# CLUTCH SIMULATION
# FINAL REPORT TO
# FORD MOTOR CORPORATION

by

C. Pierre, Assistant Professor
J.R. Barber, Professor
P. Papalambros, Associate Professor
P. Cha, Graduate Student

April 1986

#UM-MEAM-86-38

# CLUTCH SIMULATION
# FINAL REPORT TO
# FORD MOTOR CORPORATION

by

*C. Pierre*

Assistant Professor

*P. Cha*

Graduate Student

*J.R. Barber*

Professor

*P. Papalambros*

Associate Professor

UM Contract No. 388792

April, 1986

TABLE OF CONTENTS

(1) Numerical Simulation of Clutch Dynamics

a. Introduction

b. Current Status

c. System Model

d. Numerical Solution

e. Programming Logic

f. Program Usage

g. Future Goals

(2) Supplementary Tables and Diagrams

a. Parameter Description

b. State Variable Listing

c. System Control Diagram

d. Programming Hierarchy

(3) Programming Logic of Clutch.Kut

a. Primary Level

b. Secondary Level: Interactive Options

c. Differential Equation Subroutines Called by KUTTA for Appropriate State

d. Forcing Function Called by KUTTA to Provide Input Data

e. Data Base Subroutines Called by State Subroutines

(4) Program Flow Chart

(5) Graphical State Representation

(6) Translational Model

a. Model Description and Parameters

b. Initial Clutch Positions and Physical Clutch Parameters

c. System States

d. Translational Equations of Motion Summary

(7) **Rotational Model**

a. Model Description and Parameters

b. Rotational Equations of Motion

(8) **Analysis of Coulomb Friction**

(9) **Results**

a. Case 1: Clutch Engagement

b. Case 2: Engine Torque Transmission

(10) **Program Listing**

# 1. NUMERICAL SIMULATION OF CLUTCH DYNAMICS

## 1.a Introduction

The primary purpose of this report is to provide Ford with a clear understanding of the current status of the main clutch simulation program. Material is presented which documents the system model, numerical solution method, programming logic, and program usage.

## 1.b Current Status

Major accomplishments since the last update are as follows:

- A new numerical integration scheme has been implemented to replace the IMSL subroutine DGEAR. For the major drawback of DGEAR was that it could not jump accurately from one set of equations to another, i.e., from state to state. The program now uses a fourth order Runge-Kutta integration scheme. Very accurate numerical results have been obtained. In addition, the system now passes from state to state while undergoing stiction/friction phenomenon.

- Obtained numerical solutions have been cheched against known analytical results, and excellent agreement has been observed. Numerical results are presented in Section 9.

- Previous errors associated with incorrect axes scaling in subroutine GRAPH have also been eliminated.

## 1.c System Model

The system model is described on several levels which are covered thoroughly in Sections 2 through 7. These include a System Control Diagram, and the state equation derivations with corresponding component free body diagrams. These are fully up dated and fairly self explanatory.

At this point, some comments can be made concerning current limitations of the overall system model. By reviewing the System Control Diagram it can be seen that the clutch is only one of several important components. Thus far, other components have been treated with great simplicity. For example, the engine is modeled as a constant torque/power output device. Obviously, engine output is a complicated function of many variables. In addition, the load or drive system and vehicle inertia are made up of a few lumped components. The pedal linkage has not been modeled at all. Instead, a time dependent forcing function ($F_{input}$) is applied directly to the release bearing. Finally, human interaction has not been accounted for in the code. Many of these deficiencies have been considered in the present code in such a way that when data becomes available, it may be inserted into the appropriate subroutines immediately. However, it may be proven necessary to increase modeling detail as well, by using a finer distribution of lumped springs, masses, and dampers.

Present data used in the code refelects closely the actual clutch. An exception to this is the current linear approximation of highly non-linear characteristics. Also, it should be noted that the translational damping in the clutch model is now of viscous type. This should be changed to coulomb type. Another possibly significant effect has been temporarily neglected by not including momentum transfer and impact. This may be included between translational components such as the pressure plate and clutch assembly, or rotational components such as the outer clutch hub and inner clutch hub (transmission input shaft).

Another area requiring enhancement is the stiffness and kinematics of the bellville spring, clutch cover, and drive straps. These are now lumped into one spring and mechanical lever. While this may be acceptable from a quasi-static point of view (i.e. stiffnesses may be added), it may not hold for a dynamic model. Each component should be modeled independently. Also, because of initial flex in the clutch cover during disengagement, the bellville spring

actually pivots about two different points; first it pivots at the pressure plate, then it pivots at the clutch cover. Therfore, there is a need for defining an additional translational state and logic to determine the transition point within KUTTA, the subroutine which determines the correct system state.

Lastly, it remains to apply dependent friction coefficient logic and data. This includes temperature effects, velocity effects, and static/slipping consideration. There can be little question that the friction coefficient treatment is very important to obtaining an accurate model. A current error in the code is the inability of the torsional friction damper in the clutch hub interface to resist motion in either direction. The same is true at the rear wheel. That is, a change in sign on the friction term is necessary in the equation of motion which describes the appropriate degrees of freedom where coulomb friction is concerned. Otherwise, energy will be incorrectly input to the system. This problem has been addressed in the main clutch pad/pressure plate/flywheel interface (see flow chart KUTTA). Since there are so many friction dependent equations of motion, it might make sense to create a seperate, generalized friction/stiction subroutine which could be routinely called. This would ease the modeling expansion process greatly.

### 1.d Numerical Solution

The numerical solution method used in the earlier version of the program was an integration subroutine called DGEAR. This subroutine solved a simultaneous system of first order differential equations. It had the option of either using the Adams solution method or the Gear solution method (for stiff system). Essentially, DGEAR was called at the incremental time steps, through a DO-loop in the main program. Each time DGEAR was called, it in turn called subroutine FCN. FCN was responsible for determining the correct system state, whereupon that state was then called. Once this had occured, the state variable first order derivatives were calculated and passed back through FCN to DGEAR. At this point , DGEAR was able to carry out the integration process to determine the appropriate state variable values which could then be used as input for the next time step.

The major drawback of this method is that DGEAR cannot jump from one set of equations to another. This is due to the fact that DGEAR does not need to call FCN at every time step. DGEAR is a very efficient method because of its ability to extrapolate future values. It then corrects itself upon the next call to FCN. It, however, causes a transition between state equations to be delayed since FCN is only called when DGEAR deems it necessary. This problem was found to be very significant, since it was observed that the use of DGEAR led to erroneous results. Thus, the program had to be modified by using a different solution method, namely the fourth order Runge-Kutta time integration scheme.

The principal advantages of the Runge-Kutta method lie in its self starting features and its ease of programming. Here, it also has the added advantage in that it enables a smooth transition between state equations to occur since subroutine KUTTA (the new subroutine which determines the correct system state) is called at every time step. Thus, if the time step is small enough, transition between states can be effectively and quite accurately simulated.

A fourth order Runge-Kutta method employs a recurrence formula of the form

$$y_{i+1} = y_i + \frac{(k_1 + 2k_2 + 2k_3 + k_4)}{6}$$

to calculate successive values of the dependent variable y of the differential equation

$$\frac{dy}{dt} = f(y, t)$$

The $k_i$'s are as follows,

$$k_1 = \Delta t\, f(t_i, y_i)$$

$$k_2 = \Delta t\, f(t_i + \frac{\Delta t}{2}, y_i + \frac{k_1}{2})$$

$$k_3 = \Delta t\, f(t_i + \frac{\Delta t}{2}, y_i + \frac{k_2}{2})$$

$$k_4 = \Delta t\, f(t_i + \Delta t, y_i + k_3)$$

In this fourth order Runge-Kutta method, the per step error is of the order $(\Delta t)^5$.

Subroutine KUTTA has the same function as FCN in the earlier version. It determines the correct system state, then calls that corresponding state. Once this has occured, the state variable first order derivatives are computed and passed back to KUTTA, at which point KUTTA may perform the Runge-Kutta time integration process to determine the appropriate state variable values which can then be used as input for the next time step.

### 1.e Programming Logic

The code is of modular design for ease of revision and understanding. The section contains a programming hierarchy chart and a brief written description of each subroutine to accompany it. Of further use are parameter and state variable listings. These sources of information should be combined for a full interpretation.

Essentially, the code hierarchy works as follows: First, the main program, or primary level, is run interactively. There are several secondary subroutine options which may be selected from this point. They include DEFALT, MODIFY, and GRAPH, all of which are for pre- and post-processing purposes such as parameter selection, modification, and graphics. As described in the numerical solution section, the Main program calls KUTTA. Subroutine KUTTA's primary prupose is to select the appropriate state equations, evaluate them using fourth order Runge-Kutta time integration method, and then send information back to the main program. KUTTA must therefore contain the friction/stiction logic previously discussed in order to choose the correct state. KUTTA always calls FINPUT and TORQIN. These subroutines provide forcing functions which act on the state equations. These are the release bearing forces application and engine torque output. Generally speaking, these subroutines require input information since it is a closed-loop system. Once KUTTA has selected the corrected system state, database subroutines are called to evaluate system parameters such as spring rate and damping rates. The reason for seperate subroutines is that these parameters may be highly non-linear, and therefore depend on current state variable values. these subroutines are BELSPR, MARCEL, and ENGINE. Notice that additional subroutines for friction coefficients and perhaps new parameters associated with a refined model may be easily appended to the code.

### 1.f Program Usage

The program is designed to be used interactively. Within the interactive menus are examples which explain their usage. There are currently three basic options. There are pre-processing, running the code, and post-processing. The pre-processing includes the ability to modify parameter values or select default values. Eventually, this will be expanded to enable initial values of state variables to be altered. Code running options include time step size and

total number of time steps for the system response. The post-processor contains a graphics package which plots any chosen variable versus time.

## 1.g Future Goals

Future goals have been mentioned throughout this report. They shall be listed here as follows:

(1) Develop a friction coefficient subroutine for the clutch/flywheel interface to take into account static and slipping behavior, as well as temperature and velocity dependence.

(2) Consider the addition of a general stiction/friction logic subroutine.

(3) Correct all potential energy input situations which will not occur in reality. These include angular and translational momentum conservation, impact losses, and friction force opposition to forward motion.

(4) Enforce the clutch process to be reversible. Presently, the code is intended for engagement only. Universal application to various situations might necessiate some further programming logic. Work is underway concerning this topic.

(5) Implement a two stage spring and possible backlash effects into the clutch hub.

(6) Replace temporary numerical values in code with parameters. In general, these are currently associated with the bellville spring linear approximation and certain displacement constants.

(7) Change viscous damping in clutch model to Coulomb damping.

(8) Replace the single spring of stiffness $k_2$ with independent components for the bellville spring, clutch cover, and drive straps. Also, add an additional translational state which would correctly model the two-pivot action of the bellville spring.

(9) Include accurate engine mapping information in subroutine TORQIN.

(10) Determine representative release bearing load versus time curves for use in subroutine FORCIN.

(11) Refine drive train model to include more springs, masses, dampers, etc. Consider backlash.

(12) Model the hydraulic/mechanical pedal linkage.

(13) Correct subroutine MODIFY to work interactively.

(14) Add new options to interactive menus as described in program usage section.

Obviously, these goals must be attemped in a correct priority. This topic warrants further discussion with Ford.

## 2. SUPPLEMENTARY TABLES AND DIAGRAMS

### 2.a Parameter Description

$m_2$—pressure plate mass

$m_4$—clutch assembly mass

$I_1$—pressure plate/flywheel/crankshaft/clutch cover moment of inertia

$I_2$—outer clutch pad assembly moment of inertia

$I_3$—transmission moment of inertia

$I_4$—vehicle mass transformed into equivalent moment of inertia

$I_e$—engine block moment of inertia

$k_e$—engine mount sprint rate

$k_2$—release bearing load spring rate

$k_3$—marcel (cushion) spring rate

$c_e$—engine mount viscous damping rate

$c_2$—release bearing viscous dampring rate

$c_3$—marcel (cushion) viscous damping rate

$T_e$—engine output torque

$F_{input}$—release bearing input force

$Torqin$—engine torque which drives system inner and outer clutch

$THKT$—temporary constant used to calculate release bearing load due to spring $k_2$

$r_D$—radius of driving wheel

$r_H$—mean radius to clutch hub damping surface

$r_C$—mean radius to clutch pad torque transfer surface

$r_E$—radius from engine crankshaft axis to engine block mount

$\mu_f$—clutch pad friction coefficient

$F_D$—tire rolling friction force

$T_2$—clutch pad torque transfer from engine

$N_1$—normal force on pressure plate side clutch pad

$N_2$—normal force on flywheel side clutch pad

$N_3$—normal force on clutch hub coulomb friction damper

$TR$—transmission gearing ratio

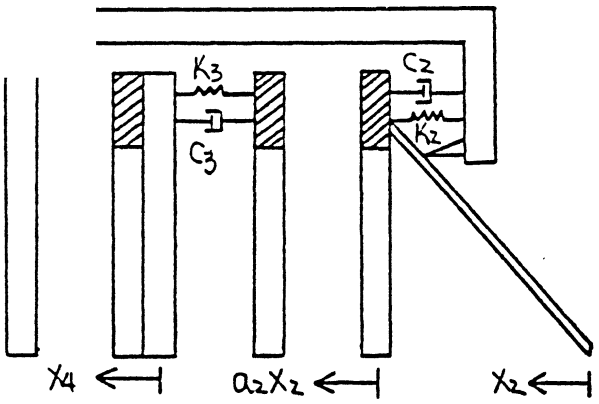
$a_2$—ratio of release bearing displacement to pressure plate displacement

$D_t$—initial gap between clutch pad and flywheel plus compressed clutch width

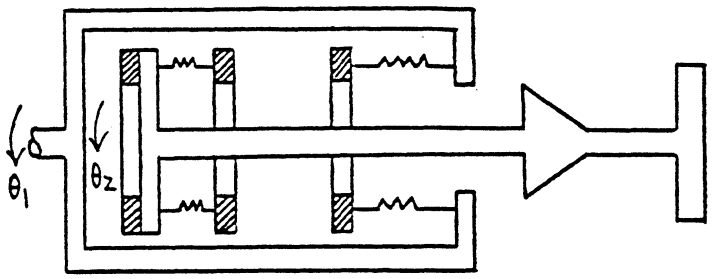
## 2.b State Variable Listings

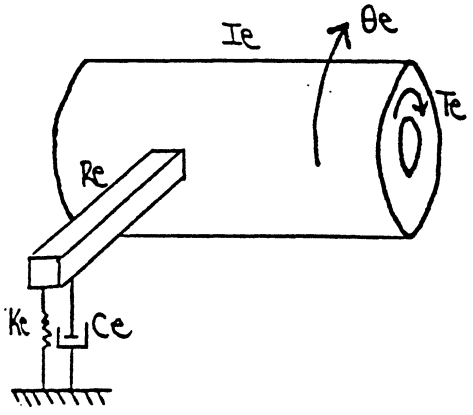
| Displacements | Velocities | Accelerations |
|---|---|---|
| Q(1)=$x_2$—release bearing (translation) | QDOT(1)=Q(2)=$\dot{x}_2$ | QDOT(2)=$\ddot{x}_2$ |
| Q(3)=$x_4$—clutch assembly (translation) | QDOT(3)=Q(4)=$\dot{x}_4$ | QDOT(4)=$\ddot{x}_4$ |
| Q(5)=$\theta_e$—engine block (rotation) | QDOT(5)=Q(6)=$\dot{\theta}_e$ | QDOT(6)=$\ddot{\theta}_e$ |
| Q(7)=$\theta_1$—crankshaft/flywheel (rotation) | QDOT(7)=Q(8)=$\dot{\theta}_1$ | QDOT(8)=$\ddot{\theta}_1$ |
| Q(9)=$\theta_2$—clutch pad assembly (rotation) | QDOT(9)=Q(10)=$\dot{\theta}_2$ | QDOT(10)=$\ddot{\theta}_2$ |

Translational Clutch Model

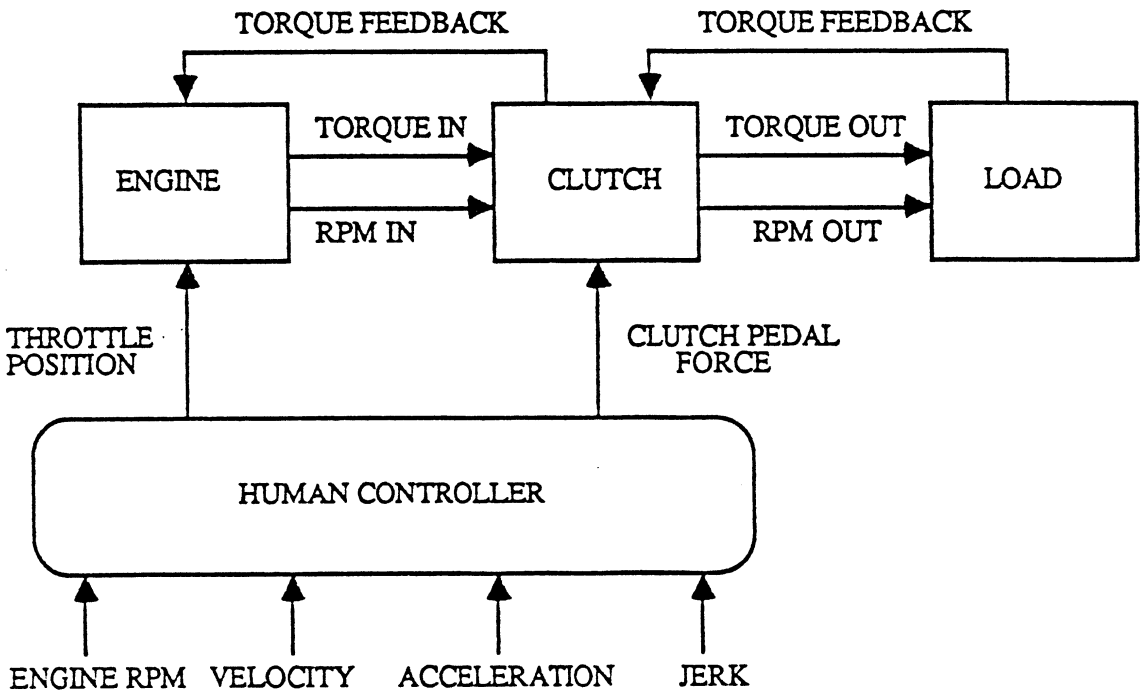



Rotational Clutch Model

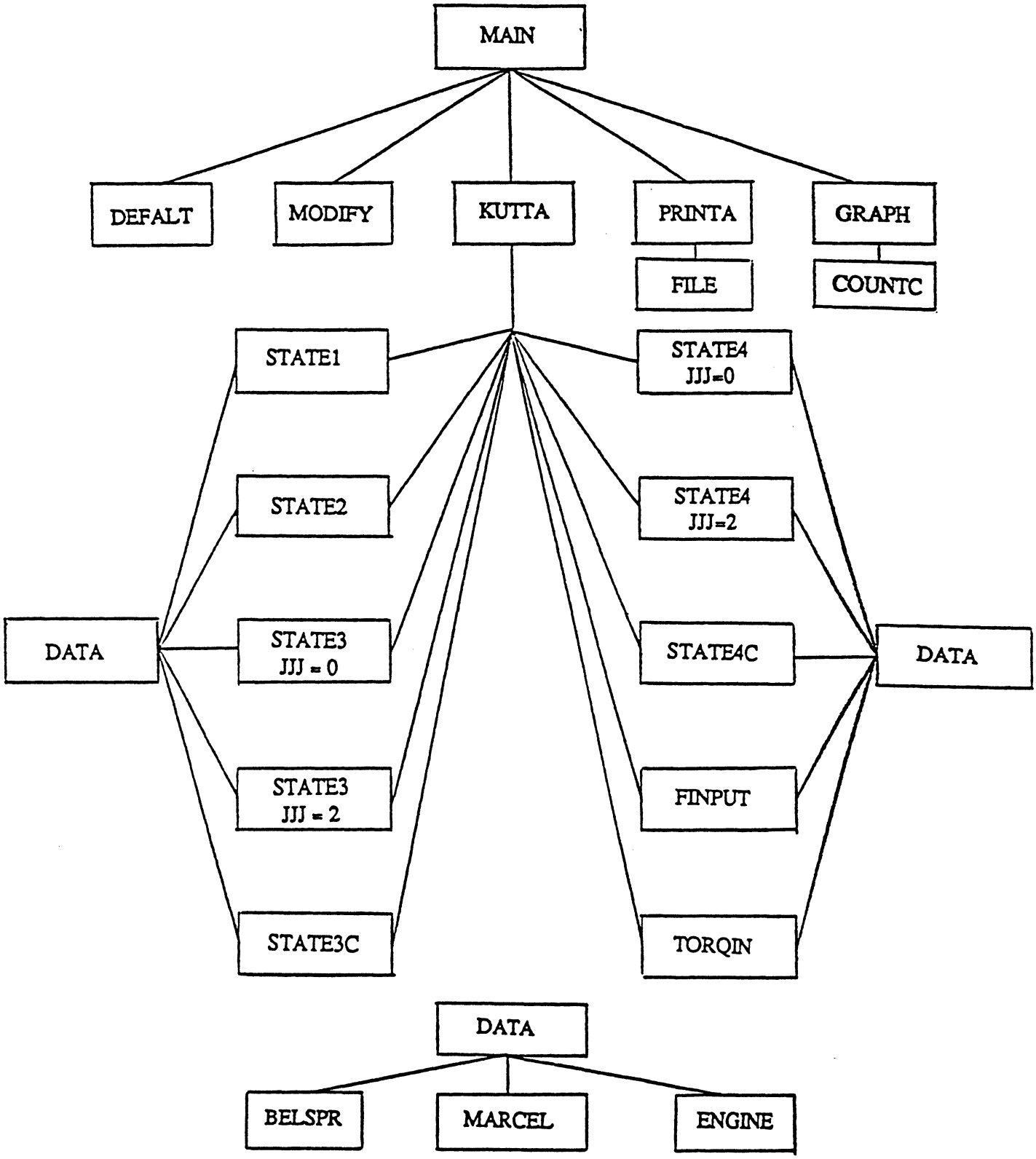


## 2.c System Control Diagram



## 2.d Programming Hierarchy




JJJ is a slip/stick indicator within the program.

## 3. PROGRAMMING LOGIC OF CLUTCH.KUT

Clutch.Kut is the primary code being developed for the modeling of the clutch system. The code incorporates a fourth order Runge-Kutta time integration scheme to handle the numerical solution. While subroutine KUTTA performs the numerical solution, other subroutines provide the logic, mathematical functions and data bases necessary to model the system.

To better understand Clutch.Kut logic, a programming hierarchy chart has been included, as well as a flow chart. A brief description follows.

### 3.a Primary Level

MAIN — The main program performs several functions.

(a) Initialization of system variables and parameters by calling subroutine DEFALT.

(b) Initialization of interactive command sequence by offering user several primary options. These options include output graphics (subroutine GRAPH), modification of system variables or parameters (subroutine MODIFY), execution of program (subroutine KUTTA), and termination of program.

(c) Writes time, state, and system variables Q(1-10) for each time step on file 08.

(d) Writes time, system variables Q(1-10), as well as Q(11) (=Q(7)-Q(9)) and Q(12) (=Q(8)-Q(10)) for each time step on file 07 for post-processing usage, namely through the use of subroutine GRAPH.

### 3.b Secondary Level: Interactive Options

(1) MODIFY — Allows user to set system parameters to default values by calling DEFALT. Instead, the user may elect to alter parameters interactively by renewing appropriate constants or entering discrete data points for non-linear parameters such as a bellville spring curve.

(2) DEFALT — Sets system parameters to previously designated values.

(3) GRAPH — Reads dependent variable as specified by user into in y array, while reading a corresponding x value of time. Assembles data for graphing purposes. Generates

x-y plots with axis labels provided by user. Software incorporates *PLOTSYS plotting routines. Requires Tektronix terminal or one which emulates a Tektronix.

### 3.c Differential Equation Subroutines Called by KUTTA for Appropriate State

(1) STATE1 — The first translational state of the clutch. No contact between clutch components exists here.

(2) STATE2 — The second translational state of the clutch. The pressure plate and clutch face are in contact.

(3) STATE3(JJJ=0) — The third translational state of the clutch. All translational clutch components are in full contact. The system is not fully compressed however. Interface surfaces are slipping. In the program, this state is designated by JJJ=0.

(4) STATE3(JJJ=2) — The third translational state of the clutch. This state is identical to STATE3(JJJ=0) in the translational sense. The immediate histoy of the interface surfaces is one of non-slip. Therefore the clutch is contacting the flywheel and the pressure plate. The interface surfaces are beginning to slip. This state is designated as JJJ=2 in the program.

(5) STATE3C — The third translational state of the clutch. Pressure plate and flywheel are both in contact with clutch. Interface surfaces are not slipping, they are sticking. Inner clutch components are not fully compressed.

(6) STATE4(JJJ=0) — The fourth translational state of the clutch. All translational components are fully compressed. The clutch face is slipping.

(7) STATE4(JJJ=2) — The fourth translational state of the clutch. All translational components are fully compressed. The immediate history of interface surfaces is one of non-slip. The interface surfaces are beginning to slip.

(8) STATE4C — The fourth translational state of the clutch. All translational components are fully compressed. There is no slipping at the clutch face.

### 3.d Forcing Function Subroutines Called by KUTTA to Provide Input Data

(1) FORCIN — Supplies time dependent clutch pedal effort. In the future, this subroutine will be dependent on other variables as well, such as vehicle acceleration. This requires further insight to human response and some optimal criteria.

(2) TORQIN — Supplies engine torque input values. These values are highly influenced by

engine rpm., throttle setting, and load. This subroutine will be expanded to consider these items.

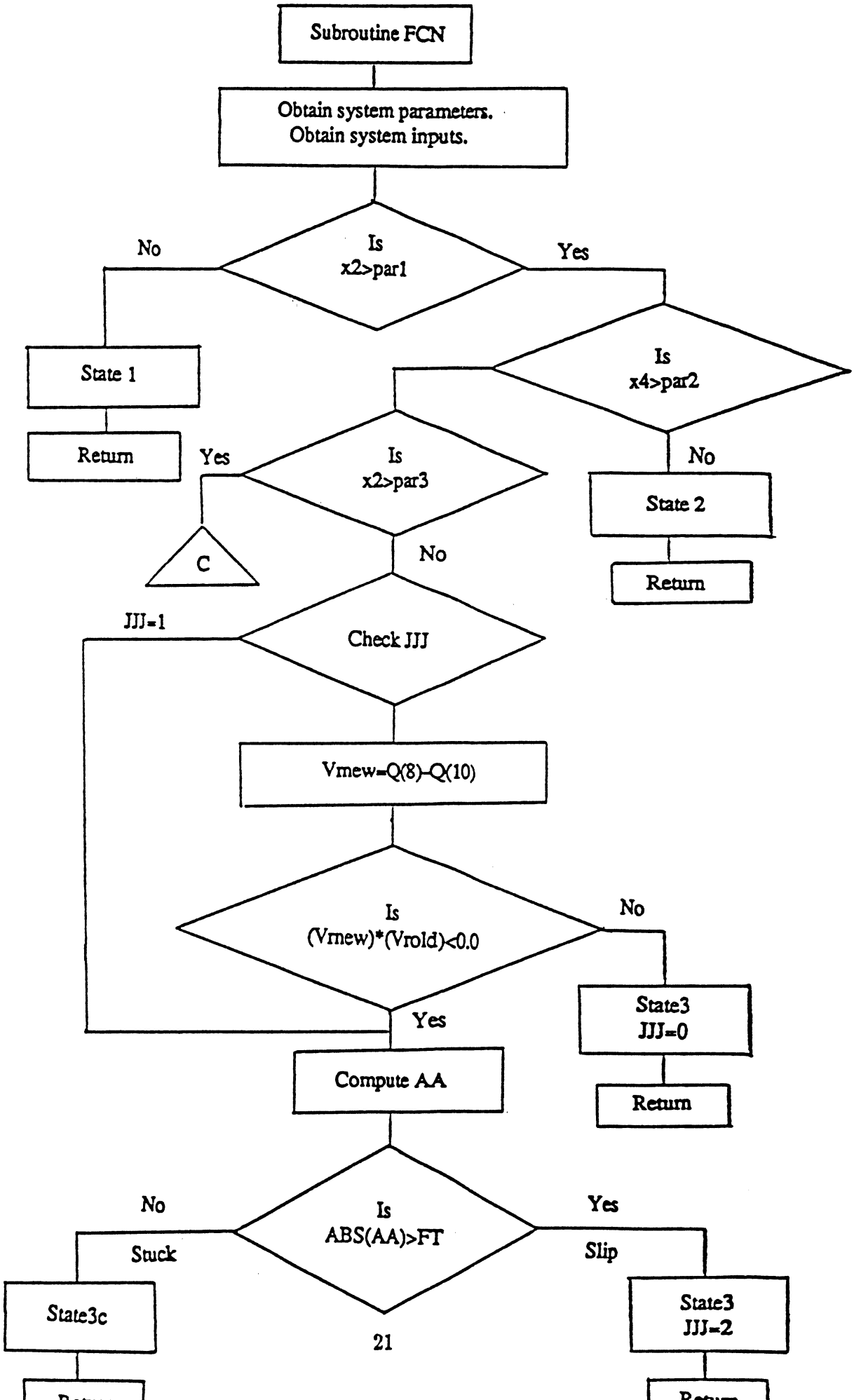
### 3.e Data Base Subroutines Called by State Subroutines

(1) BELSPR — This subroutine computes values for the release bearing stiffness rate, damping rate, and hysterisis effects as functions of system variables. These rates are designated by $k_2$ and $c_2$ respectively. Non-linearity would require the use of equations whose coefficients are determined from MODIFY and LSTSQR. This is left for future work.

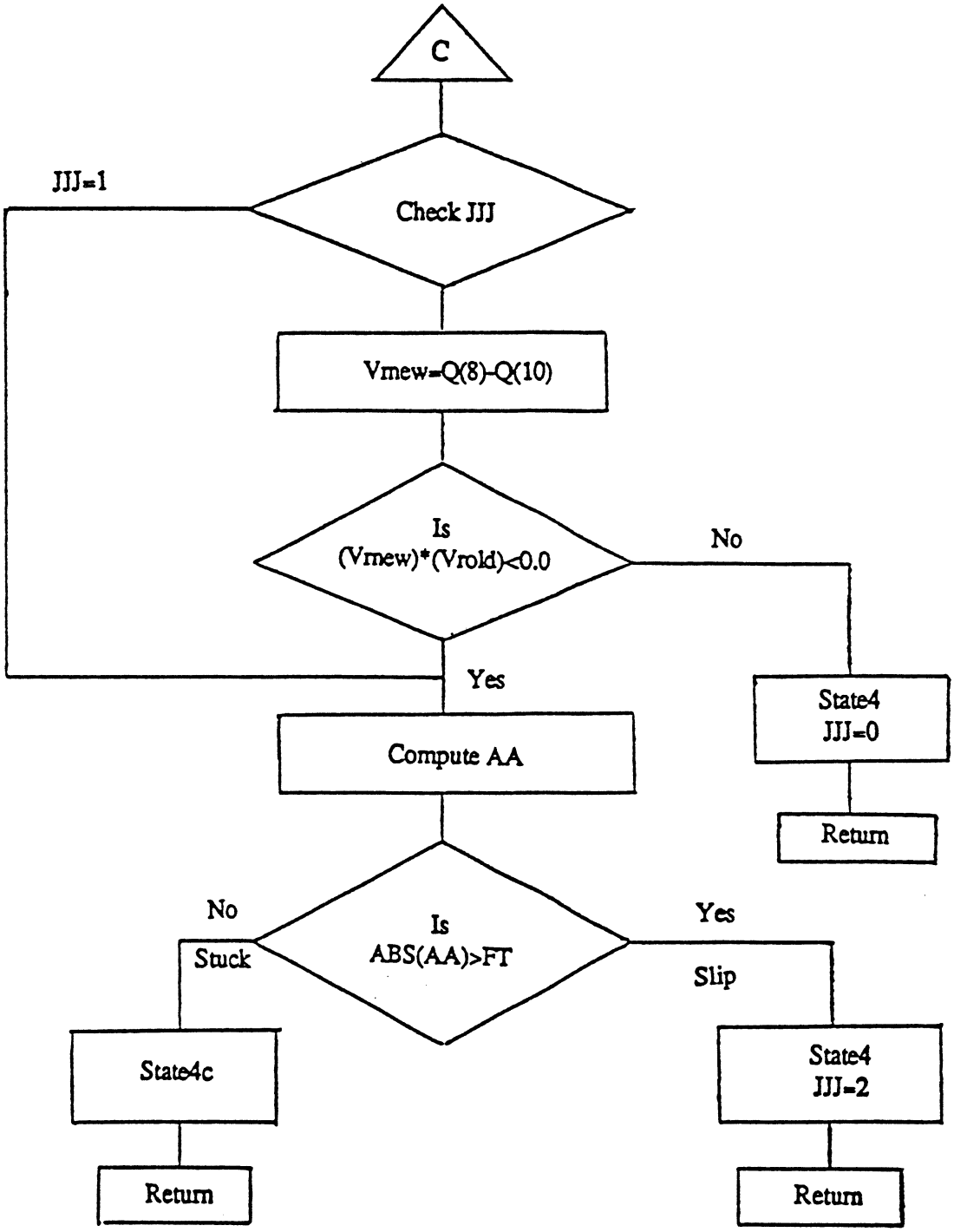
(2) MARCEL — Similar to BELSPR, MARCEL employs techniques which compute values of the cushion spring rate and damping factor as functions of system variables. These rates are referred to as $k_3$ and $c_3$ respectively and are in general, non-linear.

(3) ENGINE — Computations are made for engine mouting spring rates and damping factors.

## 4. PROGRAM FLOW CHART



## 4. PROGRAM FLOW CHART(continued)



## 5. GRAPHICAL STATE REPRESENTATION

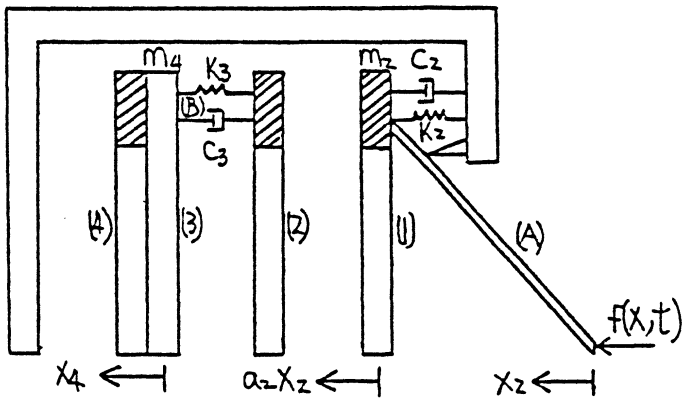
| state number | contact between first friction pad and pressure plate | contact between second friction pad and flywheel | inner clutch components fully compressed | slipping between first friction pad and pressure plate | slipping between second friction pad and flywheel |
|---|---|---|---|---|---|
| #1 | NO | NO | — | — | — |
| #2 | YES | NO | NO | YES | — |
| #3 | YES | YES | NO | YES | YES |
| #3C | YES | YES | NO | NO | NO |
| #4 | YES | YES | YES | YES | YES |
| #4C | YES | YES | YES | NO | NO |

Note: As far as the translational equations are concerned, states #3 and #3C are the same state. Also, translational equations do not exist for states #4 and #4C. This is due to the fact that for these states, translational displacement of the clutch components does not exist, since the inner clutch components are fully compressed. This is due to physical constraint.

As far as rotational equations are concerned, states #3 and #4 are the same; states #3C and #4C are the same.

# 6. TRANSLATIONAL MODEL

## 6.a Model Description and Parameters




(1) —Pressure plate

- mass=$m_2$
- displacement=$a_2 x_2$; where $a_2$ is a displacement coefficient.

(2) —First friction pad

- mass=negligible
- displacement=irrelevant (for state #1)
- displacement=$a_2 x_2$ (for other states)

(3) —Outer and inner hub assembly

- mass=$m_4$
- displacement=$x_4$
- coefficient of friction at interface=0.0

(4) —Second friction pad

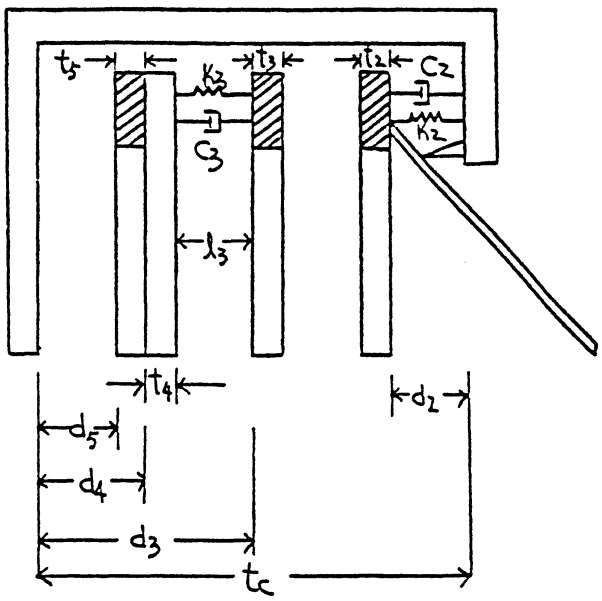
- mass=negligible
- displacement=irrelevant (for all states)

(A) —Diaphragm spring (represented by lever A and $k_2$ and $c_2$)

- force input=$f(x_2, t)$, where $x_2$ is the displacement of clutch pedal.

(B) —Marcel or cushion (represented by $k_3$, $c_3$)

## 6.b Initial Clutch Positions and Physical Clutch Parameters




Initial Positions

$d_2$=initial distance between pressure plate and cover

$d_3$=initial distance between first friction pad and flywheel

$d_4$=initial distance between clutch hub and flywheel

$d_5$=initial distance between second friction pad

Physical Parameters

$l_3$=free length of k (marcel)

$t_c$=thickness of entire clutch

$t_2$=thickness of pressure plate

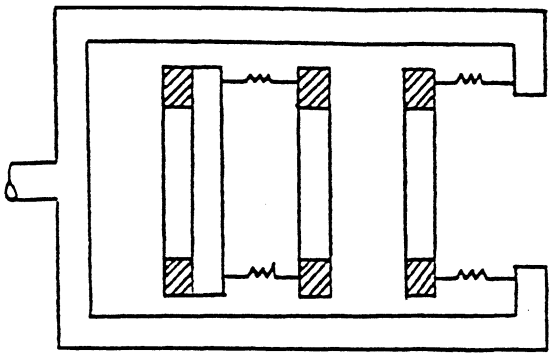
$t_3=t_5$=thickness of friction pad

$t_4$=thickness of marcel

Initial positions are determined by user.

## 6.c System States

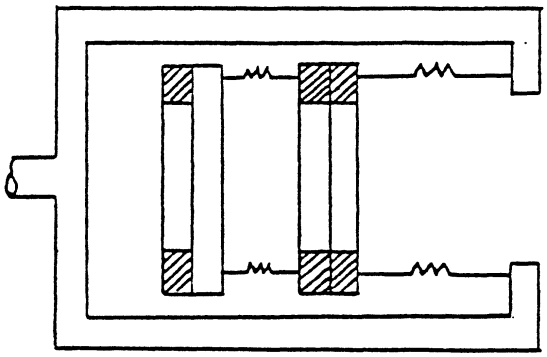
State #1



—From clutch pedal fully depressed to point where pressure plate is just about to contact the first friction pad.

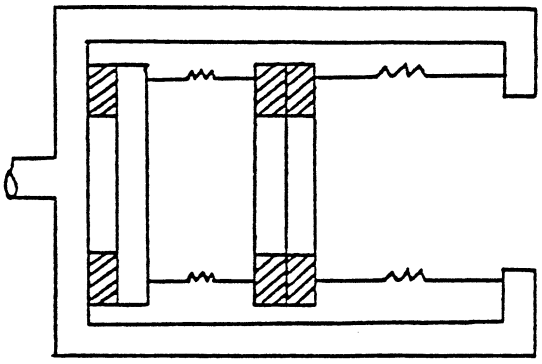
—Engine at idle speed.

State #2



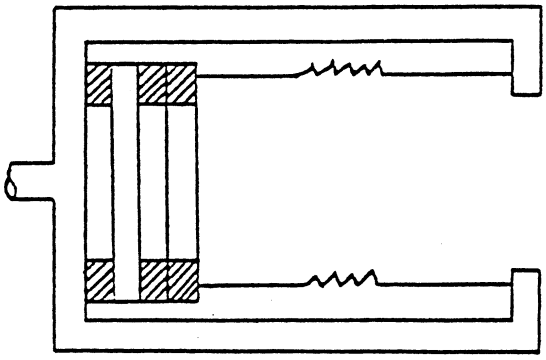
—From when the pressure plate contacts the first friction pad to the point where the second friction pad is just about to contact the flywheel.

State #3, #3C



—From when the second friction pad contacts the flywheel, to the point where the inner clutch assembly is just about compressed (i.e. clutch pedal is just about fully released)

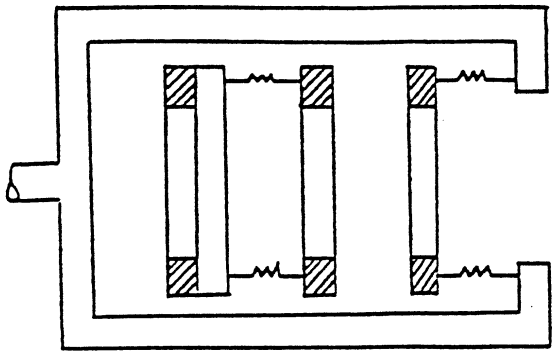
States #4, #4C



—When the inner clutch pedal assembly is fully compressed (i.e. clutch pedal is fully released)

## 6.d Translational Equations of Motion Summary

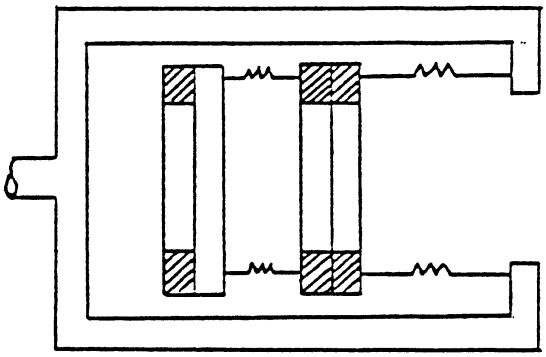
State #1



$$m_2 a_2 \ddot{x}_2 + c_2 a_2 \dot{x}_2 + k_2 a_2 x_2 = k_2 THKT - F_{input}$$

where THKT=constant=9.34

State #2



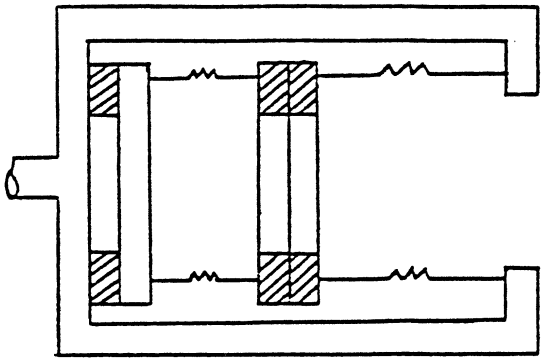
$$m_2 a_2 \ddot{x}_2 + (c_2 + c_3) a_2 \dot{x}_2 + (k_2 + k_3) a_2 x_2 = k_2 THKT - F_{input} - k_3 (D_t - x_4) + c_3 \dot{x}_4$$

$$m_4 \ddot{x}_4 + c_3 \dot{x}_4 + k_3 x_4 = c_3 a_2 \dot{x}_2 + k_3 (D_t + a_2 x_2)$$

where THKT=9.34, $D_t$=1.8

## 6.d Translational Equations of Motion Summary (continued)

State #3



$$m_2 a_2 \ddot{x}_2 + (c_2 + c_3) a_2 \dot{x}_2 + (k_2 + k_3) a_2 x_2 = k_2 THKT - F_{input} - k_3 (D_t - x_4)$$

$$x_4 = 1.0$$

where THKT=9.34, $D_t$=1.8

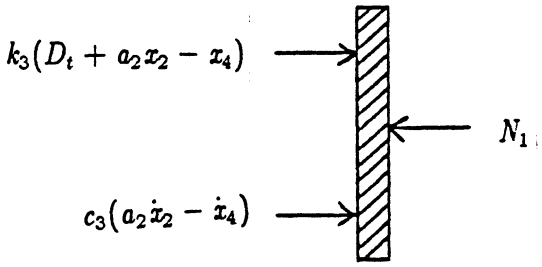
THKT=Temporary constant used to calculate release bearing load due to linear spring $k_2$.

$D_t$=Initial gap between clutch pedal pad and flywheel plus compressed clutch width.

## Calculations of $N_1$ and $N_2$

State #2, #3, #3C

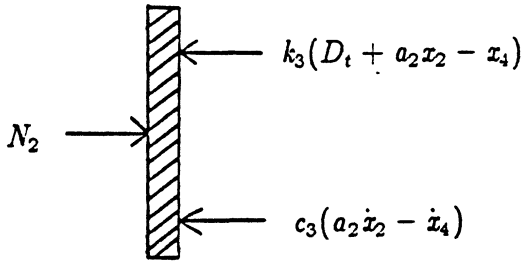
Free-body diagram of first friction pad




$$N_1 = k_3(D_t + a_2x_2 - x_4) + c_3(a_2\dot{x}_2 - \dot{x}_4)$$

State #3, #3C

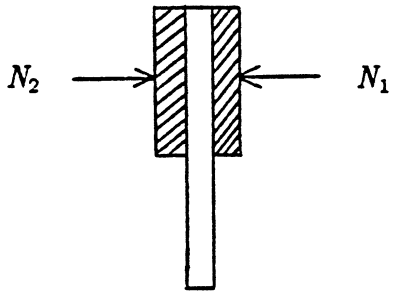
Free-body diagram of the second friction pad




$$N_1 = N_2$$

State #4, #4C

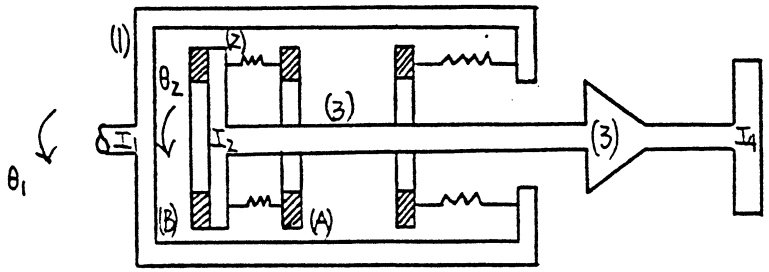
Free-body diagram of hub assembly and both friction pad




$$N_1 = N_2 = k_2(x_2)_{max}$$

# 7. ROTATIONAL MODEL

## 7.a Model Description and Parameters




(1) —Flywheel, cover,and pressure plate

- mass moment of inertia=$I_1$
- angular displacement=$\theta_1$

(2) —Outer hub

- mass moment of inertia=$I_2$
- angular displacement=$\theta_2$

(3) —Inner hub and transmission

- mass moment of inertia=$I_3$

(4) —Drivetrain (from output of transmission to rear wheel)

- mass moment of inertia=$I_4$

(A) —First friction pad

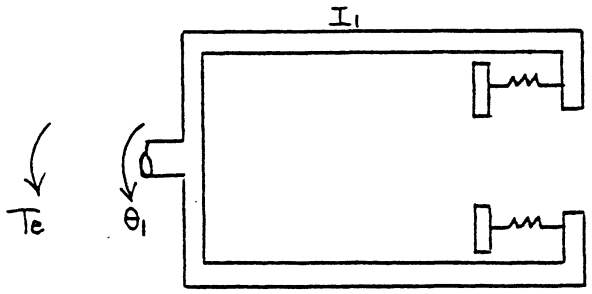
- mass moment of inertia=negligible
- angular displacement=irrelevant

(B) —Second friction pad

- mass moment of inertia=negligible
- angular displacement=irrelevant

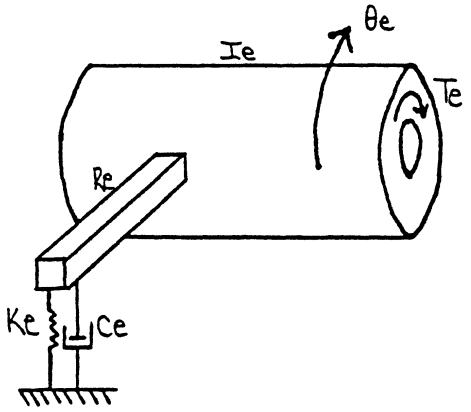
## 7.b Rotational Equations of Motion: Derivation for state #1

$\theta_1$ :




$$I_1\ddot{\theta}_1 = Te$$

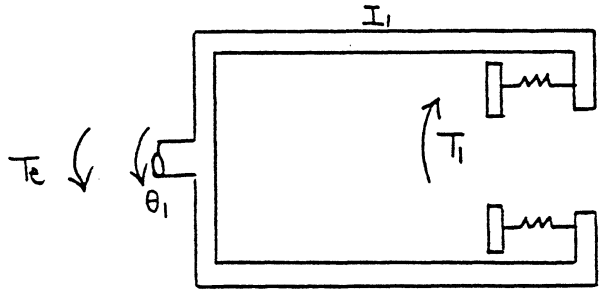
$\theta_e$ :




$$I_e\ddot{\theta}_e + c_e r_e\dot{\theta}_e + k_e r_e\theta_e = T_e$$

## 7.b Rotational Equations of Motion: Derivation for state #2

$\theta_1$ :

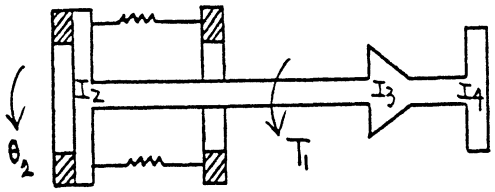



$$I_1\ddot{\theta}_1 = T_e - T_1$$

$$T_1 = \mu_f N_1 r_c$$

$$N_1 = k_3(D_t + a_2 x_2 - x_4) + c_3(a_2\dot{x}_2 - \dot{x}_4)$$

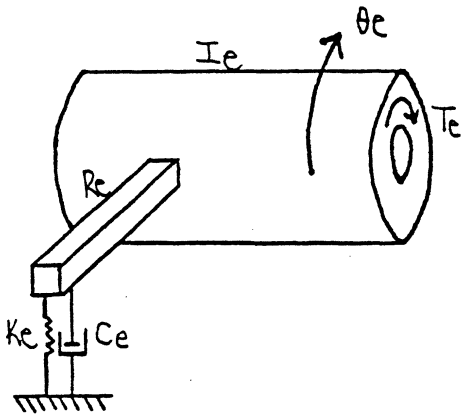
$\theta_2$ :




$$(I_2 + I_3 + I_4)\ddot{\theta}_2 = T_1$$

$$T_1 = \mu_f N_1 r_c$$

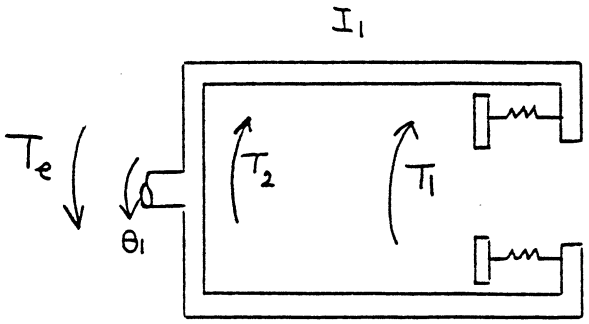
$\theta_e$ :




$$I_e\ddot{\theta}_e + c_e r_e \dot{\theta}_e + k_e r_e \theta_e = T_e$$

## 7.b Rotational Equations of Motion: Derivation for states #3 and #4

$\theta_1$ :



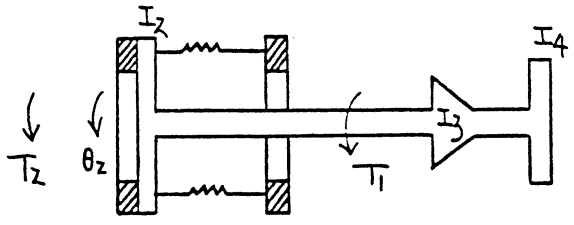

$$T_1 = \mu_f N_1 r_c$$

$$T_2 = \mu_f N_2 r_c$$

$$N_1 = N_2$$

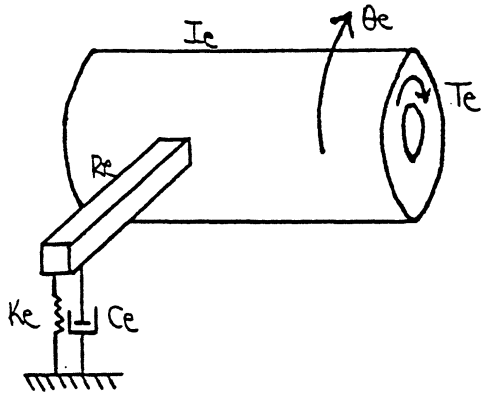
$$I_1 \ddot{\theta}_1 = T_e - 2\mu_f N_1 r_c$$

$\theta_2$ :




$$(I_2 + I_3 + I_4)\ddot{\theta}_2 = T_1 + T_2 = 2\mu_f N_1 r_c$$

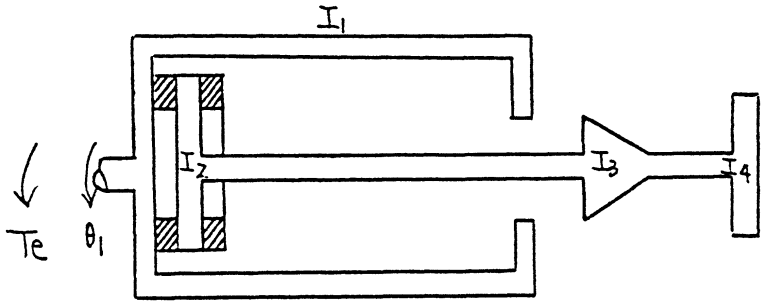
$\theta_e$ :




$$I_e\ddot{\theta}_e + c_e r_e\dot{\theta}_e + k_e r_e\theta_e = T_e$$

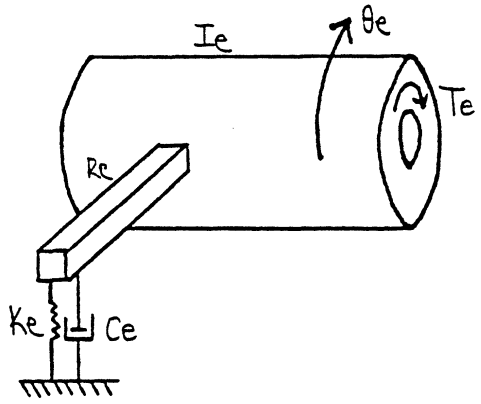
## 7.b Rotational Equations of Motion: Derivation for states #3C and #4C

$\theta_1$ :




$$(I_1 + I_2 + I_3 + I_4)\ddot{\theta}_1 = T_e$$

$\theta_e$ :




$$I_e\ddot{\theta}_e + c_e r_e \dot{\theta}_e + k_e r_e \theta_e = T_e$$

# 8. ANALYSIS OF COULOMB FRICTION

The model shown in Fig.1 was selected to study coulomb friction. The idealized relationship between $V_r$ (relative velocity) and $F_f$ (friction force) shown in Fig.2 is assumed. Thus we can state that

$$F_f = -\mu N sign\, V_r.$$

The case of $V_r = 0$, that is sticking deserves some comment. During the motion of the masses $m_1$ and $m_2$, two possible events can happen, either the two masses will stick to each other, or the two masses will slide. If the two masses start to slide, then the detection of the correct direction of $F_f$ is important. Table 1 summarizes the procedure to detect sliding and to find the correct direction of friction force for an equivalent translational model.

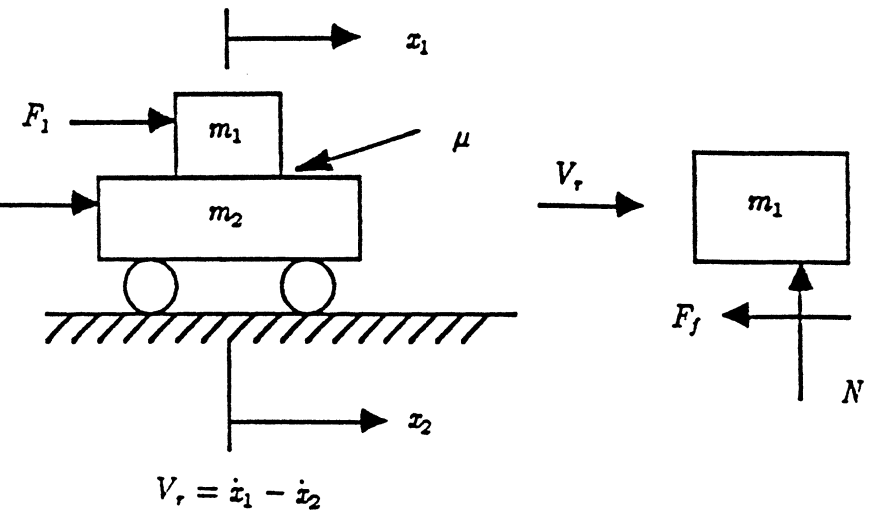



Fig 1

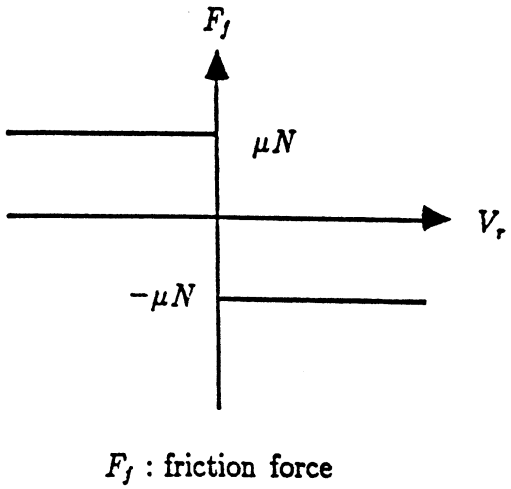



Fig 2

| $\left\| \frac{F_1 + F_2}{m_1 + m_2} - \frac{F_1}{m_1} \right\|$ | $SIG(\frac{F_1 + F_2}{m_1 + m_2} - \frac{F_1}{m_1})$ | Sliding or stiction<br>In case of sliding<br>FBD for mass $m$ |
|---|---|---|
| $> \left\| \frac{\mu N}{m_1} \right\|$ | Positive | sliding<br>$x_1$<br>$F_1$<br>$m_1$<br>$\mu N$ |
| $> \left\| \frac{\mu N}{m_1} \right\|$ | Negative | sliding<br>$x_1$<br>$F_1$<br>$m_1$<br>$\mu N$ |
| $< \left\| \frac{\mu N}{m_1} \right\|$ | | stiction |

Table 1

## 9. RESULTS

Two cases are investigated and a brief description of each follows:

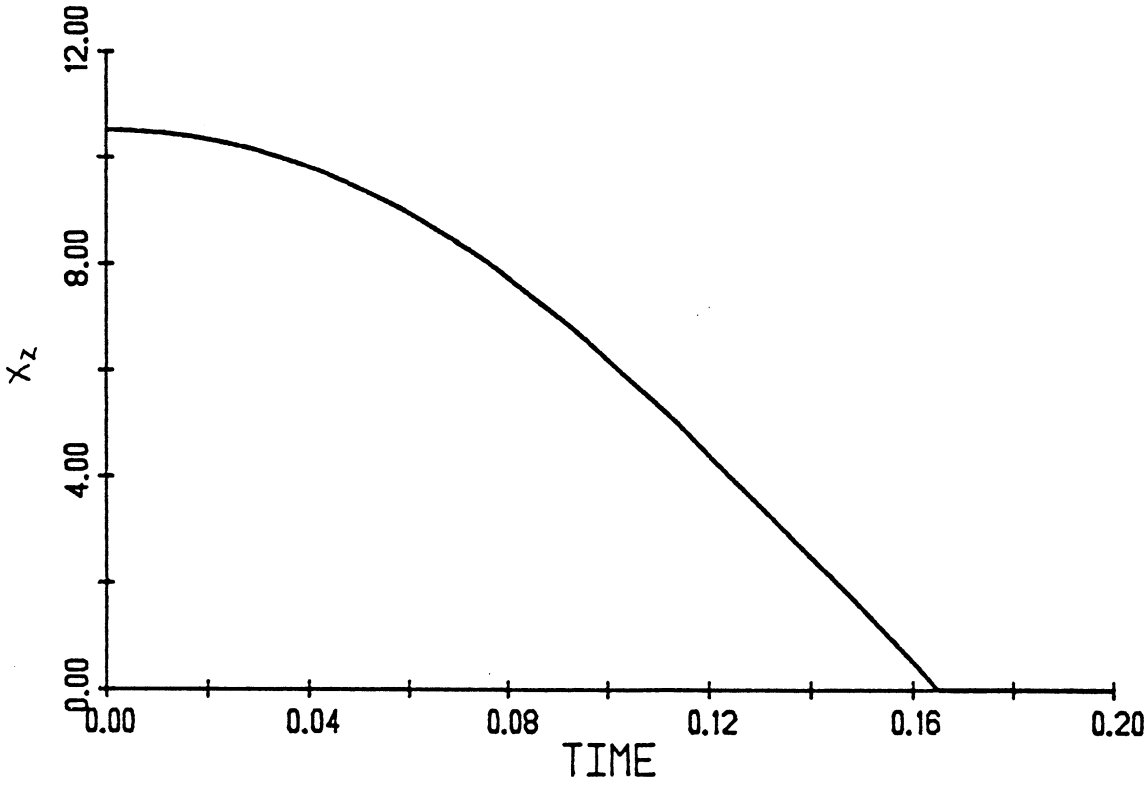
Case 1 shows the engagement of a clutch as it goes through all the different states. The initial conditions and system constants are listed in Table 2. The system's motion occurs sequentially from State 1 to State 4. Sticking occurs in State 3. The results obtained in States 1,2 and 4 have been checked against exact analytical solutions, and excellent agreement has been observed.

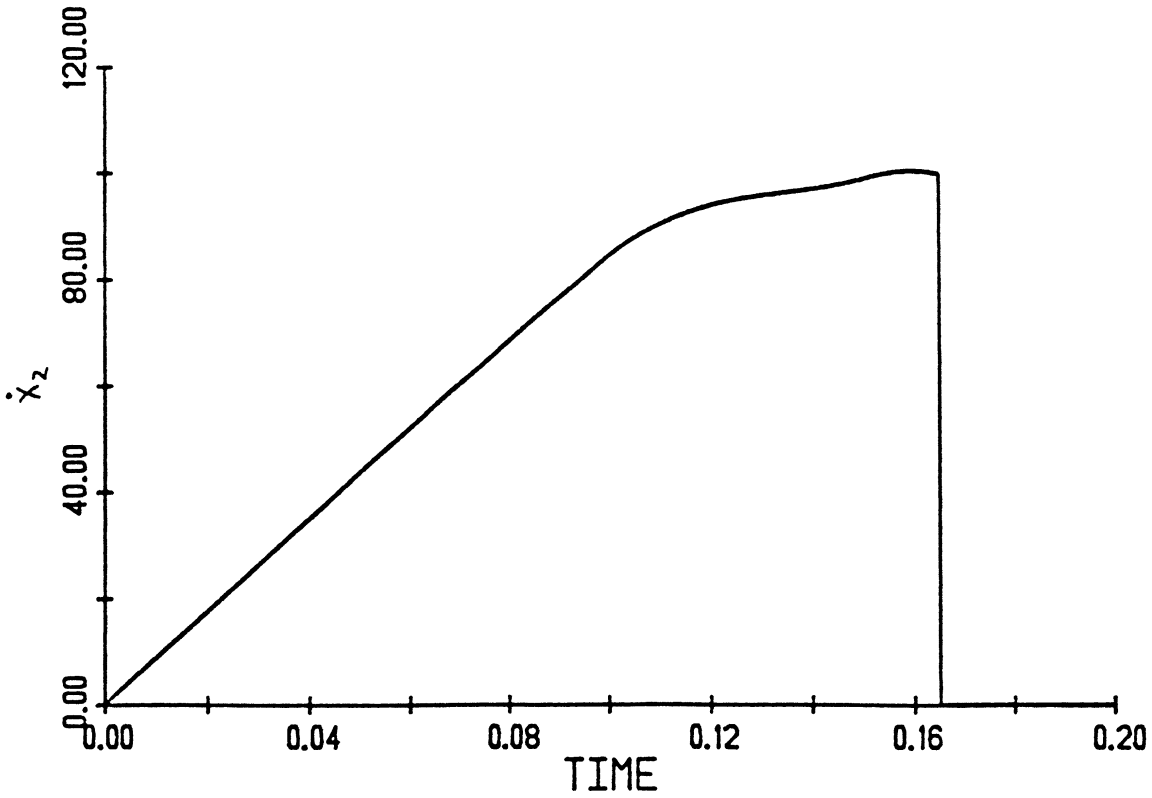
In Case 2, a constant step torque of 77000 is added to the original engine torque of 10000 at State 4. The initial conditions of State 4 are obtained from the final conditions of Case 1, and are listed in Table 2. The step torque is applied at time step number 400, or at $t = .004$ second. All the system constants are the same as in Case 1. Motion only occurs in rotational State 4. At first, the interface surfaces are stuck. When the step torque is applied, the friction force is overcome and slipping occurs. The step torque necessary to overcome the friction force has been determined theoretically and confirmed by numerical results.

Table 2

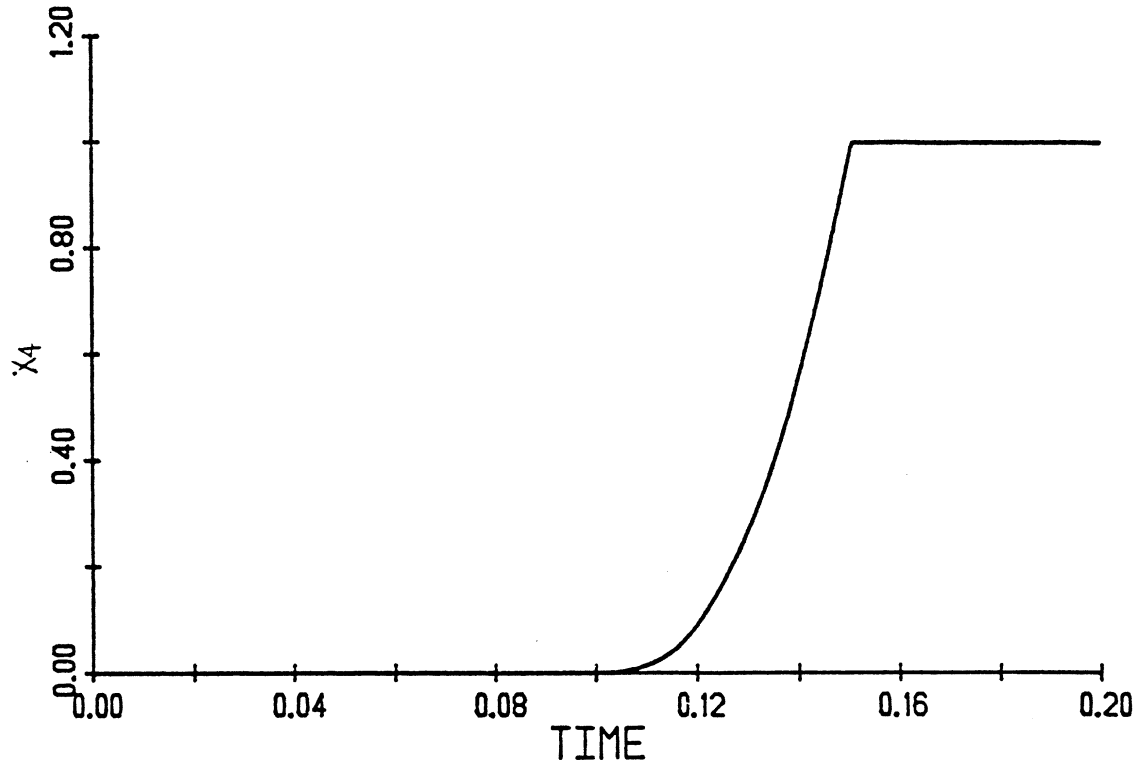
| | Case 1 | Case 2 |
|---|---|---|
| | Initial Conditions | |
| $x_2$ | -10.5 | 0.0 |
| $\dot{x}_2$ | 0.0 | 0.0 |
| $x_4$ | 0.0 | 1.0 |
| $\dot{x}_4$ | 0.0 | 0.0 |
| $\theta_e$ | 0.0 | 0.01711 |
| $\dot{\theta}_e$ | 0.0 | -0.1987 |
| $\theta_1$ | 0.0 | 10.86 |
| $\dot{\theta}_1$ | 52.36 | 52.76 |
| $\theta_2$ | 0.0 | 3.626 |
| $\dot{\theta}_2$ | 0.0 | 52.77 |
| | System Constants | |
| $T_e$ | 10000.0 | 10000.0 |
| $F_{input}$ | 0.0 | 0.0 |
| $k_2$ | 200.0 | 200.0 |
| $c_2$ | 0.0 | 0.0 |
| $k_3$ | 3000.0 | 3000.0 |
| $c_3$ | 0.0 | 0.0 |
| $k_e$ | 2000.0 | 2000.0 |
| $c_e$ | 10.0 | 10.0 |

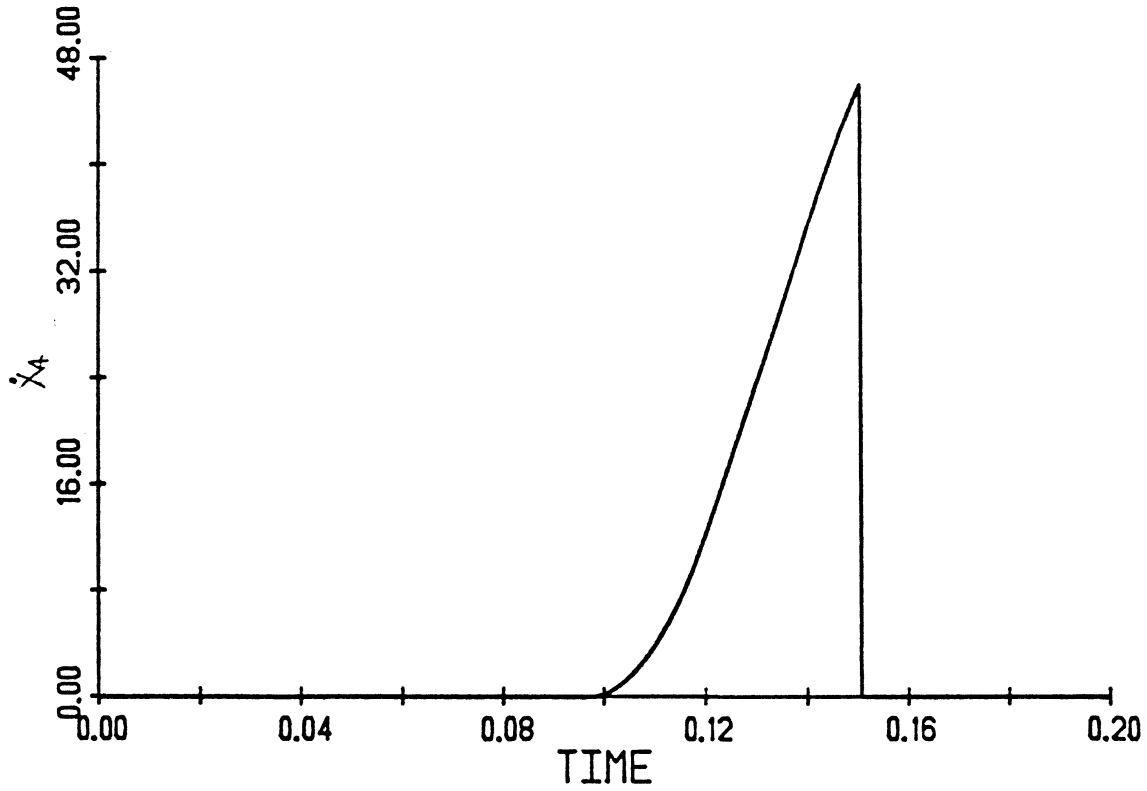
Case 1


12.00
8.00
4.00
0.00
$x_2$
0.00
0.04
0.08
0.12
0.16
0.20
TIME

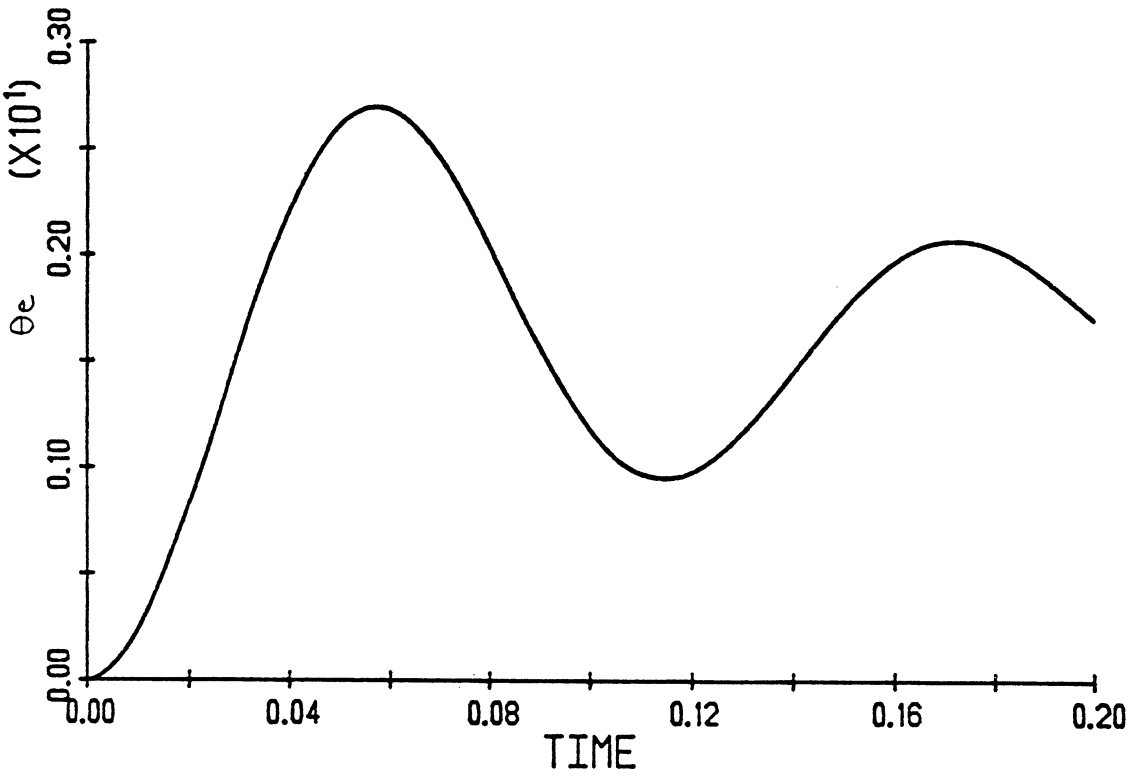

120.00
80.00
40.00
0.00
$\dot{x}_2$
0.00
0.04
0.08
0.12
0.16
0.20
TIME

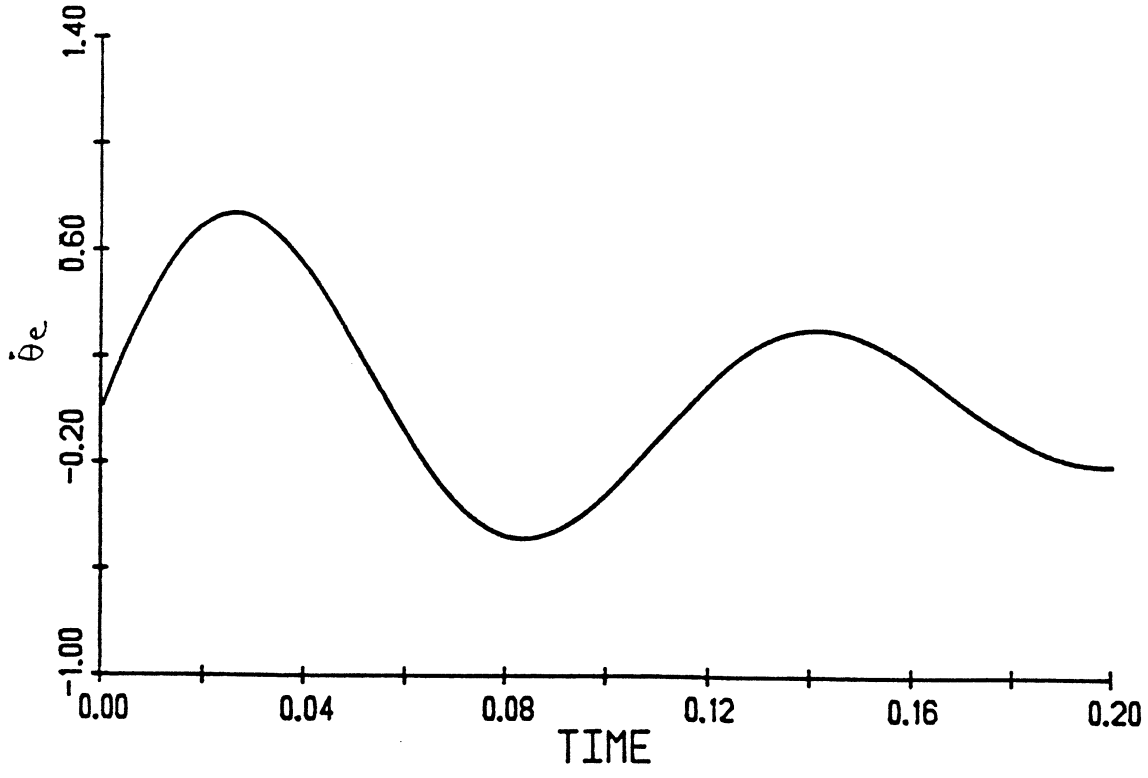
Case 1


1.20
0.80
0.40
0.00
$\dot{\chi}_4$
0.00
0.04
0.08
0.12
0.16
0.20
TIME

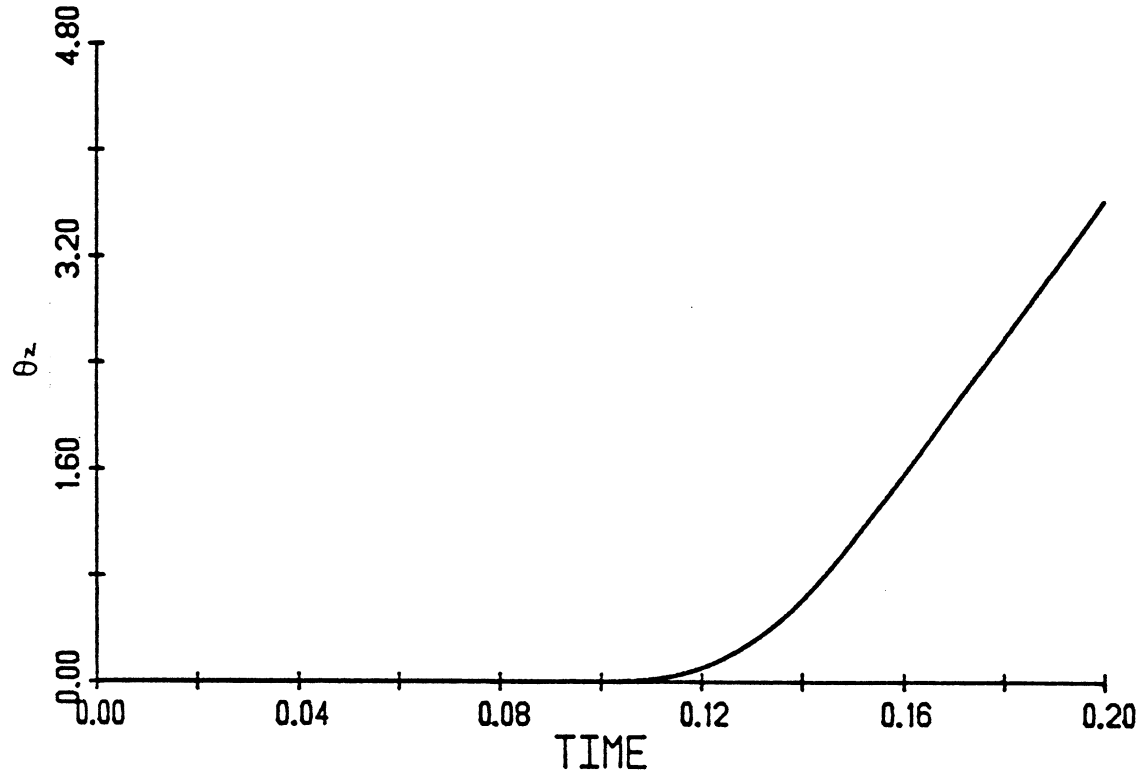

48.00
32.00
16.00
0.00
$\dot{\chi}_4$
0.00
0.04
0.08
0.12
0.16
0.20
TIME

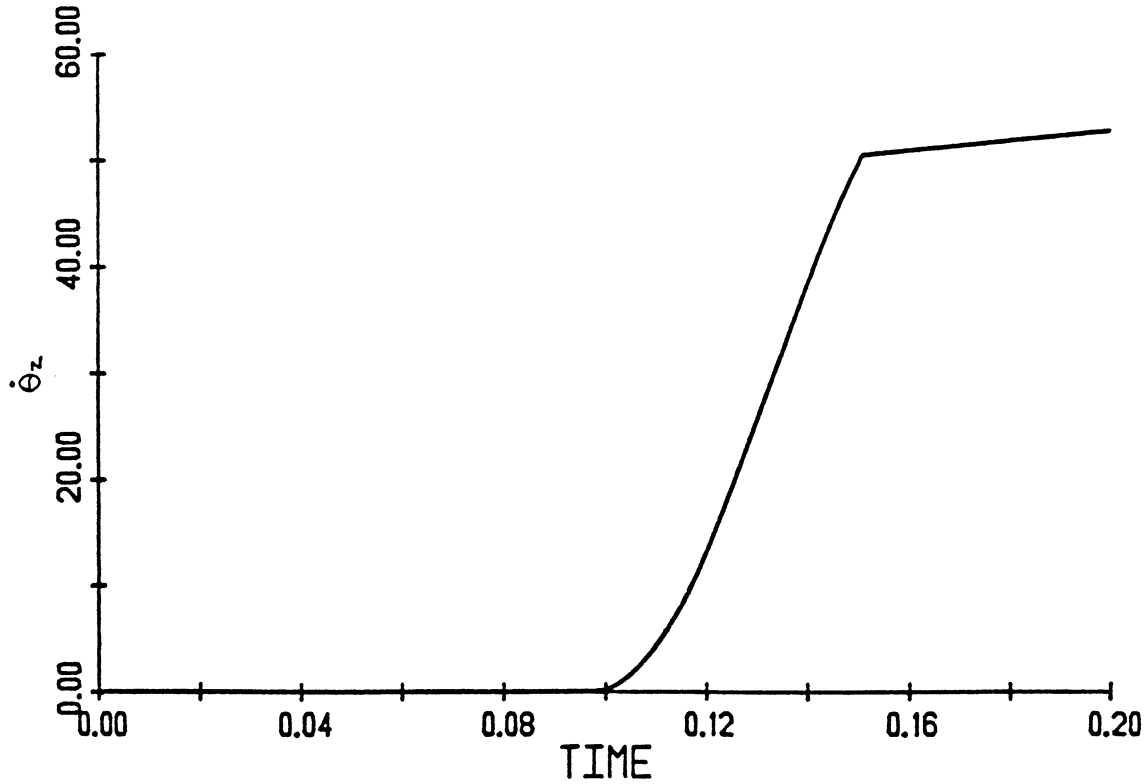
Case 1


$\theta_e$ (X10$^1$)
0.30
0.20
0.10
0.00
0.00
0.04
0.08
0.12
0.16
0.20
TIME

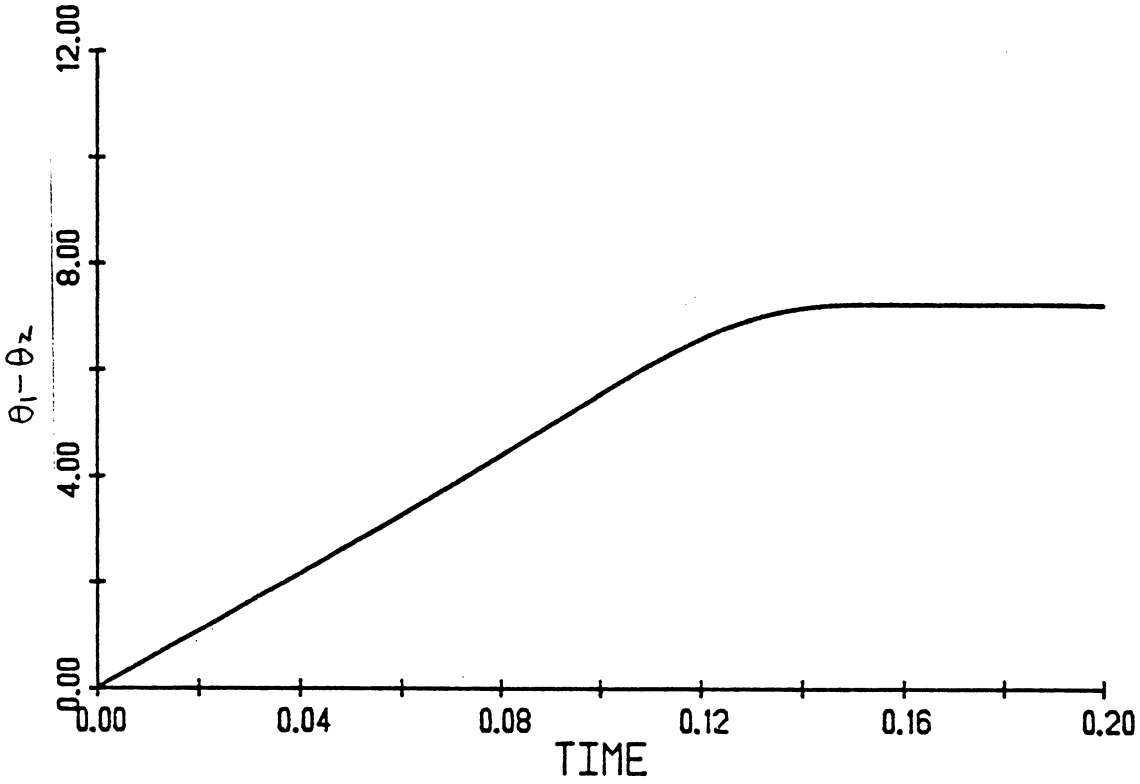

$\dot{\theta}_e$
1.40
0.60
-0.20
-1.00
0.00
0.04
0.08
0.12
0.16
0.20
TIME

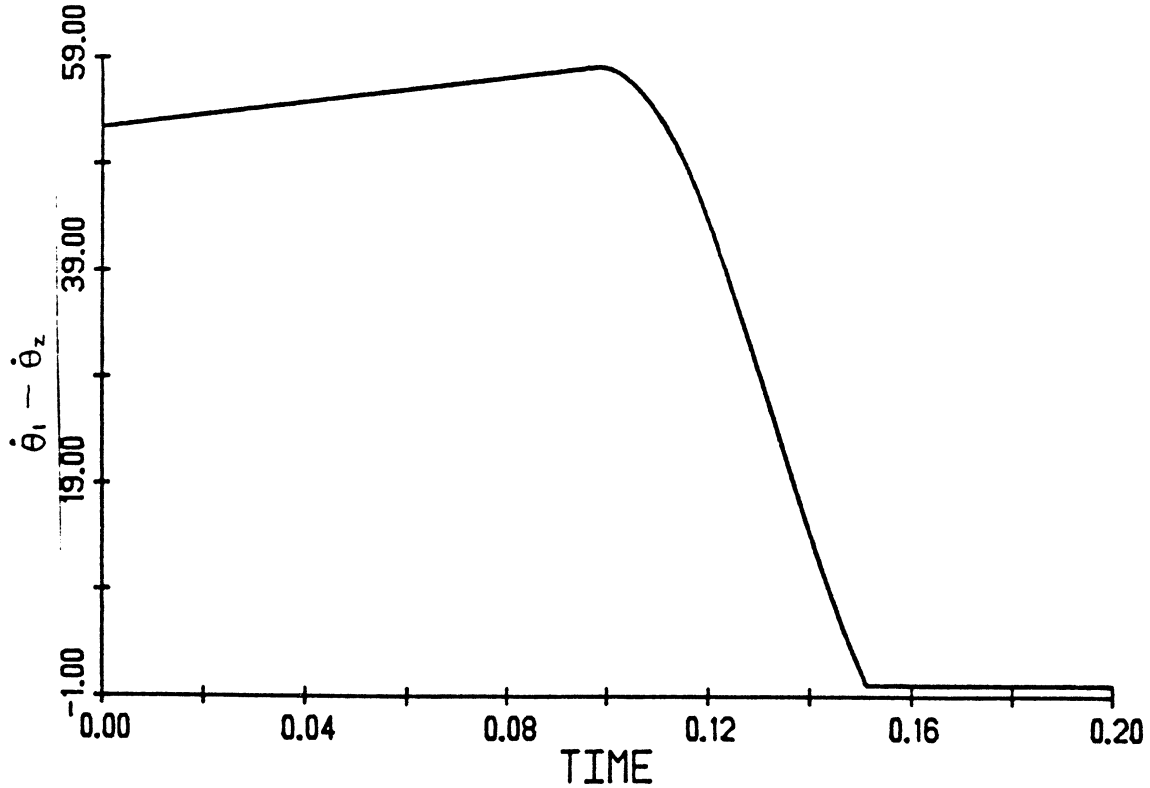
Case 1


4.80
3.20
1.60
0.00
θz
0.00
0.04
0.08
0.12
0.16
0.20
TIME

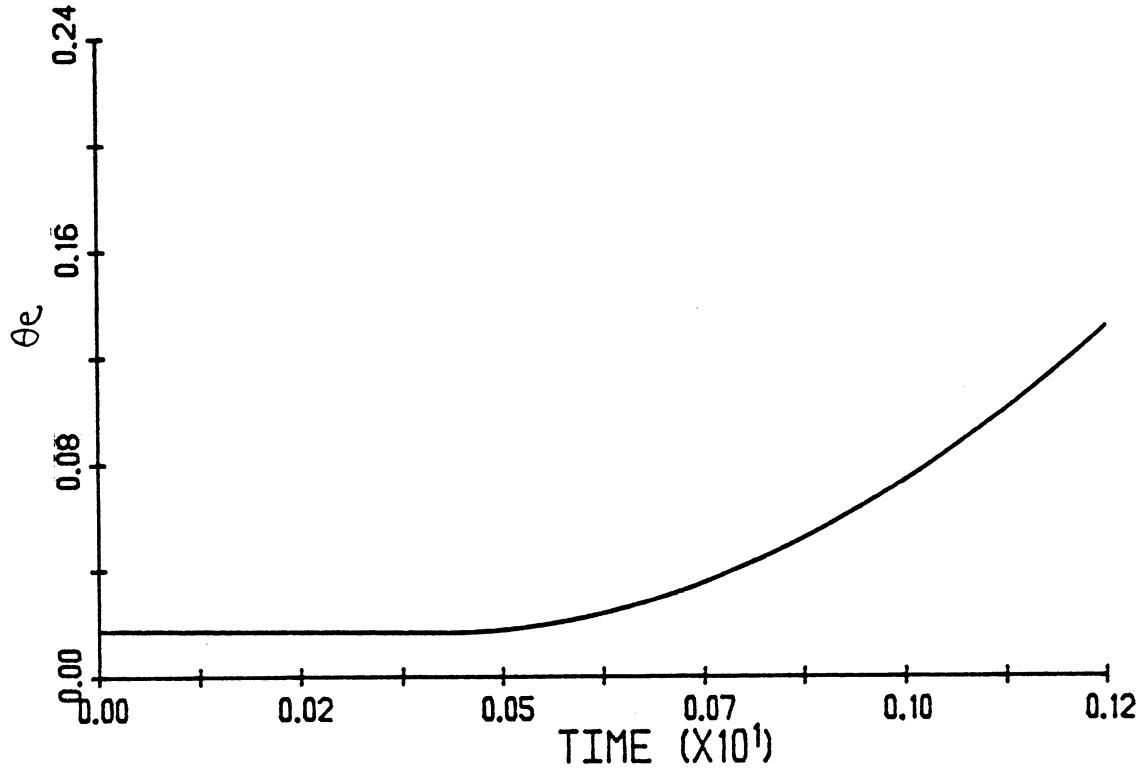

60.00
40.00
20.00
0.00
θ̇z
0.00
0.04
0.08
0.12
0.16
0.20
TIME

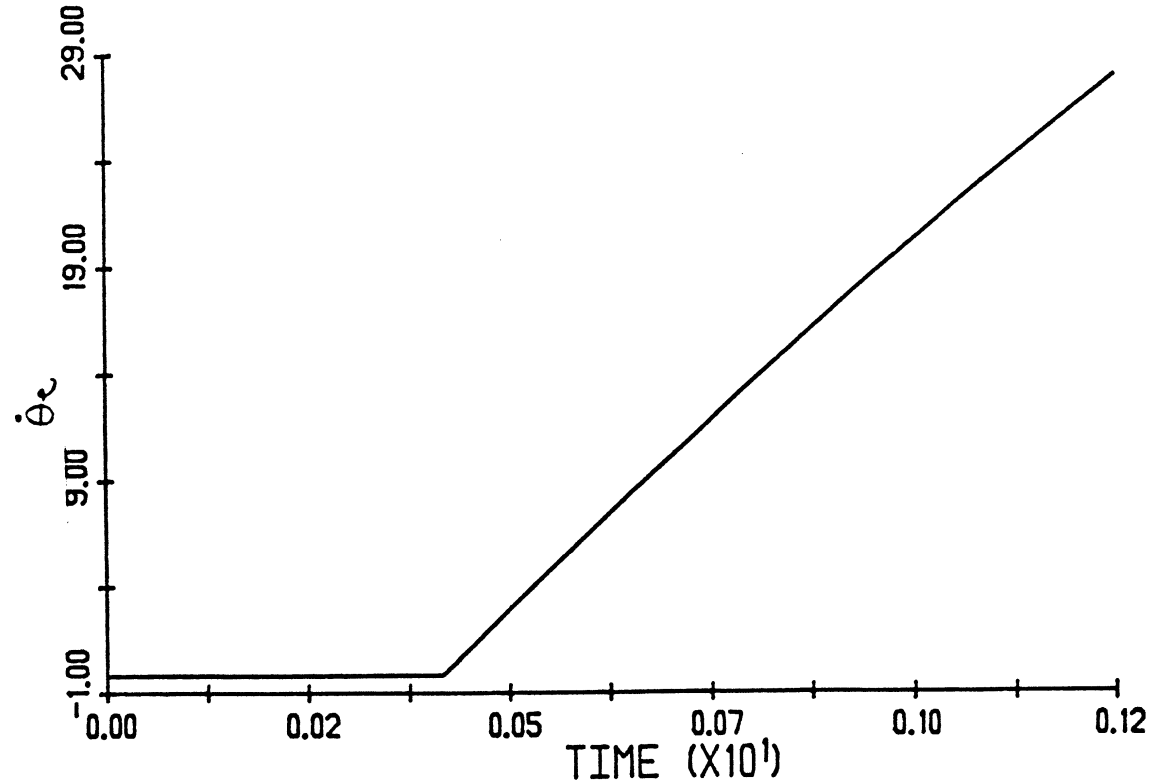
Case 1


12.00
8.00
4.00
0.00
$\theta_1 - \theta_z$
0.00
0.04
0.08
0.12
0.16
0.20
TIME

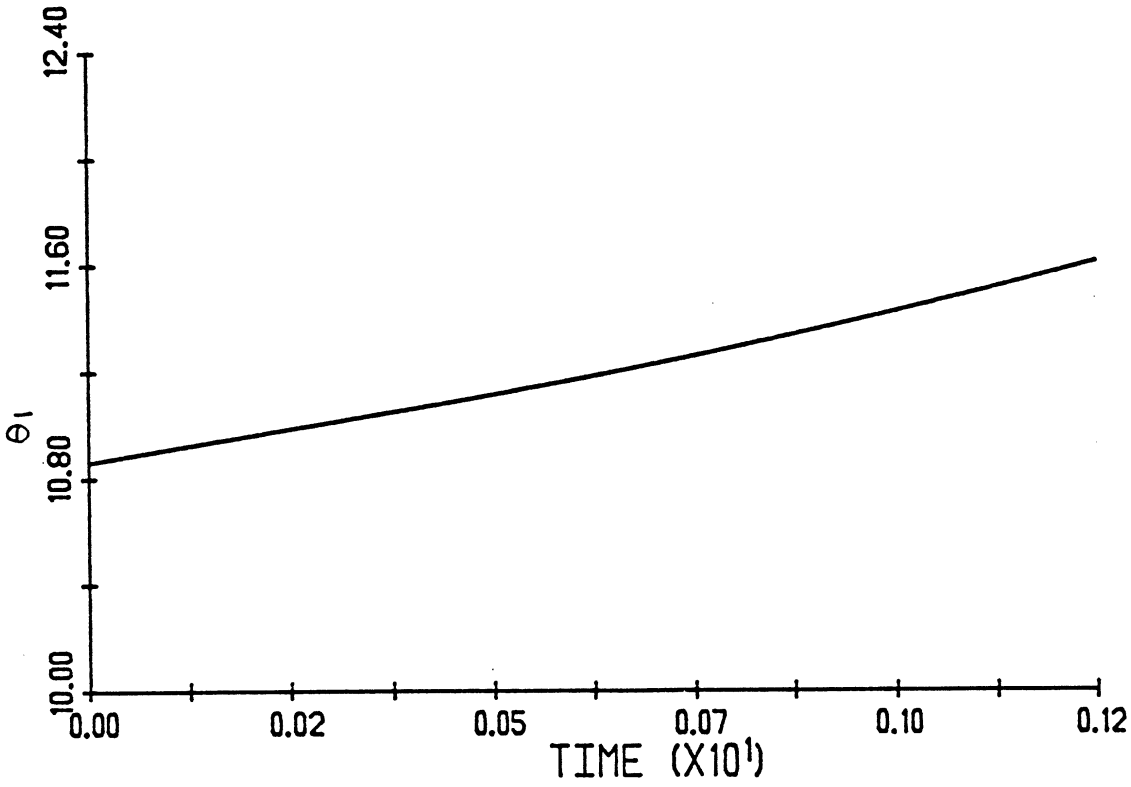

59.00
39.00
19.00
-1.00
$\dot{\theta}_1 - \dot{\theta}_z$
0.00
0.04
0.08
0.12
0.16
0.20
TIME

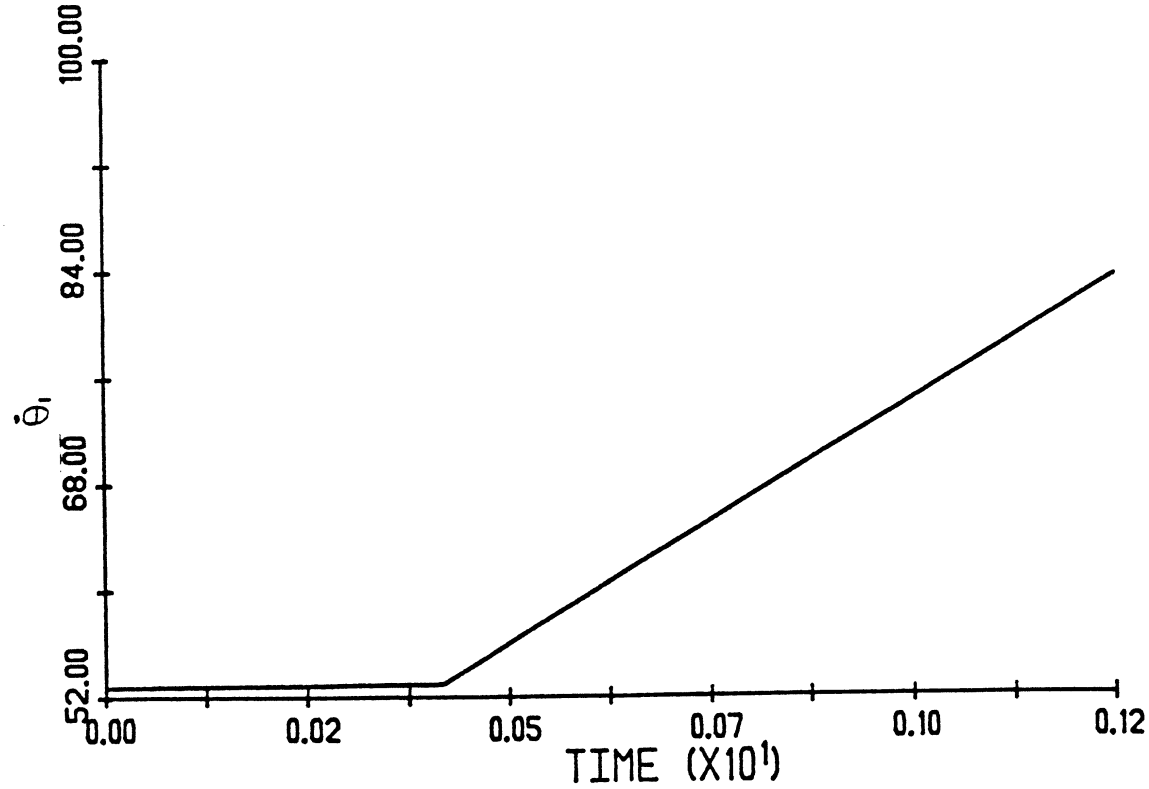
Case 2


0.24
0.16
0.08
0.00
θe
0.00
0.02
0.05
0.07
0.10
0.12
TIME (X10¹)

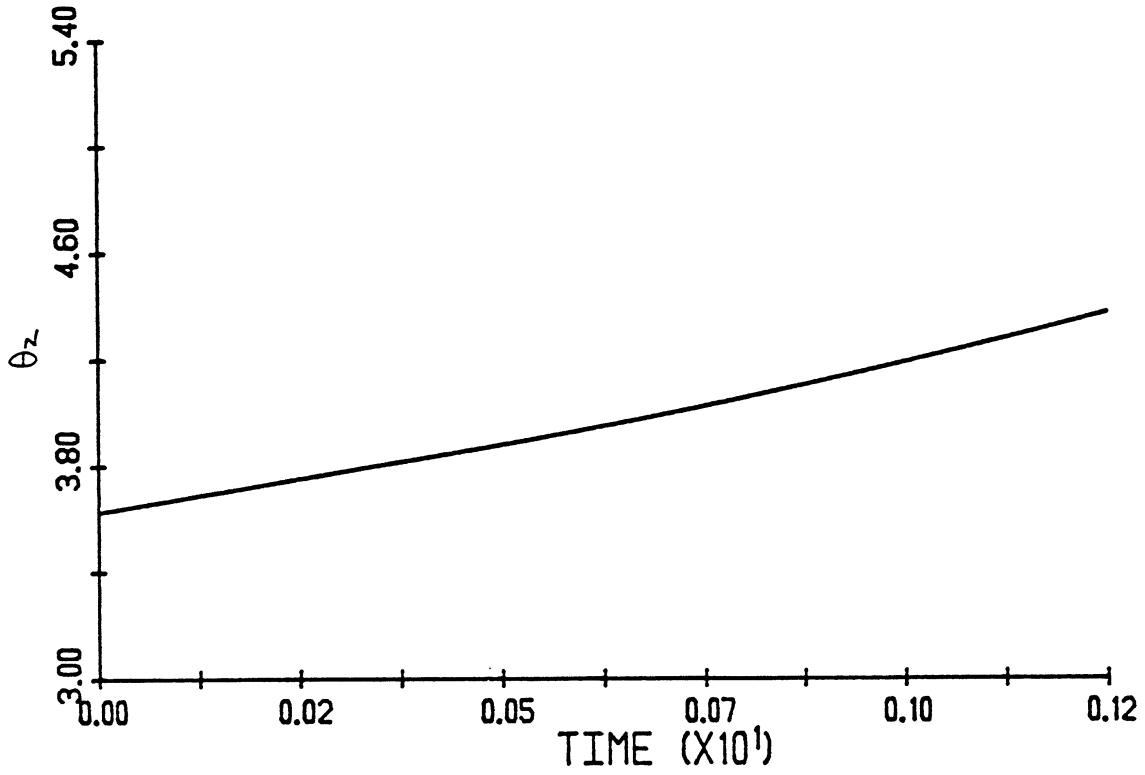

29.00
19.00
9.00
-1.00
θ̇e
0.00
0.02
0.05
0.07
0.10
0.12
TIME (X10¹)

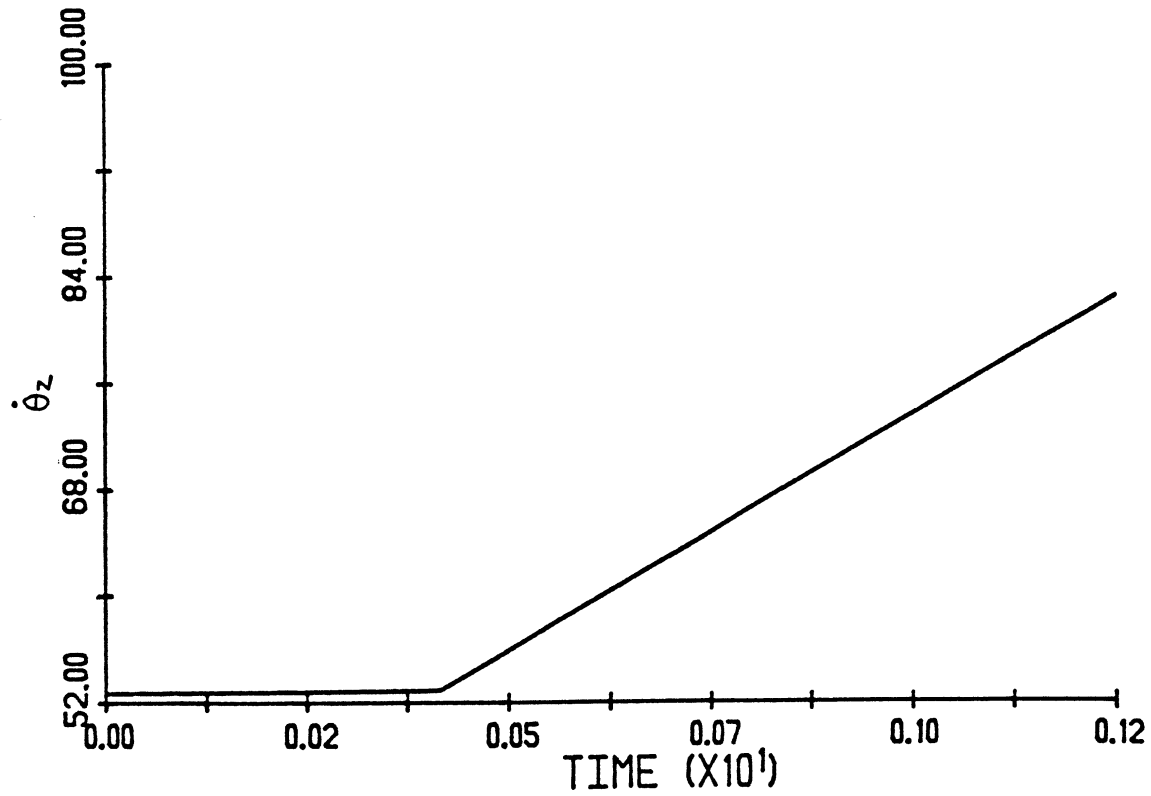
Case 2


12.40
11.60
10.80
10.00
$\theta_1$
0.00
0.02
0.05
0.07
0.10
0.12
TIME (X10$^1$)

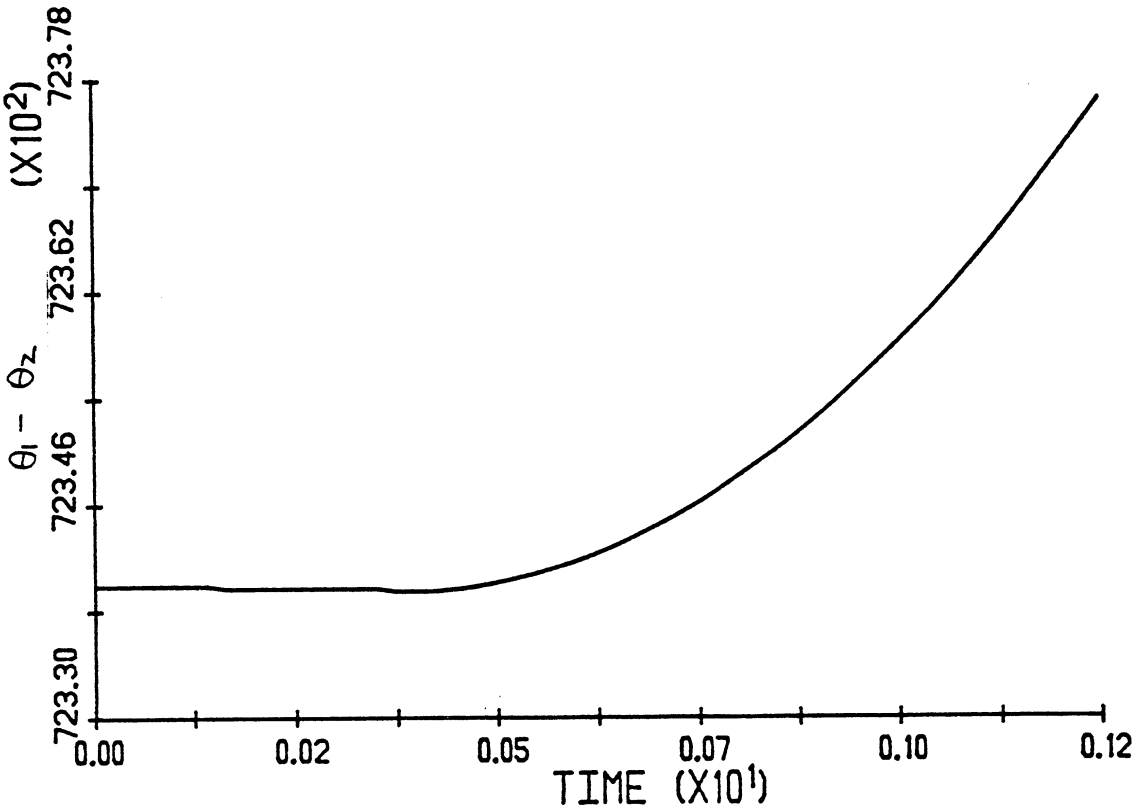

100.00
84.00
68.00
52.00
$\dot{\theta}_1$
0.00
0.02
0.05
0.07
0.10
0.12
TIME (X10$^1$)

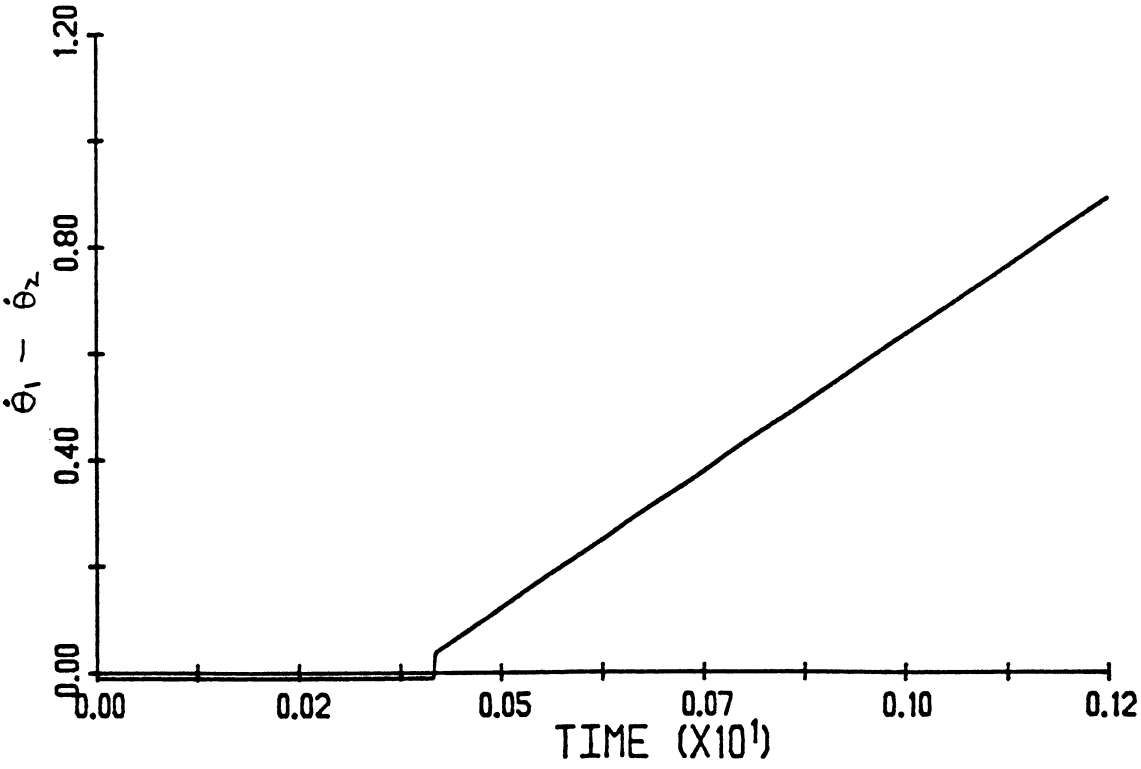
Case 2


5.40
4.60
3.80
3.00
θz
0.00
0.02
0.05
0.07
0.10
0.12
TIME (X10¹)


100.00
84.00
68.00
52.00
θ̇z
0.00
0.02
0.05
0.07
0.10
0.12
TIME (X10¹)

Case 2


$\theta_1 - \theta_2$ (X10²)
723.78
723.62
723.46
723.30
0.00
0.02
0.05
0.07
0.10
0.12
TIME (X10¹)


$\dot{\theta}_1 - \dot{\theta}_2$
1.20
0.80
0.40
0.00
0.00
0.02
0.05
0.07
0.10
0.12
TIME (X10¹)

# 10. PROGRAM LISTING

```
C*****************************************************************
C     MARCH 24, 1986
C     ========FOURTH ORDER RUNGE-KUTTA SCHEME===========
C
C   THIS PROGRAM GENERATES  A COMPUTATIONAL MODEL OF THE
C   AUTOMOTIVE CLUTCH SYSTEM. IT IS MEANT TO REDUCE PRESENT
C   "TRIAL AND ERROR" METHODS USED IN CLUTCH DESIGN WHICH
C   HAVE NOT ONLY PROVED TO BE EXPENSIVE AND TIME CONSUMING,
C   BUT DO NOT ALLOW THE ENGINEER TO REALIZE THE POTENTIAL FOR
C   THE IMPROVED AND OPTIMAL DESIGNS WHICH COULD BE ACHIEVED.
C
C   THE STRUCTURE OF THE PROGRAM CODING IS QUITE MODULAR,
C   ENABLING EASE OF MODIFICATION, ADDITION AND DELETION OF
C   VARIOUS SYSTEM PARAMETERS. IT ALSO INCORPORATES A FRIENDLY
C   ENVIRONMENT WITH THE USER, EQUIPPED WITH MENUS AND EXAMPLES,
C   THEREFORE REQUIRING LITTLE KNOWLEDGE OF THE PROGRAM
C   TO RUN IT. THE USER CAN GAIN VALUABLE UNDERSTANDING OF
C   SYSTEM VARIABLES BY GENERATING A GRAPH VERSUS TIME, OR A
C   TABLE LISTING. ALSO, SYSTEM PARAMETERS CAN INTERRACTIVELY BE
C   MODIFIED BY THE USER AND THEN RERUN THE INTEGRATION.
C   THE NEW RESULTS WILL ONCE AGAIN BE AVAILABLE FOR POST-
C   PROCESSING.
C
C   SUBROUTINE DESCRIPTIONS ::
C      DEFALT :: SETS DEFAULT VARIABLE CONSTANTS AS WELL AS
C                DEFAULTED COEFFICIENTS FOR PARAMETER EQUATIONS.
C      GRAPH  :: GENERATES A X-Y GRAPH OF ANY TIME VARIANT
C                PARAMETER VERSUS TIME, AS SPECIFIED BY THE
C                USER
C      COUNTC :: COUNTS THE NUMBER OF CHARACTERS IN A PARTICULAR
C                CHARACTER STRING. CALLED BY GRAPH.
C      FILE   :: ASSIGNS A FORTRAN UNIT NUMBER TO A FILE NAME
C                WHICH IS SPECIFIED BY USER. CALLED BY PRINTA.
C      MODIFY :: ALLOWS USER TO MODIFY SYSTEM CONSTANTS, OR
C                SYSTEM PARAMETER EQUATION COEFFICIENTS BY
C                ENTERING DISCRETE DATA POINTS. CALLED BY MAIN.
C      PRINTA :: ALLOWS USER TO HAVE A TABLE OF THE TIME
C                DEPENDENT VARIABLES EITHER PRINTED AT THE
C                TERMINAL OR ONTO A SPECIFIED FILE.
C
C
C*****************************************************************
      INTEGER CHR,YY,QQ,SS,MM,RR,GG,PP,ICODE(14)
      REAL Q(12),M2,M4,I1,I2,I3,I4,IE,N3
      REAL K2,K3,KH,KD,KE
C
      COMMON/STATE/CHR
      COMMON/VEL/VRNEW,VROLD
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
      DATA JQ / 0 /
      DATA YY,QQ,GG,MM,PP,RR,SS / 'Y','?','G','M','P','R','S' /
C
C
1002 FORMAT (//,' NEXT STEP? (ENTER "?" FOR MENU)')
1004 FORMAT (//,' AVAILABLE OPTIONS ARE ::',//,'    G   GRAPH ',
            'VARIABLES VS TIME' / '    M   MODIFY CONSTANTS',/,
```

```
     &        '   P   PRINT VARIABLE TABLE',/,'   R   RUN INTEGRATION',/,
     &        '   S   STOP')
 1006 FORMAT (//,' ENTER DESIRED NUMBER OF TIME STEPS AND INCREMENT')
 1010 FORMAT (//,' ** INVALID OPTION -- TRY AGAIN **')
 1020 FORMAT (//,' ** OUTPUT OF QS FOR EACH TIME STEP CAN BE',
     &        ' FOUND ON ',/,30A1,'**',///)
 1022 FORMAT (//,' *** POSTPROCESSING OPTIONS ARE ',
     &        'NOT AVAILABLE UNTIL ',/,'    INTEGRATION',
     &        ' TECHNIQUE IS RUN (ENTER OPTION R) *** ')
 2000 FORMAT (A1)
 2002 FORMAT (I5,F7.4)
 2004 FORMAT (1X,I4)
 20   FORMAT(2F9.4)
 60   FORMAT(I5)
 C
 C
 C
 C
      IFL = 0
      CALL DEFALT
 C
 C
 C---INITIALIZE NUMBER OF INDEP VARIABLES AND TIME---
 C
      N     = 10
 C
 C---INITIALIZE INDEPENDENT VARABLES---
 C              .
      Q(1)  = -10.5
      Q(2)  = 0.0
      Q(3)  = 0.0
      Q(4)  = 0.0
      Q(5)  = 0.0
      Q(6)  = 0.0
      Q(7)  = 0.0
      Q(8)  = 52.36
      Q(9)  = 0.0
      Q(10) = 0.0
      Q(11)=Q(7)-Q(9)
      Q(12)=Q(8)-Q(10)
 C
 C
 C---INITIALIZE COUNTER
      ICOUNT = 0
 C*****************************************************************
 C
 C
 C---PROCESS NEXT OPTION---
 C
  105 WRITE (06,1002)
  110 READ (05,2000) IANS
 C
 C---LIST OPTIONS---
 C
      IF (IANS .NE. QQ) GO TO 115
         WRITE (06,1004)
         GO TO 105
 C
 C---STOP---
 C
```

```
 115 IF (IANS .NE. SS) GO TO 120
        GO TO 500
C
C---GRAPH---
C
 120 IF (IANS .NE. GG) GO TO 130
        IF (IFL .NE. 0) GO TO 122
           WRITE (06,1022)
           GO TO 105
 122    CALL GRAPH
        GO TO 105
C
C---MODIFY VARIABLE VALUES---
C
 130 IF (IANS .NE. MM) GO TO 140
        IF (IFL .NE. 0) GO TO 133
           WRITE (06,1022)
           GO TO 105
 133    CALL MODIFY
C
C---PRINT TABLE---
C
 140 IF (IANS .NE. PP) GO TO 150
        IF (IFL .NE. 0) GO TO 144
           WRITE (06,1022)
           GO TO 105
 144    CALL PRINTA (Q,N)
        GO TO 105
C
C---RUN INTEGRATION TECHNIQUE---
C
 150 IF (IANS .NE. RR) GO TO 155
     IFL = 1
     REWIND 8
C
C---WRITE NUMBER OF TIME STEPS TO PLOT FILE---
C
     WRITE (06,1006)
     READ (05,2002) NSTEP,TINC
     WRITE (06,2002) NSTEP,TINC
     WRITE (08,2004) NSTEP
     TFINAL=TINC*NSTEP
     WRITE (07,2019) TFINAL
2019 FORMAT (E10.4)
C
C---FOR EACH TIME STEP---
C
      WRITE (06,23)
23    FORMAT (' ',' PRINT OUTPUT AT EVERY HOW MANY STEPS ?')
      READ (05,22) ISTEP
22    FORMAT(I2)
C
C      STORE THE NUMBER OF PRINTED TIME STEP IN UNIT 07.
      NPSTP=NSTEP/ISTEP
      WRITE (07,2004) NPSTP
C
      NS=0
      TAU=0.0
C
79    DO 77 J=1,ISTEP
```

```
          NS=NS+1
          CALL KUTTA (Q,QDOT,TINC)
          TAU=TAU+TINC
77     CONTINUE
C---WRITE TIME, AND Q'S TO PLOT FILE---
C
       WRITE (08,53) CHR,TAU,(Q(I), I=1,10)
53     FORMAT (2X,'STATE=',I4,10X,'TIME=',E10.4,/,
     $ 2X'Q(1)=',E10.4,5X,'Q(2)=',E10.4,5X,'Q(3)=',E10.4,/,2X,'Q(4)=',
     $ E10.4,5X,'Q(5)=',E10.4,5X,'Q(6)=',E10.4,/,2X,'Q(7)=',E10.4,5X,
     $ 'Q(8)=',E10.4,5X,'Q(9)=',E10.4,/,2X,'Q(10)=',E10.4,/)
C
       Q(11)=Q(7)-Q(9)
       Q(12)=Q(8)-Q(10)
C       STORE THESE VALUES IN FILE PLOT.DATA FOR PLOT USAGE.
       WRITE (07,71) TAU,(Q(I),I=1,12)
71     FORMAT (13E15.7)
C
C
       IF (NS.GE.NSTEP) GO TO 105
       GO TO 79
C
C---INVALID OPTION---
C
 155 WRITE (06,1010)
       GO TO 110
C
C
 500 WRITE (06,1020)
C
       STOP
       END
C
C
       SUBROUTINE DEFALT
C=============================================================
C
C
C=============================================================
       REAL M2,M4,I1,I2,I3,I4,IE,N3
       REAL K2,K3,KH,KD,KE
C
       COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
       COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
C---SET VARIABLE CONSTANTS---
C
       FD=0.0
       RD=300.0
       A2=.281
       A1=1.0
       M2=10.0
       M4=1.4
       I1=169.48
       I2=0.5
       I3=5.99
       I4=30.0
       IE=200.0
```

```
      THK5=3.8
      THK4=0.6
      THK3=3.8
      D5=1.0
      RH=25.8
      RS=13.49
      UH=0.0
      N3=100.0
      UF=0.3
      TR=1.000
      RE=304.8
      DT=1.8
      RC=93.75
C
C
      RETURN
      END
C
C
      SUBROUTINE KUTTA (Q,QDOT,TINC)
C=============================================================
C
C=============================================================
      INTEGER N,CHR
      REAL Q(20),QDOT(20),M2,M4,I1,I2,I3,I4,IE,N3,OMEGA
      REAL K2,K3,KH,KD,KE,NN1
      REAL Q1(10),Q2(10),Q3(10),DQ1(10),DQ2(10),DQ3(10),DQ4(10),DQ(10)
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON/STATE/CHR
      COMMON/VEL/VRNEW,VROLD
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
C             .
      OMEGA=Q(8)-Q(6)
C
      CALL TORQIN (OMEGA)
      CALL FORCIN (T)
C
C---STATE 1 TEST---
C
       IF(Q(1).GE.( DT/A2)) GOTO 10
       CALL STATE1 (Q,QDOT)
       CHR=1
C
       DO 700 I=1,10
         DQ1(I)=TINC*QDOT(I)
         Q1(I)=Q(I)+DQ1(I)/2.0
700    CONTINUE
C
       CALL STATE1 (Q1,QDOT)
C
       DO 710 I=1,10
         DQ2(I)=TINC*QDOT(I)
         Q2(I)=Q(I)+DQ2(I)/2.0
710    CONTINUE
C
       CALL STATE1 (Q2,QDOT)
```

```
C
      DO 720 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
720   CONTINUE
C
     CALL STATE1 (Q3,QDOT)
C
      DO 730 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
730   CONTINUE
      RETURN
C
C---STATE 2 TEST---
C
10    IF(Q(3) .GE. .999) GO TO 20
      CALL STATE2 (Q,QDOT)
      CHR=2
C
      DO 800 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
800   CONTINUE
C
      CALL STATE2 (Q1,QDOT)
C
      DO 805 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
805   CONTINUE
C
      CALL STATE2 (Q2,QDOT)
C
      DO 810 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
810   CONTINUE
C
     CALL STATE2 (Q3,QDOT)
C
      DO 815 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
815   CONTINUE
      RETURN
C
C---STATE 3 TEST---
C
20    Q(4)=0.0
      IF (Q(1).GE.-.001) GO TO 60
      ICOUNT=ICOUNT+1
      IF (ICOUNT.EQ.1) VRNEW=Q(8)-Q(10)
      IF (ICOUNT.EQ.1) JJJ=0
      IF (ICOUNT.EQ.1) GO TO 500
C
C      NOATION:          (1) JJJ=0, SLIPPING.
C                        (3) JJJ=2, THE IMMEDIATE HISTORY IS ONE
```

```
C                              OF NON-SLIP.  THE INTERFACE SURFACES
C                              ARE BEGINNING TO SLIP.
C                         (2) JJJ=1, STICKING.
C
      IF (JJJ.EQ.1) GO TO 200
      IF ((VRNEW*VROLD) .LT. 0.0) GO TO 200
C
C
500   CALL STATE3 (Q,QDOT)
      IF (JJJ.EQ.0) CHR=3
      IF (JJJ.EQ.2) CHR=35
C
      DO 820 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
820   CONTINUE
C
      CALL STATE3 (Q1,QDOT)
C
      DO 825 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
825   CONTINUE
C
      CALL STATE3 (Q2,QDOT)
C
      DO 830 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
830   CONTINUE
C
      CALL STATE3 (Q3,QDOT)
C
      DO 835 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
835   CONTINUE
C
      VROLD=VRNEW
      VRNEW=Q(8)-Q(10)
C
      RETURN
C
C
 200  TT1 = TE
      NN1 = K3*((DT+A2*Q(1))-Q(3))
      AA = I1*(TT1)/(I1+I2+I3+I4)-TT1
      FT = NN1*UF*RC*2.0
C
      IF(ABS(AA) .GT. FT) JJJ=2
      IF(ABS(AA) .GT. FT) GO TO 500
      IF(ABS(AA) .LT. FT) JJJ=1
      IF(JJJ. NE. 1) RETURN
C
      CALL STAT3C(Q,QDOT)
      JJJ=1
      VRNEW = 0.0
      CHR = 33
C
```

```
      DO 840 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
840   CONTINUE
C
      CALL STAT3C (Q1,QDOT)
C
      DO 845 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
845   CONTINUE
C
      CALL STAT3C (Q2,QDOT)
C
      DO 850 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
850   CONTINUE
C
      CALL STAT3C (Q3,QDOT)
C
      DO 855 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
855   CONTINUE
      RETURN
C
C
C-----BEGIN STATE4 FRICTION LOGIC----
 60   Q(2)=0.0
      WRITE(8,9999) Q(8),Q(10)
9999  FORMAT(2X,'Q(8)=',E10.4,5X,'Q(10)=',E10.4)
      WRITE(8,9998) VRNEW,VROLD
9998  FORMAT(2X,'VRNEW=',E10.4,4X,'VROLD=',E10.4)
      IF (JJJ.EQ.1) GO TO 205
      IF((VRNEW*VROLD) .LT. 0.0) GO TO 205
C
C
      JJJ=0
505   CALL STATE4 (Q,QDOT)
      IF (JJJ.EQ.0) CHR =4
      IF (JJJ.EQ.2) CHR =45
C
      DO 860 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
860   CONTINUE
C
      CALL STATE4 (Q1,QDOT)
C
      DO 865 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
865   CONTINUE
C
      CALL STATE4 (Q2,QDOT)
C
      DO 870 I=1,10
        DQ3(I)=TINC*QDOT(I)
```

```
        Q3(I)=Q(I)+DQ3(I)
870   CONTINUE
C
      CALL STATE4 (Q3,QDOT)
C
      DO 875 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
875   CONTINUE
C
      VROLD=VRNEW
      VRNEW=Q(8)-Q(10)
C
      RETURN
C
C
205   TT1 = TE
      NN1 = K3*((DT+A2*Q(1))-Q(3))
      AA = I1*(TT1)/(I1+I2+I3+I4)-TT1
C
C
      FT = NN1*UF*RC*2.0
C
      IF(ABS(AA) .GT. FT) JJJ=2
      IF(ABS(AA) .GT. FT) GO TO 505
      IF(ABS(AA) .LT. FT) JJJ=1
      IF(JJJ. NE. 1) RETURN
C
C
      CALL STAT4C (Q,QDOT)
      JJJ=1
      VRNEW = 0.0
      CHR=43
C
      DO 880 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
880   CONTINUE
C
      CALL STAT4C (Q1,QDOT)
C
      DO 885 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
885   CONTINUE
C
      CALL STAT4C (Q2,QDOT)
C
      DO 890 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
890   CONTINUE
C
      CALL STAT4C (Q3,QDOT)
C
      DO 895 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
```

```
895    CONTINUE
       RETURN
       END
C
C
C=======================================================
      SUBROUTINE STATE1 (Q,QDOT)
C=======================================================
C      Pressure plate not in contact with clutch;
C      clutch pedal starting return.
C=======================================================
      REAL Q(10),QDOT(10),K2,K3,KH,KD,KE,M2,M4,I1,I2,I3,I4,IE,N3
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
C      BELSPR and ENGINE supply non-linear parameters K2 and
C      KE,CE respectively.
C
      CALL BELSPR
      CALL ENGINE
C
C      STATE #1 SET I - ROTATIONAL EQUATIONS OF MOTION
C      Flywheel/clutch cover/pressure plate/engine crank assembly
C
      QDOT(7)=Q(8)
      QDOT(8)=TE/I1
C
C      STATE #1 SET II - ROTATIONAL EQUATIONS OF MOTION
C      Engine block
C
      QDOT(5)=Q(6)
      QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
C
C      STATE #1 SET I - TRANSLATIONAL EQUATIONS OF MOTION
C      Pressure Plate
C
      THKT=9.34
      QDOT(1)=Q(2)
      QDOT(2)=K2/(M2*A2)*(THKT-A2*Q(1))-
     $FINPUT/(M2*A2)-(C2/M2)*Q(2)
C
      QDOT(3)  = 0.
      QDOT(4)  = 0.
      QDOT(9)  = 0.
      QDOT(10) = 0.
      RETURN
      END
C
C
C=======================================================
      SUBROUTINE STATE2 (Q,QDOT)
C=======================================================
C      Pressure plate in contact with clutch; flywheel
C      not in contact with clutch; slipping at clutch/pressure plate
C      interface assumed
C=======================================================
      REAL Q(10),QDOT(10),K2,K3,KH,KD,KE,M2,M4,I1,I2,
```

```
     &      I3,I4,IE,N3,N1,K4
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
C      BELSPR, MARCEL, and ENGINE supply non-linear
C      parameters (K2,C2), (K3,C3), and (KE,CE)
C      respectively.
C
      CALL BELSPR
      CALL MARCEL
      CALL ENGINE
C
C      N1, and T2 represent clutch normal
C      force, and clutch output torque respectively.
C      These quantities are used frequently in
C      subsequent equations and logic throughout the program.
C
      N1=K3*((DT+A2*Q(1))-Q(3))+C3*((A2*Q(2))-Q(4))
      T1=N1*UF*RC
C
C      STATE #2 SET I - ROTATIONAL EQUATIONS OF MOTION
C      Flywheel assembly
C
      QDOT(7)=Q(8)
      QDOT(8)=(TE-T1)/I1
C
C      STATE #2 SET II ROTATIONAL EQUATIONS OF MOTION
C      Outer clutch rotation
C
      QDOT(9)=Q(10)
      QDOT(10)=(T1)/(I2+I3+I4)
C
C      STATE #2 SET V - ROTATIONAL EQUATIONS OF MOTION
C      Engine block rotation.
C
      QDOT(5)=Q(6)
      QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
C
C      STATE #2 SET I - TRANSLATIONAL EQUATIONS OF MOTION
C      Pressure plate travel.
C
      THKT=9.34
C
      QDOT(1)=Q(2)
      QDOT(2)=K2/(M2*A2)*(THKT-A2*Q(1))-
     $FINPUT/(M2*A2)-(C2+C3)/M2*Q(2)+C3/(M2*A2)*Q(4)-
     $K3/(M2*A2)*(DT+A2*Q(1)-Q(3))
C
C      STATE #2 SET II - TRANSLATIONAL EQUATIONS OF MOTION
C      Clutch displacement
C
      QDOT(3)=Q(4)
      QDOT(4)=C3/M4*(Q(2)*A2-Q(4))+K3/M4*(DT+A2*Q(1)-Q(3))
      RETURN
      END
C
C=======================================================
```

```
      SUBROUTINE STATE3 (Q,QDOT)
C=======================================================
C     JJJ=0
C     Pressure plate and flywheel both in contact with clutch.
C     Interface surfaces are slipping.  Clutch hub springs
C     are not fully compressed.
C
C     JJJ=2
C     The immediate history is one of non-slip.  Therefore,
C     the clutch is contacting the flywheel and the pressure
C     plate.  The interface surfaces are BEGINNING to slip.
C     The direction of slip is determined by the sign of
C     the quantity "AA" below.
C=======================================================
      REAL Q(10),QDOT(10),K2,K3,K4,KH,KD,KE,N1
     $,M2,M4,I1,I2,I3,I4,IE,N3,N1MN2
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C     BELSPR, MARCEL, and ENGINE supply non-linear
C     parameters (K2,C2), (K3,C3), and (KE,CE)
C     respectively.
C
      CALL BELSPR
      CALL MARCEL
      CALL ENGINE
C
C     N1, and T2 represent clutch normal
C     force, and clutch output torque respectively.
C     These quantities are used frequently in
C     subsequent equations and logic throughout the program.
C
      N1=(K3*(DT+A2*Q(1)-Q(3))+C3*((A2*Q(2))-Q(4)))
      T2=N1*UF*RC*2.0
C
C     STATE #3 SET I - ROTATIONAL EQUATIONS OF MOTION
C     Equations check for direction of relative rotation  (i.e. friction
C     torque direction) between flywheel assembly and clutch during
C     evaluation of flywheel assembly output values.
C
      QDOT(7)=Q(8)
      IF (JJJ.EQ.0) QDOT(8) = (TE-SIGN(1.0,Q(8)-Q(10))*T2)/I1
      IF (JJJ.EQ.2) QDOT(8) = (TE+SIGN(1.0,AA)*T2)/I1
C
C     STATE #3 SET II - ROTATIONAL EQUATIONS OF MOTION
C     Equations check for direction of relative rotation
C     between flywheel assembly and clutch during
C     evaluation of clutch output characteristics.
C
      QDOT(9)=Q(10)
      IF (JJJ.EQ.0) QDOT(10)=(SIGN(1.0,Q(8)-Q(10))*T2)/(I2+I3+I4)
      IF (JJJ.EQ.2) QDOT(10)=-(SIGN(1.0,AA)*T2)/(I2+I3+I4)
C
C     STATE #3 SET V - ROTATIONAL EQUATIONS OF MOTION
C     Engine block rotation.
C
      QDOT(5)=Q(6)
      QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
```

```
C
C      STATE #3 SET I - TRANSLATIONAL EQUATIONS OF MOTION
C      Pressure plate
C
       THKT=9.34
C
       QDOT(1)=Q(2)
       QDOT(2)=K2/(M2*A2)*(THKT-A2*Q(1))-
      $FINPUT/(M2*A2)-C2/M2*Q(2)-C3/(M2*A2)*(Q(2)*A2-Q(4))-
      $K3/(M2*A2)*(DT+A2*Q(1)-Q(3))
C
C      Constraints for variables with fixed values;
C      specifically, the clutch is in contact with
C      the flywheel.
C
       QDOT(3) = Q(4)
       QDOT(4)=0.0
       RETURN
       END
C
C=======================================================
       SUBROUTINE STAT3C (Q,QDOT)
C=======================================================
C      Pressure plate and flywheel both in contact with clutch.
C      Interface surfaces are NOT slipping.  Clutch hub springs
C      are NOT fully compressed; i.e. the stop pins are NOT
C      contacted.
C=======================================================
       REAL Q(10),QDOT(10),K2,K3,K4,KH,KD,KE,N3,I1,I2,I3,I4,IE,
      &       M2,M4,N1,N2
C
       COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
       COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
      $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
      $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
C      BELSPR, MARCEL, and ENGINE supply non-linear
C      parameters (K2,C2), (K3,C3), and (KE,CE)
C      respectively.
C
       CALL BELSPR
       CALL MARCEL
       CALL ENGINE
C
C---STATE #3C SET I - ROTATIONAL EQUATIONS OF MOTION
C      No slip is occuring at the clutch surface, therefore, the
C      flywheel assembly and clutch are treated as one rigid body.
C      For this reason, the respective rotational values are equated below.
C
       QDOT(7)=Q(8)
       QDOT(8)=(TE)/(I1+I2+I3+I4)
       QDOT(10)=(TE)/(I1+I2+I3+I4)
       QDOT(9)=Q(10)
C
C      STATE #3C SET IV - ROTATIONAL EQUATIONS OF MOTION
C      Engine block rotation.
C
       QDOT(5)=Q(6)
       QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
```

```
C      STATE #3C SET I - TRANSLATIONAL EQUATIONS OF MOTION
C      Pressure plate
C
       THKT=9.34
C
       QDOT(1)=Q(2)
       QDOT(2)=(K2/(M2*A2))*(THKT-A2*Q(1))-
      $FINPUT/(M2*A2)-C2/M2*Q(2)-C3/(M2*A2)*(Q(2)*A2-Q(4))-
      $K3/(M2*A2)*(DT+A2*Q(1)-Q(3))
C
C---STATE #3C SET II - TRANSLATIONAL EQUATIONS---
C
C      Constraints for variables with fixed values;
C      specifically, the clutch is in contact with
C      the flywheel.
C
       QDOT(3)=Q(4)
       QDOT(4)=0.0
       RETURN
       END
C
C=================================================================
       SUBROUTINE STATE4 (Q,QDOT)
C=================================================================
C      JJJ=0, slipping.
C      JJJ=2, the immediate history is one of non-slip.
C           The interface surfaces are BEGINNING to slip.
C=================================================================
       REAL Q(10),QDOT(10),K2,K3,K4,KH,KD,KE,M2,M4,I1,I2,
      $      I3,I4,IE,N1,N3
C
       COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
       COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
      $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
      $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
       CALL ENGINE
       CALL BELSPR
       CALL MARCEL
       N1=(K3*(DT+A2*Q(1)-Q(3))+C3*((A2*Q(2))-Q(4)))
       T2=N1*UF*RC*2.0
C
C---STATE #4 SET I - ROTATIONAL EQUATIONS---
C
       QDOT(7)=Q(8)
       IF (JJJ.EQ.0) QDOT(8)=(TE-SIGN(1.0,Q(8)-Q(10))*T2)/I1
       IF (JJJ.EQ.2) QDOT(8)=(TE+SIGN(1.0,AA)*T2)/I1
C
C
C---STATE #4 SET II & III - ROTATIONAL EQUATIONS---
C
       QDOT(9)=Q(10)
       IF (JJJ.EQ.0) QDOT(10)=(SIGN(1.0,Q(8)-Q(10))*T2)/(I2+I3+I4)
       IF (JJJ.EQ.2) QDOT(10)=-(SIGN(1.0,AA)*T2)/(I2+I3+I4)
C
C---STATE #4 SET V - ROTATIONAL EQUATIONS---
C
       QDOT(5)=Q(6)
       QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
C
```

```
C---STATE #4 SET VI - TRANSLATIONAL EQUATIONS - CONSTRAINTS---
C
      QDOT(1) = Q(2)
      QDOT(2) = 0.0
      QDOT(3) = Q(4)
      QDOT(4) = 0.0
C
      RETURN
      END
C
C=======================================================
      SUBROUTINE STAT4C (Q,QDOT)
C=======================================================
C
C
C=======================================================
      REAL Q(10),QDOT(10),KH,KD,KE,K2,K3,
     $     I1,I2,I3,I4,IE,N3,M2,M4
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
      CALL ENGINE
C
C
      QDOT(7)=Q(8)
      QDOT(8)=(TE)/(I1+I2+I3+I4)
C      CONSTRAINTS
      QDOT(9)=Q(10)
      QDOT(10)=QDOT(8)
      QDOT(1)=Q(2)
      QDOT(2)=0.0
      QDOT(3)=Q(4)
      QDOT(4)=0.0
      QDOT(5)=Q(6)
      QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
      RETURN
      END
C
C
C
      SUBROUTINE TORQIN (OMEGA)
C=======================================================
C
C=======================================================
      REAL TE,K2,K3,KH,KD,KE
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      TE=10000.0
      RETURN
      END
C
C
      SUBROUTINE FORCIN (T)
C=======================================================
C
C=======================================================
      REAL T,FINPUT,K2,K3,KH,KD,KE
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
```

```
C
      FINPUT=0.0
      RETURN
      END
C
C
      SUBROUTINE BELSPR
C=======================================================
C
C=======================================================
      REAL K2,C2,K3,KH,KD,KE
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
      K2=200.0
      C2=0.0
      RETURN
      END
C
C
      SUBROUTINE MARCEL
C=======================================================
C
C=======================================================
      REAL K3,C3,K4,C4,K2,KH,KD,KE,M2,M4,I1,I2,I3,I4,IE
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
      K3=3000.0
      C3 = 0.0
      RETURN
      END
C
C
      SUBROUTINE ENGINE
C=============================================================
C
C=============================================================
      REAL KE,CE,K2,K3,KH,KD
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
      KE=2000.0
      CE=10.0
      RETURN
      END
C
C
       SUBROUTINE GRAPH
C=========================================================
C
C=========================================================
       LOGICAL*1 XT(50),YT(50),NAME(8)
       REAL X1(2500),Y1(2500),DUM(12)
       INTEGER DD,TT,QQ,ICODE(12)
C
      DATA DD,TT,QQ / 'D ','T ','? ' /
      DATA ICODE / '1 ','2 ','3 ','4 ','5 ','6 ','7 ',
     &             '8 ','9 ','10','11','12' /
C
```

```
C
50    FORMAT (13E15.7)
70    FORMAT(5X,'ENTER X-AXIS LABEL')
80    FORMAT(50A1)
90    FORMAT(5X,'ENTER Y-AXIS LABEL')
C
1000 FORMAT (/,' WHICH VARIABLE?--ENTER "?" FOR MENU')
1002 FORMAT (/,' VARIABLES ARE ::',
     &        /,' 1    Q1',/,' 2    Q2',/,' 3    Q3',
     &        /,' 4    Q4',/,' 5    Q5',/,' 6    Q6',
     &        /,' 7    Q7',/,' 7    Q7',/,' 8    Q8',
     &        /,' 9    Q9',/,' 10   Q10',/,' 11   Q7-Q9'
     &        /,' 12   Q8-Q10',/,' D   DONE')
1006 FORMAT (//' *** INVALID ENTRY--TRY AGAIN ***')
2000 FORMAT (A2)
2002 FORMAT (1X,I4)
2004 FORMAT (I2)
C
C
C
C---DEFINE VARIABLE TO BE PLOTTED VERSUS TIME---
C
 100 WRITE (06,1000)
     READ (05,2000) IANS
C
C---PRINT LIST OF VARIABLES, IF REQUESTED---
C
     IF (IANS .NE. QQ) GO TO 110
        WRITE (06,1002)
        GO TO 100
C
C---IF USER DONE, RETURN---
C
 110 IF (IANS .EQ. DD) GO TO 500
C
C---TORQUE---
C
C       IF (IANS .NE. TT) GO TO 10
C          NN = 1
C          GO TO 30
C
C---PROCESS ANY OF THE Q'S, IF CHOSEN---
C
  10 DO 20 I=1,12
        IF (IANS .EQ. ICODE(I)) GO TO 25
  20 CONTINUE
     WRITE (06,1006)
     GO TO 100
C
  25 NN = I
C
C---REWIND TEMPORARY FILE 7---
C
  30  REWIND 7
C
C---READ FROM FILE 7 AND PACK X,Y ARRAYS---
C
      READ (07,2019) TFINAL
2019  FORMAT (E10.4)
      READ (07,2002) N1
```

```
      DO 40 I=1,N1
        IF (NN .NE. 1) GO TO 35
        READ (07,50) X1(I),Y1(I),(DUM(J),J=1,11)
        Y1(I)=-Y1(I)
        GO TO 40
C
C---READ TORQUE PLOT INFO---
C   FOR THIS VERSION, TORQUE IS NO LONGER A VARIABLE.
C
C            READ (07,50) X1(I),Y1(I),(DUM(J),J=1,12),ICHR
C            GO TO 40
C
  35    IF (NN .NE. 12) GO TO 37
           READ (07,50) X1(I),(DUM(J),J=1,11),Y1(I)
           GO TO 40
C
  37    NZ = NN-1
        NF = 12-NN
        READ (07,50) X1(I),(DUM(J),J=1,NZ),Y1(I),(DUM(J),J=1,NF)
C
40    CONTINUE
C
C
C---GET AXIS LABEL INFORMATION---
C
      WRITE(6,70)
      READ(5,80) XT
      CALL COUNTC(XT,50,NXT)
      WRITE(6,90)
      READ(5,80) YT
      CALL COUNTC(YT,40,NYT)
C
C
      XF=TFINAL/5.
      XM=0.0
C
      CALL PSCALE (3.,0.5,YM,YF,Y1,N1,1)
      CALL PLTOFS (XM,XF,YM,YF,1.,1.)
      CALL PLINE (X1,Y1,N1,1,0,0,1)
      CALL PAXIS (1.,1.,XT,-NXT,5.,0.,XM,XF,.5)
      CALL PAXIS (1.,1.,YT,NYT,3.,90.,YM,YF,.5)
      CALL PLTEND
      GO TO 100
 500  CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE COUNTC (STRN,MAXC,NCHR)
C=============================================================
C
C=============================================================
      LOGICAL EQUC
      LOGICAL*1 STRN(1),BLNK
      DATA BLNK/' '/
      DO 20 I=1,MAXC
      IF(EQUC(STRN(I),BLNK)) GO TO 10
      GO TO 20
10    IF(EQUC(STRN(I-1),BLNK)) GO TO 15
```

```
      GO TO 20
15    NCHR=I-2
      GO TO 30
20    CONTINUE
      NCHR=I
      IF(EQUC(STRN(I),BLNK)) NCHR=I-1
30    RETURN
      END
C
C
      SUBROUTINE FILE (IUNIT,NAME)
C*********************************************************************
C
C  THIS SUBROUTINE ASSIGNS FORTRAN UNIT NUMBER, IUNIT, TO A FILE OF
C  WHICH IS CALLED NAME
C
C*********************************************************************
      LOGICAL*1 NAME(30)
      CALL SETLIO (IUNIT,NAME,&85)
      RETURN
  85 WRITE (06,87)
  87 FORMAT (' ILLEGAL I/O UNIT OR FILE NAME')
     STOP
      END
C
C
      SUBROUTINE MODIFY
C=============================================================
C
C=============================================================
      INTEGER ICODEA(13),ICODEB(16),QQ
      REAL K2,K3,KH,KD,KE,M2,M4,I1,I2,I3,I4,IE,N3
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
      DATA JQ / 0 /
      DATA ICODEA / 'TE','FI','K2','K3','KH','KD','KE','C2',
     &              'C3','CE','BD','FD','RD' /
      DATA ICODEB / 'M2','M4','I1','I2','I3','I4','IE','RH',
     &              'UH','US','N3','UF','TR','RE','RC','A2' /
      DATA DD / 'D ' /
      DATA QQ / '?' /
C
C
1000 FORMAT (/,' ON ONE INPUT LINE, ENTER VARIABLE CODE',
     &      ' AND MODIFY ID (C...CONSTANT ; L...LEAST SQUARES FIT).
     &     /,'TERMINATE INPUTS WITH "D <CR>". FOR',
     &      ' VARIABLE LISTING AND EXAMPLE ENTER "?" ')
1002 FORMAT (//,' VARIABLES THAT MAY BE CHANGED ARE ::',
     &        /,'   TE    -- ENGINE TORQUE',
     &        /,'   FI    -- CLUTCH PEDAL FORCE',
     &        /,'   K2    -- BELLEVILLE SPRING RATE',
     &        /,'   K3    -- CUSHION SPRING RATE',
     &        /,'   KE    -- ENGINE MOUNT STIFFNESS',
     &        /,'   KH    -- TORSIONAL CLUTCH SPRING RATE',
     &        /,'   KD    -- TORSIONAL DRIVE TRAIN STIFFNESS',
     &        /,'   C2    -- BELLEVILLE LINKAGE DAMPING COEFFICIENT',
```

```
     &       /,'    C3    -- CUSHION DAMPING COEFFICIENT',
     &       /,'    CE    -- ENGINE MOUNT DAMPING COEFFICIENT',
     &       /,'    BD    -- DRIVE TRAIN VISCOUS DAMPING COEFFICIENT',
     &       /,'    FD    -- ROLLING DRAG FORCE ON TIRE',
     &       /,'    RD    -- TIRE ROLLING RADIUS',
     &       //,' EXAMPLE INPUT SEQUENCE ::'
     &       /,'    K3,C <CR>     OR      D  <CR>',/)
1010 FORMAT (//,' ** INVALID VARIABLE CODE INPUT--',
     &        'RE-ENTER LINE **')
2000 FORMAT (A2,A2)
C
C
C
 110 WRITE (06,1000)
 120 READ (05,2000) IVAR,ICD
     IF (IVAR .NE. QQ) GO TO 200
        WRITE (06,1002)
        JQ = JQ+1
        IF (JQ .GT. 1) GO TO 120
        GO TO 110
C
C---CHECK IVAR FOR VALID INPUT---
C
 200 IC = 1
     DO 300 I=1,13
        IF (IVAR .EQ. ICODEA(I)) GO TO 600
 300 CONTINUE
C
 400 IC = 2
     DO 500 I=1,16
        IF (IVAR .EQ. ICODEB(I)) GO TO 600
 500 CONTINUE
     WRITE (06,1010)
     GO TO 120
C
C
 600 IV = I
     CALL MODCON (IC,IV)
     RETURN
     END
C
C
     SUBROUTINE PRINTA (Q,N)
C=============================================================
C
C=============================================================
     LOGICAL*1 NAME(30)
     REAL Q(N),M2,M4,N3,I1,I2,I3,I4,IE
     INTEGER TT,FF
C
     COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
    $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
    $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
     DATA TT,FF / 'T','F' /
1000 FORMAT (/,' WOULD YOU LIKE YOUR TABLE TO BE PRINTED',
    &       ' AT THIS TERMINAL',/,' OR SENT TO AN OUTPUT FILE',
    &       ' (T OR F) ?')
1002 FORMAT (/,' PLEASE ENTER FILE NAME TO EMPTY DATA TABLE')
1004 FORMAT (//,'  Q1',6X,'Q2',6X,'Q3',6X,'Q4',6X,'Q5',6X,'Q6',
```

```
     &            6X,'Q7',6X,'Q8',6X,'Q9',5X,'Q10',/,' Q11',5X,'Q12',
     &            5X,'Q13',5X,'Q14',5X,'QD1',5X,'QD2',5X,'QD3',
     &            5X,'QD4',5X,'QD5',5X,'QD6',/,'  QD7',5X,'QD8',5X,
     &           'QD9',4X,'QD10',4X,'QD11',4X,'QD12',4X,'QD13',
     &           4X,'QD14',4X,'STATE')
2000 FORMAT (A1)
2002 FORMAT (30A1)
2004 FORMAT (2X,2E10.4,/,2X,6E10.4,/,2X,6E10.4,/,
    $2X,2E10.4,/,2X,I8)
2006 FORMAT (1X,I4)
2008 FORMAT (2X,7F10.4,/,2X,4F10.4,/2X,5F10.4,/,2X,I8)
C
      WRITE (06,1000)
      READ (05,2000) IANS
C
      IF (IANS .EQ. TT) GO TO 200
         IFILE = 4
         WRITE (06,1002)
         READ (05,2002) NAME
         CALL FILE (IFILE,NAME)
         GO TO 300
C
 200 IFILE = 6
C
C
C---WRITE OUT TITLES FOR Q AND QDOT OUTPUT---
C---ON DATA CHART FILE---
C
 300  WRITE (IFILE,1004)
C
C---REWIND FILE 8 FOR READING---
C
      REWIND 8
C
C---READ NUMBER OF STEPS---
C
      READ (08,2006) NSTEP
C
C---FOR EACH TIME STEP---
C
      DO 500 I=1,NSTEP
         READ (08,2004) T,TOR,(Q(J),J=1,10),ICHR
         WRITE (IFILE,2008) T,TOR,(Q(J),J=1,10),ICHR
 500 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE MODCON (IC,IV)
C=================================================
C
C=================================================
      REAL RA(13),RB1(7),RD(7),RB2(8),DUM(3)
C
      COMMON /VARS/ RA,IDUM
      COMMON /PAR/ RB1,RD,RB2,DUM
C
1000 FORMAT (/,' ENTER NEW CONSTANT VALUE')
1002 FORMAT (/,' *** INVALID IC IN MODCON *****')
```

```
2000 FORMAT (F9.4)
C
 300 IF (IC .NE. 1) GO TO 500
        WRITE (06,1000)
        READ (05,2000) RA(IV)
        WRITE (06,111) IV,RA(IV)
 111   FORMAT (' --VARS ARRAY---',I6,F10.4)
        GO TO 900
C
 500 IF (IC .NE. 2) GO TO 800
        WRITE (06,1000)
        IF (IV .GT. 7) GO TO 700
           READ (05,2000) RB1(IV)
           WRITE (06,222) IV,RB1(IV)
 222       FORMAT (' --PAR ARRAY---',I6,F10.4)
           GO TO 900
C
 700 IN = IV-7
     READ (05,2000) RB2(IN)
     WRITE (06,222) IN,RB2(IN)
C
 800 WRITE (06,1002)
C
 900 CONTINUE
     RETURN
     END
```

```
C*****************************************************************
C     MARCH 21, 1986
C     ========FOURTH ORDER RUNGE-KUTTA SCHEME===========
C
C           ======== STEP TORQUE =========
C
C  THIS PROGRAM GENERATES  A COMPUTATIONAL MODEL OF THE
C  AUTOMOTIVE CLUTCH SYSTEM. IT IS MEANT TO REDUCE PRESENT
C  "TRIAL AND ERROR" METHODS USED IN CLUTCH DESIGN WHICH
C  HAVE NOT ONLY PROVED TO BE EXPENSIVE AND TIME CONSUMING,
C  BUT DO NOT ALLOW THE ENGINEER TO REALIZE THE POTENTIAL FOR
C  THE IMPROVED AND OPTIMAL DESIGNS WHICH COULD BE ACHIEVED.
C
C  THE STRUCTURE OF THE PROGRAM CODING IS QUITE MODULAR,
C  ENABLING EASE OF MODIFICATION, ADDITION AND DELETION OF
C  VARIOUS SYSTEM PARAMETERS. IT ALSO INCORPORATES A FRIENDLY
C  ENVIRONMENT WITH THE USER, EQUIPPED WITH MENUS AND EXAMPLES,
C  THEREFORE REQUIRING LITTLE KNOWLEDGE OF THE PROGRAM
C  TO RUN IT. THE USER CAN GAIN VALUABLE UNDERSTANDING OF
C  SYSTEM VARIABLES BY GENERATING A GRAPH VERSUS TIME, OR A
C  TABLE LISTING. ALSO, SYSTEM PARAMETERS CAN INTERRACTIVELY BE
C  MODIFIED BY THE USER AND THEN RERUN THE INTEGRATION.
C  THE NEW RESULTS WILL ONCE AGAIN BE AVAILABLE FOR POST-
C  PROCESSING.
C
C  SUBROUTINE DESCRIPTIONS ::
C     DEFALT :: SETS DEFAULT VARIABLE CONSTANTS AS WELL AS
C               DEFAULTED COEFFICIENTS FOR PARAMETER EQUATIONS.
C     GRAPH  :: GENERATES A X-Y GRAPH OF ANY TIME VARIANT
C               PARAMETER VERSUS TIME, AS SPECIFIED BY THE
C               USER
C     COUNTC :: COUNTS THE NUMBER OF CHARACTERS IN A PARTICULAR
C               CHARACTER STRING. CALLED BY GRAPH.
C     FILE   :: ASSIGNS A FORTRAN UNIT NUMBER TO A FILE NAME
C               WHICH IS SPECIFIED BY USER. CALLED BY PRINTA.
C     MODIFY :: ALLOWS USER TO MODIFY SYSTEM CONSTANTS, OR
C               SYSTEM PARAMETER EQUATION COEFFICIENTS BY
C               ENTERING DISCRETE DATA POINTS. CALLED BY MAIN.
C     PRINTA :: ALLOWS USER TO HAVE A TABLE OF THE TIME
C               DEPENDENT VARIABLES EITHER PRINTED AT THE
C               TERMINAL OR ONTO A SPECIFIED FILE.
C
C
C*****************************************************************
      INTEGER CHR,YY,QQ,SS,MM,RR,GG,PP,ICODE(14)
      REAL Q(12),M2,M4,I1,I2,I3,I4,IE,N3
      REAL K2,K3,KH,KD,KE
C
      COMMON/STATE/CHR,NS,NTE,ATE
      COMMON/VEL/VRNEW,VROLD
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
      DATA JQ / 0 /
      DATA YY,QQ,GG,MM,PP,RR,SS / 'Y','?','G','M','P','R','S' /
C
C
1002 FORMAT (//,' NEXT STEP? (ENTER "?" FOR MENU)')
```

```
1004 FORMAT (//,' AVAILABLE OPTIONS ARE ::',//,'    G   GRAPH ',
    &        'VARIABLES VS TIME',/,'    M   MODIFY CONSTANTS',/,
    &        '    P   PRINT VARIABLE TABLE',/,'    R   RUN INTEGRATION',/,
    &        '    S   STOP')
1006 FORMAT (//,' ENTER DESIRED NUMBER OF TIME STEPS AND INCREMENT')
1010 FORMAT (//,' ** INVALID OPTION -- TRY AGAIN **')
1020 FORMAT (//,' ** OUTPUT OF QS FOR EACH TIME STEP CAN BE',
    &         ' FOUND ON ',/,30A1,'**',///)
1022 FORMAT (//,' *** POSTPROCESSING OPTIONS ARE ',
    &        'NOT AVAILABLE UNTIL ',/,'    INTEGRATION',
    &        ' TECHNIQUE IS RUN (ENTER OPTION R) *** ')
2000 FORMAT (A1)
2002 FORMAT (I5,F7.4)
2004 FORMAT (1X,I4)
20   FORMAT(2F9.4)
60   FORMAT(I5)
C
C
C
C
     IFL = 0
     CALL DEFALT
C
C
C---INITIALIZE NUMBER OF INDEP VARIABLES AND TIME---
C
     N     = 10
C
C---INITIALIZE INDEPENDENT VARABLES---
C
C
C      STEADY STATE VALUES FROM PREVIOUS RUN.
C
     Q(1)  = 0.0
     Q(2)  = 0.0
     Q(3)  = 1.0
     Q(4)  = 0.0
     Q(5)  = 0.01711
     Q(6)  = -0.1987
     Q(7)  = 10.86
     Q(8)  = 52.76
     Q(9)  = 3.626
     Q(10) = 52.77
     Q(11)=Q(7)-Q(9)
     Q(12)=Q(8)-Q(10)
C
C
C---INITIALIZE COUNTER
     ICOUNT = 0
C*************************************************************
C
C
C---PROCESS NEXT OPTION---
C
 105 WRITE (06,1002)
 110 READ (05,2000) IANS
C
C---LIST OPTIONS---
C
     IF (IANS .NE. QQ) GO TO 115
```

```
        WRITE (06,1004)
        GO TO 105
C
C---STOP---
C
 115 IF (IANS .NE. SS) GO TO 120
        GO TO 500
C
C---GRAPH---
C
 120 IF (IANS .NE. GG) GO TO 130
        IF (IFL .NE. 0) GO TO 122
           WRITE (06,1022)
           GO TO 105
 122    CALL GRAPH
        GO TO 105
C
C---MODIFY VARIABLE VALUES---
C
 130 IF (IANS .NE. MM) GO TO 140
        IF (IFL .NE. 0) GO TO 133
           WRITE (06,1022)
           GO TO 105
 133    CALL MODIFY
C
C---PRINT TABLE---
C
 140 IF (IANS .NE. PP) GO TO 150
        IF (IFL .NE. 0) GO TO 144
           WRITE (06,1022)
           GO TO 105
 144    CALL PRINTA (Q,N)
        GO TO 105
C
C---RUN INTEGRATION TECHNIQUE---
C
 150 IF (IANS .NE. RR) GO TO 155
     IFL = 1
     REWIND 8
C
C---WRITE NUMBER OF TIME STEPS TO PLOT FILE---
C
     WRITE (06,1006)
     READ (05,2002) NSTEP,TINC
     WRITE (06,2002) NSTEP,TINC
     WRITE (08,2004) NSTEP
     TFINAL=TINC*NSTEP
     WRITE (07,19) TFINAL
19   FORMAT(E10.4)
C
C---FOR EACH TIME STEP---
C
      WRITE (06,23)
23    FORMAT (' ',' PRINT OUTPUT AT EVERY HOW MANY STEPS ?')
      READ (05,22) ISTEP
22    FORMAT(I2)
C
C---INPUT FORCE-------
      WRITE (06,700)
700   FORMAT (' ',' ENTER STEP TORQUE:')
```

```
      READ (05,701) ATE
701   FORMAT (E10.4)
      WRITE (06,703)
703   FORMAT (' ','STEP TORQUE AT WHAT TIME STEP ?')
      READ (05,29) NTE
29    FORMAT (I3)
C
C       STORE THE NUMBER OF PRINTED TIME STEP IN UNIT 07.
      NPSTP=NSTEP/ISTEP
      WRITE (07,2004) NPSTP
C
      NS=0
      TAU=0.0
C
79    DO 77 J=1,ISTEP
          NS=NS+1
          CALL KUTTA (Q,QDOT,TINC)
          TAU=TAU+TINC
77    CONTINUE
C---WRITE TIME, AND Q'S TO PLOT FILE--
C
      WRITE (08,53) CHR,TAU,(Q(I), I=1,10)
53    FORMAT (2X,'STATE=',I4,10X,'TIME=',E10.4,/,
     $ 2X'Q(1)=',E10.4,5X,'Q(2)=',E10.4,5X,'Q(3)=',E10.4,/,2X,'Q(4)=',
     $ E10.4,5X,'Q(5)=',E10.4,5X,'Q(6)=',E10.4,/,2X,'Q(7)=',E10.4,5X,
     $ 'Q(8)=',E10.4,5X,'Q(9)=',E10.4,/,2X,'Q(10)=',E10.4,/)
C
      Q(11)=Q(7)-Q(9)
      Q(12)=Q(8)-Q(10)
C       STORE THESE VALUES IN FILE PLOT.DATA FOR PLOT USAGE.
      WRITE (07,71) TAU,(Q(I),I=1,12)
71    FORMAT (13E15.7)
C
C
      IF (NS.GE.NSTEP) GO TO 105
      GO TO 79
C
C---INVALID OPTION---
C
 155  WRITE (06,1010)
      GO TO 110
C
C
 500  WRITE (06,1020)
C
      STOP
      END
C
C
      SUBROUTINE DEFALT
C*************************************************************
C
C
C*************************************************************
      REAL M2,M4,I1,I2,I3,I4,IE,N3
      REAL K2,K3,KH,KD,KE
C
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
```

```
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
C---SET VARIABLE CONSTANTS---
C
      FD=0.0
      RD=300.0
      A2=.281
      A1=1.0
      M2=10.0
      M4=1.4
      I1=169.48
      I2=0.5
      I3=5.95
      I4=30.0
      IE=200.0
      THK5=3.8
      THK4=0.6
      THK3=3.8
      D5=1.0
      RH=25.8
      RS=13.49
      UH=0.0
      N3=100.0
      UF=0.3
      TR=1.000
      RE=304.8
      DT=1.8
      RC=93.75
C
C
      RETURN
      END
C
C
      SUBROUTINE KUTTA (Q,QDOT,TINC)
C=============================================================
C
C=============================================================
      INTEGER N,CHR
      REAL Q(20),QDOT(20),M2,M4,I1,I2,I3,I4,IE,N3,OMEGA
      REAL K2,K3,KH,KD,KE
      REAL Q1(10),Q2(10),Q3(10),DQ1(10),DQ2(10),DQ3(10),DQ4(10),DQ(10)
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON/STATE/CHR,NS,NTE,ATE
      COMMON/VEL/VRNEW,VROLD
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
C
      OMEGA=Q(8)-Q(6)
C
      CALL TORQIN (OMEGA)
      IF (NS.GE.NTE) TE=TE+ATE
C       WRITE (06,11) TE
C 11    FORMAT (' ','CURRENT TE IS:',E10.4)
C
C--------PROGRAM MODIFIED SO USER INPUT FORCE--------
      CALL FORCIN (T)
```

```
C
C---STATE 1 TEST---
C
      IF(Q(1).GE.(-DT/A2)) GOTO 10
      CALL STATE1 (Q,QDOT)
      CHR=1      .
C
      DO 700 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
700   CONTINUE
C
      CALL STATE1 (Q1,QDOT)
C
      DO 710 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
710   CONTINUE
C
      CALL STATE1 (Q2,QDOT)
C
      DO 720 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
720   CONTINUE
C
     CALL STATE1 (Q3,QDOT)
C
      DO 730 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
730   CONTINUE
      RETURN
C
C---STATE 2 TEST---
C
10    IF(Q(3) .GE. .999) GO TO 20
      CALL STATE2 (Q,QDOT)
      CHR=2
C
      DO 800 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
800   CONTINUE
C
      CALL STATE2 (Q1,QDOT)
C
      DO 805 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
805   CONTINUE
C
      CALL STATE2 (Q2,QDOT)
C
      DO 810 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
810   CONTINUE
C
```

```
      CALL STATE2 (Q3,QDOT)
C
      DO 815 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
815   CONTINUE
      RETURN
C
C---STATE 3 TEST---
C
20    Q(4)=0.0
      IF (Q(1).GE.-.001) GO TO 60
      ICOUNT=ICOUNT+1
      IF (ICOUNT.EQ.1) VRNEW=Q(8)-Q(10)
      IF (ICOUNT.EQ.1) JJJ=0
C
C      NOATION:          (1) JJJ=0, SLIPPING.
C                        (3) JJJ=2, THE IMMEDIATE HISTORY IS ONE
C                             OF NON-SLIP.  THE INTERFACE SURFACES
C                             ARE BEGINNING TO SLIP.
C                        (2) JJJ=1, STICKING.
C
      IF (JJJ.EQ.1) GO TO 200
      IF ((VRNEW*VROLD) .LT. 0.0) GO TO 200
C
C
500   CALL STATE3 (Q,QDOT)
      IF (JJJ.EQ.0) CHR=3
      IF (JJJ.EQ.2) CHR=35
C
      DO 820 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
820   CONTINUE
C
      CALL STATE3 (Q1,QDOT)
C
      DO 825 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
825   CONTINUE
C
      CALL STATE3 (Q2,QDOT)
C
      DO 830 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
830   CONTINUE
C
      CALL STATE3 (Q3,QDOT)
C
      DO 835 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
835   CONTINUE
C
      VROLD=VRNEW
      VRNEW=Q(8)-Q(10)
```

```
C
C
C
      RETURN
C
C
 200  TT1 = TE
      NN1 = K3*((DT+A2*Q(1))-Q(3))
      AA = I1*(TT1)/(I1+I2+I3+I4)-TT1
      FT = NN1*UF*RC*2.0
C
      IF(ABS(AA) .GT. FT) JJJ=2
      IF(ABS(AA) .GT. FT) GO TO 500
      IF(ABS(AA) .LT. FT) JJJ=1
      IF(JJJ. NE. 1) RETURN
C
      CALL STAT3C(Q,QDOT)
      JJJ=1
      VRNEW = 0.0
      CHR = 33
C
      DO 840 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
840   CONTINUE
C
      CALL STAT3C (Q1,QDOT)
C
      DO 845 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
845   CONTINUE
C
      CALL STAT3C (Q2,QDOT)
C
      DO 850 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
850   CONTINUE
C
      CALL STAT3C (Q3,QDOT)
C
      DO 855 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
855   CONTINUE
      RETURN
C
C
C-----BEGIN STATE4 FRICTION LOGIC----
 60   Q(2)=0.0
      ICOUNT=ICOUNT+1
      IF (ICOUNT.EQ.1) GO TO 205
      WRITE(8,9999) Q(8),Q(10)
9999  FORMAT(2X,'Q(8)=',E10.4,5X,'Q(10)=',E10.4)
      IF (JJJ.EQ.2) WRITE(8,9998) VRNEW,VROLD
9998  FORMAT(2X,'VRNEW=',E10.4,4X,'VROLD=',E10.4)
      IF (JJJ.EQ.1) GO TO 205
      IF((VRNEW*VROLD) .LT. 0.0) GO TO 205
```

```
C
C
      JJJ=0
505   CALL STATE4 (Q,QDOT)
      IF (JJJ.EQ.0) CHR =4
      IF (JJJ.EQ.2) CHR =45
C
      DO 860 I=1,10
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
860   CONTINUE
C
      CALL STATE4 (Q1,QDOT)
C
      DO 865 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
865   CONTINUE
C
      CALL STATE4 (Q2,QDOT)
C
      DO 870 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
870   CONTINUE
C
      CALL STATE4 (Q3,QDOT)
C
      DO 875 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
875   CONTINUE
C
      VROLD=VRNEW
      VRNEW=Q(8)-Q(10)
C
C
      RETURN
C
205   K3=3000.0
      TT1 = TE
      NN1 = K3*((DT+A2*Q(1))-Q(3))
      AA = I1*(TT1)/(I1+I2+I3+I4)-TT1
C
C
      FT = NN1*UF*RC*2.0
C
      IF(ABS(AA) .GT. FT) JJJ=2
      IF(ABS(AA) .GT. FT) GO TO 505
      IF(ABS(AA) .LT. FT) JJJ=1
      IF(JJJ. NE. 1) RETURN
C
C
      CALL STAT4C (Q,QDOT)
      JJJ=1
      VRNEW = 0.0
      CHR=43
C
      DO 880 I=1,10
```

```
        DQ1(I)=TINC*QDOT(I)
        Q1(I)=Q(I)+DQ1(I)/2.0
880   CONTINUE
C
      CALL STAT4C (Q1,QDOT)
C
      DO 885 I=1,10
        DQ2(I)=TINC*QDOT(I)
        Q2(I)=Q(I)+DQ2(I)/2.0
885   CONTINUE
C
      CALL STAT4C (Q2,QDOT)
C
      DO 890 I=1,10
        DQ3(I)=TINC*QDOT(I)
        Q3(I)=Q(I)+DQ3(I)
890   CONTINUE
C
      CALL STAT4C (Q3,QDOT)
C
      DO 895 I=1,10
        DQ4(I)=TINC*QDOT(I)
        DQ(I)=(DQ1(I)+2.0*(DQ2(I)+DQ3(I))+DQ4(I))/6.0
        Q(I)=Q(I)+DQ(I)
895   CONTINUE
      RETURN
      END
C
C
C=======================================================
      SUBROUTINE STATE1 (Q,QDOT)
C=======================================================
C     Pressure plate not in contact with clutch;
C     clutch pedal starting return.
C=======================================================
      REAL Q(10),QDOT(10),K2,K3,KH,KD,KE,M2,M4,I1,I2,I3,I4,IE,N3
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
C     BELSPR and ENGINE supply non-linear parameters K2 and
C     KE,CE respectively.
C
      CALL BELSPR
      CALL ENGINE
C
C     STATE #1 SET I - ROTATIONAL EQUATIONS OF MOTION
C     Flywheel/clutch cover/pressure plate/engine crank assembly
C
      QDOT(7)=Q(8)
      QDOT(8)=TE/I1
C
C     STATE #1 SET II - ROTATIONAL EQUATIONS OF MOTION
C     Engine block
C
      QDOT(5)=Q(6)
      QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
C
```

```
C      Engine block rotation.
C
       QDOT(5)=Q(6)
       QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
C
C      STATE #2 SET I - TRANSLATIONAL EQUATIONS OF MOTION
C      Pressure plate travel.
C
       THKT=9.34
C
       QDOT(1)=Q(2)
       QDOT(2)=K2/(M2*A2)*(THKT-A2*Q(1))-
      $FINPUT/(M2*A2)-(C2+C3)/M2*Q(2)+C3/(M2*A2)*Q(4)-
      $K3/(M2*A2)*(DT+A2*Q(1)-Q(3))
C
C      STATE #2 SET II - TRANSLATIONAL EQUATIONS OF MOTION
C      Clutch displacement
C
       QDOT(3)=Q(4)
       QDOT(4)=C3/M4*(Q(2)*A2-Q(4))+K3/M4*(DT+A2*Q(1)-Q(3))
       RETURN
       END
C
C=============================================================
       SUBROUTINE STATE3 (Q,QDOT)
C=============================================================
C      JJJ=0
C      Pressure plate and flywheel both in contact with clutch.
C      Interface surfaces are slipping.  Clutch hub springs
C      are not fully compressed.
C
C      JJJ=2
C      The immediate history is one of non-slip.  Therefore,
C      the clutch is contacting the flywheel and the pressure
C      plate.  The interface surfaces are BEGINNING to slip.
C      The direction of slip is determined by the sign of
C      the quantity "AA" below.
C=============================================================
       REAL Q(10),QDOT(10),K2,K3,K4,KH,KD,KE,N1
      $,M2,M4,I1,I2,I3,I4,IE,N3,N1MN2
C
       COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
       COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
      $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
      $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C      BELSPR, MARCEL, and ENGINE supply non-linear
C      parameters (K2,C2), (K3,C3), and (KE,CE)
C      respectively.
C
       CALL BELSPR
       CALL MARCEL
       CALL ENGINE
C
C      N1, and T2 represent clutch normal
C      force, and clutch output torque respectively.
C      These quantities are used frequently in
C      subsequent equations and logic throughout the program.
C
       N1=(K3*(DT+A2*Q(1)-Q(3))+C3*((A2*Q(2))-Q(4)))
       T2=N1*UF*RC*2.0
```

```
C      STATE #1 SET I - TRANSLATIONAL EQUATIONS OF MOTION
C      Pressure Plate
C
       THKT=9.34
       QDOT(1)=Q(2)
       QDOT(2)=K2/(M2*A2)*(THKT-A2*Q(1))-
      $FINPUT/(M2*A2)-(C2/M2)*Q(2)
C
       QDOT(3)  = 0.
       QDOT(4)  = 0.
       QDOT(9)  = 0.
       QDOT(10) = 0.
       RETURN
       END
C
C
C=============================================================
       SUBROUTINE STATE2 (Q,QDOT)
C=============================================================
C      Pressure plate in contact with clutch; flywheel
C      not in contact with clutch; slipping at clutch/pressure plate
C      interface assumed
C=============================================================
       REAL Q(10),QDOT(10),K2,K3,KH,KD,KE,M2,M4,I1,I2,
      &     I3,I4,IE,N3,N1,K4
C
       COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
       COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
      $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
      $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
C      BELSPR, MARCEL, and ENGINE supply non-linear
C      parameters (K2,C2), (K3,C3), and (KE,CE)
C      respectively.
C
       CALL BELSPR
       CALL MARCEL
       CALL ENGINE
C
C      N1, and T2 represent clutch normal
C      force, and clutch output torque respectively.
C      These quantities are used frequently in
C      subsequent equations and logic throughout the program.
C
       N1=K3*((DT+A2*Q(1))-Q(3))+C3*((A2*Q(2))-Q(4))
       T1=N1*UF*RC
C
C      STATE #2 SET I - ROTATIONAL EQUATIONS OF MOTION
C      Flywheel assembly
C
       QDOT(7)=Q(8)
       QDOT(8)=(TE-T1)/I1
C
C      STATE #2 SET II ROTATIONAL EQUATIONS OF MOTION
C      Outer clutch rotation
C
       QDOT(9)=Q(10)
       QDOT(10)=(T1)/(I2+I3+I4)
C
C      STATE #2 SET V - ROTATIONAL EQUATIONS OF MOTION
```

```
C
C      STATE #3 SET I - ROTATIONAL EQUATIONS OF MOTION
C      Equations check for direction of relative rotation  (i.e. friction
C      torque direction) between flywheel assembly and clutch during
C      evaluation of flywheel assembly output values.
C
      QDOT(7)=Q(8)
      IF (JJJ.EQ.0) QDOT(8) = (TE-SIGN(1.0,Q(8)-Q(10))*T2)/I1
      IF (JJJ.EQ.2) QDOT(8) = (TE+SIGN(1.0,AA)*T2)/I1
C
C      STATE #3 SET II - ROTATIONAL EQUATIONS OF MOTION
C      Equations check for direction of relative rotation
C      between flywheel assembly and clutch during
C      evaluation of clutch output characteristics.
C
      QDOT(9)=Q(10)
      IF (JJJ.EQ.0) QDOT(10)=(SIGN(1.0,Q(8)-Q(10))*T2)/(I2+I3+I4)
      IF (JJJ.EQ.2) QDOT(10)=-(SIGN(1.0,AA)*T2)/(I2+I3+I4)
C
C      STATE #3 SET V - ROTATIONAL EQUATIONS OF MOTION
C      Engine block rotation.
C
      QDOT(5)=Q(6)
      QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
C
C      STATE #3 SET I - TRANSLATIONAL EQUATIONS OF MOTION
C      Pressure plate
C
      THKT=9.34
C
      QDOT(1)=Q(2)
      QDOT(2)=K2/(M2*A2)*(THKT-A2*Q(1))-
     $FINPUT/(M2*A2)-C2/M2*Q(2)-C3/(M2*A2)*(Q(2)*A2-Q(4))-
     $K3/(M2*A2)*(DT+A2*Q(1)-Q(3))
C
C      Constraints for variables with fixed values;
C      specifically, the clutch is in contact with
C      the flywheel.
C
      QDOT(3) = Q(4)
      QDOT(4)=0.0
      RETURN
      END
C
C=======================================================
      SUBROUTINE STAT3C (Q,QDOT)
C=======================================================
C      Pressure plate and flywheel both in contact with clutch.
C      Interface surfaces are NOT slipping.  Clutch hub springs
C      are NOT fully compressed; i.e. the stop pins are NOT
C      contacted.
C=======================================================
      REAL Q(10),QDOT(10),K2,K3,K4,KH,KD,KE,N3,I1,I2,I3,I4,IE,
     &      M2,M4,N1,N2
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
```

```
C      BELSPR, MARCEL, and ENGINE supply non-linear
C      parameters (K2,C2), (K3,C3), and (KE,CE)
C      respectively.
C
       CALL BELSPR
       CALL MARCEL
       CALL ENGINE
C
C---STATE #3C SET I - ROTATIONAL EQUATIONS OF MOTION
C      No slip is occuring at the clutch surface, therefore, the
C      flywheel assembly and clutch are treated as one rigid body.
C      For this reason, the respective rotational values are equated below.
C
       QDOT(7)=Q(8)
       QDOT(8)=(TE)/(I1+I2+I3+I4)
       QDOT(10)=(TE)/(I1+I2+I3+I4)
       QDOT(9)=Q(10)
C
C      STATE #3C SET IV - ROTATIONAL EQUATIONS OF MOTION
C      Engine block rotation.
C
       QDOT(5)=Q(6)
       QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
C
C      STATE #3C SET I - TRANSLATIONAL EQUATIONS OF MOTION
C      Pressure plate
C
       THKT=9.34
C
       QDOT(1)=Q(2)
       QDOT(2)=(K2/(M2*A2))*(THKT-A2*Q(1))-
      $FINPUT/(M2*A2)-C2/M2*Q(2)-C3/(M2*A2)*(Q(2)*A2-Q(4))-
      $K3/(M2*A2)*(DT+A2*Q(1)-Q(3))
C
C---STATE #3C SET II - TRANSLATIONAL EQUATIONS---
C
C      Constraints for variables with fixed values;
C      specifically, the clutch is in contact with
C      the flywheel.
C
       QDOT(3)=Q(4)
       QDOT(4)=0.0
       RETURN
       END
C
C=================================================================
       SUBROUTINE STATE4 (Q,QDOT)
C=================================================================
C      JJJ=0, slipping.
C      JJJ=2, the immediate history is one of non-slip.
C           The interface surfaces are BEGINNING to slip.
C=================================================================
       REAL Q(10),QDOT(10),K2,K3,K4,KH,KD,KE,M2,M4,I1,I2,
      $      I3,I4,IE,N1,N3
C
       COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
       COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
      $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
      $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
```

```
      CALL ENGINE
      CALL BELSPR
      CALL MARCEL
      N1=(K3*(DT+A2*Q(1)-Q(3))+C3*((A2*Q(2))-Q(4)))
      T2=N1*UF*RC*2.0
C
C---STATE #4 SET I - ROTATIONAL EQUATIONS---
C
      QDOT(7)=Q(8)
      IF (JJJ.EQ.0) QDOT(8)=(TE-SIGN(1.0,Q(8)-Q(10))*T2)/I1
      IF (JJJ.EQ.2) QDOT(8)=(TE+SIGN(1.0,AA)*T2)/I1
C
C
C---STATE #4 SET II & III - ROTATIONAL EQUATIONS---
C
      QDOT(9)=Q(10)
      IF (JJJ.EQ.0) QDOT(10)=(SIGN(1.0,Q(8)-Q(10))*T2)/(I2+I3+I4)
      IF (JJJ.EQ.2) QDOT(10)=-(SIGN(1.0,AA)*T2)/(I2+I3+I4)
C
C---STATE #4 SET V - ROTATIONAL EQUATIONS---
C
      QDOT(5)=Q(6)
      QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
C
C---STATE #4 SET VI - TRANSLATIONAL EQUATIONS - CONSTRAINTS---
C
      QDOT(1) = Q(2)
      QDOT(2) = 0.0
      QDOT(3) = Q(4)
      QDOT(4) = 0.0
C
      RETURN
      END
C
C=======================================================
      SUBROUTINE STAT4C (Q,QDOT)
C=======================================================
C
C
C=======================================================
      REAL Q(10),QDOT(10),KH,KD,KE,K2,K3,
     $     I1,I2,I3,I4,IE,N3,M2,M4
C
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
      CALL ENGINE
      CALL MARCEL
      CALL BELSPR
C
C
      QDOT(7)=Q(8)
      QDOT(8)=(TE)/(I1+I2+I3+I4)
C      CONSTRAINTS
      QDOT(9)=Q(10)
      QDOT(10)=QDOT(8)
      QDOT(1)=Q(2)
      QDOT(2)=0.0
```

```
      QDOT(3)=Q(4)
      QDOT(4)=0.0
      QDOT(5)=Q(6)
      QDOT(6)=-(RE/IE)*(CE*Q(6)+KE*Q(5))+TE/IE
      RETURN
      END
C
C
C
      SUBROUTINE TORQIN (OMEGA)
C=============================================================
C
C=============================================================
      REAL TE,K2,K3,KH,KD,KE
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      TE=10000.0
      RETURN
      END
C
C
      SUBROUTINE FORCIN (T)
C=============================================================
C
C=============================================================
      REAL T,FINPUT,K2,K3,KH,KD,KE
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
      FINPUT=0.0
      RETURN
      END
C
C
      SUBROUTINE BELSPR
C=============================================================
C
C=============================================================
      REAL K2,C2,K3,KH,KD,KE
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
      K2=200.0
      C2=0.0
      RETURN
      END
C
C
      SUBROUTINE MARCEL
C=============================================================
C
C=============================================================
      REAL K3,C3,K4,C4,K2,KH,KD,KE,M2,M4,I1,I2,I3,I4,IE
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
      K3=3000.0
      C3 = 0.0
      RETURN
      END
C
```

```
C
      SUBROUTINE ENGINE
C=======================================================================
C
C=======================================================================
      REAL KE,CE,K2,K3,KH,KD
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
C
      KE=2000.0
      CE=10.0
      RETURN
      END
C
C
       SUBROUTINE GRAPH
C=======================================================
C
C=======================================================
       LOGICAL*1 XT(50),YT(50),NAME(8)
       REAL X1(2500),Y1(2500),DUM(12)
       INTEGER DD,TT,QQ,ICODE(12)
C
      DATA DD,TT,QQ / 'D ','T ','? ' /
      DATA ICODE / '1 ','2 ','3 ','4 ','5 ','6 ','7 ',
     &             '8 ','9 ','10','11','12' /
C
C
50     FORMAT (13F15 7)
70     FORMAT(5X,'ENTER X-AXIS LABEL')
80     FORMAT(50A1)
90     FORMAT(5X,'ENTER Y-AXIS LABEL')
C
1000 FORMAT (/,' WHICH VARIABLE?--ENTER "?" FOR MENU')
1002 FORMAT (/,' VARIABLES ARE ::',
     &         /,' 1    Q1',/,' 2    Q2',/,' 3    Q3',
     &         /,' 4    Q4',/,' 5    Q5',/,' 6    Q6',
     &         /,' 7    Q7',/,' 7    Q7',/,' 8    Q8',
     &         /,' 9    Q9',/,' 10   Q10',/,' 11   Q7-Q9'
     &         /,' 12   Q8-Q10',/,' D   DONE')
1006 FORMAT (//' *** INVALID ENTRY--TRY AGAIN ***')
2000 FORMAT (A2)
2002 FORMAT (1X,I4)
2004 FORMAT (I2)
C
C
C
C---DEFINE VARIABLE TO BE PLOTTED VERSUS TIME---
C
 100 WRITE (06,1000)
     READ (05,2000) IANS
C
C---PRINT LIST OF VARIABLES, IF REQUESTED---
C
      IF (IANS .NE. QQ) GO TO 110
         WRITE (06,1002)
         GO TO 100
C
C---IF USER DONE, RETURN---
C
 110 IF (IANS .EQ. DD) GO TO 500
```

```
C
C---TORQUE---
C
C       IF (IANS .NE. TT) GO TO 10
C          NN = 1
C          GO TO 30
C
C---PROCESS ANY OF THE Q'S, IF CHOSEN---
C
  10 DO 20 I=1,12
        IF (IANS .EQ. ICODE(I)) GO TO 25
  20 CONTINUE
     WRITE (06,1006)
     GO TO 100
C
  25 NN = I
C
C---REWIND TEMPORARY FILE 7---
C
  30  REWIND 7
C
C---READ FROM FILE 7 AND PACK X,Y ARRAYS---
C
      READ (07,19) TFINAL
19    FORMAT (E10.4)
      READ (07,2002) N1
      DO 40 I=1,N1
        IF (NN .NE. 1) GO TO 35
        READ (07,50) X1(I),Y1(I),(DUM(J),J=1,11)
        Y1(I)=-Y1(I)
        GO TO 40
C
C---READ TORQUE PLOT INFO---
C   FOR THIS VERSION, TORQUE IS NO LONGER A VARIABLE.
C
C             READ (07,50) X1(I),Y1(I),(DUM(J),J=1,12),ICHR
C             GO TO 40
C
  35    IF (NN .NE. 12) GO TO 37
           READ (07,50) X1(I),(DUM(J),J=1,11),Y1(I)
           GO TO 40
C
  37    NZ = NN-1
        NF = 12-NN
        READ (07,50) X1(I),(DUM(J),J=1,NZ),Y1(I),(DUM(J),J=1,NF)
C
40    CONTINUE
C
C
C---GET AXIS LABEL INFORMATION---
C
      WRITE(6,70)
      READ(5,80) XT
      CALL COUNTC(XT,50,NXT)
      WRITE(6,90)
      READ(5,80) YT
      CALL COUNTC(YT,40,NYT)
C
C
22    FORMAT (F10.4)
```

```
      XF=TFINAL/5.
      XM=0.0
C
      CALL PSCALE (3.,0.5,YM,YF,Y1,N1,1)
      CALL PLTOFS (XM,XF,YM,YF,1.,1.)
      CALL PLINE (X1,Y1,N1,1,0,0,1)
      CALL PAXIS (1.,1.,XT,-NXT,5.,0.,XM,XF,.5)
      CALL PAXIS (1.,1.,YT,NYT,3.,90.,YM,YF,.5)
      CALL PLTEND
      GO TO 100
 500  CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE COUNTC (STRN,MAXC,NCHR)
C=============================================================
C
C=============================================================
      LOGICAL EQUC
      LOGICAL*1 STRN(1),BLNK
      DATA BLNK/' '/
      DO 20 I=1,MAXC
      IF(EQUC(STRN(I),BLNK)) GO TO 10
      GO TO 20
10    IF(EQUC(STRN(I-1),BLNK)) GO TO 15
      GO TO 20
15    NCHR=I-2
      GO TO 30
20    CONTINUE
      NCHR=I
      IF(EQUC(STRN(I),BLNK)) NCHR=I-1
30    RETURN
      END
C
C
     SUBROUTINE FILE (IUNIT,NAME)
C**************************************************************************
C
C  THIS SUBROUTINE ASSIGNS FORTRAN UNIT NUMBER, IUNIT, TO A FILE OF
C  WHICH IS CALLED NAME
C
C**************************************************************************
     LOGICAL*1 NAME(30)
     CALL SETLIO (IUNIT,NAME,&85)
     RETURN
  85 WRITE (06,87)
  87 FORMAT (' ILLEGAL I/O UNIT OR FILE NAME')
     STOP
     END
C
C
     SUBROUTINE MODIFY
C=============================================================
C
C=============================================================
     INTEGER ICODEA(13),ICODEB(16),QQ
     REAL K2,K3,KH,KD,KE,M2,M4,I1,I2,I3,I4,IE,N3
C
```

```
      COMMON /VARS/ TE,FINPUT,K2,K3,KE,C2,C3,CE,FD,RD,IFL
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
      DATA JQ / 0 /
      DATA ICODEA / 'TE','FI','K2','K3','KH','KD','KE','C2',
     &              'C3','CE','BD','FD','RD' /
      DATA ICODEB / 'M2','M4','I1','I2','I3','I4','IE','RH',
     &              'UH','US','N3','UF','TR','RE','RC','A2' /
      DATA DD / 'D ' /
      DATA QQ / '?' /
C
C
1000 FORMAT (/,' ON ONE INPUT LINE, ENTER VARIABLE CODE',
     &      ' AND MODIFY ID (C...CONSTANT ; L...LEAST SQUARES FIT).',
     &     /,'TERMINATE INPUTS WITH "D <CR>". FOR',
     &      ' VARIABLE LISTING AND EXAMPLE ENTER "?" ')
1002 FORMAT (//,' VARIABLES THAT MAY BE CHANGED ARE ::',
     &        /,'   TE    -- ENGINE TORQUE',
     &        /,'   FI    -- CLUTCH PEDAL FORCE',
     &        /,'   K2    -- BELLEVILLE SPRING RATE',
     &        /,'   K3    -- CUSHION SPRING RATE',
     &        /,'   KE    -- ENGINE MOUNT STIFFNESS',
     &        /,'   KH    -- TORSIONAL CLUTCH SPRING RATE',
     &        /,'   KD    -- TORSIONAL DRIVE TRAIN STIFFNESS',
     &        /,'   C2    -- BELLEVILLE LINKAGE DAMPING COEFFICIENT',
     &        /,'   C3    -- CUSHION DAMPING COEFFICIENT',
     &        /,'   CE    -- ENGINE MOUNT DAMPING COEFFICIENT',
     &        /,'   BD    -- DRIVE TRAIN VISCOUS DAMPING COEFFICIENT',
     &        /,'   FD    -- ROLLING DRAG FORCE ON TIRE',
     &        /,'   RD    -- TIRE ROLLING RADIUS',
     &        //,' EXAMPLE INPUT SEQUENCE ::'
     &        /,'   K3,C <CR>    OR     D  <CR>',/)
1010 FORMAT (//,' ** INVALID VARIABLE CODE INPUT--',
     &         'RE-ENTER LINE **')
2000 FORMAT (A2,A2)
C
C
C
  110 WRITE (06,1000)
  120 READ (05,2000) IVAR,ICD
      IF (IVAR .NE. QQ) GO TO 200
         WRITE (06,1002)
         JQ = JQ+1
         IF (JQ .GT. 1) GO TO 120
         GO TO 110
C
C---CHECK IVAR FOR VALID INPUT---
C
  200 IC = 1
      DO 300 I=1,13
         IF (IVAR .EQ. ICODEA(I)) GO TO 600
  300 CONTINUE
C
  400 IC = 2
      DO 500 I=1,16
         IF (IVAR .EQ. ICODEB(I)) GO TO 600
  500 CONTINUE
      WRITE (06,1010)
```

```
      GO TO 120
C
C
  600 IV = I
      CALL MODCON (IC,IV)
      RETURN
      END
C
C
      SUBROUTINE PRINTA (Q,N)
C=============================================================
C
C=============================================================
      LOGICAL*1 NAME(30)
      REAL Q(N),M2,M4,N3,I1,I2,I3,I4,IE
      INTEGER TT,FF
C
      COMMON /PAR/ M2,M4,I1,I2,I3,I4,IE,
     $THKC,THK5,THK4,THK3,D2,D3,D5,RH,US,N3,UF,TR,
     $RE,RC,A2,A1,AA,ICOUNT,DT,JJJ
C
      DATA TT,FF / 'T','F' /
 1000 FORMAT (/,' WOULD YOU LIKE YOUR TABLE TO BE PRINTED',
     &        ' AT THIS TERMINAL',/,' OR SENT TO AN OUTPUT FILE',
     &        ' (T OR F) ?')
 1002 FORMAT (/,' PLEASE ENTER FILE NAME TO EMPTY DATA TABLE')
 1004 FORMAT (//,'  Q1',6X,'Q2',6X,'Q3',6X,'Q4',6X,'Q5',6X,'Q6',
     &            6X,'Q7',6X,'Q8',6X,'Q9',5X,'Q10',/,' Q11',5X,'Q12',
     &            5X,'Q13',5X,'Q14',5X,'QD1',5X,'QD2',5X,'QD3',
     &            5X,'QD4',5X,'QD5',5X,'QD6',/,'  QD7',5X,'QD8',5X,
     &           'QD9',4X,'QD10',4X,'QD11',4X,'QD12',4X,'QD13',
     &           4X,'QD14',4X,'STATE')
 2000 FORMAT (A1)
 2002 FORMAT (30A1)
 2004 FORMAT (2X,2E10.4,/,2X,6E10.4,/,2X,6E10.4,/,
     $2X,2E10.4,/,2X,I8)
 2006 FORMAT (1X,I4)
 2008 FORMAT (2X,7F10.4,/,2X,4F10.4,/2X,5F10.4,/,2X,I8)
C
      WRITE (06,1000)
      READ (05,2000) IANS
C
      IF (IANS .EQ. TT) GO TO 200
         IFILE = 4
         WRITE (06,1002)
         READ (05,2002) NAME
         CALL FILE (IFILE,NAME)
         GO TO 300
C
  200 IFILE = 6
C
C
C---WRITE OUT TITLES FOR Q AND QDOT OUTPUT---
C---ON DATA CHART FILE---
C
  300  WRITE (IFILE,1004)
C
C---REWIND FILE 8 FOR READING---
C
      REWIND 8
```

```
C
C---READ NUMBER OF STEPS---
C
      READ (08,2006) NSTEP
C
C---FOR EACH TIME STEP---
C
      DO 500 I=1,NSTEP
         READ (08,2004) T,TOR,(Q(J),J=1,10),ICHR
         WRITE (IFILE,2008) T,TOR,(Q(J),J=1,10),ICHR
  500 CONTINUE
C
      RETURN
      END
C
C
      SUBROUTINE MODCON (IC,IV)
C=================================================
C
C=================================================
      REAL RA(13),RB1(7),RD(7),RB2(8),DUM(3)
C
      COMMON /VARS/ RA,IDUM
      COMMON /PAR/ RB1,RD,RB2,DUM
C
1000 FORMAT (/,' ENTER NEW CONSTANT VALUE')
1002 FORMAT (/,' *** INVALID IC IN MODCON *****')
2000 FORMAT (F9.4)
C
  300 IF (IC .NE. 1) GO TO 500
         WRITE (06,1000)
         READ (05,2000) RA(IV)
         WRITE (06,111) IV,RA(IV)
  111    FORMAT (' --VARS ARRAY---',I6,F10.4)
         GO TO 900
C
  500 IF (IC .NE. 2) GO TO 800
         WRITE (06,1000)
         IF (IV .GT. 7) GO TO 700
            READ (05,2000) RB1(IV)
            WRITE (06,222) IV,RB1(IV)
  222       FORMAT (' --PAR ARRAY---',I6,F10.4)
            GO TO 900
C
  700 IN = IV-7
      READ (05,2000) RB2(IN)
      WRITE (06,222) IN,RB2(IN)
C
  800 WRITE (06,1002)
C
  900 CONTINUE
      RETURN
      END
```