Two Stage Flow Regulation Valve Control Optimization by Software Techniques and Mathematics of Digital Systems Approach

Professor Dr.-Ing. Massimiliano Ruggeri

IMAMOTER, Institute for Agricultural and Earthmoving Machines of the National Research Council of Italy, Via Canal Bianco 28, Ferrara - 44124 Italy, E-mail: m.ruggeri@imamoter.cnr.it

Dr.-Ing. Matteo Fracassi

Research and Development Department, Tecnord S.r.l. Company, Via Malavolti 36, Modena – 41100, Italy, mfracassi@tecnord.com

Dr.-Ing. Massimo Martelli

IMAMOTER, Institute for Agricultural and Earthmoving Machines of the National Research Council of Italy, Via Canal Bianco 28, Ferrara - 44124 Italy, E-mail: m.ruggeri@imamoter.cnr.it

Dr.-Ing. Massimo Dian

IMAMOTER, Institute for Agricultural and Earthmoving Machines of the National Research Council of Italy, Via Canal Bianco 28, Ferrara - 44124 Italy, E-mail: m.ruggeri@imamoter.cnr.it

Abstract

The increased systems complexity and performance request for electro-hydraulic applications, ask for more performing electronic systems and control functions. The new more performing microcontrollers and efficient cross compilers, encourage the floating point mathematics usage in the software control routines, useful to directly reuse the routines generated by the simulation tools, despite the lack of control for precise resulting routine execution time. The paper describes the improvements in performance of a practical experience carried out on an electronic system optimization managing an electro-hydraulic two stage directional valve for vehicular applications. A deeper analysis done on the software side of the application, revealed that a custom firmware setup and local mathematical software impolementation optimizations, led to an optimal system configuration for performance. Here it is shown that, without lack of precision, fixed point mathematics, locally optimized, and a higher attention paid to tasks timing, results in a more performing software schedule executed by the embedded hardware, even if more instructions are executed due to the necessary rescaling of factors needed by the requested precision and if and more control tasks are activated.

KEYWORDS: floating point, fixed point, electronic valve control, embedded systems, real time

1. Introduction

The increased demand for performance of electro-hydraulic applications ask for more complex and prompt electronic control systems. Luckily the simulation tool offer co-simulation features that help designers in control systems synthesis with a high level of success. On the contrary, the model system based control strategies resulting from co-simulation of complex model are normally rich of equations expressed in floating point mathematics. As discussed in /1/ the code directly wrote by designers and simulated with the controlled system model in a suite, like for example *AmeSim* as in our case, or synthesized in a automatic way by other kind of tools, such as *Simulink* with the *TargetLink* tool, simply offer a control code that is totally general purpose and normally not designed for a particular embedded or real platform. In that way the obtained code does not focus any attention to hardware resource usage. In particular the first comparison over which we focused our study was on Floating point mathematic, that was found very difficult to be managed by an embedded microcontroller not featuring a Floating Point Unit (FPU). Anyway the relative simple direct implementation of an automatically generated code in embedded systems, enables less experienced people directly test on machine the designed control strategies, using standard firmware, device drivers and operating system set up provided by the general purpose templates normally offered alongside the evaluation boards. This paper demonstrates that the keys of success of a real application, that needs critical control functions, are also resident in an optimized usage of electronic hardware resources, due to strict real time control function execution constraint, impossible to be satisfied with a standard control code implementation.

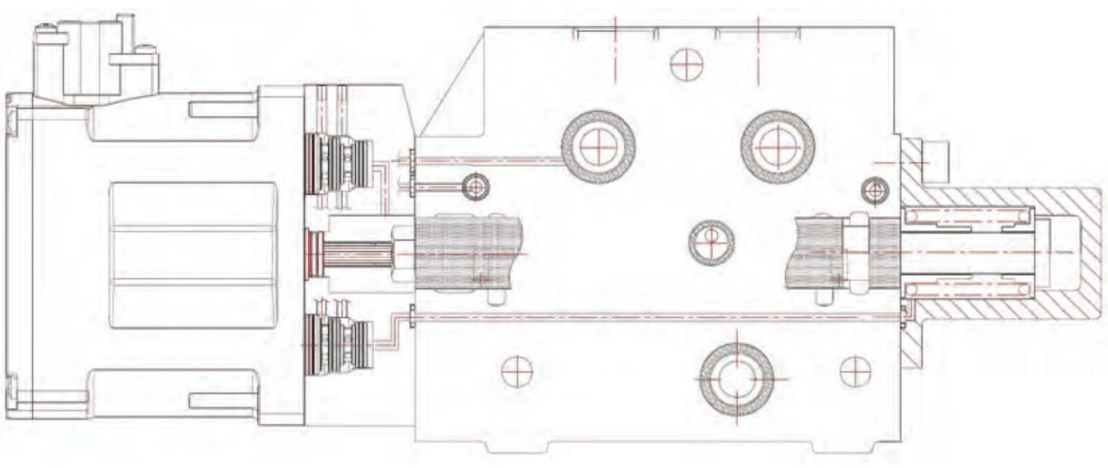

Figure 1: The Multidrom module coupled with a directional valve

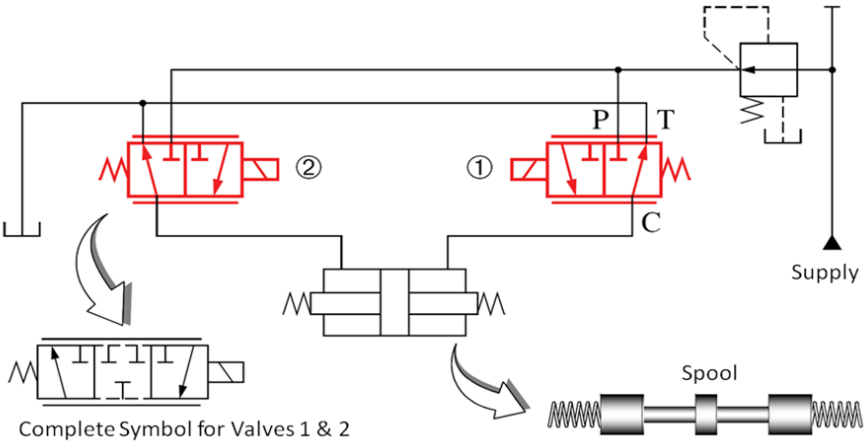

Figure 2: The Multidrom hydraulic scheme with the electro-hydraulic valves 1 and 2 and the second stage flow regulation spool

2. The system

The paper describes a real case study of a Two Stage Directional Valve Spool Position Control System; the system is called Multidrom®, produced by Tecnord Company in Italy. Basically it is a fist stage control of a valve spool adaptable at various products in the market and at various valves size. The flexibility of this complex mechatronic system is one of the strengths of Multidrom success in the market and a challenging characteristic because of the control stability and robustness for the entire range of applications. The directional valve is a CAN controlled directional valve and the spool position is controlled by a couple of three-way flow regulation valves, electronically and independently controlled on the basis of the spool position feedback from a contactless couple of position sensors (redundant). As shown in **figure 1** and **figure 2**, the two proportional flow regulation valves are controlled by two electronically controlled PWM signals, regulated by a Freescale HCS12 series microcontroller, a 16-bit core microcontroller widely used in automotive applications. The only feedback related to the position of the two proportional flow regulation valves is the current feedback acquired through the Analog to Digital Converter (ADC) of the microcontroller, while the current regulation is realized through an hardware circuitry, directly controlled by the PWM signal, by the specialized output of the microcontroller. The two proportional flow regulation valves are totally independent, and can be separately controlled, in order to increase the spool dynamic control performance. This first stage valve configuration gives the possibility to reduce the spool speed during transients, opening the opposite valve in respect to the desired direction of the spool movement, in order to avoid spool position overshoot, that is undesirable in actuators control.

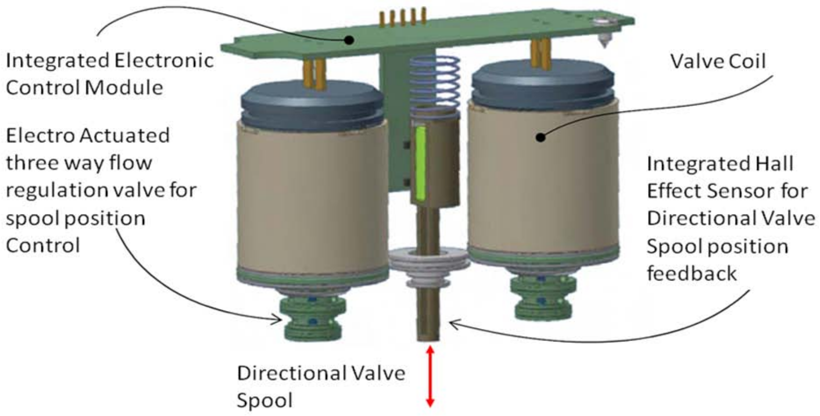

Figure 3: The Multidrom structure with electro-hydraulic valves and the spool position sensor

3. The control characteristics

The Multidrom module is provided with a complex control strategy, based on a feed forward function, implementing the model of the electro-hydraulic flow control valves and the second stage flow regulation valve. Then, a feedback function, based on a variable gains PID and a differentiated anti-windup strategy, is implemented, coupled with the feed forward function, in order to compensate the model uncertainties and the environment variables. The control scheme in shown in **figure 4**.

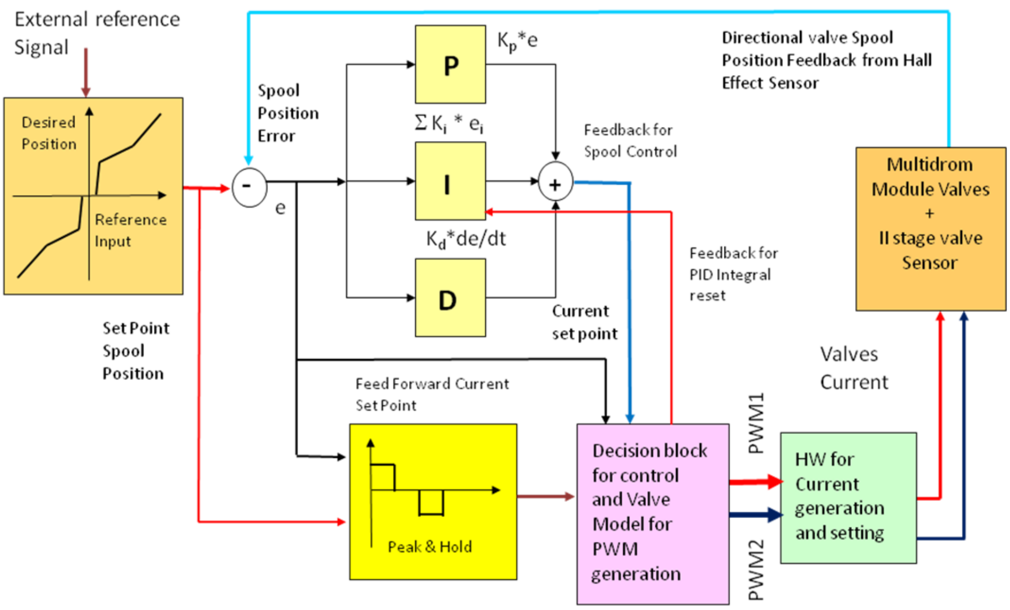

Figure 4: The Multidrom control function structure

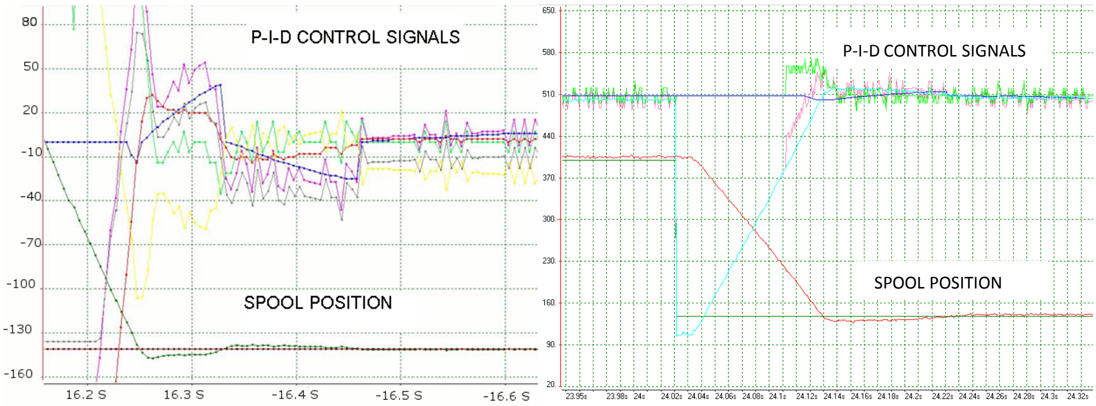

Figure 5: Step PID control response with floating point (left) and optimized (right) software

As discussed in /1/, the first version of Multidrom was released with a floating point mathematics (numbers represented as in (1)) based control; the resulting maximum rate for the control function execution and the necessary other functionalities for diagnosis and communication was 5 ms (cycle period). This function executed at 200 Hz frequency is apt to control a classic second stage dynamic of the flow regulation valve coupled with Multidrom modules, whose step response was measured to be 110 ms from central neutral position to the complete open position both in extend and retract direction, as shown in **figure 5** left diagram; although the control dynamics can result unable to take advantage of the higher dynamics of the electro-hydraulic controlled valves of which is provided the Multidrom module. Tests carried out in /1/ demonstrated the linear controls limits when used in highly nonlinear systems: steady state error (in Dark-Green in Figure 5 left), too long settling time, presence of overshoot. Using fixed point mathematics (as represented in (2)) the control function execution frequency was incremented until 1 kHz, then with a 5 factor, simply changing the control function implementation. That CPU time optimization proved that observing the dynamics of the controlled systems in a shorter time period, where the model linearization is allowed, the PID linear controller is apt to regulate the system behavior.

S	E (8 bits)	M(23 bits)

$$Value = (-1)^S * 1.M_2 * 2^{E-127} \quad (1)$$

I (12 bits)	F(20 bits)

$$Value = I.F \quad (2)$$

Figure 6: Floating Point (above) and Fixed Point (below) number representation

The resulting control function task execution time was then optimized considerably from the worst case of 820 µs, to 150 µs allowing a task control repetition frequency good for directional valve spool control in the most of applications and working conditions.

4. The control optimization

After the good results reached in /1/, a further analysis over the system performance and real time was carried out, detecting local real time problems and possible margins for performance improvement. In fact the control implementation was not adapted to the hardware system structure. Here follows the different control aspects description.

4.1. The embedded system optimization

The control function is just one of the task of the electronic controller. In this application the embedded system is not provided with a prehemptive operating system, conversely the different tasks are managed in a non prehemptive way, and the hard real time operations are managed using multilevel priority interrupts. As shown in **figure 7**, below scheme, due to the total number of function executed at occurrence of 20 and 50 ms, the total length of the scheduled tasks was more than 5 ms, thus limiting the maximum frequency for control function execution.

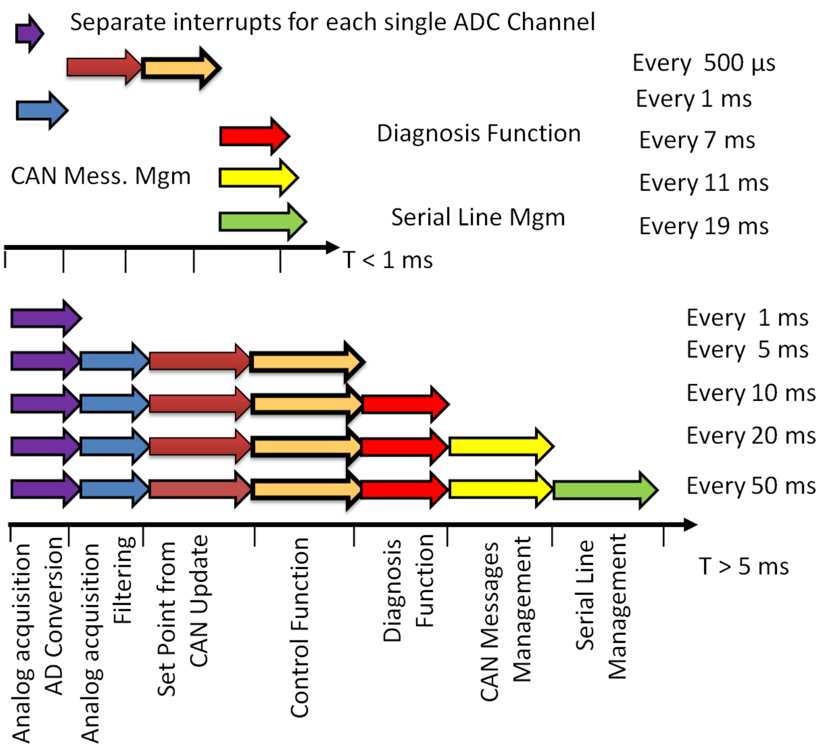

Figure 7: Task repetition and execution order timeouts new (high) and old (low) software

The total length of the tasks planned every 20 and 50 ms often exceeds the 5 ms time and, consequently, the control task execution frequency is locally affected by delays (**figure 8** left); that local delay had a bad impact on valve position control, due a longer time between two control function execution and, consequently, between two control action update on the control valves of the Multidrom module. Figure 7 shows the comparison in terms of precise task timing between the original software version and the optimized in terms of task scheduling, in which the task timing is managed with 5 independent timers with scheduling at prime numbers, using the first prime number smaller than the previous used timing value or less. The 5 groups in that way are almost executed in less than 1 ms where the other tasks are not executed in 5 ms. In the figure 7 above graph it can be noted that the control task period is 500 μs and in figure 8 right it can be noted that the task is ever executed exactly as scheduled, with no delays. Only in case of least common multiples in timer values, there are common execution but is very rare that more than a couple of group of function is executed in the same time. After this improvement the fixed point software timing was modified increasing of a 2 factor the control function execution, the can messages management and the diagnosis function. But a very performing optimization in terms of CPU time, was actuated in the field of fixed point mathematics: despite the standard, that reserves a fixed data dimension and fixed number of bits for the integer and fractional parts of data, thus allowing only a multiplicative factor for all data on a control function, fixed point data and calculation were optimized in a different way for each equation, thus founding that in many cases a simple 16 bit based calculation was enough precise for the system under test. The Time required for Control function is now 87 μs.

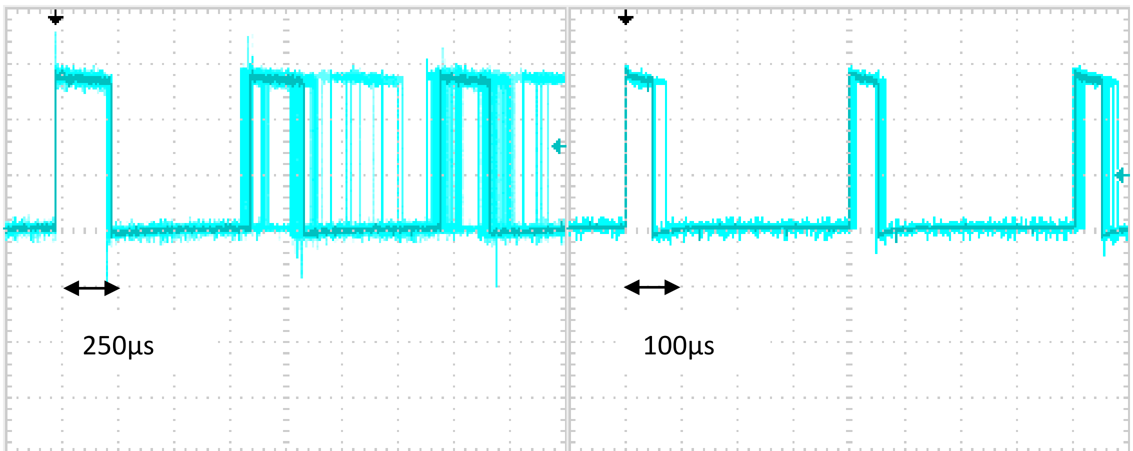

Figure 8: Control function task timing with original (left) and modified (right) scheduling

In fact the 16 bit mathematics are faster for the Freescale microcontroller because of the natural data dimension of the core, and for the ALU consequently, is 16 bit and the most of 16-bit based operation are executed in a instruction cycle. The last improvement is represented by the optimization of the Analog to Digital Converter (ADC) scheduling and peripheral management. The HCS12 ADC presents only the “start on request” functionality and is not provided by the “free running” functionality that could allow the totally independent continuous AD conversion without lack of time of the microcontroller core. As commonly known, the AD conversion is normally affected by electrical noise and a single channel acquisition is not enough precise to be used in the control function; then more AD conversion are requested by a weighted arithmetic mean or filtering. All these limitation were present also in the fixed point software revision, and affected the tasks repetition frequency because of the long time of ADC operations required. In the first version of software the total time needed for the complete sensors AD conversion was around 300 μs. In fact, in order to obtain a good AD conversion, the conversion time was settled at the longest value, and a single conversion was operated for each ADC function call, treating all sensors in sequence. In order to make ADC functionality independent from the other tasks, a time based interrupt was created, and a large number of high speed conversions (16 conversions automatically executed by the ADC peripheral) were settled and stored in a FIFO circular buffer queue, for each channel in the ADC peripheral. Next, a filter function was created, synchronous with the control function, regardless of the relative timing for sensor value acquisition, because of the high frequency of the control function execution. That oversampling, even if less precise because the lower ADC channel settling time, and the weighted mean of values calculated every 500 μs, obtained very good results in respect to the filter used in the previous software.

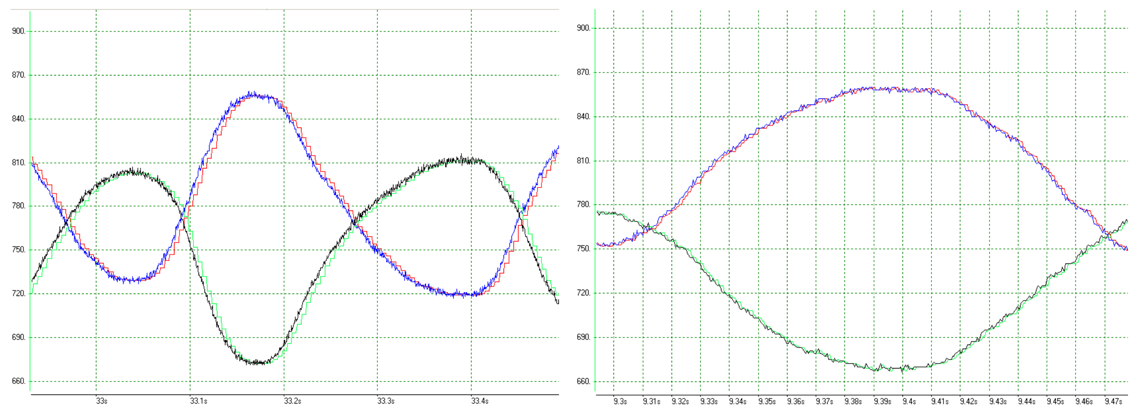

Figure 9: Spool position sensors acquisition in old (left) and new (right) ADC management

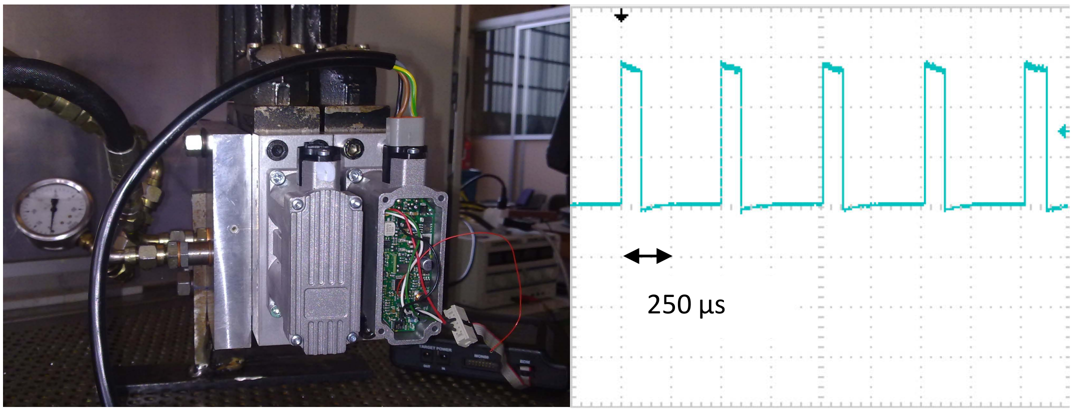

Figure 10: Multidrom at test bench and the 500µs control task frequency acquisition

As shown in **figure 9** the new AD conversion management is quite better because of a lower delay in position sensor update and a lower step in low dynamic conditions with an acceptable noise level that does not affect the control stability, while it allows a higher control response readiness.

5. The tests

Test were performed at test bench (**figure 10** left) using a valve and a Multidrom electronic hardware equipped with probes, allowing the real time electronic signals acquisition by an oscilloscope (figure 10 right graph), in order to evaluate the precise task control timing. All other relevant data, both from sensors and from control corrections, were acquired through the Multidrom CAN network with Vector CanOe, at 500 µs frequency, running with the optimized software. The same CAN network analyzer CanOe, was used to simulate CAN commands to the module through dynamic scripts, providing set point time histories for flow regulation commands automatically sent by the tool over the CAN network and acquired by the Multidrom.

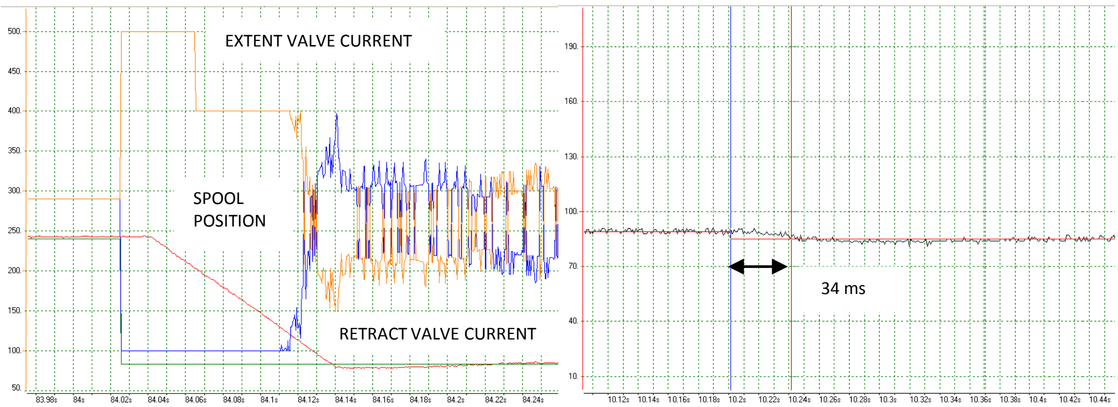

Figure 11: Big step and small step regulation with 500 µs control task frequency acquisition

The Dynamic and promptness of the system was increased with the *Peak & Hold* strategy, allowing the full step application for big set point transients as shown in the **figure 11** left graph**,** where the orange line represents the current at the valve that controls the spool in the extent side, while the blue is for retract side. As shown in the Figure, the dynamics of the valves switch, used to reach the right spool position, are high and, due to the closed center characteristic of the electro-hydraulic valves, the effective commands are represented by big PWM steps. That dynamic control is critical from steady state attainment: small steps in spool regulation must be achieved with care at the control action because of the risk to overcome the set point. For that reason the previous software versions were slow in small set point corrections: to avoid local instabilities and for the regulation difficulties, due to a less frequent control for valve spool position correction. Conversely, in the right diagram of figure 11 it can be noted that the new position with a 0,1 mm step (1 % of the spool displacement) is reached in less than 35 ms for a 450 l/min flow regulation valve in the metering area, around the 20% of time requested for the same transient in the previous software version. From the control stability point of view, the two diagrams in **figure 12**, show two of the most difficult to reached set point track for spool regulation position. In the left graph on the figure, it can be noted that a sinusoidal set point (in red color) is perfectly copied by the spool position acquired through the position sensors and sent by CAN at 500 μs of frequency, except in the step stimulus, where the spool dynamic can't be neglected. In the right diagram steps, of varying sizes are followed without lack of precision and the saw-tooth envelope is also followed perfectly. Similar graphs were obtained for different stimulus frequencies and different time histories. The improvement in steady state error nullification had a very important consequence in the valve control quality because of the total absence of hysteresis in valve control.

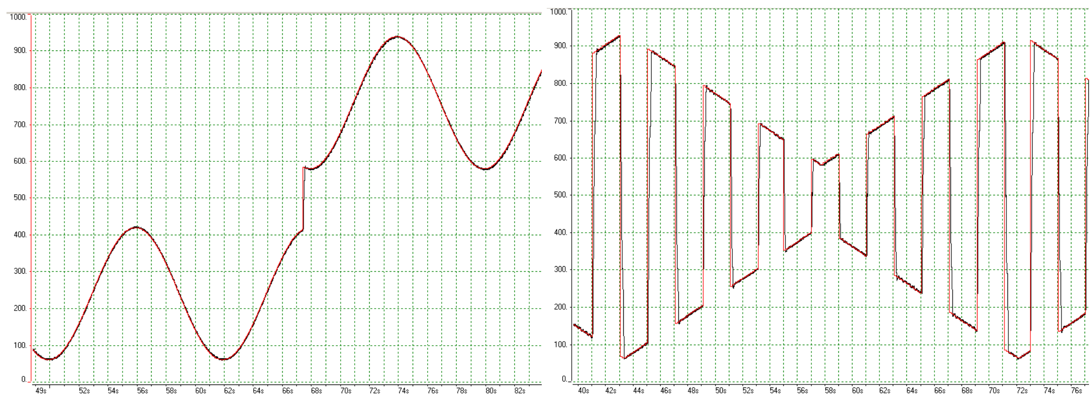

Figure 12: Dynamic set point track of the new Multidrom control function

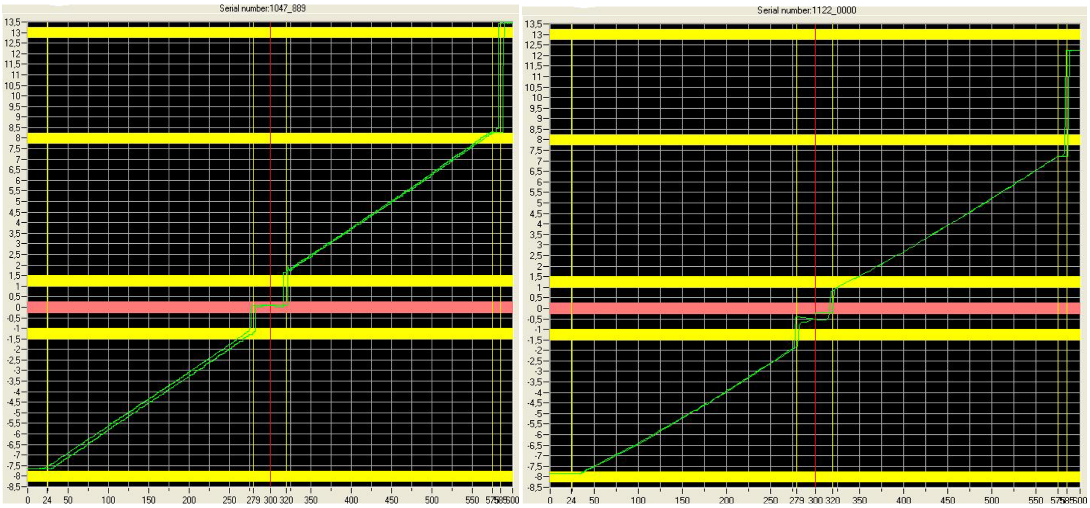

Figure 13: Dynamic set point track of the new Multidrom control function

In fact, the **figure 13** shows the difference between the valve spool position control with the old control (left graph) and with the new control strategy (right graph), obtained in the Tecnord End of Line bench, at Multidrom production line; it can be noted that the two green lines in the right graph are overlapped, while in the left graph there is an hysteresis especially in extent position because of the asymmetry of the mechanical part of the spool control system.

6. Conclusion

The state of the art for the product is now represented by this solution that, after a complete test validation is in production in all product versions. The outstanding control obtained simply modifying the substratum on which the control function is built, indicate that often the success of a real application is both in control function design and in electronic system design. The real time can be reached using a more performing hardware, like a HWIL platform (like DSpace) and any software, without worrying about the software performance, but just for tests; conversely, the required real time can be achieved using a real embedded hardware ready for production, with a special attention to the system and software performance. The case study demonstrates that a special expertise is necessary in order to obtain the requested the optimal performance in a cost effective electronic system for valve control.

7. Bibliographical references

/1/ Ruggeri M., Fracassi M., Martelli M., Dian M., Fixed Point versus Floating Point mathematics in embedded system programming for fluid power mechatronic components control: a real case study, 8th Japan Fluid Power Symposium, JFPS, Okinawa Japan, October 2012.

/2/ IEEE 754, ed. 2008: IEEE Standard for Binary Floating-Point Arithmetic, IEEE

/3/ IEC 60559 ed. 2.0: 1989, Binary floating-point arithmetic for microprocessor systems, IEC, Swiss.

/4/ James W. Gray, PID Routines for MC68HC11K4 and MC68HC11N4 Microcontrollers, AN1215/D Freescale Semiconductor Inc., 2004.

/5/ Elmenreich W., Rosenblatt M., Wolf A., Fixed Point Library Based on ISO/IEC Standard DTR 18037 for Atmel AVR Microcontrollers, 5th Workshop on Intelligent Solutions in Embedded Systems, June 2007,Madrid, Spain, pages 101 – 113.

/6/ Kum K.; Kang J.; Sung W; AUTOSCALER for C: an optimizing floating-point to integer C program converter for fixed-point digital signal processors, IEEE Transactions on Circuits and Systems II, Vol. 47/9, 2000, Page(s): 840 – 848.

/7/ Goossens G., Van Praet J., Lanneer D., Geurts W., et al., Embedded software in real-time signal processing systems: design technologies, Proceedings of the IEEE, Vol. 85, Issue 3, 1997, pages 436 -454.

8. Symbols

I	Electrical current	A
ms	Time, milliseconds	ms
µs	Time, microseconds	µs
ns	Time, nanoseconds	ns
P	Oil Pressure	bar
Q	Oil Flow Rate	l/min

SP	Valve Spool Position	mm
V	Electrical Voltage	V