Volume 5, Issue 12, December – 2020 International Journal of Innovative Science and Research Technology

ISSN No:-2456-2165

Microcomputer Application in Motion Control

Muhtar H. Alhassan
Department of Computer Science
National Open University of Nigeria
Abuja

Abstract:- This paper briefly examines the microcomputer both as a control element and as a feedback-signal processor in modern closed-loop motion-control systems. Problems of interfacing, effective data-acquisition and real-time control are briefly highlighted.

I. INTRODUCTION

The gradual replacement of analogue controllers for speed, position, torque, etc, with digital ones has facilitated the evolution of the digital computer into one of the major components of modern motion control systems. Generally, the microcomputer when properly interfaced is suitable for implementing several types of functions in a power electronic based motion control system. These include:

- Gate-firing control of power converters,
- PWM and step-wave signal generation,
- Feedback control loops
- Signal Processing
- Sequencing control
- Monitoring and warning
- Diagnostics

This paper focuses on the roles of the microcomputer both as a control element and as a feedback-signal synthesizer in a closed-loop electric drive system.

II. INTERFACING

Interfacing is the means by which the microcomputer bus can communicate with any peripheral device. It is the means by which a computer is connected to external equipment for data acquisition, monitoring or control. Signals coming from the external device must be transformed into a form suitable for access by the computer. Similarly, any signals generated by the computer for transmission to the external device must be appropriately transformed before transmission to the equipment. The set-up is as shown in Fig. 1.

Thus, interfacing usually involves one or more of the following:

- Electrical buffering
- Data transmission control
- Code conversion
- Interrupt handling
- Data conversion
- Timing
- Multiplexing

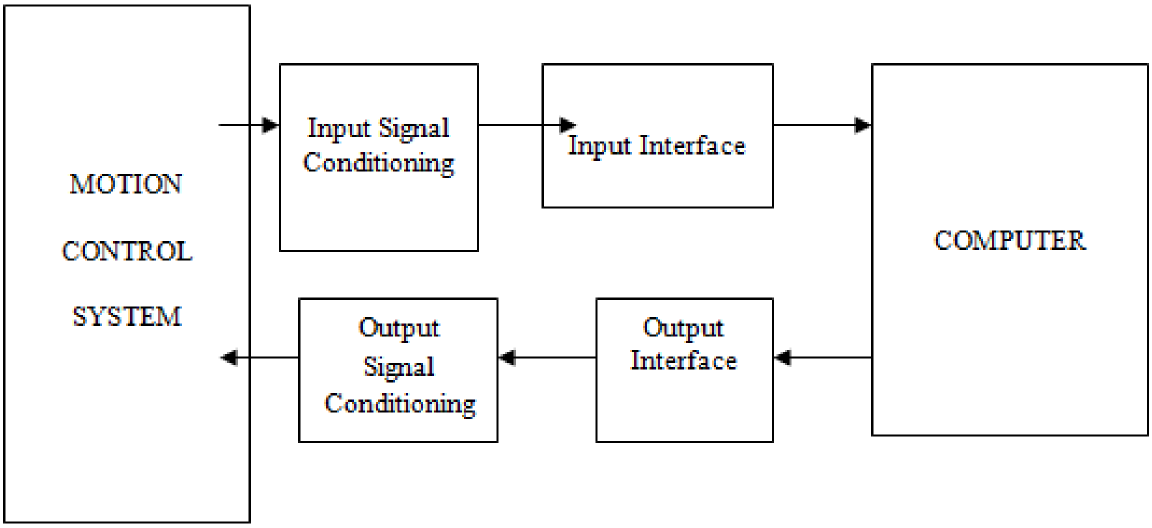

Fig 1:- Interfacing the computer to a control system

The IBM PCs and compatibles can only function as effective tools in a data-acquisition / control context if they are **extended** by the addition of appropriate **expansion boards.** Such boards are designed to plug directly into any of the computer's expansion slots. A typical board, like the PC30A, includes 16 analogue input lines, 2 analogue output lines, 24 programmable I/O lines and usually a counter timer.

Some non-IBM compatible machines, like the BBC microcomputer, come equipped with on-board A/D conversion capabilities. The 12-BIT A/D converter on the BBC, which is usually the integrating type, can accept analogue inputs in the range between 0 and 1.8 volts. Here, the upper range (1.8 volts) will produce the number 65520, which represents a 12-bit increment of 16. Thus in order to obtain the normal 12-bit range of 0 to 4095, a division by 16 is necessary. Although there may be just one A/D converter on-board a machine like the BBC, several analogue inputs can be accommodated by multiplexing. Software access to the A/D converter is provided by the BBC BASIC keyword ADVAL (n) where n refers to the channel number (1, 2, 3 or 4 in this case).

III. SIGNAL CONDITIONING

Signals generated from sensors in motion control systems are sometimes analogue in nature and generally must be appropriately treated before they become suitable for sampling by a computer. This is referred to as signal conditioning and can take any of the following forms:

- **Voltage Amplification:** This helps to match the full-scale voltage change of a sensor output to the full-scale voltage range of the input interface. Thus, a sensor might produce a voltage from 0V to 1mV full-scale whereas the input interface may linearly convert voltages in the range of 0 to 1.023 V to a digital number between 0 and 1023. In such a case, it becomes necessary to amplify the input signal by a factor of 1023 to utilise the full digital range.Units
- **Level Shifting**: This refers to the alignment of the minimum sensor output voltage with the minimum voltage that can be converted by the input interface. For example, if a sensor produces a voltage between –0.5 and 0.5 V, and the input interface converts a voltage between 0 to 1 V to a digital number, a voltage of 0.5 V must be added to the sensor output voltage to match the interface requirement.
- **Frequency-Range Limiting:** This is effectively the filtering out of unwanted high-frequency components from the input signal. Filtering can also be used to block a dc voltage component using a high-pass filter, or by *ac coupling* using a simple capacitor in series with the input signal.
- **Electrical Isolation:** It is often necessary to electrically isolate signal generators from the computer to reduce the effect of power-line voltage difference and noise that are usually generated when there is a common electric circuit connecting two physically separate systems.

IV. DATA ACQUISITION

There are three main ways that the PC acquires data from the outside world, namely.

- Memory Mapped I/O
- Dedicated Port Addressed I/O
- Direct Memory Access

In **memory mapped I/O**, the I/O lines are grouped in sets of 8, and each set is called a **port** and treated as a single memory location. The ports are accessed using memory transfer instructions such as PEEK and POKE in BASIC, or LDA and STA in assembly language. Since memory addresses are used up for this type of I/O, communication is relatively slow.

In **dedicated port addressed I/O**, ports and channels are assigned unique addresses on a *dedicated I/O bus*. They are therefore accessed using additional software instructions like IN (INP) and OUT in low- or high-level programming languages.

Direct memory access (DMA) is usually adopted when high sampling rates are required. Since non-DMA data-acquisition techniques use program control for data transfer to memory, they have frequency limitations of about 100 kHz on the sampling rate [2]. DMA, on the other hand, is hardware based, and with the help of tri-state buffers, causes the microprocessor to momentarily be isolated from the system buses so that the *DMA device* can *directly* access *memory*, thereby increasing the data transfer rate up to a possible 1 MHz.

V. COMPUTER CONTROL

Microcomputer /digital control of electric drive systems has several advantages over analogue types, notable among which are:

1. Significant reduction in controller-hardware cost, especially in systems using powerful VLSI microcomputers with high speed and greater functional integration.
2. Improved reliability leading to higher MTBF (mean time-between-failures), due to the replacement of multiple electronic components by LSI and VLSI chips.
3. Drift-free operation with minimal parameter variation.
4. Possibility of implementing sophisticated control functions involving complex computations as for example in variable speed ac drives.
5. Easy implementation of such functions as temperature monitoring, over current warning, etc.

There are two notable limitations to the use of a microcomputer in the control of an electric drive system:

1. The introduction of quantisation error due to D/A and A/D conversions. Increasing both the bit-size and the sampling rate can usually reduce this type of error.
2. A relatively sluggish response due to the serial way signals are processed in the microcomputer. Sluggishness is most noticeable in centralised control of multi-drive systems involving multitasking operations that tend to slow down conventional microprocessors. This situation has improved tremendously with the development of faster RISC type processors that can be used to produce microcontrollers with excellent dynamic responses.

VI. TYPES OF COMPUTER CONTROL

The digital computer can play a variety of roles in motion control systems based on the strategy adopted. Thus in **Computer Supervisory Control,** the controllers are essentially analogue with the computer accessing the drive variables and generating driving signals for establishing the *desired set point.*

In **Direct Digital Control**, the computer directly controls all the loops of the system. Thus the computer directly sets the driving power of each loop-actuator, loop by loop. A serious setback in this type of system is that whenever the computer 'goes down' or 'crashes', all the loops become open thereby making the system 'runaway'. An effective but rather expensive remedy is to have a stand-by computer acting as a monitor ready to take-over control when things go wrong with the main machine.

Microprocessor control is a distributed type computer control strategy, in which each loop of the system has its own microprocessor or single-chip microcomputer. This method guarantees stability, self-tuning, remote re-programmability and ease of implementation of complex algorithms for compensating dead time delays and enhances the application of feedforward control.

In **Integrated Computer Control,** several microcomputer-based single-loop controllers are individually connected to a central computer referred to as a *cell supervisor*. The cell supervisor provides displays, keeps records and issues commands to the individual loop controllers. In some cases, the supervising computer itself reports to a higher-level management computer system. Compatibility of physical, electrical and data structures in such a system must be ensured by the adoption of standards such as the Manufacturing Automation Protocol **(**MAP)[3].

VII. IMPLEMENTING DIGITAL COMPENSATORS

The block diagram of Fig.2 represents a microcomputer-based position control system. The discontinuous nature of the power converter that usually feeds the motor is often neglected for the purpose of analysis. The system is characterized by Multirate sampling, i.e. the various loops have different sampling times. Thus, the speed loop, which usually has a large bandwidth, is characterized by a smaller sampling time than that of the outer position loop.

In the position loop, the microcomputer samples the command position θ_r^*, feedback position θ_r and updates the speed command I_{sq} * through the PI position controller all at a sampling rate of $1/T_{s1}$. In the speed loop, the microcomputer samples the speed command ω_r^*, the feedback speed ω_r, and generates the q-axis current command I_{sq}^* at a sampling rate of $1/T_{s2}$

VIII. THE PID CONTROLLER

The output equation for a PID controller is given by

$$v_O = K_p e + K_i \int edt + K_d \frac{de}{dt} \qquad \textbf{(1)}$$

where e is the error, $v_o$is the output and K_p, K_i and K_d are the proportional, integral and derivative coefficients of the controller. The precise value of constants K_p, K_i and K_d determine the quality of control of the system. Implementing the PID controller in a microcomputer requires the conversion of Equation 1 into a corresponding difference equation. First, both sides of the equation are differentiated with respect to time to obtain

$$\frac{dv_o}{dt} = K_p \frac{de}{dt} + K_i \frac{d}{dt}(\int edt) + K_d \frac{d^2e}{dt^2} \qquad \textbf{(2)}$$

Since computer implementation is discrete in nature, the equation must now be transformed into the discrete mode byreplacing continuous time with discrete time as follows:

$$\frac{\Delta v_o}{T_s} = K_p \frac{\Delta e}{T_s} + K_i e + K_d \frac{\Delta}{T_s}\left(\frac{\Delta e}{T_s}\right) \qquad \textbf{(3)}$$

Here, T_s is the cycle time, $\Delta v_o$is the change in output and Δe is error change. Thus

$$\Delta v_O = v_O(k) - v_O(k-1) \qquad \textbf{(4)}$$

$$\Delta e = e(k) - e(k-1) \qquad \textbf{(5)}$$

The difference equation for the PID controller implies that we compute the present value of the output based on the previous output, the present error, previous errors, the sampling time and the controller constants.

IX. SOFTWARE CONSIDERATIONS

The PID equation can easily be implemented using a high-level language like C, Pascal, FORTRAN PL/M, and so on. Alternatively, an assembly language can be used for this purpose. High-level implementation has the following advantages:

- Possibility of using floating point arithmetic
- Automatic handling of negative numbers
- Possibility of utilizing a math co-processor

Assembly language programming would require the separation of the PID equation into several smaller steps that are definable in the assembly language of the host processor.

Computer implementation of PID controllers, whether software- or hardware-based, can lead to unstable response for some specific combinations of the proportional gain K_p and the sampling time T_s especially in second-order systems. Another setback is the fact, that derivative computations in a sampled-data system can produce unpredictable output jerks due to the introduction of DAC related delays. This phenomenon makes the use of simple derivative controllers usually undesirable. More sophisticated derivative computations are usually resorted to, if stability is to be achieved. Alternatively, derivative action can be entirely dropped so that a PI controller results.

X. THE PI CONTROLLER

The transfer function of a PI controller has the form

$$\frac{C(s)}{R(s)} = K_p + \frac{K_i}{s} \qquad \textbf{(6)}$$

In finite difference form, the transfer function becomes

$$\frac{C(k+1)-C(k)}{T_s} = K_i R(k) + K_p\left[\frac{R(k+1)-R(k)}{T_s}\right] \qquad \textbf{(7)}$$

The smaller is T_s , the sampling time, the more accurate is the representation of equation 6 by equation 7. Further manipulation of equation 8 yields

$$C(k+1) = C(k) + K_p R(k+1) + (K_i T_s - K_p) R(k) \qquad \textbf{(8)}$$

For computer implementation, the difference equation is transformed into state variable form as follows:

$$X(k+1) = AX(k) + BR(k) \qquad \textbf{(9)}$$

$$C(k) = MX(k) + NR(k) \qquad \textbf{(10)}$$

Here X(k) is the state variable, A = 1, B = K_iT_s, M = 1 and N = K_p. These equations can be easily implemented digitally. The gain coefficients are adjustable software parameters that can be tuned experimentally using, say, the Ziegler-Nicol's method.

XI. FEEDBACK SIGNAL SYNTHESIS

Most modern strategies applied to the control of servo drives rely on complete state feedback. Unfortunately, however, it is often impossible or very costly to monitor some of the state variables. In such cases when some state feedback signals are not directly measurable, they have to be computed from any that have been measured. This is an important area of application for the microcomputer in feedback control. For example, in a PWM-inverter based AC drive system, it is possible to synthesise such feedback signals as flux, torque, and the like, from measured voltages and currents.

Another example of feedback signal synthesis is the digital measurement of rotor speed from incremental position signals generated by a simple incremental encoder. The pulse train from the encoder is accumulated in a counter, and the speed in rpm computed as

$$N = 60 \times \frac{m}{MT_s} \qquad \textbf{(11)}$$

where Ts is the sampling time in seconds, m is the number of encoder pulses in one sampling interval, and M is the number of encoder pulses per revolution. The larger the sampling period, the more accurate the speed measurement. Furthermore, the pulses per revolution should be as high as possible for a given sampling period. The sampling period is limited by the bandwidth of the speed-control loop, whereas the pulses per revolution are speed-sensor dependent. Thus in the final analysis, a compromise has to be made between the accuracy with which the speed is estimated, and the bandwidth of the speed loop. It should be noted, that at lower speeds, due to the fall in frequency of the encoder pulse train, the method of pulse-interval measurement is usually adopted. In this case the speed in rpm is obtained as

$$N = 60 \times \frac{f_c}{Mn} \qquad \textbf{(12)}$$

where $f_c$is the clock frequency in Hz, M is the encoder pulses per revolution, and n is the number of clock pulses in the interval Tw of the encoder pulse.

Tests and Diagnostics

Computer based tests and diagnostics of motor drive systems are easily and quickly carried out either *on-line* or *off-line*. In the first case, the tests are carried when the system is in operation, with the microcomputer acquiring data and executing the necessary application in real time. In the second case, data obtained from the system while operating is analyzed separately on the microcomputer.

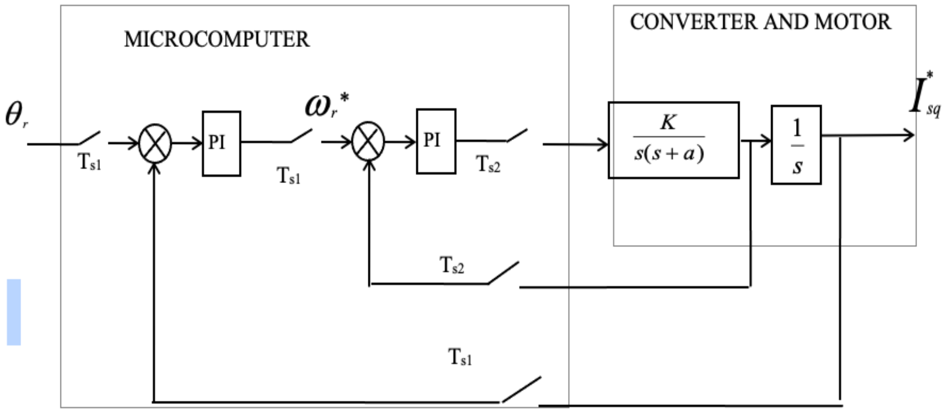

Fig 2:- Multirate position control

The level of skill needed to carry out even the most complicated tests is considerably lowered. Furthermore, because the microcomputer-based procedure is methodical, there is little chance of faulty diagnosis. Also, the need for sophisticated purpose-built test equipment is eliminated and safety problems are minimal during the test.

Another advantage here is the possibility of making the test highly automated without any need for user intervention.

XII. CONCLUSION

Some of the functions that can be implemented on a digital computer in a closed loop motion control system have been briefly described. The need for proper interfacing of the microcomputer to the target control system has been highlighted.

REFERENCES

[1]. BOSE, B.K.: ‘Technology Trends in Microcomputer Control of Electrical Machines’, *IEEE Transactions on Industrial Electronics, 1989,35, (1), pp 160 - 177.*
[2]. BRICKWEDDE, A.: ‘Microprocessor-based adaptive speed and position control for electrical drives’, *IEEE Transaction on Industry Applications, 1985, IA-21, pp1154 - 1161.*
[3]. FRASER, C.J and MILNE, J.S.: ‘Microcomputer Application in Measurement Systems’, *Macmillan Education Ltd, Hounmills, 1990.*