THE DESIGN OF A PLANAR ACTING MANIPULATOR AND ITS CONTROL

A THESIS

Presented to

The Faculty of the Division

of Graduate Studies

By

Lawrence Elliot Field

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in Mechanical Engineering

Georgia Institute of Technology

May, 1978

THE DESIGN OF A PLANAR ACTING MANIPULATOR

AND ITS CONTROL

Approved:

Wayne Book, Chairman

S. L. Dickerson

Don S. Harmer

John Mills

Date approved by Chairman: 5 June 78

ACKNOWLEDGMENTS

The author wishes to thank Dr. Wayne Book for his guidance and tolerance in the course of this project. Special thanks must also go to our electronic technician, Mr. Gene Clopton, who contributed many patient hours of instruction. An acknowledgment would not be complete without thanking Mrs. Sharon Butler who, with great speed, was able to decipher my handwriting and type this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

SUMMARY

A program of research is planned for the development of relationships between task parameters and design specifications for manipulators. The purpose of this thesis was to design a manipulator and its control for use in this research. The manipulator has two links with two rotary joints with a torque motor providing power at each joint. The manipulator moves in the horizontal plane and is capable of moving at velocities comparable to those of a human arm.

The manipulator is controlled by a Texas Instruments 990/4 microcomputer. The computer accepts velocity and displacement error information from sensors on the manipulator and a master and sends signals to the manipulator torque motors. The computer hardware and software were both designed for flexibility and speed of sampling. The sample rate is one kilohertz.

Flexibility is provided in the hardware design through the use of standardized modules for interface circuitry and a wire-wrap jumper board on the main chassis. This approach permits the experimenter to make changes in instrumentation or in signal converters without doing large rewiring jobs. Adaptation to different instrumentation has also been facilitated in the software through the use of programmable scale factors.

CHAPTER I

INTRODUCTION

The application of manipulator technology, most notably in the areas of automated production and human prosthesis, has expanded greatly in recent years. The sources of this expansion have been advances in artificial intelligence, instrumentation, computation hardware and control techniques.

Research in advanced control techniques as applied to manipulators has pointed out the lack of understanding of the manipulation task itself. The manipulation task is not well understood and what understanding exists is not expressed in terms which concisely specify the design and the feedback control hardware required for a specified task.

To meet this problem a program of experiments is planned which would relate task and manipulator specification to manipulator performance. The characteristics of the manipulator required for this research such as deadband, noise, saturation and compliance would need to be variable. Since the tasks used in the program would be simple peg-in-the-hole tasks, the manipulator would not need a large payload capacity and would need to act in a plane only. Operating speeds and positioning accuracies approaching

those of the human arm which generates the command signals are desirable for experimental purposes. The goal of this thesis is to design the arm and its servo control to meet the above criteria.

The design consists of a two link planar acting arm with a torque motor controlling each joint giving it two degrees of freedom in a plane. The arm is controlled by a Texas Instruments 990/4 microcomputer which, using a proportional control algorithm, accepts angular position and velocity information from the arm then outputs signals to the torque motors. The computer software can be varied to alter the arm's characteristics.

This thesis has two major divisions: system design and system test. Since the computer software and interface hardware designs are related, they can not be treated as completely separate subjects within the design section. For the system user's convenience, however, to the greatest extent possible, they are treated independently so that they may serve as a user's guide for program modification and equipment servicing.

CHAPTER II

ARM DESIGN

2.1. Arm Design Criteria

Before the design of the arm could be "blocked out" the specifications that it needed to meet had to be determined. To do this two questions had to be answered. First, what quantities need to be defined? Second, how are they to be defined? The first question is addressed by Book [1]. Briefly, the design is constrained by strength and stiffness considerations and these considerations do not necessarily involve direct tradeoffs. For instance, increasing the stiffness of a member also generally makes it stronger. Limitations can be discerned, however, by considering the effects of strength and stiffness on the types of motion that the arm undergoes.

Two types of motion can be defined, gross motion and fine motion. Gross motions are large changes in the configuration of the arm where large is defined with respect to the deflection of the members when the maximum torque is applied by the actuator. A design constraint can be derived by considering the tradeoff between gross and fine motion ability. If the links of the arm are designed to withstand stresses in their cross sections just exceeding the maximum

stresses produced by acceleration the arm could be very limber. The arm could have a high gross motion speed, but the links would go through large amplitude lower frequency oscillations. Increasing the cross-sectional inertia of the links to reduce the amplitude of the oscillations would increase the mass of the arm and, therefore, reduce its gross motion speed. The goal, then, is to define acceptable limits for fine and gross motion speeds and to relate these limits to structural variables.

Book [2] derives, using ω_c, the clamped joint natural frequency of the arm, and ω_s, the lowest undamped natural frequency of the servo control, a gross motion to fine motion frequency ratio as a specification along with the constraint that $\omega_s \leq \omega_c/2$ for a damping ratio of at least 0.7. For this particular design a slight variation of this approach will be used.

Assume that the maximum gross motion speed will enable the arm to sweep through an angle, θ, and back to the starting point again in some time, τ. For a pure inertia with torque limited actuators the payload must be an acceleration a_p applied in the manner shown in Figure 2-1. The following relations also follow:

$$V(\tau/4) = \frac{a_p \tau}{4}, \quad V(\tau/2) = 0, \quad d(\tau/2) = \theta L \qquad (2\text{-}1)$$

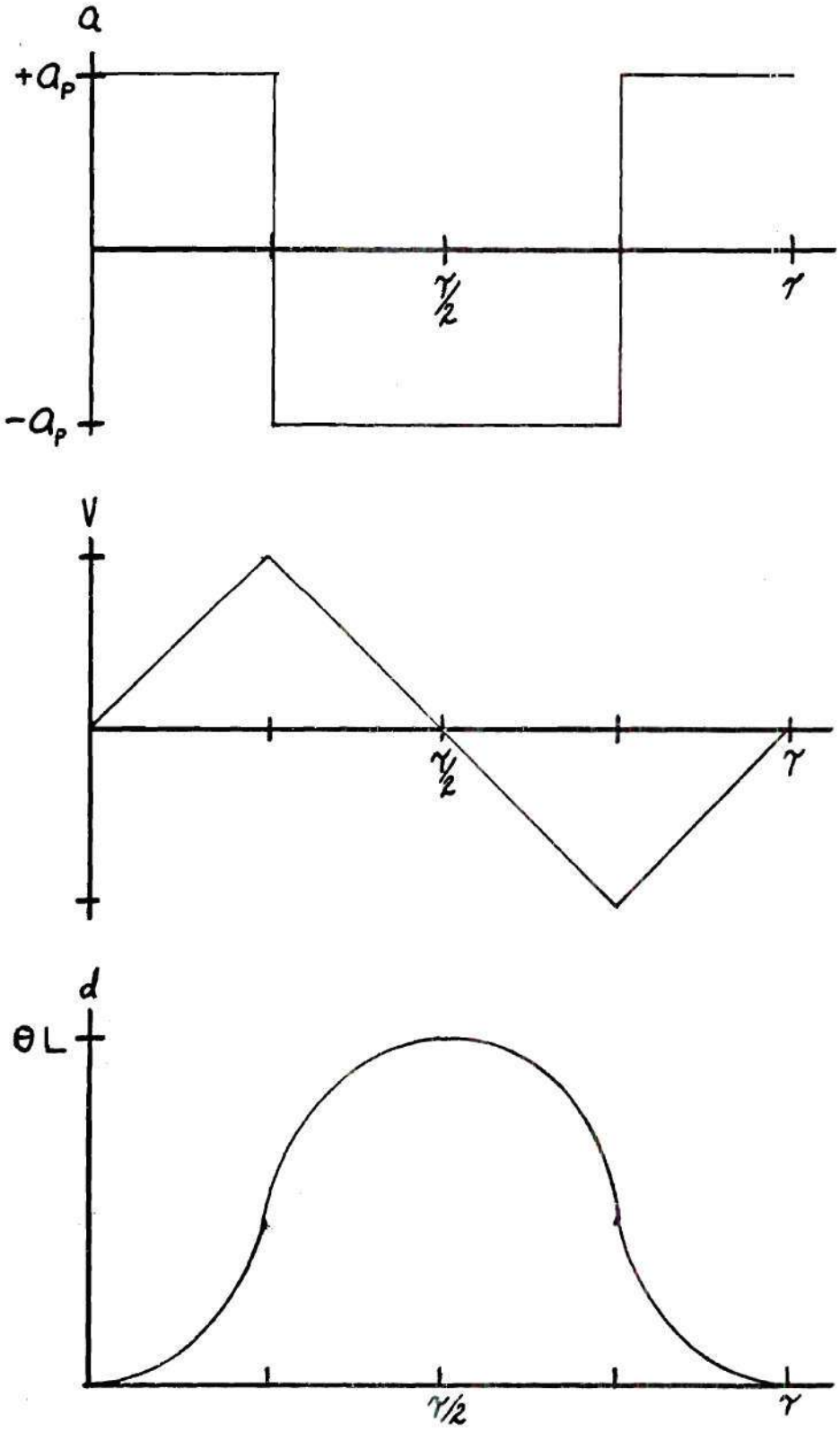


Figure 2-1a. Bang-Bang Gross Motion Speed

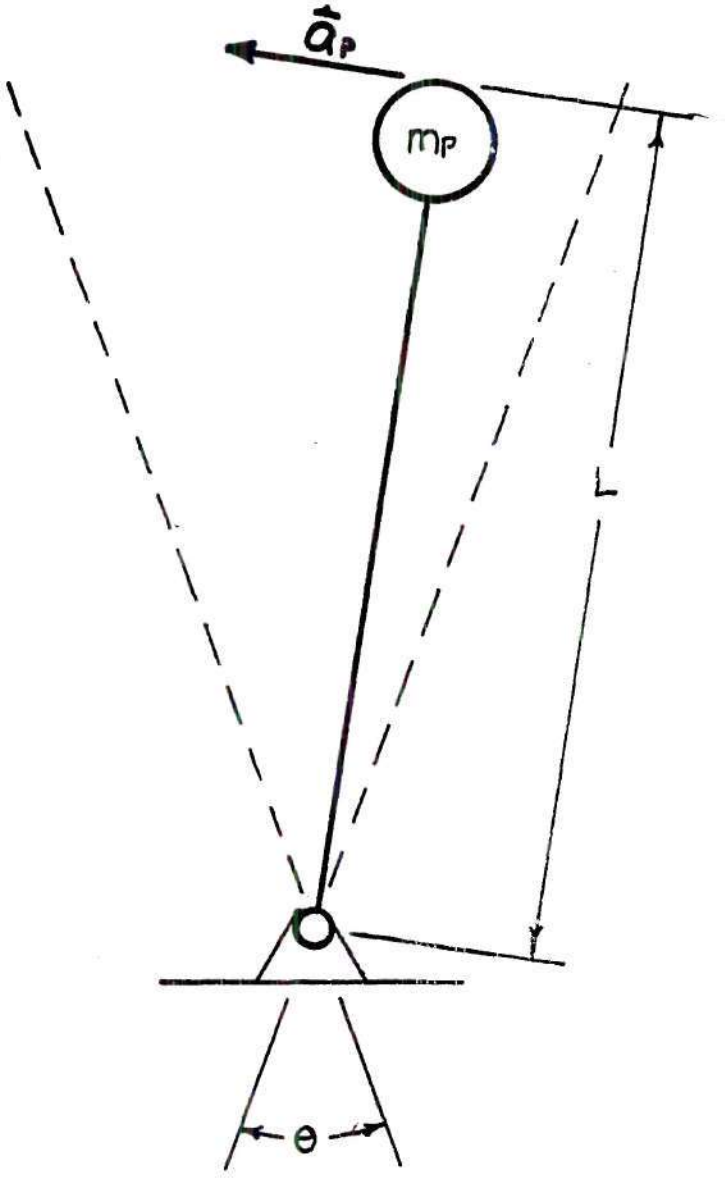


Figure 2-1b. Bang-Bang Gross Motion Single Link Diagram

where

L = arm length from joint to payload

V = arm velocity

d = arm displacement

Applying some elementary calculus, the following equation is derived:

$$d(\tau/2) = \int_0^{\tau/4} a_p t\,dt + \int_{\tau/4}^{\tau/2} [(\frac{a_p \tau}{4} - a_p(t-\tau/4)]dt,$$

resulting in,

$$d(\tau/2) = \theta L = \frac{a_p \tau^2}{16} . \qquad (2\text{-}2)$$

From elementary dynamics the following is known,

$$a_p = \alpha L \qquad (2\text{-}3)$$

$$T = I_L \alpha \qquad (2\text{-}4)$$

where

α = angular acceleration

T = torque supplied by the motor

I_L = the mass moment, or load, inertia of the arm

Equations (2-3) and (2-4) can be used to eliminate a_p in equation (2-2). Doing this and rearranging terms results in

$$\frac{16\theta_L}{\tau^2} = \frac{T_L}{I_L} ,$$

$$\frac{\theta}{\tau^2} = \frac{T}{16I_L}$$

and finally,

$$\frac{4\sqrt{\theta}}{\tau} = \sqrt{\frac{T}{I_L}} . \qquad (2\text{-}5)$$

Equation (2-5) is very useful. It relates a gross motion specification of the arm to the actuator's capability and the arm's mass and geometry.

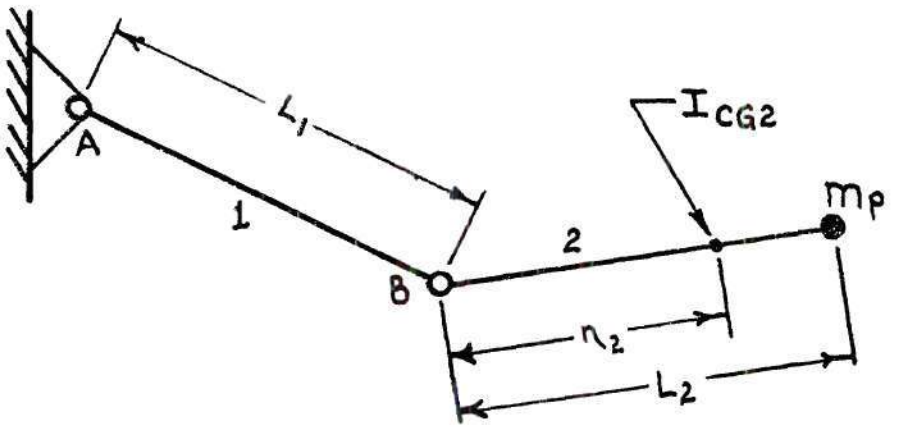
The clamped joint natural frequency can be found using a lumped parameter model. In this case a derivation done by Fertis [3] can be adapted directly. If the arm is modeled as two lumped masses connected by two massless springs as in Figure 2-2, then:

$$m_1\ddot{y}_1 + k_1y_1 - k_2(y_2-y_1) = 0 \qquad (2\text{-}6)$$

$$m_2\ddot{y}_2 + k_2(y_2-y_1) = 0 \qquad (2\text{-}7)$$

If the motions are harmonic then,

$$y_1 = Y_1 \sin\omega t \qquad (2\text{-}8)$$



I_{A1} = load inertia of link 1 at point A = .019 lb_f-ft-sec^2

I_{A2} = load inertia of link 2 at point A = .082 lb_f-ft-sec^2

I_{B2} = load inertia of link 2 at point B = .013 lb_f-ft-sec^2

T_1 = torque exerted by motor at A = 7.0 ft-lb_f

T_2 = torque exerted by motor at B = 2.0 ft-lb_f

L_1 = length of link 1 = 12.0 in

L_2 = length of link 2 = 12.0 in

n_2 = distance from B to center of mass of link 2 = 5.42 in

m_1 = mass of link 1 = 2.42 lb_m

m_2 = mass of link 2 = 1.88 lb_m

I_1 = minimum cross-section inertia of link 1 = 0.60 in^4

I_2 = minimum cross-section inertia of link 2 = 0.78 in^4

E = modulus of elasticity = 10.3×10^6 lb_f/in^2

k_1 = spring constant of link 1 = 10,709 lb_f/in

k_2 = spring constant of link 2 = 11,108 lb_f/in

Figure 2-2a. Manipulator Schematic Diagram

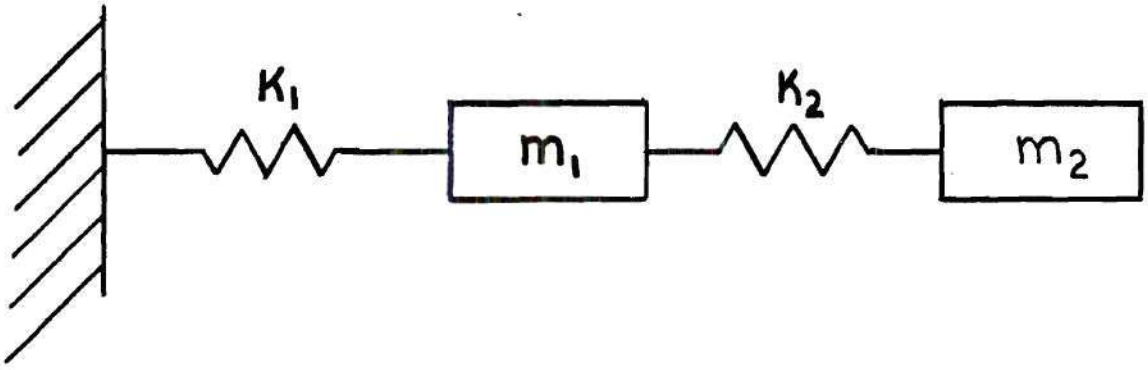


Figure 2-2b. Two Link Lumped Parameter Model

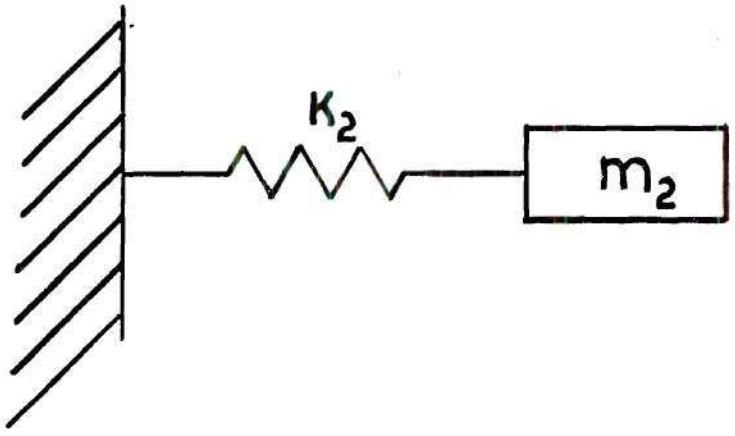


Figure 2-2c. Single Link Lumped Parameter Model

$$y_2 = Y_2 \sin\omega t \qquad (2\text{-}9)$$

Substituting (2-8) and (2-9) into (2-6) and (2-7) yields

$$(m_1\omega^2 - k_1 - k_2)Y_1 + k_2 Y_2 = 0 \qquad (2\text{-}10)$$

$$k_2 Y_1 + (m_2\omega^2 - k_2)Y_2 = 0 \qquad (2\text{-}11)$$

For a non-trivial solution,

$$\begin{vmatrix} (m_1\omega^2 - k_1 - k_2) & k_2 \\ k_2 & (m_2\omega^2 - k_2) \end{vmatrix} = 0$$

The natural frequencies, then, are:

$$\omega_{c1,2}^2 = \frac{1}{2}\left(\frac{k_1+k_2}{m_1} + \frac{k_2}{m_2}\right) \pm \frac{1}{2}\left[\left(\frac{k_1+k_2}{m_1} + \frac{k_2}{m_2}\right)^2 - \frac{4k_1k_2}{m_1m_2}\right]^{1/2} \qquad (2\text{-}12)$$

As mentioned earlier, for adequate damping, $\omega_c \geq 2\omega_s$.

A relation for the stresses in a cross-section has not been mentioned because the equation would be particular to the design. Simply stated, the maximum shear and moment stresses generated by motion must not exceed the fatigue limits of the material.

2-2. Determination of Performance Specifications

Some useful criteria being established, the next task is to specify values to achieve the desired performance which, as stated, is to approach that of a human arm. To do this, some simple planar arm motion experiments were performed on several people. Each subject, while holding an accelerometer in his hand, moved his arm back and forth between two points as fast as he could. Each person did this in a locked elbow test and in a forearm only test. The results are tabulated in Table 2-1.

Tests 1 and 3 will result in the more severe gross motion specification while results 2 and 4 may be used to provide minimum ω_c limits.

Recalling equation (2-5),

$$\frac{4\sqrt{\theta}}{\tau} = \sqrt{\frac{T}{I_L}}$$

the gross motion requirement for the shoulder motor is found for the extreme case of the fully extended arm.

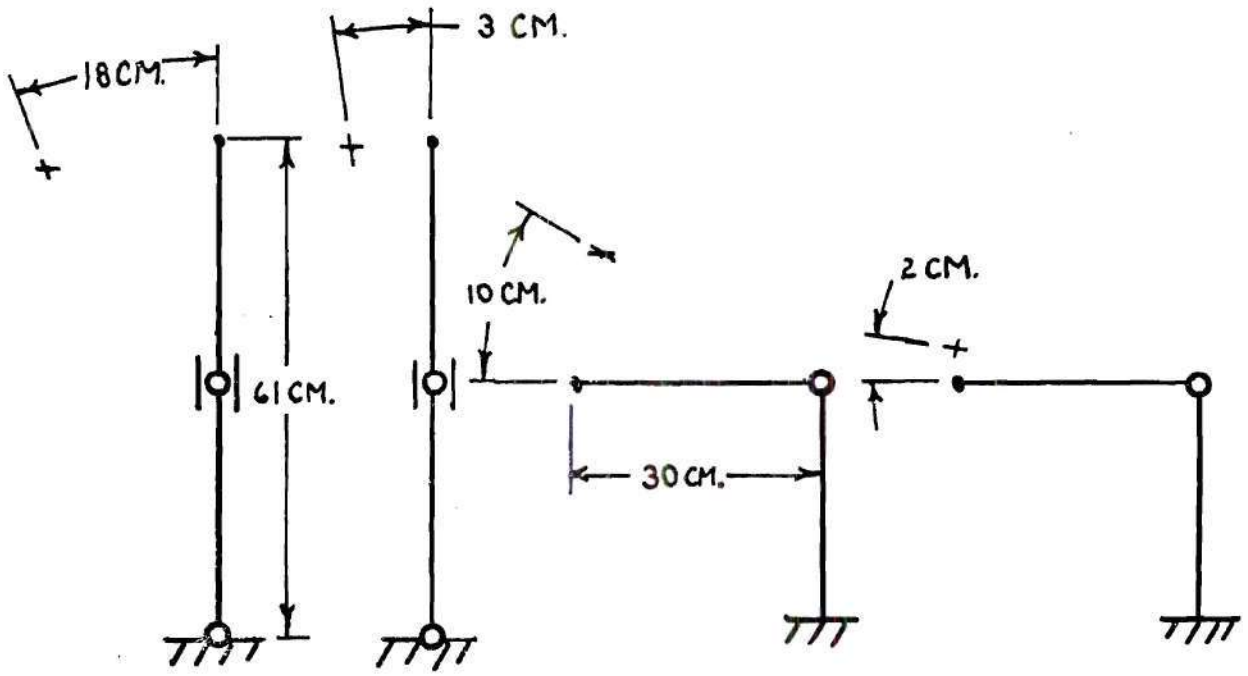
From test 1,

$$\theta = \frac{18}{61} = .295 \text{ rad}$$

$$\tau = \frac{2\pi}{17.4} = .361 \text{ sec}$$

Table 2-1. Human Arm Fine Motion/Gross Motion Frequencies

	Frequencies of Oscillation (rad/sec)			
	Test 1	Test 2	Test 3	Test 4
	17.4	26.7	29.2	45.3
	14.1	22.1	20.8	31.9
	16.7	33.2	28.6	33.0
	13.3	26.8	18.9	34.2
	12.6	32.5	26.8	36.3
	17.3	35.9	31.0	41.5
Max	17.4	35.9	31.0	45.3



Test 1	Test 2	Test 3	Test 4
Locked Elbow	Locked Elbow	Forearm Only	Forearm Only

$$\frac{4\sqrt{\theta}}{\tau} = \frac{4\sqrt{.295}}{.361} = 6.019/\text{sec} = \sqrt{\frac{T_s}{I_{Ls}}} \qquad (2\text{-}13)$$

Equation (2-13) defines the torque requirement of the shoulder motor to satisfy the extreme loading case of the arm.

As an additional gross motion constraint the forearm can be treated as a single link single joint arm in test 3.

$$\theta = \frac{10}{30} = .333 \text{ rad}$$

$$\tau = \frac{2\pi}{31.0} = .203 \text{ sec}$$

$$\frac{4\sqrt{.333}}{.203} = 11.388/\text{sec} = \sqrt{\frac{T_e}{I_{Le}}} \qquad (2\text{-}14)$$

Equation (2-14) defines a torque requirement for the elbow motor to meet the forearm gross motion performance specification.

To adequately track a triangular wave of fundamental frequency, f, the servo control should have a break frequency of 10f. Therefore, ω_c should equal 20f. Using the results of tests 2 and 4, we can conclude that the lowest natural frequency of the extended arm should be greater than 720 rad/sec and that the natural frequency of the forearm model should be greater than 920 rad/sec.

2-3. General Arm Configuration

With the help of the previously discussed constraints a two link arm was designed to meet the demands. The links are I-beams with slots in their webs to reduce mass. To further reduce the loading on the shoulder motor, the elbow motor was mounted at the shoulder and an aircraft cable and pully arrangement used to drive the elbow. A chain and sprocket drive was rejected because of the noise disturbance characteristic of chains in controls and the large mass of the assembly compared to the aircraft cable assembly. The aircraft cable is attached to brackets by threaded fittings allowing tension to be maintained by a simple adjustment. An air bearing is mounted on each link to offset gravitational force. Sketches of the links are included in Appendix II.

A 7 foot-pound torque motor is used for the shoulder and a 2 foot-pound torque motor for the elbow. Each motor is housed in an aluminum housing to protect it from ferrous material. Bearings used are ABEC-5 class to meet the tolerance requirements for the rotor/stator fit.

2-4. Design for Stiffness

As previously mentioned a lumped parameter model of the arm was used to find natural frequencies. The holes in the links give them a variable cross-sectional area moment of inertia. Rather than go through the very complex process of deriving an equivalent constant cross-section model for

the arm, the average inertia was found using the average of the minimum and maximum cross section inertias using the equation

$$I_{AVE} = (N_{MX}(I_{MX}) + N_{MN}(I_{MN})-/(N_{MX}+N_{MN}) \qquad (2\text{-}15)$$

where

N_{MX} = the number of continuous mass sections

N_{MN} = the number of slots

I_{MX} = the moment of inertia in a continuous mass section

I_{MN} = the minimum moment of inertia in the slot

This was considered an acceptable simplification for estimating purposes. In the following discussion cross-sectional inertias used are averages. These and other symbols used are defined in Figure 2-2.

The first step in evaluating the design is to determine its payload capacity under the worst case conditions. This is done by considering equation (2-13) repeated here

$$6.019/\text{sec} = \sqrt{\frac{T_1}{I_A}} \qquad (2\text{-}13)$$

Now,

$$I_A = I_{A1} + I_{A2} + \frac{m_p}{32.2}(L_1+L_2)^2 \qquad (2\text{-}16)$$

Substituting equation (2-16) into equation (2-13) and rearranging results in

$$m_p = 32.2\ [T_1-(6.019)^2(I_{A1}+I_{A2})]/[(6.019)^2(L_1+L_2)^2] \qquad (2\text{-}17)$$

Using the values from Figure 2-2, we find

$$m_p = .75\ lb_m$$

By examining equation (2-17) one can see that the payload capacity can be increased by lowering the load inertias of the links or by using a torque motor with a higher output. For the purposes of this design the above value will be adequate.

The lumped parameter model shown in Figure 2-2b was used to estimate the natural frequencies of the arm. To find the spring constants, each link was treated as a cantilevered beam. The deflection, y_L, at the end of a cantilever beam is

$$y_L = \frac{PL^3}{3EI}$$

where

y = endpoint deflection--in

P = load--lb_f

E = modulus of elasticity--lb_f/in^2

I = area moment of inertia--in^4

The spring constant for the beam is found by finding the load that will produce a one inch deflection at the endpoint

$$1 \text{ in} = \frac{P'L^3}{3EI}$$

or

$$k = \frac{3EI}{L^3} \qquad (2\text{-}18)$$

Applying equation (2-18) to link 1,

$$k_1 = \frac{3EI_1}{L_1^3} = 10{,}709 \text{ } lb_f/in$$

and to link 2,

$$k_2 = \frac{3EI_2}{L_2^3} = 11{,}108 \text{ } lb_f/in$$

Recalling equation (2-12),

$$\omega_{c1,2}^2 = \frac{1}{2}\left(\frac{k_1+k_2}{m_1} + \frac{k_2}{m_2}\right) \pm \frac{1}{2}\left[\left(\frac{k_1+k_2}{m_1} + \frac{k_2}{m_2}\right)^2 - \frac{4k_1k_2}{m_1m_2}\right]^{1/2} \qquad (2\text{-}12)$$

and using the values in Figure 2-2, the natural frequencies are found to be

$$\omega_{c1,2} = 882 \text{ \& } 2225 \text{ rad/sec}$$

The lower value is of interest and compares favorably with the previously defined required minimum of 720 rad/sec.

The single mass lumped parameter model of Figure 2-2c was used for the forearm model estimate. From elementary dynamics,

$$\omega_c = \sqrt{\frac{k_2}{m_2}} = 1506 \text{ rad/sec}$$

which exceeds the 920 rad/sec requirement.

2-5. Design for Strength

To verify that the arm cross-section is adequate for all loadings the worst case of a fully extended arm with motors applying maximum torque is assumed. Looking at a cross-section near the base where a portion of the beam web has been removed, we recall from the strength of materials

$$\sigma_{max} = \frac{CT_1}{I} \tag{2-19}$$

and

$$\tau = \frac{T_1}{XA} \tag{2-20}$$

where

σ_{max} = maximum axial stress produced by bending

C = distance from neutral axis to extreme fiber

I = area moment of inertia of cross section

τ = shear stress

X = distance from centerline of joint to cross section

A = cross section area

Equations (2-19) and (2-20) will yield values that are high, but this is acceptable in the name of conservativeness. From design measurements,

$$C = 1.7500 \text{ in}$$

$$I = .7827 \text{ in}^4$$

$$X = 3.875 \text{ in}$$

$$T_1 = 7.0 \text{ lb}_f\text{-ft} = 84.0 \text{ lb}_f\text{-in}$$

$$A = .2812 \text{ in}^2$$

$$\sigma_{max} = \frac{(84.0)(1.7500)}{.7827} = 187.8 \text{ lb}_f/\text{in}^2$$

$$\tau = \frac{84.0}{(3.875)(.2812)} = 77.1 \text{ lb}_f/\text{in}^2$$

Clearly, these values are well within the capabilities of aluminum.

The design has been shown to meet the criteria developed earlier in the chapter. Interestingly, the stiffness criteria ultimately proved more severe than the strength criteria.

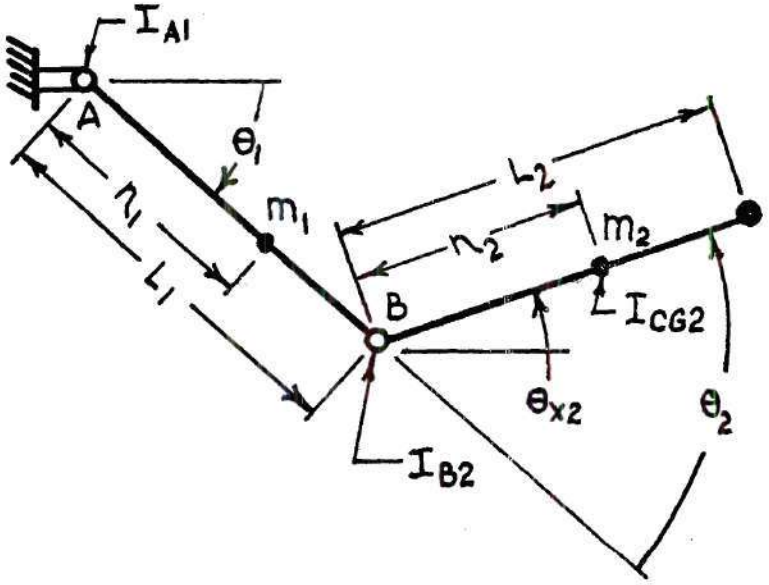
CHAPTER III

CONTROL ALGORITHM

The first step in defining the control equations is the derivation of the equations of motion. This is done in this case using the Euler-Lagrange equation with generalized coordinates. For a coordinate to be a generalized coordinate it must be independent of all other coordinates. One generalized coordinate must be defined for each degree of freedom. Since the arm has two rotary joints it has two degrees of freedom and, therefore, two generalized coordinates, θ_1 and θ_{x2}, are defined for the shoulder and elbow respectively. These are used to find expressions for the kinetic energy and potential energy of the system which are in turn used to derive the Lagrangian. A process of taking derivatives and combining terms then follows resulting, finally, in the equations of motion for the system.

3-1. Arm Equations of Motion

The equations of motion for the arm were derived using the Euler-Lagrange equation for a non-conservative system since the torque motors would be supplying energy across the system boundary. Figure 3-1 is a schematic of the arm showing the symbols used in this development including the generalized coordinates.



I_{A1} = load inertia of link 1 about point A

I_{B2} = load inertia of link 2 about point B

I_{CG2}= load inertia of link 2 about its center of mass

L_1 = length of link 1

L_2 = length of link 2

n_1 = distance from A to center of mass of link 1

n_2 = distance from B to center of mass of link 2

θ_1 = generalized coordinate for link 1

θ_{x2} =generalized coordinate for link 2

θ_2 = angle between link 2 and link 1

m_1 = mass of link 1

m_2 = mass of link 2

Figure 3-1. Schematic Diagram for Equations of Motion

Briefly, the Euler-Lagrange equation is:

$$\frac{d}{dt}\left(\frac{dL}{d\dot{\theta}_i}\right) - \frac{dL}{d\theta_i} = Q_i \qquad (3\text{-}1)$$

where, L = KE - PE, and,

KE = kinetic energy of system

PE = potential energy of the system

θ_i = i^{th} angular displacement in generalized coordinates

$\dot{\theta}_i$ = i^{th} angular velocity in generalized coordinates

Q_i = i^{th} generalized torque across the system boundary

Referring to Figure 3-1, the kinetic energy is found to be

$$KE = \frac{1}{2} I_{A1}\dot{\theta}_1^2 + \frac{1}{2} m_2 V_2^2 + \frac{1}{2} I_{2CG}\dot{\theta}_{x2}^2 \qquad (3\text{-}2)$$

The first goal is to replace the V_2^2 term with its equivalent expression in generalized coordinates. From trigonometry:

$$x_2 = L_1\cos(\theta_1) + n_2\cos(\theta_{x2}) \qquad (3\text{-}3a)$$

$$y_2 = L_1\sin(\theta_1) + n_2\sin(\theta_{x2}) \qquad (3\text{-}3b)$$

$$\frac{dx_2}{dt} = -\dot{\theta}_1 L_1\sin(\theta_1) - \dot{\theta}_{x2} n_2\cos(\theta_{x2}) \qquad (3\text{-}3c)$$

$$\frac{dy_2}{dt} = \dot{\theta}_1 L_1\cos(\theta_1) + \dot{\theta}_{x2} n_2\cos(\theta_{x2}) \qquad (3\text{-}3d)$$

$$V_2^2 = [\frac{dx_2}{dt}]^2 + [\frac{dy_2}{dt}]^2 \qquad (3\text{-}3e)$$

Using the relations (3-3) we derive,

$$V_2^2 = \dot{\theta}_1^2 L_1^2 + \dot{\theta}_{x2}^2 n_2^2 + 2\dot{\theta}_1 \dot{\theta}_{X2} L_1 n_2 \cos(\theta_1 - \theta_{x2}) \qquad (3\text{-}4)$$

Substituting into (3-2), the final expression for kinetic energy is

$$KE = \frac{1}{2} [I_{A1}\dot{\theta}_1^2 + m_2(\dot{\theta}_1^2 L_1^2 + \dot{\theta}_{x2}^2 n_2^2 + 2\dot{\theta}_1 \dot{\theta}_{x2} L_1 n_2 \cos(\theta_1 - \theta_{x2})) + I_{2CG}\dot{\theta}_{x2}^2] \qquad (3\text{-}5)$$

Referring to Figure 3-1 again to determine the potential energy relation,

$$PE_1 = m_1 g n_1 \sin\theta_1$$

$$PE_2 = m_2 g(L_1 \sin\theta_1 + n_2 \sin\theta_{x2})$$

$$PE = g(m_1 n_1 \sin\theta_1 + m_2 L_1 \sin\theta_1 + m_2 n_2 \sin\theta_{x2}) \qquad (3\text{-}6)$$

Substituting (3-5) and (3-6) into the equation for L, we have

$$L = \frac{1}{2}\left[I_{A1}\dot{\theta}_1^2+m_2(\dot{\theta}_1^2L_1^2+\dot{\theta}_{x2}^2n_2^2+2\dot{\theta}_1\dot{\theta}_{x2}L_1n_2\cos(\theta_1-\theta_{x2}))+I_{2CG}\dot{\theta}_{x2}^2\right]$$

$$- g[(m_1n_1+m_2L_1)\sin\theta_1+m_2n_2\sin\theta_{x2}] \qquad (3\text{-}7)$$

The derivatives for forming (3-1) are then taken

$$\frac{dL}{d\theta_1} = -\dot{\theta}_1\dot{\theta}_{x2}m_2L_1n_2\sin(\theta_1-\theta_{x2})-g(m_1n_2+m_2L_1)\cos(\theta_1) \qquad (3\text{-}8a)$$

$$\frac{dL}{d\dot{\theta}_1} = I_{A1}\dot{\theta}_1 + m_2L_1^2\dot{\theta}_1+m_2\dot{\theta}_{x2}L_1n_2\cos(\theta_1-\theta_{x2})$$

$$\frac{d}{dt}\left(\frac{dL}{d\dot{\theta}_1}\right) = I_{A1}\ddot{\theta}_1+m_2L_1^2\ddot{\theta}_1+\ddot{\theta}_{x2}L_1n_2\cos(\theta_1-\theta_{x2})-\dot{\theta}_{x2}(\dot{\theta}_1-\dot{\theta}_{x2})L_1n_2m_2\sin$$

$$(\theta_1-\theta_{X2}) \qquad (3\text{-}8b)$$

$$\frac{dL}{d\theta_{x2}} = \dot{\theta}_1\dot{\theta}_{x2}m_2L_1n_2\sin(\theta_1-\theta_{x2})-gm_2n_2\cos(\theta_{x2}) \qquad (3\text{-}9a)$$

$$\frac{dL}{d\dot{\theta}_{x2}} = m_2n_2^2\dot{\theta}_{x2}+\dot{\theta}_1m_2L_1n_2\cos(\theta_1-\theta_{x2}) + I_{2CG}\ddot{\theta}_{x2}$$

$$\frac{d}{dt}\left(\frac{dL}{d\dot{\theta}_{x2}}\right) = m_2n_2^2\ddot{\theta}_{x2}+\ddot{\theta}_1m_2L_1n_2\cos(\theta_1-\theta_{x2})-\dot{\theta}_1(\dot{\theta}_1-\dot{\theta}_{x2})m_2L_1n_2\sin$$

$$(\theta_1-\theta_{x2})+I_{2CG}\ddot{\theta}_{x2} \qquad (3\text{-}9b)$$

The equations (3-8) and (3-9) are each substituted into (3-1) to find an Euler-Lagrange equation for each generalized

coordinate defined, are the two equations

$$(I_{A1}+m_2L_1^2)\ddot{\theta}_1+\ddot{\theta}_{x2}L_1n_2\cos(\theta_1-\theta_{x2})+\dot{\theta}_{x2}^2L_1n_2m_2\sin(\theta_1-\theta_{x2})+$$

$$g(m_1n_1+m_2L_1)\cos(\theta_1) = Q_1 \qquad (3\text{-}10)$$

and,

$$(m_2n_2^2+I_{2CG})\ddot{\theta}_{x2}+\ddot{\theta}_1m_2L_1n_2\cos(\theta_1-\theta_{x2})-\dot{\theta}_1^2m_2L_1n_2\sin(\theta_1-\theta_{x2})+$$

$$gm_2n_2\cos(\theta_{x2}) = Q_2 \qquad (3\text{-}11)$$

Q_1 and Q_2 are defined by the following relation,

$$Q_k = \sum_{j=1}^{p} T_j \frac{dr_j}{dq_k} \qquad (3\text{-}12)$$

where

T_j = the j^{th} torque input

r_j = displacement due to j^{th} torque

q_k = k^{th} generalized coordinate

In this case

$q_1 = \theta_1$

$q_2 = \theta_{x2} = \theta_1+\theta_2$

$q_2-q_1 = \theta_2$

Finding Q_1 and Q_2,

$$Q_1 = T_1 \frac{dq_1}{dq_1} + T_2 \frac{d(q_2-q_1)}{dq_1} = T_1-T_2 \qquad (3\text{-}13)$$

$$Q_2 = T_1 \frac{dq_1}{dq_2} + T_2 \frac{d(q_2-q_1)}{dq_2} = T_2 \qquad (3\text{-}14)$$

Initially, equations (3-10) and (3-11) can be simplified by ignoring the gravity terms since the arm will operate in the horizontal plane. One should also note that

$$I_{B2} = I_{2CG} + m_2 n_2^2$$

With these simplifications the equations of motion are as follows,

$$(I_{A1}+m_2L_1^2)\ddot{\theta}_1+\ddot{\theta}_{x2}L_1n_2\cos(\theta_1-\theta_{x2})+\dot{\theta}_{x2}^2L_1n_2m_2\sin(\theta_1-\theta_{x2}) = T_1-T_2 \qquad (3\text{-}15)$$

$$I_{B2}\ddot{\theta}_{x2}+\ddot{\theta}_1m_2L_1n_2\cos(\theta_1-\theta_{x2})-\dot{\theta}_1^2m_2L_1n_2\sin(\theta_1-\theta_{x2}) = T_2 \qquad (3\text{-}16)$$

3-2. Linearization of the Equations of Motion and Quazilinear Control Equations

The arm is a non-linear system, but linear control equations are desirable for minimizing computing time. This section describes the manipulation of the equations of motion and linearizing approximations made to them to obtain quazilinear equations. Before beginning this process some

new terms must be defined. As can be seen in Figure 3-1, θ_2 is the angle between the first and second link. This coordinate is used because it would be the angle "sensed" by a potentiometer mounted on the arm for measuring angle. Two other coordinates, θ_{10} and θ_{20}, come from a "master" and are references for the control, that is, the control will always attempt to make

$$\theta_1 = \theta_{10}$$

and

$$\theta_2 = \theta_{20}.$$

These two terms will also be used in the process of linearization. By using a Taylor series expansion of the trigonometric terms in equations (3-15) and (3-16) around θ_{20} equations can be derived that would be linear in the region of a θ_{20}. Each θ_{20} would have a slightly different set of coefficients in the linear equations. These equations are then manipulated to produce the quazilinear control equations.

Because of the trigonometric terms equations (3-15) and (3-16) are nonlinear. To obtain a linearized approximation a Taylor series approximation is used for the sine and cosine. The derivation begins by defining the reference angles θ_{10} and θ_{20} for θ_1 and θ_2 respectively. These

reference angles are the commanded angle values and will also be used as the operating point for linearization. As a simplifying assumption

$$\dot{\theta}_{10} \to 0, \quad \dot{\theta}_{20} \to 0.$$

This assumption is acceptable because low velocities are generally associated with accurate motions and under those conditions the above assumption will be valid. High velocity motion is generally large angle sweep gross motions that take place prior to the accurate low velocity motions. Under these conditions the error introduced by the assumption is not important. With the above assumptions we define the difference terms:

$$\beta_1 = \theta_1 - \theta_{10}, \quad \dot{\beta}_1 = \dot{\theta}_1, \quad \ddot{\beta}_1 = \ddot{\theta}_1 \tag{3-17}$$

$$\beta_2 = \theta_2 - \theta_{20}, \quad \dot{\beta}_2 = \theta_2, \quad \ddot{\beta}_2 = \ddot{\theta}_2 \tag{3-18}$$

Next, applying the Taylor series approximation,

$$f(z) = \sum_{m=o}^{\infty} \frac{f^m(a)}{m!} (z-a)^m$$

to the sine and cosine terms, we have,

$$\cos(\theta_2) = \cos(\theta_{20}) - \beta_2\sin(\theta_{20}) \tag{3-19}$$

$$\sin(\theta_2) = \sin(\theta_{20}) + \beta_2\cos(\theta_{20}) \tag{3-20}$$

Recalling that $\theta_{x2} = \theta_1+\theta_2$ and substituting equations (3-17) through (3-20) into (3-15) and (3-16) we have, finally:

$$[I_{A1}+m_2L_1^2+m_2L_1n_2\cos(\theta_{20})]\ddot{\beta}_1+[m_2L_1n_2\cos(\theta_{20})]\ddot{\beta}_2 = T_1-T_2 \tag{3-21}$$

$$[I_{B2}+m_2L_1n_2\cos(\theta_{20})]\ddot{\beta}_1 + I_{B2}\ddot{\beta}_2 = T_2 \tag{3-22}$$

Manipulating equations (3-21) and (3-22), one derives,

$$\ddot{\beta}_1 = \frac{1}{I_2}\,T_1 - \frac{I_{B2}+I_1}{I_{B2}I_2}\,T_2 \tag{3-23}$$

$$\ddot{\beta}_2 = -\frac{I_{B2}+I_1}{I_{B2}I_2}\,T_1 + \frac{I_{B2}+I_1+I_2}{I_{B2}I_2}\,T_2 \tag{3-24}$$

where,

$$I_1 = m_2L_1n_2\cos(\theta_{20})$$

$$I_2 = I_{A1} + m_2L_1^2 + \frac{I_1^2}{I_{B2}}.$$

Stating these results in state variable form,

$$\frac{d}{dt}\begin{bmatrix}\beta_1\\ \beta_2\\ \beta_1\\ \beta_2\end{bmatrix} = \begin{bmatrix}0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0\end{bmatrix}\begin{bmatrix}\beta_1\\ \beta_2\\ \dot{\beta}_1\\ \dot{\beta}_2\end{bmatrix} + \begin{bmatrix}0 & 0\\ 0 & 0\\ b_{31} & b_{32}\\ b_{41} & b_{42}\end{bmatrix}\begin{bmatrix}T_1\\ T_2\end{bmatrix} \quad (3\text{-}25)$$

where,

$$b_{31} = \frac{1}{I_2}$$

$$b_{32} = -\frac{I_{B2}+I_1}{I_{B2}I_2}$$

$$b_{41} = -\frac{I_{B2}+I_1}{I_{B2}I_2}$$

$$b_{42} = \frac{I_{B2}+I_1+I_2}{I_{B2}I_2}$$

We desire a control that will produce torque signals that will be proportional to the angular velocity and displacement errors, or,

$$\begin{bmatrix}T_1\\ T_2\end{bmatrix} = \bar{K}\begin{bmatrix}\beta_1\\ \beta_2\end{bmatrix} + \bar{C}\begin{bmatrix}\dot{\beta}_1\\ \dot{\beta}_2\end{bmatrix} \quad (3\text{-}26)$$

or

$$\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} K_{11} & K_{12} & C_{11} & C_{12} \\ K_{21} & K_{22} & C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \dot{\beta}_1 \\ \dot{\beta}_2 \end{bmatrix} \tag{3-27}$$

The elements of the $\bar{K}$ and $\bar{C}$ matrices can be found by using a linear approximation and the state variable formulation described by Tahahashi [4].

The arm system behaves according to

$$\frac{d}{dt}\bar{X} = \bar{A}\bar{X} + \bar{B}\bar{U} \tag{3-28}$$

which is a restatement of equation (3-25), and has an output $\bar{y} = \bar{D}\bar{X}$ (3-29). Desired is a matrix, $\bar{M}$, such that,

$$\frac{d}{dt}\bar{X} = \bar{A}_d\bar{X},$$

for

$$\bar{U} = -\bar{M}\bar{y}, \tag{3-30}$$

where $\bar{A}_d$ is the desired plant behavior. In this case,

$$\bar{A} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \bar{U} = \begin{bmatrix} T_1 \\ T_2 \end{bmatrix}$$

$$\bar{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ b_{31} & b_{32} \\ b_{41} & b_{42} \end{bmatrix}, \quad \bar{X} = \bar{Y} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_1 \\ \beta_2 \end{bmatrix}$$

$$M = \begin{bmatrix} K_{11} & K_{12} & C_{11} & C_{12} \\ K_{21} & K_{22} & C_{21} & C_{22} \end{bmatrix}; \quad \bar{C} = \bar{I}$$

Substituting (3-29) into (3-30) we have,

$$\bar{U} = -\bar{M}\bar{C}\bar{X} \tag{3-31}$$

Substituting (3-31) into (3-28) results in

$$\frac{d}{dt}\bar{X} = (\bar{A}-\bar{B}\bar{M}\bar{C})\bar{X} = \bar{A}_d\bar{X}.$$

From this we see that

$$\bar{A} - \bar{B}\bar{M}\bar{C} = \bar{A}_d \tag{3-32}$$

The desired plant behavior would be to have β_1 and $\dot{\beta}_1$ decoupled from β_2 and $\dot{\beta}_2$. Thus,

$$\bar{A}_d = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -\omega_1^2 & 0 & -2\rho\omega_1 & 0 \\ 0 & -\omega_2^2 & 0 & -2\rho\omega_2 \end{bmatrix}$$

where,

ρ = damping ratio

ω_1, ω_2 = system eigenvalues

Carrying out the operations in (3-32) we arrive at the following set of simultaneous equations,

$$b_{31}K_{11} + b_{32}K_{21} = -\omega_1^2 \qquad (3\text{-}33a)$$

$$b_{31}K_{12} + b_{32}K_{22} = 0 \qquad (3\text{-}33b)$$

$$b_{31}C_{11} + b_{32}C_{21} = -2\rho\omega_1 \qquad (3\text{-}33c)$$

$$b_{31}C_{12} + b_{32}C_{22} = 0 \qquad (3\text{-}33d)$$

$$b_{41}K_{11} + b_{42}K_{21} = 0 \qquad (3\text{-}33e)$$

$$b_{41}K_{12} + b_{42}K_{22} = -\omega_2^2 \tag{3-33f}$$

$$b_{41}C_{11} + b_{42}C_{21} = 0 \tag{3-33g}$$

$$b_{41}C_{12} + b_{42}C_{22} = -2\rho\omega_2 \tag{3-33h}$$

$\bar{K}$ and $\bar{C}$ are found by manipulating the expressions in (3-33).

$$\bar{K} = \frac{1}{|\bar{b}|}\begin{bmatrix} -b_{42}\omega_1^2 & b_{32}\omega_2^2 \\ b_{41}\omega_1^2 & -b_{31}\omega_2^2 \end{bmatrix}$$

$$\bar{C} = \frac{1}{|\bar{b}|}\begin{bmatrix} -2b_{42}\rho\omega_1 & 2b_{32}\rho\omega_2 \\ 2b_{41}\rho\omega_1 & -2b_{31}\rho\omega_2 \end{bmatrix}$$

and

$$|\bar{b}| = b_{31}b_{42} - b_{32}b_{41} \tag{3-34}$$

Substituting the values for $\bar{b}$ into (3-34) and doing a lot of arithmetic,

$$\frac{1}{|\bar{b}|} = \frac{(I_{B2}I_2)^2}{I_3} \tag{3-35}$$

where

$$I_3 = I_{B2}(I_{A1}+m_2L_1^2-I_1)$$

Substituting the derived values for $\bar{K}$ and $\bar{C}$ into (3-26) we have finally,

$$\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} \frac{-(I_{B2}I_2)(I_{B2}+I_2+I_1)}{I_3}\omega_1^2 & \frac{-(I_{B2}I_2)(I_{B2}+I_1)}{I_3}\omega_2^2 \\ \frac{-(I_{B2}I_2)(I_{B2}+I_1)}{I_3}\omega_1^2 & \frac{-(I_{B2}I_2)^2}{I_3I_2}\omega_2^2 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \end{bmatrix}$$

$$+ \begin{bmatrix} \frac{-(I_{B2}I_2)(I_{B2}+I_2+I_1)}{I_3}2\rho\omega_1 & \frac{-(I_{B2}I_2)(I_{B2}I_1)}{I_3}2\rho\omega_2 \\ \frac{-(I_{B2}I_2)(I_{B2}+I_1)}{I_3}2\rho\omega_1 & \frac{-(I_{B2}I_2)^2}{I_3I_2}2\rho\omega_2 \end{bmatrix} \begin{bmatrix} \dot{\beta}_1 \\ \dot{\beta}_2 \end{bmatrix} \qquad (3\text{-}36)$$

where

$$I_1 = m_2L_1n_2\cos(\theta_{20}) \qquad (3\text{-}36b)$$

$$I_2 = I_{A1}+m_2L_1^2+I_1^2/I_{B2} \qquad (3\text{-}36c)$$

$$I_3 = I_{B2}(I_{A1}+m_2L_1^2-I_1) \qquad (3\text{-}36d)$$

This completes the derivation of linear proportional control algorithm for the two disk arm. The implementation of the algorithm in the Texas Instruments 940/4 microcomputer is the subject of the next chapter.

CHAPTER IV

CONTROL SYSTEM SOFTWARE

4-1. Texas Instruments 990/4 Microcomputer

A brief description of the 990/4 microcomputer architecture will help the reader understand the reasoning behind the design of interfacing hardware and why some operations in the software are carried out the way that they are.

The word length of the 990/4 is 16 bits with the most significant bit (MSB) being a sign bit for most instructions. The word length permits the computer to have both word (16 bits) and byte (8 bits) instructions. The machine also uses a "workspace" concept which uses user defined workspaces or groups of registers in memory rather than a fixed set of hardware registers. This eliminates the need to save the contents of registers on a stack when servicing an interrupt. The interrupt just vectors execution to the workspace for the interrupt handler then, when the interrupt service is complete, execution returns to the original unchanged workspace.

Because of the number of times the operations are carried out, the multiplication and division instructions are of particular interest. Both commands are register commands, that is they operate on numbers that are in the

workspace registers. The multiply command multiplies a one word multiplicand by a one word multiplier to make a two word product destroying the multiplicand in the process. In short, where "R" refers to register:

$$RA \times RB = (RB,RC).$$

The division command divides a two word dividend by a one word divisor to form a one word quotient or:

$$(RA,RB)/RC = R0$$

Both commands have complicating qualities. For instance, the division command may fail to execute because the relationship between the dividend and divisor is such that the quotient would be a number that is too large to fit in one word. The multiply command assumes the multiplier and multiplicand are unsigned 16 bit numbers rather than signed 15 bit numbers requiring the user to do some extra coding if the system will be using signed numbers. Both commands have long execution times. The multiply always takes 17 μsec and the division takes from 30 to 40 μsec.

The 990/4 does all input and output through the communications register unit or CRU which, in this particular machine, is a single register 64 bits long. Each of the bits can be addressed for both input and output giving the machine

64 bits for each mode. Addressing is done through register 12 of any workspace and bits can be addressed singly or in groups of any length up to 16 bits. The instruction set has single bit and multiple bit I/O and testing commands.

The CRU is made available for interfacing with analog-to-digital (A/D) and digital-to-analog (D/A) interface circuit boards through Texas Instruments TTL Interface Boards. These boards each handle 16 CRU lines and are fitted into slots in the computer. In addition, a line from a 120 Hz clock inside the computer and an interrupt line are made available through the TTL board along with logic ground and 5VDC for TTL logic application.

The computer communicates with a Texas Instruments ASR 733 terminal through the CRU. The lowest 16 bits of the CRU are reserved for the ASR 733 and its interface board is plugged into the corresponding slot in the computer chasis. The ASR 733 has a keyboard, a printer and a cassette unit.

4-2. General Definition of Structure

The 990/4 microcomputer will interface with the rest of the system through analog-to-digital and digital-to-analog converters. For this purpose, the computer's interface, the communications register unit or CRU, has available sixty four bits for input and sixty four bits for output. Before a program can be written to execute equation (3-36), one must determine how to use this interface. The

determination, in turn, is based upon the number of variables to input and output and the desired sensitivity of the converters.

Equation (3-36) uses four errors, β_1, β_2, $\dot{\beta}_1$ and $\dot{\beta}_2$. To repeat, these are,

$$\beta_1 = \theta_1 - \theta_{10} \tag{4-1a}$$

$$\beta_2 = \theta_2 - \theta_{20} \tag{4-1b}$$

$$\dot{\beta}_1 = \dot{\theta}_1 \tag{4-1c}$$

$$\dot{\beta}_2 = \dot{\theta}_2 \tag{4-1d}$$

According to equations (4-1c) and (4-1d), $\dot{\theta}_1$ and $\dot{\theta}_2$ must be input to the computer to provide $\dot{\beta}_1$ and $\dot{\beta}_2$. Two approaches may be taken, however, to get β_1 and β_2. Either θ_1, θ_{10}, θ_2 and θ_{20} can be input to the computer or β_1 and β_2 may be read from analog-to-digital interfaces that have amplifiers continually performing the subtractions. The latter approach has the advantage of avoiding the computing time involved in doing the arithmetic in the computer. It should be recalled that a one millisecond computing time limit must be met. The latter approach would reduce the number of variables that must be input which is advantageous because of the limited size of the CRU interface. Examination of equations (3-36b)

through (3-36d) shows that θ_{20} will have to be entered anyway because the terms in $\bar{K}$ and $\bar{C}$ are dependent on that angle.

Five input variables have been determined. These are β_1, β_2, $\dot{\theta}_1$, $\dot{\theta}_2$ and θ_{20}. These will require five analog-to-digital converters connected to some portion of the sixty-four bits of the CRU. The length of each converter word needs now to be determined. Considering cost, setting time, resolution and available CRU space, a twelve bit converter was chosen. The cost is reasonable. The settling time is twenty microseconds and five converters would use sixty bits. A twelve bit converter would also have a 1.2 millivolt resolution for a 5 VDC full analog input range.

Since the computer will have only two outputs, T_1 and T_2, the CRU space limitation is not important. A twelve bit digital-to-analog converter was considered adequate for that application, however, for all the other reasons mentioned.

Control Algorithm Implementation

Before the development of the program can proceed, the equation (3-36a) must be stated in the form that the computer will execute. This entails the substitution of the scale factors from the converters. The scale factors are needed to convert the binary input variable codes into numbers and back again for output when calculations are finalized. In the relations below the symbols with apostrophes are analog to digital and digital to analog converter codes and the others are magnitudes of physical variables.

$$T_1' = T_{1SF}(T1) \tag{4-2a}$$

$$T_2' = T_{2SF}(T2) \tag{4-2b}$$

$$\beta_1' = ASF(\beta_1) \tag{4-2c}$$

$$\beta_2' = ASF(\beta_2) \tag{4-2d}$$

$$\dot{\beta}_1' = VSF(\dot{\beta}_1) \tag{4-2e}$$

$$\dot{\beta}_2' = VSF(\dot{\beta}_2) \tag{4-2f}$$

where

T_{1SF} = shoulder motor torque scale factor = counts/ lb_f-ft

T_{2SF} = elbow motor torque scale factor = counts/ lb_f-ft

ASF = angular position scale factor = counts/rad

VSF = angular velocity scale factor = counts/rad/sec

Substituting equation (4-2) into (3-36) gives,

$$\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = \begin{bmatrix} \frac{T_{1SF}}{ASF} a_{11}\omega_1^2 & \frac{T_{1SF}}{ASF} a_{12}\omega_1^2 \\ \frac{T_{2SF}}{ASF} a_{21}\omega_1^2 & \frac{T_{2SF}}{ASF} a_{22}\omega_1^2 \end{bmatrix} \begin{bmatrix} \beta_1' \\ \beta_2' \end{bmatrix} +$$

$$\begin{bmatrix} \frac{T_{1SF}}{VSF}\, a_{11}{}^{2\rho\omega_1} & \frac{T_{1SF}}{VSF}\, a_{12}{}^{2\rho\omega_2} \\ \frac{T_{2SF}}{VSF}\, a_{21}{}^{2\rho\omega_1} & \frac{T_{2SF}}{VSF}\, a_{22}{}^{2\rho\omega_2} \end{bmatrix} \begin{bmatrix} \dot{\beta}_1' \\ \dot{\beta}_2' \end{bmatrix} \tag{4-3}$$

where,

$$\bar{a} = \begin{bmatrix} -\frac{(I_{B2}I_2)(I_{B2}+I_1+I_2)}{I_3} & -\frac{(I_{B2}I_2)(I_{B2}+I_1)}{I_3} \\ -\frac{(I_{B2}I_2)(I_{B2}+I_1)}{I_3} & -\frac{(I_{B2}I_2)^2}{I_3 I_2} \end{bmatrix} \tag{4-4}$$

and, again,

$$I_1 = m_2 L_1 n_2 \cos(\theta_{20}) \tag{4-5a}$$

$$I_2 = I_{A1}+m_2 L_1{}^2+I_1{}^2/I_{B2} \tag{4-5b}$$

$$I_3 = I_{B2}(I_{A1}+m_2 L_1{}^2-I_1) \tag{4-5c}$$

The goal of the program, thus, is to solve equation (4-3) using $\bar{a}$ in (4-4) which, in turn, requires the solution of the relations in (4-5). To carry out all these operations in one computing period would not be practical. The total amount of time consumed just multiplying and dividing would

far exceed one millisecond. A program is desired that will, in effect, carry out the above operations while, in fact, doing a minimal amount of computing.

The simplest way to achieve this is to have a table of $\bar{K}$ and $\bar{C}$ matrices in the computer memory. The computer would use the value of θ_{20} input to find the proper pair of matrices. To further increase the speed of computing the terms should be grouped in the order in which they are used rather than segregated in two separate matrices. Restating equation (4-3),

$$\begin{bmatrix} T_1' \\ T_2' \end{bmatrix} = \begin{bmatrix} K_{11}' & K_{12}' \\ K_{21}' & K_{22}' \end{bmatrix} \begin{bmatrix} \beta_1' \\ \beta_2' \end{bmatrix} + \begin{bmatrix} C_{11}' & C_{12}' \\ C_{21}' & C_{22}' \end{bmatrix} \begin{bmatrix} \dot{\beta}_1' \\ \dot{\beta}_2' \end{bmatrix} \tag{4-6}$$

one can see that the terms should be tabulated in the following manner,

$$K_{11}', K_{12}', C_{11}', C_{12}', K_{21}', K_{22}', C_{21}', C_{22}'.$$

The second computing obstacle is the fact that the terms in the $\bar{K}$ and $\bar{C}$ matrices will undoubtably not be integers. This means that the coefficient would have to be expressed in floating point binary and that to find a torque each operation on an error term would require both a multiplication and a division. Since the execution time for a

multiplication is 17.33 microseconds and for a division is 40.0 microseconds, eight multiplications and eight divisions would consume 459 microseconds. This is still probably too much time for just these operations alone. The computer must do input and output operations, addition, sign checking of errors and other searching and moving operations. A one millisecond turn around time is unlikely if almost half the computing time is used to multiply and divide.

Since the divide command is of the longest duration it is the prime candidate for removal. This can be done by capitalizing on the qualities of the multiply command and of floating point binary. The multiply command multiplies a one word number by another one word number to give a product that is two words in length. Ideally, rather than go through the time consuming torture of doing arithmetic on two-word numbers after multiplication, one would like to simply "forget" the low order word. This "forgetting" is equivalent, interestingly, to dividing the product by 2^{16}. Clearly, to eliminate the divide command from the computations, the terms of the $\bar{K}$ and $\bar{C}$ matrices should all be expressed in the table in the form:

$$\text{integer} \times 2^{-16}.$$

Of course, since all the exponents are -16, only the integers need to be listed. This scheme also hinges on the

magnitudes of the physical parameters making up the terms permitting the coefficients to be expressed in this fashion. Calculations of extreme values show that they do.

Finally, programmable (keyboard alterable) and non-programmable constants should be distinguished. For experimental reasons the eigenvalues and damping ratio should be programmable. Also, the converter scale factors should be programmable. This will permit one to change gains or sensors or to increase the position error sensitivity. The terms in the $\bar{a}$ matrix can be considered non-programmable.

In order to have a coefficient table when some of its terms are programmable two more tables need to exist. One table would be an $\bar{a}$ matrix table with the sets of terms tabulated in order of increasing θ_{20}. The second table would be a list of programmable variables. When the arm system is in an inoperative mode the user could change a variable on the programmable list by typing it in on the keyboard. The computer would then use the new list with the $\bar{a}$ matrix table to make a new coefficient table.

The structure of the program is now broadly defined. It will have two divisions, an arm control that will solve equation (4-6) and a section that will work with the keyboard. These two divisions cannot operate simultaneously or either the arm will go out of control or keyboard input will be lost. Information used in control will be in the form of five inputs and a table of coefficients. The table of

coefficients will be made by the keyboard section. The details of the structure will be discussed more fully in the next section.

4-3. Program Organization and Execution

A. Arm Controller

For the following discussion the reader is referred to Appendix I which contains a flowchart of the program. For an outline of operating instructions for the system the reader should refer to Appendix III.

To permit the experimenter to control the sample rate of the controller, interrupt programming will be used for the control section. The computer manufacturer supplied TTL interface boards have interrupt bits from the computer brought out onto them. One of these bits is connected to a connector for a variable frequency pulse generator thus allowing the experimenter to vary the sample rate by varying the pulse generator frequency.

The following scheme has been divised for the coefficient table. Since each link will operate in a region of ±90° a table covering 90° is necessary. For computing convenience 90° is divided into 64 equal increments. As a result the table consists of sixty four sets of coefficients. When θ_{20} is read a simple process of shifting and adding is all that is necessary to find the first address of the proper set of coefficients.

When an interrupt is received program execution jumps to the subroutine RED. This subroutine first initializes the error sign storage words then tests the emergency stop bit. This bit is set if either link of the arm travels beyond its maximum permissible angle. If it is set then it initializes the coefficient pointer to the beginning address of the emergency stop coefficients. If the bit is not set then RED reads θ_{20} and initializes the coefficient pointer to the first word of the proper set of coefficients. Having done this, the routine then reads the four errors taking their absolute values and remembering their signs. If either velocity exceeds its maximum limit a constant is added to bias the torque towards a velocity reduction.

At this point the program is ready to find the torques. Using automatic incrementing instructions it moves a coefficient to a register then multiplies it by the error. Division by 2^{16} is implied by ignoring the low order word of the product. The sign of the error is checked and the sign of the product altered if necessary. The product is finally summed into the proper torque word and the program proceeds to the next error/coefficient pair. When four multiplications have been executed the program knows that the first torque has been found and it repeats the process for the second torque. When both torques have been found they are output through the CRU. Before output both values are inverted (one's compliment, i.e. zeros become ones and ones become

zeroes). This is done to compensate for an inversion that takes place on the TTL interface board.

B. Keyboard Interface

The keyboard interface is designed to execute the commands listed in Table 4-1 and to accept numbers in a particular fashion. A number must be entered on the keyboard as an integer divided by a power of two with only the exponent being typed, i.e.

(integer numerator)/(exponent of power of two)

For example, the number 6.73 could be approximated by $6892/2^{10}$. This would be entered on the keyboard as 6892/10.

When the user types a command on the keyboard the keyboard interrupt causes a branch to KBI. KBI brings the arm to a safe stop then disables the arm interrupt. At this point it can safely read the character and decode it. Having done so, it asks the user the proper questions, accepting numerical input and calling CONV to convert the ASCII-decimal characters to binary numbers. It then updates the programmable constants list and calls BILD to rebuild the coefficient table. When the new table has been made KBI reenables the arm interrupt and returns from interrupt. If a symbol other than a command is typed in KBI will reply "WHAT?" and wait for input.

Called by KBI, CONV takes digits off the read buffer

Table 4-1. Command List

P--alter eigenvalues and damping ratio

A--alter angular position scale factors

V--alter velocity scale factors

T--alter torque scale factors

E--end of program. Disables arm interrupt. Typing any character will restart

G--mistake. Return to arm control

and puts them on a decimal-to-binary converter buffer. It searches for a slash code and for the carriage return code. Using these codes it determines whether it is looking at a numerator or a denominator exponent. When it encounters one of the terminating codes it calls DBIN which performs the conversion on the digits in the buffer leaving the resulting binary number in R0. CONV then transfers the number to the proper location on the programmable constants list and either returns to KBI or searches for the next number. CONV can handle the following input strings:

(INTEGER)/(EXPONENT) CR

(INTEGER)/ CR

(INTEGER) CR

CR

The last case would tell CONV that the particular programmable constant is not to change. The preceding two cases would tell it that the magnitude of the denominator is one.

To understand the logic of BILD one needs to know how the tables are organized. The order of terms is depicted in Tables 4-2 through 4-4. To carry out the

Table 4-2. Programmable Constants List

Location Label	Contents
NF11N	ω_1 numerator
NF11D	ω_1 denominator exponent
NF21N	ω_2 numerator
NF21D	ω_2 denominator exponent
DRN	ρ numerator
DRD	ρ denominator exponent
NF12N	ω_1^2 numerator
NF12D	ω_1^2 denominator exponent
NF22N	ω_2^2 numerator
NF22D	ω_2^2 denominator exponent
NF1DN	$2\rho\omega_1$ numerator
NF1DD	$2\rho\omega_1$ denominator exponent
NF2DN	$2\rho\omega_2$ numerator
NF2DD	$2\rho\omega_2$ denominator exponent
ASF	angle scale factor numerator
	angle scale factor denominator
VSF	velocity scale factor numerator
	velocity scale factor denominator
TSF1	shoulder torque scale factor numerator
	shoulder torque scale factor denominator
TSF2	elbow torque scale factor numerator
	elbow torque scale factor denominator

Table 4-3. Angle Dependent Constants and Coefficients Tables

COEF	a_{11}	TBL	K_{11}
	a_{12}		K_{12}
	a_{21}		C_{11}
	a_{22}		C_{12}
	a_{11}		K_{21}
	a_{12}		K_{22}
	⋮		C_{21}
	a_{22}		C_{22}
CEND	ϕ		⋮
		ETBL	ϕ

Table 4-4. Intermediate Buffer for BILD

TMP	$\Delta SF \times TSF1 \times \omega_1^2$	numerator
		denominator exponent
	$ASF \times TSF1 \times \omega_2^2$	numerator
		denominator exponent
	$VSF \times TSF1 \times 2\rho\omega_1$	numerator
		denominator exponent
	$VSF \times TSF1 \times 2\rho\omega_2$	numerator
		denominator exponent
	$ASF \times TSF2 \times \omega_1^2$	numerator
		denominator exponent
	$ASF \times TSF2 \times \omega_2^2$	numerator
		denominator exponent
	$VSF \times TSF2 \times 2\rho\omega_1$	numerator
		denominator exponent
	$VSF \times TSF2 \times 2\rho\omega_2$	numerator
		denominator exponent
ETMP	0	

operation of making the coefficients in the list beginning with address TBL, two steps are needed. The first step makes an intermediate list of the scale factors and eigenvalues multiplied together as shown in Table 4-4. To do this it first calls TERM which makes the $2\rho\omega_1$ and $2\rho\omega_2$ terms. It then does a series of multiplications using indirect pointer commands. The eigenvalue pointer starts with ω_1^2 and autoincrements through $2\rho\omega_2$. The angle and velocity scale factor pointer points the angle scale factor for two operations then at the velocity scale factor for two operations. The torque scale factor pointer remains at the same location for all four operations. After the fourth multiplication the torque scale factor pointer is incremented and the process repeated. When the output pointer to the intermediate buffer points to ETMP, BILD knows that the buffer is complete.

Each multiplication consists of the multiplication of numerators and the addition of exponents. BILD then calls SHORT which shortens the two word numerator product to one word adjusting the exponent accordingly. This is done by a simple process of successive division of the product by two and the subtraction of one from the exponent. When the high order word equals zero one more cycle is done to make the low order word positive and the process is complete.

When it has completed the intermediate buffer, BILD initializes pointers to COEF, the first address of the $\bar{a}$

matrix table, to TMP, the first address of the intermediate buffer, and to TBL, the first address of the coefficient table. For each reference angle BILD calculates the eight coefficients by going down the list of eight intermediate buffer terms once and the four $\bar{a}$ terms twice. After each multiplication SHORT is called to reduce the numerator products to one word length. FIT is then called to further manipulate the term to produce a denominator exponent equal to sixteen. This is done by another process of successive divisions or multiplications which ends when the exponent reaches sixteen. BILD transfers the numerator to the coefficient list. The transfer of the dominator exponent is not necessary since it is known to be equal to sixteen. BILD knows it has completed the table when the input pointer points to CEND.

INIT is the "main program" of the system. When the program is run execution begins here. It first clears pending interrupts and defines a workspace. It then initializes the CRU/keyboard interface. Finally, before permitting arm operation to begin, it moves the arm into a starting position by calling RED. When the position error between the arm and its reference is less than a certain value for each link INIT enables the arm interrupt and sends a start message to the user. The computer is set into IDLE mode and responds to interrupts from that state. After servicing the interrupt it reenters IDLE.

Character transmission is handled by four routines. KBR reads a character and echoes it back to the printer. The character is stored on the read buffer which starts at location RBUF. KBW writes a character string. Each string consists of the character string in continuous bytes with the number of characters in the string in the word immediately preceding it. KBW puts the number in a counter and calls SND repeatedly decrementing the counter after each call. When the counter equals zero KBW knows that the whole string has been transmitted. SND transmits one character and introduces a transmission delay by calling SND. The need for the delay arises from the different transmission rates of the computer and keyboard interfaces. The computer transmits at 1200 baud and the keyboard interface transmits at 300 baud. For characters a three character relay is necessary to synchronize the two devices. Since the duration of a carriage return is equivalent to the printing time of eight characters a twenty-four character delay is introduced for carriage returns.

This approach exceeds the one millisecond computing time limit specification. The longest path computing time for RED is approximately 800 microseconds. This will give the researcher 200 free microseconds for subroutines simulating effects such as Coulomb friction or deadband. This can be enough time to execute from one to as many as three simple subroutines.

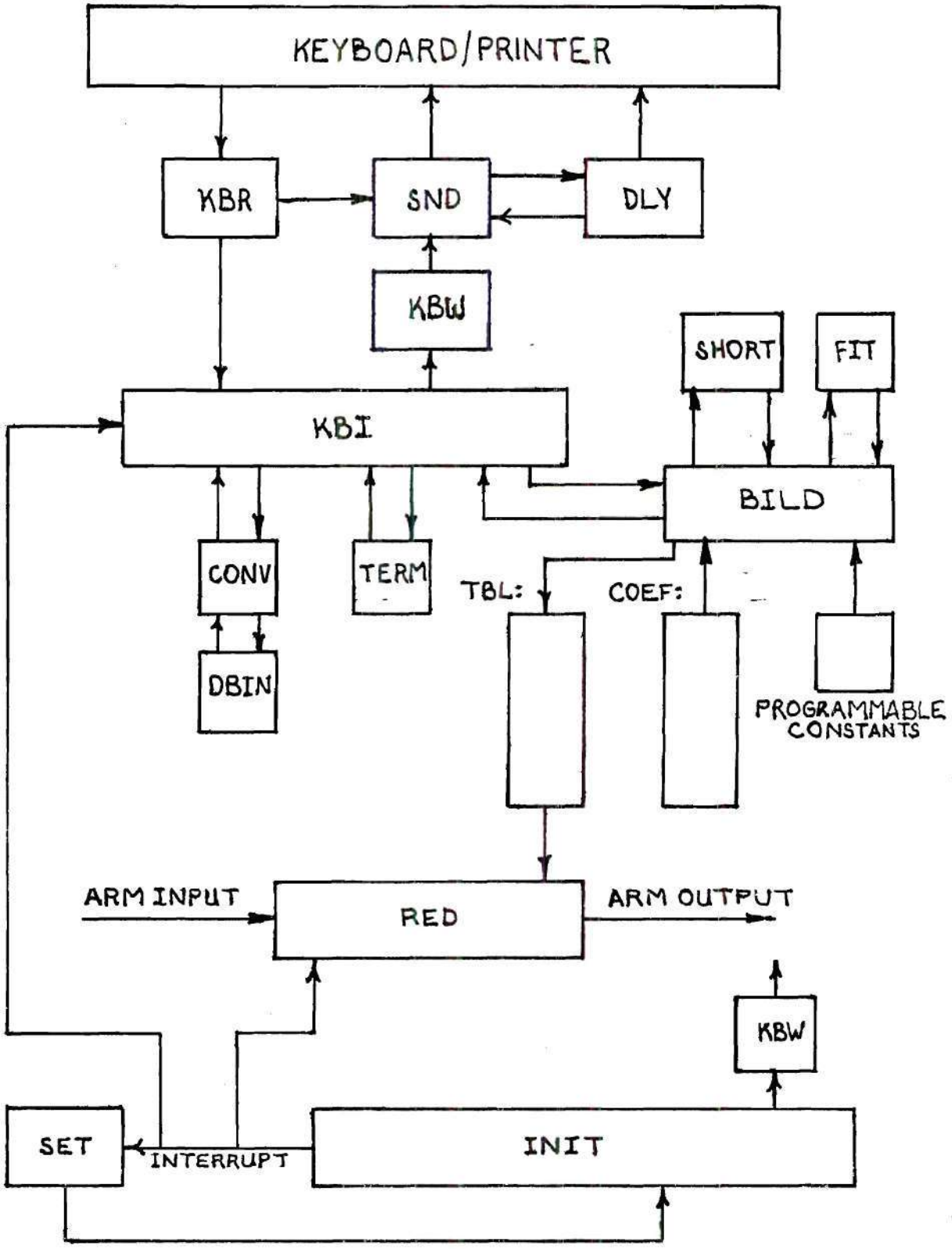


Figure 4-1. Computer Program Organization

References to input and output have undoubtably been unclear. They have been minimized in order not to bog the reader down in a mire of cross references between software and manufacturer and "homemade" hardware. The next chapter should fill out the reader's understanding of the system and, the writer hopes, clarify references made earlier in this chapter.

CHAPTER V

CONTROL HARDWARE

5-1. Analog Devices

As stated previously five analog signals, β_1, β_2, $\dot{\beta}_1$, $\dot{\beta}_2$ and θ_{20}, will be input to the computer from the arm and two signals, T_1 and T_2, will be output from the computer. For the following discussion the reader is referred to the schematic diagram in Figure 5-1. As can be seen in the figure the angles θ_{10} and θ_{20} are sensed by two 5KΩ conductive plastic potentiometers. The full scale voltage range across the potentiometers is +5 VDC to -5 VDC. This range is used because it is compatible with the ±90 degrees operating region convention and because it is a convenient magnitude for use with analog-to-digital converters. The +5 VDC and -5 VDC are supplied by a group of voltage dividers that are supplied with +15 VDC and -15 VDC. Each of the four terminals of the two potentiometers has its own voltage divider. This was done to minimize the sensitivity of the voltage drop across each potentiometer to changes in the equivalent parallel resistance in the other divider.

The angular velocities are measured by using tachometers. The tachometers are mounted on the opposite end of the drive shaft for each joint from the torque motors and will generate 92 VDC at a maximum operating speed of

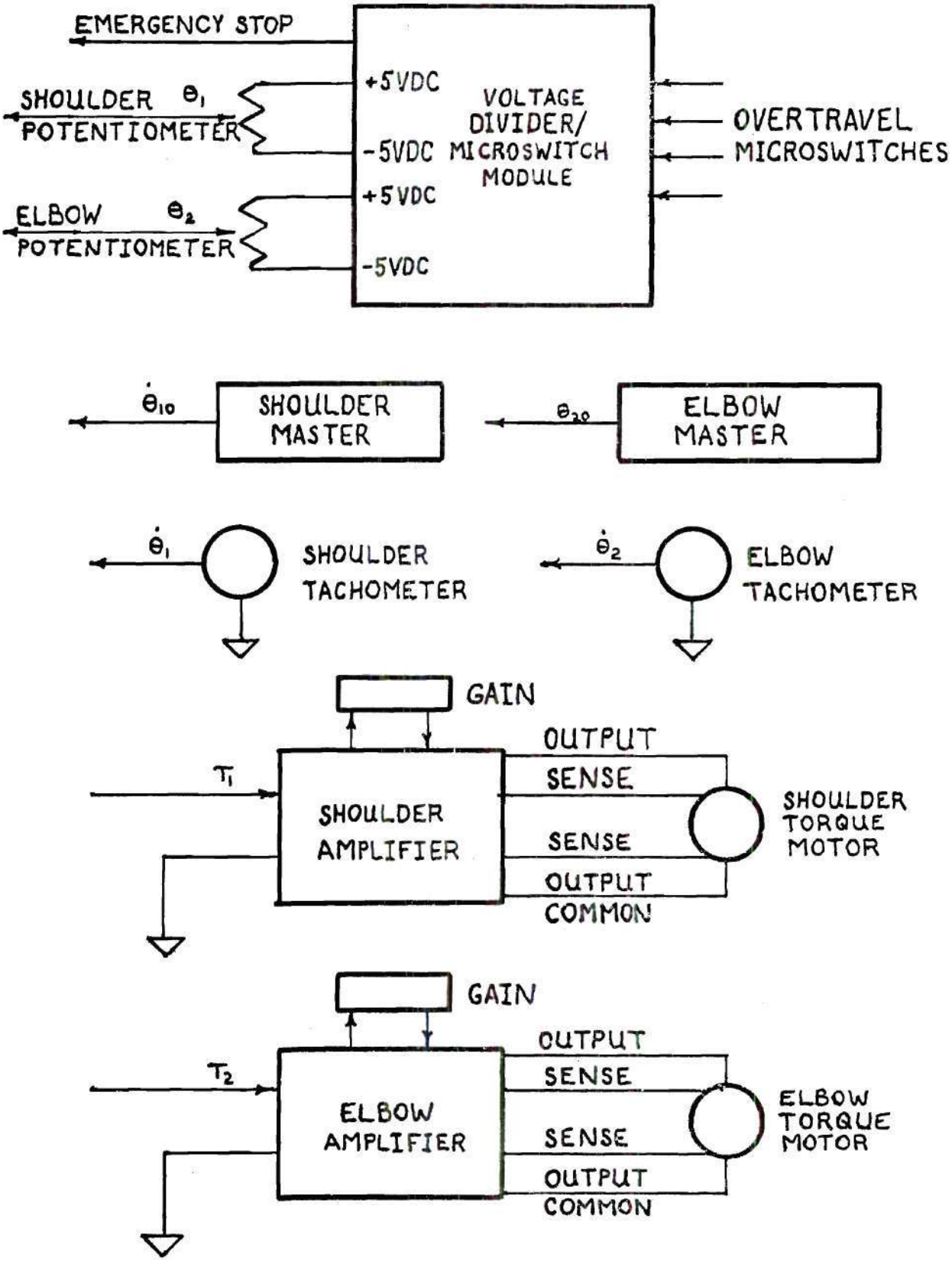


Figure 5-1. Schematic Diagram of Analog Instrumentation

77 rad/sec. The output of the tachometers is inconveniently large for analog-to-digital converters and will, therefore, require a voltage divider on the analog-to-digital interface.

The torque motors, as previously mentioned, are both mounted at the shoulder end of the arm. The elbow torque motor will require a 25.9 VDC and 4.80 amp to produce its maximum 2.0 ft-lb_f torque. Since a digital-to-analog converter produce only 1.2 VDC, voltage and low current, the converter's output is fed into a power amplifier to produce the required voltage and current. The same situation exists for the shoulder torque motor which requires 19.8 VDC and 13.2 amps to produce 7.0 ft-lb_f of torque. Both amplifiers have adjustable gains, but the gains can only go up to ten. They are provided, however, with connectors by the manufacturer so that external resistors may be used to get gains greater than ten. That has been done to both amplifiers.

Limit Switches

Although it is, strictly speaking, a digital circuit the limit switch circuit will be discussed here. Four microswitches are installed on the arm assembly. These switches are situated such that if either link rotates beyond the boundary of its operating range a switch will be hit. The circuit schematic for the switches is shown in Figure 5-2. When any switch is hit a pulse is generated which clocks one of four flip-flops changing the state of

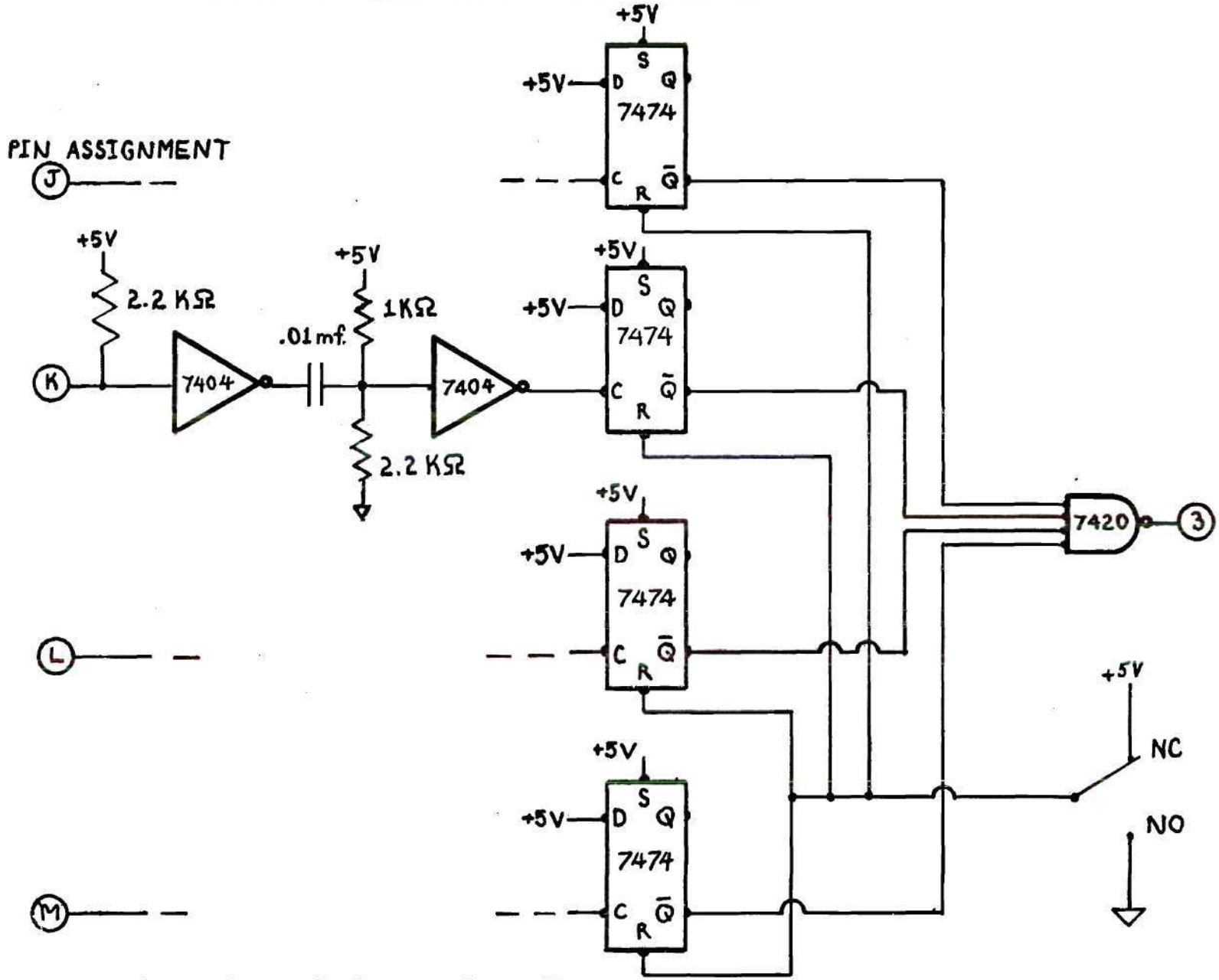


Figure 5-2a. Microswitch Signal Conditioning and Overtravel Signal Circuit

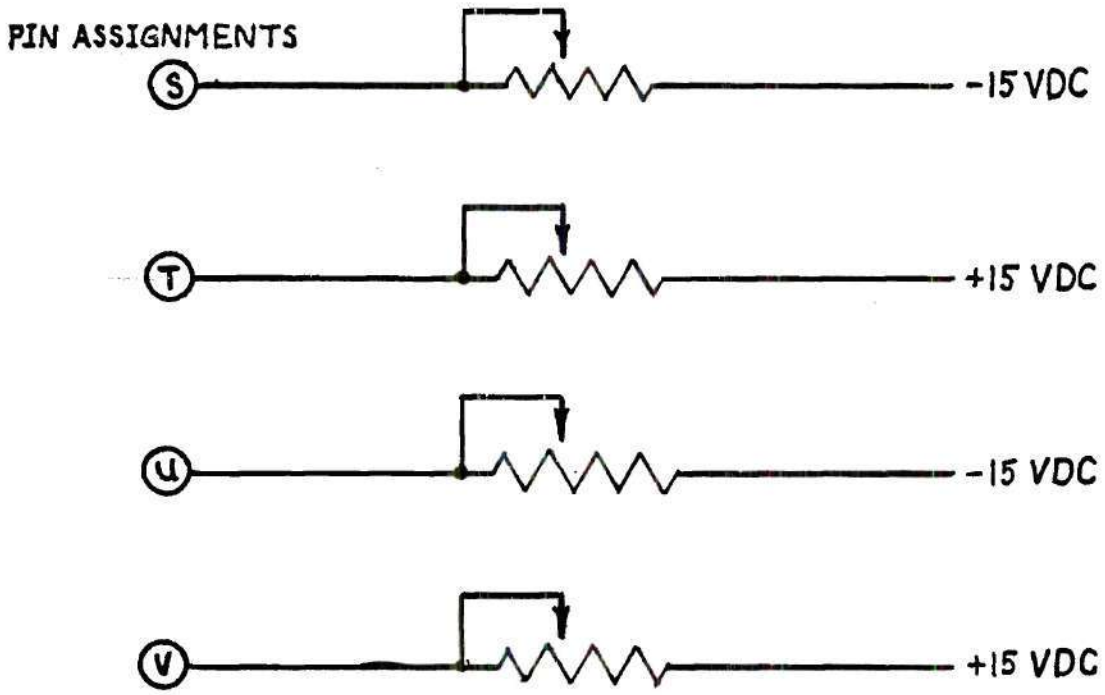


Figure 5-2b. Voltage Divider Circuit

the flip-flop output. The outputs are NANDED together so that when the state one of the flip-flops outputs changes the output of the NAND gate will switch from low to high. This output is the emergency stop flag checked by the control program RED. As an added precaution, the circuit is designed to keep the emergency stop signal on until the user resets it by depressing a button on the front of the circuit module. Since this act requires walking over to the arm the user, the designer hopes, will use the opportunity to place the arm back into the safe operating region before trying to restart it.

5-2. Computer Interface Circuit Boards

Digital to Analog Converters

Figure 5-3 is a schematic diagram of the two digital-to-analog interface circuit boards. The heart of the circuit is a Datel DAC 6912B1 digital-to-analog converter. With the NAND gate at the input to the most significant bit (MSB) the converter will accept 12 bits of twos complement binary. The 10 KΩ variable offset resistor is used to make the output bipolar. The 1.2 KΩ and 3.9 KΩ resistors cause the maximum amplitude of the output to be 1.2 VDC. The converters operate continuously having no start or stop bits.

The circuit in Figure 5-4 accepts the θ_{20} potentiometer signal and converts it to 12 bits of twos complement binary. The MC1456 operational amplifier is used as an impedance

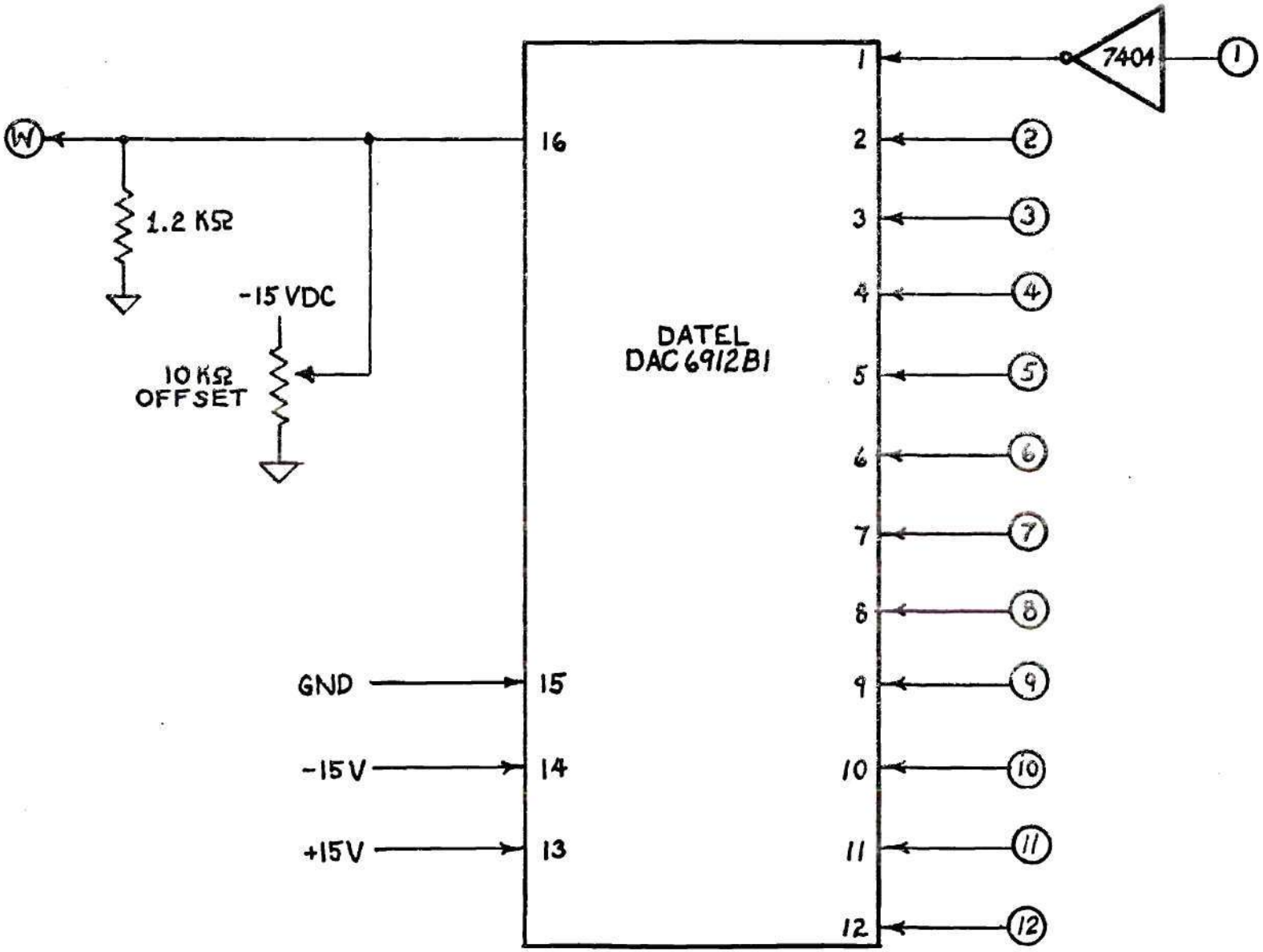


Figure 5-3. D/A Interface Boards Schematic

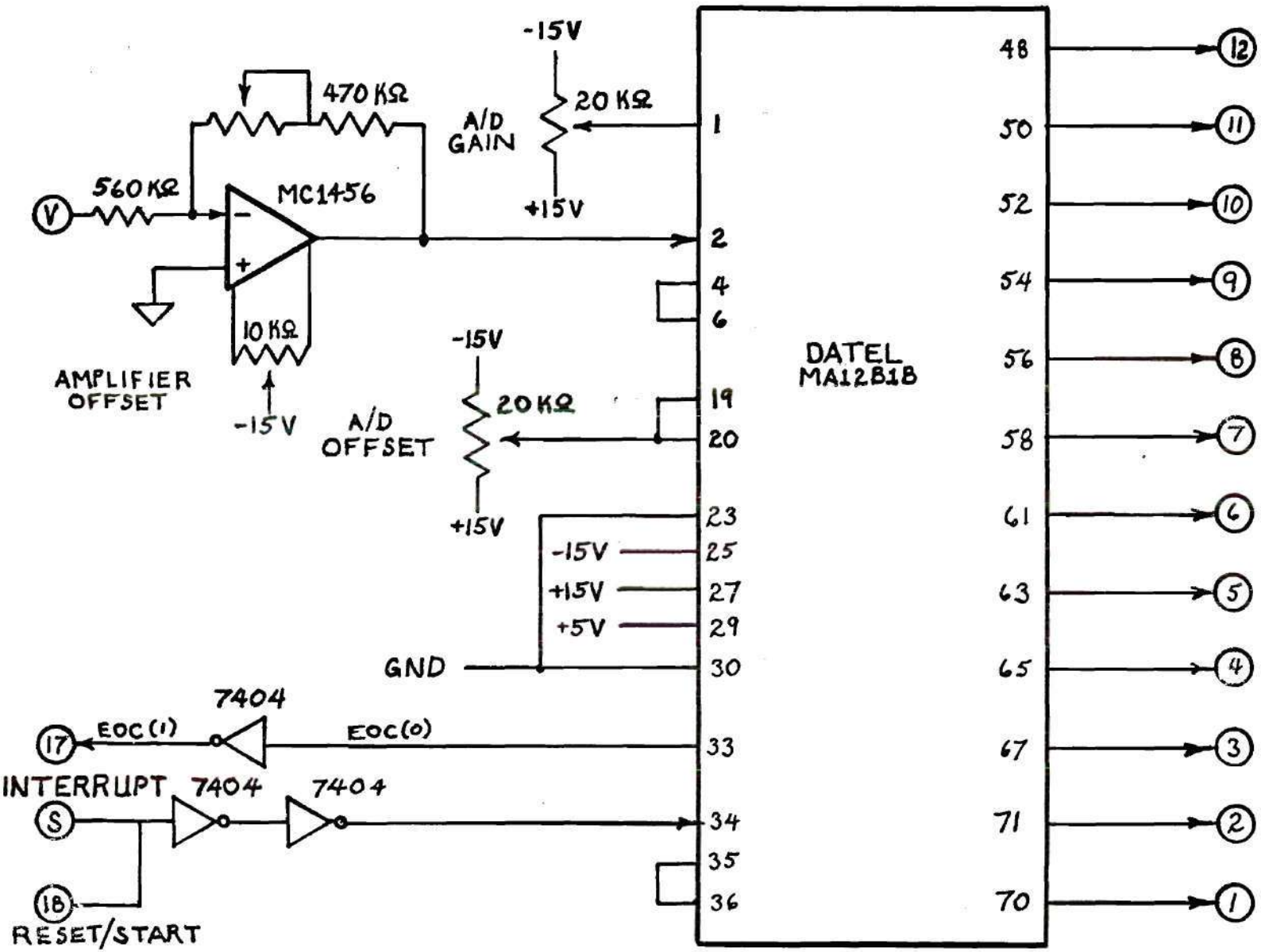


Figure 5-4. θ_{20} A/D Interface Schematic

buffer. The amplifier has an adjustable offset used to trim the amplifier output to zero when both inputs are grounded. It also has, for the user's convenience, a gain adjustable from .84 to 2.63.

Analog to Digital Converters

The analog-to-digital converter, a DATEL ADC-MA12B1B, requires a two step calibration procedure. For both calibrations a generator should be connected to the Start Convert bit. First, with the input voltage equal to -4.999 VDC the offset knob should be turned until the least significant bit flickers on and off. All the other bits, with the exception of the MSB, should equal zero. The gain adjustment requires an input of +4.996 VDC. The gain knob is turned until the LSB flickers and all the other bits, with the exception, again, of the MSB, equal one.

The converter accepts a five volt range bipolar input and starts when it receives either an interrupt pulse from the pulse generator, which is brought to pin "S" of the circuit board or when it receives a pulse from the alternative start/reset on pin 18. When the start conversion line goes from low to high the converter is reset and the end of conversion line, EOC, is set low at pin 17. When the start conversion line goes from high to low conversion is begun. When conversion is complete the converter sends the EOC signal by causing the level at pin 17 to go from low to high.

Figure 5-5 is a schematic of the circuit board that will interface the tachometers to the computer. The user can adjust the voltage divider to reduce the input signal from 90% to 95%. The signal enters a buffer amplifier then the A-to-D converter. The converter and amplifier require the calibrations previously described.

The schematic in Figure 5-6 is the circuit that finds the angular position error. The first operational amplifier finds the difference between the angle and its reference and the second amplifier provides an adjustable gain. Offset adjustment of the amplifiers is carried out in the following manner. The inputs to the first amplifier are grounded then OFFSET 1 is adjusted until the output of the first amplifier is zero. Then, keeping the inputs to the first amplifier grounded, OFFSET 2 is adjusted until the output of the second amplifier is zero.

All the converters are bipolar and all the circuit boards use the same pin assignment convention, which will be described in greater detail in the next section. Despiking capacitors are placed at the power entry point of each circuit board and INVERTERS are used to buffer the EOC and start conversion signal.

5-3. Interface Chassis

Each circuit board described in the preceding section fits into a module which, in turn, is inserted in a chassis. A 40 pin male connector at the back of the module

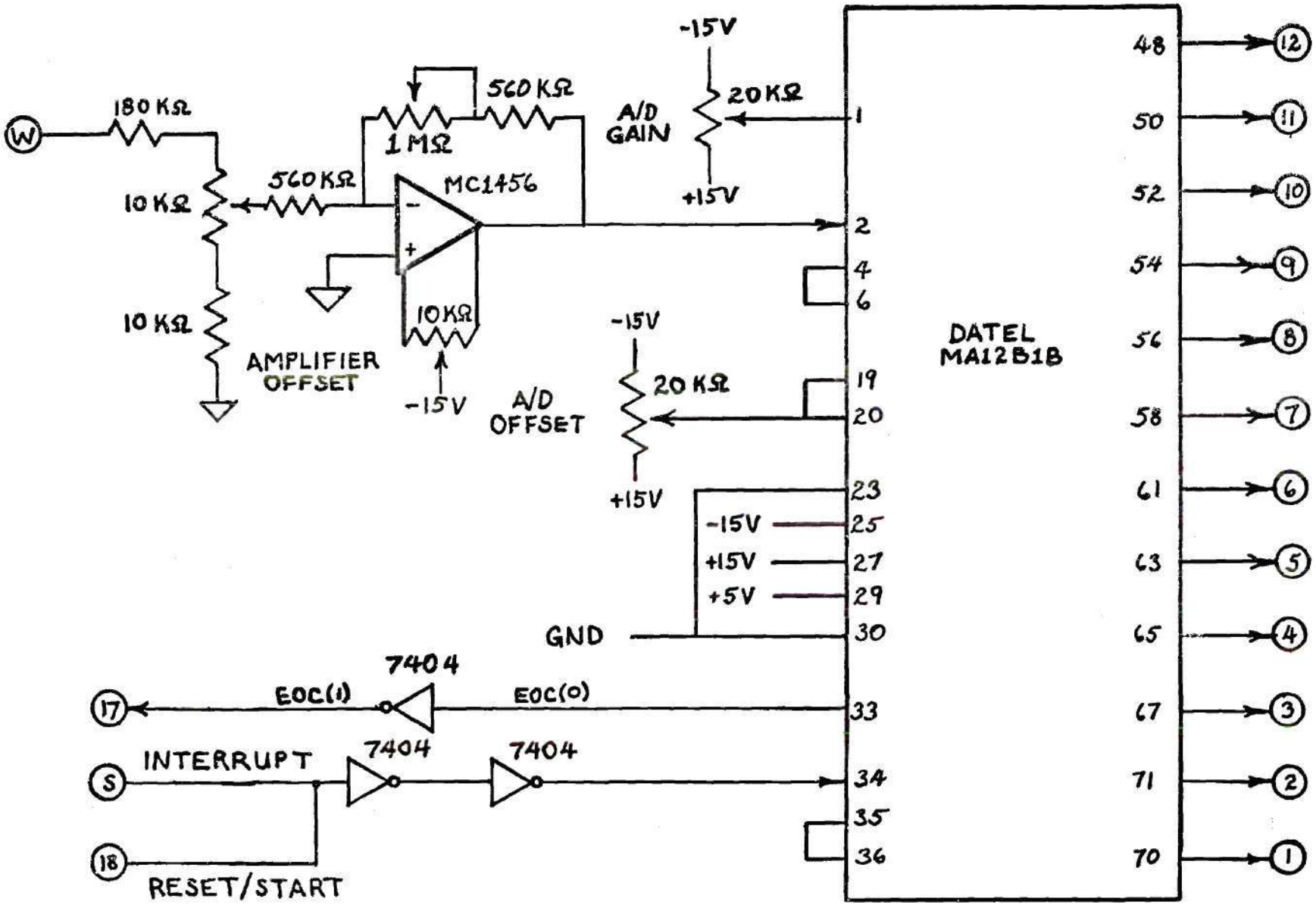


Figure 5-5. Angular Velocity A/D Interface Schematic

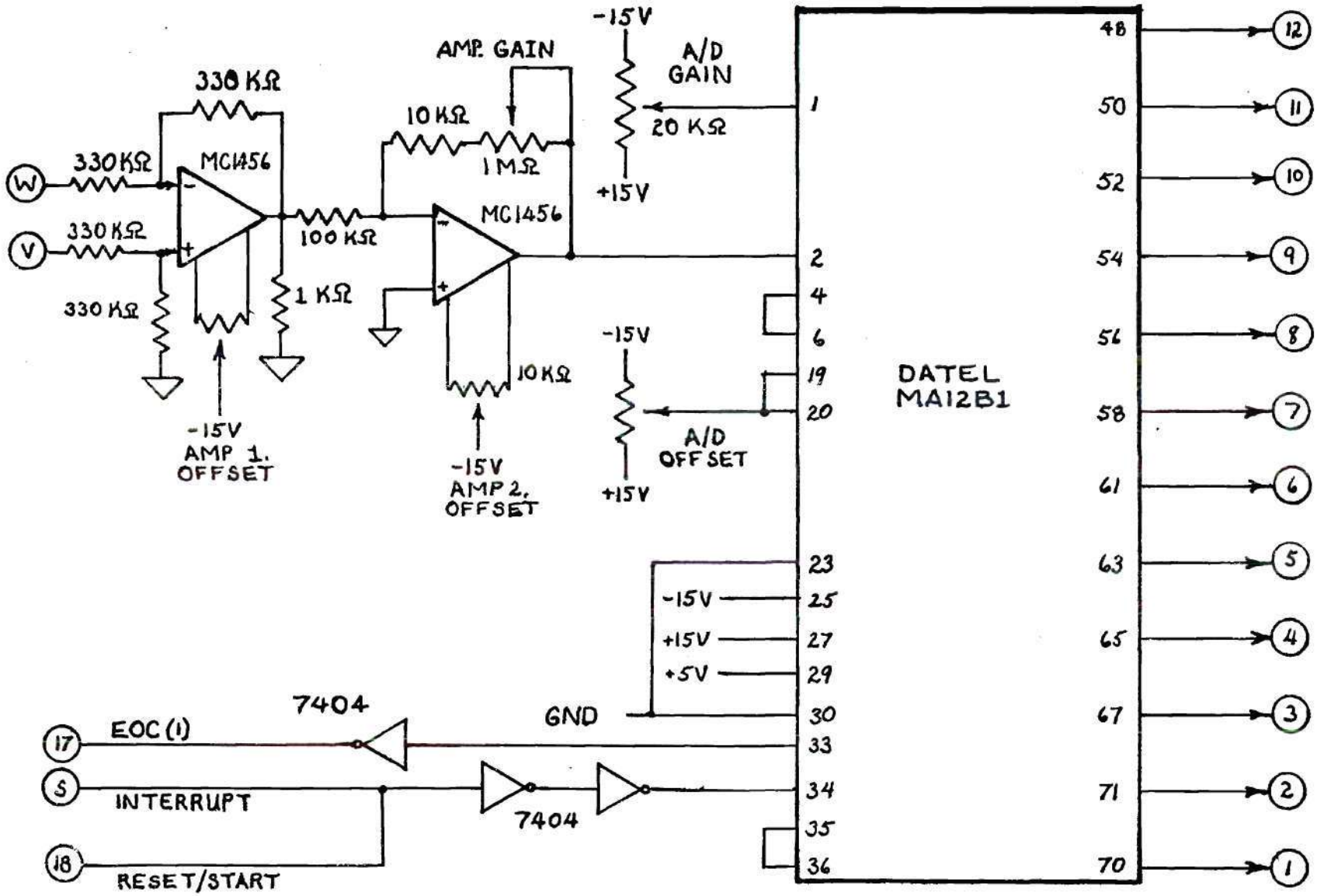


Figure 5-6. β_1 and β_2 A/D Interface Schematic

plugs into a female connector attached to the chassis. At this point an orderly method is needed to connect the computer to the chassis. The approach decided upon was to bring all the CRU bits via ribbon cable and all the pins on the female connectors in the chassis to a wire wrap board mounted on the back of the chassis. The setup is described more fully below.

The CRU is split into four, sixteen bit sections. Each section interfaces with a sixteen bit TTL interface board with a ribbon cable. This division is a convenience for the manufacturer and presents no difficulty to the user. The CRU still looks to the computer like sixty four contiguous bits. The ribbon cables are wire wrapped to a line of sixty four input pins and a line of sixty four output pins on the back of the chassis. Seven rows of eighteen pins are connected to seven module slots in the chassis. The pin assignment convention is shown in Figure 5-7. The eighth module slot is for the power supply. The power supply supplies ±15 VDC and +5 VDC to the modules in the chassis and to the potentiometer voltage dividers.

Analog inputs are through seven pairs of coaxial connectors, an inverting amplifier input and a non inverting amplifier input. These are connected to the module pins via shielded cable.

With this design the user is free to plug in the modules in any order he desires and can connect them via

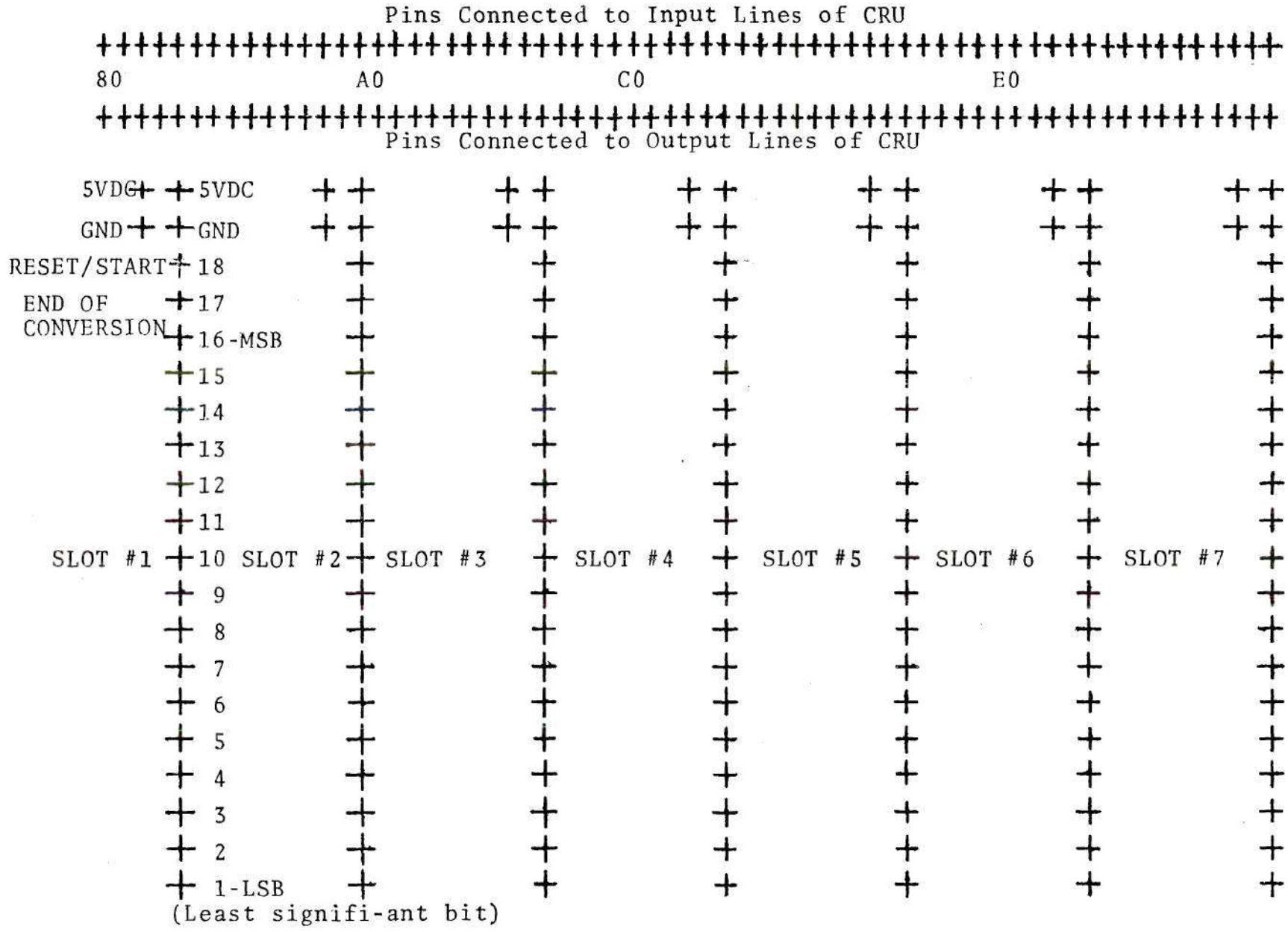


Figure 5-7. Wire-Wrap Board Pin Convention

jumpers on the wire wrap board to any part of the CRU he desires. The connection for this particular setup are shown in Figure 5-8. These assignments are consistent with commands in the subroutine RED. It should also be noted that since sixteen bits are reserved for binary signals, the user may, at a later date, use converters with finer resolutions if that proves necessary.

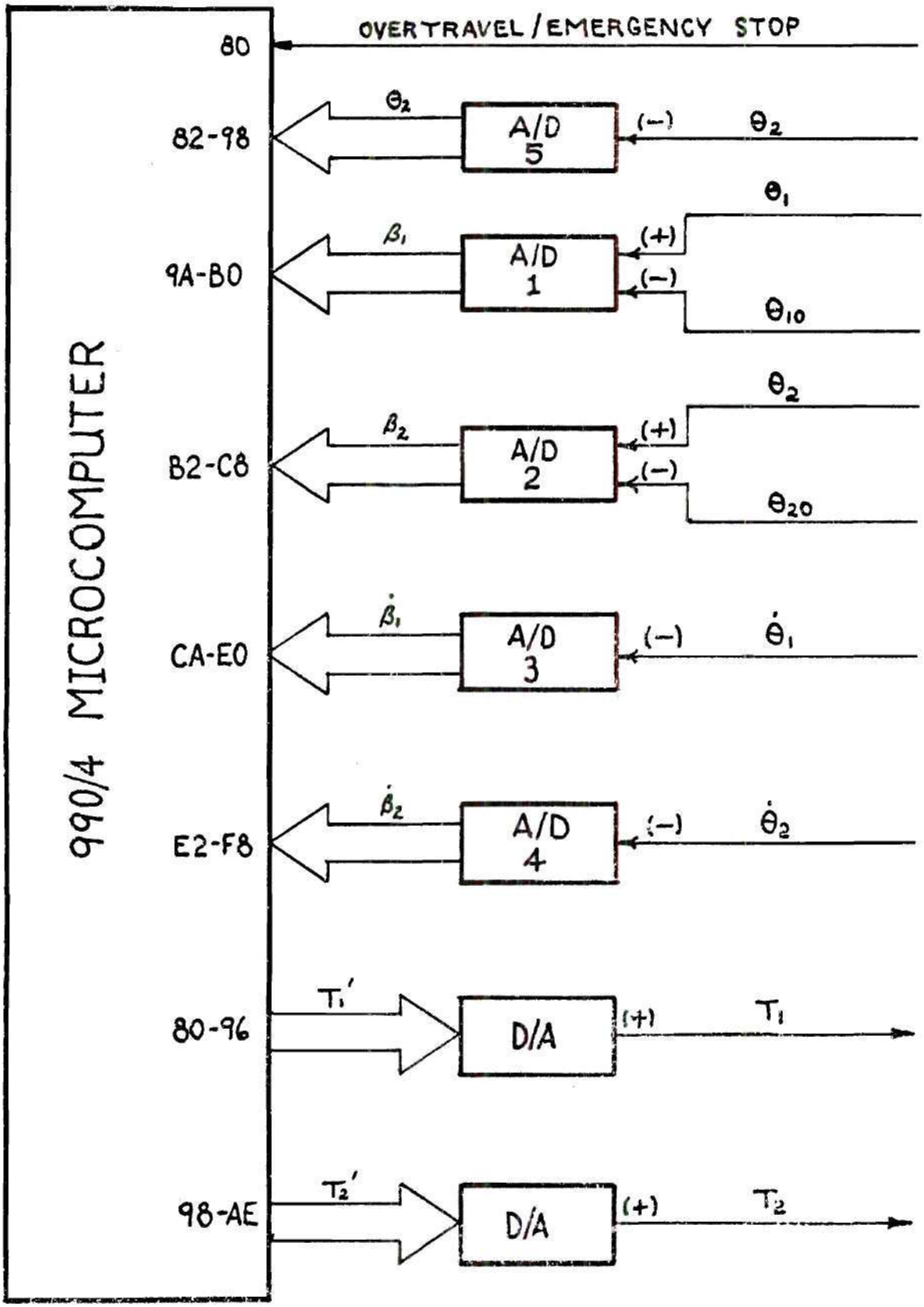


Figure 5-8. Computer Interface and CRU Address Assignments

CHAPTER VI

CONCLUSIONS AND RECOMMENDATIONS

A difficult portion of this thesis was the development of arm design specifications. A systematic approach relating the manipulator structural variables to task variables does not really exist except for the single link case. The problem lies, first, in the lack of a known set of task variables pertinent to the design of a manipulator with several degrees of freedom, then, in the lack of an approach relating those variables to structural design variables.

The development of a general design approach and a system of defining task parameters would probably not alone be sufficient. The process of using such an approach would be involved enough, certainly for manipulators with four or more degrees of freedom, to require a great deal of computer aid. This would appear to be an area with great potential for interactive graphics type programs. Computer simulations enabling sophisticated modeling of joints and actuators would also be valuable.

Parts of the control program could probably be rewritten to reduce computing time. The reductions, however, would represent a relatively small percentage of the total computing time. Whether the reduction would be worth the

effort is an open question. This question may be answered with the writing of subroutines modeling the effects to be studied in research. Excessive time penalties might require any savings that can be found elsewhere. Ultimately, one can visualize a new control program, using the present program as a basis that includes all the effects to be studied as integral part of the computations rather than as subroutines to be tacked on. With the simulations always available and a longer list of programmable constants, such a program's flexibility would be very convenient to the experimenter.

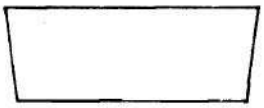
APPENDICES

APPENDIX I

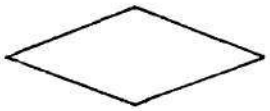
COMPUTER PROGRAM FLOWCHART

Flowchart Symbols

Operations



Decisions usually associated with Compare Immediate or Test Bit instructions in the program



Subroutine calls accompanied by explanation of purpose of the call

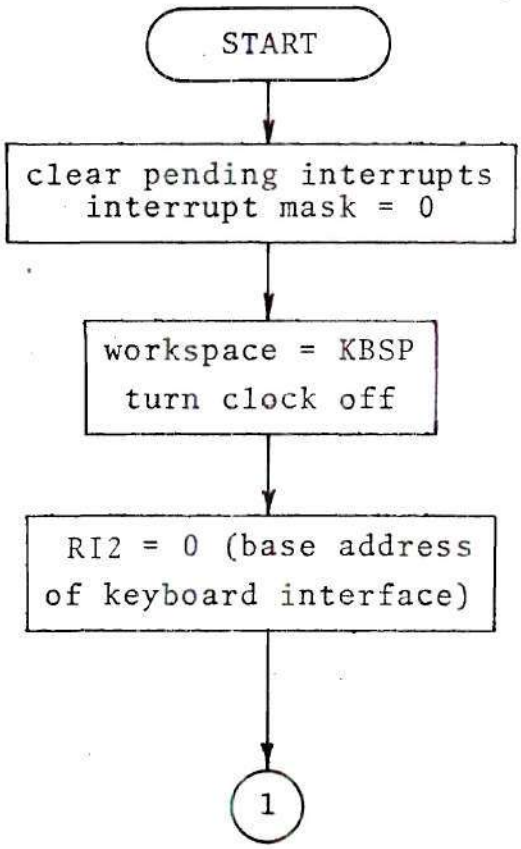
Underlined alphanumeric symbols are the symbolic addresses in the program corresponding to the labeled steps in the flowchart. RTWP is the mnemonic for "return via workspace pointer."

Numbers preceded by ">" are hexadecimal numbers. All other numbers are decimal numbers.

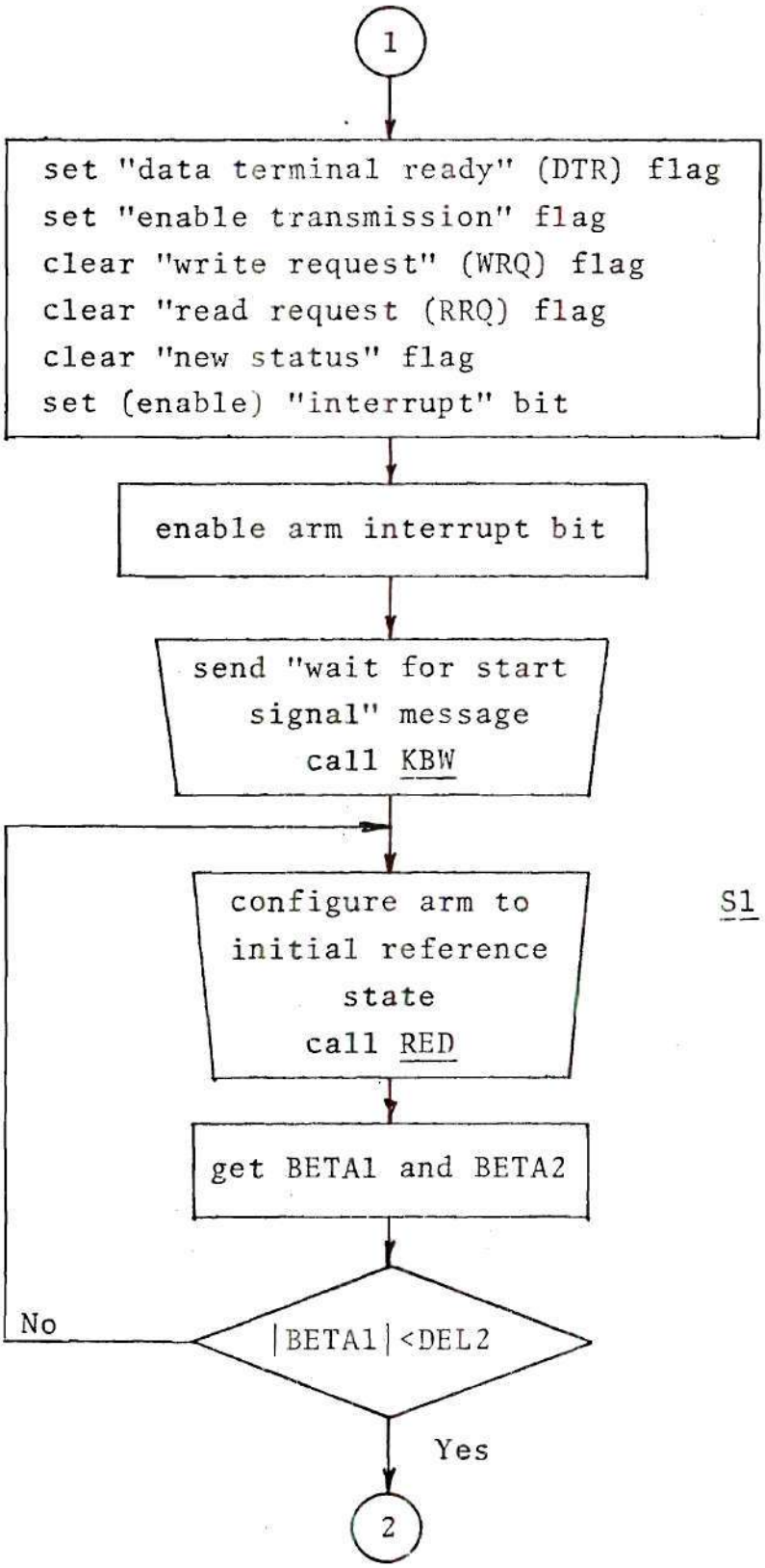
INIT

This is the "main" program. It initializes the keyboard interface then positions the arm for startup. When initialization has been completed, the computer is put into IDLE.

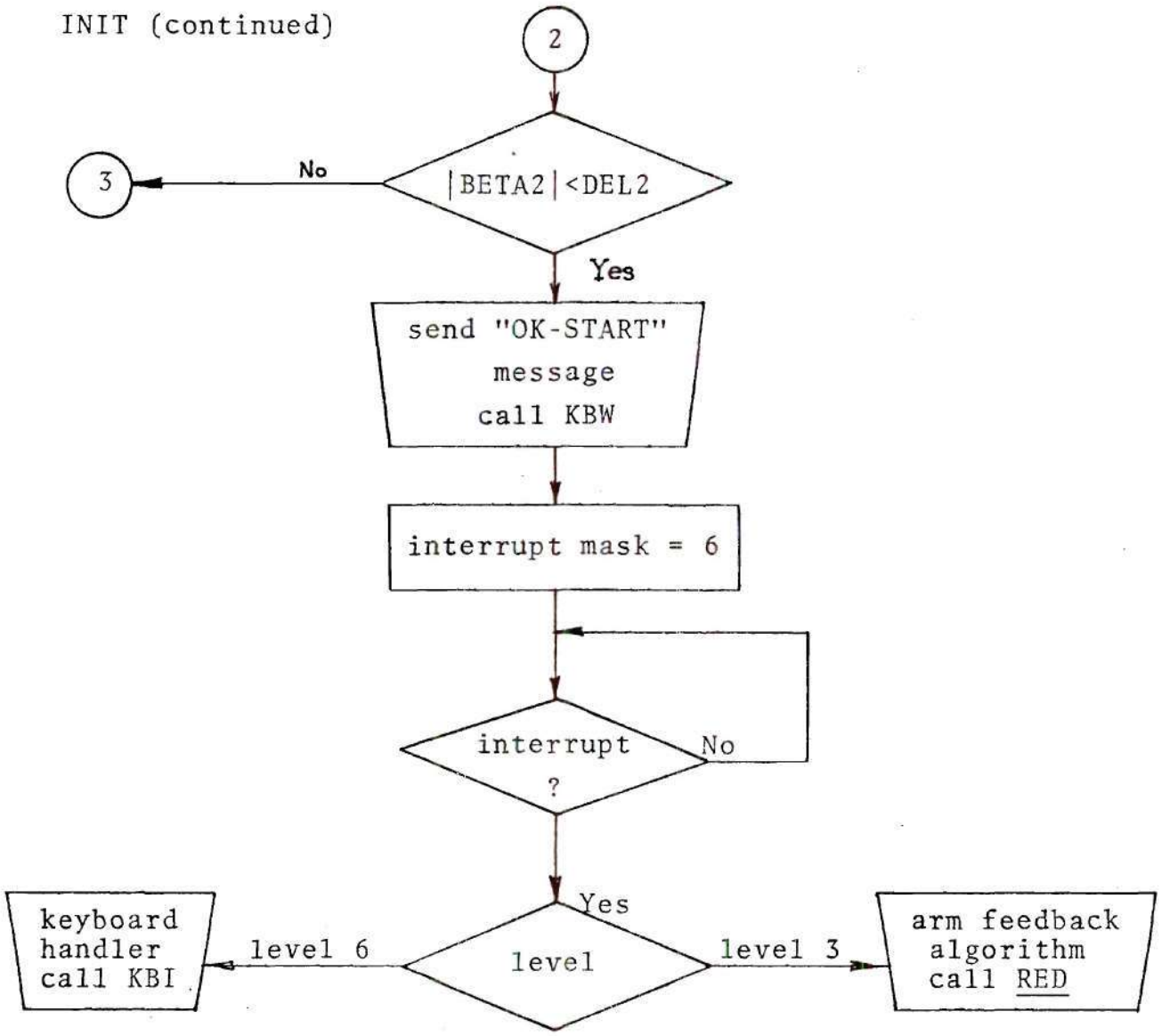
STRT



INIT (continued)



INIT (continued)



KBI

KBI is the keyboard interrupt handler. It decodes the user's commands then executes them. Command list:

A: change A/D angle scale factor (rad/part)

E: end program

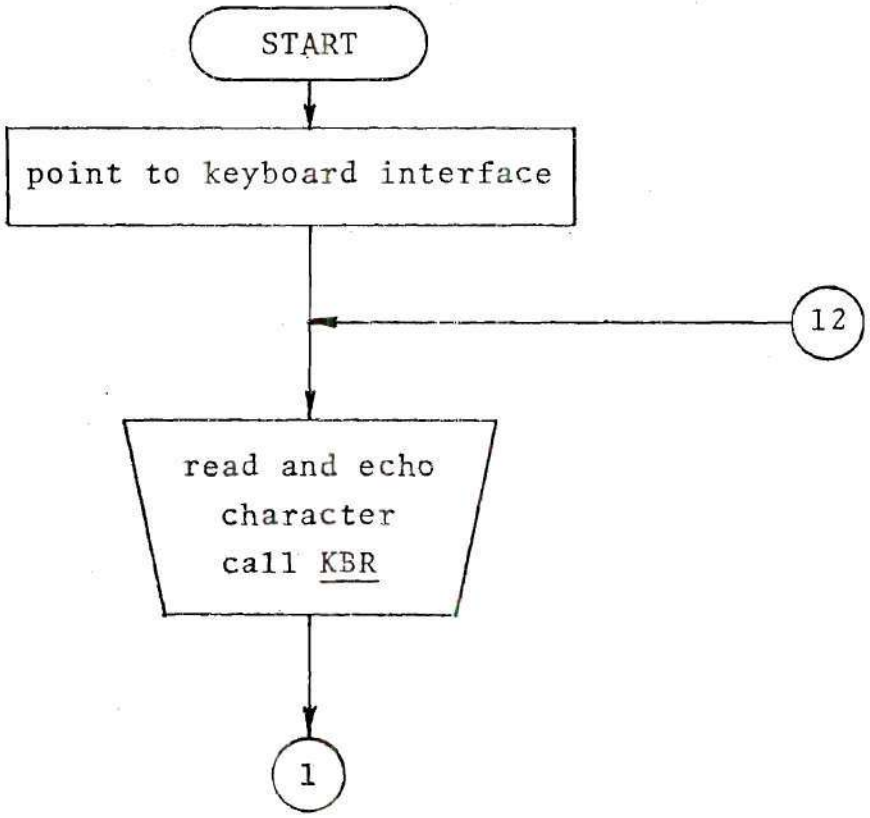
G: mistake (will return to arm control)

P: change parameters; allows user to change eigenvalues and damping ratio

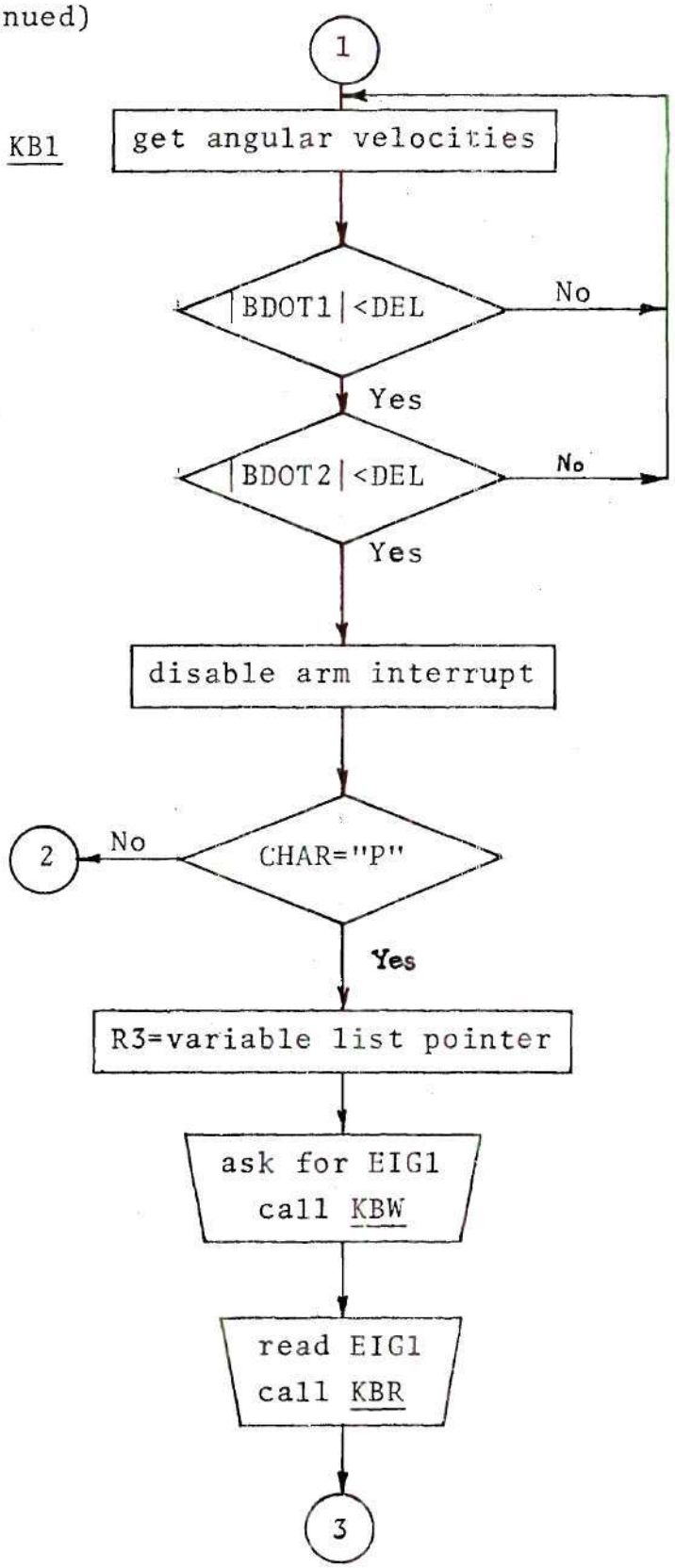
T: change D/A torque motor scale factors (parts/ft-lb_f)

V: change A/D angular velocity scale factor (rad/sec/part)

KBI



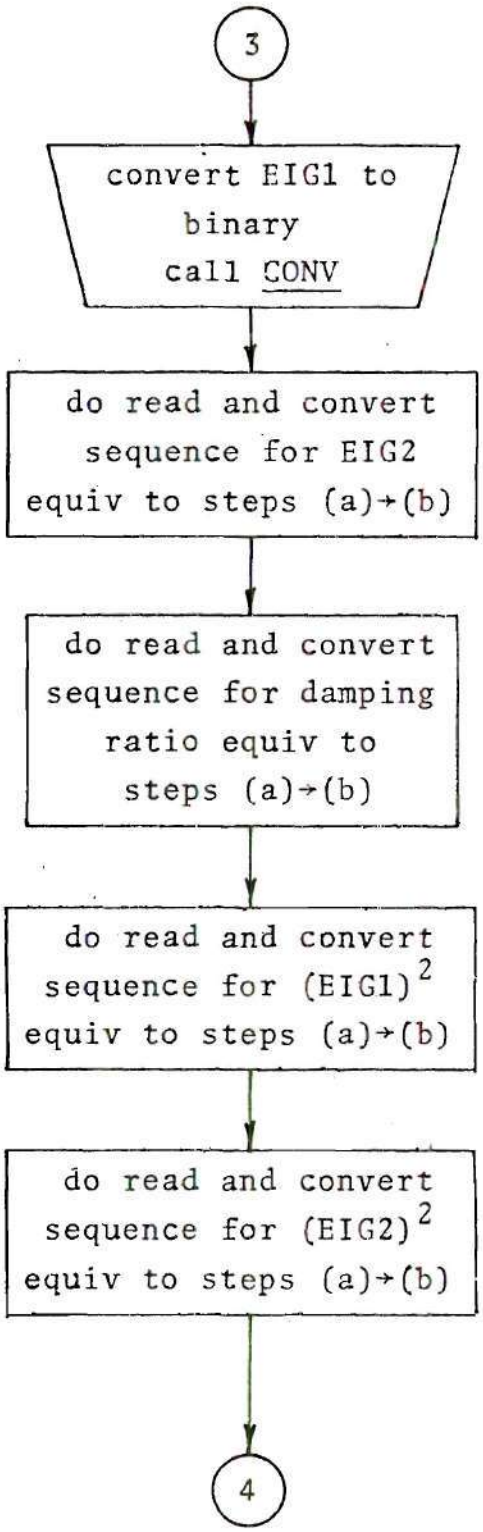
KBI (continued)



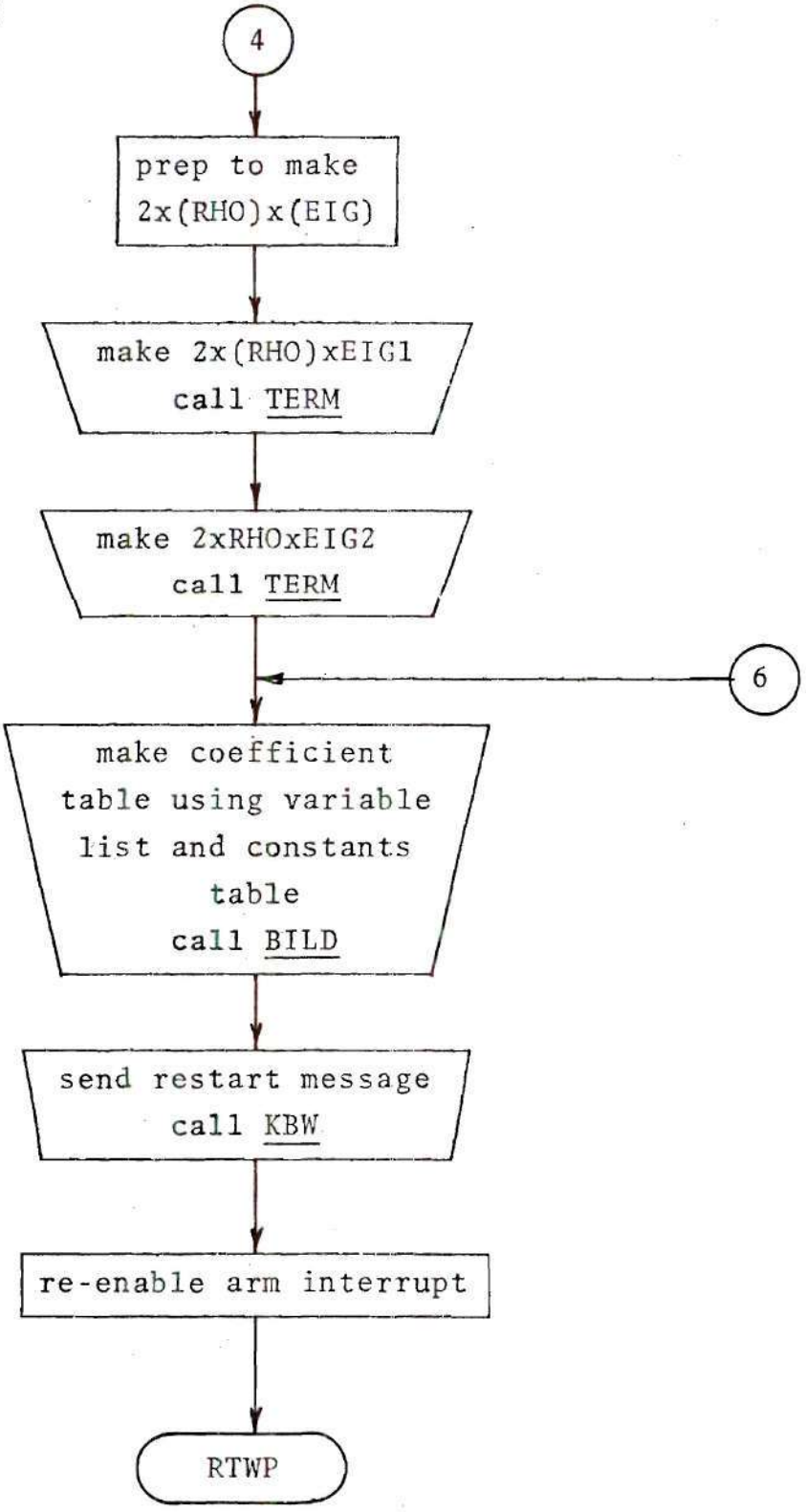
(a)

KBI (continued)

(b)

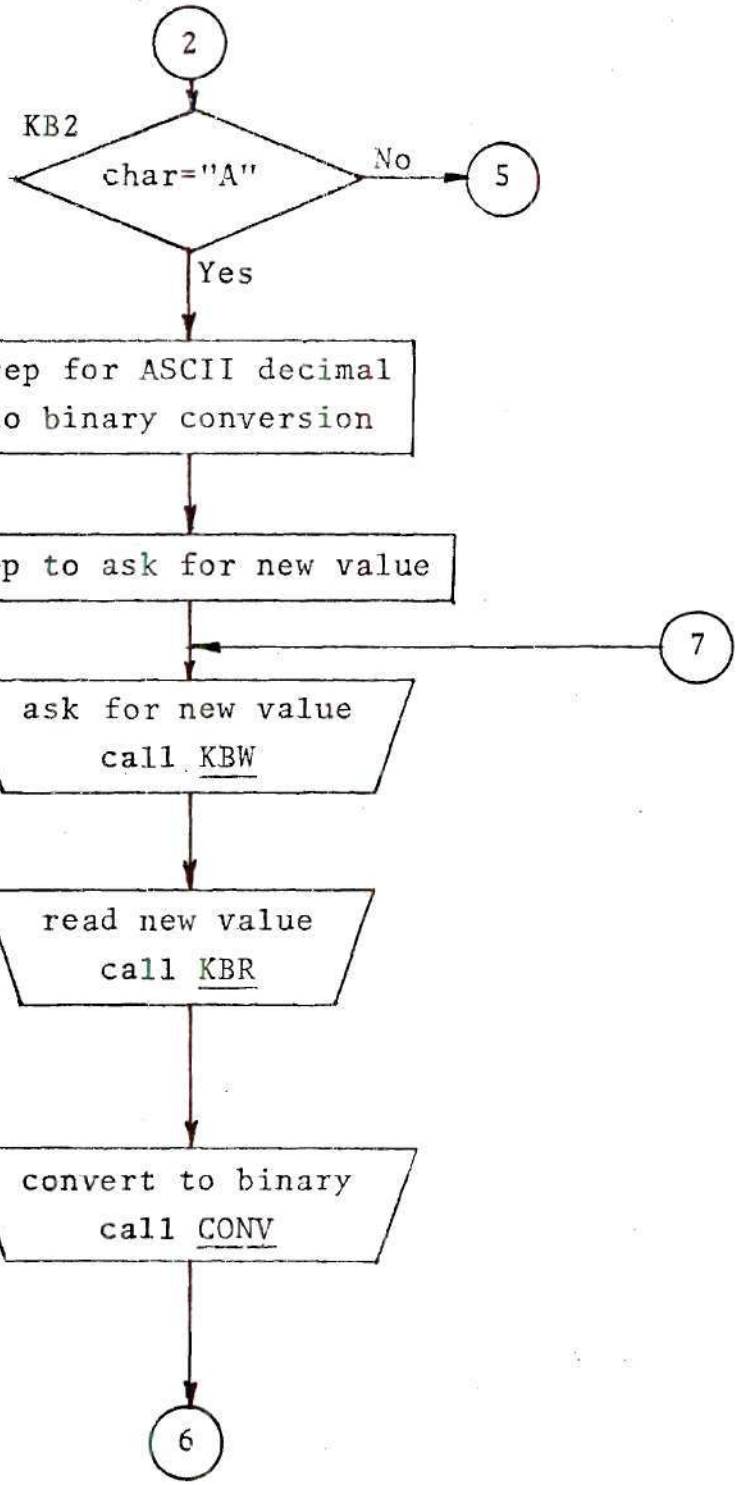


KBI (continued)

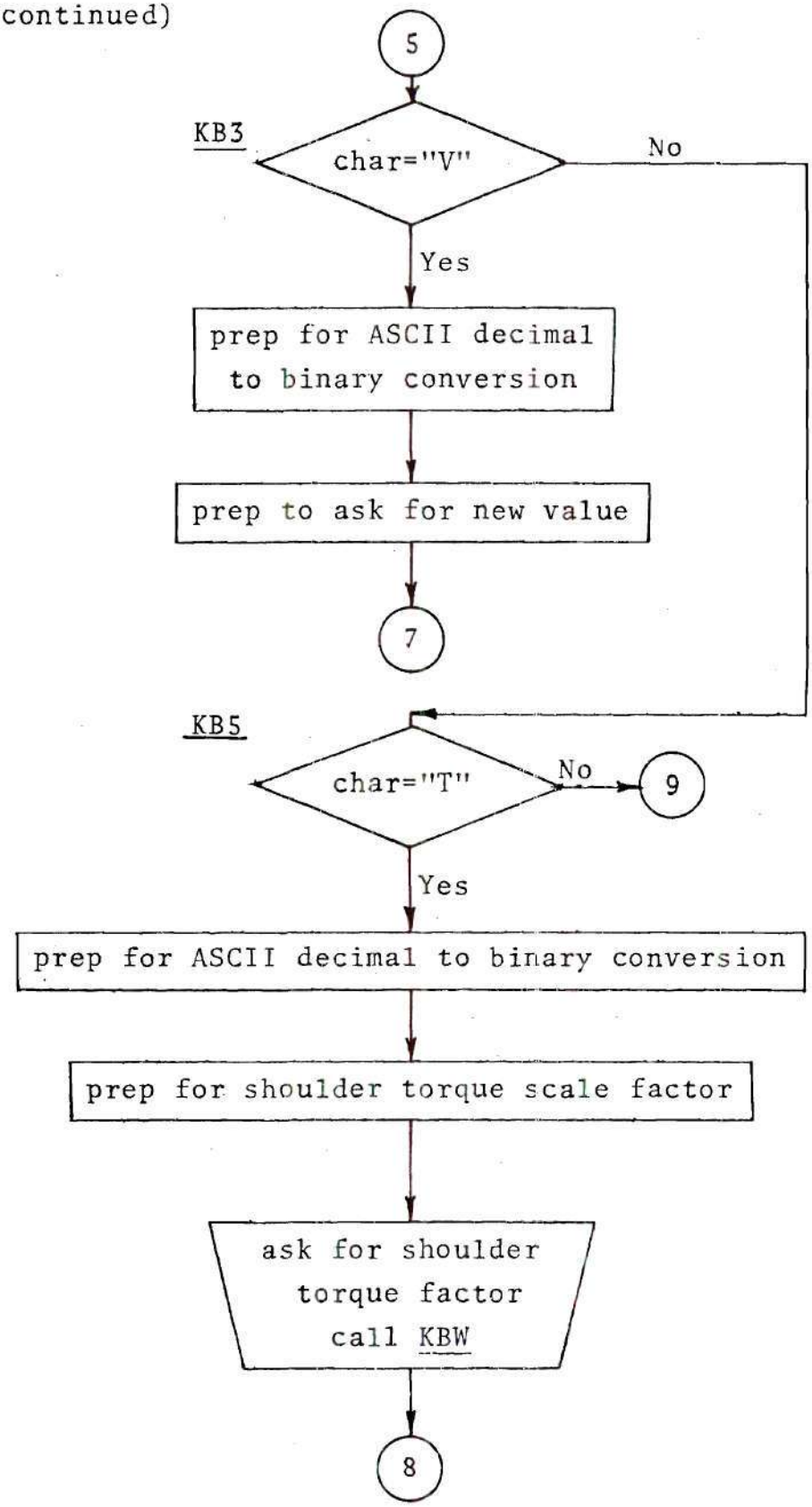


KB7

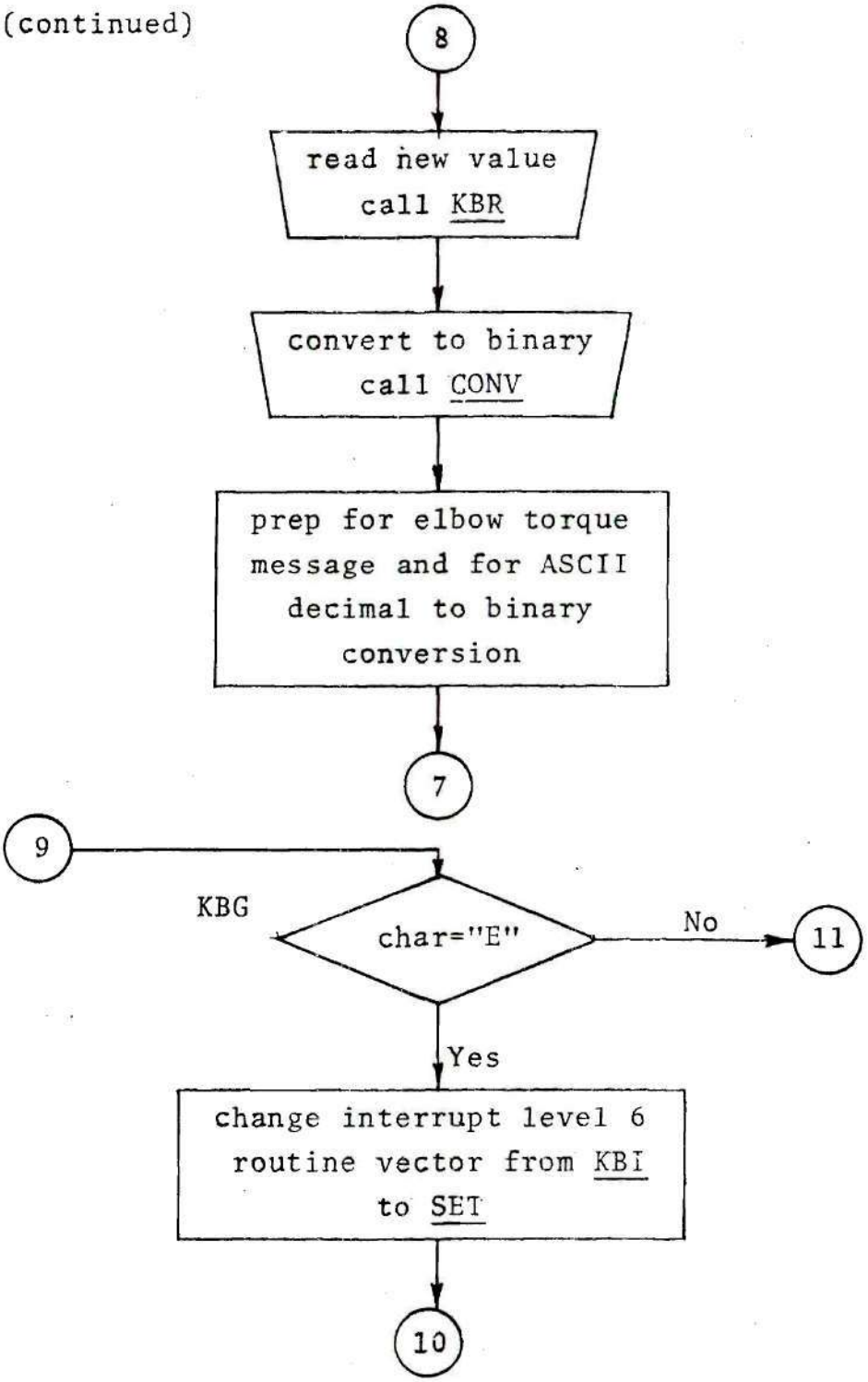
KBI (continued)



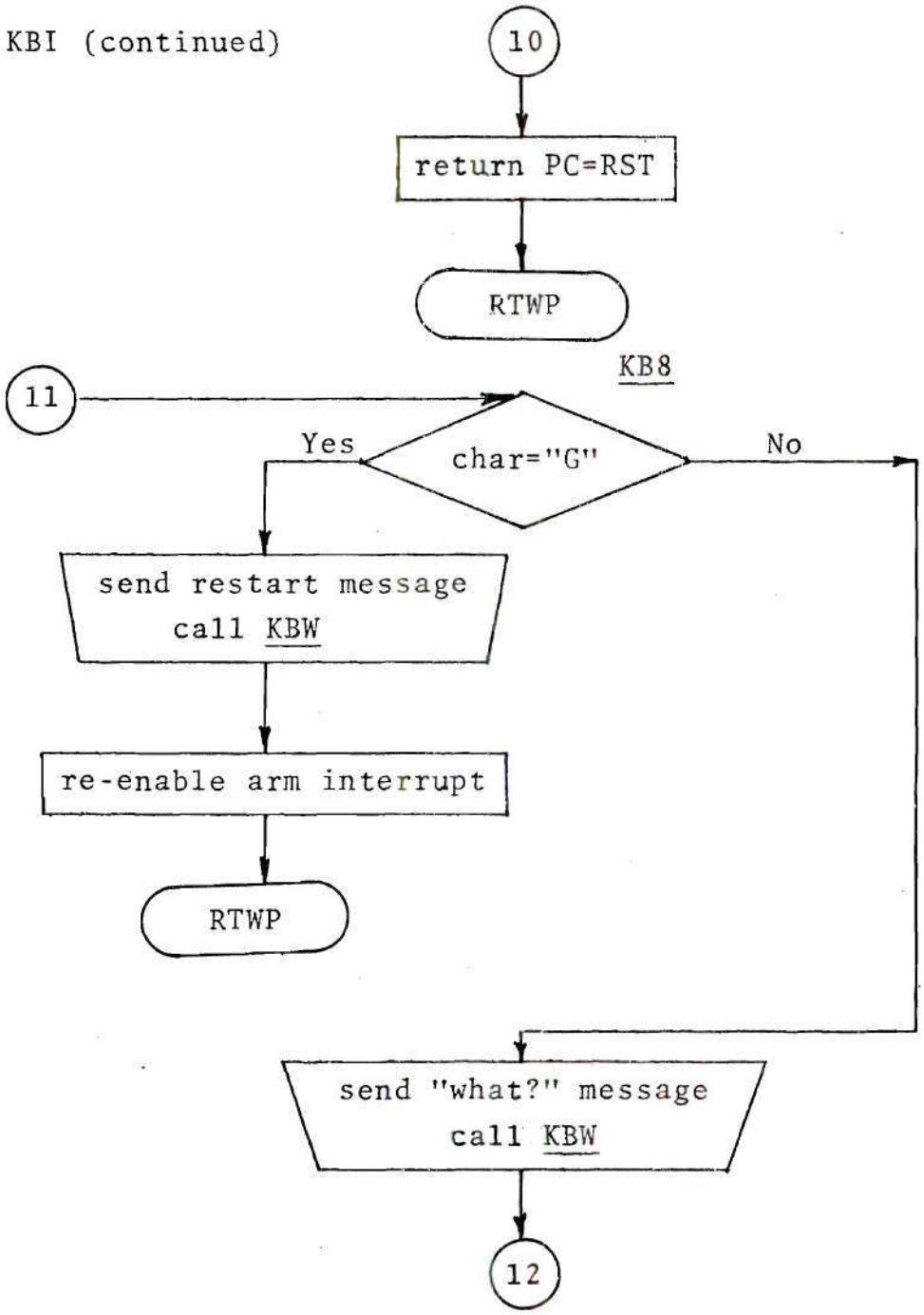
KBI (continued)



KBI (continued)

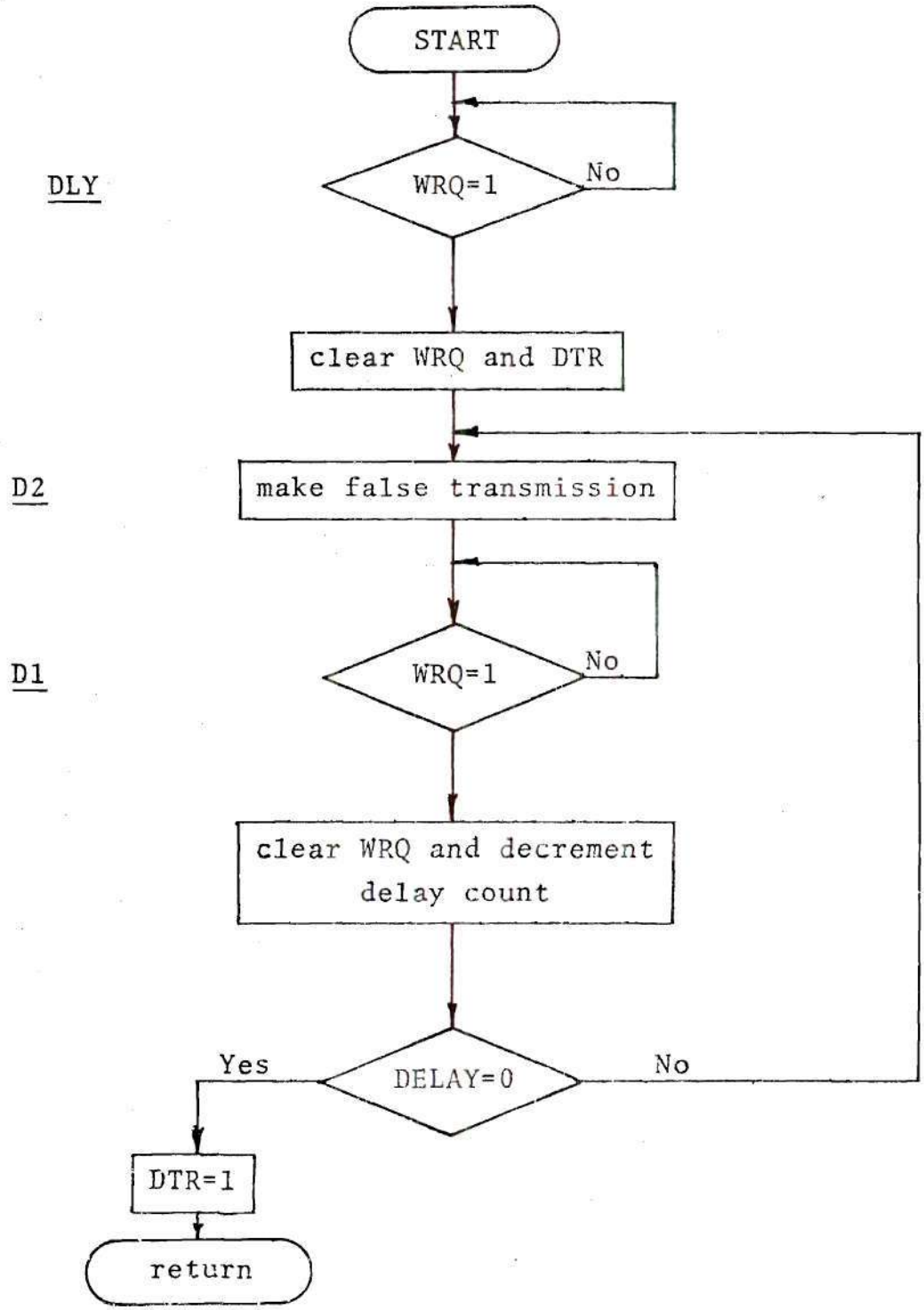


KBI (continued)



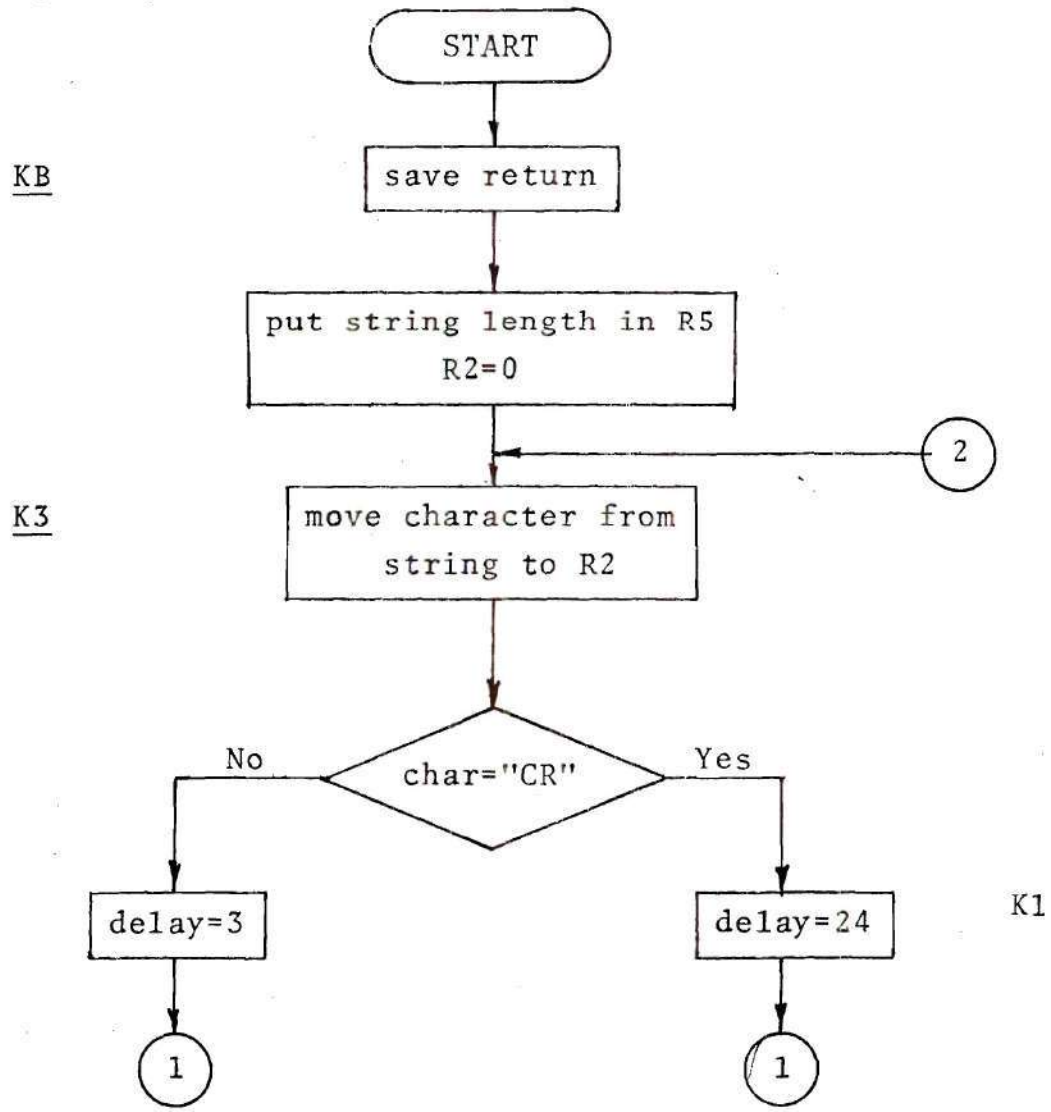
DLY

This routine installs transmission delays in writing to make the 1200 baud interface compatible with the 300 baud keyboard.



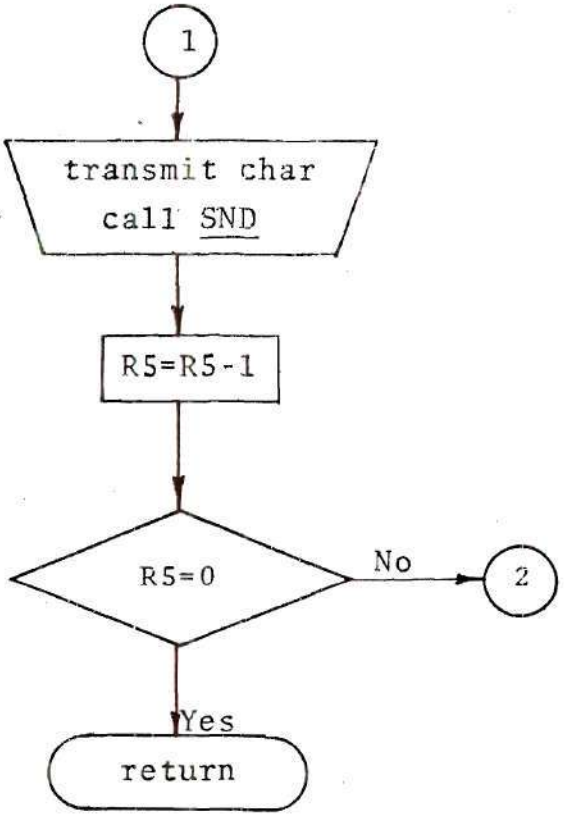
KBW

KBW prints character strings for the program. It assumes that R4 contains a pointer to the string and that the first word on the string contains the length of the string in bytes.



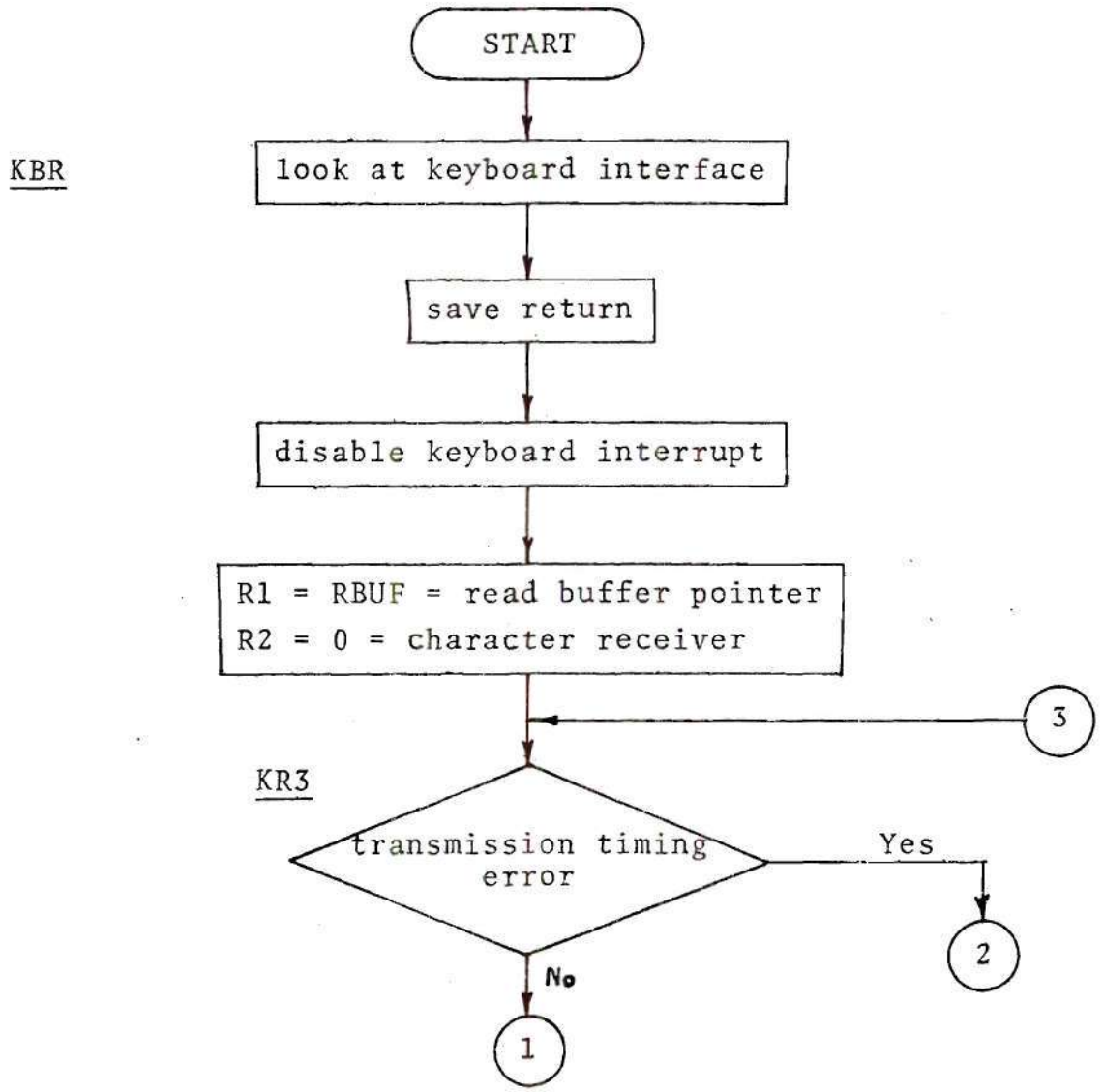
KBW (continued)

K2

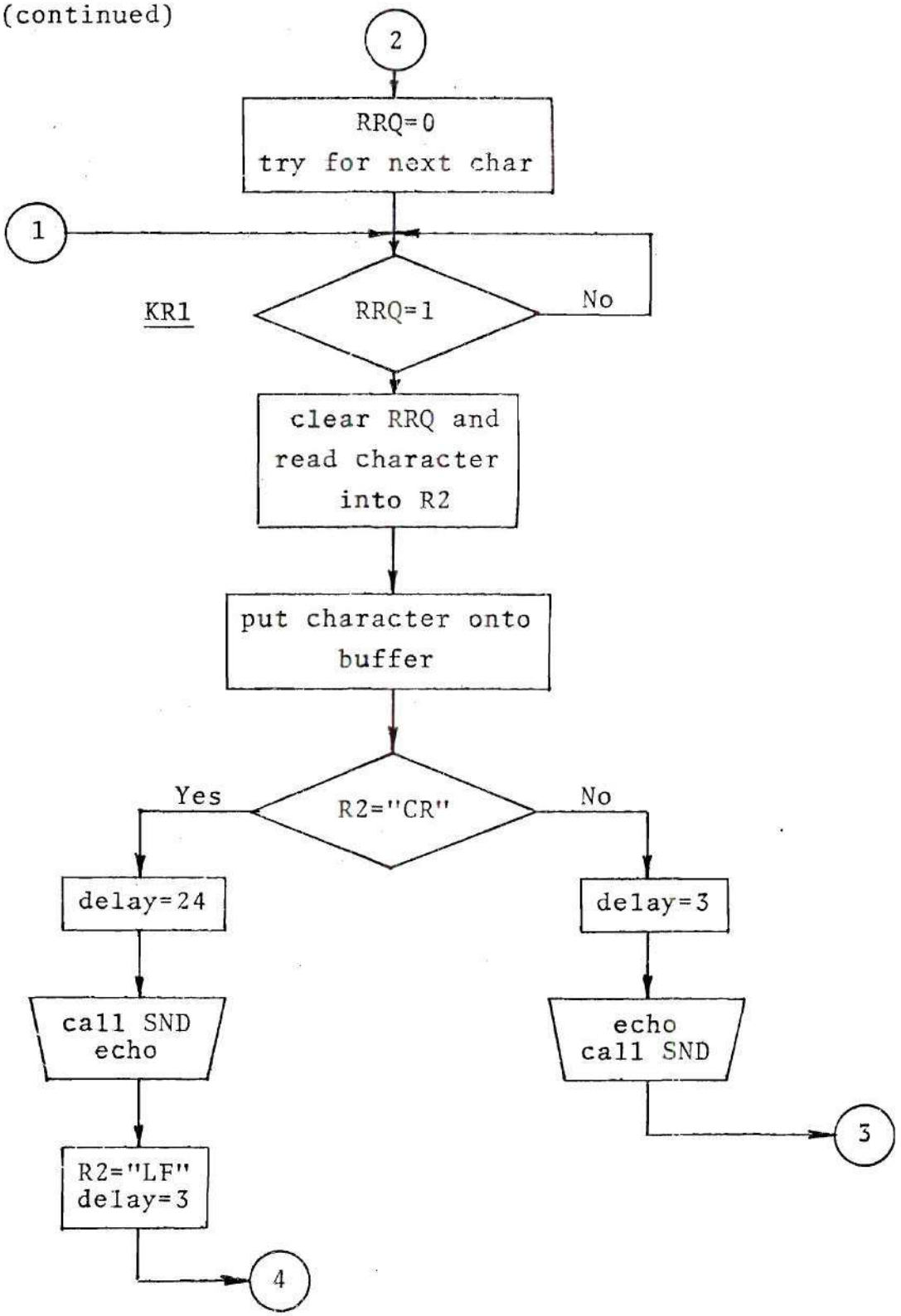


KBR

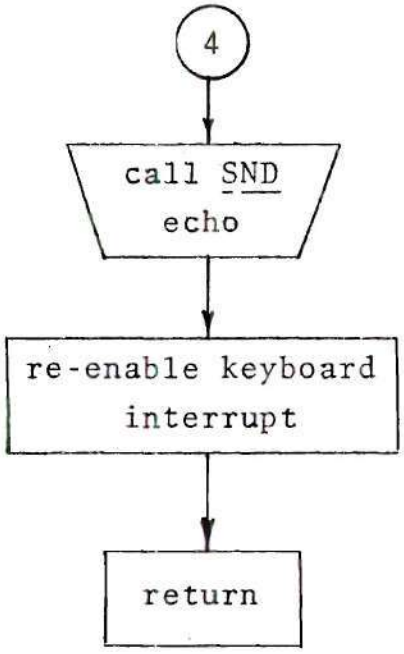
KBR reads user input. Once called by KBI it continues to read until it receives a carriage return, CR. The CR is put on the read buffer as an end-of-line symbol and a CR LF is sent to the data terminal



KBR (continued)



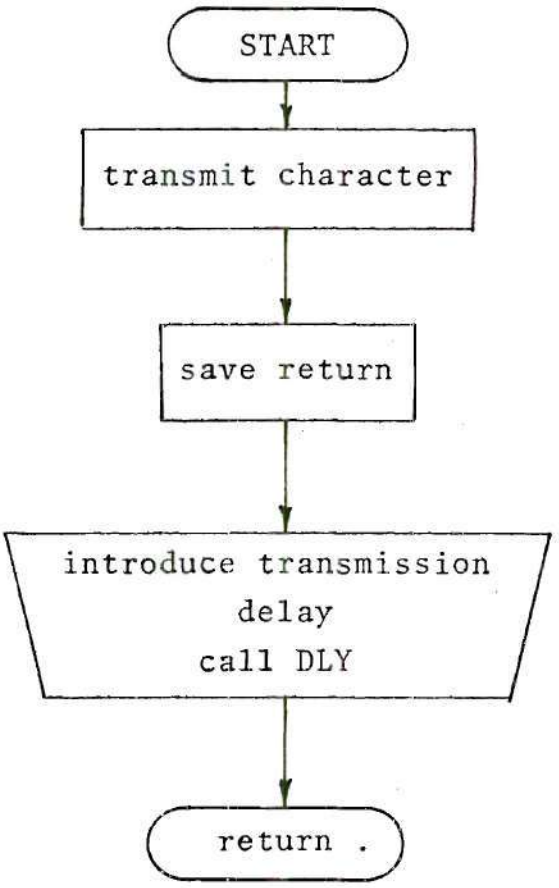
KBR (continued)



SND

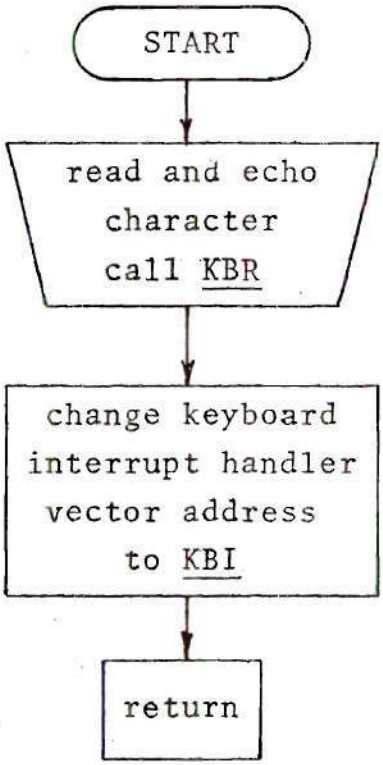
SND transmits one character through the interface and calls DLY to insert the proper delay count in transmission. The delay count is provided by the routine that calls SND.

SND



SET

This routine executes a program restart after an end program command has been executed. Typing any key after the program has been stopped will restart the program.



CONV

This is an ASCII decimal to binary encoder.

R0 = binary return from DBIN

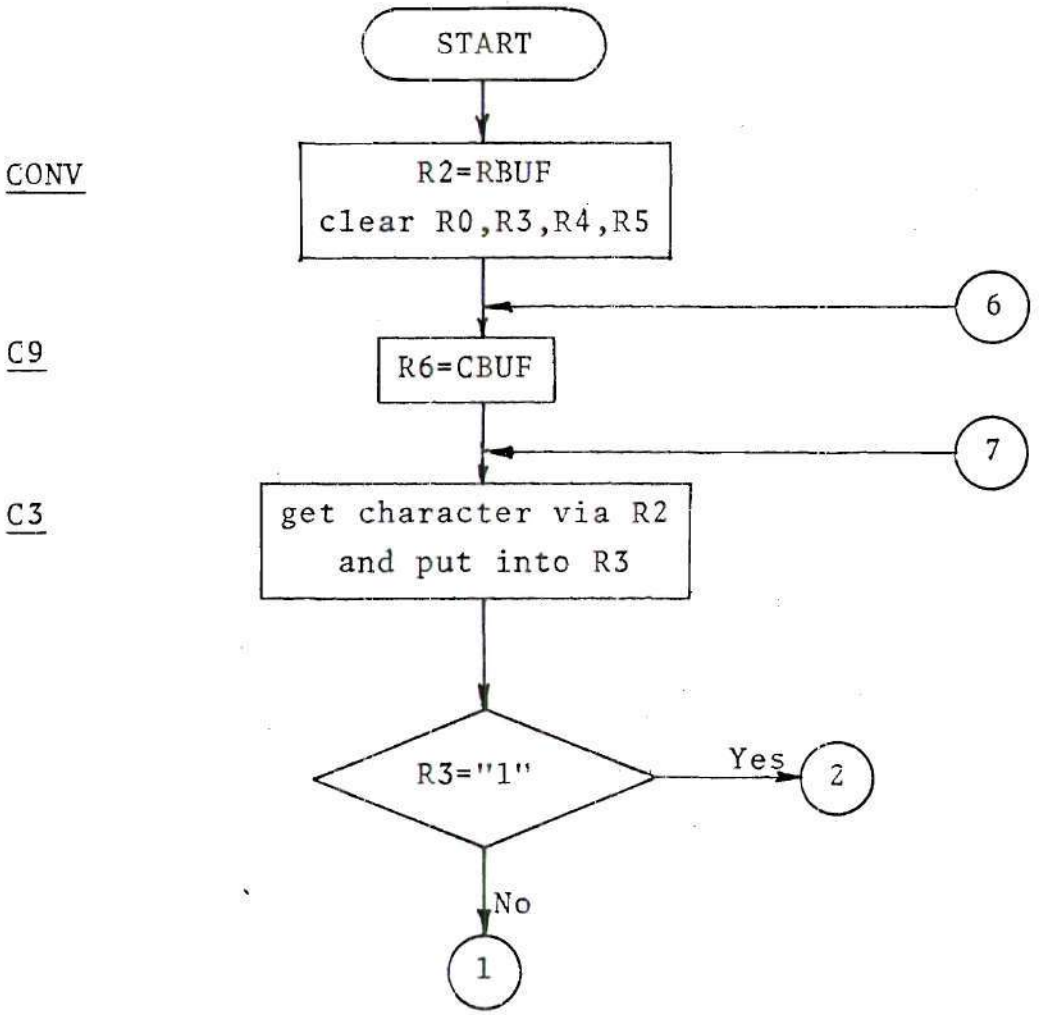
R2 = read buffer pointer

R3 = character fold

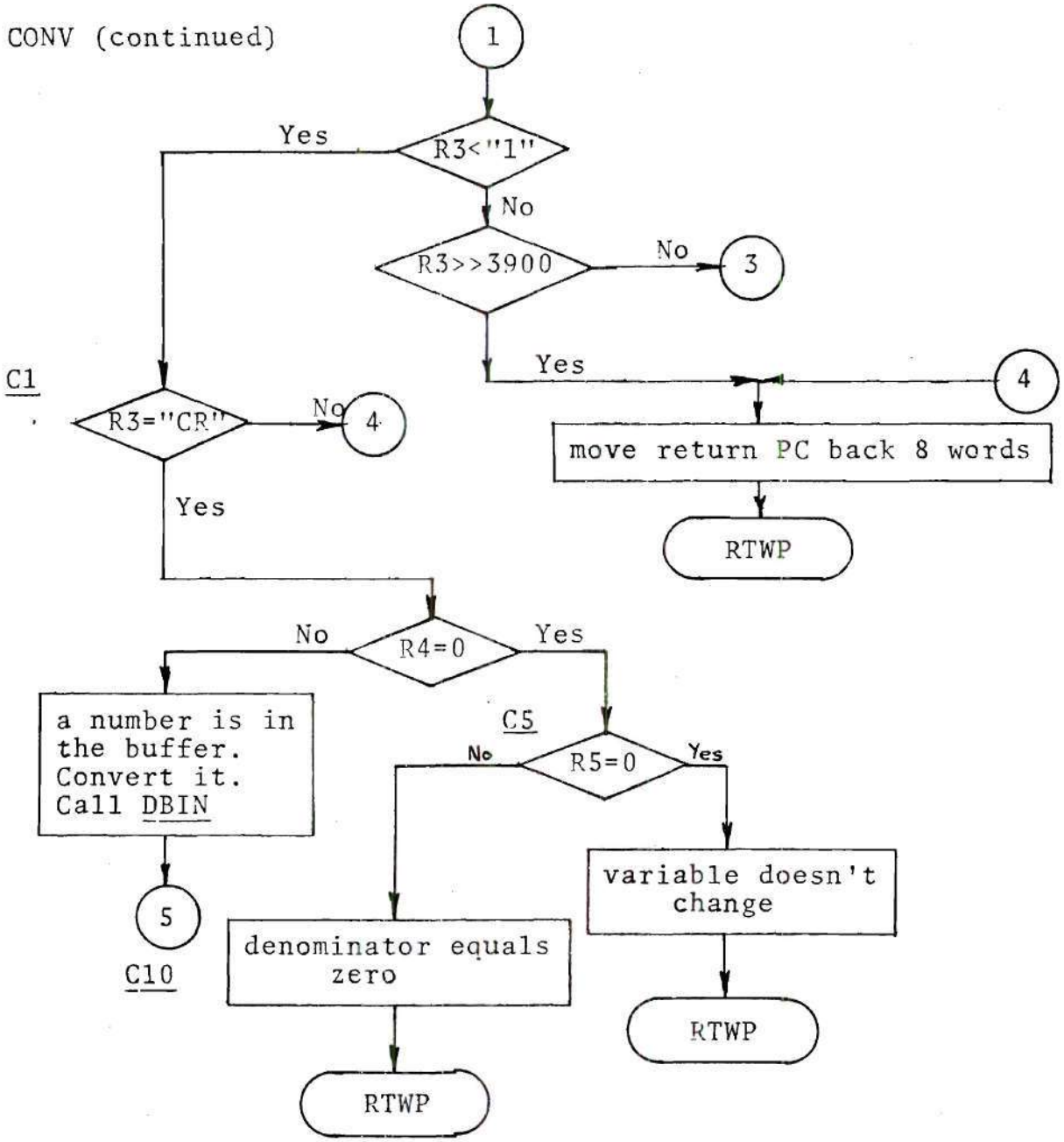
R4 = digit count

R5 = flag; num = 0, DENOM = 1

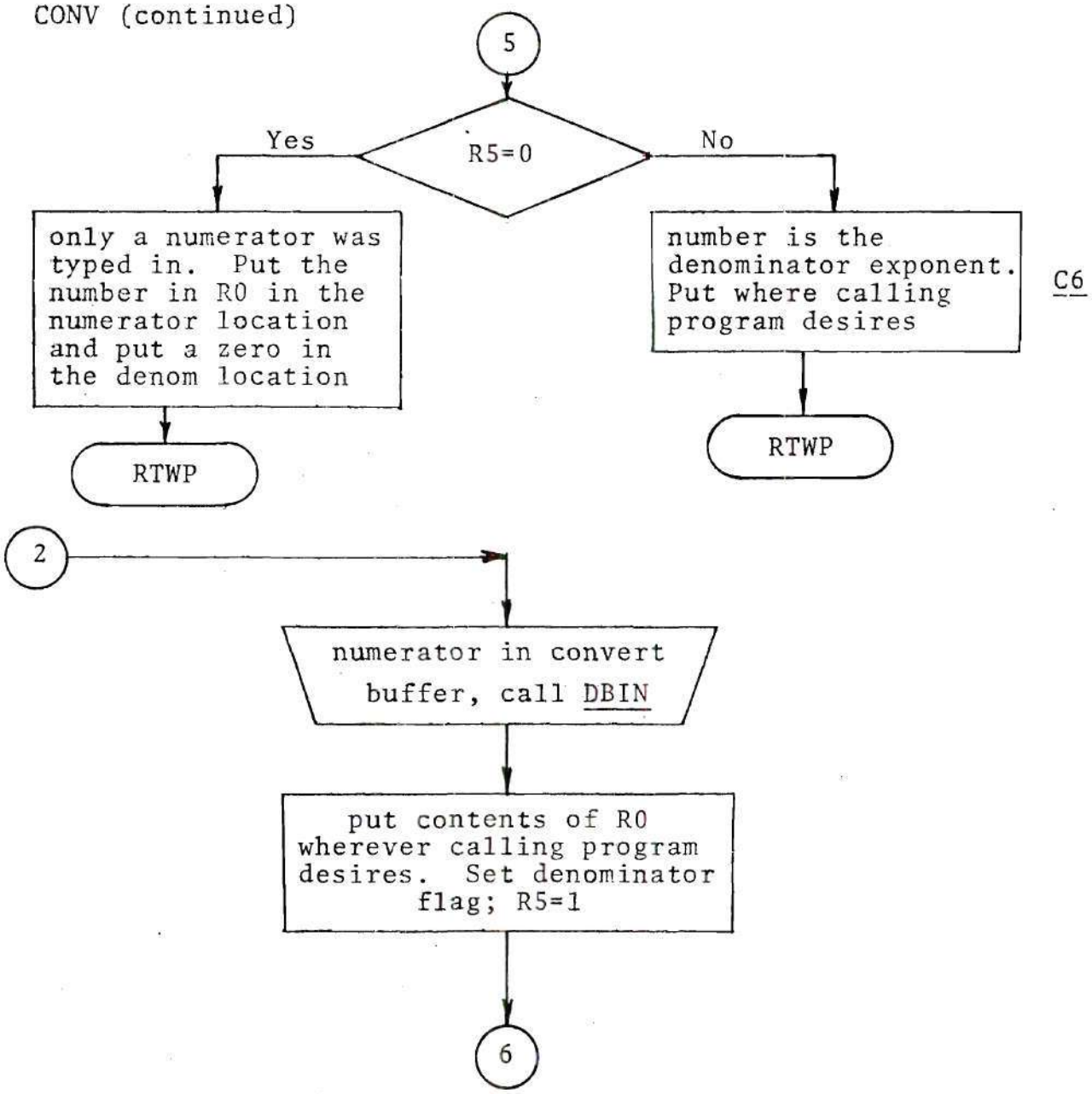
R6 = converter buffer pointer



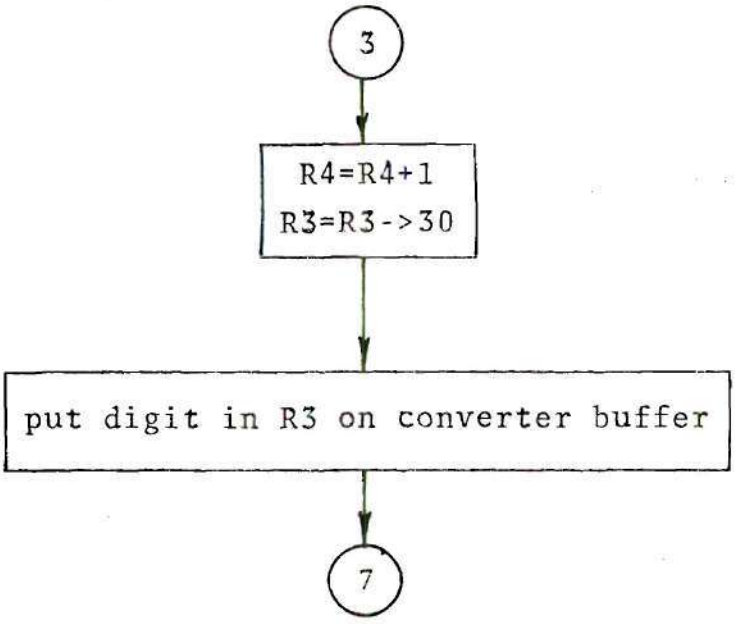
CONV (continued)



CONV (continued)



CONV (continued)



DBIN

This routine uses a series of hexadecimal digits in the converter buffer to build a binary number in R0.

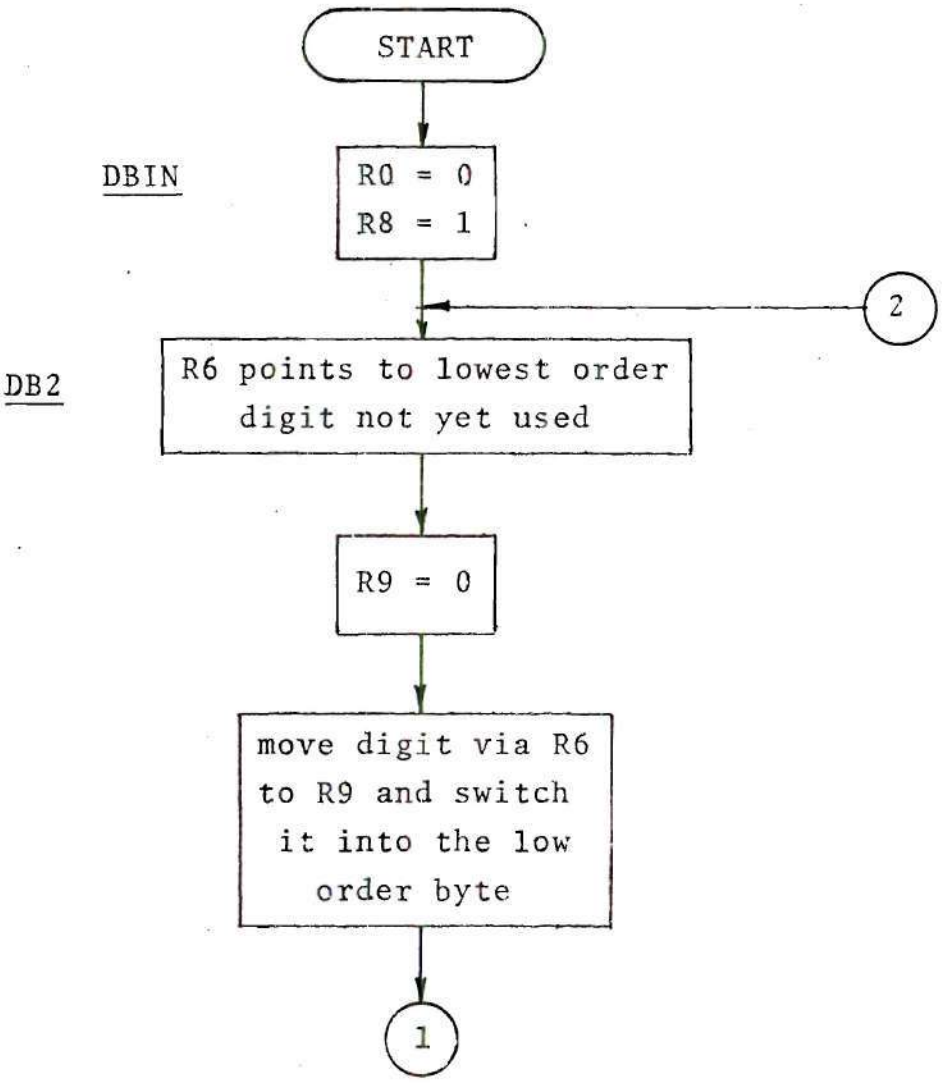
R0 = binary number

R4 = digit count

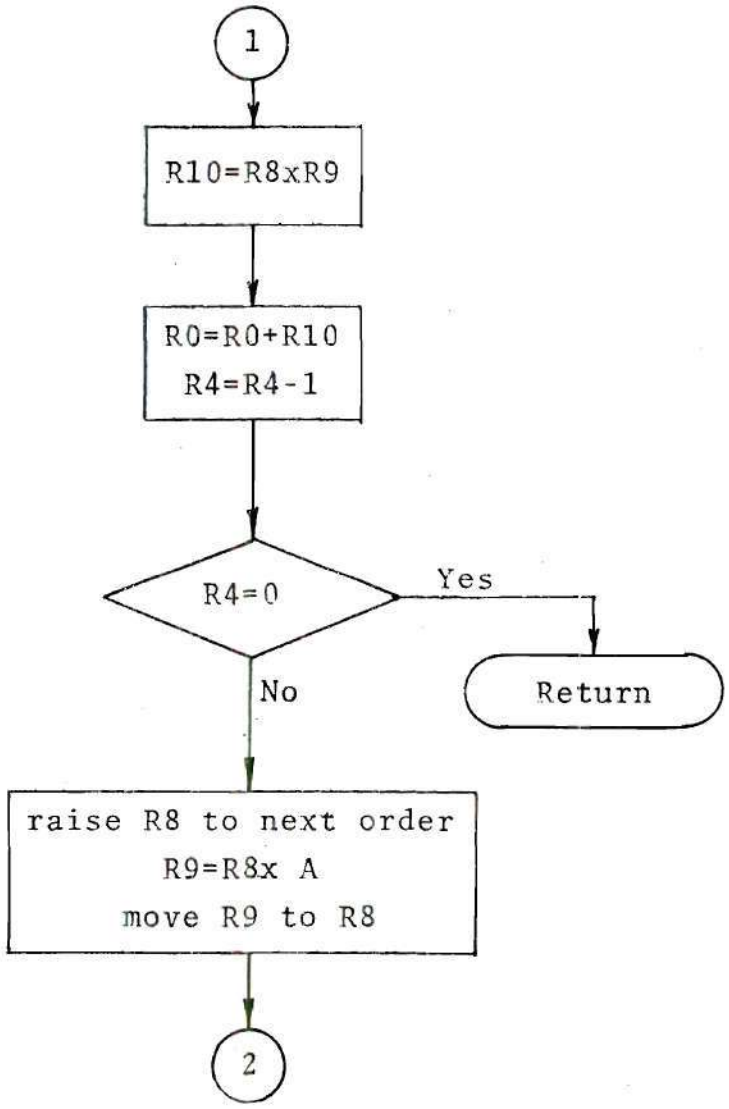
R6 = converter buffer pointer

R8 = digit order

R9 = digit register



DBIN (continued)



BILD

This subroutine combines the variable list with the constants table to make the coefficient table for the arm feedback routine.

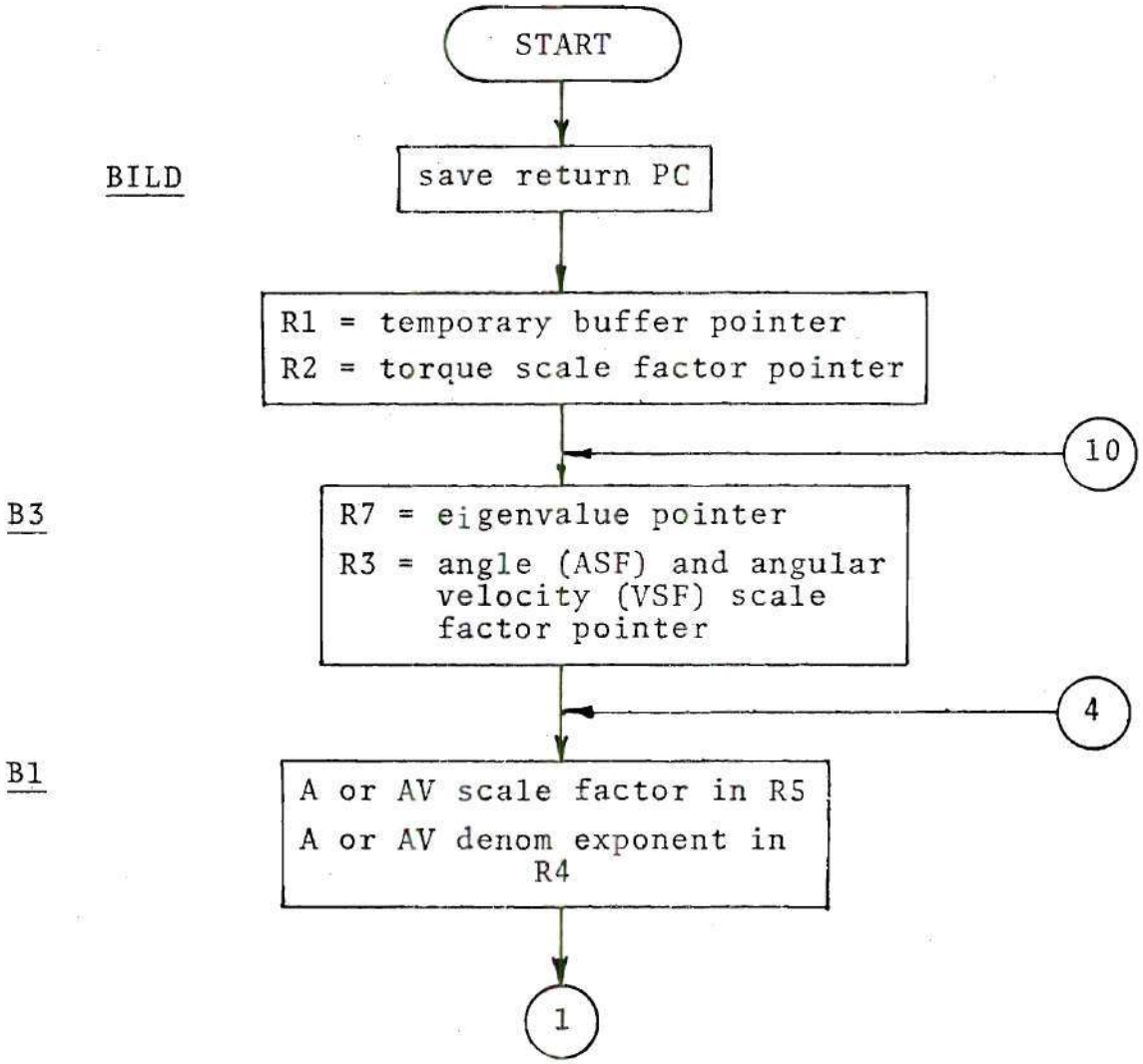


BILD (continued)

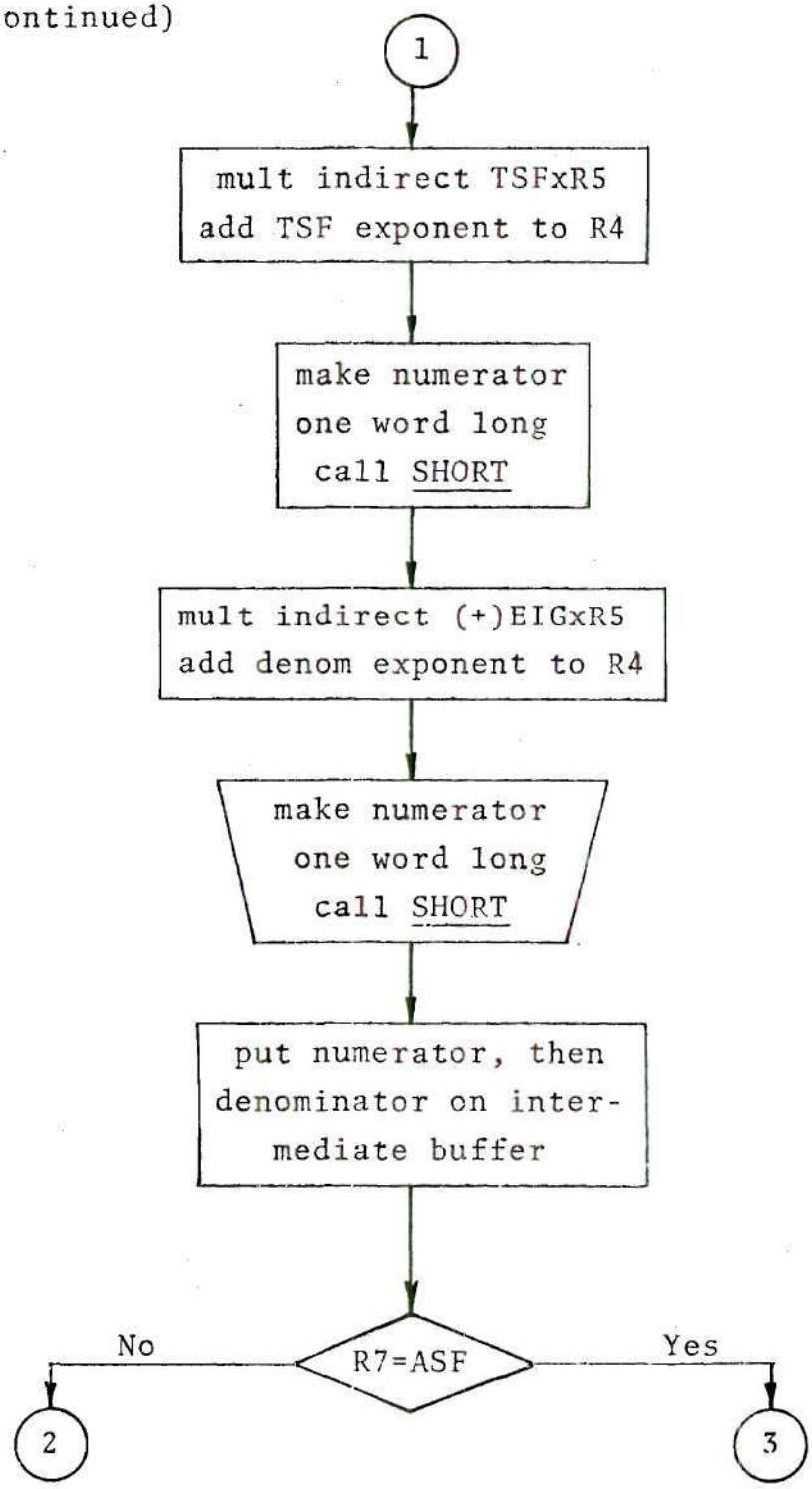


BILD (continued)

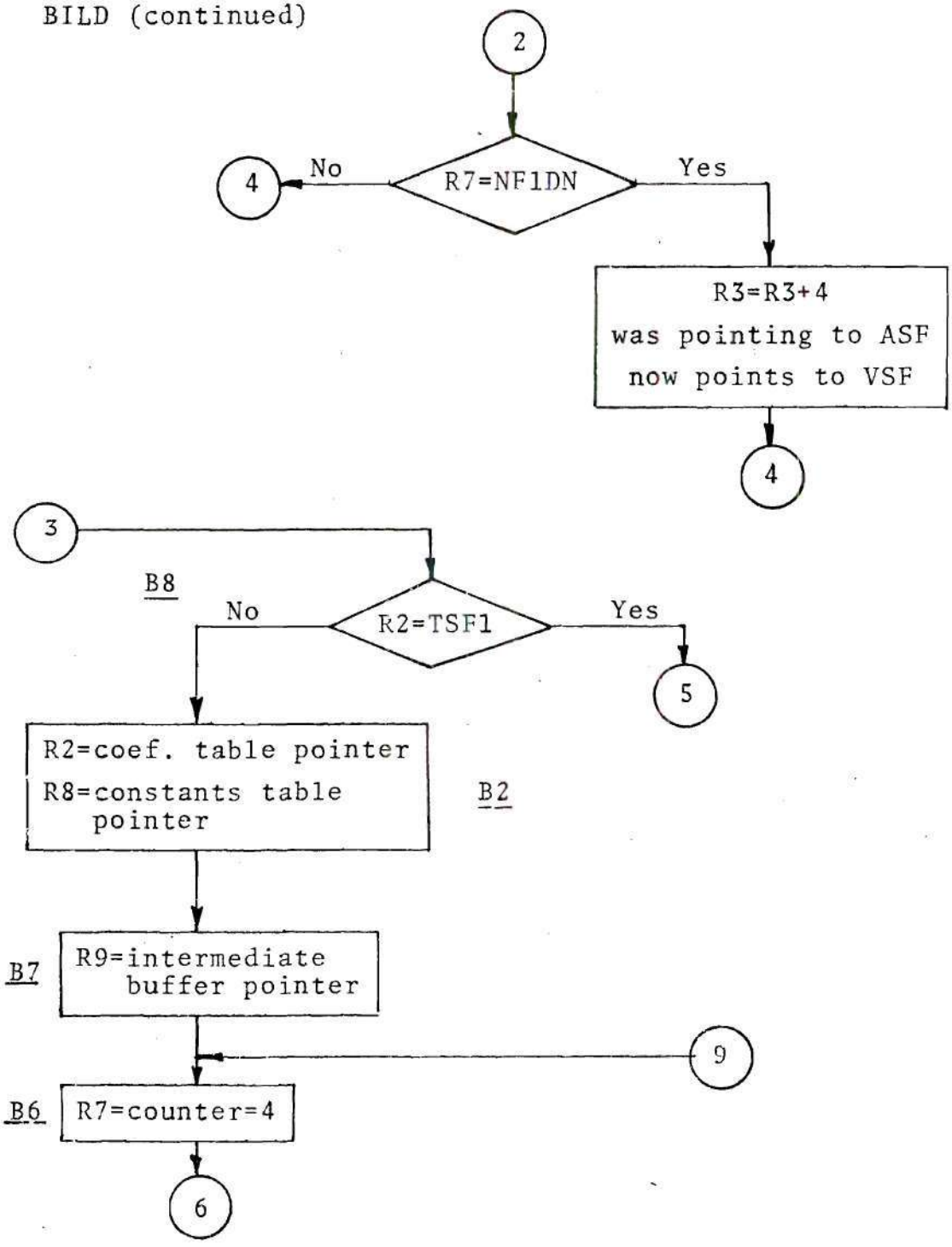


BILD (continued)

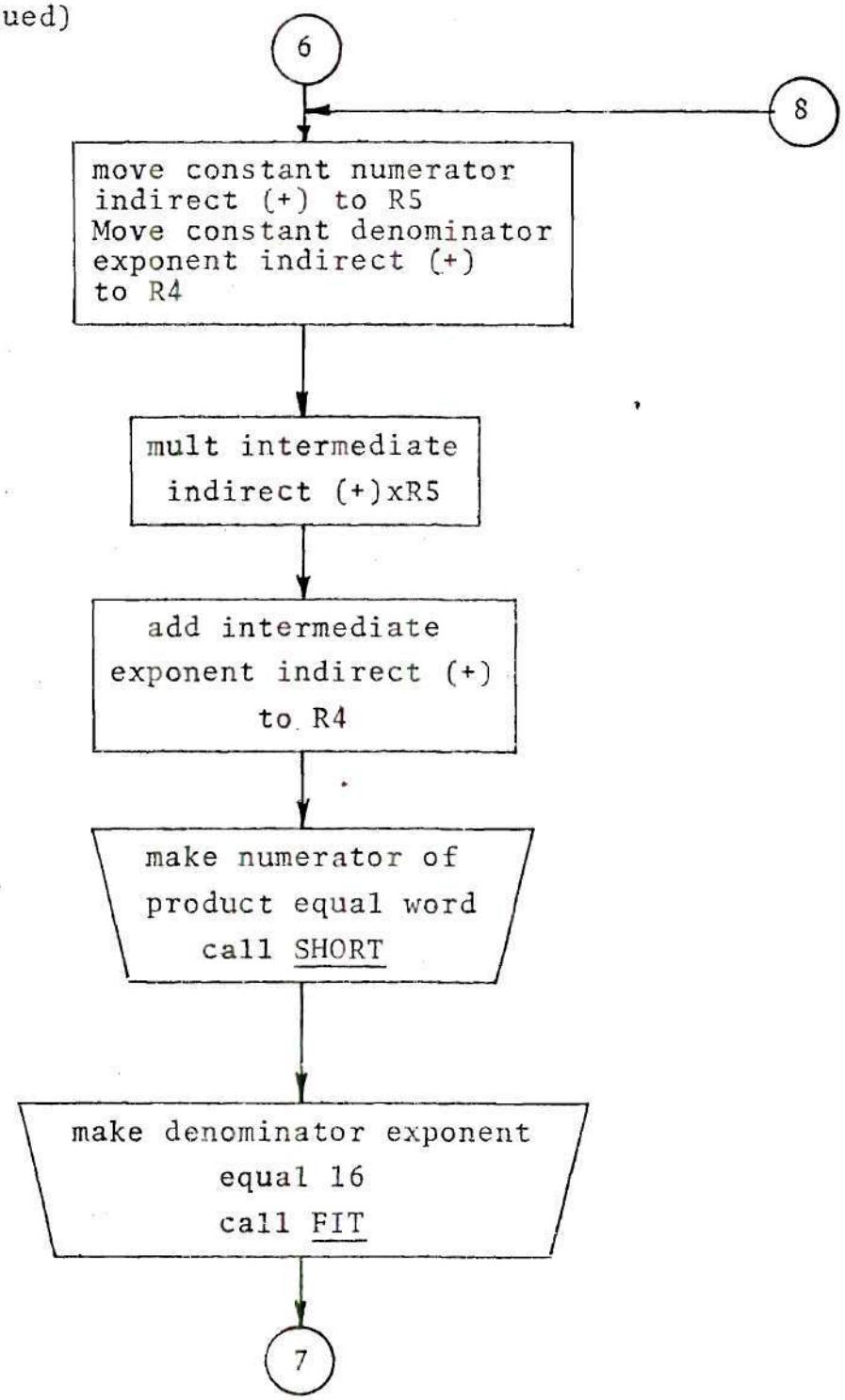
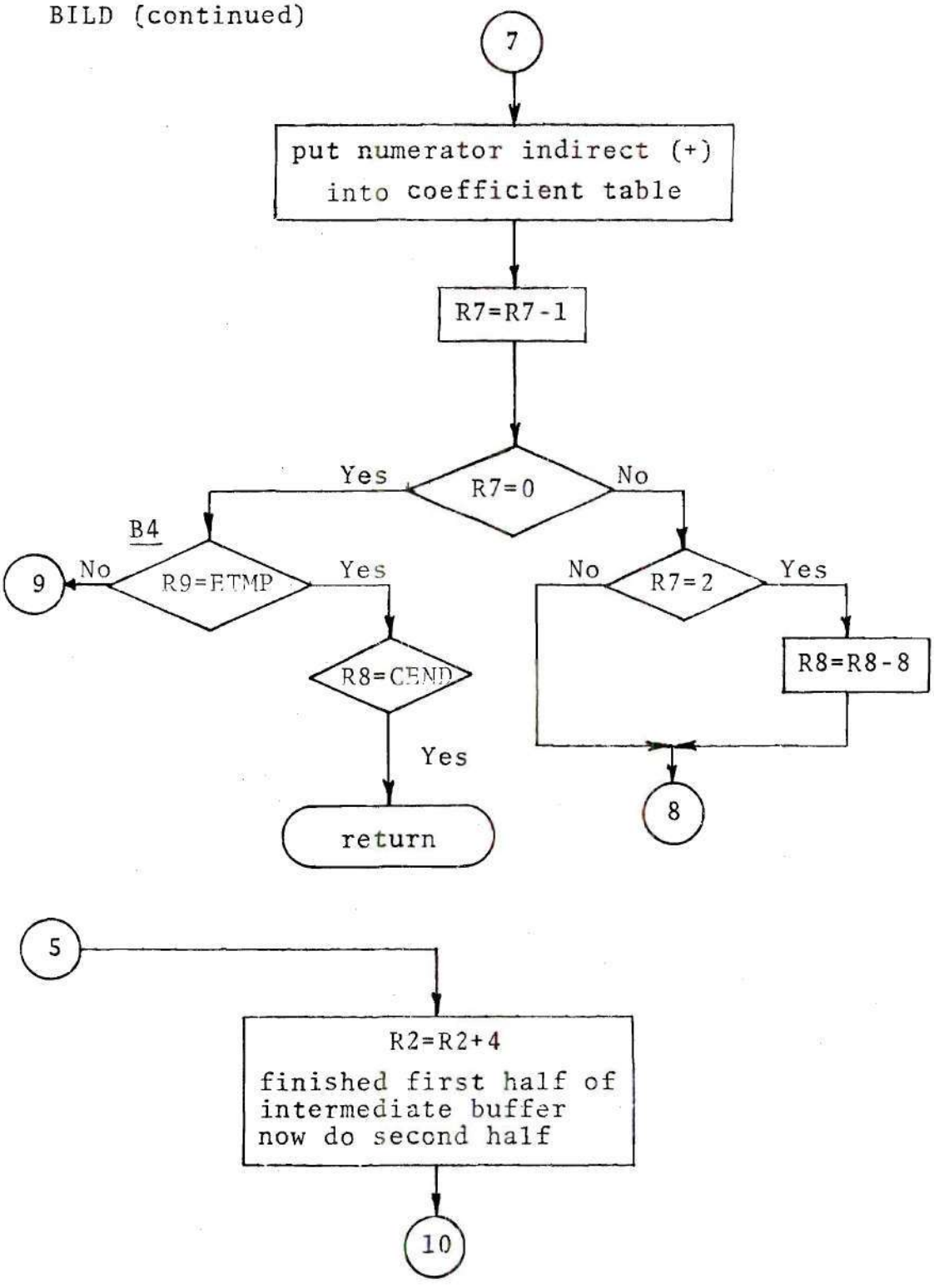
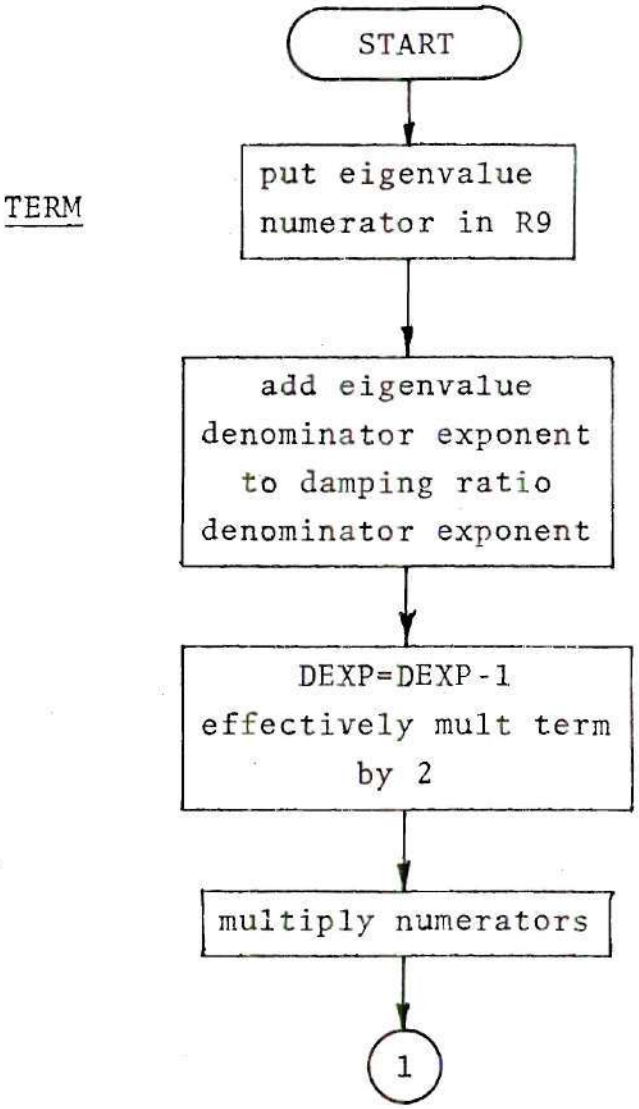


BILD (continued)

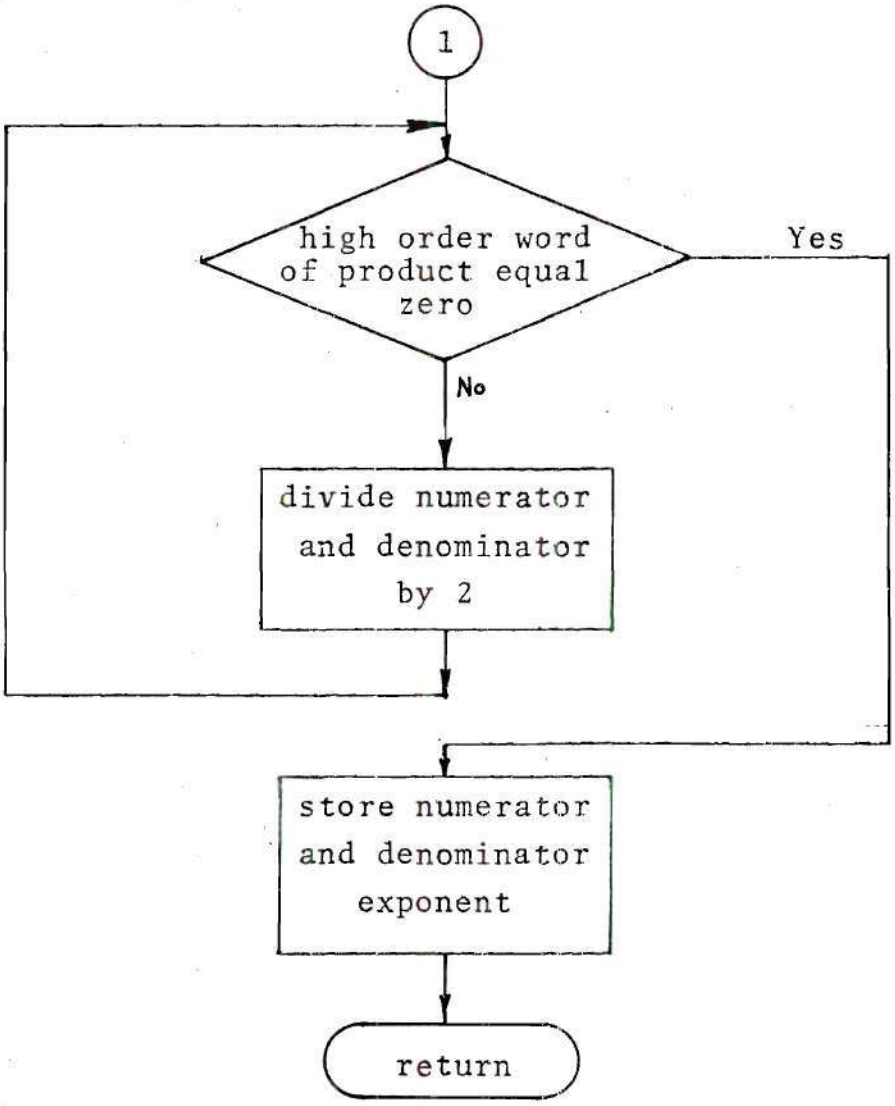


TERM

This routine makes the 2xRHOxOMEGA terms for velocity feedback from the eigenvalues and the damping ratio.



TERM (continued)



SHORT

This subroutine shortens the numerator of the product of two ratios to a one word integer. The denominator exponent is also adjusted to maintain proportionality.

(1 word integer/2**x)x(1 word integer/2**y)

= (2 word integer/2**[x+y])

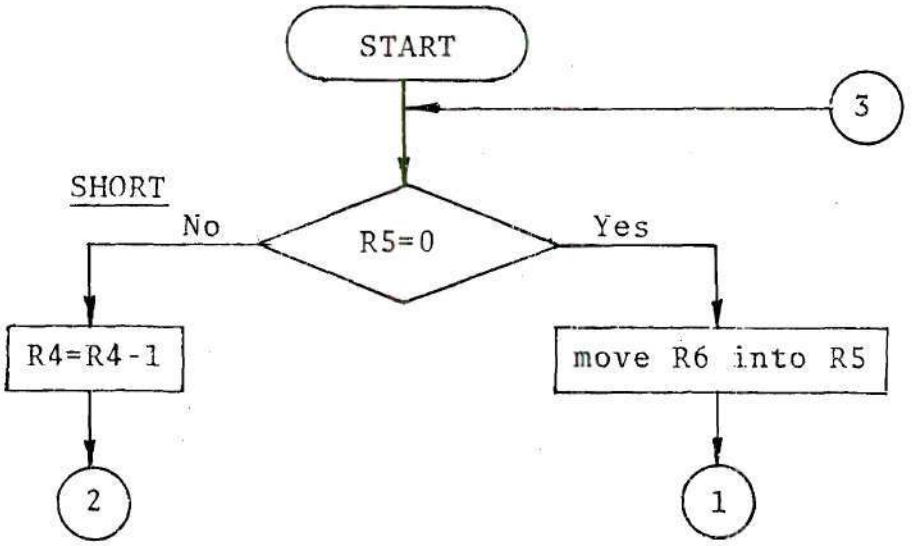
call SHORT:

= (1 word integer/2**z)

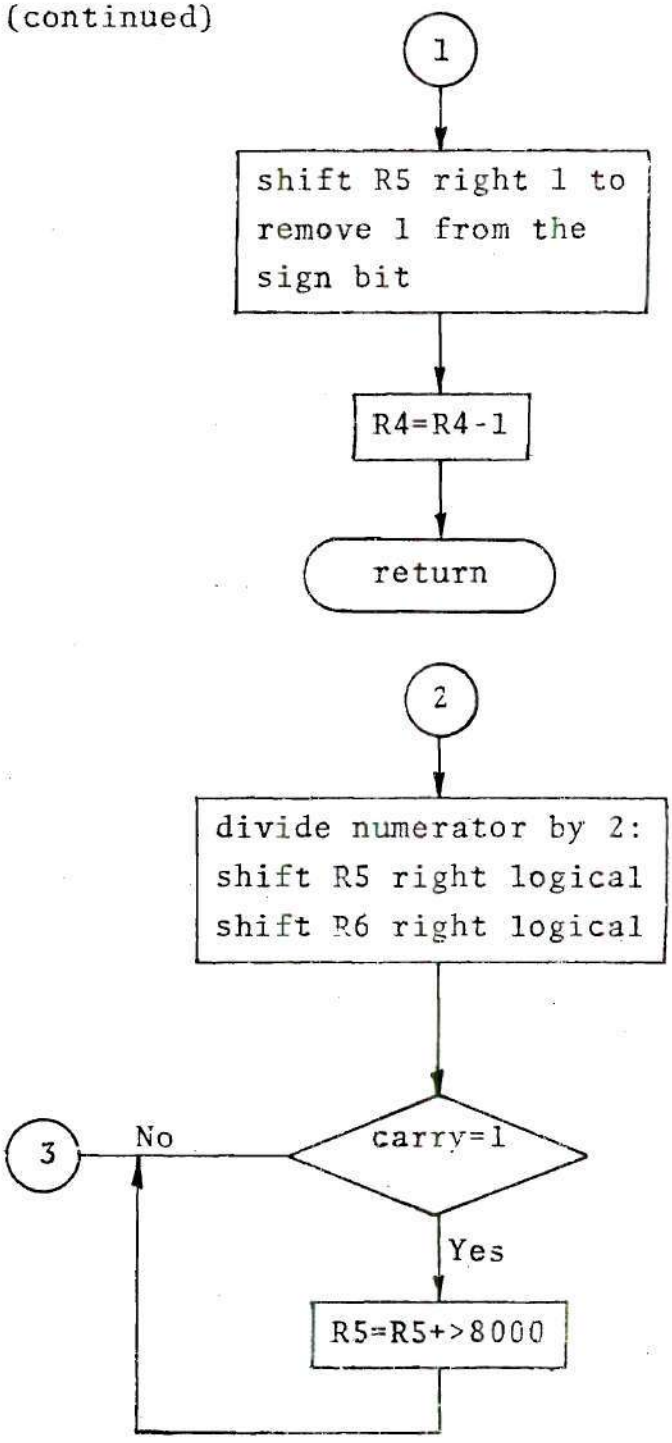
R5 = high order word of numerator

R6 = low order word of numerator

R4 = denominator exponent



SHORT (continued)



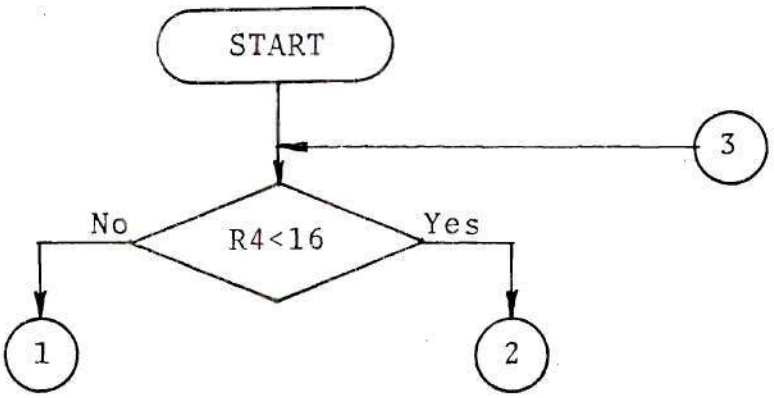
FIT

FIT will adjust the ratio (1 word integer/2**x) to (1 word integer/2**16) adjusting the numerator to maintaining proportionality. If the adjustment is impossible, FIT will stop the table making process and send an error message to the user. Pertinent registers will also be saved on a list starting at SVE and the program will return to KBI at a location that will allow the user to make a new variable change command.

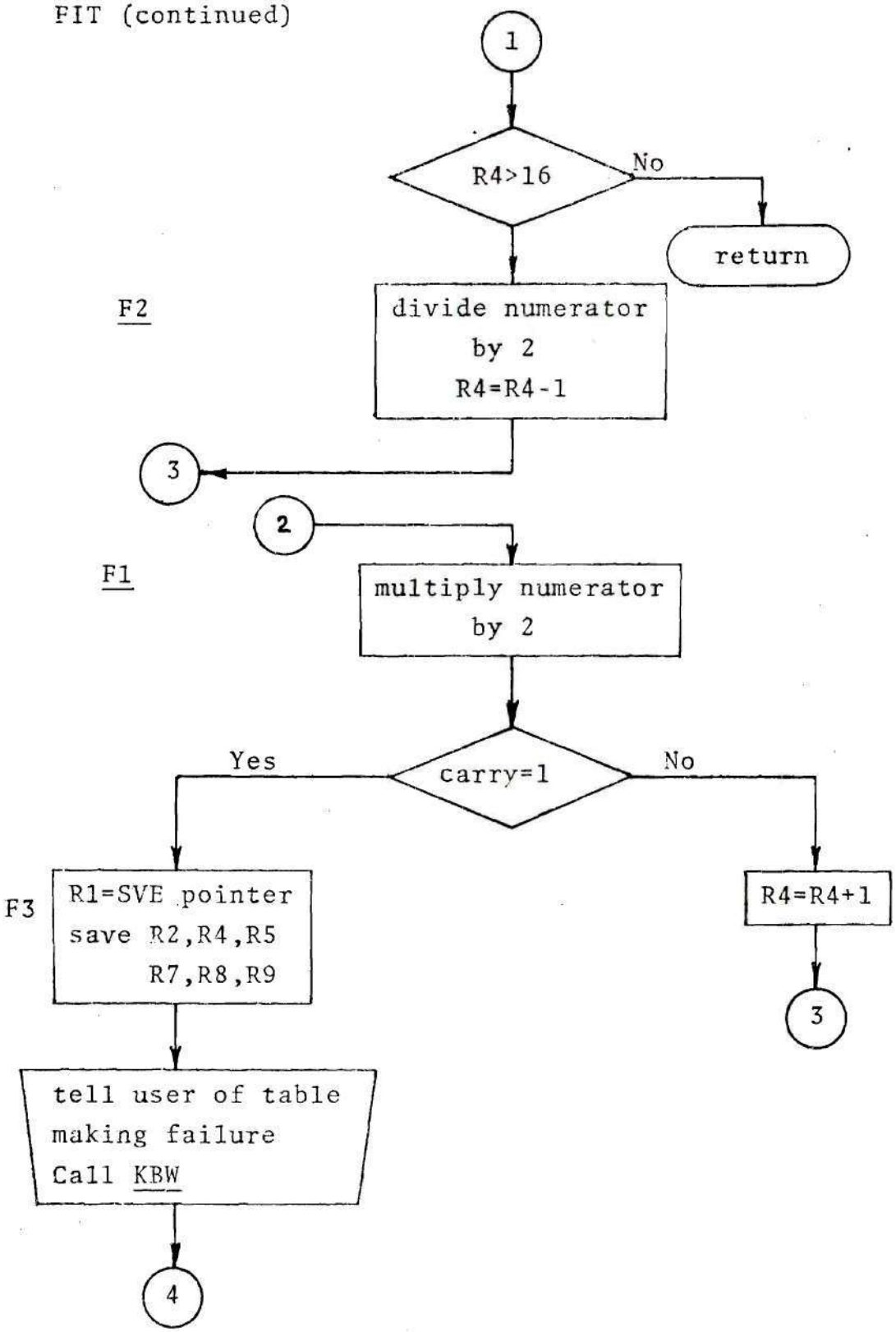
R5 = numerator

R4 = denominator exponent

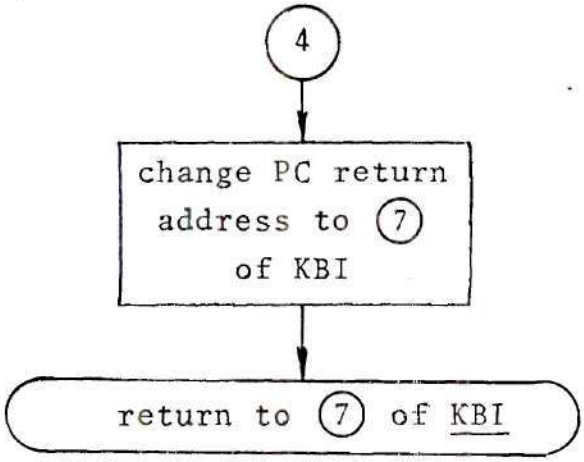
FIT



FIT (continued)



FIT (continued)



RED

This is the arm feedback routine called by a level 3 interrupt, it reads the errors, calculates the torques, then outputs the torques to the torque motors.

R1 = BETA1

R2 = BETA2

R3 = BDOT1

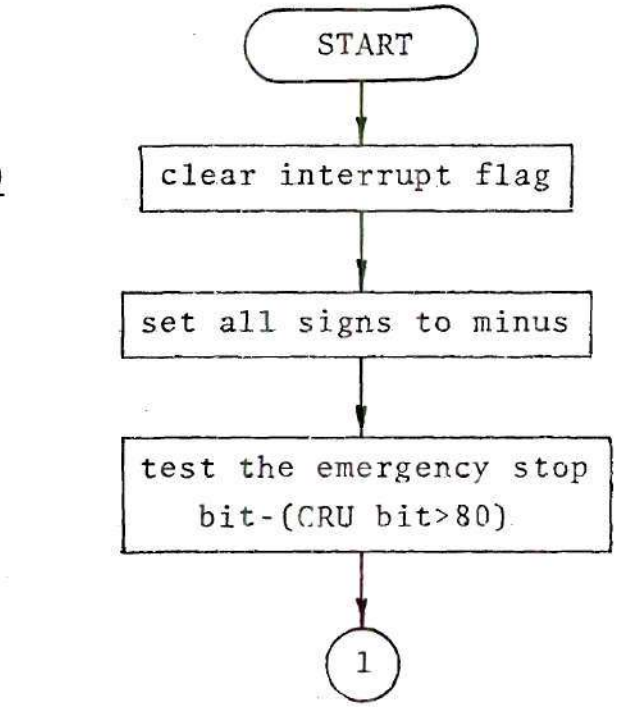
R4 = BDOT2

R5 = coefficient table pointer

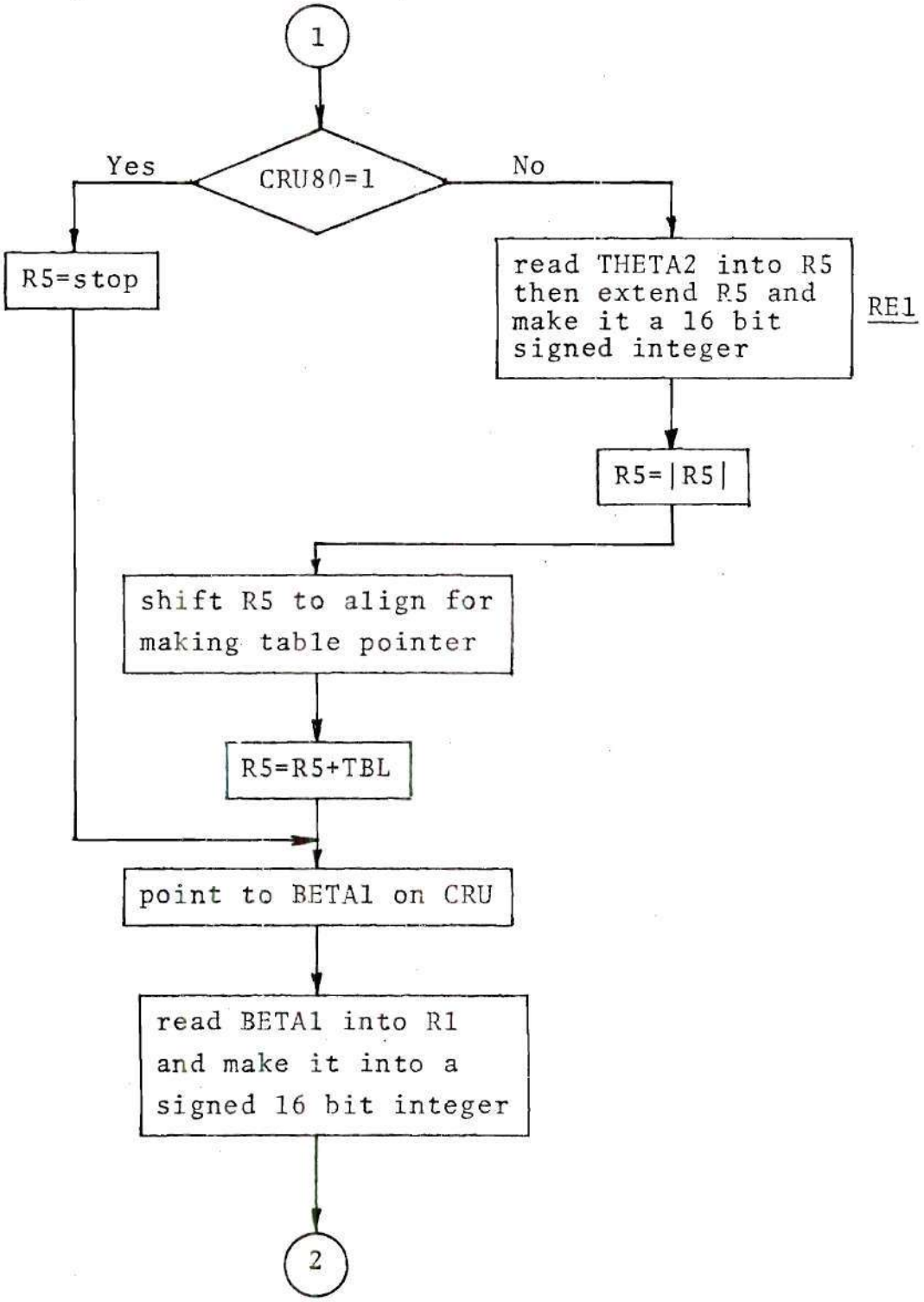
R6 = sign list pointer

R7,R8 = product of error and coefficient

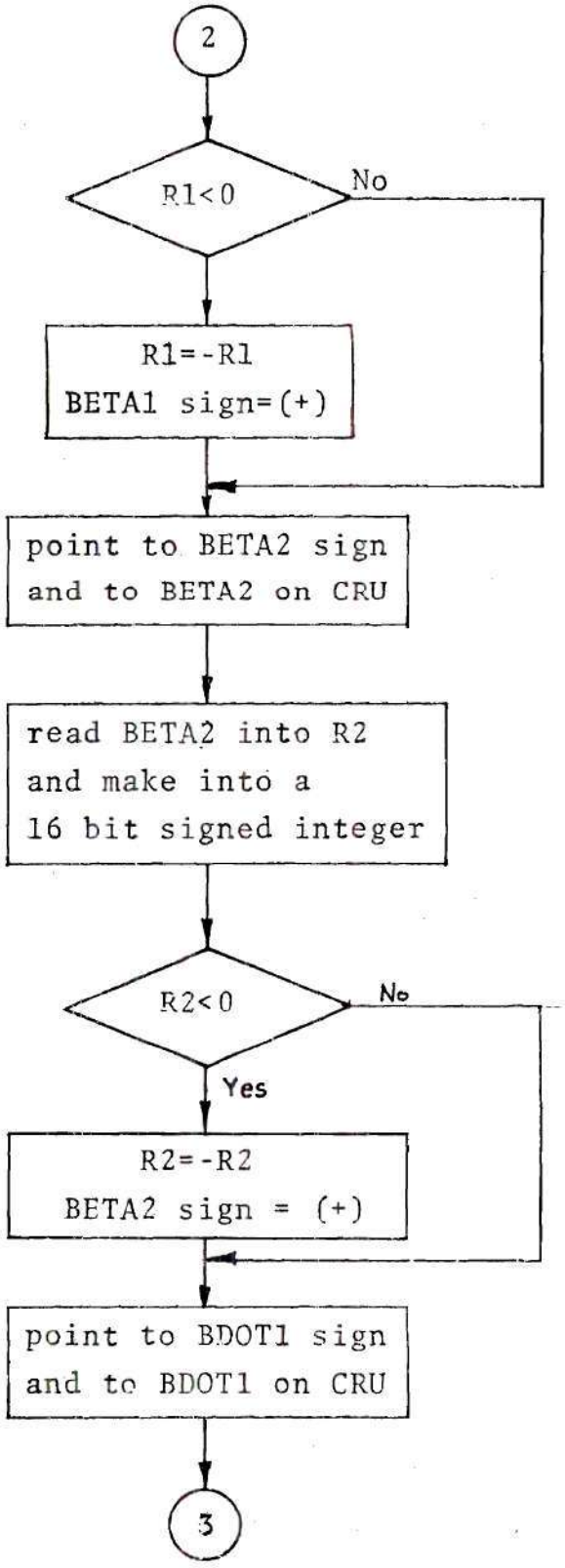
R9 = summing register



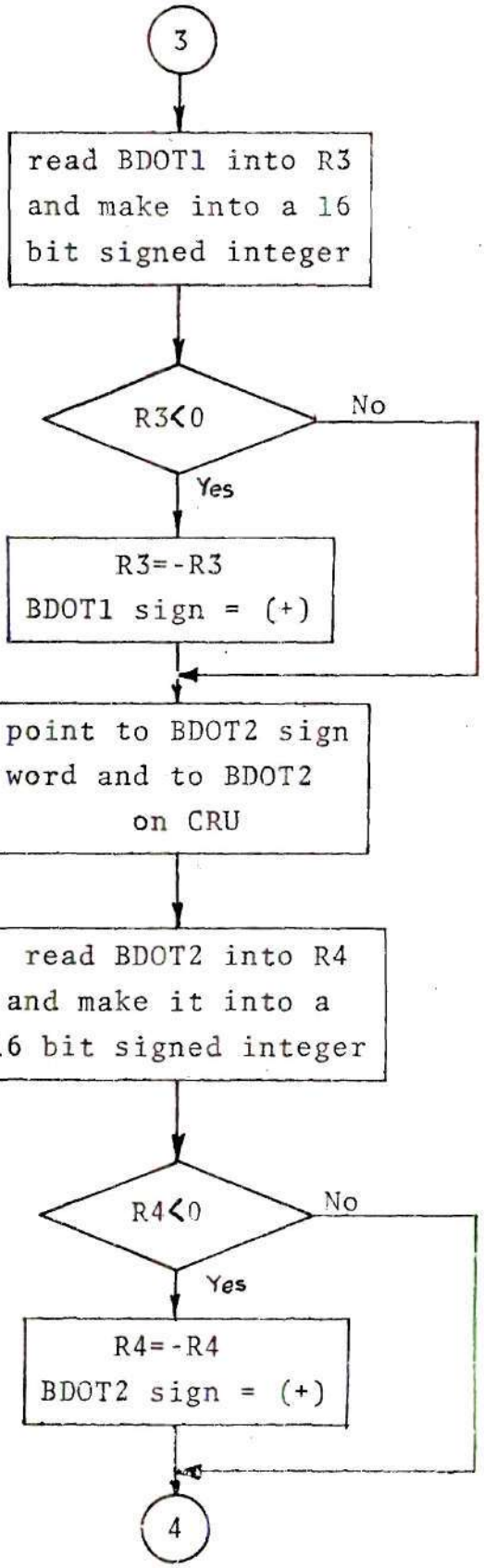
RED (continued)



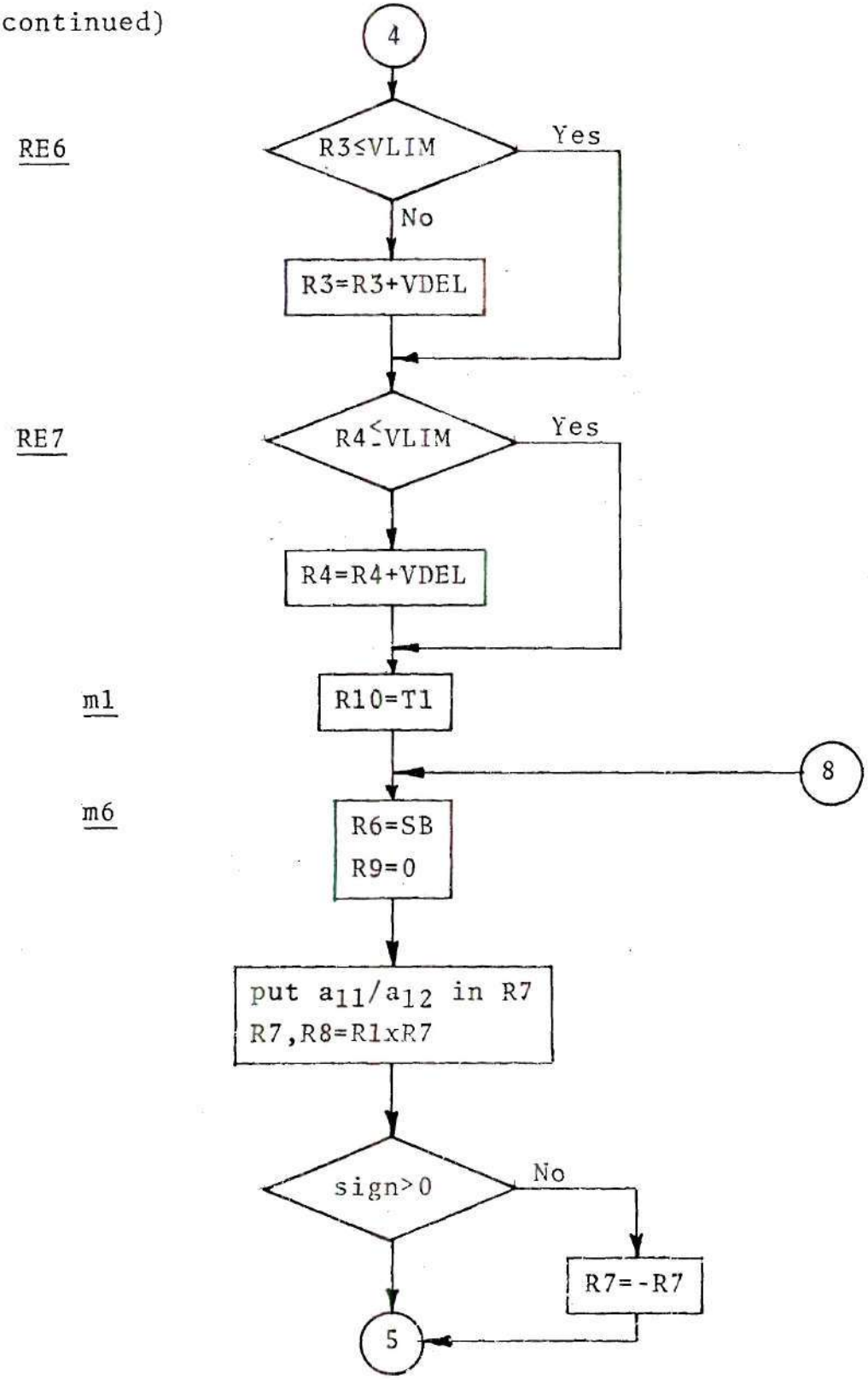
RED (continued)



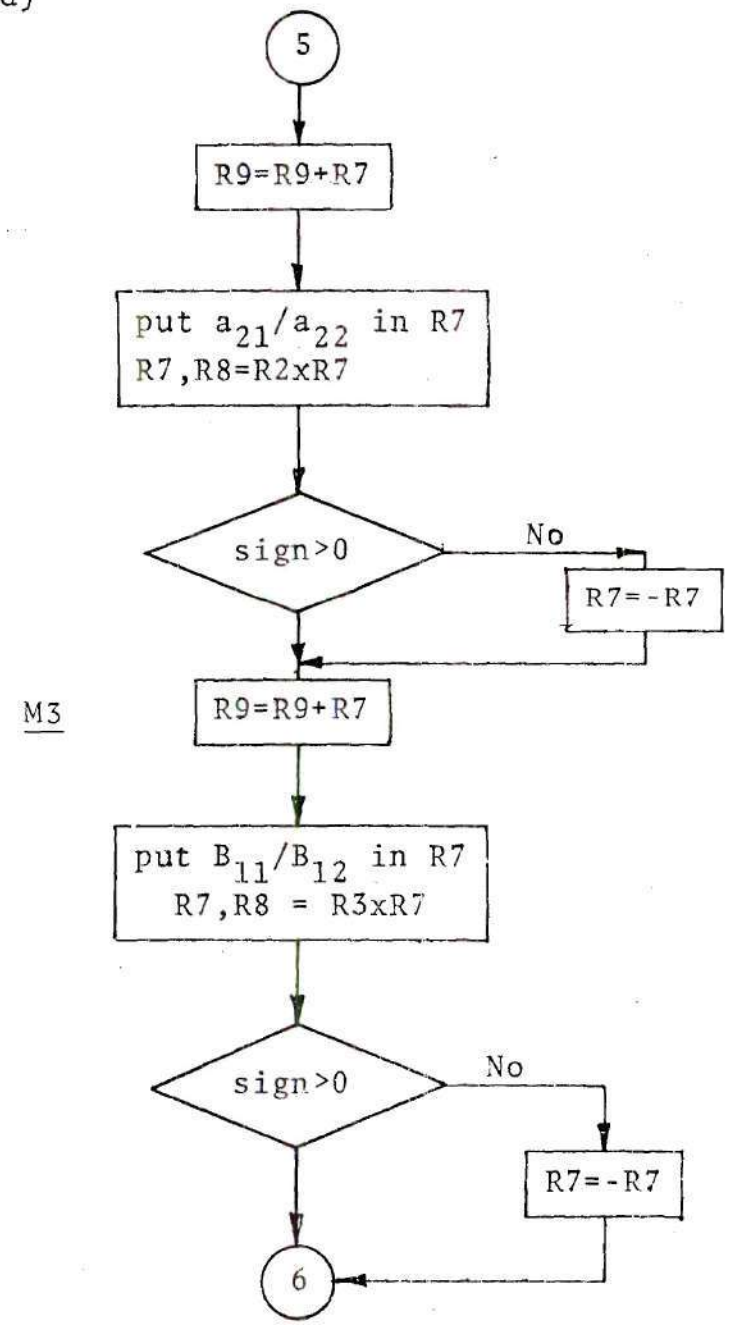
RED (continued)



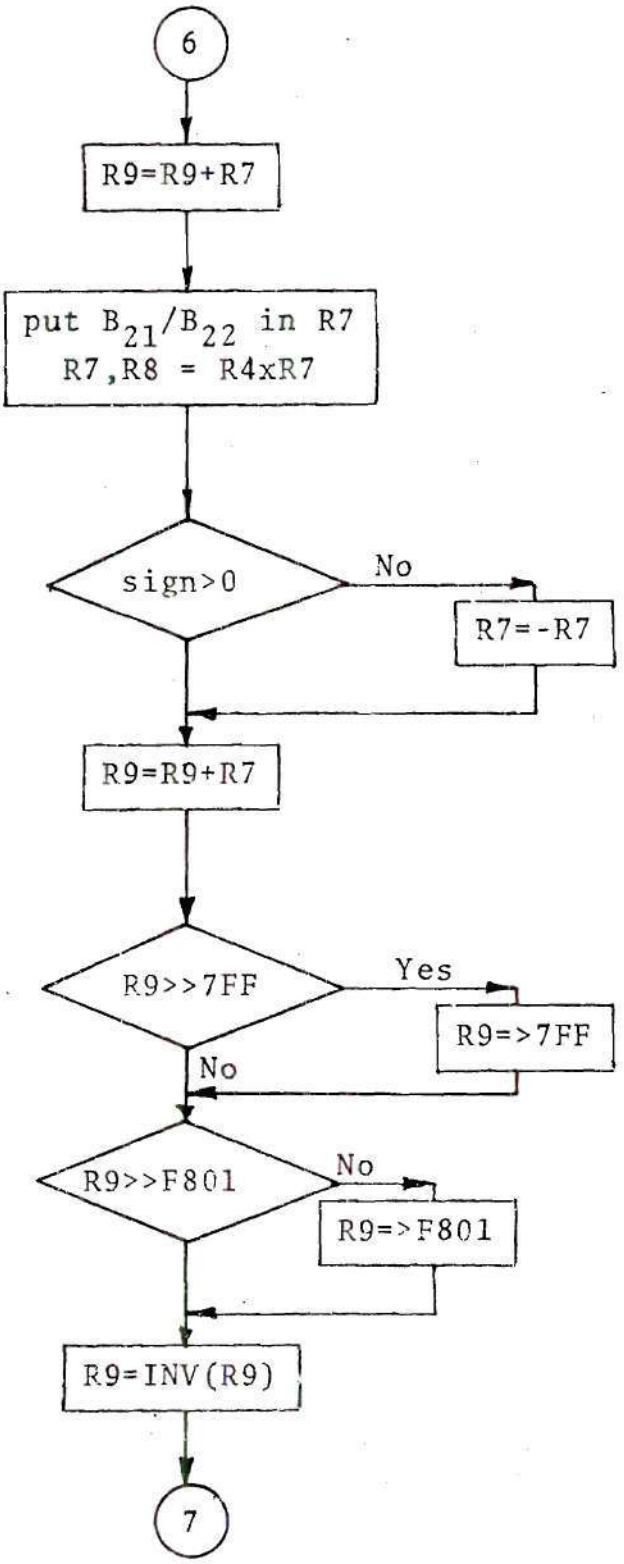
RED (continued)



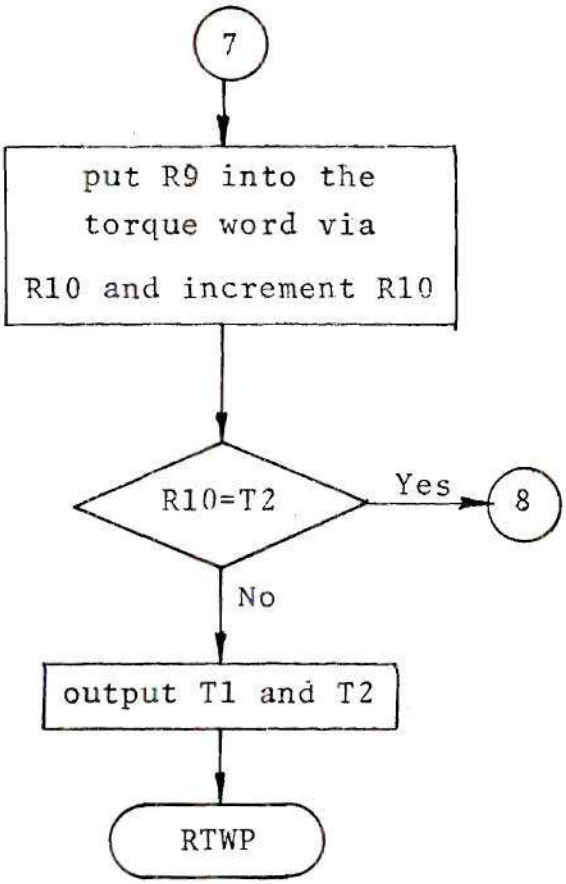
RED (continued)



RED (continued)



RED (continued)



APPENDIX II

SCHEMATIC OF LINK #1 AND LINK #2

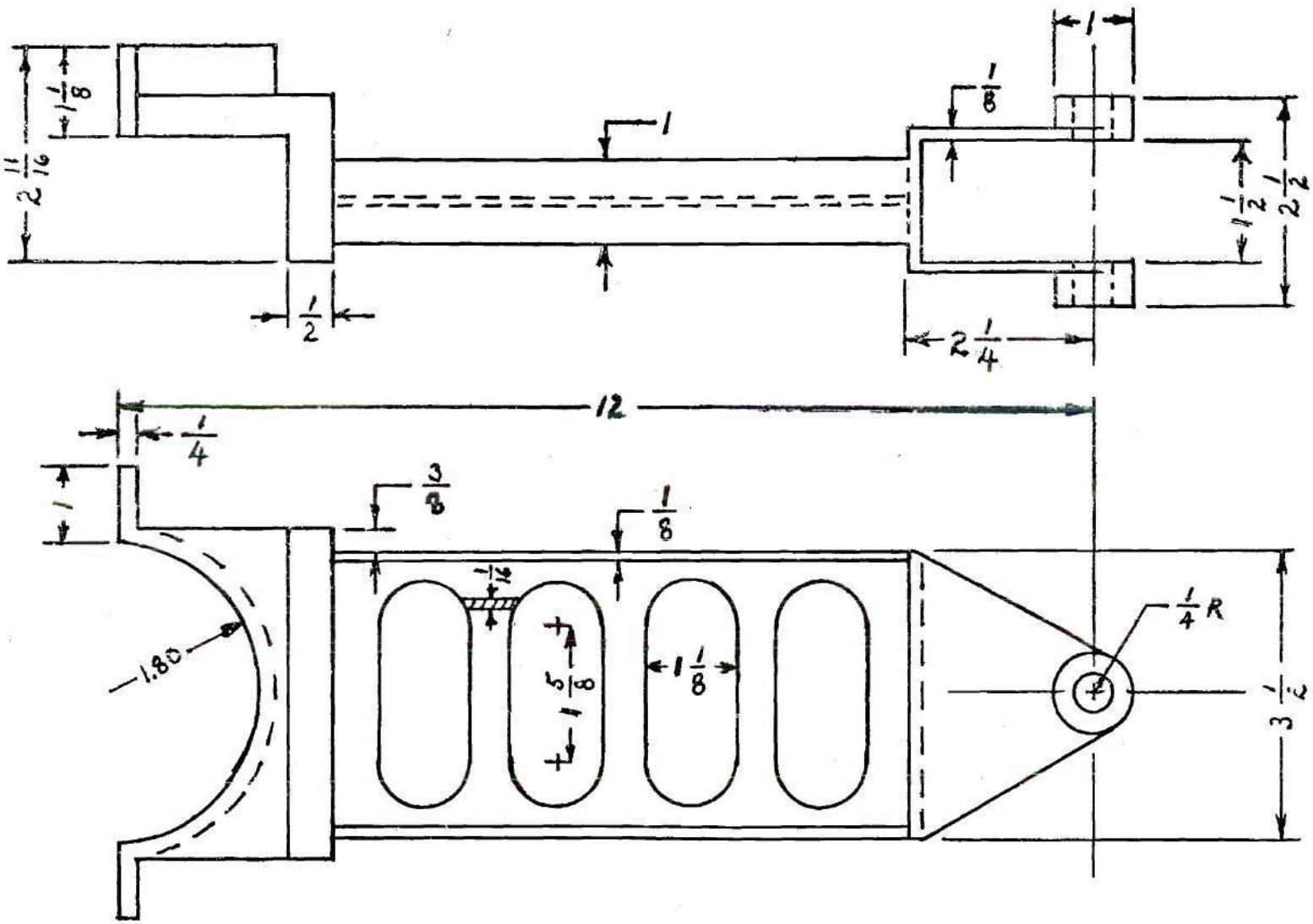


Figure A-2-1. Schematic of Link #1

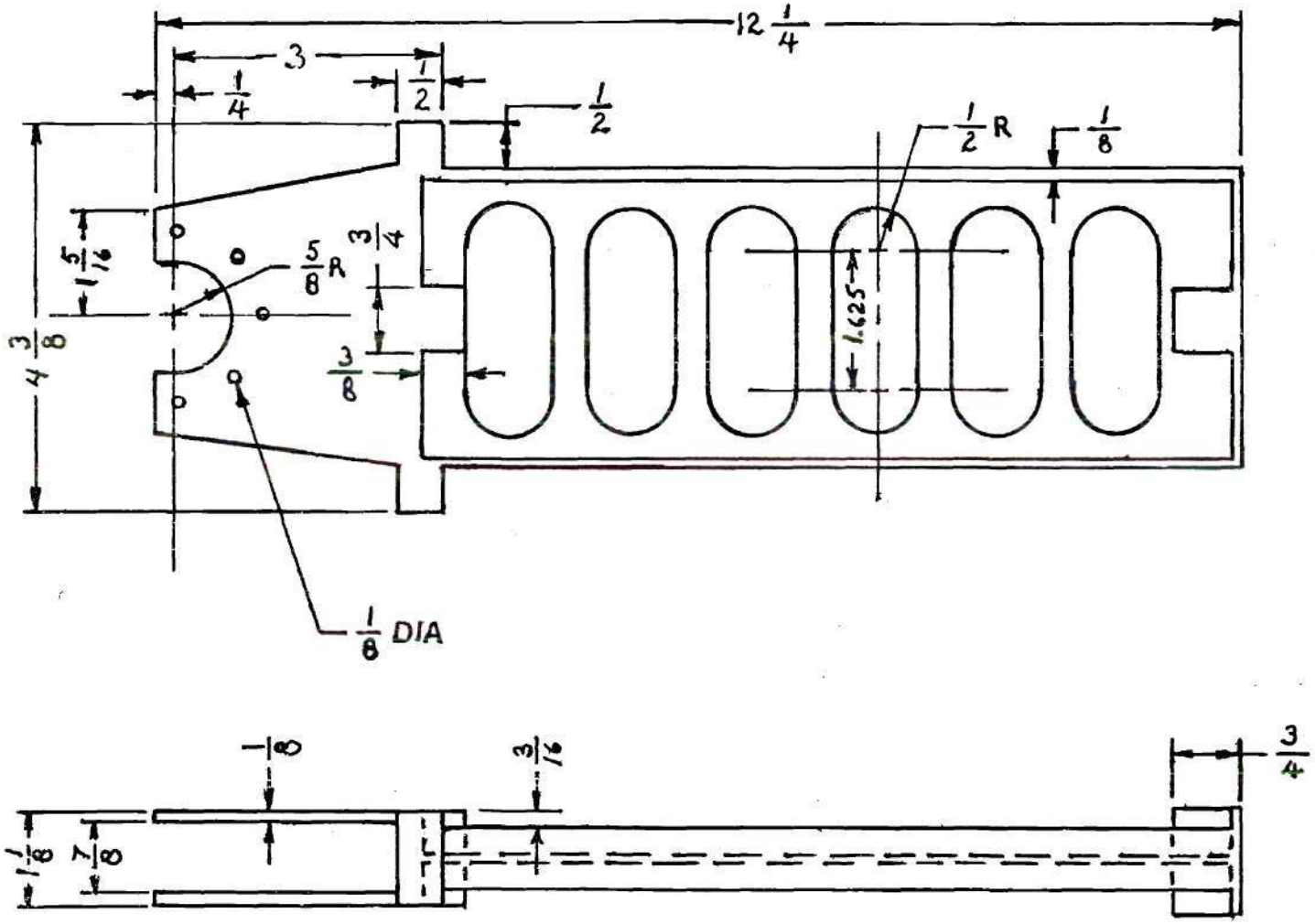


Figure A-2-2. Schematic of Link #2

APPENDIX III

OPERATING INSTRUCTIONS

Start-up Procedure

Before following the procedure below the user should have all circuit boards balanced and calibrated, power amplifier gains set and all wiring to actuators and on the wire-wrap board completed.

1. Align arm in same position as master. The elbow and shoulder angles should be as close to the angles of the master as possible.

2. Depress RESET button on emergency stop circuit module. This lowers the emergency stop flag permitting normal control operation.

3. Connect pulse train source to INTERRUPT socket on the back of interface chassis and pulse height should be between 3.5 VDC and 5 VDC and the low state should be between 0 VDC and 0.8 VDC.

4. Mount system cassette on ASR 733 and load. System has an automatic start command at the end of the load. When load is complete the arm may go through small motions as it initializes arm position.

5. When "OK-START" is typed on the printer by the system the master may begin motions and the arm will follow.

At any time during use of the arm, after the initial printing of "OK-START", the user may change programmable variables by typing in the proper command on the keyboard. When the system receives the command it will bring the arm to a halt then ask the user for the new values. When it has received the new values and updated the coefficient table it will initialize the arm position again and send the "OK-START". At this time operation may resume. Tabulated in Table A-3-1 are the initial values for the programmable variables. Of course, the user may change these by changing the program source and assembling it again.

Keyboard Commands

Table A-3-2 lists the commands used for changing the values of programmable variables. Following are several examples of their use.

Example 1

The user wants to change ω_1 to 4.35. First, 4.35 must be expressed in the proper format for input. A little arithmetic shows that,

$$4.35 \approx 1114/256 = 1114/2^8$$

The proper format would be,

1114/8

For ω_1^2,

$$(4.35)^2 = 18.92 \approx 4844/256$$

which has an input format of

4844/8.

The squared term does not need to have the same magnitude denominator as the first degree term. Input is as follows:

P CR (carriage return)

Input eigenvalue #1.

1114/8 CR

input engenvalue #2

CR

input damping ratio

CR

input (eigen.1)**2

4844/8 CR

input (eigen.2)**2

CR

wait for start signal

OK--start.

The user would use a similar procedure for changing the damping ratio or ω_2. Typing a carriage return when the computer asks for numerical input leaves the parameter unchanged.

Example 2

The user wants to change ω_1 to 5.0. The input format may be any of the following for ω_1:

5

5/

5/0

and, for ω_1^2,

25

25/

25/0

Input would proceed as in Example 1.

Example 3

A new tachometer has a maximum speed of 50.0 rad/sec and the user wants to change the A/D velocity scale factor.

All the A/D's are 12 bits long giving them a range of 4095 "counts" or "parts" and a sign bit. The scale factor is:

$$50/4095 \simeq 50/4096 = 50/2^{12}$$

which has an input format of:

50/12

The input procedure would be as follows:

V CR

input velocity scale factor

50/12 CR

wait for start signal

OK--start

Example 4

The user inadvertently hits the keyboard causing the system to stop the arm and go into keyboard mode.

G CR

wait for start signal

OK--start

Typing "G" causes the system to jump out of keyboard mode and restart.

Example 5

An 8.5 foot-pound torque motor is installed for the shoulder. The user must change the shoulder torque motor scale factor.

The input variables in the preceding examples entered the computer in codes with units of "parts" and this scale factors converted them into engineering units. Since torque is an output variable the problem is reversed. It is calculated in the engineering units of foot-pounds and must be converted into "parts" for output through the D/A.

We have,

$$\frac{4095 \text{ parts}}{8.5 \text{ ft-lb}_f} = 481.8 \approx \frac{1927}{2^2}$$

The input format is,

1927/2,

and the procedure is,

T CR

input shoulder motor torque scale factor

1927/2 CR

input elbow motor torque scale factor

CR

wait for start signal

OK--start.

Example 6

The user wishes to stop the arm without making any variable changes. The user types in

E CR

and the system will bring the arm to a controlled stop then wait as if it were shut off. Typing in any character will cause the system to restart at arm position initialization. If variables had been changed before an "E" had been typed the new values wil be preserved when the "E" is typed.

Emergency Stop

If, during an experiment, either link of the arm should

hit any one of four microswitches the emergency stop bit will be set on the CRU causing the control to go into emergency stop mode. The control will bring the arm to a stop then continue to monitor the input variables remaining in emergency stop mode.

The system will stay in emergency stop mode until the user pushes the RESET button on the microswitch signal conditioning circuit module. The proper procedure for restarting after an emergency stop is as follows:

1. Type "E" on keyboard. This will disable the control.
2. Depress RESET button on microswitch module.
3. Move arm into start-up configuration in safe operating region. The configuration should match the master configuration as in the initial startup.
4. Type in any character. When the computer sends the "OK--START" message the user can begin operations again.

BIBLIOGRAPHY

1. Book, Wayne J., "Modeling, Design and Control of Flexible Manipulator Arms," Ph.D. Thesis, Department of Mechanical Engineering, Massachusetts Institute of Technology, Chapters 4-5, April, 1974.

2. Book, Wayne J., Op. cit., Chapter 3, pp. 151-155.

3. Fertis, Demeter G., Dynamics and Vibrations of Structures, pp. 25-26, John Wiley and Sons, Inc., New York, New York, 1973.

4. Takahashi, Rabins and Auslander, Controls, pp. 421-423, Addison-Wesley Publishing Company, Reading, Massachusetts, 1972.