Hardware Experiments for Simultaneous Control of Several Motors by a Microcontroller and a VHL Programming Language

Dan Mihai
University of Craiova / Department for Engineering in Electromechanics, Environment and Industrial Informatics
Craiova, Romania, e-mail: dmihai@em.ucv.ro

Abstract - **The paper presents a hardware solution for the digital control of a multi-motor drive system using a very high level programming language and a single microcontroller. It follows a previous one for developing the software design. These two papers are related because each side (hardware / software) is interrelated and linked to each other. The hardware platform as well as the results recorded during on-line tests by means of a logic analyzer are presented and analyzed. The purpose of the research is the simultaneous control of several motor units with their own motion parameters using a fast design cycle for the software support and a minimal hardware configuration. The main contribution refers to a software / hardware solution for obtaining independent / parallel control of the time delays for several channels. The programming environment is Flowcode 5. As hardware, a modular platform based on a PIC microcontroller is used. The application implements a simultaneous control of 2 different steppers and 2 different DC motors, each of them with different adjustable speed and its own motion direction. Another contribution is a practical tool for revealing the entire on-line timing by recording all real-time tasks. The obtained results are useful both for a qualitative check as well as for precise quantitative evaluations, inserted examples being illustrative.**

Keywords: *multi-motor, motion control; microcontroller; Flowcode; on-line timing.*

I. INTRODUCTION

During the last decades, the motion control field had benefit of many performance solutions: ASICs, microcontrollers, DSP-controllers, FPGA, PSOCs. It is not at all a simple task for a designer to optimize his options, accordingly many criteria. Sometimes the chip manufacturers make not easy this choice task because their products offer too many operation modes and facilities.

The main solutions to control motors are grouped in:

- chip level: specialized chips (ASIC), microcontrollers and DSPs more or less dedicated, PLD (CPLD, FPGA);
- system / equipment level: multi-processor systems, industrial servodrives.

The study is focused on multi-motor control, an important problem in robotics, machines-tools, electrical vehicles, drones and many others.

Next examples give an image for the aspects and solutions concerning the multi-motor control, from chip to systemic approach.

L 6460 [11] is a very complex hardware programmable IC in a 64 pins chip, with the possibility to drive simultaneously stepper and DC motors, with control and drive multimotor sections. With 4 full bridge driver configurable, the chip is able to control simultaneously two DC and one stepper motor or four DC motor. It can communicate with an external microprocessor by using an integrated slave SPI. It is quite difficult to manage the whole signal / selection / programming operation mode set and an extra MCU is necessary to exploit this kind of ASIC. The main advantage remains the embedding of several power stages with all protections included (max operative current = 2.5 A / 1.5 A).

Paper [4] explains how correctly designed modular FPGA based multi motor controls can produce flexible and customized control systems with a very short development time compared to using modern Digital Signal Processor (DSP) technology. Unjo's Gepard - platform, built around a true parallel FPGA technology, is able to drive many types of motors: DC, Stepper, BLDC, PMSM, SRM, asynchronous etc.

Some multi-motor controllers are delivered like a chipset. The MC5x000 series (Magellan family from PMP [15]) controls up to four axes of DC brush, brushless DC, or step motors in a two-IC chipset. Motion Control ICs provide a flexible and powerful instruction set to initialize and control motion axes, monitor performance and synchronize overall machine behavior. As programming tools, are available both a GUI to facilitate an easy way to graph and analyze system performance and C-Motion support that allows to develop an application using C/C++. The chipset must be surrounded by a host processor and a power amplifier.

Implementation issues related with the limited number of dedicated multipliers of a FPGA solution were overcome in [1] using an efficient computational block, based on resource sharing strategy. The developed IP Cores were carefully optimized to fit in a low cost XC3S1000. Experimental results, obtained with a multi-motor EV prototype, demonstrate the proper operation of the proposed propulsion system.

Procedures to tune controller of individual loop and for an overall tuning of a system with several motion axes are given by the paper [2].

As for the power interface, [3] uses an IC with three high side FET pre-drivers and three low side FET predrivers, interfaced to a MCU (via six direct input control signals, an SPI port for device setup and asynchronous

reset, enable and interrupt signals). All that makes the programming control task not quite easy.

With the same "multi-motor" term, some industrial equipment concern the starting of more than one motor with the same soft-starter, with the purpose of reducing the cost of the motor starting system [12].

Because modern microcontrollers have many hardware / software features, from many hardware options, the author preferred the solution with a single 8 bits microcontroller working inside a modular platform. The proposed structure still has availability for more complex applications.

First, the paper presents the block diagram of the platform. For the hardware support, the author used E-blocks modules [8]. An additional interface board, locally designed / manufactured was necessary for the two DC motors. Then, the results of several real-time tests are inserted, both for internal operation proofs and for macroscopic signals associated with the motors.

The hardware platform confirms the simulation results from [7], where the software support was designed using Flowcode 5 (IDE / VHL programming language). Previous experience of the author for designing a multifunctional sequential controller in Flowcode 3 for a stepper (described in [5]), although useful, was not enough for extending the control algorithm to several motors, where specific problems arise.

II. THE HARDWARE PLATFORM

Fig. 1 presents the block diagram of the experimental platform. An image of the electronic boards is given by Fig. 2 and Fig. 3 is for the multi-motor unit.

MCU system is built around a PIC16F1937 processor [13]. The module is a multiprogrammer board [8], connected to computer by an USB port.

The control block (CB) board uses the A port of the processor:

- 3 analog inputs A0, A1 and A2 (speed references for the steppers and for the DC motors – these last two having the same reference but the speeds are in complement to 8 bits);
- 5 digital inputs (A3- A7) for:
  - Start / Stop program;
  - Directions of each of the 4 motors.

The port B is dedicated to the control of the 2 steppers. There are many hardware solutions for interfacing the steppers to a microcontroller, some of them presented in [6].

D1 is a driver / power board [14] from the E-blocks family, based on L293 IC [9].

The step by step motors are different size, construction, parameters and control sequence – see [7]. For the main stepper, in this work, the author operates with both kind of control sequence (half / full step). The designer must avoid the use of elements like motors without a full datasheet. However, it happens often in applications. The main stepper is fully described in [16], while the second has only the labeling data.

The port C is split into 3 sections:

- the first 2 lines deliver PWM pulses for the speed control of the two DC motors;
- the next 4 bits are allocated for the control of the 2 bridges of the driver IC L298 [10], controlling Direction and Enable signals;

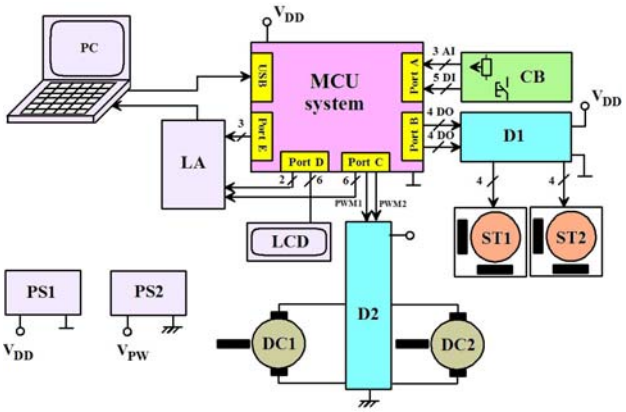

Fig. 1. The architecture of the experimental platform.

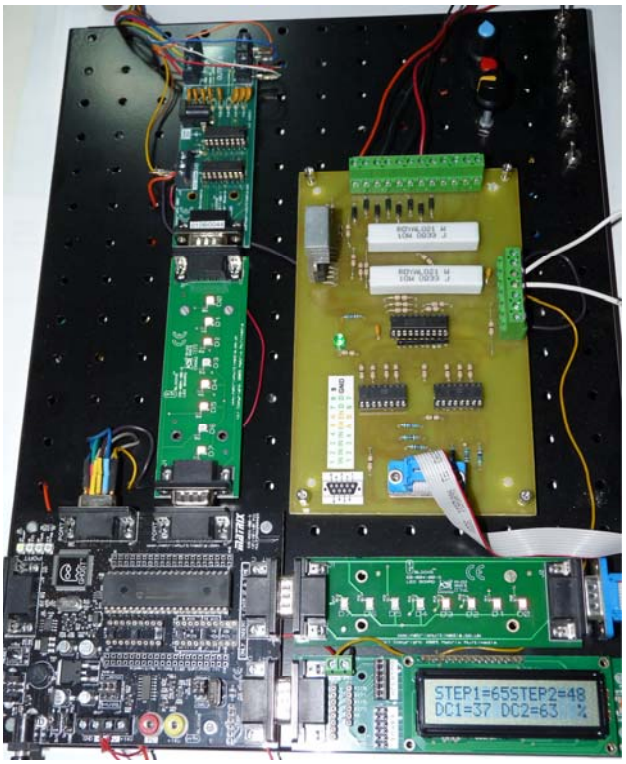

Fig. 2. The control board and the interfacing units.

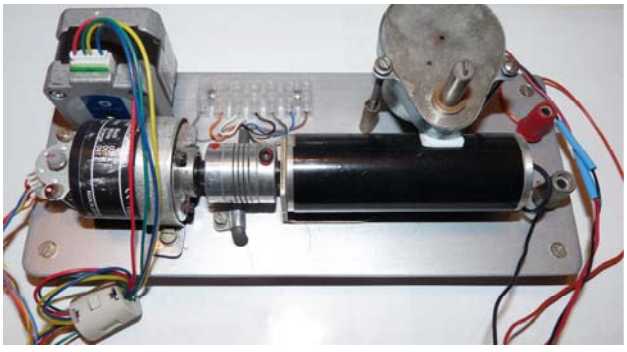

Fig. 3. The multi-motor unit.

- 2 bits will be markers for the interrupt signals of the timers TMR0 and TMR6, responsible for the delays that provide the speed of the two steppers.

D2 designates a driver board based on the power IC L 298, with galvanic separation by optocouplers.

The port D is split into 2 sections:

- 6 bits for the LCD display (alphanumeric 2 x 16 characters);
- 2 bits as markers for the A/D conversion task, respectively for the digital acquisition task.

The port E has available only 3 lines, used as follows:

- the DC control task;
- the LCD task.
- the whole control loop.

For having available all port pins, the author had programmed the internal oscillator, giving up to the external quartz circuit. Otherwise, two pins of the port A had to be connected to the external circuit.

LA from Fig. 1 is a logic analyzer (pod & software) that allows the on-line recording on a computers of all pin signals of the control processor.

Fig. 4 presents the on-line data displayed and an image of the logic analyzer probes around the microcontroller for on-line records.

Two power supply are necessary:

- PS1 (12 V DC) for the microcontroller board and for the steppers;
- PS2 (24 V DC) for the DC motors.

The author preferred battery type sources.

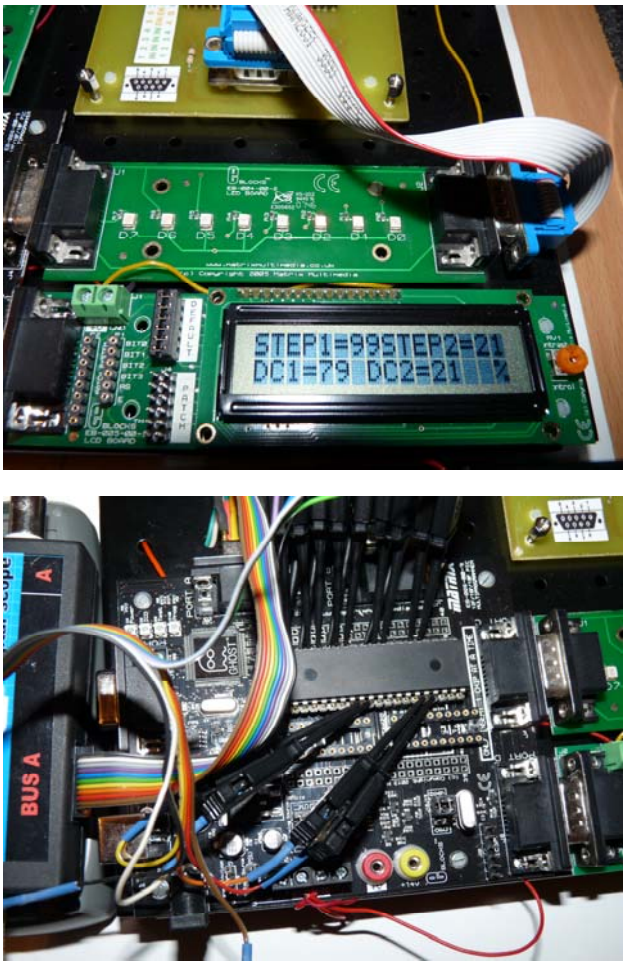

Fig. 4. Details of the hardware platform.

III. Experimental results

Next results are timing diagrams recorded in real-time by the logic analyzer for different operation conditions / parameters. The control program in Flowcode was designed so that these signals are available for the logic analyzer probes.

Fig. 5 reveals the interrupt signals delivered by the timers T0 and T6 that control the step delays, so the speed of the two stepper motors. It is possible to check the real-time timing versus the programming parameters of these timers:

The second on-line record from Fig. 5 gives the image the marker pulses of 3.21 μs.

Taking into account the operation mode of the microcontroller in interrupt mode (finishing the instruction in progress and some program counter – stack memory exchanges, it can say that there is a perfect match between the programming assumptions and the on-line timing. Indeed:

$$f_{INT\,Timer\,0/6} = 3900\ Hz \text{ [7].}$$

$$T_{INT\,Timer\,0/6} = \frac{1}{f_{INT\,Timer\,0/6}} = 256.5\ s - programmed \quad (1)$$

$$T_{INT\,Timer\,0/6} = 256.9\ \mu s - real \quad (0.15\ \%\ \text{error})$$

Fig. 6 contains the recording of the whole loop of the main program. Most of the time is necessary for displaying the data to the LCD, a secondary task that could be minimized. Anyway, it doesn't affect the interrupt timing where the speed updating is made.

A more detailed view for the real-time tasks is presented by Fig. 7. The useful tasks take around 300μs (with a better acquisition clock and a zoomed view, the exact value is 296.1 μs). Most of the time is consumed by the A/D conversion for 3 analog channels; then; the digital acquisition DIG_ACQ (5 channels) and the control of the two power bridges of the L298 driver – DC-CTRL are much faster.

As it was mentioned in [7], the microcontroller PIC16F1937 was driven by its internal oscillator at 32 MHz, close to upper operational frequency limit. By that, in addition, two lines of the A port (A6 and 7), usually allowed to the external quartz, are made free for I/O connections.

A quite interesting analyze could concern the interaction between several interrupts, aspect that could become a real constraint and challenge when more timers / interrupts are involved and a prioritizing procedure must be put in place.

Fig. 8 contains all 10 control signals for the motors:

- first 4 for the stepper 1;
- next 4 for the stepper 2;
- next 2 for the 2 DC motors (PWM pulses).

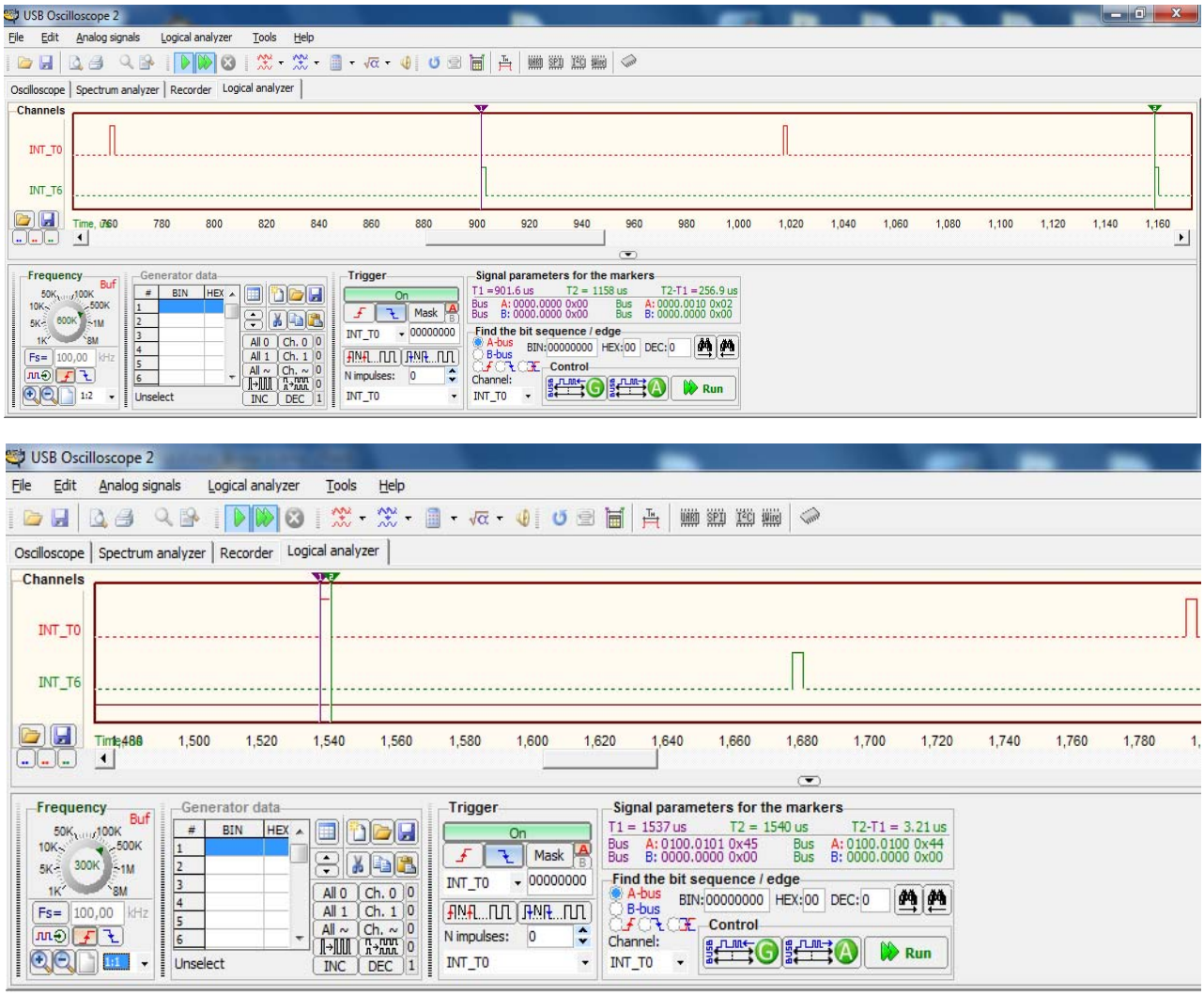

Fig. 5. The interrupt signals delivered by timers T0 and T6.

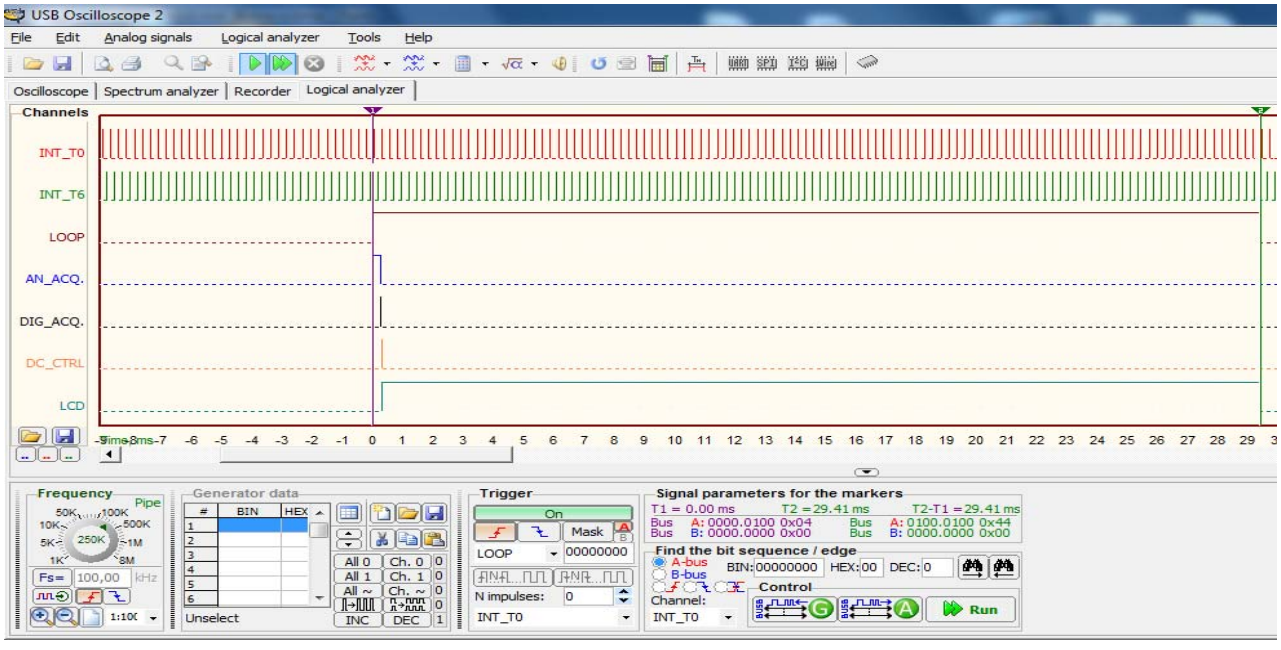

Fig. 6. The whole real-time loop timing.

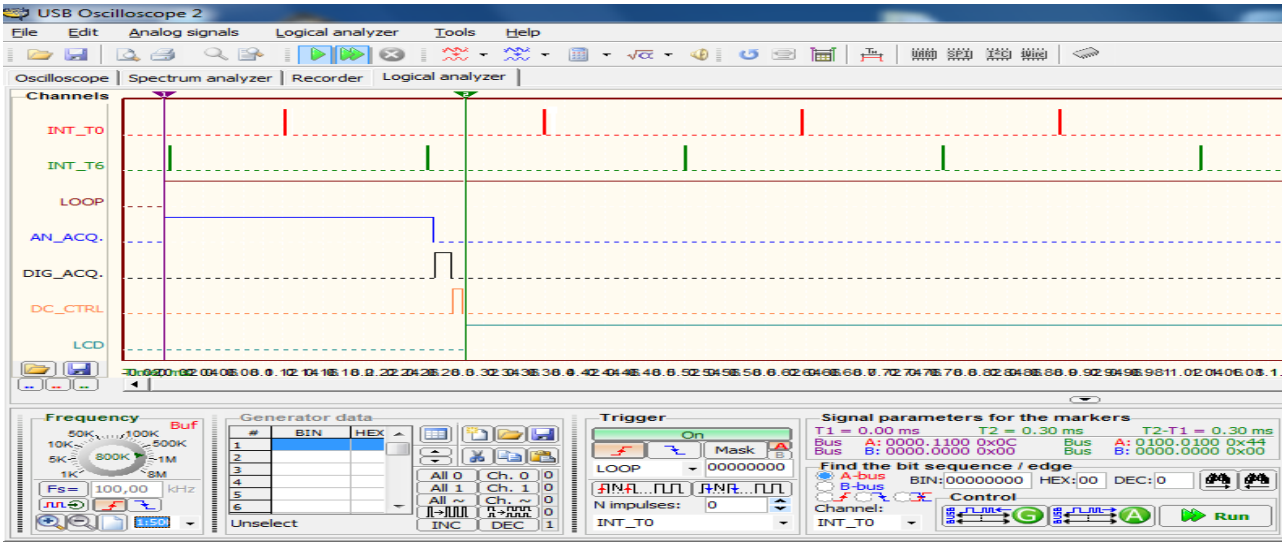

Fig. 7. The main tasks inside the control loop.

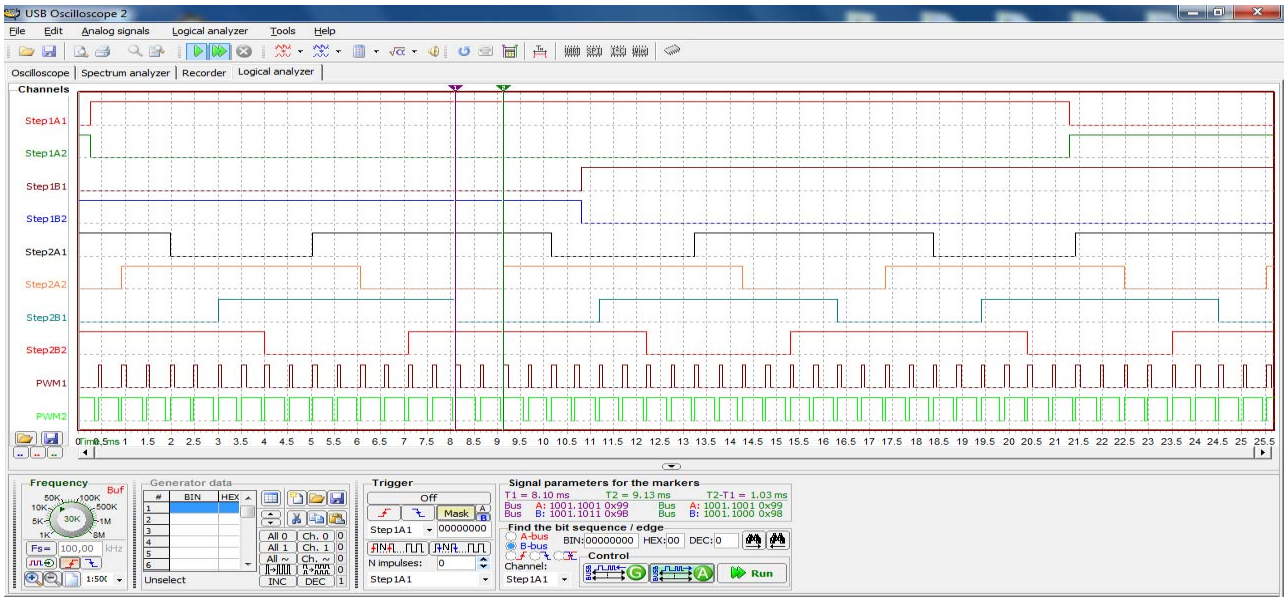

Fig. 8. The control signals for all the motors' windings.

The cursors are bordering in Fig. 8 a time interval of 1.03 ms for a step of the second step by step motor (hybrid sequence) and by that, the recorded diagram provides the precise real speed:

$$n_2 = \frac{60}{\Delta t_{step} \cdot 2 \cdot n_{steps./rev.}} = 145.6 \ \ RPM \qquad (2)$$

Fig. 9 and 10 refer to the PWM control of the two DC motors. The programmed PWM frequency is 1953.125 Hz. The real PWM frequency, recorded in real-time is:

$$f_{PWM} = \frac{1}{511.6 \cdot 10^{-6}} Hz = 1954.652 \ Hz \qquad (3)$$

with an error:

$$\varepsilon_{PWM} = \frac{1953.125 - 1954.652}{1953.125} \cdot 100 = 0.0781 \ \% \qquad (4)$$

For a good application design, it is important to correlate the real parameters of the motors (like the electromagnetic time constant, the mechanical inertia) with the range of adjusting parameters of the programmed signal, like the PWM frequency (for DC motors) and the minimum step duration for the steppers. In simulation, this correlation is not relevant (only the GUI limitations are involved) but with the tool and the real-time records like in this paper, a guaranteed operation range could be done.

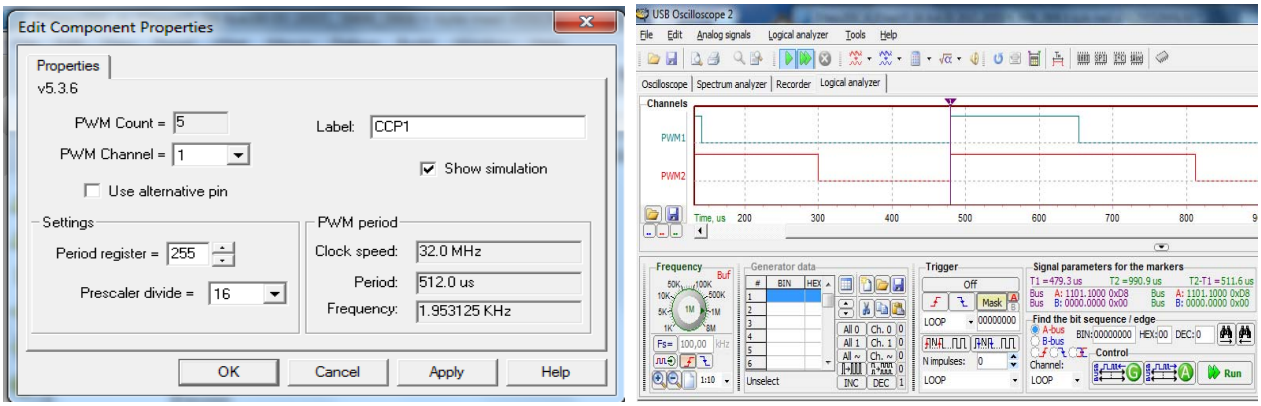

Fig. 9. The PWM channels: setting / programmed (left) and the real PWM signals (right).

IV. Conclusions

A modular hardware solution based on an 8 bit modern microcontroller is able to offer a very good solution for the multi-motor control, after a fast design cycle.

The timing and the motion parameters programmed for two different steppers (as construction and control sequence) and two DC motors are found with minimum errors in the real-time experiment. The performance / accuracy of the results is proved by some error values under 0.2 % .

An algorithm based on multiple interrupts was the key for a right control of the on-line timing. The way for the very good results starts with the software design, described in another paper.

A VHL programming language, component of a complex but easy to use IDE (Flowcode) made possible a tight timing control, from C accepted code lines for some clock mode improvement, to powerful customer macros and program code sections for real-time tracking of the timing, tasks and the duration of all events.

The on-line recorded signals were captured using an USB logic analyzer. The experimental platform combined modules manufactures by companies with additional interface modules realized in the laboratory.

Received on July 18, 2015
Editorial Approval on November 19, 2015

References

[1] R. de Castro, R.E. Araujo, H. Oliveira, "Control in multi-motor electric vehicle with a FPGA platform", IEEE International Symposium on Industrial Embedded Systems, 2009. SIES '09, pp.219-227.

[2] V. Chauhan, V.P. Patel, "Multi-motor synchronization techniques", *International Journal of Science*, Engineering and Technology Research (IJSETR), Volume 3, Issue 2, February 2014.

[3] M. Dmochowski, "Selected problems of multi-motor drive control", Master thesis, Politechnika Wroclawska, Poland, 2014, http://rab.ict.pwr.wroc.pl/~mw/Stud/Dypl/mdmochowski/ Marcin Dmochowski180503_MT.pdf.

[4] J. Hemming, "Designing multi motor controls with high performance and maintained modularity", Unjo, Sweeden, SPS/IPC/Drives, 2011.

[5] D., Mihai, "Modular H / S system as an experimental microcontrolled platform for the stepping motors", 7th International Conference on Electromechanical and Power Systems (SIELMEN), Iasi, Ed. PIM, 2009, pp. 391-394.

[6] D. Mihai, *Digital Electronics. Design Elements for Applications (in Romanian)*, Ed. Sitech, Craiova, 2013

[7] D. Mihai, "Designing a simultaneous control of several motors with a VHL programming language for a single microcontroller", *Annals of the Univ. of Craiova, Electrical Series*, Ed. Universitaria, No. 39, 2015, in press.

[8] E-Blocks PICmicro multiprogrammer version 9, Matrix Technology Solutions Ltd., 2014.

[9] L293D Quadruple half-H drivers, Texas Instruments Incorporated Dallas, Texas 2002.

[10] L298 Dual full-bridge driver, STMicroelectronics, 2000.

[11] L6460 SPI configurable stepper and DC multi motor driver, Doc ID 17713 Rev 1, 2010, STMicroelectronics group of companies, http://www.st.com/st-web-ui/static/active/en/resource/technical/document/datasheet/CD00278518.pdf.

[12] Multimotor application Guide Series: SSW-06 V1.6X, Doc. Nr: 10000603753 / 00, 2009, http://ecatalog.weg.net/files/wegnet/WEG-ssw-06-multimotor-10000603753-v1.6x-application-configuration-file-english.pdf.

[13] PIC 16(L)F1934/6/7, Data Sheet, DS41364E, Microchip Technology Inc., 2011.

[14] Power board datasheet EB011-00-1, Matrix Multimedia Limited, 2005.

[15] The Magellan family of motion control ICs, MC58113 Series motion control ICs, http://www.pmdcorp.com/downloads/MC58113-Datasheet.pdf.

[16] 42HS Series hybrid Stepping motors, Leadshine Technology Co. Ltd., http://ww1.microchip.com/downloads/en/DeviceDoc/Leedshine%2042HS03%20Stepper%20Motor%20Datasheet.pdf.