or hierarchy changes.

C. Offset Error

As shown in Figure 1, the coarse set signal is directly routed to the output SR-FF while the fine reset pulses are routed though a 4-to-1 MUX, which introduces an extra delay that causes an offset. This can be compensated by placing an identical dummy 4-to-1 MUX in the path of the set signal where all four input signals are connected to each other.

In a Spartan 3 FPGA these MUXs can be implemented using a single slice by using the two LUTs as 2-to-1 MUXs and the dedicated F5MUX cell as a third 2-to-1 MUX multiplexing the LUT outputs. This single slice solution results in minimised delay differences leading to minimised linearity errors.

To incorporate a dummy MUX using Verilog HDL, a simple module should be written and instantiated for both MUXs while setting a `KEEP` attribute of the dummy instantiation using either Verilog 2001 attributes, meta comments or a user-constrain file (.ucf). The synthesiser then infers both LUTs and the F5MUX, but only connects one of the two identical inputs of each LUT, which is acceptable for the offset correction.

In addition, the dummy MUX can also be used to match different routing delays of the four inputs of the real MUX in order to minimise non-linear behaviour. For this however all four dummy MUX inputs must be used which can be realised by manually instantiating the two LUTs (for XIlinx FPGAs as `LUT2_L`), constraining them to the same slice (`RLOC=X0Y0`) and inferring the F5MUX by coding a 2-to-1 MUX. The *set* and *reset* signals now route through identical cells with similar routing paths so that delay differences which cause non-linear behaviour are minimised.

The dummy MUX should be placed in close proximity to the real MUX and the output register, ideally in a symmetrical configuration to minimise routing delay differences.

IV. Experimental Verification

The proposed DPWM has been implemented on a low-cost Xilinx Spartan-3 XC3S1000-FT256-4 FPGA as part of a novel bi-directional data transmission scheme for a digital controlled isolated SMPS [16]. Figure 7 shows the SMPS circuit board with the DPWM FPGA attached on the upper left side. The second FPGA board on the right side contains the secondary side communication block together with interface logic to the

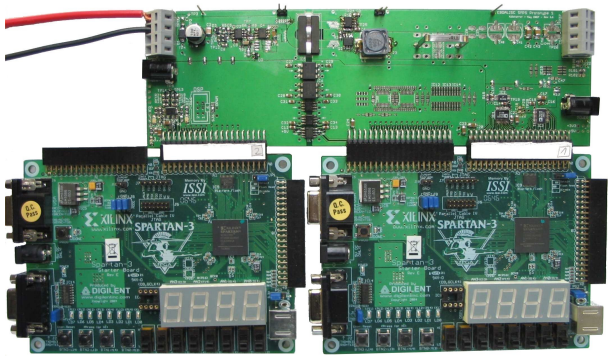

Fig. 7. SMPS (top board) driven by the presented DPWM on a Xilinx Spartan-3 FPGA (bottom left board). The second FPGA (bottom right board) is used to produce digital feedback and is not directly related to the DPWM.

analog-to-digital converter (ADC) and digitally programmable voltage reference.

The on-board 50 MHz clock was used together with the DCM chain shown in Figure 1 to produce the required higher clocks frequencies and phases.

The hardware description language Verilog HDL together with the Xilinx Integrated Synthesis Environment (ISE) was used to design all parts of the DPWM and the surrounding system. The required DCMs were instantiated with the help of Xilinx Architecture Wizard. While the design was carried out for a Spartan 3 FPGA, the same code can be used for other Xilinx FPGAs (e.g. Virtex) as long as the necessary number of DCMs are present. A transfer to a FPGA supplied by a different manufacturer is also possible as long as an equivalent DCM/DLL block is available. In the case of a transfer to a mixed-signal ASIC, a PLL block with a four-phase clock output can be used.

A. Simulation Results

To ensure correct functional behaviour of the DPWM including its internal data interface, the design has been simulated using the Questa/Modelsim simulator and a behavioural simulation model without logic cell delays. The results of this simulation showed only the need of the extra clock delay for the two duty cycle LSBs for the four widest pulses as described in section II-A3.

The linearity and monotonic behaviour was then simulated using a *post-place and route* simulation model produced by the synthesis tools by creating a Verilog gate level file instantiating simulation modules of all used FPGA cells. The delay timing is provided using a generated SDL file. This simulation shows the influence of cell placing in the FPGA and was used to compare the automatic and the manually optimized placing of the DPWM cells. The excellent DCM timing simulation model provided by Xilinx must be highlighted here which allows the accurate simulation of the produced clocks including phase offsets and periodic jitter as shown in Figure 4. The model also reports interface errors such as too long input clock periods or too short reset pulses during the simulation which allows for easy error tracking. Originally a normal reset synchroniser built from two FFs was used for the DCM resets but this has been extended to three FFs after the DCM reported the need for a three clock period long reset pulse during the first simulation.

B. Synthesis Results

The design was synthesized using the Xilinx XST which is part of the ISE. The only timing constraint used was the input clock period length where the DCM clock constrains are automatically calculated from the DCM parameters by the Xilinx tools. Verilog 2001 attributes were used to set synthesis constrains like `KEEP` and `KEEP_HIERARCHY` to the register transfer level (RTL) modules as well as the I/O port location of the external signals direct in the source files. An additional user constraint file (.ucf) was used to store the location (`LOC`) constrains for the manual placed cells.

The synthesized DPWM occupies only 28 out of the available 7680 logic slices (0.36%), where 22 LUTs (0.14%), 32 FF (0.21%) and 1 IOB was used. However, the design needs three out of the four DCMs and seven out of the eight global clock buffers (BUFG). The maximal frequency of the design is 143.6 MHz.

C. Experimental Results

The linearity of the implementation has been measured using a test circuit synthesised on the same FPGA which increments the duty cycle by one and produces a trigger signal at the start of every pulse. A modern digital LeCroy oscilloscope was used to measure the DPWM pulse length as well as to save the sampled pulses to data files for later interpretation using Matlab.

The linearity over the full duty cycle range is shown in Figure 8. In addition, Figure 9 shows the linearity and monotony of the delay line steps (fine resolution) in relation to the counter steps (coarse resolution) of both non-optimized and optimized layout.

The DPWM pulses produced by the used FPGA can be seen in Figure 11. The initial over- and final under-shoot is due to bandwidth limitations in the FPGA output drivers. When connecting the FPGA to a power stage or other circuit an external driver must be used to ensure adequate driving strength and speed as well as electrical protection of the FPGA output pin. This driver has a higher bandwidth and gain and therefore removes the over- and under-shoots by saturating to its maximal and minimal output voltage, respectively.

V. Conclusions

FPGA based DPWM implementations offer a method for practicing engineers to prototype digitally controlled SMPSs. This paper proposes an architecture with a DCM based synchronous delay line suitable for implementation of high-speed high-resolution DPWM with good linearity on low-cost FPGAs using only a few logic cells. The proposed manually optimized DPWM reduces the integral non-linearity error by 0.5 LSB. A DPWM implementation with 9-bit resolution with a switching frequency of 1 MHz has been verified on a Xilinx Spartan-3 FPGA.

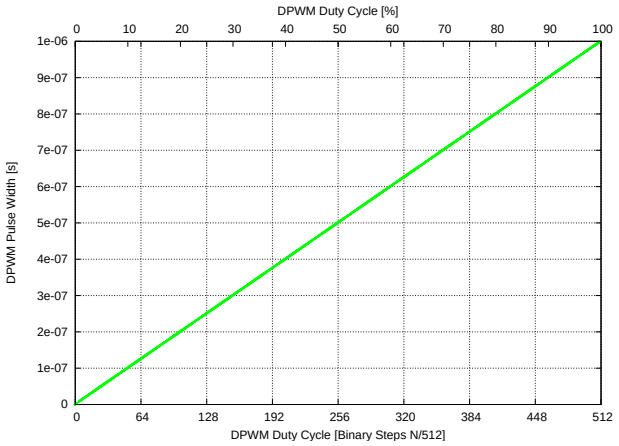

Fig. 8. Measured DPWM linearity over full duty cycle range.

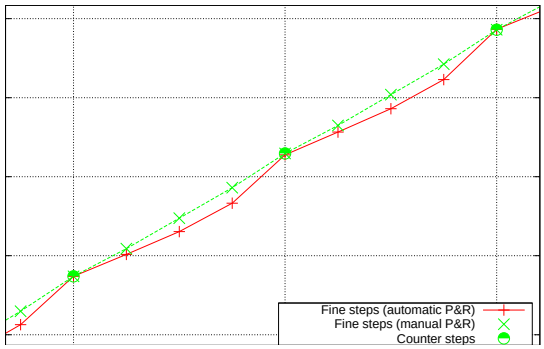

Fig. 9. Comparison of fine resolution linearity between automatic and manual place & route.

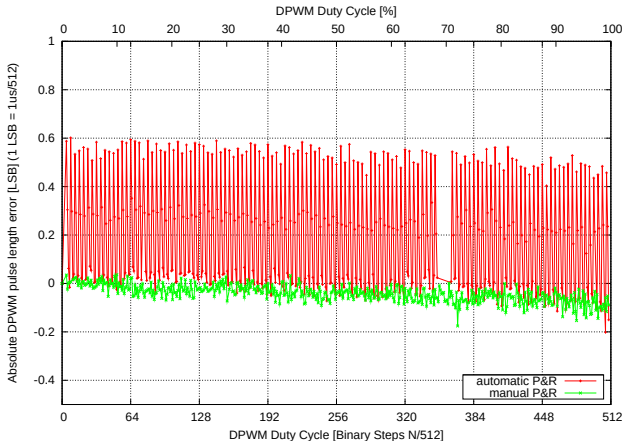

Fig. 10. Measured pulse width error.

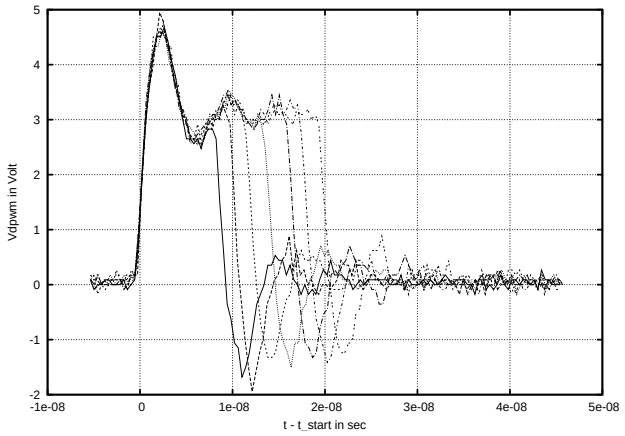

Fig. 11. Superposition of measured DPWM pulses with D={4..10}/512.

ACKNOWLEDGMENT

This work was funded under Enterprise Ireland Technology Development fund (CFTD/219/05).

REFERENCES

[1] C. Mullett, "A 5-year power technology roadmap," in *Applied Power Electronics Conference and Exposition, 2004. APEC '04*, vol. 1, 2004, pp. 11–17.

[2] Y.-F. Liu and P. Sen, "Digital control of switching power converters," in *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*, 2005, pp. 635–640.

[3] *IEEE Transactions on Power Electronics. (Special issue on digital control in power electronics)*, vol. 18, n. 1, part II, Jan 2003.

[4] A. Peterchev and S. Sanders, "Quantization resolution and limit cycling in digitally controlled PWM converters," *Power Electronics, IEEE Transactions on*, vol. 18, pp. 301–308, 2003.

[5] H. Peng, A. Prodic, E. Alarcon, and D. Maksimovic, "Modeling of quantization effects in digitally controlled DCDC converters," *Power Electronics, IEEE Transactions on*, vol. 22, pp. 208–215, 2007.

[6] A. Dancy and A. Chandrakasan, "Ultra low power control circuits for PWM converters," in *Power Electronics Specialists Conference, 1997. PESC '97*, vol. 1, 1997, pp. 21–27.

[7] E. O'Malley and K. Rinne, "A programmable digital pulse width modulator providing versatile pulse patterns and supporting switching frequencies beyond 15 MHz," in *Applied Power Electronics Conference and Exposition, 2004. APEC '04*, vol. 1, 2004, pp. 53–59.

[8] K. Wang, N. Rahman, Z. Lukic, and A. Prodic, "All-digital DPWM/DPFM controller for low-power DC-DC converters," in *Applied Power Electronics Conference, 2006. APEC '06*, 2006, p. 5 pp.

[9] A. Syed, E. Ahmed, D. Maksimovic, and E. Alarcon, "Digital pulse width modulator architectures," in *Power Electronics Specialists Conference, 2004. PESC 04*, vol. 6, 2004, pp. 4689–4695.

[10] A. de Castro and E. Todorovich, "DPWM based on FPGA clock phase shifting with time resolution under 100 ps," in *Power Electronics Specialists Conference, 2008. PESC 08*, 2008, pp. 3054–3059.

[11] M. G. Batarseh, W. Al-Hoor, L. Huang, C. Iannello, and I. Batarseh, "Segmented digital clock manager- FPGA based digital pulse width modulator technique," in *Power Electronics Specialists Conference, 2008. PESC 08*, 2008, pp. 3036–3042.

[12] S. C. Huerta, A. de Castro, O. Garcia, and J. Cobos, "FPGA based digital pulse width modulator with time resolution under 2 ns," in *Applied Power Electronics Conference, APEC '07*, 2007, pp. 877–881.

[13] R. Foley, R. Kavanagh, W. Marnane, and M. Egan, "An area-efficient digital pulsewidth modulation architecture suitable for FPGA implementation," in *Applied Power Electronics Conference and Exposition, 2005. APEC 2005*, vol. 3, 2005, pp. 1412–1418.

[14] ——, "A versatile digital pulsewidth modulation architecture with area-efficient FPGA implementation," in *Power Electronics Specialists Conference, 2005. PESC '05*, 2005, pp. 2609–2615.

[15] Xilinx Inc., *DS099: Spartan-3 FPGA family: Complete data sheet*, v2.4, June 25 2008. [Online]. Available: http://www.xilinx.com/support/documentation/data_sheets/ds099.pdf

[16] M. Scharrer, M. Halton, and T. Scanlan, "Digitally controlled isolated SMPC with bi-directional data transmission scheme," in *Industrial Electronics, 2008. ISIE 2008. IEEE International Symposium on*, vol. 1, May 2008, pp. 380–383.