Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 53(67), Fascicola 1, 2008

PWM PLC Control of a DC Motor

Robert Pazsitka[1], Aurel Gontean[1], Septimiu Mischie[1]

Abstract – The goal of this paper is to introduce the control of a DC motor with a S7-200 SIMATIC PLC via a PWM signal. Experimental result has proven the validity of the setup.
Keywords: programmable logic controller, pulse width modulated signal, DC motor, algorithm.

I. INTRODUCTION

DC motors are widely used in industrial and consumer applications. In many cases precise speed control is an issue. The speed of a DC motor is directly proportional to the supply voltage. For example, if the supply voltage is reduced from 12 Volts to 6 Volts, the motor will run approximately at half the speed it would run at 12 Volts. We will obtain the voltage for the motor from a fixed voltage source. There are several ways to modify the actual voltage supplied to the DC motor. The first option is to use a variable resistor, which implies a poor efficiency. Another option (way more efficient) is to use a chopper (PWM controlled) in order to obtain a variable voltage [1].
If the motor is connected directly to the power supply, then it will start to spin at the moment the connection is made. It will take a small time to reach full speed. If the power is switched off before the motor reaches full speed, then the motor will start to slow down. If the power is switched on and off fast enough, the motor will run at some speed part way between zero and full speed. This can be done with a Pulse Width Modulated (PWM) signal to switch the motor on in a series of pulses.
In this paper the authors present the possibility to drive a DC Motor with a PWM signal generated by a Programmable Logic Controller (PLC) from Siemens SIMATIC S7-200 series.
In the next section we briefly present the PLC S7-214 used. In section III we describe the program and in section IV we present some of the experimental results. Our conclusions and future works are presented in the last section.

II. SHORT PRESENTATION OF THE USED PLC

The SIMATIC S7-200 series contain PLC's with compact design, possibility to expand and powerful instruction set. That makes the S7-200 controllers an ideal solution for controlling small applications [2]. The S7-200 CPU module combines a central processing unit (CPU) and discrete I/O points into a compact, stand-alone device. The S7-200 series are expandable with expansion modules, which contain additional inputs and outputs.
The inputs and outputs of the CPU and expansion modules are the system control points. The inputs monitor the signals from sensors and switches and the outputs control devices (motors, pumps, light) from the process.
The PLC program is executed as part of a repetitive process referred to as a *scan*. In a scan the inputs are read, the CPU executes the program, performs internal diagnostics and communication tasks and at the end of the scan cycle the outputs are updated. The CPU executes the program using the status of the inputs, processing and stores the data for controlling the automation task or process.
The S7-200 family includes a wide variety of CPUs. We chose the S7-214 PLC with a 24 V DC power supply. It contains 14 digital inputs, labeled with I, from I0.0 to I0.7 and I1.0 to I1.5, and 10 digital outputs, labeled with Q, from Q0.0 to Q0.7 and Q1.0 to Q1.1. The first number identifies the byte; the second number identifies the bit in the byte.
Two of the digital outputs, noted with Q0.0 and Q0.1 (port or byte 0, lines or bits 0 and 1), can be used to generate pulse train or PWM signals. The cycle time and pulse width can be set independently of each other. The pulse width corresponds to the length of time in which the output signal is 'high' during a cycle.
The status and control functions of the PLC are programmed through special memory bits. They also serve as a mean of communicating information between the CPU and user program. Special memory locations can be used as bits, bytes, words, or double words [2].

III. ALGORITHM AND SOFTWARE

The DC motor supply voltage is grown from 0 to a desired value, noted V_c, in a period of time denoted *Time*1, or the acceleration time. After *Time*1, for another period of time, denoted *Time*2, the supply

[1] Facultatea de Electronică şi Telecomunicaţii, Bd. V. Pârvan Nr. 2, 300223 Timişoara,
e-mail {robert.pazsitka, aurel.gontean, septimiu.mischie}@etc.upt.ro

voltage is maintained constant. In this period, *Time*2, the speed will be constant too.
The average value V_c of a PWM signal is given by

$$V_c = V_o \frac{T_{on}}{T_{PWM}}, \qquad (1)$$

where V_0 is the voltage value of the logic 1 output, T_{on} is the pulse width and T_{PWM} is the period of the PWM signal.
In the last period of time, denoted *Time*3, the supply voltage is reduced to 0. During this time period, the speed is decelerated to 0.
The DC motor is connected to the Q0.0 output where a PWM signal is generated. Within the *Time*1 period the pulse width is grown from 0 to a value directly proportional with the value of the desired supply voltage. In the *Time*2 period the pulse width is unmodified. In the *Time*3 period the pulse width is decreased to 0. A graphical representation of the desired shape for the supply voltage is shown in Fig 1.

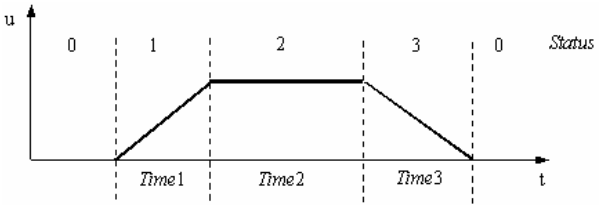


Fig. 1. The desired shape for the DC motor supply voltage.

A flag, called *Status*, which will mark the characteristic of the signal – zero volts (0), rising edge (1), constant value (2) or falling edge (3) is used.
To measure the time, respectively, to modify the pulse width an interrupt, namely interrupt 0 is used. With the Special Memory Byte 34 (SMB34) we control the period between two successive interrupt requests from interrupt 0. Due to the S7-200 limitations, the time interval, denoted T_{i0}, can be chosen from 5 ms to 255 ms with 1-ms increment. The interrupt 0 correspond to the event 10 and that event must be attached to an interrupt routine.
The number of steps used to increase the voltage from 0 to V_c within the *Time*1 period is given by

$$N_a = \frac{Time1}{T_{i0}}. \qquad (2)$$

The corresponding increase in voltage for each step will be

$$\Delta V_c = \frac{V_c}{N_a}. \qquad (3)$$

If in (1) T_{on} is replaced with Δt_a that represent the increase of the pulse length, and V_c with ΔV_c we obtain

$$\Delta t_a = \frac{V_c \cdot T_{PWM} \cdot T_{i0}}{Time1 \cdot V_0}. \qquad (4)$$

A similar equation can be obtained for the decrease of the pulse length corresponding to *Time3* period.
The I0.1 input, also labeled START button, is used to start the DC motor. If the I0.1 is activated with logic 1 (24V is applied to this input) then the motor starts to spin. The motor stops at the end of *Time*3 interval or if the input I0.0 (also labeled STOP button) is activated, If a rising edge is applied to I0.0 input, then the interrupt event 0 is activated. This interrupt has the higher priority in its class. We attach this interrupt event to an interrupt routine to force to stop the motor, in case we need to do this.
In the following we present the program used to command the DC motor that is connected to the Q0.0 output of S7-214 PLC.

Main program
In the first cycle scan the next 5 steps are executed:
1. Establish the time interval between two consecutive interrupts form interrupt 0 (event 10) – SMB34 = 6 – the time interval between two successive interrupt requests will be 6 ms.
2. Interrupt 0 routine – INT_0 – is attached to the interrupt event 10 (START)
3. Interrupt 1 routine – INT_1 – is attached to the interrupt event 0 (STOP).
4. The routine SBR_0 is called.
5. The interrupts are enabled.

For the next scans the main program is just waiting for an interrupt.

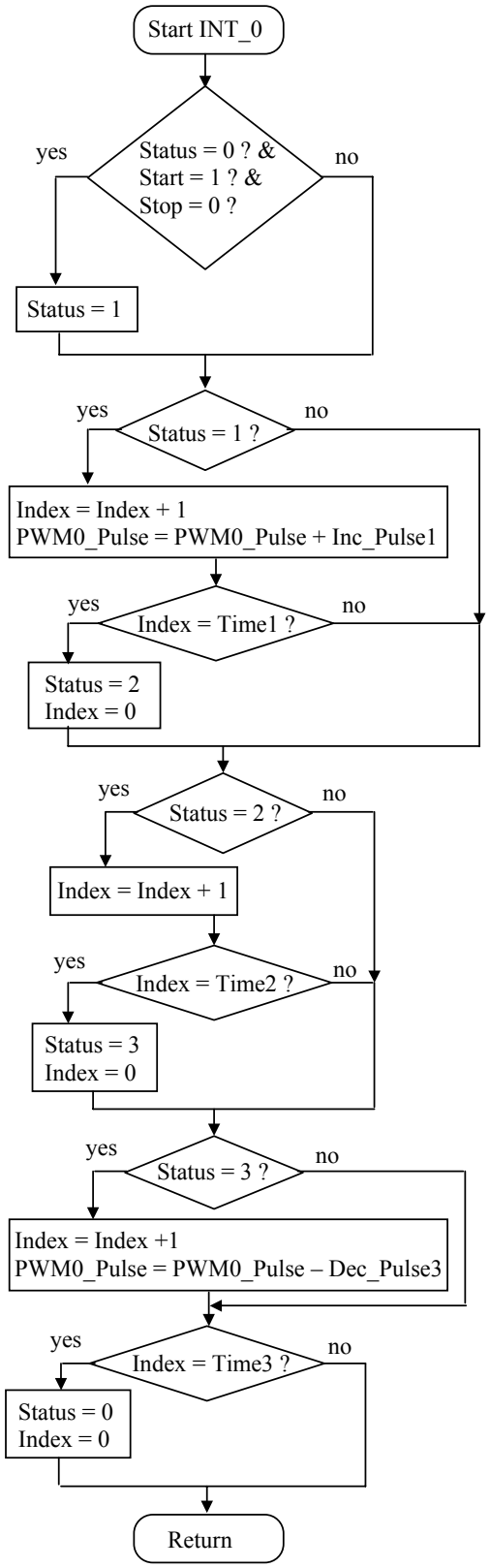
Routine SBR_0
The routine SBR_0 is used for initial settings. These settings are:
1. Programming the Q0.0 output to generate PWM signal. For that is used SMB67. The tick used is 1 μs.
2. The PWM periods will be 500 ticks · 1 μs = 0,5 ms => the frequency of PWM will be 2 kHz. It is used SMW68 (Special Memory Word) for the period programming.
3. The PWM pulse is initial 0. For that is used SMW70 noted *PWM0_Pulse*.
4. To Q0.0 output is activated the PWM function.
5. The time intervals *Time*1, *Time*2 and *Time*3 are selected (*Time*1 = 50 · 6 ms, *Time*2 = 100 · 6 ms and *Time*3 = 75 · 6 ms).
6. Setting the time increment for output pulse for *Time*1: *Inc_Pulse*1 = 9.
7. Setting the time decrement for output pulse for *Time*3: *Dec_Pulse*3 = 6.
8. Setting a temporary variable, noted *Index*, to 0. That variable will indicate if the time periods (*Time*1, *Time*2 respectively *Time*3) are ended.
9. Set the *Status* indicator to 0 (0 = Stop, 1 = Accelerate, 2 = Constant velocity, 3 = Decelerate).

Routine INT_1

The routine INT_1 will be executed if the STOP button will be pushed. The routine contains the next three settings: *PWM0_Pulse* = 0, *Status* = 0, *Index* = 0

Routine INT_0

The routine INT_0 will be executed periodically at every interrupts from interrupt 0. The logic diagram of the routine INT_0 is presented below.



IV. EXPERIMENTAL RESULTS

To verify if the PWM signal works within parameters at the output Q0.0 was connected a brushless motor, namely a fan. The used fan starts to spin at a voltage greater than 3 volts. The speed is increased proportionally with the average value of the voltage generated by the PLC.

In figure 2 we can see the output from one oscilloscope used to capture the PWM signal. This PWM signal represent a short period of time, which corresponds to the rising slope.

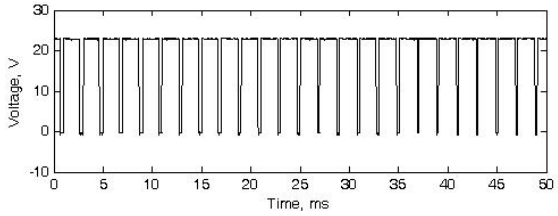


Fig. 2. The PWM signal for the rising slope, witch corresponds to the *Time*1 period.

In figure 3 we can see the same signal, this time on a short period of time, which corresponds to the falling slope, is presented.

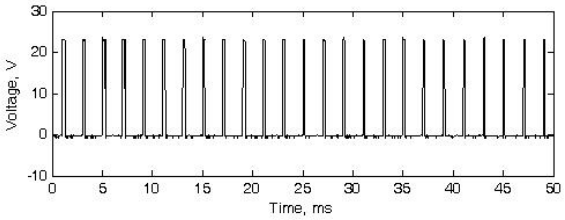


Fig. 3. The PWM signal for the falling slope, which corresponds to the *Time*3 period.

In figure 1 we represented the desired supply voltage for the DC motor. To verify that the average value of the PWM generated signal has the same shape with the desired signal, the PLC output is connected to a low-pass filter. The output of the filter is presented in figure 4.

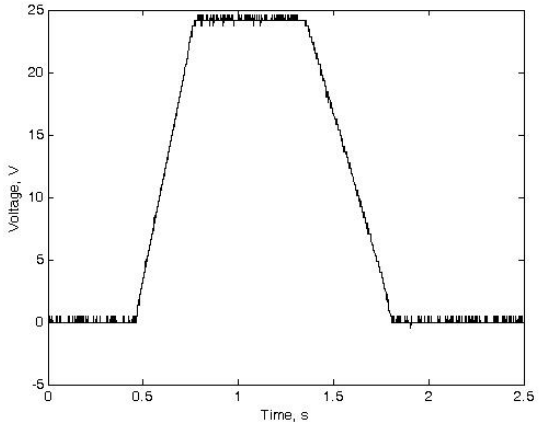


Fig. 4. The average value of the PWM signal applied to the DC motor.

V. CONCLUSIONS AND FUTURE WORKS

The shape of the average value of the PWM signal obtained to the Q0.0 output of the PLC, depicted in figure 4, corresponds to the desired shape. The DC motor ran very well at PWM signals with 2kHz frequency. At lower frequency of the PWM signals the DC motor generated unwanted noise at small velocity.

One advantage of using the PLC to generate the PWM driven signal for a motor instead of a dedicate devices is that the PLC can resolve and other needs, in parallel with the control of an electrically motor.

Another advantage to use a PLC is that the application can be implemented very easy on any other PLC because the programming language is the same or very similar.

If the DC motor starts to spin from a lower bound, higher than 0 volts, than the average value of the PWM signal must begin with the value of the lower bound. In this case the equations presented in this paper must be written again.

To facilitate the programming of the PLC, the calculus for the constants like the increase or decrease of the pulse length can be implemented in a program. This approach will offer for a workman or technician a simple and easy way to program the PLC.

REFERENCES

[1] Digital Current Loops and Direct PWM Control, DELTA TAU Data Systems, Inc., www.deltatau.com

[2] SIMATIC S7-200 Programmable Controller System Manual, Siemens AG 2005

[3] Barr, Michael. "Pulse Width Modulation," *Embedded Systems Programming*, September 2001, pp. 103-104.

[4] Ali Emadi, Handbook of Automotive Power Electronics and Motor Drives, CRC Press Taylor & Francis Group, 2005