FPGA Implementation of a Digital Tachometer with Input Filtering

Daniel Mic, Stefan Oniga

Electrical Department, North University of Baia Mare

Dr. Victor Babeş Street 62 a, 430083 Baia Mare, Romania

danmic@ubm.ro, onigas@ubm.ro

Abstract

This paper presents and compares two methods of FPGA implementation of a digital tachometer with input filtering and forward/reverse detection. Within first method, usually used in low speed range, , the elapsed time between two successive pulses is counted. With the second method usually recommended in high speed range, pulses coming from an encoder are counted in a fixed period of time. In this paper it will be proved that using an FPGA circuit, working at the right frequency, the first method of speed measuring can be adequate for all speed range. The digital tachometer will be implemented into XC3S200 FPGA circuit and will be used for closing the velocity loop for a brushless DC motor.

1. Introduction

The tachometer is an essential part of motor velocity closed loop control. The role of this device is to measure the frequency of pulses train coming from an optical encoder mounted on the motor shaft and to convert the result in speed information.

There are known methods to determine the speed of a drive using digital tachometers. Next will be presented the two most commonly used: first is based on measuring the elapsed time between two successive pulses and the second is based on counting the pulses during an established period of time. The last method is also known as CET method (Constant Elapsed Time) [2].

In this paper two types of digital tachometers will be implemented using FPGA (Field Programmable Gate Array) circuits, based on previous presented methods. Performances of both types of digital tachometers will be tested in a speed closed loop control of a brushlees DC (BLDC) motor. The digital designing environment Xilinx ISE and the toolbox System Generator from Matlab are used for simulation and implementation [7].

2. Digital Tachometer Method I

For hardware implementation of digital tachometer based on measuring the elapsed time between two successive pulses, two blocks will be designed, see figure 1. First of them gives the interface with the optical encoder and it also filters the signals. The second block takes the filtrated signal and determines the value of speed. In figure 1 it can be noticed signals *Ea, Eb* that come from the encoder. Based on these signals a pulses train is generated (signal *PT*) for the measurement block. A signal that indicates the direction in which the motor shaft moves is also generated.

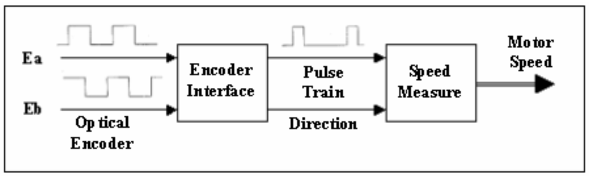

Fig. 1 Digital Tachometer blocks

The interface block is designed using state diagrams, see truth tables shown in table 1. State diagram shown in figure 2 generates pulse train *PT,* based on quadrature signals coming from the encoder and it also filtrates unwanted states. The signal *Dir* that indicates direction is generated based on *Ea, Eb* sequences. From the state diagram VHDL code is generated and an HDL co-simulation is also performed. [3], [5].

States	Ea	Eb
S0	0	0
S1	0	1
S2	1	1
S3	1	0
Dir = 0		
States	**Ea**	**Eb**
S0	0	0
S4	1	0
S5	1	1
S6	0	1
Dir = 1		

Tab 1. Truth table for interface block

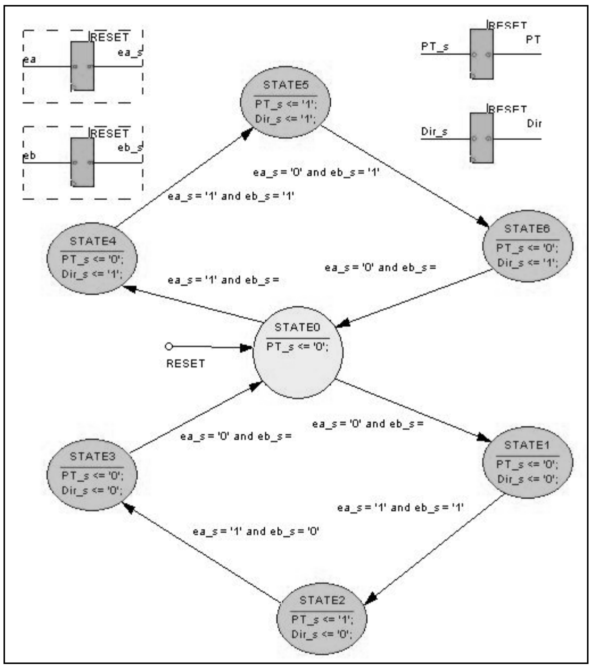

Fig 2. State diagram based on truth tables

The second building block of the digital tachometer computes the shaft speed of the motor. The computation algorithm was described in VHDL based on diagram shown in figure 3. According to this diagram the time elapsed between two successive pulses is to be counted, the obtained value *count_enc* is used in equation (5). The constant value from equation (5) was obtained considering that the optical encoder used for these experiments gives 500 pulses/rotation [1] and the sample rate was selected to be equal with FPGA working frequency (20 ns), see equations (1,2,3).

$$f_{codif} = 500 * \frac{V_{rpm}}{60} \tag{1}$$

$$T_{codif} = \frac{1}{f_{codif}} = \frac{60}{500 * V_{rpm}} \tag{2}$$

$$count\_enc = \frac{T_{codif}}{T_{FPGA}} = \frac{1}{V_{rpm}} \frac{60}{500} * \frac{1}{20 * 10^{-9}} \tag{3}$$

$$const. = \frac{60}{500} * \frac{1}{20 * 10^{-9}} = 6 * 10^{6} \tag{4}$$

$$V_{rpm} = \frac{const.}{count\_enc} = \frac{6.000.000}{count\_enc} \tag{5}$$

where:

f_{codif} – the frequency of the signal from the optical encoder;
T_{codif} – the period of time of the signal from the optical encoder;
count_enc – the counted time elapsed between two successive pulses;
T_{FPGA} – FPGA working frequency, $20x10^{-9}$ s;
V_{rpm} – the speed of motor, measured in *rpm.*

The block for speed measure is described in VHDL and needs to implement the division from equation (5), so that using *Core Generator* utility an *IP core* division block is generated. This block is adequate for an optimized implementation into FPGA.

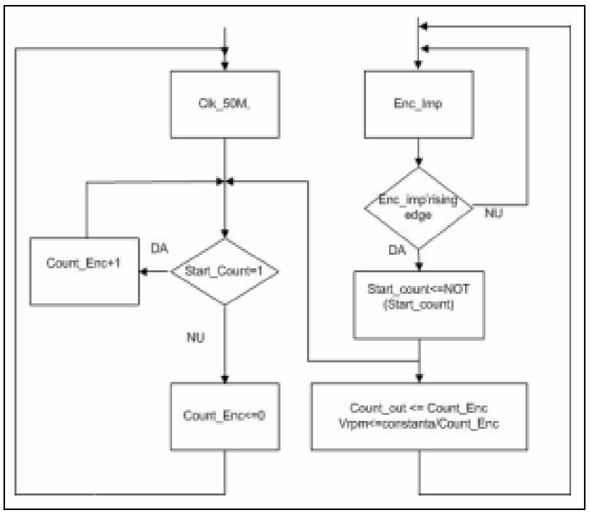

Fig. 3. Frequency to speed conversion algorithm

Because of the high value of the sample rate measure error is minimum. In figure 4 is shown the error distribution in *rpm*, it can be noticed that depends by motor speed range. For speeds under 2500 rpm the error is zero and then gradually increase so that at a speed of 8000 rpm it becomes 10 rpm. The error is due to rounding, the speed measure module described in VHDL works only with integers. The error is computed as difference between the computed value and the measured value. In the reference [3] an error of 1-2 rad/s (9-19rpm) in high-speed range is considered acceptable. So that a digital tachometer with a maximum error of 10 rpm at a speed of 8000 rpm can be considered as having good precision.

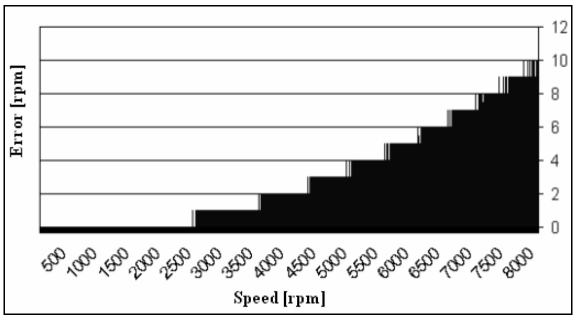

Fig. 4. Error distribution depending on speed range

Functional simulation of the digital tachometer was accomplished using ModelSim simulator integrated in Xilinx ISE digital designing environment.

2.1 FPGA implementation results

Two FPGA implementation was accomplished, one optimized for speed (41% occupied resources) a division takes place in one period of clock and the other optimized for area (19% occupied resources), a division takes place in 4 periods of clock.

2.2 Hardware co-simulation results

During the last time, it became possible to make functional simulation for projects, already implemented in hardware, this type of simulation is known as hardware in the loop (HIL) simulation, see [3], [4], [5]. Figure 5 shows a HIL simulation of a speed control system with a digital tachometer acquiring only one signal from the optical encoder. It can be noticed that for a high sample rate (20 ns) there are a lot of spikes, that is way it is mandatory to acquire both encoder signals and filter the unwanted values like in diagram presented in figure 2. In figure 6 the filtrated signal is presented. Both the reference speed and the speed read by the tachometer are shown in figure 6. Due to the small error of tachometer the signals appears overlapped.

2.3 Experimental results

Figure 7 shows signals *Eb* and *PT* captured from the oscilloscope. It can be noticed that for a frequency around of 16.5 kHz, the computed speed is 2000 rpm, according to equations (1-5).

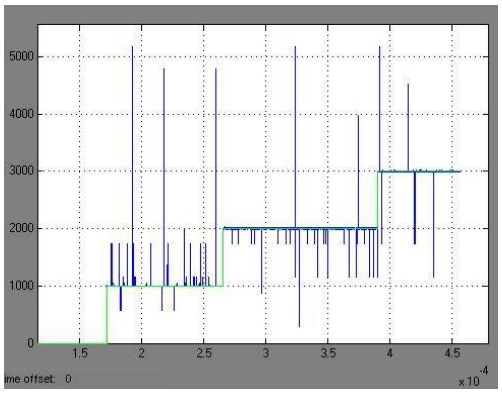

Fig. 5. Hardware co-simulation waveforms for only one signal acquired from the optical encoder

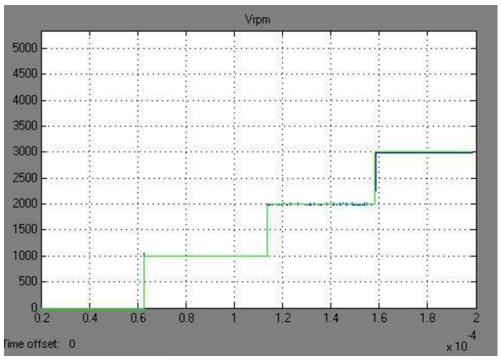

Fig. 6. Hardware co-simulation waveforms for filtrated signal (*PT*) acquired from the optical encoder

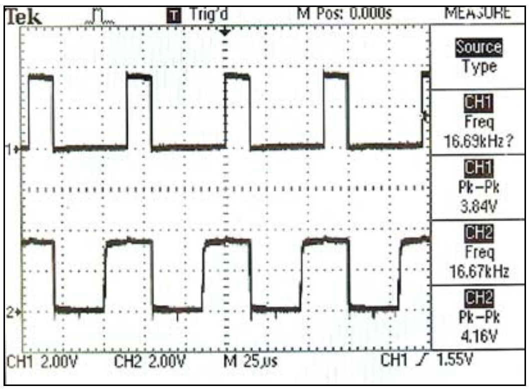

Fig. 7. Filtrated signal *PT* and optical encoder acquired signal (Eb) for a speed of 2000 rpm

3. DIGITAL TACHOMETER METHOD II

The second version of digital tachometer implemented in hardware is based on so called CET method that consists in pulses counting during an established period of time. Besides speed measurement this module also integrates conversion blocks, binary to BCD and BCD to seven segments that allow displaying the results of computed speed.

This version of digital tachometer is totally described in VHDL.

From figure 8, it can be noticed that the tachometer is made up from three blocks. First of them (*enc_interface*) works at 1 kHz and during an established period of time counts the pulses coming from the encoder. Every one second, the refreshing period, the counting result together with a *start* signal is transmitted next to the binary to BCD conversion block (*bin_to_bcd*). The BCD to seven segments block (*bcd_to_7seg*) besides conversion also allows signals multiplexing for 7-segment displays from the Spartan 3 testing board made by Digilent.

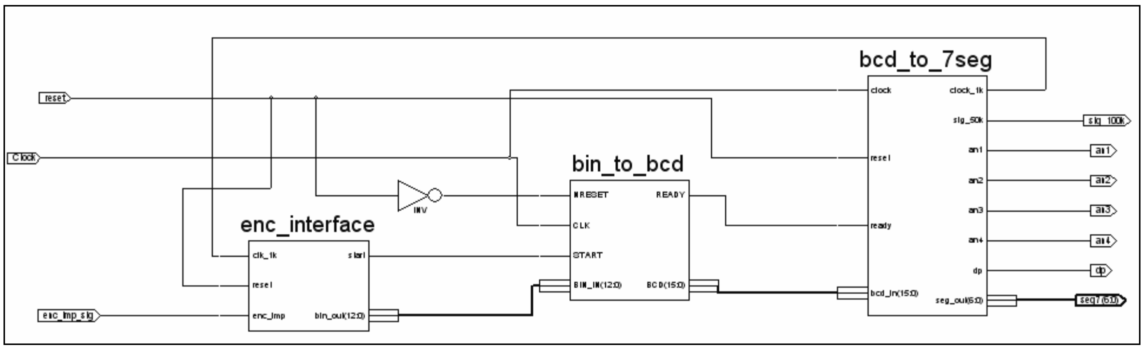

Fig. 8. RTL architecture of digital tachometer based on CET method

VHDL description of first module is based on algorithm shown in figure 9 and it can be noticed that this module is made up from counters. Counter 1 (*count1*) establishes the period of time of 1 second. Another counter (*count2*) counts the 500 pulses corresponding to one complete revolution for the encoder, counter *count5* keeps the number of complete revolutions carried on one second. Counter *count4* is used to minimize the counting error caused by rounding. This counter keeps with approximation the value for uncompleted revolutions that take place in 1 second.

Results of functional simulation and measurement error determination

Figure 10 shows the functional simulation waveforms of the digital tachometer based on CET method. It can be noticed that for a computed speed of 7400 rpm the displayed value is 7396. The error comes from *count4* that counts with a step of 8 a number of 62 steps.

Figure 11 shows the error distribution obtained as difference between the ideal value and the computed value of *count4*.

.

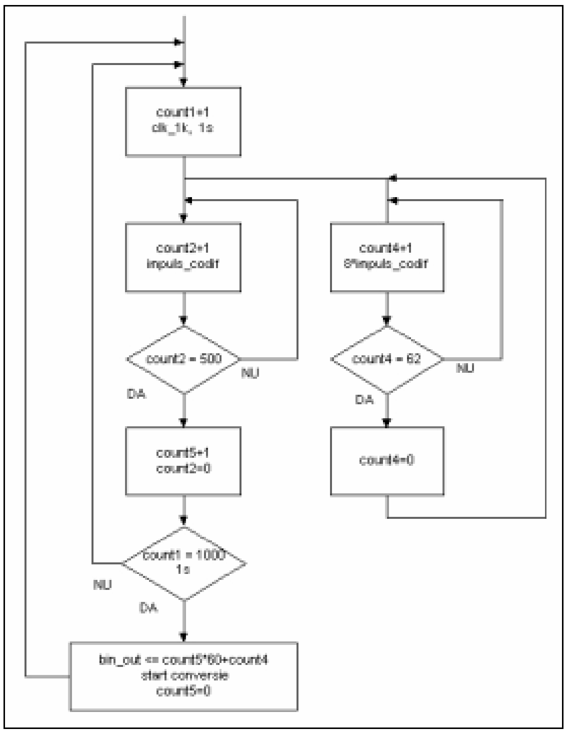

Fig. 9. CET method implementation algorithm

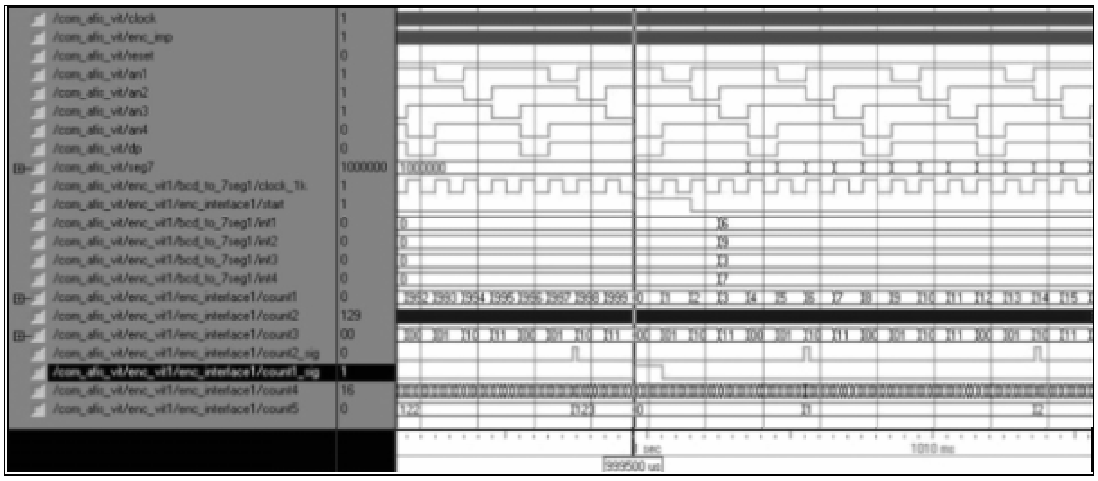

Fig.10. Functional simulation waveforms of digital tachometers based on CET method

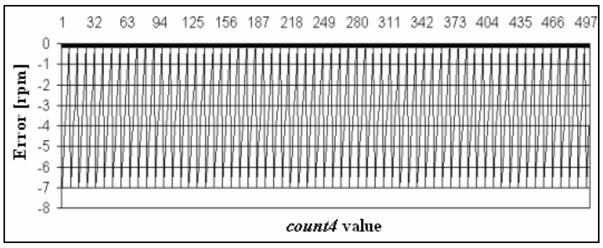

Fig. 11. Error distribution for digital tachometer based on CET method

Implementation took place in XC3s200 Xilinx FPGA Spartan family. The consumed resources were around 5% and the maximum working frequency is around 250 MHz.

4. CONCLUSIONS

Take into account good dynamic performances and small measurement error it can be concluded that a competitive digital tachometer was successfully implemented into an FPGA circuit, especially the version based on first method. In order to have good precision the first method-based tachometer must work at high frequency especially for motors that runs in high-speed range. It was a good approach to accomplish the implementation into an FPGA circuit that works at 50 MHz, that giving the opportunity for closing the control loop of high-speed BLDC motors. Due to poor dynamic performances but having good precisions that does not depend by speed, digital tachometer based on CET method, together with adequate converters, can be implemented in user interface modules.

REFERENCES

[1] Agilent Technologies, Threee Channel Optical Incremental Encoder Modules, Technical Data, February 2002;

[2] Galvan E., Torralba A., Franquelo L., ASIC Implementation of a Digital Tachometer with High Precision in Wide Speed Range, IEEE Transactions on Industrial Electronics, vol. 43, no. 6, 1996;

[3] Mic D., Micu E., Oniga S., *Hardware and Software Co-Design Method for Implementation of Closed Loop Control for Brushless DC Motor*, Proceedings of 10th International Conference on Optimization of Electrical and Electronic Equipments OPTIM'06, 2006, Braşov, Romania, vol. 3 pp. 59-66;

[4] Visser P. Groothuis M, and others, FPGA as versatile configurable I/O devices in Hardware in the Loop Simulation, RTSS Lisbon, December 2004;

[5] Xilinx Inc., - Nabeel Shirazi, Jonathan Ballagh, Put Hardware in the Loop with Xilinx System Generator for DSP, Fall 2003;

[6] Xilinx Inc., Spartan-3 FPGA Family, Complete Data Sheet, DS099, 2006;

[7] Xilinx Inc., System Generator for DSP (www.xilinx.com).