JIEE Časopis priemyselnej elektrotechniky
Journal of Industrial Electrical Engineering
ISSN 2454-0900
www.jiee.eu

Modifiable closed-loop control system for speed regulation of various DC motors with digital encoders

[1]*Matej BEREŠ*, [2]*Jozef DZIAK*, [3]*Daniel ŽENČUCH*,

[1,2,3] Department of Theoretical and Industrial Electrical Engineering, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Slovakia

[1]matej.beres@tuke.sk, [2]jozef.dziak@tuke.sk, [3]daniel.zencuch@student.tuke.sk,

***Abstract* — the paper provides a brief description of conventional and discrete PI controller. The goal of this paper was creating a PI controller that can be modified during the time of operation in proposed GUI. The main field of using this PI controller is speed regulation of a various DC motors with digital encoders. Such PI controller can be used for education purposes to showing what happen when the PI parameters are set incorrectly without removing or changing any physical parts. The paper also shows the main algorithm and the type of communication between the microcontroller and the proposed GUI.**

***Keywords* — ARM, control algorithm, DC motors, PI controller, speed regulation**

I. INTRODUCTION

Many control systems which can be used for speed regulation of DC motors exist in present days [1], [2]. Each commercially available controller can be used for specific DC motor. So it can't be modified or used for different motors. Therefore a proposed control system which can be used for various DC motors is presented. The goal of this paper is creating a control system that can be modified during the time of operation with help of proposed GUI (Graphic User Interface).

The following chapters briefly describes the function of proposed closed-loop system with using an ARM microcontroller and a GUI programmed in a Visual studio. Also the microcontroller will be sending the real value of RPM (Revolution per Minute) to GUI.

II. PROPOSED PI CONTROLLER

The control systems are widely used in present days. The main reason is because they can be useful in many applications. It is also well known that the control systems can be divided into two categories. The first category is an open-loop control system. This kind of the system doesn't know how input value influenced the output. The system does not consist of feedback loop. Therefore the second category of the control system removes this issue. The second category is called a closed-loop control system. This control system is obtaining feedback from the output. In this case the system measure output value and compare it with desire value. If an error occurs the system provide changes to ensure that the error is zero or close to zero. For the sake of simplicity the basic principle can be seen in the figure, Fig. 1.

Where the G1 represents the open-loop gains of the controller and is the forward path. G2 represents the gain of the sensor, transducer or measurement system in the feedback path. A transfer function of the closed-loop system can be simply determined by equation (1).

$$\frac{Output}{Input} = \frac{y}{w} = \frac{G1}{1 + G1 \cdot G2} \tag{1}$$

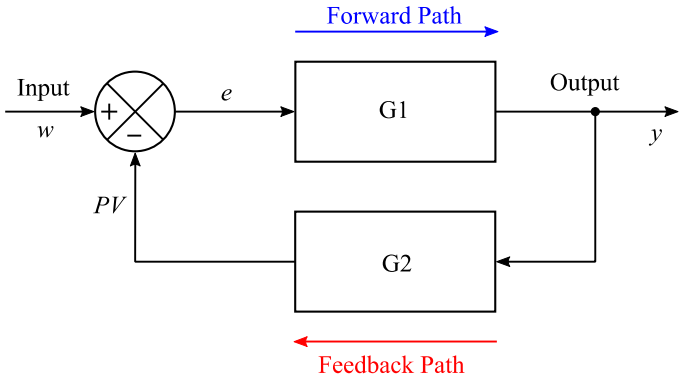

Fig. 1 Closed-loop system representation

The closed-loop control system can be created with the help of an analog electrical components such as an operational amplifiers or with a digital electrical components such as a microcontrollers. The digital devices provide more accurate and more stable results than analog devices. Also the control system which is using digital devices can be easily modified. Because of the bigger advantage of the digital devices, they will be considered for the purpose of the DC motor control. The principle of function of the DC motors is well known. Therefore the principle of function of DC motor will be not presented in this paper.

If the system will be composed with digital electrical component, it would be good to know what the difference between conventional and discrete control system is. The difference between conventional and discrete PI converter will be briefly described in following parts.

A. Conventional PI control

The PI controller is the most popular variation. The value of the controller output $u(t)$ can be determined by the equation (2).

$$u(t) = u_{bias} + K_c e(t) + \frac{K_c}{\tau_I}\int_0^t e(t)dt \tag{2}$$

The u_{bias} is a constant which can be set to the value of $u(t)$. This constant gives smooth transfer if the error is zero in time when the controller is starting. K_c represents gain constant of the controller and τ_I represent integral time constant. Higher value of K_c makes the controller more aggressive in responding to the difference between desired value and the measured value. The set point (w) is desired value and process variable (PV) is the measured value. The error is the difference between desired value and measured value which can be determined by the equation (3).

$$e(t) = w - PV \tag{3}$$

B. Discrete PI control

Digital controllers are implemented with discrete sampling periods and a discrete form of the PI equation is needed to approximate the integral of the error. This modification shown in equation (4) replaces the continuous form of the integral with a summation of the error and uses Δt as the time between sampling instances and n_t as the number of sampling instances.

$$u(t) = u_{bias} + K_c e(t) + \frac{K_c}{\tau_I}\sum_{i=1}^{n_t} e_i(t)\Delta t \tag{4}$$

The limited range of this paper does not provide more space to describing the discrete PI controllers more detailed. Therefore the next chapter will continue with other parts of this controller. The literatures [3]-[8] provide more information about a discrete PI control.

The basic rule is: the system must be known before proposing any type of conventional PI

controller. Therefore the suitable type of output measuring must be considered. Therefore the next chapter will briefly describe one way of RPM measurement.

III. Procedure for Speed Measure of DC Motor

DC motor speed measurement can be provided by many techniques. In case when the microcontroller is considered as a control part of the whole system the digital encoder is the right choice. The digital encoder exists in many variations in present days. It can be even more easily made. But in this case the DC motor with build-in encoder will be applied. The applied encoder generates 1250 pulses per rotation. Due of that the IC (Input Capture) mode of the microcontroller will be handling generated pulses.

The principle of function of IC mode is as follows: First of all, let's consider that the timer clock of the microcontroller is 72 MHz. The IC reset mode is using 16-bit timer/counter which increasing value from zero to 65535. When the pulse (rising edge) from the encoder occurs the timer/counter save the actual value to a CCR (Compute Compare Register) register and clear value of timer (CNT) to zero. The principle is illustrated in the figure, Fig. 2.

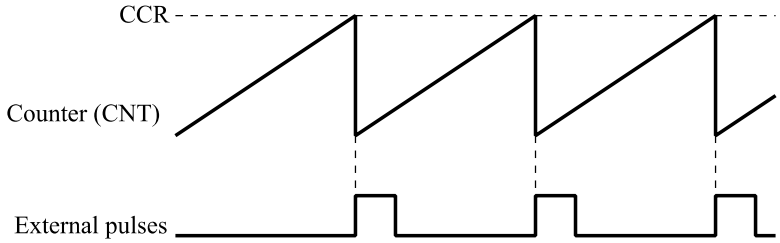

Fig. 2 IC reset mode of timer/counter

The timer/counter clears a counter value every time when the pulse from encoder occurs. Then the RPM value can be determined by the equation (5). Although the CCR register value is updated very quickly the algorithm is running slower. Updating the CCR register is running independently of the algorithm.

$$RPM = \frac{F_{clk}}{CCR \cdot k_P} \cdot 60 \tag{5}$$

Where F_{CLK} is timer/counter frequency, CCR is comparing register of timer which contain highest timer value, k_P is pulse number constant which represent number of pulses per one revolution coming from digital encoder. The constant value 60 converts result from revolution per second to revolution per minutes. This equation can be used for any type of MCU.

IV. Proposed Control Algorithm

The real value of RPM is known so now the control algorithm can be implemented. The whole C code consists of peripheral initialization, header files and so on. The describing them is not a subject of this paper. Therefore the main code where the main algorithm is running will be described more detailed. For better understanding the algorithm itself is illustrated in the figure, Fig. 3. From presented algorithm can be seen that the discrete PI control is implemented which is based on equation (4). In this case the number of sampling instances is set to100.

Firstly the algorithm measure real value of RPM, which is calculated from the equation (5). Then the output bias is determined. After that, the error is determined. The error variable consists of 100 values which are saved accordingly step by step. In next the sum of all determined errors will be calculated. With all known variables the output value (y) can be determined according of the equation (4). According of measurements the value of the dt is set to 1 ms.

The output value (y) is next compared with minimum and maximum value which represents a duty cycle of PWM signals connected to MOSFET drivers. If the output value is larger than the maximum value of the duty cycle, the duty cycle of the PWM signal is set to maximum. Same process for the

minimum value. The maximum and the minimum value have to be set to prevent the faulty function of generating control pulses. If the output value is between the maximum and the minimum value of the duty cycle, the *y* value is directly loaded into CCR registers of use timers that generates PWM signals.

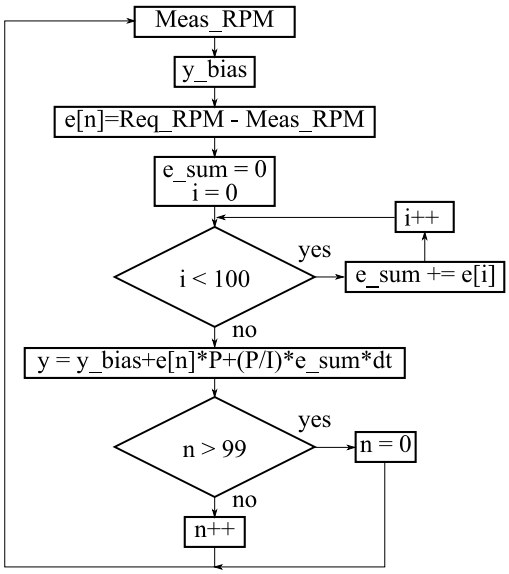

Fig. 3 The main algorithm of PI controller

The next part of the control system is GUI itself. Therefore the next chapter will firstly provide brief description about proposed communication than showing GUI itself.

V. THE GUI OF THE CONTROL SYSTEM

As it was mentioned in the beginning of this paper the GUI was created in an integrated development environment called a Visual Studio. The communication between MCU and PC is provided by universal asynchronous receiver-transmitter (UART). This type of communication can be easily implemented into the proposed control system. The principle of function of UART communication is not the subject of this paper. Therefore, only the transmitted and received data form will be presented.

Both transmitted and received data consists of three parts. The first part is initial character which indicates that a new data is arriving. In this case the initial character is '*&*'. The second part is the data itself, which can be also separate into another few parts. The third part is end character which indicated that all the data was transmitted/received. With this character the MCU or GUI knows that all the data was sent correctly and now the necessary operation can be provided. In this case the end character is '*'. For example the data form for DC motor to start with desired speed looks as follows: &R11500*. The character *R* indicates that the command is for DC motor rotation setting. The first number 1 starts timers for generating PWM signals. This PWM signals are connected to MOSFET drivers. The second number, 1 indicated the direction of rotation. The rest number represent the desired speed of motor. This number is saved into a *Req_RPM* variable which is constantly compared with the measured RPM value. A similar process is for sending regulation parameters P and I. Also the same process is for sending data from the MCU to GUI. The real value of motor speed is constantly sending into GUI after the MCU starts controlling algorithm. The illustration picture of the GUI can be seen if figure, Fig. 4.

The GUI is divided into few parts. The first part is a Serial port Configuration. With this part a serial port can be chosen and opened or closed. The second part is a command line. The command line is used for sending raw data to the MCU. This command line is used only for special cases. The third part is a Reports window. This window contains a list of all transmitted and received commands. It is used only for debugging purposes. The fourth part is an Engine control. The engine control provides control of speed, direction and regulation parameter. With the proposed control system the regulation parameters can be easily determined. For the sake of simplicity a detailed procedure for creating such GUI will not be presented here.

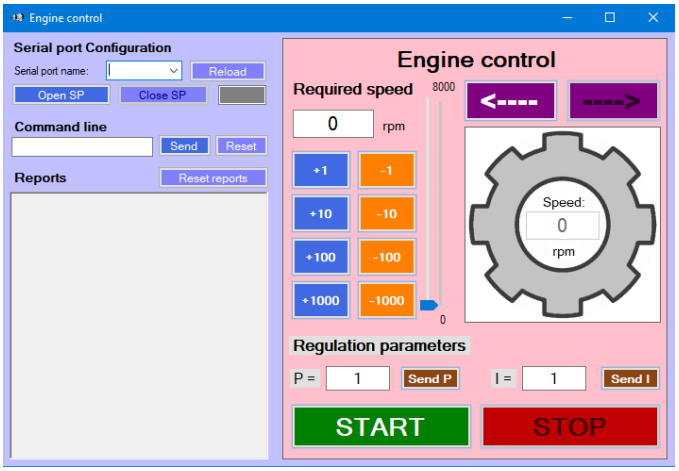

Fig. 4 Proposed GUI of PI controller

VI. CONCLUSION

The paper briefly described main parts of the control system which can be implemented for speed regulation of DC motors with digital encoders. Of course the power parts such as H-bridge and MOSFET drivers have to be considered. These parts can be chosen accordingly of DC motor power consumption. With proposed control system the regulation parameters can be easily determined. Although the proposed control system can be used for education purposes. The goal was achieved. In the next the GUI can be modified or change. Only the format data must be considered, to successful communication with ARM MCU. In addition, the GUI can provide a time diagram, and so on. It is many ways how to improve the whole system. The possible improvements can be described in another paper.

ACKNOWLEDGMENT

The paper has been prepared under the support of Grant FEI project FEI-2018-50.

REFERENCES

[1] HAMIDA, Mohamed Lamine, et al. Control of separately excited DC motor with series multi-cells chopper using PI-Petri nets controller. Nonlinear Engineering, 2018.
[2] EL MOUCARY, Chady, et al. Complete Design of a Hardware and Software Framework for PWM/Discrete PID-Based Speed Control of a Permanent-Magnet DC Motor Without Prior Knowledge of the Motor's Parameters. In: Recent Trends in Computer Applications. Springer, Cham, 2018. p. 153-173.
[3] BARTON, Paul I.; PANTELIDES, Constantinos C. Modeling of combined discrete/continuous processes. AIChE journal, 1994, 40.6: 966-979.
[4] PELTOMAA, Arto; KOIVO, Heikki N. Tuning of a multivariable discrete time PI controller for unknown systems. International Journal of Control, 1983, 38.4: 735-745.
[5] WU, Xuanlyu, et al. Simplified discrete-time modeling and dynamic characteristics analysis of PI-controlled voltage source inverter. In: Applied Power Electronics Conference and Exposition (APEC), 2018 IEEE. IEEE, 2018. p. 2914-2917.
[6] KHAN, Pathan Fayaz, et al. Design and Implementation of a Discrete-Time Proportional Integral (PI) Controller for the Temperature Control of a Heating Pad. SLAS TECHNOLOGY: Translating Life Sciences Innovation, 2018, 2472630318773697.
[7] XIE, Yuanlong, et al. Adaptive fractional order PI controller design for a flexible swing arm system via enhanced virtual reference feedback tuning. Asian Journal of Control, 2018, 20.3: 1221-1240.
[8] MAHVASH, Hossein, et al. DFIG performance improvement in grid connected mode by using fractional order [PI] controller. International Journal of Electrical Power & Energy Systems, 2018, 96: 398-411.