7TH INTERNATIONAL CONFERENCE ON ELECTROMECHANICAL AND POWER SYSTEMS
October 8-9, 2009 - Iaşi, Romania

MODULAR H / S SYSTEM AS AN EXPERIMENTAL MICROCONTROLLED PLATFORM FOR THE STEPPING MOTORS

Dan S. MIHAI

University of Craiova, Romania, dmihai@em.ucv.ro

Abstract **– The paper is continuing a previous work on the design of a modular digital controller, dealing with the experimental platform and the associated real-time results. A modular and flexible easy-programming system is presented, allowing to the students to understand and to manage many hardware and software abilities of a RISC microcontroller. Both polling and interrupt strategies are put into action. The real-time requirements and data lead to specific relations for the on-line computations. The platform was tested for several stepper motors and functional views of the system are inserted. It offers possibilities in developing a large variety of functions and studies.**

Keywords: *Stepper motor, Microcontroller, Modular system*

1. INTRODUCTION

Nowadays, many educational projects concern the control system applications and try to find the best efficiency using the computer tools – [1], [8]. The author consider that even the best computer aided learning support will not be enough for a complete training of the future engineers and the experimental platform must remain a final training step - [6]. In a previous paper – [7], the main aspects were structured for the preparatory stage of an application concerning the digital control of a stepper motor by microcontroller. The problems identification, the off-line simulation under different programs were oriented to a target platform and associated software support: a modular system around a RISC processor (PIC family) and a very high level language (Flowcode), able to make both the simulation and the real-time code generation. Now, the essential details of the physical platform are discussed. The author presents also several own formulas for the on-line computations associated with the timer programming and the displayed data. The main goal is not only to give details of this experimental system, but to reveal a strategy for the practical training of the students in the electrical engineering field.

2. THE HARDWARE DESIGN

The students must know the basics on the stepper motors, in a practical manner, closed to an industrial presentation – [2], [3], [10]. It is very useful also to identify and study some typical applications – [9], [15]. The hardware approach implies the next steps:

- the choice of a control processor and, if possible, of a modular system with different boards. The author preferred the very popular (and successful) PIC family, the PIC16F277 microcontroller – [17] and the E-blocks system containing all typical boards – [16].
- the configuration of the system – fig.1, with a main processor(s) board, data acquisition boards, peripheral boards, power supply units;
- the choice of some stepper motors. Two models were considered: a “didactized” one – M1: 2-phase motor ASTROSYN N 351-1 / 15 ^{o} / 5 V / 0.16 A and a typical one – M2: 2-phase motor SANYO Type 4034900220 / 1.8^{o}/ 9 V TC / 0.4 A;
- the choice of the appropriate drivers: for the first motor, a L293D based driver board – [12] and for the second a local made board driver based on the circuit L298 - [13];
- the identification of all physical connections between modules / circuits and the adaptation of the binary control words, accordingly to these connection lines – fig. 2;
- the synthesis and the checking of the formulas for the timers programming;
- the transfer of the program code into the flash memory of the microcontroller;
- the real-time tests.

3. THE ON-LINE COMPUTATION FORMULAS

For a stepper motor having $N_{st/rev.}$ steps per revolution and maxim speed v_{max}, for a reference speed v^*_{ADC} (delivered by an analog digital converter ADC with n_{ADC} bits), the delay for a single step (that must be obtained by the timer programming of the μC) is:

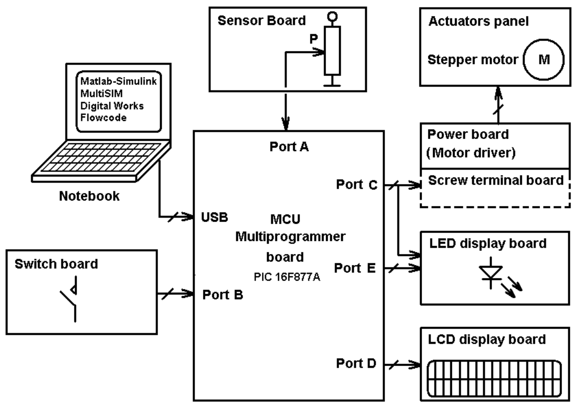

Figure 1: The modular system – hardware configuration.

$$\Delta t_{st}\ [s] = \frac{60 \cdot (2^{n_{ADC}} - 1)}{v^*_{ADC} \cdot N_{st/rev.} \cdot v_{\max}[rev./\min.]} \quad (1)$$

Another formula takes into account a correlation with the speed range intended for the motor and a factor k, depending on the control sequence type (full or half step):

$$\Delta t_{st\,x}\ [s] = \frac{60}{k \cdot N_{st/rev.} \cdot v_x};\ v\ [rev./\min.] \quad (2)$$

$$\Delta t_{st\,x}[s] = v_{\min} + \frac{v_{\max} - v_{\min}}{v_{ADC\max} - v_{ADC_{\min}}} \cdot \quad (3)$$

$$\cdot (v^*_{ADC\,x} - v_{ADC_{\min}})$$

Another way to compute the timer programming is given by the formulas:

$$\Delta t_{st.}\ [ms] = k\,\frac{\Delta t_{timer\max.}[ms]}{v^*_{ADC\ \lim}} \quad (4)$$

$$1 \le v^*_{ADC\ \lim} \le 2^{n_{ADC}} - 1$$

A null reference is detected and treated in a specific way. k is a scale factor, including the adaptation to the control sequence (full or half step).

A distinct formula is necessary for displaying the motor speed. A simple way is to give an image in percentage:

$$v_{LCD\ x}\ [\%] = v^*_{ADC} \cdot \frac{100}{2^{n_{ADC}} - 1} \quad (5)$$

It is possible to use other formulas too, without any analytical justification but verifying some minimum requirements.

The formulas were adjusted for the on-line control, based on real data and constraints. For example, by means of a simple Flowcode program, for the M1 stepper, were found with a logic analyzer (or with a standard oscilloscope):

- the minimum step delay (at the limit frequency, without lost of steps): $\Delta t_{st,\ min}$ = 7 ms and the according maximum speed: v_{max} = 440 rev. / min;
- the possible speed range, considering the maxim and easy programmable time delay at $\Delta t_{st,\ max}$ = 999 sec, as 1: 176,000!
- a reasonable time delay range, with $\Delta t_{st,\ max}$ = 999 ms, ensuring v_{min} = 2.5 rev. / min – hence a speed range of 1: 176 < 255, so the possible control range on 8 bits is not entirely reachable;
- a tuned time delay scaling, as follows:
  - for 10 bits control (ADC resolution of the speed reference): $\Delta t_{st,\ min}$ = 7 ms; $\Delta t_{st,\ max}$ = 7,161 s ;
  - for an exact 8 bits control: $\Delta t_{st,\ min}$ = 7 ms; $\Delta t_{st,\ max}$ = 1,785 s;

SQ.	Ph. A1 BROWN Pin L293D IN 7 / OUT6 Pin HPACT B0 – I0	Ph. A2 BLACK Pin L293D IN 10 / OUT11 Pin HPACT B3 – I3	Ph. B1 YELLOW Pin L293D IN 2 / OUT3 Pin HPACT B1 – I1	Ph. B2 ORANGE Pin L293D IN 15/ OUT14 Pin HPACT B2 - I2	Control words 4 bits Lower nibble
1	+	-	0	0	0 0001 - 1
2	+	-	+	-	0 0011 - 3
3	0	0	+	-	0 0010 - 2
4	-	+	+	-	0 1010 - 10
5	-	+	0	0	0 1000 - 8
6	-	+	-	+	0 1100 - 12
7	0	0	-	+	0 0100 - 4
8	+	-	-	+	0 0101 - 5

Figure 2: The connection data between modules / motor and the control binary words for the half step sequence.

- for other alternative scales, like: $\Delta t_{st,\ min}$ = 6 ms; $\Delta t_{st,\ max}$ = 6 s, in accordance with the stepper capabilities and closed to the 10 bits control range;
- an optimized on-line computation for the µC arithmetic unit with integer numbers on maximum

15 bits:

$$\Delta t_{st\,x}[ms] = \frac{29034}{20 + 10 \cdot V_{ADCx}} \qquad (6)$$

having a special program branch for null reference; the Δt_{st} range is from 3 ms to 966 ms, covered by an 8 bits control and extending experiments beyond the limit speed of the stepper.

Some formulas must be adapted to the physical connections of the hardware. For example, because the sensor board takes the analog signal between the +Vdd and the potentiometer (and not to the ground), the formulas must be modified as follows:

$$\Delta t_{st.\,base}[ms] = \frac{\Delta t_{timer\max}[ms]}{(2^{n_{ADC}} - v^*_{ADC\lim})} \qquad (7)$$

A good scaling for (7) is $\Delta t_{timer\max} = 1023$ ms, giving a full 10 bits range: $\Delta t_{st,\,base} = 1$ms…1023 ms. A similar scaling is possible for an 8 bits range control. Again, a null reference is detected and treated in a specific way. k is a scale factor, including the adaptation to the control sequence (full or half step).

A distinct formula is necessary for displaying the motor speed. A simple way is to give an image in percentage:

$$v_{LCD\,x}[\%] = v^*_{ADC} \cdot \frac{100}{2^{n_{ADC}} - 1} \qquad (8)$$

A correction must be made for this formula too, if the physical connection of the potentiometer is made to the + Vdd:

$$v_{LCD\,x}[\%] = \left(2^{n_{ADC}} - v^*_{ADC}\right) \cdot \frac{100}{2^{n_{ADC}} - 1} \qquad (9)$$

4. THE EXPERIMENTAL PLATFORM

Fig. 3 presents the image of the platform equipped with E-blocks modules – [16], for the M1 motor. The processor board has an application unit (PIC16F877) and an additional PIC processor for the programming in system (ISP) of the main PIC through USB.

The B port is an input digital port and four push-buttons are used as follows:

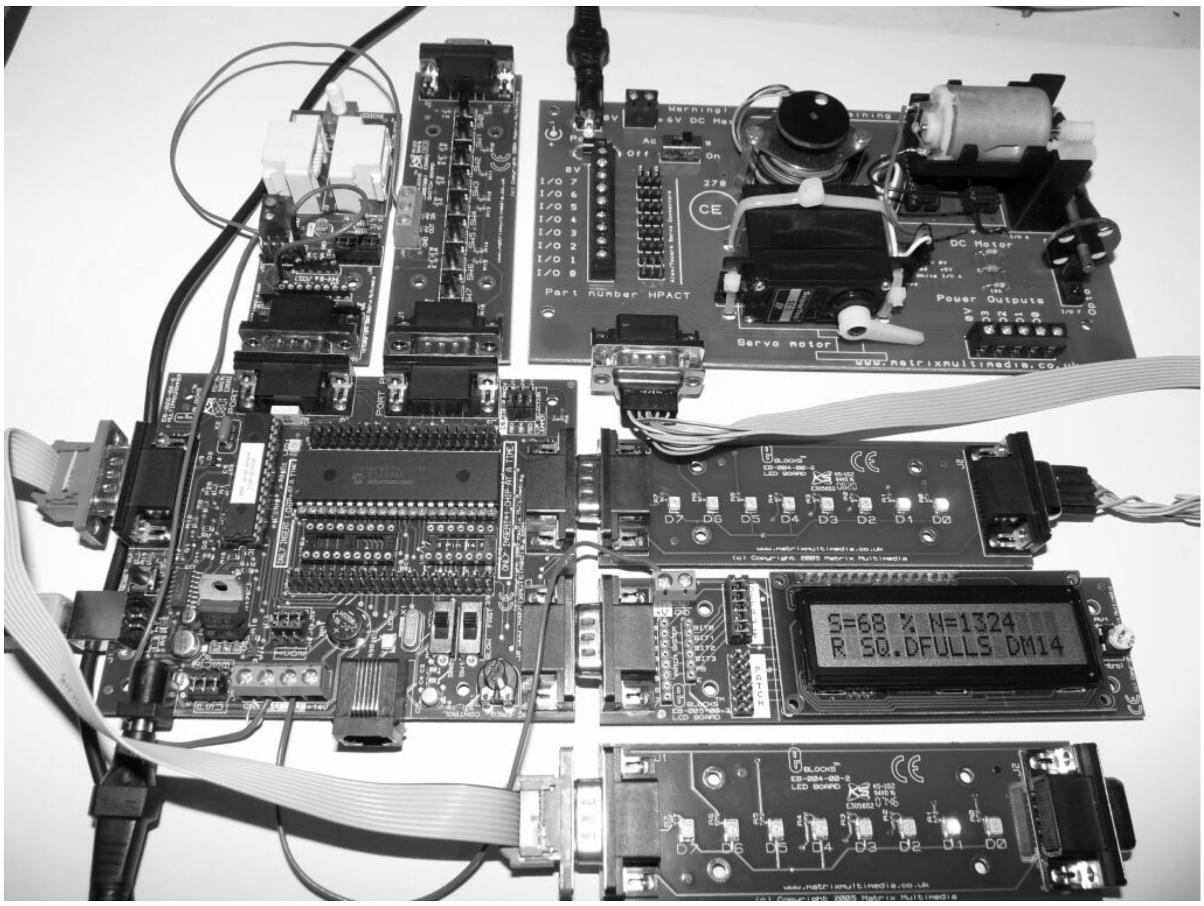

Figure 3: The experimental platform – first configuration.

B0 - Start/Stop, acting on the hardware interrupt system;
B1 - Motion Direction (first push - Forward, next push -Reverse);
B2 - Control sequence (first push - Full step single phase supplied, next push - Full step two phases supplied, next push - Half step);
B3 - Erase LCD display (on request).
The C port (digital output) has 2 functions:
- the delivery of the binary control words, sent to the motor driver;
- viewing some binary states by LEDs:
    - C0 ... C3: State of each motor phase;
    - C5: Reverse motion;
    - C6: Direct (forward) motion;
    - C7: Run / Standstill of the motor.

The D port is connected to the LCD display for supporting the next data viewing:
- motor speed, in percentage;
- the steps number, counted (up / down) with sign, upon the motion direction;
- the motion direction (D / R);
- the sequence type;
- the program version.

The three bits of the E port confirm the sequence type (one LED for each control sequence).
A second version was prepared for the motor M2, with an independent driver (based on the circuit L298 – [13]) – fig. 4.

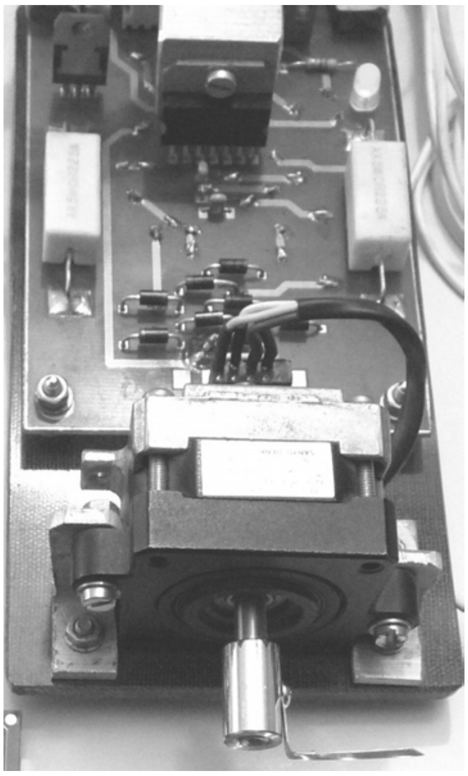
Figure 4: The second driver / stepper

The platform requires 2 power supply units: a first one for the microcontroller board plus the standard peripherals and another for the power board. Fig. 5 presents two captions of the LCD display, in different operating conditions.

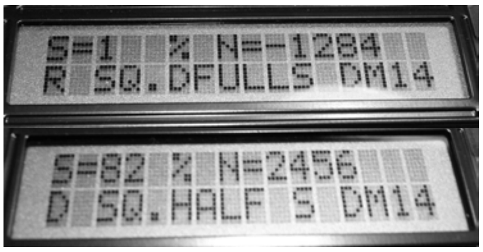

Figure 5: Two LCD display captures in different operationg conditions.

5. CONCLUSIONS

The experimental platform validates entirely the design requirements and its behaviour is very closed to the virtual operation under the graphical programming language. Several useful computation formulas were proposed and they has been confirmed by the real-time results. The platform is intended to be an open system for new functions, operation modes and applications. It could be an interesting challenge for the students in finding new ideas for extending the abilities of the platform. Some examples, in this meaning, are: the overcurrent protection, the gradual acceleration /deceleration to the set-point speed, a preset position etc. The merits of this work, with an initial multi-approach design and a final experimental modular system, are
related to several aspects:
- the students must follow a preparatory stage, necessary for a good understanding of the requirements, data, tasks and useful tools for testing in advance (by computer simulations) the control strategies;
- the implementation on real-time software of some algorithmic state machines proves to the students the utility of some so-called "theoretical" methods;
- the possibilities to make a fast developing software / hardware cycle, from new data / functions to the experiment.

References

[1] Bex S., Doclo S., Ysebaert G., Gielen G., Dehaene W., Man H. De, Moor B. De, *The PeopleMover Educational Project, Design, simulation, and implementation of a "real" control system application*, IEEE Control Systems Magazine, Oct., 2004.

[2] Condit R., Jones D. W., *Stepping Motors Fundamentals*, AN907, DS00907A, pp. 1-20, Microchip Technology Inc., University of Iowa, 2004.

[3] Douglas W. Jones, *Control of Stepping Motors - A Tutorial*, The University of Iowa, Department of Computer Science, 1998.

[4] Lian M., *3D Stepper Motor System and its GUI Design*, Thesis for Masters degree, Imaging, Robotics, and Intelligent Systems

Laboratory, Dept. of Electrical and Computer Engineering, The University of Tennessee, USA, 2004.

[5] Mihai D., *Comenzi numerice pentru sisteme electromecanice*, Ed. Did. Nova, Craiova, 1996.

[6] Mihai D., *Virtual vs. Experiment, Programmable vs.Wired Logic, Hardware vs. Software in Teaching Digital Control for Electromechanical Engineering*, Proceedings, IEEE EUROCON - Computer as a Tool, Ljubljana, Slovenia, 2003, Vol. II, pp 7-12.

[7] Mihai D., *A Multi-approach Design in Training Motion Control for the Stepping Motors*, SIELMEN, Iasi, Romania, 2009.

[8] Molengraft R. van de, M. Steinbuch, B. de Kraker, *Integrating Experimentation into Control Courses*, IEEE Control Systems Magazine, Febr. 2005

[9] Payet Burin P., *Bipolar Stepper Motor Control*, Application Note AN266/0189, SGS-THOMSON Microelectronics, 1995.

[10] Pinson G., *Physique Appliquée, Moteur pas à pas* - C35, http://www.syscope.net/elec/C35, p.1-5.

[11] Yeadon W. H., Yeadon A. W, *Handbook of Small Electric Motors*, McGraw-Hill, 2001.

[12] *L293D Quadruple Half-H Drivers*, Texas Instruments Incorporated Dallas, Texas 2002.

[13] L*298 Dual Full-Bridge Driver*, STMicroelectronics, October 1998.

[14] *Drivers and software*, CDRom ELSAM, Matrix Multmedia, London, 2008.

[15] *Driving Unipolar Stepper Motors Using C51/C251, C51/C251,* Application Note, Rev. 4214B–8051–12/02, Atmel Corporation 2002.

[16] *E-blocks Solutions*, Matrix Multimedia, London, 2006, http://www.matrixmultimedia.com/eblocks/index.php.

[17] *PIC16F87XA, Data Sheet DS39582B*, Microchip Technology Inc., 2003