AN EXPERIMENTAL DIGITAL FLIGHT CONTROL SYSTEM

Maier Margolis - J. B. Rea Company, Inc., Santa Monica, California
Eric Weiss - Dynalysis, Inc., Los Angeles, California

Introduction

Development of aircraft with large ranges in speed and altitude has resulted in a need for autopilot systems with optimum control characteristics over a wide range of conditions and with very high accuracy and resolution. In designing a digital flight controller to pilot a highly maneuverable aircraft whose flight conditions are constantly changing, an investigation has been made into the necessary and desirable control characteristics. This paper describes the control equations resulting from this investigation as well as the special purpose breadboard model computer which is being tested and analyzed in conjunction with an analog flight simulator (see Figure 1).

The breadboard model computer-controller has been built for the sole purpose of proving that digital flight controlling is possible and advantageous over conventional analog methods. No effort has been made to package the unit or to make it flyable. Even though the machine has a fixed program, it is flexible enough to permit considerable freedom in choice of control equations and parameters for research study, as will be described later in the text.

The flight controller will direct the aircraft in three modes of operation: (1) Altitude control, (2) Attitude control, (3) Pitch rate control. These modes of operation must have good stability and fast response (Technical Report, Project 1, Phase 1 under Air Force contract AF 33(616)-273 entitled "Preliminary Investigation", classified Confidential, describes the necessary control and stability criteria for a linear system). An increase in the speed of response is provided through programming a nonlinear control function. At the same time the airframe must be guarded against excessive maneuvers that might bring the airframe into unsafe stall or structural load conditions. Therefore, limiting the airframe maneuvers so as not to exceed safe stall and load limits has been provided. These nonlinear controls, so difficult to perform in analog computing systems, lend themselves directly to a digital control system. The fire control command signals to the flight control system are usually exceedingly noisy. Therefore, the last nine pieces of data are digitally weighted and added to provide a smoothed input. Investigations show that digital smoothing by means of weighting discrete readings is more advantageous than comparative analog filtering methods.

Development of Control Equations

When the digital flight controller is functioning to maintain pitch rate control, the error signal commanding a change in control surface position becomes (definition of symbols will be found in the Appendix):

$$\Delta = \lambda = k_e(\dot{\theta}_D - \dot{\theta}_o) \tag{1}$$

$$\text{where } \lambda = K_{\dot{\theta}}(\dot{\theta}_D - \dot{\theta}_o) \tag{2}$$

Pitch rate control is furnished by the inner or primary loop of the basic control system illustrated in Figure 2. When operating in the pitch rate control mode, the command is $\dot{\theta}_D$ and the outer loops are broken at the attitude error input. The pitch rate command signal $\dot{\theta}_D$ will be provided by the fire-control computer; a discussion of the smoothing necessary on this signal will be found later in this paper.

When the digital flight controller is operating to maintain attitude control,

$$\Delta = \lambda = k_e\left[k_7(\theta_D - \theta_o) - \dot{\theta}_o\right] \tag{3}$$

$$\text{where} \quad \lambda = K_\theta(\theta_D - \theta_o) - K_{\dot{\theta}}\dot{\theta}_o \tag{4}$$

Attitude control is furnished by the primary loop plus the position feedback θ_o as illustrated in Figure 2. When operating in the attitude control mode, the command is θ_D and the outer loop is broken at the altitude error input. The error in attitude multiplied by the proper gain $K_\theta/K_{\dot{\theta}}$ becomes the new desired pitch rate.

When the flight controller is operating to maintain altitude,

$$\Delta = \lambda + k_7k_e\theta_o = k_e\left[k_7k_f(h_D - h_o) - \dot{\theta}_o\right] \tag{5}$$

$$\text{where } \lambda = K_h(h_D - h_o) - K_\theta\theta_o - K_{\dot{\theta}}\dot{\theta}_o \tag{6}$$

Altitude control is furnished by the entire system as illustrated in Figure 2. The command as shown is h_D. Since Δ is used in nonlinear control and in maneuver limiting, it is essential that Δ becomes zero in the steady state. For this reason the term $-k_7k_e\theta_o$ is intentionally omitted from Δ and will be introduced in equation 8.

The equations for the three modes of operation may be combined into one equation for convenience:

$$\Delta = k_e\left[k_7k_f(h_D - h_o)\ \textcircled{1}\ + k_7(\theta_D - \theta_o)\ \textcircled{2}\ + \dot{\theta}_D\ \textcircled{3}\ - \dot{\theta}_o\right] \tag{7}$$

where ① stands for altitude control
② stands for attitude control
③ stands for pitch rate control

The symbols (1) , (2) , and (3) have the value 1 if the particular control applies, and have the value 0 if the control does not apply. Equation 7 can be shown to reduce to equation (1), (3), or (5) as the mode of control changes.

In order to maintain a steady-state flight path, integral (or trim) control must be provided. The desired control surface position then becomes

$$\delta_d = \Delta + k_{12} \int_{t_{-\infty}}^{t_o} \Delta dt - k_7 k_e \theta_o \, (1) \tag{8}$$

where (1) = 1 for altitude control and (1) = 0 for attitude and pitch rate control. The integral term $k_{12} \int_{t_{-\infty}}^{t_o} \Delta dt$ will provide the trim control necessary to fly the aircraft along a steady-state flight path. The term $-k_7 k_e \theta_o \, (1)$ maintains the proper damping during altitude control and is necessary since the rate gyro signal does not provide the required sensitivity.

The nonlinear control function which the digital flight controller will provide to improve the speed of response of the aircraft while still maintaining the desired damping will be programmed as follows (see Figure 3):

$$\delta_D = \delta_d \quad \text{if} \quad k_8 < \Delta < k_9 \tag{9}$$

$$\delta_D = k_{10} \quad \text{if} \quad k_8 \geq \Delta \tag{10}$$

$$\delta_D = k_{11} \quad \text{if} \quad \Delta \geq k_9 \tag{11}$$

As can be seen from equation 7, Δ defines the transient error, which ultimately determines the desired control surface position. As used in equations (9), (10), and (11), the magnitude of Δ determines when the nonlinear control action is desirable.

At the same time this programming and computing is taking place to determine the proper control surface position, the necessary limiting to keep the aircraft maneuver within safe stall and load bounds is being calculated. The basis for the stall limiting scheme is to make certain that the lift available will always keep the plane aloft; accordingly

$$qs \, C_{L_{max}} \geq n_z mg \tag{12}$$

And the basis for the load limiting scheme is to make certain that the load will not exceed the critical load and snap the aircraft's wings off; therefore,

$$n_{z_{lim}} \geq n_z \tag{13}$$

The following equations will permit maximum response while keeping the maneuver within the safe load and stall limits by predicting the expected response as a function of the limited command signal $\bar{\Delta}$, where

$$\bar{\Delta} = \Delta \quad \text{if } \Delta < k_{13} \tag{14}$$

$$\bar{\Delta} = k_{13} \text{ if } \Delta \geq k_{13} \tag{15}$$

$$d_s = qk_a - k_b m_1 = k_1(qS\, C_{L_{max}} - n_z mg) \tag{16}$$

$$d_g = k_4 - k_b = k_2(n_{z_{lim}} - n_z) \tag{17}$$

$$d_L = d_g \text{ or } d_s \text{ , whichever is smaller} \tag{18}$$

$$k_c = k_5 + k_6 \bar{\Delta} \tag{19}$$

$$k_d = 1 \text{ if } \Delta < 0 \text{ or if } k_b < 0 \text{ or if } d_L > k_c \tag{20}$$

$$k_d = \frac{d_L}{k_c} \text{ unless } k_d = 1 \text{ as per equation (20)} \tag{21}$$

The error signal to the control surface servo, defined by

$$\varepsilon_i = \delta_e - k_d \delta_D \tag{22}$$

will move the control surface according to equation (22). The variable multiplier k_d computed as defined by equations (20) and (21) and illustrated in Figure 4 is utilized to limit the aircraft to safe maneuvers. An examination of Figure 4 discloses that the aircraft response will not even begin to be limited unless, first, the aircraft is close to critical stall or load limits, or, second, the command signal $\bar{\Delta}$ will carry the aircraft to the critical limits before limiting might otherwise be obtained. If the command signal $\bar{\Delta}$ is small and the aircraft is operating near the critical stall limits, maneuverability without danger is assured; Figure 4 demonstrates that fact.

δ_D as defined in equations (9), (10), and (11) to provide nonlinear control, must be modified so that stall and load limits can be accurately applied. Therefore, equations (9), (10), and (11) become:

$$\delta_D = \delta_d \quad \text{if } k_d \neq 1 \text{ or if } k_8 < \Delta < k_9 \text{ and only if } k_{10} < \delta_d < k_{11} \tag{23}$$

$$\delta_D = k_{10} \text{ if } k_d = 1 \text{ and } \quad k_8 \geq \Delta \quad \text{or if} \quad k_{10} \geq \delta_d \tag{24}$$

$$\delta_D = k_{11} \text{ if } k_d = 1 \text{ and } \quad \Delta \geq k_9 \quad \text{or if} \quad \delta_d \geq k_{11} \tag{25}$$

These decision equations make it possible to eliminate the nonlinear control action whenever limiting resulting from adverse stall and load conditions is imminent.

The equations to be solved by the breadboard digital flight controller computer have, therefore, been established as shown in Figure 5. These control equations are modified slightly when the flight control system is functioning in the fire control mode. The input command is then $\dot{\theta}_{DN}$, i.e., $\dot{\theta}_D$ with noise. To eliminate as much of the undesired noise as possible $\dot{\theta}_{DN}$ is smoothed and becomes $\dot{\theta}_D$ in the above control equations.

Therefore, $\dot{\theta}_D$ is the sum of the last nine pieces of information summed with the proper weighting constants.

$$\begin{aligned}\dot{\theta}_D(t_o) = {} & C_o\dot{\theta}_{DN}(t_o) + C_1\dot{\theta}_{DN}(t_{-1}) + C_2\dot{\theta}_{DN}(t_{-2}) \\ & + C_3\dot{\theta}_{DN}(t_{-3}) + C_4\dot{\theta}_{DN}(t_{-4}) + C_5\dot{\theta}_{DN}(t_{-5}) + C_6\dot{\theta}_{DN}(t_{-6}) \\ & + C_7\dot{\theta}_{DN}(t_{-7}) + C_8\dot{\theta}_{DN}(t_{-8})\end{aligned} \tag{26}$$

where t_o is the present instant, t_{-1} is the previous instant, t_{-2} is the next to the previous instant, etc. These weighting constants may be easily altered without introducing any new components in the computer. This digital filter will enable a rather complete survey and analysis of the effect of various weighting functions.

Computer

The breadboard computer built for this project does all the necessary computation described in Figure 5 as well as the following: scan the input disks; convert the resulting information from reflected-code into true binary numbers; compute the smoothed value of $\dot{\theta}_D$; and convert the output into a voltage of continuous step function nature.

The computer is a serial machine. Its computation cycle is broken down into 63 word-times; each word-time consists of 20 clock pulses. Thus, a computation cycle consists of 1260 clock pulses.

The computer has three arithmetic registers. Two of these registers (F and G) are 20 bits long. The third register (E) is eight bits long. In addition to these registers the computer has a magnetic drum memory which (1) stores all constants on a permanent memory channel, (2) has one seven word re-circulating line which stores the trim-integral as well as some temporary data, (3) has two delay lines 8 words long used primarily for smoothing the fire control computer signal, and (4) has an additional short delay line used to convert reflected-code numbers into true binary representation. All of these registers will be further described in the following paragraphs.

In order to better understand the operation of this unit let us examine each function separately.

Input Scanning and Conversion

All data utilized by this computer is scanned from 13 commutator disk assemblies. Six of these assemblies consist of two disks in series for better resolution. The computer scans these inputs one at a time serially, least significant digit first, by means of a diode matrix. The scanned data of all 13 inputs eventually appears in one single flip-flop I_r , in reflected-code.

It is well known that a number could not be read off directly in a true binary form (where each digit represents a power of two) from a moving commutator disc because it is impossible to get all the digits concerned to change at exactly the same instant. Thus, for example, if the number were to change from 15 to 16, any number between 0 and 31 could be read due to misalignment of one or more brushes. To avoid this difficulty, the reflected binary code was developed by the Bell Telephone Laboratories. This code has the feature that a change of one increment in the total number represented requires a change of only one bit.

Several methods for decoding such a reflected code have been devised in the past. However, these methods either convert the whole number in parallel (that is, all bits are observed simultaneously), or most significant digit first (that is, the most significant digit is observed and operated upon first and the least significant digit last). It is well known that a serial digital computer has to operate least significant digit first, and out of this consideration the following method for decoding a reflected-code, least significant digit first, has been devised.

If the count of 1's of higher significance than the digit under consideration in the reflected-code number is odd, the digit should be inverted; if it is even, the respective digit should stay unchanged. (The word "invert" amounts to substituting a "0" for a "1" and vice versa).

The following arrangement will convert a number according to the method just described, where the number circulates timewise, least significant digit first. Figure 6 is a block diagram showing the elements of the decoder.

The reflected-code number to be converted is carried serially, least significant digit first, in flip-flop I_r . A delay line feeds the reflected-code number from this flip-flop I_r into another flip-flop I_0 so that flip-flop I_0 carries the reflected-code number at least one word-time later than it was

represented in flip-flop I_r .

Flip-flop I_a carries the reflected-code number delayed by one additional clock-period from flip-flop I_0 , thus trailing I_0 by one clock-time.

Flip-flop I_c is a single-stage binary counter which changes its configuration every time a "1" is in flip-flop I_r or in flip-flop I_0 . Flip-flop I_c has to start out in the "0" configuration and will automatically leave itself in the "0" configuration after a word has been converted.

This conversion is done in two word-times. During the first word-time I_c counts the 1's in the reflected-code number and determines whether the count of 1's is even or odd. During the second word-time, without resetting, the same flip-flop counts the 1's in the reflected-code number again. The complete reflected-code number is also delayed by one clock-time through flip-flop I_a . The decoded output will be the configuration of I_a if I_c is in the "0" configuration; the output will be in the inverted condition of I_a if I_c is in the "1" configuration.

Smoothing of $\dot{\theta}_{DN}$

Smoothing of $\dot{\theta}_{DN}$ is accomplished by effectively multiplying the last nine data points of $\dot{\theta}_{DN}$ by constants and summing these products to obtain the smoothed value. The computer accomplishes this function in the following manner: the last eight data points are stored in a precessing re-circulating line A , eight words long. This line is automatically kept up to date by replacing a data point every computation cycle, which is equivalent to a revolution of the drum or 63 word-times. This method automatically replaces the oldest data in that line with the newest and advances the relative position of the information in it by one word every computation cycle.

During each computation cycle the data in the eight words of A is either added or not added to another re-circulating line B . The eight products of these data points times their respective weighting constants are thus formed in B. No attempt is made to include algebraic sign at this point. If the machine is operating on pitch rate control, the products in B are either added or subtracted into an arithmetic register during the beginning of the following computation cycle. Then the latest data of $\dot{\theta}_{DN}$ is multiplied onto this sum. The above computation results in the smoothed value of $\dot{\theta}_D$ in this arithmetic register. If the machine is operating on altitude or pitch control, terms $k_7\, k_f(h_D - h_o)$ (1) or $k_7(\theta_D - \theta_o)$ (2) are respectively computed in the arithmetic unit during this time.

Constants

All constants are stored in a special channel K , on the magnetic drum. The weighting constant digits are always stored in the first digit of a word-

time in the K channel. The other spots in this channel are used to store all other constants used throughout the computation. The ease with which the constants can be changed is the main reason for storing them in this manner. In the upper right-hand corner of the computer (see Figure 9) a set of toggle switches is located which enables the operator to arbitrarily change information in the K channel, one bit at a time, by simply setting the switches to the desired word and digit number. A "1" or "0" can then be recorded by setting an additional switch and depressing a push-button. A lock and key is provided on the recording circuit to prevent any unauthorized "knob-twisters" from altering any information in this channel. Such mis-information might be difficult to detect if not suspected. These switches also enable the operator to trigger his oscilloscope at any desired time as selected by the switches.

It should be pointed out that this K channel on the drum can be eliminated once all these constants are established and built into the computer permanently.

Control Computation

The computer follows the equations given in Figure 5. The timing of the computer is so arranged that each word of information used for computation is scanned off the input disks at the proper time so that it can be used immediately when available. Because of this feature, it should ultimately be possible to eliminate the magnetic drum memory altogether.

All computation necessary to determine k_d , namely equations (14) through (21) of Figure 5 are performed utilizing a ten-digit word. This ten-digit word is sufficiently accurate for that purpose. Machine-wise this ten-digit word is achieved by splitting, at specific times, the G-register into two ten-bit registers and performing these operations simultaneously with other operations in the remaining arithmetic registers. As a result considerable computation time is gained. The miscellaneous comparisons described in Figure 5 are performed whenever the necessary information is available, and the resulting decisions are sometimes stored in flip-flops until they can be utilized.

Output

The output of the computer is ε_i as illustrated in equation (22). It is easy to see that ε_i can be of very large magnitude when a large input command is given. The control surface is rate and position limited, consequently, starting with the third least significant digit, only seven binary digits are converted into a DC voltage output. If ε_i is of a magnitude larger (or smaller) than the maximum (or minimum) that can be represented by these seven binary digits, the computer reaches the decision to substitute for it the maximum(or minimum) expressed by these seven digits. This information is then stepped into seven flip-flops which keep it for one computation cycle until newer information is available.

The method used to convert from a binary number to a voltage is well established (see Figure 7). The binary number is shifted into an output register made up of seven flip-flops. Each flip-flop has a voltage output which is accurately clamped to a high voltage E_1 or a low voltage E_0 , depending on

whether the flip-flop contains a binary 1 or binary 0 . The output voltages of each flip-flop are indicated by A_1 through A_7 , where A_1 is the voltage output of the flip-flop containing the most significant digit and A_7 is the voltage output of the flip-flop containing the least significant digit. A_1 through A_7 will take on voltages equal to E_1 or E_0 depending on whether the particular flip-flop contains a 1 or 0 . Figure 7 illustrates the resistance network which forms the output voltage E_{out} . It can be shown that each binary digit will receive a voltage weight proportional to its significance in the binary number in the network output, or:

$$E_{out} = \frac{1}{3}\left(A_1 + \frac{A_2}{2} + \frac{A_3}{4} + \frac{A_4}{8} + \frac{A_5}{16} + \frac{A_6}{32} + \frac{A_7}{64}\right) \qquad (27)$$

From studies made on this type of conversion system, it is known that a linearity of ±1 percent is readily obtainable in the conversion from binary number to voltage. It is desirable to calibrate the system so that E_{out} will be zero when the output register holds a binary number which represents zero error signal.

Acknowledgements

This work was performed at the J. B. Rea Company under contract with the Air Force Armament Laboratory at Wright Air Development Center. The electronics for this experimental model was designed by Mr. Arthur P. Untrauer.

Appendix

Definition of Symbols

C_i — weighting constants for the digital filter

$C_{L_{max}}$ = f(M) , maximum lift coefficient

d_s — weighted difference between lift and load

d_L = d_g or d_s , whichever is smaller

d_g — weighted difference between critical load and actual load

g — gravity acceleration

h_D — desired altitude

h_o — measured altitude

k_a = $k_1 s\, C_{L_{max}}$

k_b = $k_2 n_z$

k_c — the limiting value of d_L , the factor which permits the airframe to obtain maximum response for all flight conditions

k_d — limit factor on the desired control surface position

k_e = $K_{\dot{\theta}}$ = f(q) , pitch rate gain factor

k_f = $\frac{K_h}{K_\theta}$ = f(TAS) , altitude gain factor

$k_1 k_2$ — scaling factors

$k_3 = \frac{k_1}{k_2} g$

$k_4 = k_2 n_{z_{lim}}$

k_5 absolute margin of safety for critical stall and load conditions

k_6 the weighting term on $\bar{\Delta}$, used in predicting how close the aircraft may come to critical load or stall conditions

$k_7 = \frac{K_\theta}{K_{\dot{\theta}}}$, pitch attitude gain factor

k_8 k_9 the lower and upper bounds of Δ respectively, which decide when nonlinear control action will be utilized

k_{10} maximum positive control surface deflection

k_{11} maximum negative control surface deflection

k_{12} trim coefficient

k_{13} limit for Δ for determination of k_c

L lift force, qs C_L

M Mach number

m aircraft mass

$m_1 = k_3 m$

$n_z = \frac{L}{mg}$, load factor in g's

$n_{z_{lim}}$ maximum allowable structural load limit

q dynamic pressure

s wing area

TAS true airspeed

t_o time at which the present computation is being made

Δ desired control surface position before trim, nonlinear control action, load or stall limits have been included

$\bar{\Delta}$ limited value of Δ for determination of k_c

δ_D desired control surface position including the nonlinear control function but before load or stall limits have been considered

δ_d desired control surface position including trim before nonlinear control or load or stall limiting have been considered

δ_e control surface position

ε_i error signal to the control surface servo

θ_D desired pitch attitude

θ_o measured pitch attitude

$\dot{\theta}_D$ desired pitch rate

$\dot{\theta}_{DN}$ desired pitch rate with noise

$\dot{\theta}_d$ effective desired pitch rate

$\dot{\theta}_o$ measured pitch rate

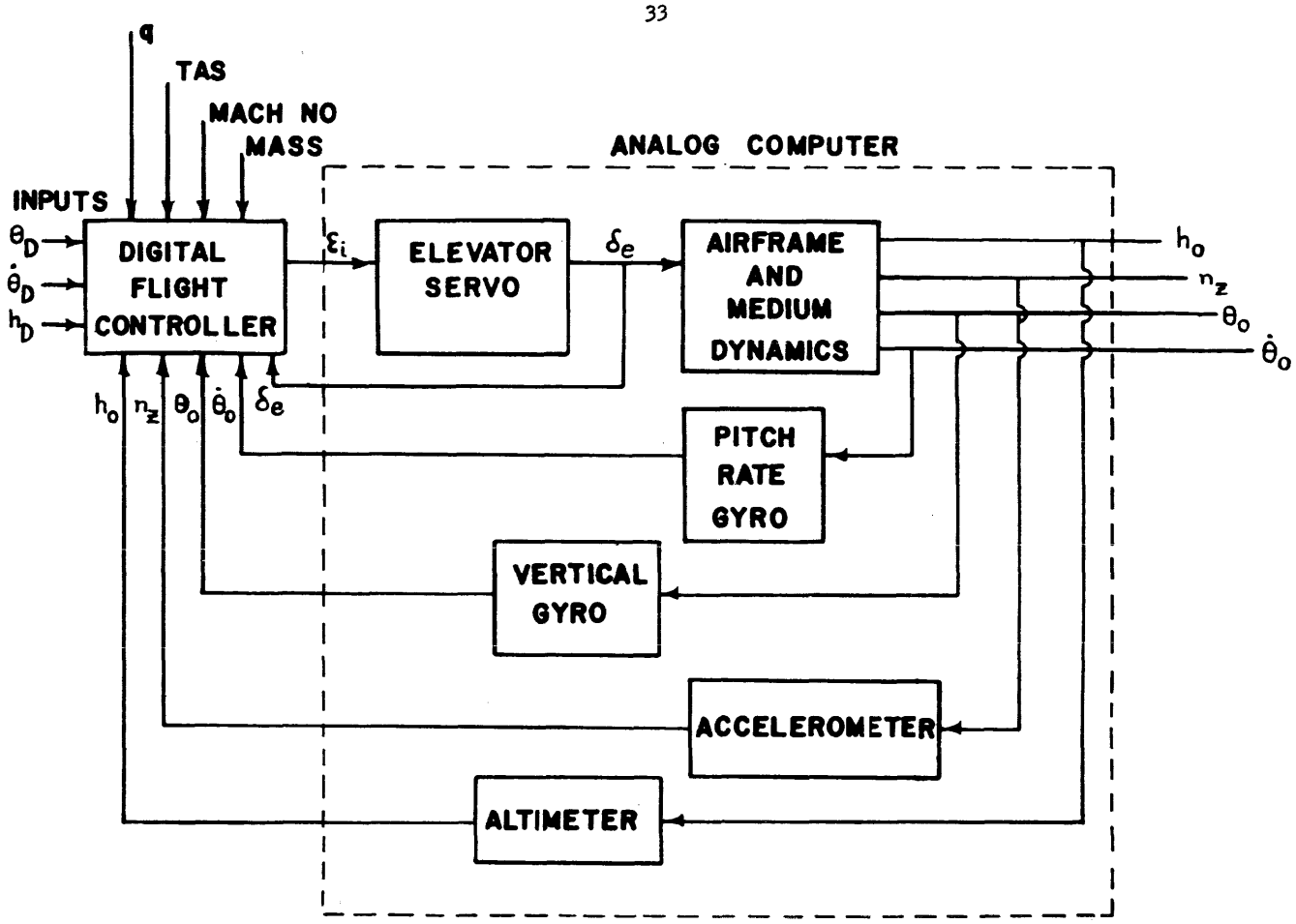

Fig. 1. Simulation mechanization for the digital flight controller

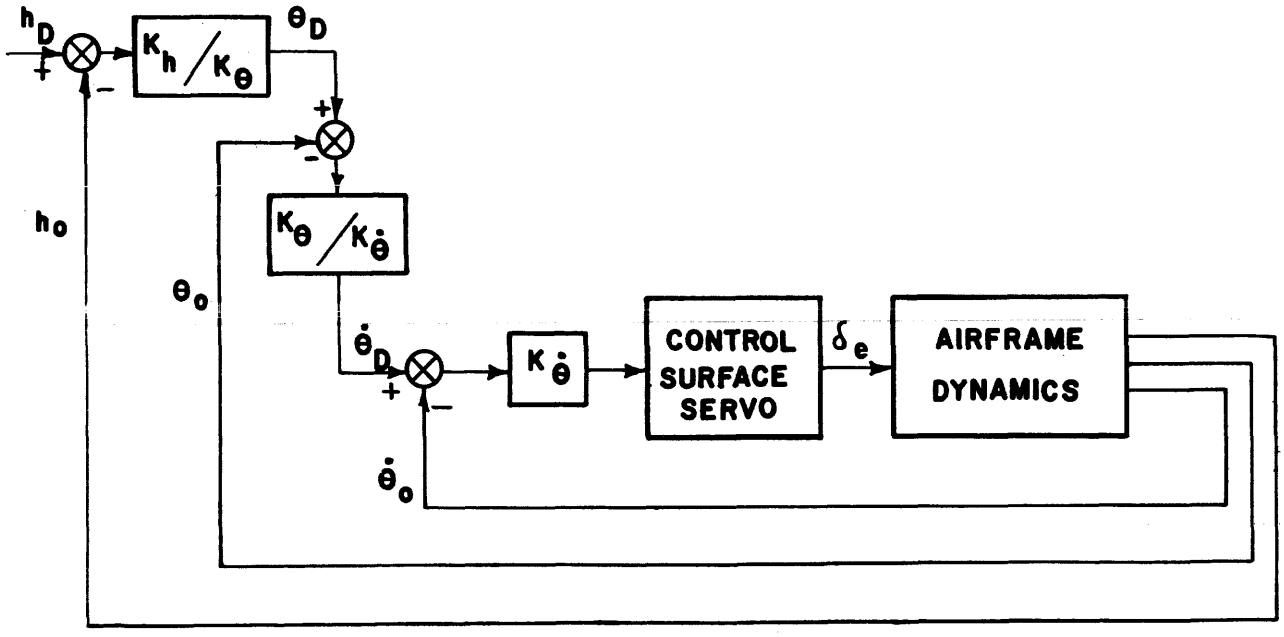

Fig. 2. Basic control system in block diagram form

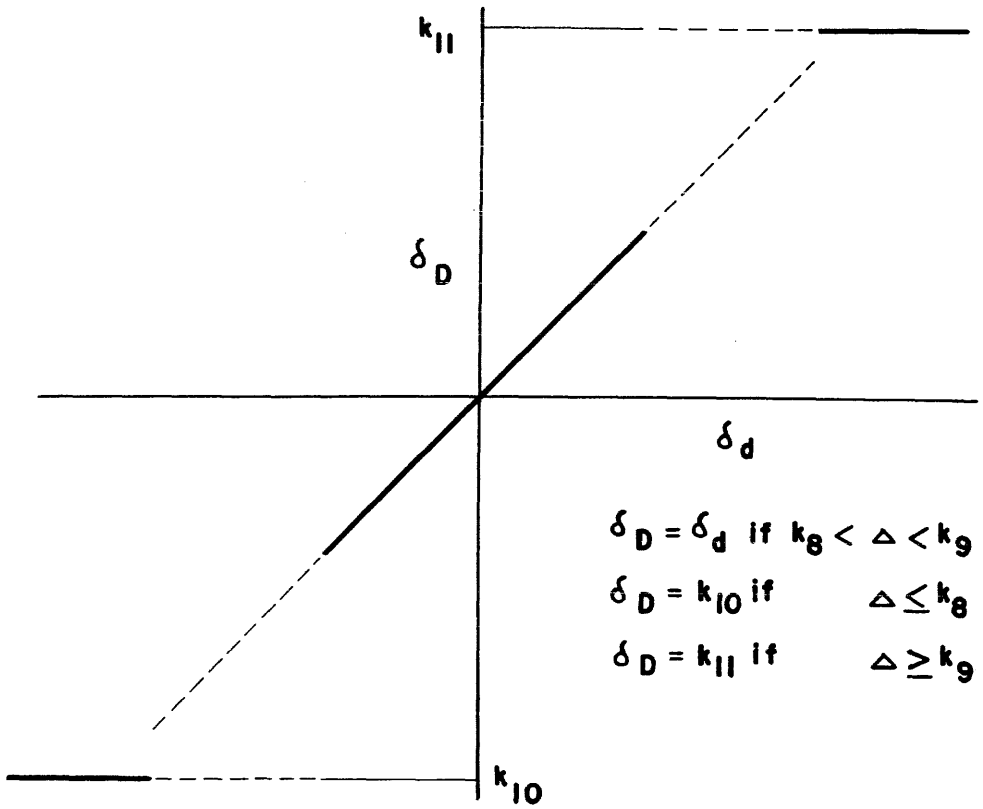

Fig. 3. Programmed nonlinear control

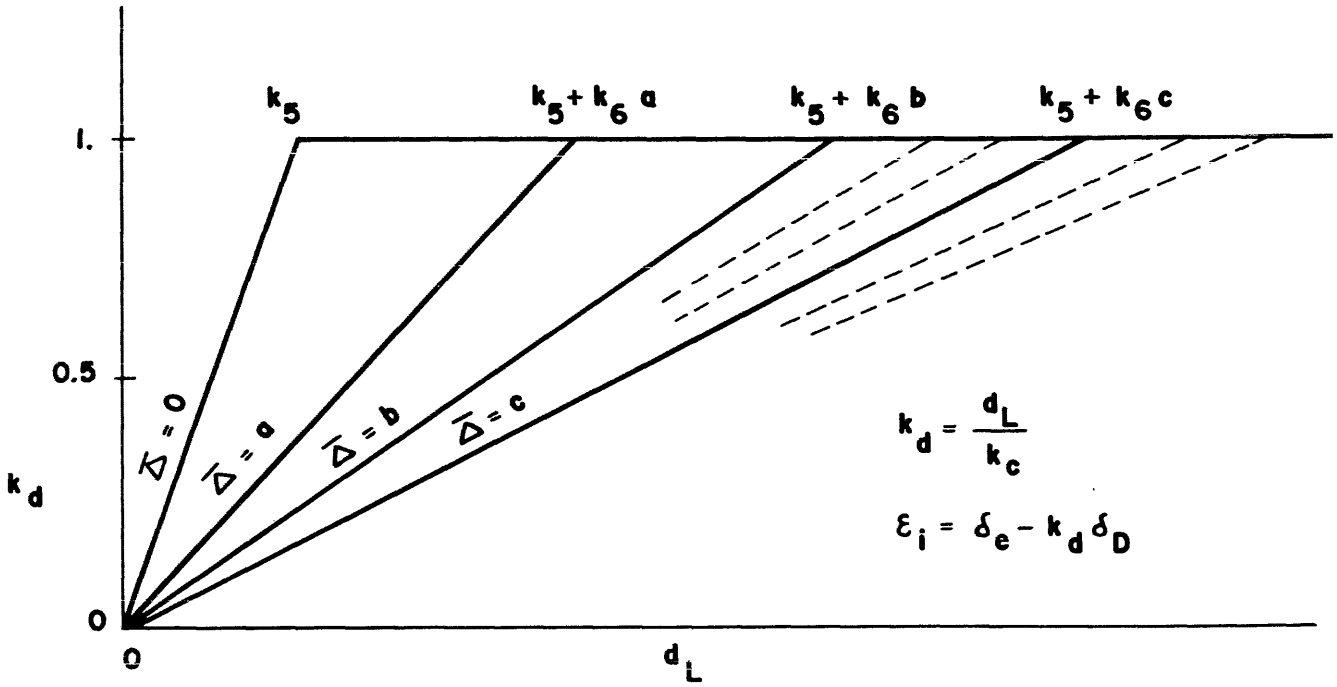

Fig. 4. Structural load and stall limiting factor

$$\Delta = k_e\left[k_7k_f(h_D - h_o)\ \textcircled{1} + k_7(\theta_D - \theta_o)\ \textcircled{2} + \dot{\theta}_D\ \textcircled{3} - \dot{\theta}_o\right] \quad (7)$$

$$\delta_d = \Delta + k_{12}\int_{t_{-\infty}}^{t_o}\Delta dt - k_7k_e\theta_o\ \textcircled{1} \quad (8)$$

$$\bar{\Delta} = \Delta \quad \text{if } \Delta < k_{13} \quad (14)$$

$$\bar{\Delta} = k_{13} \text{ if } \Delta \geq k_{13} \quad (15)$$

$$d_s = qk_a - k_bm_1 \quad (16)$$

$$d_g = k_4 - k_b \quad (17)$$

$$d_L = d_s \text{ or } d_g \text{ , whichever is smaller} \quad (18)$$

$$k_c = k_5 + k_6\bar{\Delta} \quad (19)$$

$$k_d = 1 \text{ if } \Delta<0 \text{ or if } k_b<0 \text{ or if } d_L>k_c \quad (20)$$

$$k_d = \frac{d_L}{k_c} \text{ unless } k_d = 1 \text{ per above} \quad (21)$$

$$\varepsilon_i = \delta_e - k_d\delta_D \quad (22)$$

$$\delta_D = \delta_d \text{ if } k_d \neq 1 \text{ or if } k_8<\Delta<k_9 \text{ and only if } k_{10}<\delta_d<k_{11} \quad (23)$$

$$\delta_D = k_{10} \text{ if } k_d = 1 \text{ and } k_8\geq\Delta \text{ or if } k_{10}\geq\delta_d \quad (24)$$

$$\delta_D = k_{11} \text{ if } k_d = 1 \text{ and } \Delta\geq k_9 \text{ or if } \delta_d\geq k_{11} \quad (25)$$

Figure 5. Control Equations

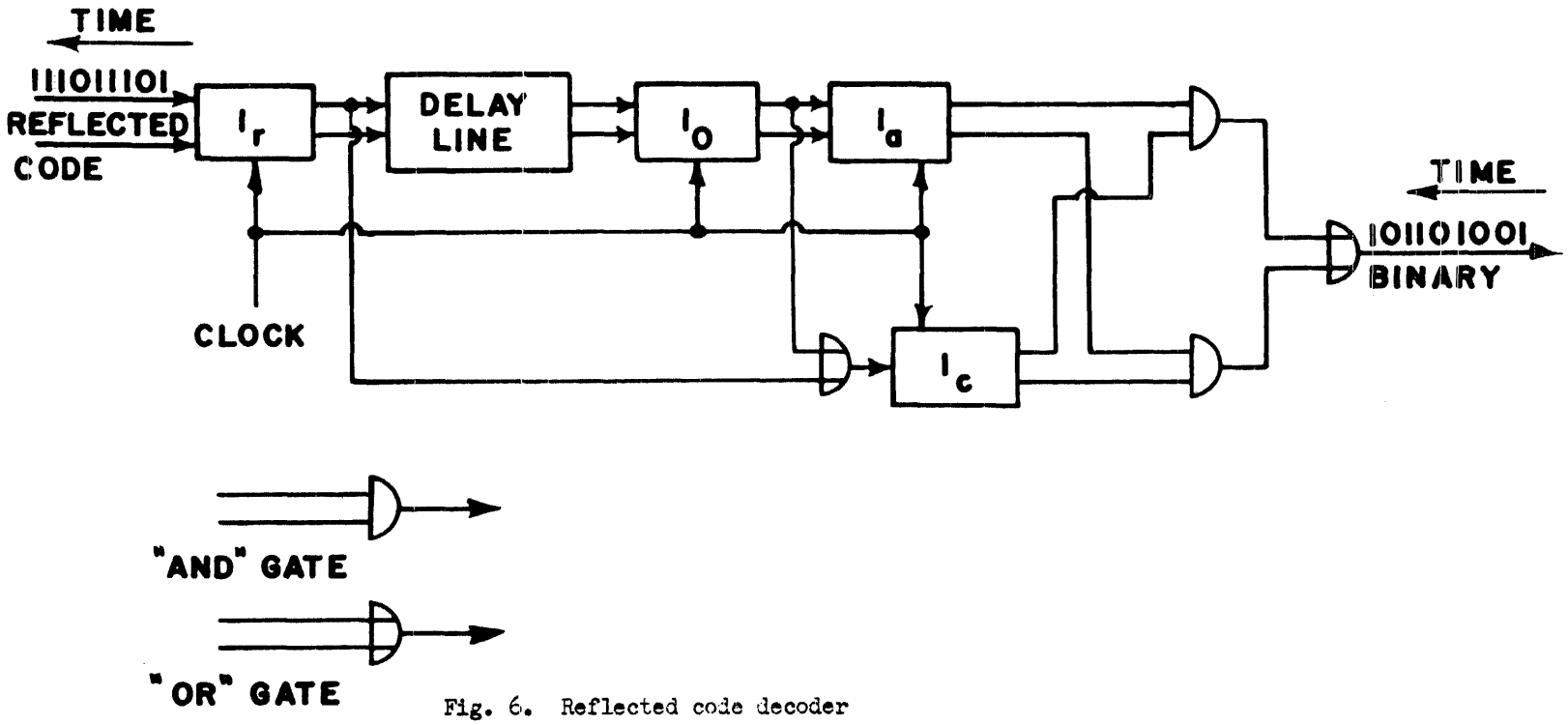

Fig. 6. Reflected code decoder

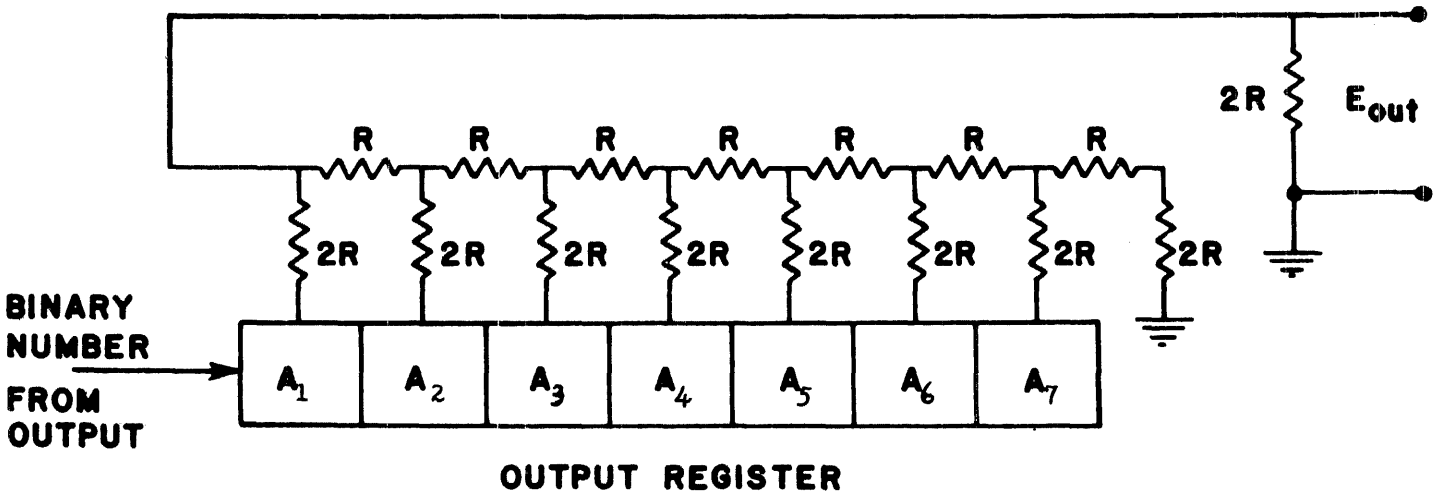

Fig. 7. Binary to d-c voltage converter

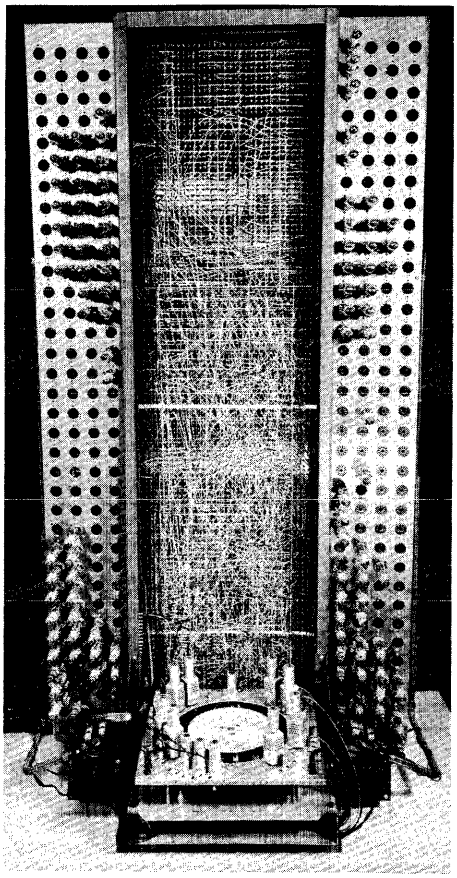

Fig. 8. Front view, the digital computer including the magnetic drum memory

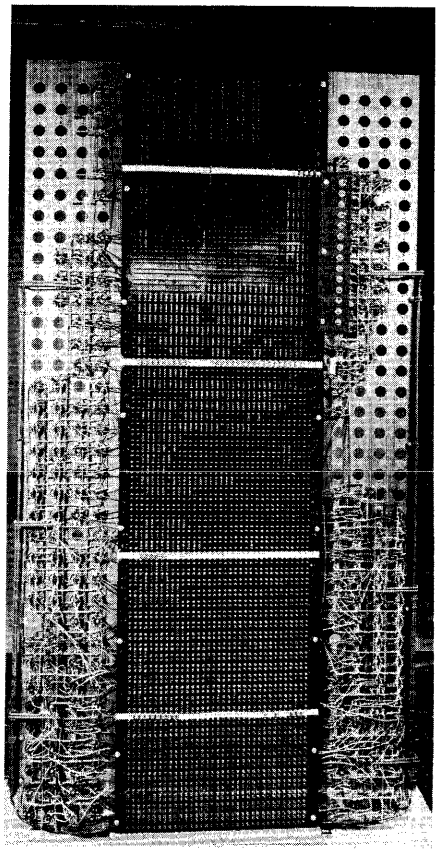

Fig. 9. Back view, the digital computer showing the diode matrices

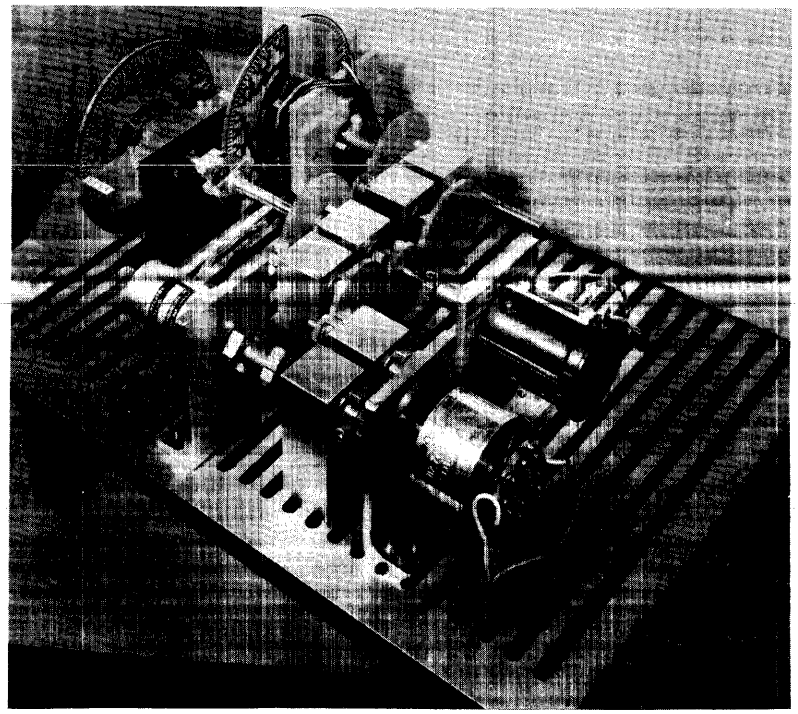

Fig. 10. The analog-to-digital servo converter

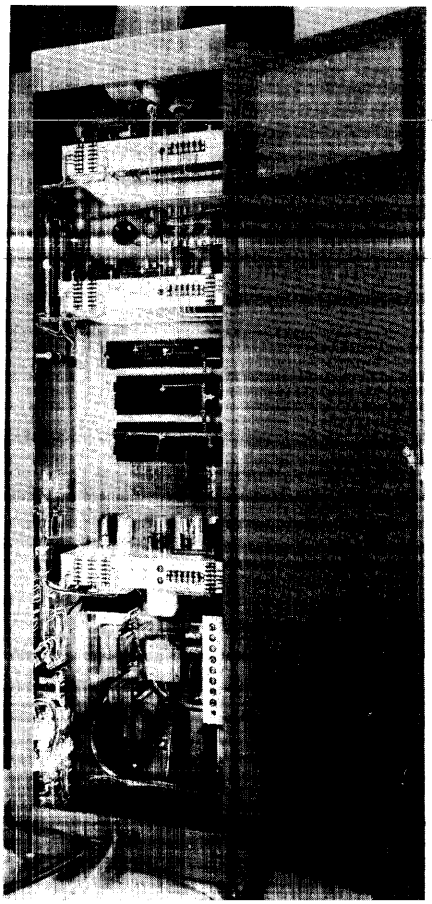

Fig. 11. The power supply