# A HARDWARE DC MOTOR EMULATOR

VAGNER S. ROSA[1], VITOR I. GERVINI[2], SEBASTIÃO C. P. GOMES[3], SERGIO BAMPI[4]

Abstract — Much work have been done lately to develop complex motor control systems. However they always rely on a physical drive/motor/encoder setup for experimental results. This paper presents a hardware DC motor emulator that can be synthesized to FPGAs. The emulator is intended to replace an actual DC motor during the development phase of motor controllers. A torque based input is required and incremental encoder output is provided, so this model can replace both the DC motor, and its power driver without modifications to the motor control system. The proposed emulator is able to reach a clock frequency of hundreds of megahertz and uses very few logic resources in current FPGA technologies. The hardware can be parameterized at synthesis time to make the model suitable for specific needs.

Key-words: DC MOTOR; Emulator; FPGA; Hardware

## INTRODUCTION

The advance in the microelectronics industry in recent years led to the availability of high performance processors, memory and logic in small packages and

[1] Computational Sciences Center, Federal University of Rio Grande, vsrosa@inf.ufrgs.br
[2] Computational Sciences Center, Federal University of Rio Grande, gervini@gmail.com
[3] Mathematics and Phisics Institute, Federal University of Rio Grande, sebastiaogomes@furg.br
[4] Informatics Institute, Federal University of Rio Grande do Sul, bampi@inf.ufrgs.br

at low cost. This advance allowed the design of motor controllers with sophisticated algorithms that are best performing than classical PID designs.

When designing a motor controller the specification of a realtime computer is needed. This computer has to meet a set of requirements such as computational speed, memory, and interrupt interval among others. During the development phase, is important to have a system that is flexible enough to support changes required by control laws.

Moreover, in the development phase of a motor control system the motor itself and its power driver are usually required to close the control loop. In some designs, several motors and drivers are required to evaluate the motor control system. Bugs in the control law, software, or hardware during the development phase can easily damage the power driver or the motor itself.

In this paper we propose a hardware brushed DC motor emulator to be used in the development phase of motor controllers. The motor model employed is very simple and does not implement electromagnetic or friction losses. The emulator is implemented in VHDL and can be synthesized to FPGA. The input is a digital valued torque and the output is a two phase incremental encoder. The model is designed to run in hardware at a discrete clock speed much higher than the control law (hundreds of megahertz), so that even small delays due to real-time system or the control law computation can be catch and simulated by the model.

This paper is organized as follows. In section II a related work on this subject is presented. Section III present the proposed model and its hardware implementation in Section IV. Section V present implementation results. Finally Section VI presents the conclusions of this work.

## RELATED WORK

Several works are found in the literature on the motor modelling subject [1-3]. Some of these are for brushed DC motors, but none deal with the problem of real-time simulation of brushed DC motors for the purpose of replacing an actual motor during the development phase. The work of [2] presents the closest approach found to the current work with a DSP-ASIC approach for this purpose, but the kind of motor considered is induction AC.

## PROPOSED MODEL

A motor system composed by the DC motor, a power drive and a incremental encoder is modelled in this work. A typical brushed DC motor is showed in Figure 1.

Fig. 1 A brushed DC motor with incremental encoder coupled in the motor shaft.

The scope of the proposed emulator is presented in Figure 2. The system model is capable of emulating the DAC and driver, the brushed DC motor and a incremental encoder connected to the output shaft of the motor, providing a complete solution to completely eliminating the motor setup during the development phase of the motor controller.

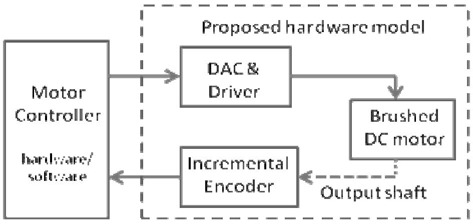

Fig. 2 Proposed hardware model scope.

a) Brushed DC motor model

Brushed DC motors usually have stator with magnets and a wounded rotor. The rotor coils are connected to a commutator that receive the electric input by means of a pair of carbon based brushes. The typical DC motor model is presented in Figure 3.

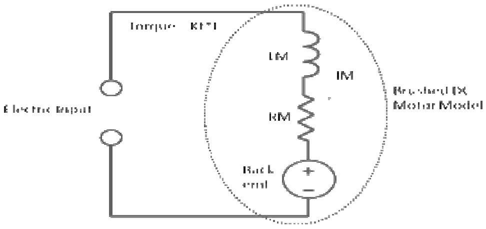

Fig. 3 Electric model of a brushed DC motor.

In Fig. 3, the LM inductor is mainly due to the coil inductance. The RM is the series resistance and is due to both coil resistance and brushing resistance. The back emf is produced by the movement of the rotor in the magnectic field produced by the stator and is proportional to the rotational speed.

An important feature of brushed DC motors is that the produced torque in the motor is proportional to the electric current flowing into the motor as stated in Eq. (1).

$$\tau = K_t . I_M \qquad (1)$$

where $I_M$ is the current flowing through the motor and $K_t$ is a constant to convert current to torque.

In DC motors a small (but sometimes significant) portion of the produced torque is lost due to friction, and some electromagnetic fenomena. *No losses are considered in the proposed model*.

b) DAC and driver

Due to the direct relation between torque and electric current in brushed DC motors, a digital to analog current converter can be used to convert a numerical torque representation direct to an output torque. Figure 4 present a schematic diagram of the digital to analog (DAC) converter and current based power amplifier drive.

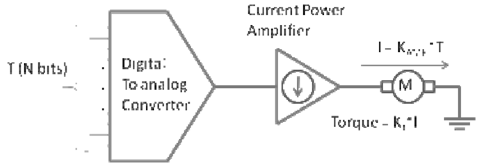

Figure 4. Digital to analog converter and power drive.

The output current of the power amplifier is given by

$$I = K_{AMP} . T \qquad (2)$$

where T is the required torque value at the input of the analog to digital converter and KAMP is the constant to convert the T input to I. The motor torque can be derived by Eq. 3.

$$\tau = K_{AMP} . K_t . T \qquad (3)$$

c) Encoder signal

The encoder signal is generated by a pair of sensors connected to a specially designed disk. The typical setup is the presented in figure 5.

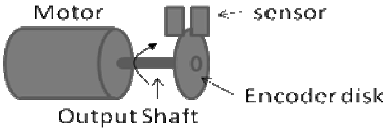

Fig. 5 Encoder connection to output shaft.

The incremental encoder output is two square shaped signals where a transition (rising or falling edge) of one of the signals represent an forward or backward movement depending on the state (zero or one) of the other. Figure 6 illustrates the behaviour of the incremental encoder in a forward and backward movement.

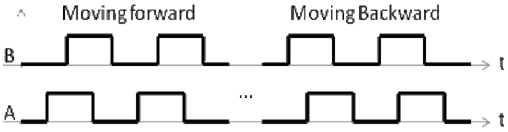

Fig. 6 Encoder output signal

d) Dynamic behaviour

The model for dynamic behaviour modelling is presented in Figure 7.

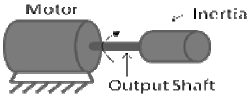

Fig. 7 Dynamic model of the motor.

In the figure 7, the brushed DC motor produces a mechanical torque that moves a load connected to the output shaft. This load is composed by the total equivalent inertia of the motor and of the load itself. The model assumes there is no friction or other parasitic torque.

The rotational acceleration is given by the Eq. 4.

$$\alpha = \frac{\tau}{I} \qquad (4)$$

where $\tau$ is the applied mechanical torque (supplied by the electric current in the motor) and *I* is the moment of inertia of the equivalent load.

The rotational speed $\dot{\theta}$ can be given as a function of the acceleration $\alpha$ by Eq. 5:

$$\dot{\theta} = \int \alpha dt + \dot{\theta}_0, \qquad (5)$$

where $\dot{\theta}_0$ is the initial speed.

Finally, the position of the output shaft $\theta$ is given by Eq. 6:

$$\theta = \int \dot{\theta} dt + \theta_0, \qquad (6)$$

where $\theta_0$is the initial position of the output shaft.

e) Discrete model

The dynamic behaviour presented above is in the continuous time domain, which is not feasible by computational means.

The discrete version assumes there is a fixed interval computation (*h*) of the next state of the dynamic. The discrete model for the output shaft position at a discrete time instant *T* is given by

$$\theta_T = h\left(\sum_{t=0}^{T} \dot{\theta}_t\right) + \theta_0 \tag{7}$$

where *h* is the discretization step, $\theta_0$ is the initial position of the output shaft (at *t*=0), $\dot{\theta}_t$ is the rotational speed of the output shaft at a given discrete time instant *t*. The output shaft position can also be computed iteratively by

$$\theta_n = \dot{\theta}_n h + \theta_{n-1} \tag{8}$$

The rotational speed $\dot{\theta}$, can be computed discretely by:

$$\dot{\theta}_T = h\left(\sum_{t=0}^{T} \alpha_t\right) + \dot{\theta}_0 \tag{9}$$

where $\dot{\theta}_0$, is the initial rotational speed of the output shaft (at *t*=0), $\alpha_t$ is the rotational acceleration of the output shaft at a given discrete time instant *t*. The output shaft speed can also be computed iteratively by

$$\dot{\theta}_n = \alpha_n h + \dot{\theta}_{n-1}, \tag{10}$$

## HARDWARE ARCHITECTURE

The proposed architecture for the motor model is presented in Figure 8.

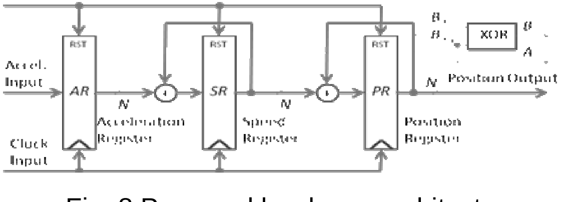

Fig. 8 Proposed hardware architecture.

In Figure 8, the Position Register (PR) stores the output shaft position, the Speed Register (SR) stores the current rotational speed and the acceleration register (AR) stores the current change rate of the rotational speed. These registers are updated every clock cycle.

a. Converting register values to physical units

The output shaft position is represented by a *N* bit register. A full revolution of the output shaft is given by a full count of the Position Register. The number of discrete positions is given by the following equation:

$$N_\theta = 2^N \tag{11}$$

And the resolution is given by

$$\theta_{step} = \frac{2\pi}{2^N}, \tag{12}$$

Another *N* bit register stores the current rotational speed. The speed resolution is given by the following equation

$$\dot{\theta}_{step} = \theta_{step} \cdot f_{clk}, \tag{13}$$

where $f_{clk}$ is the system clock frequency.

Finally another *N* bit register store the current acceleration. The smallest possible acceleration is given by:

$$\ddot{\theta}_{step} = \dot{\theta}_{step} \cdot f_{clk} \tag{14}$$

The Speed Register and the Acceleration Registers hold 2's complement values. The possible range for Speed Register is $\dot{\theta}_{step} \cdot [-2^{N-1}; 2^{N-1} - 1]$ and for Acceleration Register is $\ddot{\theta}_{step} \cdot [-2^{N-1}; 2^{N-1} - 1]$.

One important relationship is the smallest increment in acceleration for a given register size and clock frequency. From Eq. 12, 13, 14, an expression for the acceleration step can be derived:

$$\ddot{\theta}_{step} = \frac{2\pi . (f_{clk})^2}{2^N} . \frac{rad}{s^2}, \tag{15}$$

Figure 9 presents the value of $\ddot{\theta}_{step}$ for a range of values of $f_{clk}$ and *N*.

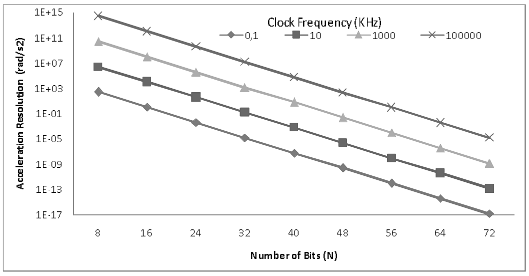

Fig. 9 Acceleration resolution.

The input of the DC motor model should be a torque. The torque and acceleration is related to the moment of inertia by:

$$\tau = \ddot{\theta}.I \tag{16}$$

where τ is the input torque, $\ddot{\theta}$ is the acceleration, and *I* is the moment of inertia. The smallest possible torque depends on the moment of inertia assumed and the resolution of the registers (value of *N*) in model.

The torque resolution ($\tau_{\text{step}}$) can be obtained from Eq. 15 and 16.

$$\tau_{step} = \frac{2\pi.I.(f_{clk})^2}{2^N}, \tag{17}$$

b. Connecting the torque input

For a given motor model implementation, *I* is the characteristic inertia of the motor and $f_{clk}$ is a constant system clock frequency. The user of this hardware model has to find a suitable *N* such that the required position output resolution ($\theta_{step}$)and the torque resolution ($\tau_{step}$) is satisfied.

The maximum input torque can be obtained by grouping M bits to form a torque Input Bus (IB) such that the LSB (least significant bit) of IB is connected to the LSB of the Acceleration Register (AR). In a typical implementation, IB is around 8 bits (a 8 bits torque resolution). The (*N*-*M*) most significant bits of AR are connected to logical zero. For large inertias and clock frequencies, the value of *M* will be much lower than *N*.

A more generic connection example is presented in Figure 10. The motor model controller has a torque output (TO) of 8 bits representing 2's complement values of torque (range from -128 to +127) and the motor model has an acceleration register (AR) of 32 bits. The LSB of TO is connected to the bit 5 of the AR. The lowest 5 bits of AR is connected to ground level (logic zero). The highest 19 bits are connected to the MSB of TO to perform sign extension.

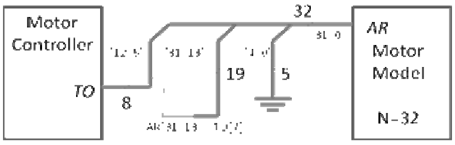

Fig. 10 Torque output (TO) to Acceleration Register (AR) connection example.

c. Connecting the incremental encoder output

The proposed architecture provides an incremental encoder output to allow a complete system integration of a typical brushed DC motor drive.

The encoder signal format was presented in Section III and the implementation in the proposed architecture is obtained by selecting a successive pair of bits of the Position Register. The selection is based on the desired resolution of the encoder.

This arrangement support power of two (*$2^N$*) based incremental encoders. The position of the lowest significant bit of the pair, counting from the most significant bit (MSB) of the Position Register (PR), defines the value of *N* (eg. $PR_{[63..0]}$ is 64 bit wide and we get two bits starting from $PR_{56}$, then we get a 256 pulses – 8 bits – resolution incremental encoder). The connection of the encoder logic to the PR is done by a parameter called ENC_BITS. The incremental encoder resolution is defined as $2^{ENC\_BITS}$.

## EXPERIMENTAL RESULTS

The proposed architecture was described in VHDL language. To validate the developed VHDL, a VHDL testbench file was written to provide the input stimuli (clock, reset and torque input) and to collect the output produced by the developed VHDL model.

Figure 11 show a simulation result where the value of N was configured to 64 bits and the clock frequency was set to 1MHz. The acceleration input was set as a step starting at 0 and jumping to $2^{40}$ (1099511627776 - the allowable range in this configuration is $[-2^{63};2^{63}-1]$) at the time marked by a vertical bar in the simulation. Starting at this time a rapid rise in the frequency of the encoder output can be observed, which can be interpreted as a rapid change in the speed or acceleration.

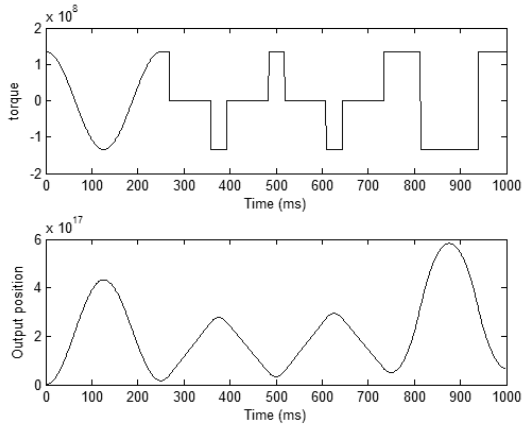

Fig. 12 Simulation result showing the output position as function of the input torque.

The waveform viewer is not adequate to observe the dynamic behavior of the developed model due to the inability to plot graphically non-digital values. A torque profile composed by a set of three different torque waveforms was produced to evaluate the motor model developed: cosine, pulse, and square. The model operating clock frequency was set to 1MHz and the size of the registers (N) was set to 64. A total of 1000 input samples was produced combining the three waveforms described above. The model receives an input sample every 1000 clock cycles (1ms). The total run time was 1s. The upper plot of Figure 12 show the input torque profile and the lower plot show the output shaft position response.

The output response position response was within the expected for the model developed. To evaluate hardware use and performance of the developed architecture, three different versions with the size of the registers set to 16, 32, and 64 of the proposed architecture was synthesized to a Xilinx FPGA. The chosen device was the XC2VP30. The synthesis results are presented in Table 1.

Table 1 Synthesis results

| ***N*** | **16** | **32** | **64** |
|---|---|---|---|
| **LUTs** | 33 | 65 | 129 |
| **FFs** | 49 | 97 | 193 |
| **$F_{clk}$ Max** | 269 | 226 | 171 |

The maximum clock frequency ($F_{clk}$ max) was 269 MHz for the 16 bit version. The number of FPGA resources used by the 64 bit version was less than 1% of the chosen FPGA. Comparisons to other works were not possible because no similar work was found in the literature.

## CONCLUSIONS

In this work a brushed DC motor model was proposed to replace an actual motor, its power drive and encoder. This model is to be used during the development cycle of motor control systems. The proposed model parameters can be adjusted to specific design needs.

The performance results show the proposed architecture is able to reach hundreds of megahertz when synthesized to a virtex II pro FPGA and using very few logic resources. At this frequency of operation the motor model can be assumed to be continuous time for the motor controller being developed, what lead to the advantage that small glitches such as unexpected control delay or instabilities in the controller realtime system can be catch by the motor model. Furthermore, the encoder output generates a realistic incremental encoder output that can be used to debug the encoder interface of the motor controller being developed

As a future work, improvements in the brushed DC motor model is planned to include realistic loss models, external load model and fine adjustment of the input torque. Hardware models for AC motors are also planned.

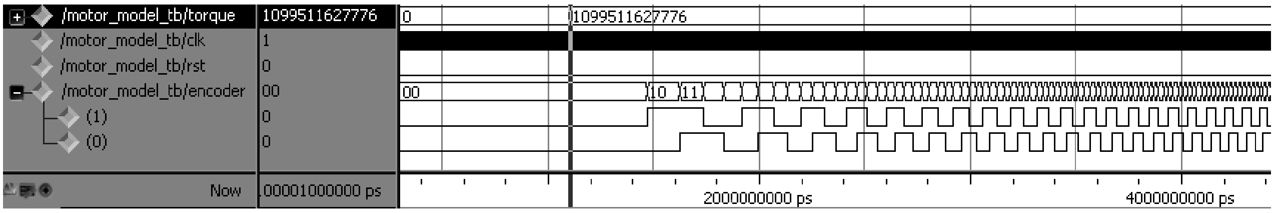

Fig. 11 Simulation result for a torque step.

## REFERENCES

[1] P. Wolm; X. Q. Chen; J. G. Chase; W. Pettigrew, *Analysis of a PM DC Motor Model for Application in Feedback Design for Electric Powered Mobility Vehicles*, Mechatronics and Machine Vision in Practice, 2008. M2VIP 2008. 15th International Conference on, *2-4 Dec. 2008 Page(s):640 – 645*

[2] S.J.; O. Vainio; J. Pasanen; *A 16/24-bit DSP-ASIC coprocessor for AC motor modeling,* Euro ASIC '91 27-31 May 1991 Page(s):53 - 56

[3] O.S. Lobosco, *Modeling and simulation of DC motors in dynamic conditions*; Electric Machines and Drives Conference Record, 1997, IEEE International, 18-21 May 1997 Page(s):MB2/1.1 - MB2/1.3