EMBEDDED SYSTEM WITH SOFT PROCESSOR FOR THE DRIVE SYSTEM OF AUTONOMOUS ROBOTS

ROTAR DAN[1*]

[1] *Vasile Alecsandri University of Bacău, Calea Mărăşeşti 156, Bacău, 600115, Romania*

Abstract: This paper presents a solution to improve mobile robot drives. This type of robots currently found in the smart environments is characterized by its own power source. It is therefore necessary that all elements of the system consume the electricity judiciously. The solution presented is based on an embedded system with a programmable processor designed to be configured with a field-programmable gate array (FPGA). Such a structure is highly flexible and allows the possibility to obtain greater performances including significantly reducing energy consumption.

Keywords: soft processor, embedded system, motor drive, FPGA, micro step, finite state machine.

1. INTRODUCTION

Modern drive systems use the embedded systems with performance CPUs that allow both the control of the DC motors and of the stepper motors. Such embedded systems contain specialized interfaces and CPUs that deliver greater performance. The performances refer to the dynamic parameters of the operation and their energy consumption.

Where such systems are used, the user is able to perform various control schemes by customizing the used software. The physical structure of such a system, however, is fixed and limited to the intended producer features.

The solution presented allows the user to modify both the software component of the embedded system and the hardware component [1]. This is possible by using a soft processor carried on a Xilinx Spartan 6 programmable logic array.

2. THE SOFT PROCESSOR

The soft processor is carried out as a finite state machine with data path (FSMD) [2]. A FSMD circuit consists of an automatic machine with finite number of states (FSM) plus the usual sequential circuits. The block diagram of such a machine is shown in Figure 1.

The FSMD operation is based on information provided by a program code stored in memory (input data). Initial state of the machine is represented by a fetch operation with the memory when the code of operation is read from memory.

[*] Corresponding author Dan Rotar, e-mail drotar@ub.ro

This command code determines the next state of the finite state machine (via the switching network) and performs the data flow processing established by that code (using functional units).

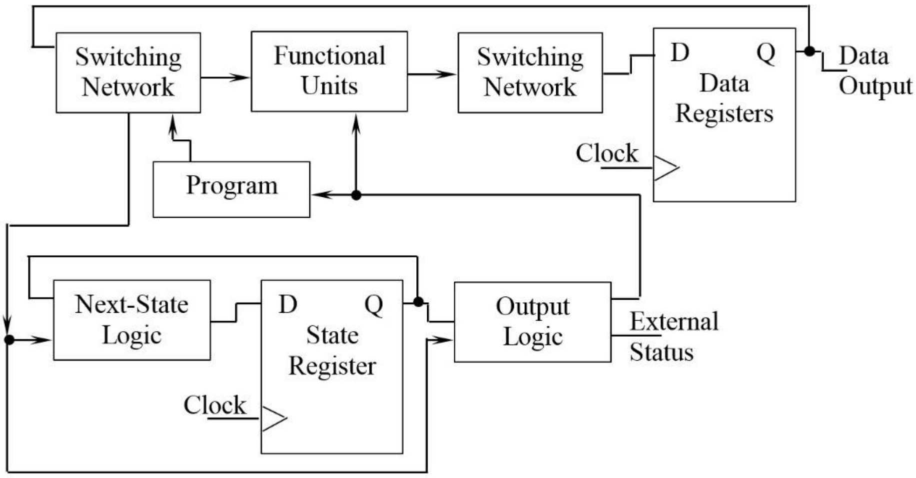

Fig 1. The block diagram of the soft processor.

At the end of the activities corresponding to the command code in execution, the finite state machine extracts the following operation code from memory by returning back to its initial state [2].

This structure was achieved by programming in VHDL language of a Xilinx Spartan 6 programmable logic matrix [3]. Such a structure has several advantages [7]. The possibility of reconfiguring the physical structure by modifying the program, the selection of the processor instruction set, achieving a balance between soft and hard components, achieving an economic functioning system, optimizing the performances etc.

3. THE stepper motor drive COMMAND

The stepper motor command drive is done by a separate block which is also synthesized in the FPGA structure. It consists of a pulse distributor block and a current control block. The distributor pulse block is capable of generating, depending on the command received from the central unit, different waveforms needed for the stepper motor in different operating modes. Current control block receives information from an external analog to digital (A/D) converter and deliver the data to the central unit [5].

For the application presented in this paper we'll discuss how to make the stepper motors command in micro stepping mode. Such a system provides the following advantages of a drive system with MPP: MPP deviation reduction, better stability of movement, movement with small oscillations, increasing the resolution by reducing the positioning step [4]. These advances are achieved under a constant torque or with slight variations.

In order to obtain the micro-stepping regime it is necessary that the successive phases of the motor be powered at the same time with the phase of signals carried out according to the micro step. MPP equations can be written as a matrix as follows:

$$[U_S] = [R_S][I_S] + \frac{d}{dt}[\Psi_S] \tag{1}$$

$$J\frac{d^2\theta_m}{dt^2} + B\frac{d\theta_m}{dt} + M_r = M_e \tag{2}$$

where: $[U_S]$ – the column matrix of the phase supply voltages; $[I_S]$ – the column matrix of the phase supply current; $[R_S]$ – diagonal matrix of the phase resistances; $[\Psi_S]$ – column matrix of the phase flows; J - total moment of inertia reduced to the shaft of stepper motor; B – the constant viscous friction coefficient; Mr -

resistant torque reduced to the shaft; Me - electromagnetic torque developed by the motor; θ_m – the instant mechanical angle of the rotor.

From these equations, it is noted that if two successive phases of the MPP are fed at the same time, then the developed torque moves the rotor in an intermediate position obtaining the micro-step to the value given by Equation (3).

$$\theta_\mu = \frac{\theta_m}{K_d} \quad (3)$$

where: θ_μ – micro step angle; K_d – divisor factor.

According to the equations (1) and (2) the division factor K_d is determined from the ratio of the currents supplied to the two adjacent phases. It follows that it is not possible to use the pulses for phases power supply and it is necessary to use a phase shifted sinusoidal voltage through adjacent phases with the control of phase current. The waveform required for the supply of the two adjacent phases in micro-stepping mode is shown in Figure 2.

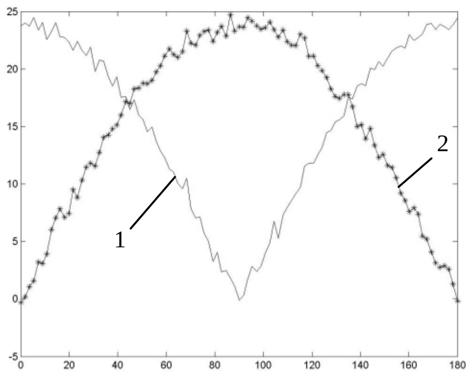

Figure 2. The currents waveforms of the stepper motor phases: 1-the current of (A) phase; 2-the current of the adjacent phase (B).

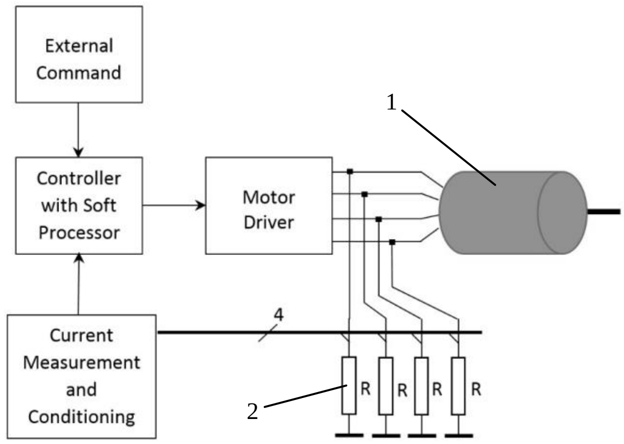

Figure 3. The diagram of the stepper motor command: 1-stepper motor; 2-current sensor.

To control the current in the windings of the stepper motor a current measurement circuit is used through its phases. The instantaneous value of the current is converted by an external analog to digital converter and the numeric value is sent to the soft processor for drive control. In this way, the driver circuit of the stepper motor control is made in such a way as to obtain the desired value of the micro step according to Equations (1), (2) and (3). By this method it is possible that the motor can be controlled in several ways.
The circuit can control stepper motor with full step or half-step up to the approximate position and then it can be commanded with micro steps in order to increase positioning accuracy. The schematic block diagram for driving a stepper motor is shown in Figure 3.

The system shown in Figure 3 contains a module for external commands that are sent from another computer system via UART (Universal Asynchronous Receiver Transmitter) or I2C (Inter-Integrated Circuit) interface and receives signals provided by the sensors of the robot. Also, it is possible through the interface to read certain values of the system, such as the number of steps, the amounts of currents in the motor phases, etc.

The module for processing and measuring the current of each motor phase eliminates parasitic voltages and converts analog current value in an 8-bit digital value.

The driver module is designed to provide the required voltages and currents in order to control the stepper motor. This module is constituted by an H-bridge with associated protection elements.

The basic module of this system is the controller carried out with the soft processor. This module generates the necessary signals for stepper motor control, achieves the current control through a chopper-type system and sets out the economic operation of the drive. This module allows the possibility of choosing multiple operating modes of the stepper motor through an automatic selection algorithm to program the pulse generator.

4. CONCLUSIONS

Using the embedded systems synthesized on the programmable logic array has several clear advantages. First, the designer has the freedom to choose a balance between the software and the hardware done. The hardware part provides a higher speed of the system under certain energy consumption [8]. The software does not consume electricity but, instead, may decrease the reaction rate of the system. Therefore, depending on the application that uses the system, the designer can decide which elements are the hardware and which are the software [9].

At the same time, during the operation, depending on the application, one can decide to reduce the energy consumption of the modules not needed at any given time. This is possible here as opposed to embedded systems which are exclusively physical.

For example, for the application presented in this paper, it was decided that the sinusoidal waveform synthesis would be carried out by the hardware. This was decided in order to increase the speed of execution. At the same time, this module can have multiple operating modes so as to reduce the energy consumption.

Another important aspect is the use of the soft processor. Processor structure is designed using a physical structure descriptive language, VHDL in this case, and it is possible that the size of the processor programs to be significantly optimized. The user can choose the appropriate physical structure and the instruction set used in concordance with the necessities of the application. In this way the program can be optimized in terms of its size and time of execution.

Such applications could achieve the superior performances when it is applied to the mobile robots. In the experiments, a mobile robot was used in order to supply the various execution elements (such as planting robots PCB parts). Such a robot requires the possibility of movement and requires a precise positioning under a relatively constant resisting torque. The experiments show that there was a reduction in the energy consumption by an average of 30% while the performance was actually improved.

REFERENCES

[1] Tong, J.G., Anderson, I.D.L., Khalid, M.A.S., Soft-Core Processors for Embedded Systems, Microelectronics, 2006. ICM '06. International Conference on Microelectronics, vol., no., 16-19 Dec. 2006, p.170-173.
[2] Chu Pong P., FPGA Prototyping by VHDL Examples, Wiley, 2008.
[3] Xilinx, Spartan-6 Family Overview, October 25, 2011.
[4] Ali Selk Ghafari, Mehdi Behzad, Investigation of the micro-step control positioning system performance affected by random input signals, Mechatronics, Volume 15, Issue 10, December 2005, Pages 1175–1189
[5] Ngoc Quy Le, Jae Wook Jeon, An open-loop stepper motor driver based on FPGA, Control, Automation and Systems, 2007. ICCAS '07. International Conference on Control, Automation and Systems, vol., no., 17-20 Oct. 2007, p. 1322-1326.
[6] Lysecky, R., Vahid, F., A study of the speedups and competitiveness of FPGA soft processor cores using dynamic hardware/software partitioning, Design, Automation and Test in Europe, 2005. Proceedings, Vol. 1, 7-11 March 2005, p. 18-23
[7] Ali, U., Malik, M.B., Munawar, K., FPGA/soft-processor based real-time object tracking system, Programmable Logic, 2009. SPL. 5th Southern Conference on Programmable Logic, 1-3 April 2009, p. 33-37.
[8] Purohit, S.S., Chalamalasetti, S.R., Margala, M., Vanderbauwhede, W.A., Design and Evaluation of High-Performance Processing Elements for Reconfigurable Systems, Very Large Scale Integration (VLSI) Systems, IEEE Transactions on Very Large Scale Integration, Volume: 21, Issue: 10, Oct. 2013, p. 1915-1927
[9] Chalamalasetti, S.R., Vanderbauwhede, W., Purohit, S., Margala, M., A low cost reconfigurable soft processor for multimedia applications: Design synthesis and programming model, Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on Field Programmable Logic and Applications, Aug. 31 2009-Sept. 2 2009, p. 534-538.