T. R. Fredriksen

Direct Digital Processor Control of Stepping Motors

Abstract: This paper describes the concept for implementing direct computer control of stepping motors. The design approach is based on using the computer to close a minor loop around the motor. The discussion includes both full and partial computer control, and the results show that the closed-loop stepping motor is well suited to such applications.

Introduction

A stored-program processor has been used in a developmental retrieval system* to control a number of positional mechanisms in addition to the regular control data. Several of these mechanisms are powered by dc stepping motors. Depending upon the specific control application, the processor may take direct control or execute actions by means of external hardware. One mechanism, the film chip selection† drive, is detailed specifically, and other installations covered by two generalized applications are also described.

In the past decade, the stepping motor has been employed extensively in open-loop controls, and the classical method of analysis and application has been covered in the literature.[1-9] Recently, it has been shown that certain closed-loop techniques significantly improve the performance of the stepping motor, making it ideally suited for bang-bang control.[10,11]

The next step in the evolution is direct processor control, and the purpose of this paper is to outline the fundamentals of such applications, showing that the closed-loop stepping motor is an external device well suited for discrete control. A set of basic relations is first established defining (1) available computer operation during time sharing and (2) the external device in general. Then the theory of the closed-loop stepping motor is described, giving basic operating characteristics and detailing some of the control hardware. Finally, the principles for direct computer control are illustrated by three typical stepping motor applications of various degrees of complexity. Each case includes precise interface definitions and control concepts. Portions of symbolic programs and actual performance data are presented where appropriate.

* The system utilizes data recorded on film "chips," small strips of photographic film.

† "Chip selection" is a mechanical operation of finding and retrieving one chip from a group stored in a unit cell.

Digital processor control of external devices

• *The concept of sharing*

When a digital computer is assigned to direct control, it normally executes the over-all strategy by operating a number of external devices which in turn apply certain conditions to the physical system under control. An output device, such as a stepping motor, forms but a small link in a complex chain of equipment and can justify only a small portion of the available computer time. The individual device is generally slow compared to the processor, and to use computer control economically it is necessary to operate several devices simultaneously. This calls for a method of sharing the "attention" of the computer for the highest possible use of the computing facility. In general, the share per device is a trade-off between system utilization and external hardware included in the device, and any idle processor time could be an indication of superfluous external logic hardware. To consider total system optimization, however, would be outside the scope of this paper; thus, the following discussion assumes the share to be fixed and defined in terms of the *activity ratio* and the *execution ratio.*

Most external devices occupy a steady state for a finite time during which no attention is required from the processor. As a relative measure of this condition, we define the activity ratio as

$$\beta_i = \frac{\Delta t_{cc}}{\Delta t_{nc} + \Delta t_{cc}}, \qquad (1)$$

where Δt_{cc} is the average time interval when periodic computer control is required and Δt_{nc} is the idle time remaining between the Δt_{cc} groups. If, for example, device No. 3 is required to perform four positioning operations, taking 200, 300, 100 and 600 ms respectively during a

12-second period,

$$\beta_3 = \frac{200 + 300 + 100 + 600}{12{,}000} = \frac{1}{10}. \quad (2)$$

The execution ratio is a measure of actual computation time given a particular device during the active period. Two different conditions exist. The processor may be flagged by the external device when attention is required. In this case, the computer may already be occupied with some other work and, depending on priority, the execution of this new call may be delayed by an interval Δt_d. If the average time between flagging is Δt_f, and the allowable computation time per call is Δt_{ex}, then the execution ratio is

$$\gamma_i = \frac{\Delta t_{ex}}{\Delta t_d + \Delta t_f}. \quad (3)$$

A slightly different situation exists when the processor checks the status of the external devices at a fixed frequency during the active periods. It is assumed that the sampling is synchronized to the termination of a call and that the computer will search a set of sense points (discussed in the next section) for a finite time Δt_{ex} at the end of each sampling interval Δt_s.

Now the execution ratio is

$$\gamma_i = \frac{\Delta t_{ex}}{\Delta t_{ex} + \Delta t_s}. \quad (4)$$

The combination of β and γ gives a measure of the maximum planned share each device has of the total available processing time; but it is not entirely a realistic figure if the sequence of events is only statistically predictable. Rather, a confidence factor should be added that expresses the statistical probability that the i^{th} device will be able to use its share of computation time.

- *Defining the external device*

Because of the large variety of external devices, it is practical to assume that the computer exchanges information with an external device through low-power logic signals. In particular, for this paper, the interface is defined in terms of a binary pair $+L$ and $-L$, which can be suitably identified as ZERO and ONE or ONE and ZERO. Such a definition implies that the external device includes all necessary signal conversion hardware and that all communications with the processor are of a discrete nature. Any control signals from the processor are by virtue of the sharing concept of short duration, and any need for retaining such information for extended periods must be taken care of by appropriate hardware within the external device.

The control interface consists of a set of *sense points* and a set of *actuate points*. A sense point is defined as a physical terminal which, because of its logic level, fully or partially describes the state of the external device. It follows that the state vector in this case is represented by a binary word composed of some ordered sequence of the sense points; each combination corresponds to a specific state of the device.

An actuate point is defined as a physical terminal which, in response to an impressed logic level, initiates an action leading to a known steady state of the external device. Since these points receive rather than generate information, it is appropriate to assign a control vector, $\bar{u}$, to describe the various inputs; thus

$$\bar{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \end{bmatrix} = [u_1, u_2, \cdots, u_m], \quad (5)$$

where $u_i = \pm L$.

The total number of sense points and actuate points must be sufficient to assure adequate performance within the share computer assigned to the particular external device.

Table 1 Windings energized for various steps

Input No.	A_1	A_2	B_1	B_2	*Relative position*[a] *in degrees*
1	x		x		$0 + n(7.2)$[b]
2	x			x	$1.8 + n(7.2)$
3		x		x	$3.6 + n(7.2)$
4		x	x		$5.4 + n(7.2)$
1	x		x		$0 + n(7.2)$

(a) One No. 1 location has been chosen as 0.0°
(b) $n = 0, 1, 2, \ldots 49$

Theory of closed-loop stepping motor operation

- *Fundamental stepping-motor characteristics*

A successful closed-loop application of the stepping motor calls for high-speed solid-state circuitry, and for this reason it is convenient to employ the bifilar synchronous inductor motor. Since recent papers[6,8] have described the theory and operation of this device in substantial detail it will not be described here.

Figure 1 shows a "black box" version of a 200-step bifilar stepping motor.* Four solid-state power switches are capable of energizing the motor windings when receiving the appropriate signals. Only four valid energized winding combinations exist. These are listed in Table 1 and defined as inputs No. 1, 2, 3, and 4.

* A 1.8°/step motor was selected to illustrate the closed-loop principles. Only slight modifications are necessary to adapt the concept to other motor specifications.

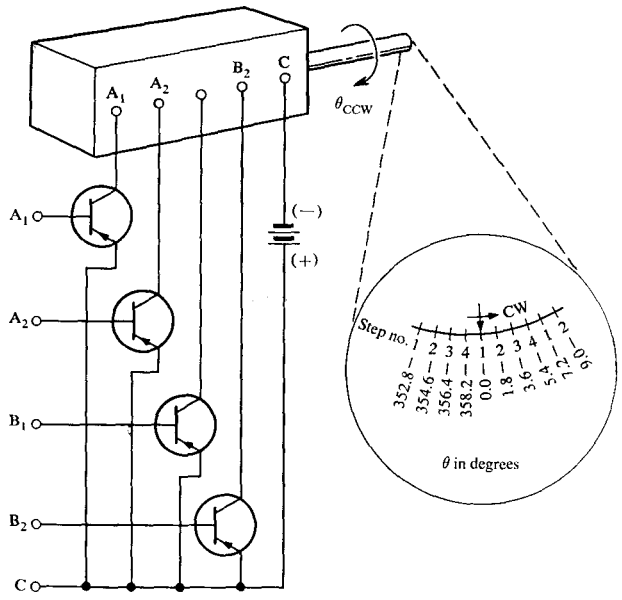


Figure 1 Black box concept of a step motor

The steady-state response to each input could be one out of fifty possible discrete shaft locations spaced 7.2° apart. They are given a "step number" corresponding to the input. The steps are located in a continuous sequence 4, 1, 2, 3, 4, 1, etc. Thus the four operational inputs yield 200 discrete steady-state shaft positions 1.8° apart.

When an input is applied, the motor will develop a torque which moves the shaft towards the *nearest* corresponding step. This immediately establishes a simple but important concept: *From a particular steady-state shaft location, only four possible input-output relations can be initiated.* For reasons which will become apparent later, these are named and defined as follows:

STOP — The motor is energized for a step it is presently occupying.

CW — The motor is energized for a step located 1.8° in a clockwise direction.

CCW — The motor is energized for a step located 1.8° in a counterclockwise direction.

HIGH SPEED — The motor is energized for a step located 3.6° in either direction (see Fig. 1).

Each of these situations has a distinct torque vs shaft position relation. Assuming the input has reached steady state, the torque applied to the shaft on a stationary basis is shown qualitatively in Fig. 2.

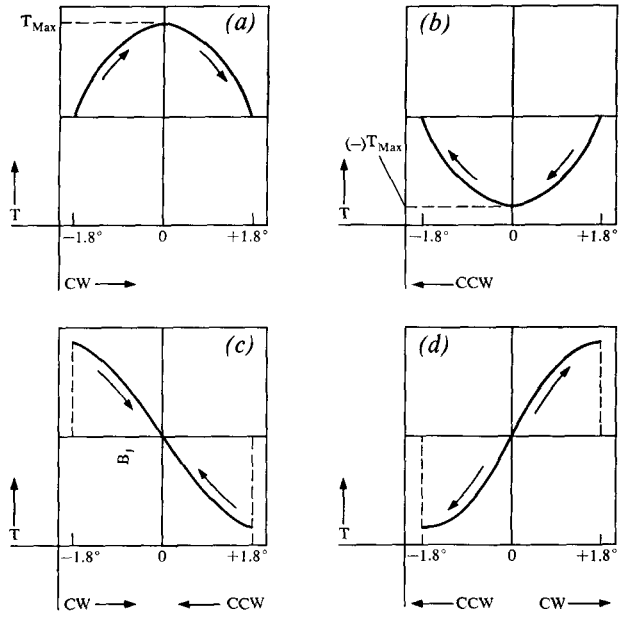
Both a STOP and a HIGH-SPEED input gives ideally zero torque for zero displacement; however, while the stop condition will counteract any attempt to move the shaft, a similar disturbance towards either side in HIGH SPEED will cause the shaft to index two steps in the direction of the disturbance. A second important concept follows immediately: *In order to initiate a desired motor action, it is only necessary to show which step number the shaft is occupying at the time of the decision.* Furthermore, if this information is continuously available and used to change the input accordingly, one should expect continuous motor action.

- *Closing the minor loop*

A minor loop, as shown in Fig. 3(a), provides the piecewise continuous operation of the stepping motor according to the four basic modes of operation. It contains four elements in addition to the stepping motor: a step discriminator, a last-step memory, a power control and a translator. The purpose of the step discriminator is to sense whether the shaft is located in the immediate neighborhood of a steady state position, and if it is, to determine the step number.

The last-step memory amplifies the photosensor signals and applies a zero-order "hold" to one set of outputs which then presents the "memorized" information as to which step was detected last by the step discriminator. The dark-space signal is formed by summing all the discrete step signals and indicates whether the shaft is near a steady state location. Figure 3(b) defines the input-output relation of the step discriminator and last-step memory in more detail when the shaft is rotated in a clockwise direction.

Figure 2 Steady state torque displacement relation for (a) CW, (b) CCW, (c) STOP and (d) HIGH-SPEED motor inputs



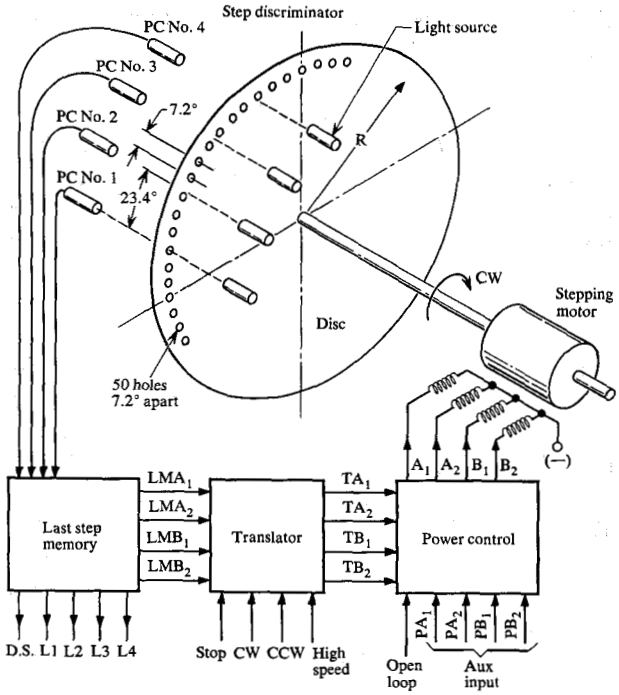


Figure 3(a) Basic minor loop control

The main function of the power control is to drive the power transistors but it also has some important secondary features. For example, the output stages are interconnected to give memory function so that it is necessary only to momentarily trigger a certain motor input in order to obtain a sustained action. A provision for opening the minor loop is added by gating the input lines TA_1, TA_2, and TB_2 and finally, a second independent set of trigger points is provided which allows the driver latches to be modified when the loop is opened.

The purpose of the translator is to execute the continuous motor operation. When the master control selects one of the basic modes CW, CCW, STOP or HIGH SPEED, this logic unit will provide motor inputs based on the information received from the last step memory according to Table 2.

Closed-loop motor characteristics

It can be qualitatively shown that for multistep positional accessing, the closed (minor) loop stepping motor behaves like a second-order system with a pure time delay injected in the execution of a step, i.e., the developed torque lags the initiation point by a time interval fixed for a given motor. This explains the synchronized speed operation of the closed-loop motor and why the magnitude of this speed can be controlled.[10] For instance, if the time lag is Δt_{1g} seconds and the initiation lead is δ_{1d} steps, the synchronized speed is

$$S_{\text{sync}} = \frac{\delta_{1d}}{\Delta t_{1g}} \quad \text{steps/sec} \tag{6}$$

and sign (torque developed) is

$$\text{sign} = \left(\frac{\delta_{1d}}{S} - \Delta t_{1g}\right). \tag{7}$$

Equations (6) and (7) show that the motor will decelerate or accelerate to reach synchronized speed and maintain constant speed for a fixed load condition. By inserting an artificial time lag Δt_c in the step initiation,[12] the synchronized speed can be reduced. For a fixed δ_{1d},

$$S_{\text{sync}} = \frac{\delta_{1d}}{\Delta t_{1g} + \Delta t_c} \tag{8}$$

or in general since Δt_{1g} is fixed

$$S_{\text{sync}} = f(\delta_{1d}, \Delta t_c). \tag{9}$$

In practical computer operations, the artificial time delay may be caused by significant computation time or by delayed attention. Thus, the performance of the motor can be directly affected by the time sharing situation.

Major loop control elements

Figure 4 shows the elements of typical digital servomechanisms. It is assumed that adequate control can be achieved by monitoring position and stepping rate. The step-sequence checker screens the sense signals from the step discriminator and produces a 10 μsec pulse on CWP per step in the 1, 2, 3, 4, 1 direction and a 10 μsec pulse on

Figure 3(b) Last step memory operation

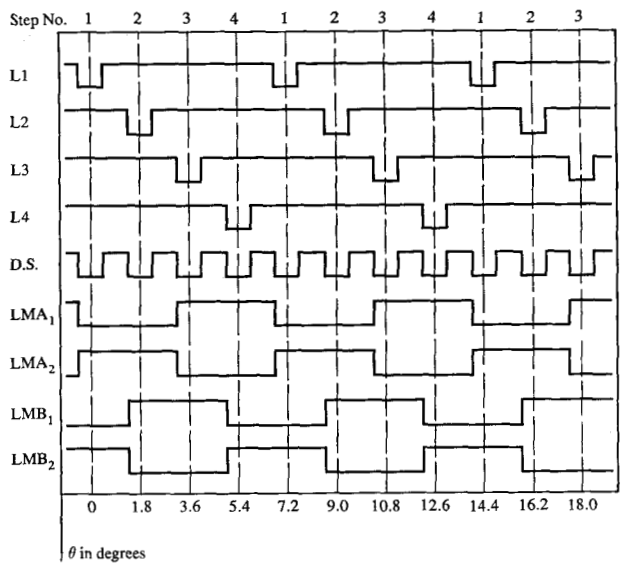
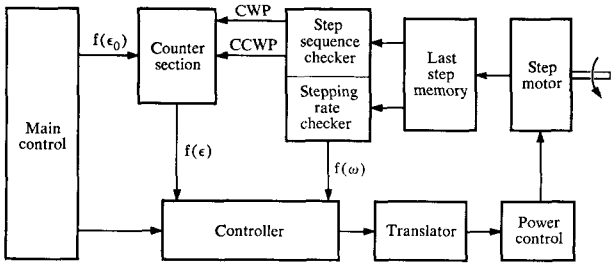


Table 2 Information received from last step memory

COMMAND INPUT	STEP DISCRIMINATOR OUTPUT				
	1	2	3	4	
STOP	1	2	3	4	TYPE MOTOR-INPUT GENERATED
CW	2	3	4	1	
CCW	4	1	2	3	
HISPEED	3	4	1	2	

Figure 4 Digital servomechanism



CCWP per step in the 4, 3, 2, 1, 4 direction. The stepping rate checker measures the duration of a step and gives a discrete output as to whether the speed is above a fixed rate. The step pulses CWP, and the CCWP are accumulated in one or more binary counters and some form of error information is produced with respect to the control input. The controller operates the motor by applying the basic inputs according to the error and the stepping rate, following some prescribed control policy.

The functions of all the discrete elements of both the major and minor loops can, of course, be performed by a digital processor, but for a particular application, it may not be economical to eliminate all the external hardware unless there is sufficient idle computer time available. In general, the available processing time and the motor performance specifications determine where the interface should be drawn between the processor and the stepping motor.

Application and performance data

When a stepping motor is operated in closed loop, a decision is made at every step leading to a new motor input. A logic operation of this nature may only take from 50 to 100 μsec if executed by an efficient subroutine, but since a closed loop stepping motor often runs at 7000 steps/sec such step-to-step control would be entirely inadequate and would require too large a share of the computation time. The alternative is to include some minor loop hardware in the external device and reduce the processor intervention to initial, terminal and other intermittent control according to the frequency and share of attention available in a time-sharing system.

The following three applications require distinctly different processor control and serve to illustrate the minimum external hardware case as well as two types of partial processor control.

- *Low speed step-to-step control*

As was outlined earlier, the closed-loop stepping motor behaves like a synchronous motor if the input is triggered in such a manner as to drive the rotor towards a position one step away from the last detected step. Depending on the motor used the resulting constant rotational speed is in the order of 200 to 500 steps per second. It is significant to note that at this speed most motors are capable of locking into a step position without having to predict a deceleration curve. Thus, the load can be stepped along until the correct position has been reached and then by maintaining this last motor input the motor will halt the load at precisely that step after a short oscillatory period.

In the developmental retrieval system referred to here, the film chip selection mechanism is controlled on such a step-to-step basis. The external device shown in Fig. 5 contains only enough hardware to provide an interface at logic signal levels. The power control has four predriver sections, one for each coil driver, and is equipped with trigger inputs. Thus, a momentary "set" of a particular motor input will be maintained as a zero-order "hold" until a different input is applied. The output of the photocells on the step discriminator is amplified and used to set two flip-flops. Again, this gives a zero-order "hold" on the step sensing, which offers a distinct advantage in a sampled interface because the probability of missing a sense signal is reduced. One pair of the flip-flop outputs is designated as x_1 and x_2 and gives a 2-bit binary coded signal for each of the four types of steps sensed. Two outputs of the trigger operating the power drivers are also brought out as sense points, and this 2-bit code is such that in steady state the step sense x_1, x_2 matches that of the motor input sense x_3, x_4.

The sense point arrangement facilitates a very simple routine for executing the step-to-step operation. First it is recognized that a steady state condition can be assumed to exist if $(x_1 + x_2) = (x_3 + x_4)$. Secondly, from each steady state condition only the two motor inputs corresponding to the adjacent steps need be considered, and

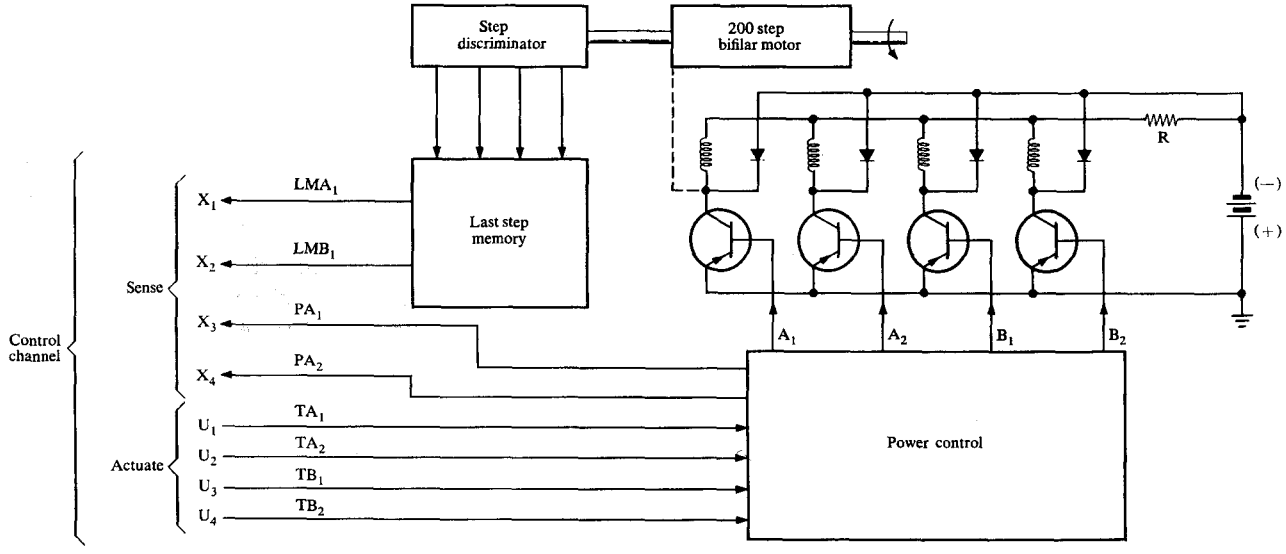


Figure 5 Minimum external hardware

Figure 6 Photographic film chip selection control

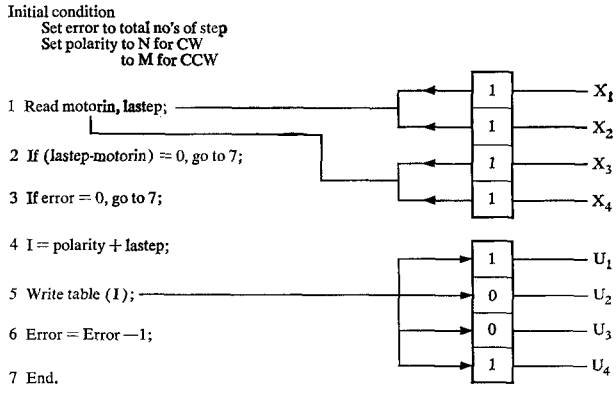


Table (I)

N+0	0	1	1	0
N+1	1	0	1	0
N+2	0	1	0	1
N+3	1	0	0	1
M+0	1	0	0	1
M+1	0	1	0	1
M+2	1	0	1	0
M+3	0	1	1	0

furthermore the specific one to apply is completely defined by the direction of rotation. The routine shown in Fig. 6 combines the steady state condition with the presence of a positional error to initiate a change in motor input and select the new motor input on the basis of the present step and the polarity of the error.

The word ERROR is initially set to the total number of steps the motor must advance and the POLARITY is set to one of two memory addresses N or M, which correspond to clockwise and counterclockwise motion respectively. If N and M are at least four locations apart, we can construct a table as shown in Fig. 6 where the address, I, of any specific element is generated by adding the step information from sense points x_1, x_2 to the POLARITY. The content of any one memory location is simply the new motor input, which should be used to advance the motor one step from the present position in the described polarity or direction.

Assume that a two-step move in the clockwise direction is to be executed and that the motor is occupying Step #1 ($x_1 = 1$, $x_2 = 1$). The processor begins the active period for this device by setting ERROR $= 2$ and POLARITY $= N$. Next, the words LASTEP and MOTORIN are read in by sensing x_1, x_2 and x_3, x_4 as two 2-bit binary words followed by two instructions which checks for "not steady state" and zero error. Since these conditions do not exist, the table address, I, is calculated to be $N + 3$ and the contents of memory location $N + 3$ "written" on the actuate points u_1, u_2, u_3 and u_4 as a 4-bit binary code. This corresponds to a Type 2 motor input and the motor immediately begins to rotate towards the nearest Type 2 step, which in this case is the next clockwise position. The processor terminates the routine by subtracting one from the ERROR and and goes on to other jobs.

After a predetermined sampling period (as in Eq. (4)) the processor again reads the sense points but will exit after the second instruction if Step #2 has not been reached. When Step #2 is sensed the remaining instructions will be executed as described above resulting in $I = N + 2$, TABLE

(1) = [0, 1, 0, 1] and ERROR = 0. One of the successive samples will find LASTEP + MOTORIN = 3 but since the error is equal to zero the processor exits, branching to the next activity. Thus, the motor locks in and stays at Step #3, having moved two steps in clockwise direction as required. Figure 7 shows a typical velocity response of a 100-step access performed by a routine similar to the one described. The velocity is quite constant, indicating a well-behaved sampled attention from the processor.

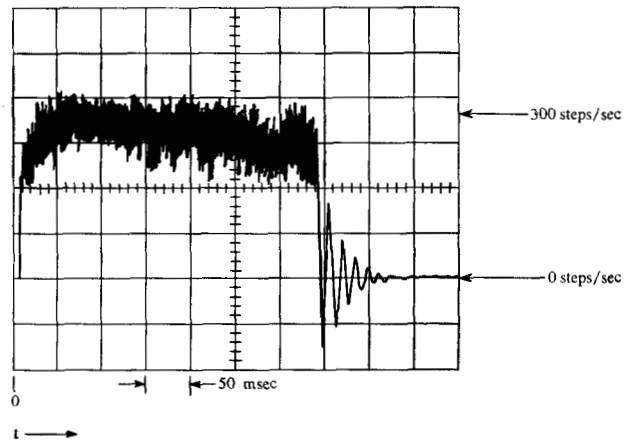


Figure 7 Velocity trace of a 100-step access (chip selection)

• *A two-speed servomechanism*

When high-speed operation is required it is not in general practical to have the processor control the closed loop stepping motor on a step-to-step basis; instead, a translator is added to the external hardware to take care of the detail motor operation during certain control modes. The positioning device now also includes two counters, a step sequence checker and a stepping rate checker, as illustrated in Fig. 8. The combined effect is that of an "auto pilot" which can take over control in periods when it is not practical to demand processing time.

The sense word, x, has the four components defined in Fig. 6, but in addition, a fifth has been added to give some information about the stepping rate. In particular,

$$x_5 = +L \Rightarrow S > S_1$$

$$x_5 = -L \Rightarrow S \leq S_1, \qquad (10)$$

where s_1 is a fixed stepping rate threshold. The other sense words y and z give a cumulative count of the clockwise and the counterclockwise steps taken by the motor since the last reset.

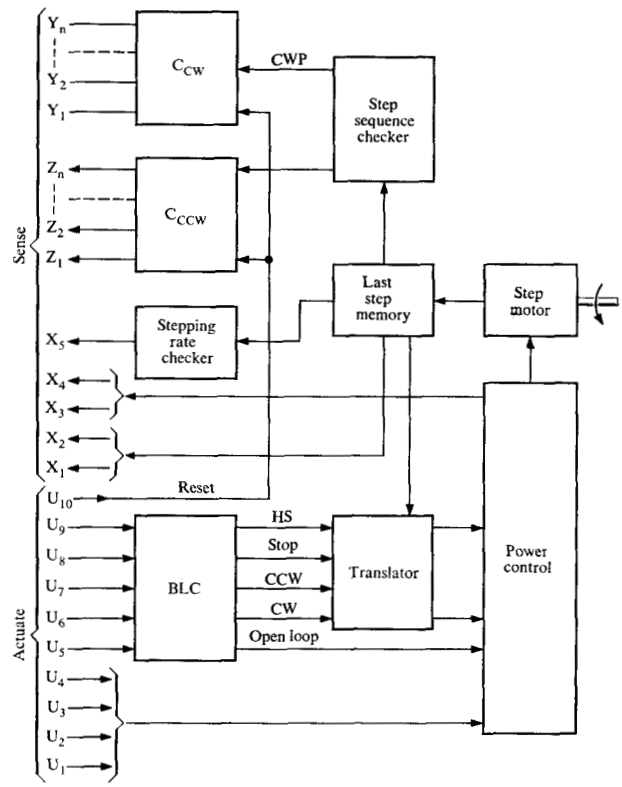
The control word has been expanded by six components, five of which are interfaced through a BLC (Binary Latch Channel) assuring a zero order hold. Component u_5, controls the minor loop closing such that

$$u_5 = -L \Rightarrow \text{loop is open}$$

$$u_5 = +L \Rightarrow \text{loop is closed.} \qquad (11)$$

The next four components, $u_6, \cdots u_9$ control the four basic minor loop command inputs, giving CW, CCW, STOP and HIGH SPEED respectively for a $(-L)$ input to the BLC. Finally, u_{10} resets the two counters when receiving a $(-L)$ input.

Figure 8 Two-speed positioning device



• *The autopilot control philosophy*

This device operates as a two-speed servo and requires only intermittent processor "attention". During an access, the processor controls directly the first few steps, initiates deceleration from high speed and secures the terminal steps. The steady state for inactive periods is identical to the previous device since the minor loop is open $[u_5 = (-L)]$ and the sense word, x, indicates which step the motor shaft occupies. An operational command is generated within the processor, establishing the number of steps and which direction to move. The control sequence begins with resetting the counters to zero then advancing the motor much the same as shown in Fig. 6. This continues until $x_5 = -L$, which indicates that the stepping rate is above S_1. The processor now selects HISPEED and closes the minor loop.

Table 3 Two-speed program concept

```
SET UP: ACCESS = ± TOTAL NO'S OF STEPS
        HISPEED = NO. STEPS TO DECELERATION
        WRITE CONTROL = (0, 0, 0, 0, 0, 1)

L1: READ NCW, NCCW, SPEED;
    IF (HISPEED—ABS (NCW-NCCW) — 0 GOTO L2;
    IF SPEED = 0 GOTO L3;
    WRITE CONTROL = (1, 0, 0, 0, 1, 0);

L2: IF SPEED = 0 GOTO L3;
    WRITE CONTROL = (1, 0, 0, 1, 0, 0)

L3: WRITE CONTROL = (0, 0, 0, 0, 0, 0)
    READ MOTORIN, LASTEP
    IF (LASTEP-MOTORIN) ≠ 0, GOTO L1;
    READ NCW, NCCW;
    IF (ERROR — NCW + NCCW) = GOTO L4
    POLARITY = SIGN (ACCESS-NCW + NCCW) × 2 +
      N + 2;
    I = POLARITY + LASTEP;
    WRITE TABLE (I);
    GOTO L1

L4: END
```

LASTEP		MOTORIN		SPEED	NCW	NCCW	
X_1	X_2	X_3	X_4	X_5	$Y_1Y_2\cdots Y_n$	$Z_1Z_2\cdots Z_n$	SENSE

TABLE (I)				CONTROL						
U_1	U_2	U_3	U_4	U_5	U_6	U_7	U_8	U_9	U_{10}	ACTUATE

By sampling the counters and comparing the difference to stored information, the processor keeps track of the stepping progress. Near a fixed number of steps from the target, the processor again intervenes and changes the control setting to STOP. The motor decelerates towards zero and when x_5 switches from $-L$ to $+L$ ($S \leq S_1$) the computer is flagged. Now direct control is resumed by opening up the minor loop and the step-to-step program is continued until the load has been rotated the required number of steps.

• *Two-speed program concept*

The main concept for programming the two-speed closed loop stepping motor is outlined in Table 3. Again, no attempt has been made to present an accurate program and many details have been left out for the sake of clarity. The preliminaries consist of setting up ACCESS and HISPEED for the correct total number of steps and high-speed steps respectively; also, the counters are reset to zero.

The program has three main blocks which execute high-speed control, stop control and step-to-step control, depending on positional error and stepping rate. If the speed is below S_1, the processor operates the motor as outlined by block L3. One major difference exists from the program in Fig. 6; the direction of motor is determined by the relative polarity of the error. Thus, the system will correct for overshoot. By taking the difference between NCW and NCCW as the *net* step advance, the effect of oscillations has been eliminated.

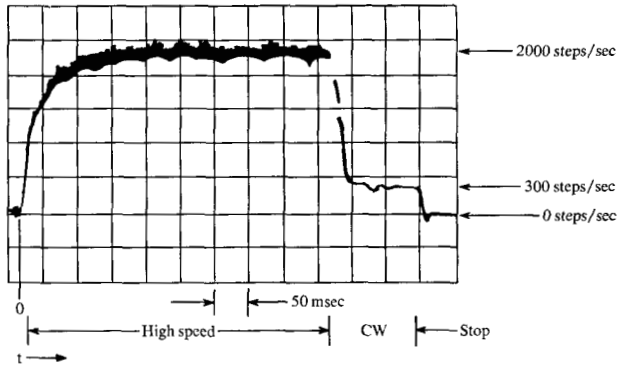
When the speed increases above S_1, the control is given to the external device for high speed as long as the net number of steps advances is less than HISPEED. Any other condition results in a STOP control being applied to the closed minor loop. Depending on the degree of optimization required, the search for deceleration on high-speed cutout may be critical. This can be overcome by proper priority control and variable sampling rate. As a final check on the terminal position, each address can be associated with a stable state, and if not achieved, at the end, corrective action taken.

A typical velocity trace of a two-speed control is shown in Fig. 9. In this case, the high-speed cutout has been predicted with substantial margin to compensate for possible lack of immediate computer attention.

• *A digital, time-optimal servomechanism*

Some positional mechanisms must be operated time-optimally to meet the performance criteria. In such cases, it is not suitable to have the computer perform any function that allows the inherent random attention delay to deteriorate the time response. Fortunately, the closed-loop stepping motor exhibits such high repeatability in responding to a specific command that it is practical to predict the required control sequences. This means that instead of giving intermittent "attention" during an active period for a particular device, the processor allots a larger amount of computation time in one block prior to execution. During this "set up" interval, the "autopilot" in the

Figure 9 Two-speed velocity trace



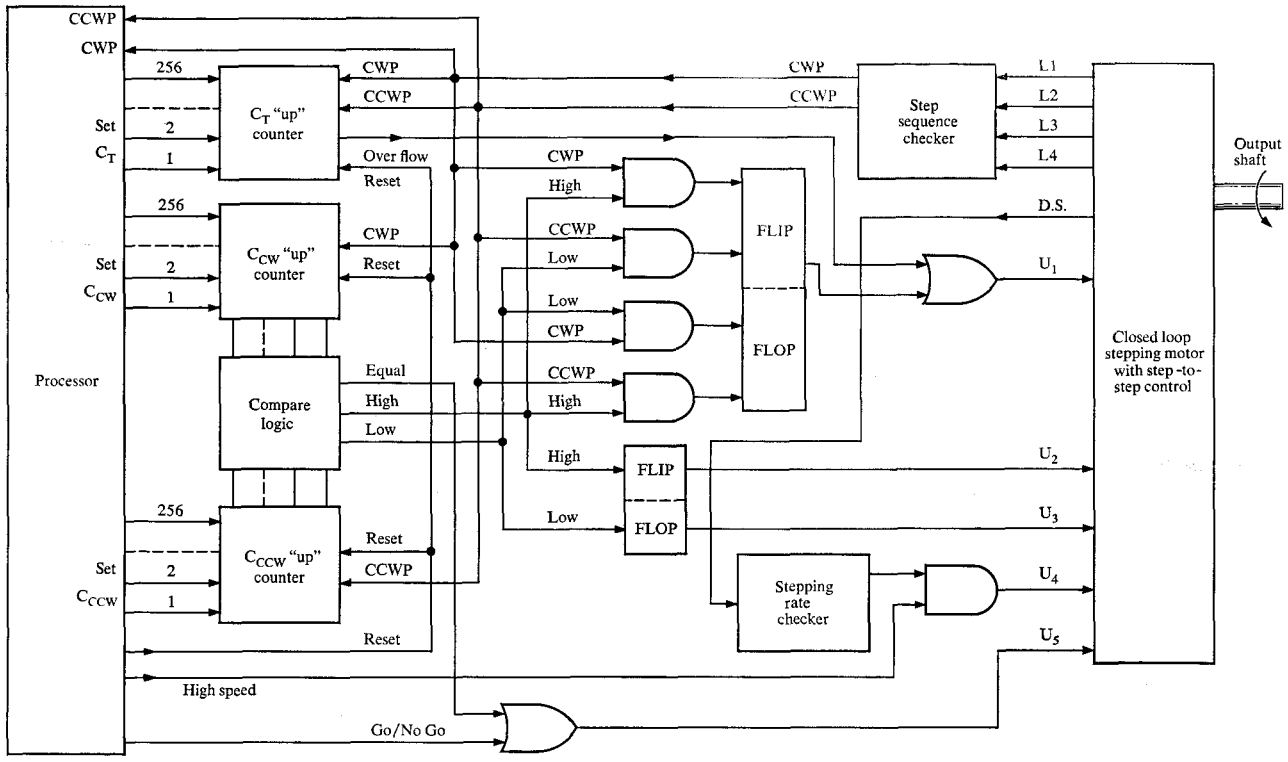


Figure 10 A digital servomechanism

external device is preset or programmed to perform a certain control sequence compatible with distinct states which the device is expected to reach.

The X-motion, chip picker (film extractor) drive is a typical example of such a situation and Fig. 10 gives a general block diagram of the external device. A step-to-step control has been added to the minor loop (not shown) which terminates all access moves in a perfectly time-optimal fashion, i.e., a no-overshoot, deadbeat response. The counters C_{CW} and C_{CCW} accumulate the steps as before, but now one of them receives an initial disturbance in the form of a preset count. The internal compare circuit translates the instantaneous difference between C_{CW} and C_{CCW} into an error signal which is used to set the basic direction (u_2, u_3) of the motor.

A third counter has been added to store the predicted deceleration control. The processor obtains this information from the given initial conditions (number of steps, direction) and from trajectory tables stored in memory. The operation of the external device can easily be monitored by the processor. The status of the counters can be read at any time during a curve and the exact speed obtained by clocking the intervals between two pulses from the step sequences. If this information is compared to the trajectory tables, the processor can determine whether the device is on target and if not, take emergency measures.

• *System operation and response*

Assume that at $t = 0$ the system is at rest and the processor generates a need for a 176-step rapid access in the clockwise direction. A search of the final trajectory table establishes that $\epsilon_T = 355$, and this number is loaded into the binary counter C_T, while C_{CCW} is set to 176. Immediately, the comparator changes from equal to high, setting $u_2 = 1$, $u_3 = 0$ and $u_5 = 1$. The fact that $NC_t \neq 512$ gives $u_1 = 1$, and we have the control vector $\bar{u} = [1, 1, 0, 0, 1]$, which initiates a cw acceleration trajectory. When the stepping rate exceeds S_{min}, $u_4 = 1$, which initiates the highspeed control vector $\bar{u} = [1, 1, 0, 1, 1]$. The stepping motor now accelerates to the maximum stepping rate, each step adding one to the count C_T and C_{CW}. After 156 steps, $NC_T = 512$, and the control changes to $\bar{u} = [0, 1, 0, 1, 1]$. This forces the motor to decelerate sharply along a final trajectory until $N_{CW} = 176$, when the control vector terminates in $\bar{u} = [0, 1, 0, 0, 0]$. Since the system is in equilibrium, no further motion is encountered.

The actual system response to an initial error of 176 steps is reproduced in Fig. 11(a) and (at least on such a scale) the trace seems to be time-optimal. Figure 11(b) shows an enlargement of the region around the final step which shows that the motion terminates without any overshoot, supporting the time-optimal case.

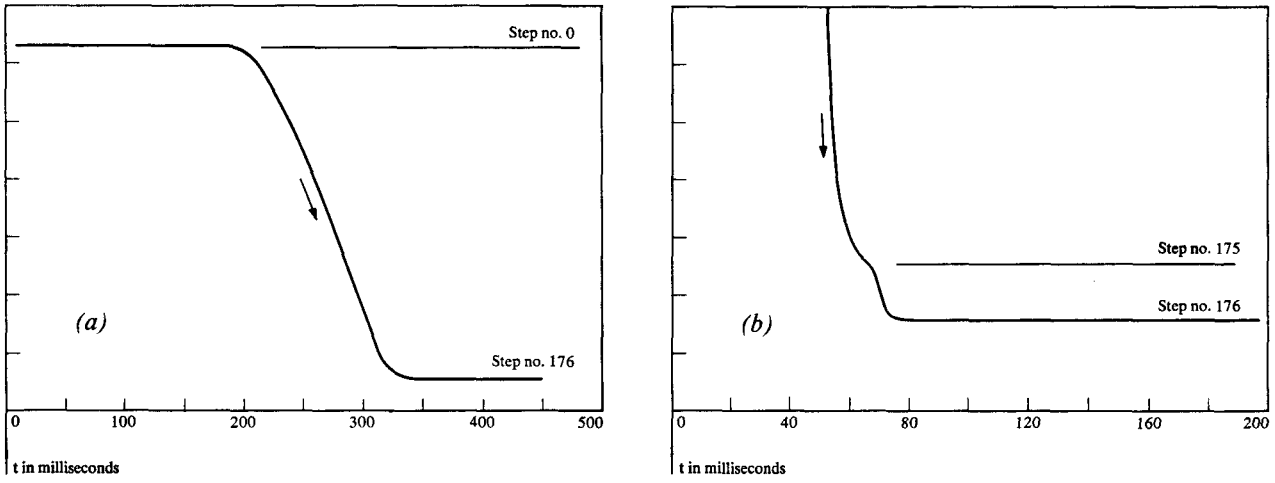


Figure 11 High-speed access response. (a) positional trajectory for 176-step high-speed access. (b) Equilibrium details

The high-speed access is but one of the many possible modes of operation. For example, the processor can select synchronized speed which, with slight modifications, can be controlled in magnitude directly from the processor. The step-to-step repeated access is also simply implemented by setting C_T to 512 at $t = 0$.

Conclusion

The stepping motor is a truly digital actuator and can be operated effectively by a digital processor. In this paper the basic approach is based on the closed-loop stepping motor, which features digital feedback control. It has been shown that the processor can perform this feedback operation but that in higher speed application an excessive amount of computation time is required. An attempt has been made to define a set of values by which the share of processor "attention" can be measured and to define an external device of some generality. The various stepping motor devices can be handled conveniently by this terminology, and hopefully these definitions will be useful for other types of external devices.

The main theme in this paper has been to emphasize the importance of selecting a processor-device interface so as to optimize the portion of time sharing, making it compatible with the performance expected.

The three applications described require varying degrees of processor time sharing if the external hardware were kept constant. However, due to the flexibility of stepping motor control, the use of excessive processor control time is avoided by adding elements to the external device. In short, because the closed-loop stepping motor is a digital device the interface can be set to minimize the cost of a unit function.

References

1. J. Proctor, "Stepping Motors Move In," *Product Engineering* **34**, 74–78 (February, 1963).
2. A. G. Thomas and J. F. Fleischauer, "The Power Stepping Motor—A New Digital Actuator," *Control Engineering* **4**, 74–81 (January, 1957).
3. S. J. Bailey, "Incremental Servos," *Control Engineering*: (a) "Introduction," **7**, 123–127 (November, 1960): (b) "Operation and Analysis," **7**, 97–102 (December, 1960); (c) "Applications," **8**, 85–88 (January, 1961); (d) "Industry Survey," **8**, 133–135 (March, 1961); (e) "Interlocking Steppers," **8**, 116–119 (May, 1961).
4. N. L. Morgan, "Versatile Inductor Motor for Industrial Control Problems," *Plant Engineering* 16, 143–146 (June 1962).
5. G. Baty, "Control of Stepping Motor Positioning Systems," *Electromechanical Design* 9, 28–39 (December, 1965).
6. A. E. Snowden and E. M. Madsen, "Characteristics of a Synchronous Inductor Motor," *Trans. AIEE* **8** (Appl. and Ind.), 1–5 (March 1962).
7. J. P. O'Donahue, "Transfer Function for a Stepper Motor," *Control Engineering* **8**, 103–104 (November 1961).
8. R. B. Kieburtz, "The Step Motor—The Next Advance in Control Systems," *IEEE Trans. on Automatic Control* **AC-9**, 98–104 (January 1964).
9. B. A. Segov, "Dynamics of Digital Routine Control Systems With a Step Motor," *Automation of Electric Drives* (in Russian), No. 4, 38–61 (1959).
10. T. R. Fredriksen, "Closed Loop Stepping Motor Application," 1965 Joint Automatic Control Conference Paper, pp. 531–538.
11. T. R. Fredriksen "New Developments and Applications of the Closed Loop Stepping Motor." 1966 Joint Automatic Control Conference Paper, pp. 767–775.
12. See Ref. 10, p. 534.

Received February 28, 1966