Mansoura Engineering Journal (MEJ) Vol. 16, No. 2. Dec. 1991.

MICROPROCESSOR BASED PID CONTROLLER DESIGN AND IMPLEMENTATION

Kamel M. Soliman , Sabry F. Saraya , and Wael R. El-Wasif

Aut. Control & Computers Dept., Faculty of Eng., El-Mansoura Univ.

عنوان البحث : تصميم و تنفيذ متحكم تناسبي تكاملي تفاضلي باستخدام الميكروبروسيسور.

ملخص البحث : يقدم البحث طريقة مقترحة لتصميم و تنفيذ متحكم تناسبي تكاملي تفاضلي باستخدام الميكروبروسيسور. و قد روعي في الطريقة المقترحة التغلب على كل المشاكل المرتبطة بالمكونات المادية و البرمجة و مستوى الاشارات و التوصيل بين مكونات النظام المختلفة ... الخ. و تتميز الطريقة المقترحة بالجمع بين مزايا المتحكم المقترح من حيث سهولة التصميم و البساطة و ملائمته لعمليات التحكم المختلفة و كذلك مزايا استخدام الميكروبروسيسور من حيث انخفاض التكلفة و سهولة الضبط، وقد تم تطبيق هذه الطريقة للتحكم في وضع و سرعة محرك تيار مستمر تحت ظروف تشغيل و تحميل مختلفة. و قد بينت النتائج فعالية استخدام الميكروبروسيسور في قوانين التحكم في العمليات الصناعية التطبيقية.

ABSTRACT

This paper presents the design and implementation of PID controller based on the use of microprocessors. The implemented technique takes into account the problems associated with hardware as well as software, e.g. interfacing, resolution, signal leveling, microprocessor limitation, .. etc. The technique combines the advantages of the conventional PID controller (e.g. simplicity, ease of design, and suitability for different types of processes) and those of microprocessors and microcomputers (e.g. low cost, easy tuning and adjustment, and enhanced controller reliability).

The designed controller is applied to control the position and speed of DC motor under different load and reference input conditions. Satisfactory results regarding process performance are obtained. Therefore, the use of microprocessors and microcomputers is recommended for control of physical processes.

1. INTRODUCTION

Digital control techniques are currently being adopted for implementation in a rapidly increasing number of industrial plants. Digital controller design techniques based on Z-transform method (e. g. Dead Beat algorithm, Dahlin's algorithm .. etc.) require the existence of a precise plant model in the Z domain [3]. However, in many situations, this model is attainable only through the use of complex parameter identification techniques, which makes the design procedure a rather complex task.

For inherently nonlinear processes, linearization techniques are used to extract a plant model for controller design purposes. However, this model is valid only for small range of plant parameters variations around their nominal values. This difficulty can be overcome through the use of adaptive control techniques. These techniques [2,4,5] can be implemented successfully only for slow processes to allow for the large computational time inherent in these methods. Furthermore, these computations often require expensive powerful digital computers with large memories to be able to deal with the large volume of data required for system adaptation. These drawbacks have made the implementation of Z-transform based controllers and adaptive controllers a rather difficult and costly operation.

The conventional PID controller continues to represent a strong candidate in controller design domain. This is due mainly to its simplicity, ease of design and implementation, and suitability for different types of plants [1]. The hard wired PID controller is adapted to the operating conditions existing in a certain process either manually or through the use of self tuning PID controllers. This tuning allows the controller to handle variations in process parameters as well as changing surrounding environmental factors. Tuning hard wired PID controllers involves changing the setting of some physical device (e. g. changing potentiometer setting, adjusting valve position .. etc.). These adjustments are sometimes limited in resolution which makes it difficult to obtain "optimum" PID settings. Self tuned PID controllers require the use of large computers to implement the tuning (adaptation) algorithm [4]. These limitations can be avoided through using relatively cheap microprocessor to implement PID controllers.

In this paper, PID control action is implemented using microprocessors. This technique reduces the task of tuning PID gains to changing the contents of one or more memory locations. In addition, the resolution of gain adjustment is limited only by A/D and D/A converters accuracy, microprocessor memory word length, and complexity of software implementing the technique. All these parameters can be easily controlled to produce the required tuning resolution. The implemented technique is applied to control the speed and position of DC motor, which is a relatively fast process. Emphasis is placed on practical implementation related aspects such as interfacing and software development. The technique could be easily extended to other types of processes by minor modifications in configuration.

2. THE CONTROL ALGORITHM

2.1. General Representation

Figure (1) shows the general block diagram of digital Closed Loop Control System (C.L.C.S). The following notations apply to signals depicted in figure:

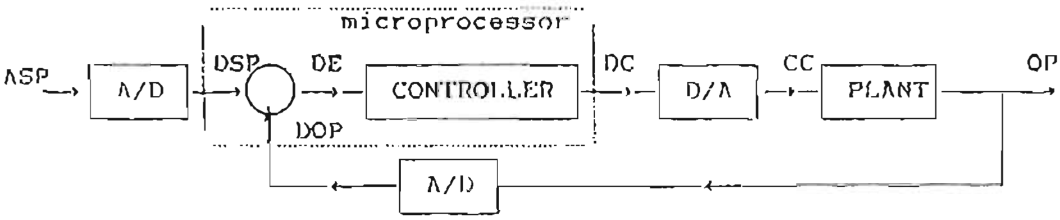

Figure (1)

- ASP, DSP : Analog and Digital Set Point;
- DOP : Discretized (digital) OutPut signal:
- DE = DSP-DOP : Digital control system Error; (1)
- DC, CC : Digital and Continuous controller output values;
- OP : analog system OutPut.

In this C.L.C.S, the microprocessor is used to implement the comparator as well as the control algorithm as explained later. The controller processes digital error (DE) to produce controller output (DC) through the relation:

$$DC = F(DE) \tag{2}$$

Where the function F is usually expressed in difference equations form to facilitate implementation of the control algorithm using microprocessor / microcomputer.

2.2. The PID Controller

In continuous time PID control systems, the controller output (control variable) m(t) is related to the control system error e(t) by the relation [1]

$$m(t) = K_p\, e(t) + K_d \frac{d\, e(t)}{dt} + K_i \int_0^t e(\tau)\, d\tau \tag{3}$$

Where K_p, K_d, and K_i are proportional, derivative, and integral mode gains, respectively. To implement this relation using microprocessor, each term must be first expressed in difference equation form as explained in the following subsections.

1 - Proportional mode control : In which controller output is determined, at any sampling time, by the error at that time. Referring to figure (1), proportional control may be realized by the relation:

$$DCP = KP * DE + DCO \tag{4}$$

Where DCP : Digital proportional controller output;
DCO : Zero error controller output.

If the output controlled variable DOP is equal to the set point DSP so that the error DE is zero, then the controller output will be DCO. Thus DCO operates to "keep the process at its current state when no error exists". Although this term (DCO) is essential in pure proportional mode control, it can be neglected when combining this mode with other control modes (as controller output is not likely to be zero).

ii-Integral mode control : In which, controller output depends not only on the error signal currently present at controller input, but also on previous values of this error. In continuous time control systems, this dependency may be expressed by:

$$c_i(t) = k_i(t)\ A_e(t) + c_i(0) \qquad (5)$$

Where $c_i(t)$: Controller output at any time t;

k_i : Integral mode gain;

$A_e(t)$: Area under error curve.

To transform this equation into a form that can be implemented using microprocessors, the area under error curve must be approximated using any approximation technique. The simplest technique is the zero order (rectangular) approximation, since it requires only the present error value and thus no memory is used to store previous errors. In this technique, the area under error curve at the k'th sampling instant is given by:

$$AREA(k+1) = AREA(k) + DE(k) * DT \qquad (6)$$

where AREA(k) : Area used to produce controller output at sample k;
- DT : Sampling period;
- DE(k) : Digital error at sample k.

To simplify coding of integral mode control using microprocessors, a new variable SUM is defined as :

$$SUM\ (k) = SUM(k-1) + DE(k) \qquad (7)$$

So, discrete integral control equation may be written as

$$DCI(K) = KII * SUM(K) * DT = KI * SUM(K) \qquad (8)$$

Where the gain KI incorporates the integral gain KII along with the sampling time DT. Thus only one multiplication operation is performed which reduces sampling time and hence increases system efficiency.

iii- Derivative control mode : In which controller output is dependent not on error values, but on the manner by which these values change with time. In continuous time control systems, this control action may be expressed by:

$$c_d(t) = k_d\ de(t) / dt \qquad (9)$$

To implement this controller using the inherently digital microprocessor, differentiation must be converted into difference equation form. Using backward difference approximation, the equation for digital derivative control becomes

$$DCD(K) = KDD * \frac{DE(K) - DE(K-1)}{DT} = KD * DDE(K) \qquad (10)$$

Where DCD(k) : Digital derivative controller output at sample K;
KDD : Derivative gain;
DE(K) : Digital error at sample K;
DDE(K) = DE(K) - DE(K-1);
KD = KDD/DT.

It is notable that combining the sampling time DT and the derivative gain KDD into one parameter KD simplifies the control implementation (by eliminating division) and thus reduces computation time.

iv- Composite control mode : Combining the previously presented controllers together yield different composite control modes (e.g. Proportional Integral (PI), Proportional Derivative (PD), and Proportional Integral Derivative (PID) controls). The PID controller is the most versatile one as it provides ability to shape system performance more accurately in most situations. So digital PID controller takes the form:

$$DC(K) = KP * DE(K) + KI * SUM(K) + KD * DDE(K) \quad (11)$$

3. DESIGN CONSIDERATIONS

When implementing any digital control algorithm using microprocessors, several factors should be considered. These factors concern both hardware and software. The former arises mainly when attempting to interface the digital microprocessor to the inherently analog outside environment, while the latter depends, among other things, on control algorithm complexity, process speed, and capabilities of available microprocessors.

3.1. Hardware Considerations

In dealing with the hardware aspects of implementing digital controllers using microprocessors, only problems related to interfacing between the controller microprocessor and the controlled process will be discussed. Characteristics of the microprocessor itself will be dealt with when considering software related aspects. That is because the microprocessor affects the process only through software. Furthermore, any limitations found in the microprocessor can be overcome via suitable software modules.

i- D/A Resolution : The resolution of a D/A converter is defined as the smallest change in the analog output as a result of a change in the digital input [6]. It can be shown that the resolution as a percentage of the full scale output analog signal can be expressed as:

$$\% \text{ resolution} = \frac{1}{2^N - 1} \quad (12)$$

Where N = No. of bits in D/A input word.

It is thus obvious that using D/A converters with large word length enhances system accuracy and thus reduces oscillations and steady state error. However, such D/A converters are usually expensive. Furthermore, problems may arise when trying to connect such D/A to microprocessor having different word length. These problems may be solved using special hardware (multiplexers) and modifications in software.

ii- Control signals level: The allowable range of variation of A/D input (and D/A output) is usually small compared with signal levels found in the analog section of the C.L.C.S shown in Fig. (1). One solution for this problem is to use sensors whose outputs match A/D input range. However, this solution is impractical and sometimes impossible. Therefore, special circuits should be built to match the digital section with the analog section of the C.L.C.S. One such circuit using operational amplifiers is proposed as shown in figure (2). In this circuit, the use of two op amps ensure that no polarity conversion is made. Furthermore, the introduction of reference voltage V_R makes it possible to shift the entire operating rang up or down as well as scaling it by means of the resistive network shown. The output voltage of this circuit can be expressed as:

$$V_o = \left[\frac{V_x}{R_1} + \frac{V_R}{R_2} \right] * R_3 * \frac{R_5}{R_4} \qquad (13)$$

Where

V_x Input voltage to scaling circuit;

V_R Reference DC voltage;

V_o Scaling circuit output voltage.

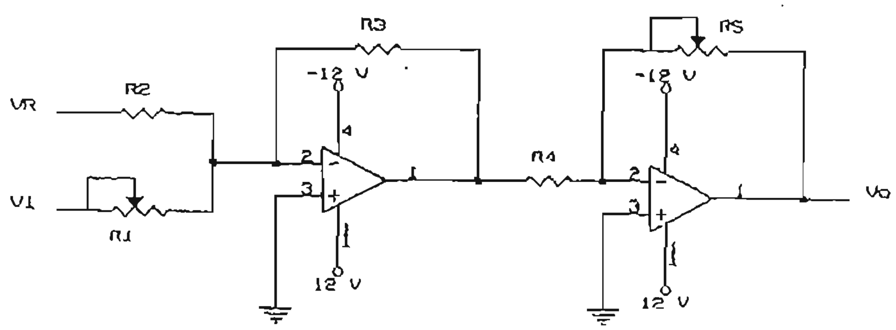

Fig. (2) Op-Amp based matching circuit

As for the digital controller output (DC), it usually needs amplification before being applied to the analog process. Also the signal itself may need to be converted into another form (hydraulic, pneumatic,.. etc.); thus requiring transducers of suitable type.

3.2. Software Considerations

Due to the unavoidable limitations in microprocessor computing capabilities, several problems arise which mainly concern how to overcome these limitations so that the coded program interprets faithfully the developed control algorithm. These problems relate mainly to microprocessor construction and how it carries out arithmetic operations.

i- **Memory word length :** The binary representation of numerical values in microprocessor memory is limited both in magnitude and in accuracy by the used microprocessor memory word length. An eight-bit word can represent integer numbers in the range from 0 to 255 (or -128 to 127) according to number representation scheme. In addition, if a memory word is allocated to the fractional part of a certain variable, it can represent fractions with step of 1/255 = 0.0039, which may be unacceptable value for high precision control. This problem can be solved by allocating more than one memory location for each variable (integer and fractional parts). However, such scheme complicates software modules and thus increases computational time.

ii- Overflow and underflow : Overflow occurs when the result of a certain mathematical operation exceeds the maximum number that can be stored in microprocessor memory word. One solution is to use more than one memory word to represent constants, intermediate results, and final results. This scheme calls for writing special software modules to perform mathematical operations involving two or more words per variable.

For practical computational time considerations, memory allocation for each variable cannot be increased to more than two or three words. So, overflow may still occur. To overcome this problem, a scheme is implemented in which whenever overflow condition is detected, the maximum value that can be stored in the allocated memory is put in place of the actual result. This scheme makes a compromise between allocating large memory space for controller variables (software complexity) and using one memory location for each variable (representation error).

Underflow occurs when the result of arithmetic subtraction operation is less than the minimum value of the digital word. This minimum value depends on number representation scheme adopted in the program. The occurrence of underflow is dealt with in a manner similar to that used with overflow (i.e. replacing result by the minimum value that can be stored).

iii- Microprocessor instruction set : The existence of certain "high level" instructions (e.g. MUL, DIV, ..etc.) in the used microprocessor instruction set simplifies the controller program significantly. However, not all microprocessors have such capabilities. Therefore, special modules are usually written to perform multiplication (and division) operations. These modules should handle variables stored in more than one memory location. Furthermore, they should handle fractional part as well as integer part in most variables (as many microprocessors do not support floating point arithmetic directly).

iv- Error polarity : The error DE applied to the controller as defined by eqn. (1) can be positive or negative. The error will be a digital number in 2's complement representation. It is very inconvenient to use this representation of the error in subsequent computations required to produce the output. Therefore, once DE has been calculated it will be converted to an unsigned number and the polarity will be remembered by a flag word. This technique is outlined in the flow chart of the integral controller shown in figure (3).

4. EXPERIMENTAL WORK

The developed control system is implemented using the following hardware modules:

i- Microprocessor module [7]

The system uses the 6502 microprocessor. This microprocessor unit has the following specifications :

Memory word length : 8 bits.
Clock speed : 1 M Hz.
Interfacing : Memory mapped interfacing using 65C02 VIA.

ii- Analog to digital (A/D) converter [8]

The used analog to digital converter has the following technical specifications:

* Based on the MPU 7581 circuit that includes the converter system and 8-channel analog multiplexer.
* Input voltage range : 0 V to +10 V.
* Conversion time : 80 μsec.
* Input circuit protection : up to 15 V.
* Binary output range : 0000 0000 to 1111 1111.

iii- Digital to analog (D/A) converter [9]

The system uses digital to analog converter having the following specifications:

* Two 8-bit channels, TTL compatible.
* DAC-08 based D/A, 8-bit quick converter with a maximum alinearity of ± 0.39 % fs.
* Output voltage range : +10 V to -10 V.
* Output through an active filter to remove switching peaks when the digital value of the input byte is altered. Fitting the circuit pass-band to the control program response.

The controller is applied to control the position and speed of DC motor under different load and reference input conditions. Different combinations of values for KP, KI ,and KD are used to obtain 'optimum' set-point tracking for position and speed of the DC motor. Figure (4) depicts the general schematic diagram of the position control case. (All other schematic diagrams, programs and flow charts are under call from authors. They are omitted here for space considerations).

Figures (5) to (10) show chosen samples from the large amount of recorded results. From all these results, the following comments can be summarized:

1- Higher values of proportional gain lead to smaller rise time, but, on the other side, increase the percentage overshoot. Also these high values conceal the effect of derivative gain (if present).

2- It is not recommended to use integral control only or to use high values of integral gain, since it causes oscillations and may cause instability.

3- The steady state error can be decreased by either increasing the proportional gain, or adding properly chosen integral action (not high) to the relatively small proportional gain.

4- PID 'optimum' tuning is not so easy, as it needs a large amount of trials. Carrying out these trials is made easy through the use of microprocessors.

5. CONCLUSION

The implemented, microprocessor based, PID controller has several advantages, such as:

1- Simplicity and non costly design and construction of the control system.
2- Easy tuning of the PID parameters by changing values stored in microprocessor memory.
3- The system can be easily extended to accommodate parameter adaptation to varying plant dynamics.

By using this technique, a good controller is designed and built to control both the speed and position of DC motor under different load and reference input conditions. Satisfactory results are obtained which agree with the required control task. The implemented technique could be easily extended to other types of physical processes by minor modifications in configuration.

REFERENCES

1- Curtis D. Johnson, "Microprocessor Based Process Control," Prentice-Hall Inc., 1984.
2- Graham C. Goodwin, Kwai Sang Sin, "Adaptive Filtering - Prediction and Control," Prentice-Hall Inc., 1984.

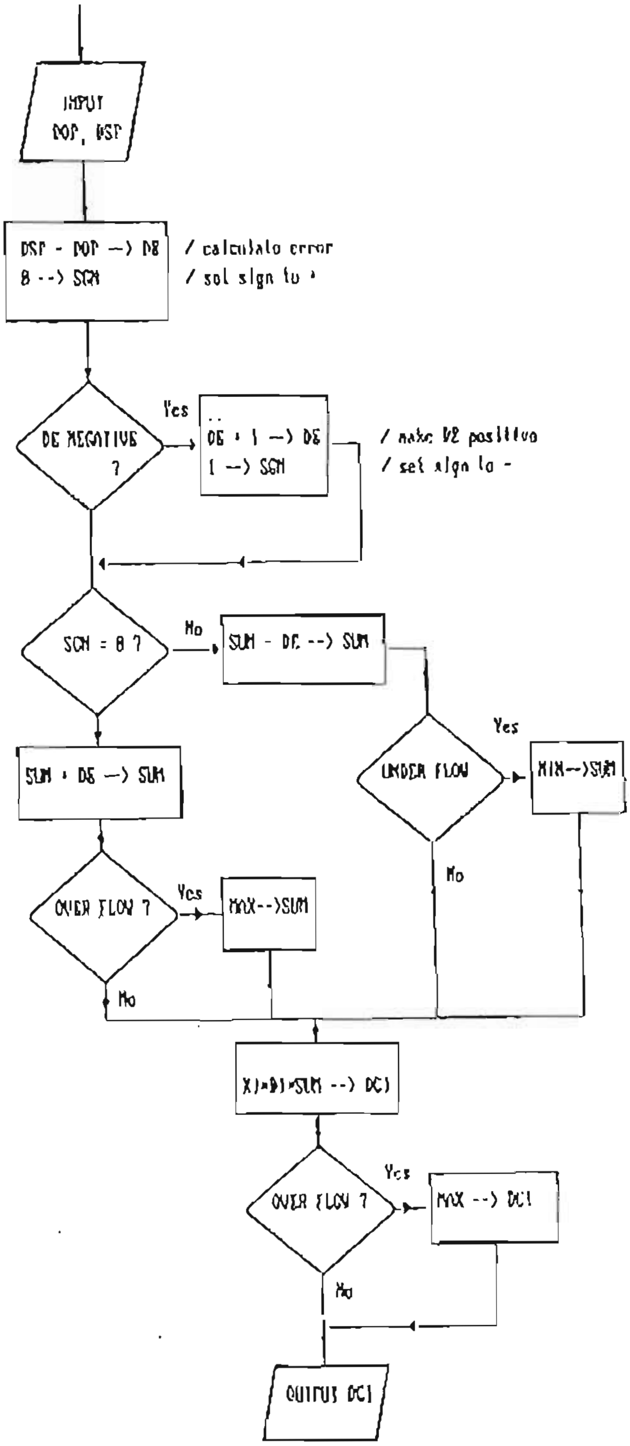

Fig. (3) Integral control flow chart

3- Karl J. Åström, Björn Wittenmark, "Computer Controlled Systems, Theory and Design," Prentice – Hall, N.J., 1984.
4- Karl J. Åström, Björn Wittenmark, "Adaptive Control." Addison – Wesley Publishing Company, 1989.
5- I. D. Landau, "Adaptive Control – Model Reference Approach," Marcell Dekker Company, 1979.
6- Ronald J. Tocci, "Digital Systems – Principles and Applications," Prentice-Hall International Inc., 1985.
7- PROMAX Company, "MI 650C Microinstructor Manual".
8- PROMAX Company, "Module A/D Converter MM-603".
9- PROMAX Company, "Module D/A Converter MM-602".

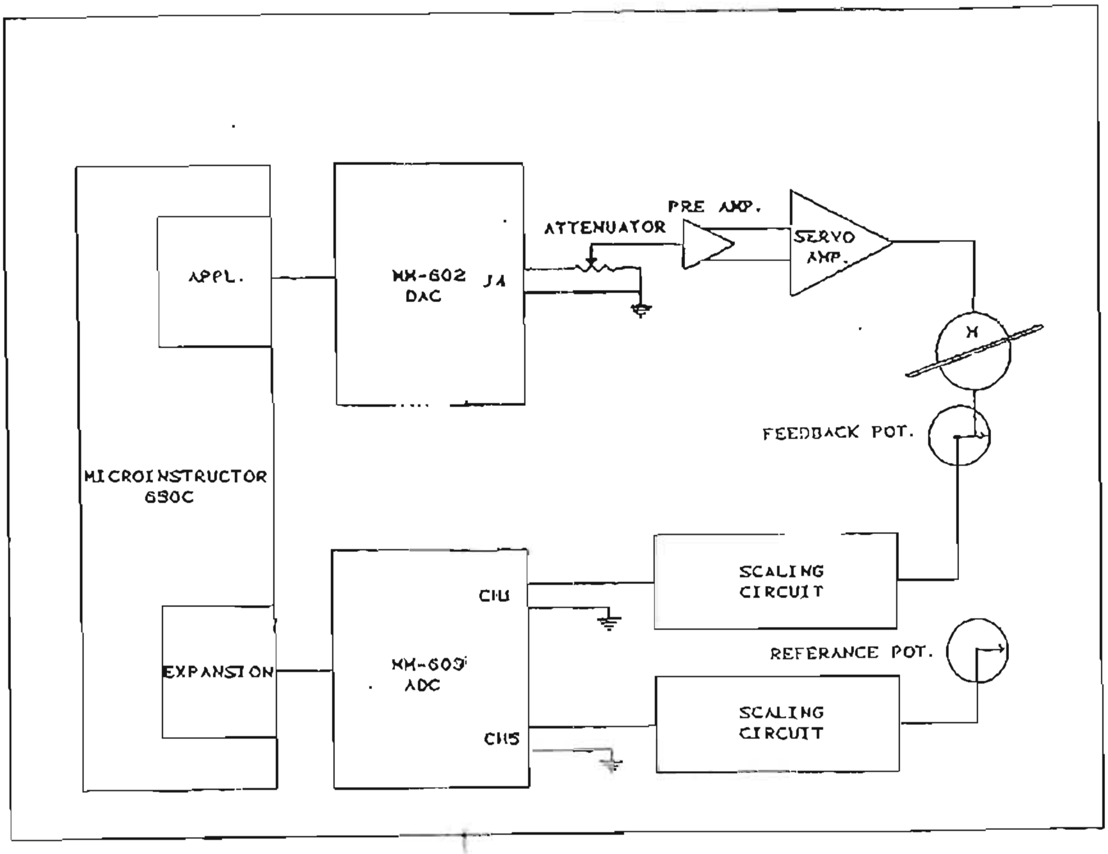

Fig. (4) DC-motor position Control

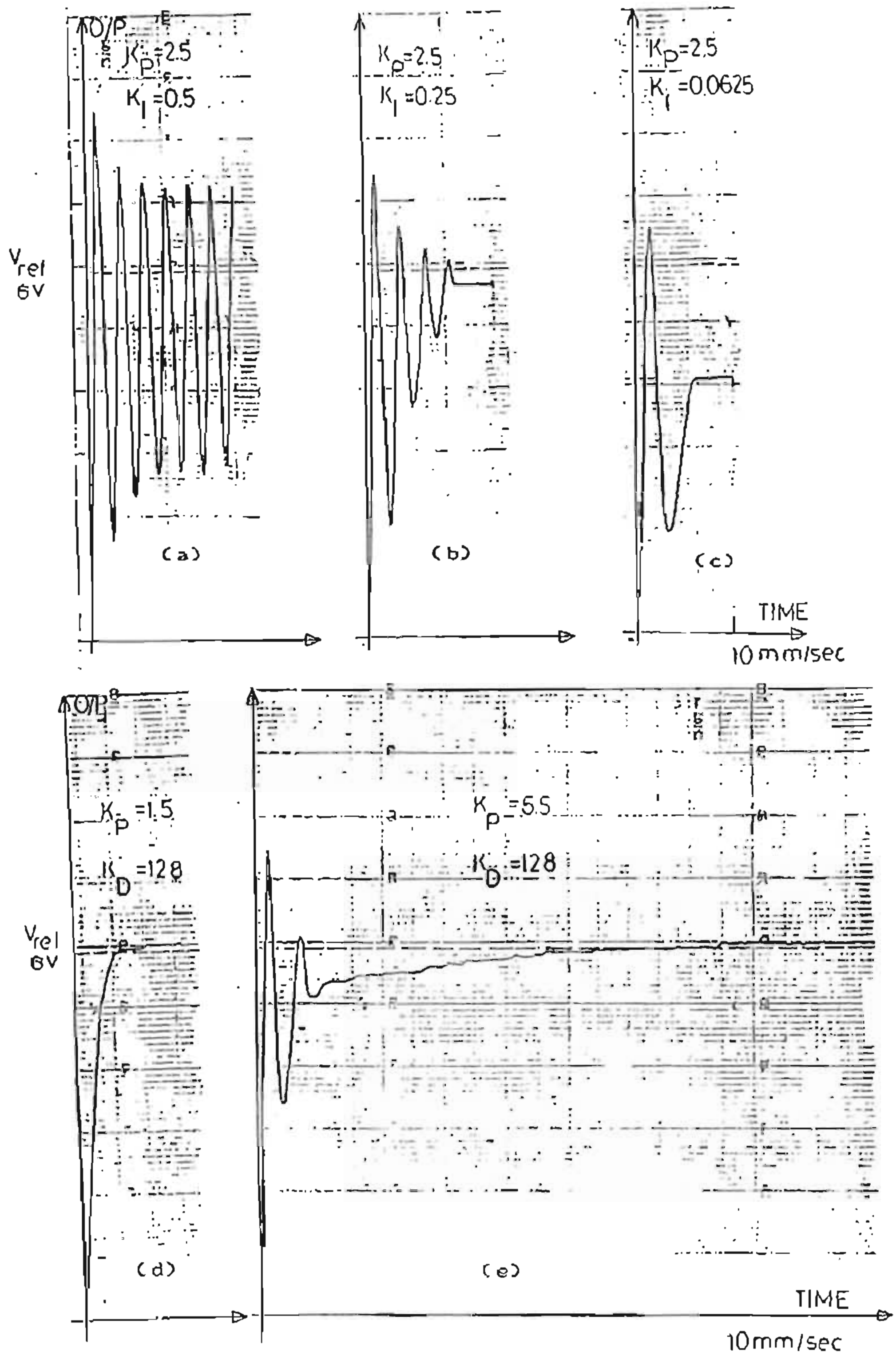

FIG. (5) - PI and PD position controller

(different control parameters)

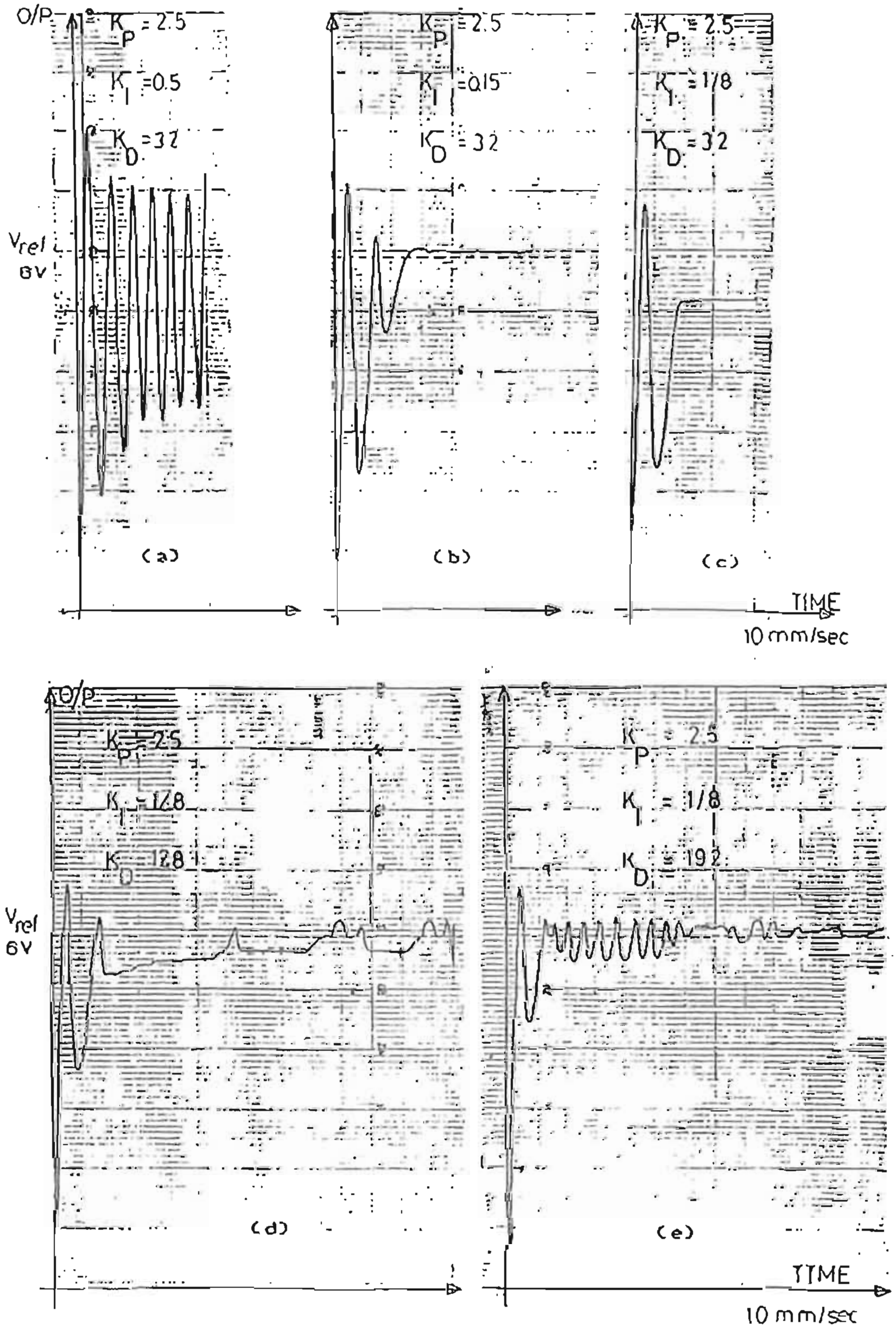

Fig (6) - PID - position controller

(different control parameters)

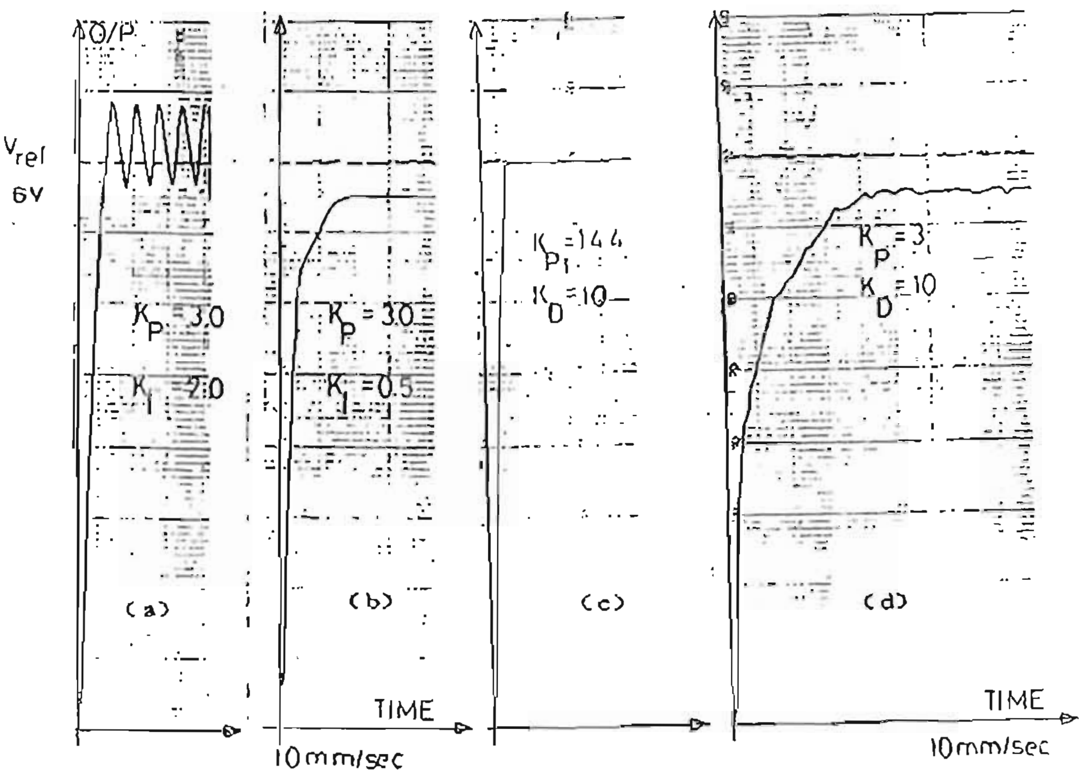

Fig (7) - PI and PD speed control

(different control parameters)

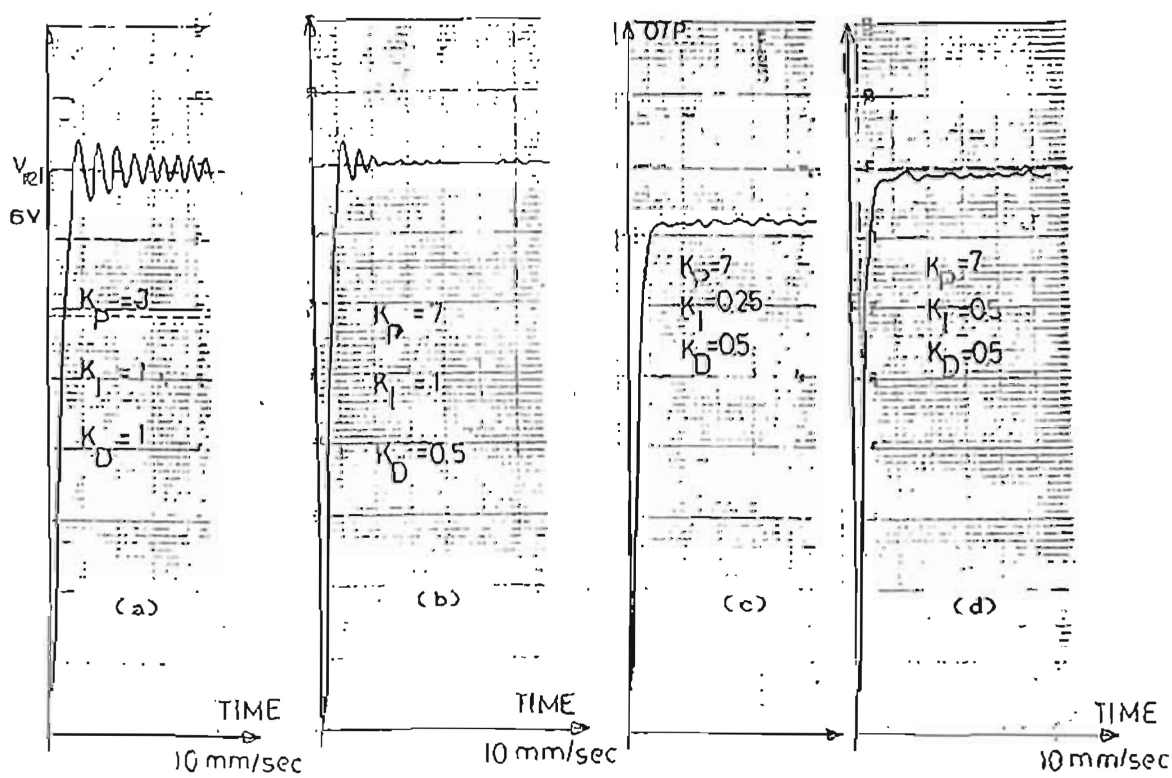

Fig (8) - PID speed control

(Different control parameters)

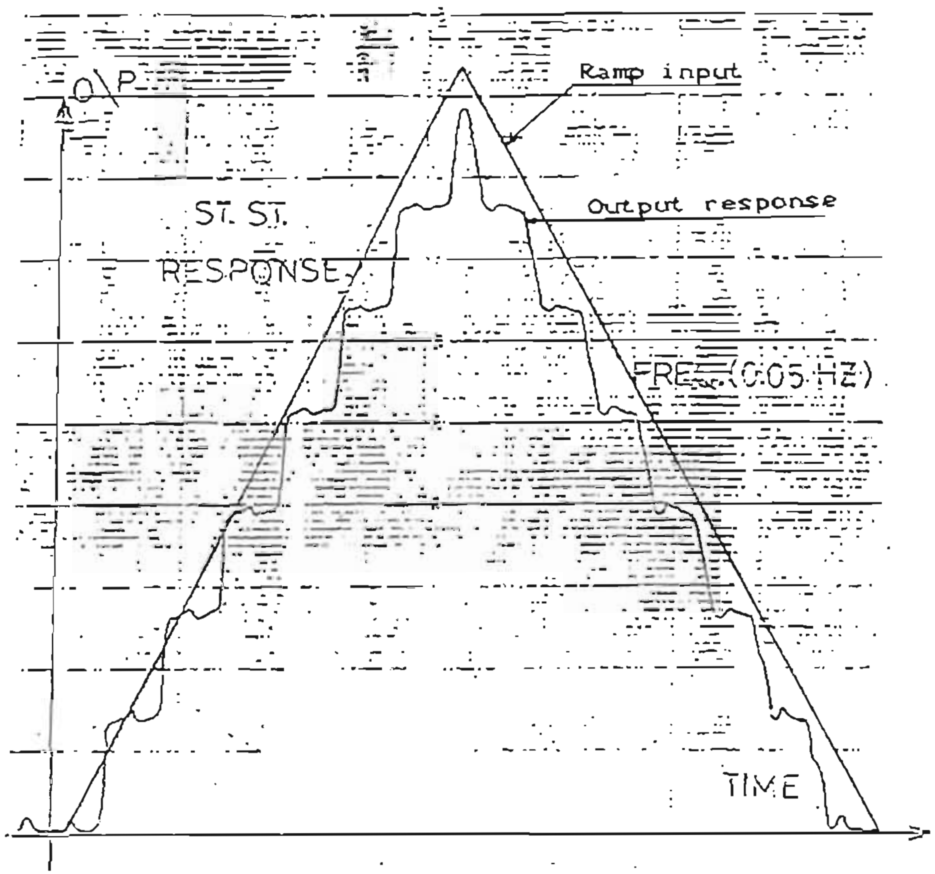

Fig. (10) PID speed control with ramp input

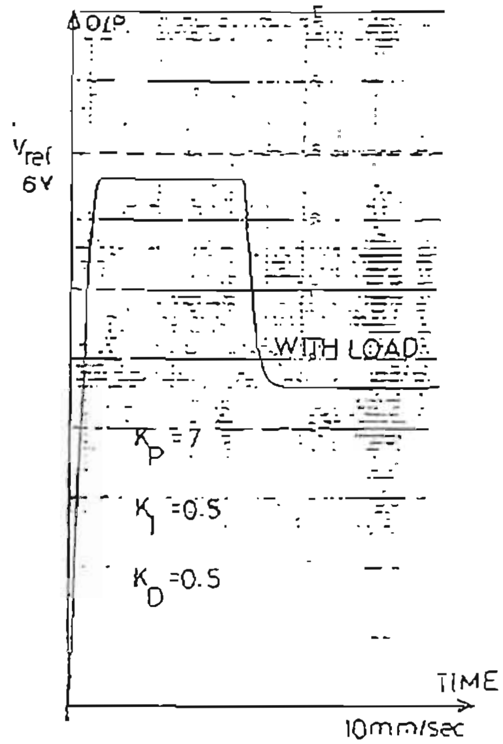

Fig. (9) Effect of sudden load on PID speed controlled motor