-039
HTSA-81-23.III

A Computer Program (VEHSIM) for Vehicle Fuel Economy and Performance Simulation (Automobiles and Light Trucks)

Volume III: Glossary and Listings

Russell W. Zub

Transportation Systems Center
Cambridge MA 02142

October 1981
Final Report

This document is available to the public through the National Technical Information Service, Springfield, Virginia 22161.

U.S. Department of Transportation
National Highway Traffic Safety Administration

Office of Research and Development
Washington DC 20590

NOTICE

This document is disseminated under the sponsorship of the Department of Transportation in the interest of information exchange. The United States Government assumes no liability for its contents or use thereof.

NOTICE

The United States Government does not endorse products or manufacturers. Trade or manufacturers' names appear herein solely because they are considered essential to the object of this report.

Technical Report Documentation Page

1. Report No.
DOT-HS-806-039

2. Government Accession No.

3. Recipient's Catalog No.

4. Title and Subtitle
A COMPUTER PROGRAM (VEHSIM) FOR VEHICLE FUEL ECONOMY AND PERFORMANCE SIMULATION (AUTOMOBILES AND LIGHT TRUCKS)
Volume III: Glossary and Listings

5. Report Date
October 1981

6. Performing Organization Code
DTS-323

7. Author's)
Russell W. Zub

8. Performing Organization Report No.
DOT-TSC-NHTSA-81-23.III

9. Performing Organization Name and Address
U.S. Department of Transportation
Research and Special Programs Administration
Transportation Systems Center
Cambridge MA 02142

10. Work Unit No. (TRAIS)
HS273/R2410

11. Contract or Grant No.

12. Sponsoring Agency Name and Address
U.S. Department of Transportation
National Highway Traffic Safety Administration
Office of Research and Development
Washington DC 20590

13. Type of Report and Period Covered
Final Report
Jan 1980-Aug 1981

14. Sponsoring Agency Code
NRD-13

15. Supplementary Notes

16. Abstract

This report presents an updated description of a vehicle simulation program, VEHSIM, which can determine the fuel economy and performance of a specified vehicle over a defined route as it executes a given driving schedule. Vehicle input accommodated by VEHSIM include accessories, engine, rear axle, converter transmission, tires, aerodynamic drag coefficient, and shift logic. The report is comprised of four volumes. Volume I presents a description of the numerical approach and equations, Volume II is a user's manual, Volume III contains the program listings and Volume IV describes a simulation of the Integrated Overdrive Transmission with a split-torque converter.

17. Key Words
Motor Vehicle, Simulation, Fuel Economy, Performance, Engine Map

18. Distribution Statement
DOCUMENT IS AVAILABLE TO THE PUBLIC THROUGH THE NATIONAL TECHNICAL INFORMATION SERVICE, SPRINGFIELD, VIRGINIA 22161

19. Security Classif. (of this report)
Unclassified

20. Security Classif. (of this page)
Unclassified

21. No. of Pages
332

22. Price

Reproduction of completed page authorized

PREFACE

Volume III is the continuation of a multi document set describing VEHSIM. This volume includes a glossary, source code listing, flow charts of selected subroutines and listing of graphics package. The glossary is a listing of variables and preset alphanumeric variables used in processing input data. An attempt was made to include a description of all these variables. However, in some instances the subroutine that the stated variable is located in is all that is listed. Additional information on these variables can be obtained by examining the context of the variable use in that subroutine. The VEHSIM source code is simply a listing of the Fortran code. The flow charts were provided for the most commonly used subroutines. The graphics package provides the user with optional graphical capability. The applicability of these subroutines is determined by the users needs. Scanning engine/data by graphical output has proved useful.

METRIC CONVERSION FACTORS

Approximate Conversions to Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
in	inches	2.5	centimeters	cm
ft	feet	30	centimeters	cm
yd	yards	0.9	meters	m
mi	miles	1.6	kilometers	km
		AREA		
in^2	square inches	6.5	square centimeters	cm^2
ft^2	square feet	0.09	square meters	m^2
yd^2	square yards	0.8	square meters	m^2
mi^2	square miles	2.6	square kilometers	km^2
	acres	0.4	hectares	ha
		MASS (weight)		
oz	ounces	28	grams	g
lb	pounds	0.45	kilograms	kg
	short tons (2000 lb)	0.9	tonnes	t
		VOLUME		
tsp	teaspoons	5	milliliters	ml
Tbsp	tablespoons	15	milliliters	ml
fl oz	fluid ounces	30	milliliters	ml
c	cups	0.24	liters	l
pt	pints	0.47	liters	l
qt	quarts	0.95	liters	l
gal	gallons	3.8	liters	l
ft^3	cubic feet	0.03	cubic meters	m^3
yd^3	cubic yards	0.76	cubic meters	m^3
		TEMPERATURE (exact)		
°F	Fahrenheit temperature	5/9 (after subtracting 32)	Celsius temperature	°C

Approximate Conversions from Metric Measures

Symbol	When You Know	Multiply by	To Find	Symbol
		LENGTH		
mm	millimeters	0.04	inches	in
cm	centimeters	0.4	inches	in
m	meters	3.3	feet	ft
m	meters	1.1	yards	yd
km	kilometers	0.6	miles	mi
		AREA		
cm^2	square centimeters	0.16	square inches	in^2
m^2	square meters	1.2	square yards	yd^2
km^2	square kilometers	0.4	square miles	mi^2
ha	hectares (10,000 m^2)	2.5	acres	
		MASS (weight)		
g	grams	0.035	ounces	oz
kg	kilograms	2.2	pounds	lb
t	tonnes (1000 kg)	1.1	short tons	
		VOLUME		
ml	milliliters	0.03	fluid ounces	fl oz
l	liters	2.1	pints	pt
l	liters	1.06	quarts	qt
l	liters	0.26	gallons	gal
m^3	cubic meters	35	cubic feet	ft^3
m^3	cubic meters	1.3	cubic yards	yd^3
		TEMPERATURE (exact)		
°C	Celsius temperature	9/5 (then add 32)	Fahrenheit temperature	°F

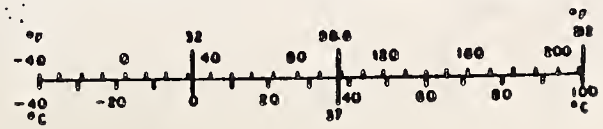


TABLE OF CONTENTS

1. GLOSSARY

Glossary for the VEHSIM program

DECEMBER 1980

A0	Initial condition-acceleration
AA1	FRC1*WGT
AA2	FRC2*WGT
AA3	FAC*CD*AREA
AA4	WGT/32.17
AA5	14./WRAD
AA6	BB1*(4*AIW+RARSQ*AIP)
AA7	BB1*RARSQ
AA8	AA8=RAR*ERAR
AA9	BB1*(EINER+AIA+AI1)
AA10	7.48052/(FSPGR*62.426i34)
AA11	Converter pump inertia
AA12	Converter turbine inertia
AASE	Required acceleration for a constant acceleration segment
ABR	Wheel hub torque during current time step
ABRO	Same during previous time step
ABROO	Save old brake
ABRUPT	Dimensioned, but not used
ACCEL	The current acceleration at the wheels
ACCESS	The name of a blocks common statement. Also , the mode parameter of the OPEN statement
ACCO	Acceleration
ACCOO	Old value of acceleration.

ACCS	Accessory rpm table array
ACCT	Accessory torque table array
ACOM	Accessory comment array
ADSE	Terminal distance specified at a relative segment end point
AEFFG	Efficiency of the current gear
AGRAT	Gear ratio
AIA	Accesory input inertia array
AIAS	Accessory loss inertia array
AIGIN	Gear input inertia Iq,in
AIGOUT	Gear output inertia iq,out
AIP	Propshaft inertia
AIW	Wheel inertia ,Iw
AI2	Torque converter turbine inertia
AKC	Coast converter input speed array
AKD	Drive converter capacity factor array Kc
ALOSSC	Horsepower loss in converter
ALOSCO	Same at previous time step
ALOSSG	Absolute value of horsepower losses within a gear box
ALOSGO	Same during preceding time step
ALOSSR	ABSolute value of horsepower losses within a differential box
ALOSRO	Same at previous time step
ALOSSL	Absolute value of horsepower losses within converter
ALOSLO	Same at previous time step
ANA	Array of numbers of data points for all the accessories
ANAME	Accessory name

APSC	Passing clearance defining segment
APCS	PASSING CLEARANCE DEFINING THE CURRENT SEGMENT END
APOS	Absolute mile post defining the current segment end
APPEND	Command to cause a new file to be appanded an old file, effectively saving what has been done.
APPN	Accessory project, programmer number
APROT	Accessory protection code
APWO	Constant % wide open throttle required for entire segment
APWOO	0.01*APWO
AREA	Vehicle frontal area (A)
ARRIVE	Logical flag indicating arrival at a requested acceleration
ASEG	Array of constant accelaration required by driving segments
ATHOLLD	Array of acceleration limits required for the drive segment
ATHR	Max rate of change for throttle for current driving segment
ATSE	Specified duration time limit for ending current driving segment
AVEL	Constant velocity to be held for entire length of current driving segment
AVSE	Terminal velocity defining the end of current driving segment (Ve,n)
AXTORQ	Axle torque array,included with vehicle or time.
BASDEV	Data base device
BASFIL	Data base file.
BASPPN	Data base project, programmer number.

BATCH	Logics for BATCH mode
BB1	2*pi/60
BB2	1/5252
BINARY	A parameter of the MODE option in the OPEN or CLOSE statement which defines the character set of an external file or record
BN	Variable used for the name of a part
BORE	Engine bore size
BPER	Current slope in % of grade being used
BPERO	Save the original route grade
BPPN	Project programmer number used for a directory of a part
BPROT	Protection code used for a directory of a part
BSFC	Brake specific fuel consumption (lb./hp.-hr)
BT	Variable used for part type
C1TO7	The sum of the percents of the engine hp-hr for accessories, torque converter, clutch, gear box, differential, tire slip
C2345	The sum of the percents of the engine hp-hr of the of the torque converter, clutch, gear box, differential and tire slip
C345	The sum of the percents of the engine hp-hr of the clutch,gear box,differential, and tire slip.
CAC	Accessory energy loss accumulator
CAE	Aerodynamic drag loss accumulator
CALLT	Percent total of the energy losses
CARD	Variable array used to indicate an input record.
CBL	REFERENCED IN INPBAT
CBR	Brake loss accumulator
CCL	energy loss accumulator
CCOM	Torque converter comment array

CCPPN	Coast converter project, programmer number
CCPROT	Coast converter protection code
CD	Vehicle drag coefficient (Cd)
CDA	Distance driven during acceleration
CDATE	The date on which a file was read and written
CDC	Sensisivity of the drag coefficient
CDCR	Distance driven during cruise
CDD	Distance driven during acceleration
CDEBUG	The name of a block common statement
CDI	Distance driven during idle
CDIAM	Torque converte diameter
CDIF	Differential box energy loss accumulators
CDPPN	Drive converter project, programmer number
CDPROT	Drive converter protection code
CEA	Engine energy used during acceleration
CECR	Engine energy used during cruise
CED	Engine energy used during deceleration
CEDN	Negative energy used during deceleration
CEU	Energy used during idle
CENERG	The present of the non-specific energy loses
CFA	Fuel consumed during acceleration
CFB	Fuel consumed during braking
CFCR	Fuel consumed during cruise
CFD	Fuel consumed during deceleration
CFI	Fuel consumed during idle

CGB	Gear box energy loss accumulator
CHOUR	The hour that the part was used on Colated
CKE	Kinetic energy of a car
CLSIPT	Subroutine used to identify a particular type of output taska30400
CLUTCH	logical flag used for the clutch being engaged or disengaged
CNAME	Convert name
CNTRL	Block common name
CNVNAM	Array to hold the name of the drive and coast converter
CNVTYP	Array used to store the hollerith literals 'DRIVE' and 'COAST'.
COAST	Logical flag indicating drive or coast condition to select converter
COL1i	
COMND	Variable for the different types of commands, USE,MOD, etc.
CONSHF	Blocks common name
CONST	6./stroke
CONTOUR	Constant input torque (S.R.converter data sheet, line 'sasa')
CPE	Energy generated by gravity force at the route segment with a grade
CPHI	Cosine of the wind angle
CPUS	The current drive segment CPU time
CPUT	Current cpu time
CRO	Wheel rolling energy loss
CROT	Kinetic energy of rotating inertias

CSTIM	Time accumulator of a shift
CTA	TIME SPENT IN ACCELERATION
CTCR	Time spent in cruise
CTD	Time spent in deceleration
CTI	Time spent in idle
CTR	Torque converter energy loss accumulator
CTRDEV	Constant used to define central device
CTRFIL	Constant used to define central file
COL1	Constant used to define column one(1)
COMND	The command being read in a record
CONTOR	CONVERTER CONSTANT TORQUE
CSTIM	Shift time
CTCR	Percent of total spent in criuse
CTD	Time step in deceleration
CTI	Percent of time spent at idle
CTIRE	Tire energy loss accumulator
CTOTAL	Subtotal energy lossses (a percent)
CTR	Torque converter energy loss accumulator

CTRDEV	Control file device
CTRFIL	Name of the control file
CTRLD	Macro subroutine used to debug output to an output device, i.e. tty
CTRPPN	Project, programmer number of control file
CUMD	Cumulative distance traveled in miles
CUMEN	Energy generated by engine from the start to the current time step
CUMENM	Energy generated by engine during coast mode from start to the current time step
CUMFE	Cumulative fuel economy (MPG) (from start to the current time step
CUMFU	weight of fuel consumed(LBs)
CUMG	Gallons of fuel consumed (cumulative value)
CUMT	Cumulative time elapsed in seconds
CUMTL	Simulation time at the end of a shift
CUMTO	Total simulation time
CURVFT	Common block name
CYCLE	The default number of cylinders (4)
CYL	number of cylinders in the engine
D	Distance traveled in feet during current time step
DACC	variable used to alter acceleration of vehicle during iteration

DATA	referenced in inpdia
DATE	Current date
DATEN	New data, data override
DATPPM	Block common name
DBEGIN	Variable in which DEBUG time begins
DBLANK	A blank field
DCKE	Increment of kinetic energy of a car during a time step
DCOM	Drive schedule comment array
DCROT	Decrement of kinetic energy of rotating inertias
DD	Cumulative distance traveled in feet
DDIS	Distance traveled in miles during current time step
DEBUG	Command used to get extra info beyond the run detail printout. It can be very useful for debugging runs that fail
DEBCMD	Same as DEBUG
DEBTAB	Array for debug commands
DEFDT	The default time step to be used
DELETE	This command along with the drpo command allow the user to delete parts from the data base
DELRPM	Incremental rpm value.
DELTOR	Incremental torque value.
DEN	Incremental engine energy produced during time step
DETPT	Detent override point array
DETRPM	Defent override speed array
DFU	Incremental fuel consumed during time step
DGF	Gear from
DGT	Gear to

DHR	Length of time step in hours
DHRD	0.5*DHR
DIALOG	Command to ,place VEHSIM in dialogue mode. This mean that vehsim will accept commands from the terminal raher than a disk control file. Also, logical for dialogue mode.
DIF	Difference between PTORQ and TORQ
DIFO	Same for previous time step
DISK	Storage unit
DISP	Displacement or volume displaced by the pistons in the disk parts data file. Engine displacement
DMPTTY	Logical flag
DNAME	Driving schedule name
DNUMG	Total # of gears
DPPN	Drive schedule project, programmer number
DPROT	Drive schedule protection code.
DRPCMD	Drop command allows the user to drop or delete a part from the data base.
DRPMC	Drop rpm when the clutch is activated
DRPME	Engine rpm difference
DRPMW	Difference of rpm at the wheels
DRVWND	Wind speed used in the drive schedule
DSAVE	Distance travelled during a gear shift step
DSEG	Array of relative distance end points for the driving segments
DSHIFT	Block common name
DSK	Disk
DSKCTR	Subroutine used for opening data base and scratch files
DSKDEL	Subr. used to drop or delete parts from the data base

DSKDIR	Subroutine used to obtain a directory of the parts on the data base.
DSTAR	Set part name wild.
DSTART	Starting absolute distance at the beginning of the driving segment
DSTOP	Variable in which DEBUG time stops.
DT	Length of time step in seconds
DT0	initial time step
DTIRE	Time loss
DTIPEO	same as at previous time step
DTO	REFERENCED IN SIMCTR
DTOT	Passing clearance for the given segment
DUMCOM	The comment of a part when using a directory
DUMMY	Dummy variable
DUDCYC	Initial condition : accessory duty cycle (%), 100% - default
DWMASK	REFERENCED IN INPBAT
DYN	New value for the dynamometer
DYNAM	Name of a block common statement.
EBL	REFERENCED IN INPBAT
ECOM	Engine comment array
ECOME	REFERENCED IN INPBAT
EFFC	=SR*TR =efficiency of converter
EFFST	REFERENCED IN INPBAT
EINER	Engine inertia
EMAP	Engine map containing torques,fuel rates, manifold vacuums, and throttle settings for engines
EMAPO	Array which contains the new values of engine map data

ENAME	Engine name
ENDD	Logical flag set when the end of the data on the control file is encountered.
ENDE	Logical flag for the end of file on a control file
ENDLIM	Logical flag indicating points limited to last time step in each driving segment
ENDRSG	Position of the end point of the drive segment
ENDRTE	Logical flag indicating the end of an entire route
ENDSEG	Logical flag indicating an end of a driving cycle segment
	driving cycle segment
ENDMIN	minimum engine speed at input
ENGINE	Fortran subroutine
ENG1	Logical flag set for engine one (1)
ENG2	Logical flag set for engine two (2)
ENGMAP	Name of a block common statement
ENMIN	Min engine speed as input.
ENTERP	The sum of the interpolated torque points
EPPN	The particular code for a particular engine
EPROT	Protection code for a particular engine
ERAR	Efficiency of the rear axle differential
ERAT	array of gear coefficients
ERPM	Engine map speed points array
ERPMO	Engine map speed point array
EXIT	Command to close all data base files, give the user the option of how to dispose of the print file and exits to the monitor.
EXP.	Intrinsic FORTRAN library functons used for exponentiation. The natural log base, e or 2.728 raised to the value of the argument

FACCEL	Force at the wheels due to acceleration
FAERO	Force at the wheels due to aerodynamic drag
FAEROO	Force at the wheels at previous time step
fbrake	REFERENCED IN INPBAT
FDUM	Force losses at the wheels.
FEINST	INstantaneous value of mpg for the current time step
FF1	Distance traveled by the car during the current time step
FF10	Distance travelled during the previous time step
FGC	REFERENCED IN INPBAT
FGRADE	Force at the wheels due to road grade
FGRADO	Force at the wheels at the previous time step
FILES	REFERENCED IN INPBAT
FILSPC	Subroutine used to get the parameters for an output file.
FIRRPM	First value of rpm
FIRTOR	First value of torque
FLOAT.	Intrinsic FORTRAN library function which returns a real number from an integer argument
FPGAL	Fuel density.
FRATE	Fuel flow rate for current time step
FRATEO	Fuel flow rate for the previous time step
FRATG	Fuel flow in gallons
FRATGO	Fuel flow at the previous time step
FRC1	First constant used with tires.
FRC2	Second constant used with tires.
FROLL	Force at wheels due to rolling resistance
FROLLO	Force at wheels at previous time step

FSPECS	New name of a VEHSIM output file.
FSPGR	Fuel specific gravity
FT	Distance travelled in feet from the start to the current time step
FWHEEL	Total force at wheels
FWHEEO	Same at previous time step
GALHR	Average fuel consumption from start to the current time step (gallons/hours)
GO	Initial condition - gear
GCOM	Comment for a particular gear
GDAT	Logical flag for
GEANAM	Array of gear names
GEANUM	Array of gear numbers
GET	Name of a block common statement
GNAME	Name of a gear
GOBACK	subroutine
GOVLIN	REFERENCED IN INPBAT
GOVPSI	REFERENCED IN SIMCTR
GPPN	Array used for the project and programmer number for the gears
GPROT	Array used for the protection code for the gears.
GRAT	Array of gear ratos
GRPM	Array used for the rpm of a particular gear.
GRTOQ	Array of torque values associated with a gear
GSEG	Array of gear ratios
GTIMAC	Time of drive occupied by given gear
GWR	Variable for a write statement
HACCES	Const for hollerith literal ACCES- accessory

HAREA	Const term for hollerith literal AREA - area
HAXLE	const for holerith literal AXLE
HBATCH	Const used -!- BATCH
HBLANK	-1- BLANK
HBMEP	-1- BMEP
HBUS	-1- BUS
HC1	-1- C1
HC2	-1- C2
HCAR	-1- CAR
HCD	Term used when the vehicle drag coefficient (CD) is being used or modified.
HCOAST	Const used for the hollerith literal COAST
HCOMND	Array used to store the name of the commands
HCONS	REFERENCED IN SIMCTR
HCONT1	REFERENCED IN INPBAT
HCONVE	Const for the hollerith literal CONVE, converter
HCYLIN	-1- CYLIN
HDATA	-1- DATA
HDETEN	-1- DETEN
HDIES	-1- DIESEL
HDIREC	-1- DIREC
HDISPL	-1- DISPL
HDN	REFERENCED IN INPBAT
HDOWNS	-1- DOWNS
HDRIVE	-1- DRIVE
HDRIVI	-1- DRIV1
HDTHRO	Code word designating that throttle position for shifting lines in data sheet 12 (shift logic) is defined in degrees
HENGIN	Word equivalent to "ENGINE RPM"

HFUEL	-1-	FUEL
HGALHR	-1-	GALHR
HGEAR	-1-	GEAR
HHP	-1-	HP
HIDLE	-1-	IDLE
HINEPT	-1-	INEPT
HINITI	-1-	INITI
HINPBA	-1-	
HIST	Array for the histogram points.	
HLBHR	Constant for the Hollerith literal LB/HR	
HLIMT	Variable data array indicating the limit on the amount of output desired by the user.	
HLOAD	-1-	LOAD
HLOCKU	-1-	LOCKUP
HM	-1-	M
HMILE	-1-	MILE
HMILEP	-1-	MILEP
HMOD	Array used when values are MODIFIED	
HNLOAD	-1-	NOT LOADED
HOFF	-1-	OFF
HON	-1-	ON
HOUTPU	Word equivalent to "OUTPUT RPM"	
HP1	horsepower on engine side of torque converteter	
HP2	Horsepower on gearbox side of torque converter	
HPA	horsepower lost to accessory	
HP10	HP1 relating to the previous time step	
HP20	HP2 relating to the previous time step	
HPART	Variable array used for vehsim hollerith part literals, eq, engines,gears, etc.	

HPARTS	-1-	PARTS
HPBC	horsepower lost to brakes	
HPCI	Horsepower at gearbox input	
hpclo	REFERENCED IN INPDIA	
hpcthr	REFERENCED IN INPDIA	
hpdyno	REFERENCED IN INPDIA	
HPE	horsepower produced by engine at the current time step	
HPEO	Horsepower produced by engine at previous time step	
HPISTO	Constant for hollerith engine units PISTON	
HPMI	total horsepower hours/mile for entire driving cycle	
HPP	horsepower between differential and gear box (at propshaft)	
HPPO	Same at the previous time step	
HPS	REFERENCED IN INPBAT	
HPVEHI	"%VEHICLE" name if not included in the vehicle	

HPW	horsepower delivered at the wheel	
HPWO	Same at the previous time step	
HQ	REFERENCED IN INPBAT	
HREAR	-1-	REAL
HROUTE	-1-	ROUTE
HSEC	-1-	SEC
HSEGME	-1-,SEGME, indicating a eqment printout	
HSHIFT	-1-	SHIFT
HSINGL	-1-,SINGLE (spin loss for a single axle)	
HSLASH	-1-	'/'
HSPEED	-1-	SPEED
HSTAR	-1-	STAR
HSTEP	-1-	STEP
HSTROK	-1-	STROK
HSUMMA	-1-	SUMMARY
HTHROT	Throttle (% WOT)	
HTIME	-1-	TIME
HTIRE	-1-	TIRE
HTORQU	-1-	TORQUE
HTPSPD	REFERENCED IN INPBAT	
HTRUCK	-1-	TRUCK
HUN	UPSHIFT	
HUP	DOWNSHIFT	
HUPSHI	-1-	UPSHI
HVACUU	Vacuum (in Hq) Cost for Holerith VACUU	
HVEHIC	Word equivalent to "VEHICLE MPH"	
HWEIGH	-1-	WEIGH
HWHEEL	-1-	WHEEL

IABS. FORTRAN library intrinsic function taking absolute value an integer argument

IB REFERENCED IN INPBAT

IC REFERENCED IN INPBAT

ICMD Integer variable for the command being processed.

ICNTT Counter for the number of parts

ICRCNT Fortran function

ICYCLE REFERENCED IN INPBAT

ICYL Number of cylinders

IDATA REFERENCED IN INPBAT

IDBUGO Counter for debug command

IDEBUG Counter for debug command

IE Printer for the number of drive scheduale record to skip

IECOND Integer variable for a particular error condition

IENG Index indicating which engine being used

IERRE Flag:"failure to find speed/torque setting on the engine map

IERROR Error counter

IDRP REFERENCED IN SIMCTR

IFIX. FORTRAN library intrinsic function whose argument is a real number and makes it an integer number

IFLAG Return status flag (1= no end, 2=hword (alpha word when found on data cards stops input), 3=x, 4=eof)

IG REFERENCED IN INPBAT

IGF =DGF+0.001 Rounded numbers of 'gears to' & 'gear from' belonging to the shift logic sheet (gear from)

IGT =DGT+0.001 ------------1--------(GEAR TO)

IHR REFERENCED IN SIMCTR

IHT	REFERENCED IN SIMCTR
IL	REFERENCED IN INPBAT
ILPT	REFERENCED IN SIMCTR
IMAX	Max engine speed on engine map
IMCNT	REFERENCED IN INPBAT
IMIN	Min engine speed on engine map
IMOD	REFERENCED IN INPBAT
ING	Counter for the number of attempts to find a gear
INPBAT	SUBROUTINE INPUT BATCH
INPCOM	REFERENCED IN INPBAT
IO	REFERENCED IN INPBAT
IOCYL	REFERENCED IN INPBAT
IP	Printer to open a file for information
IPART	REFERENCED IN SIMCTR
IPMODE	REFERENCED IN SIMCTR
IPRNT	Index indicating what component to print or use in disk routine
IR	REFERENCED IN INPDIA
IRMODE	REFERENCED IN INPBAT
IS	Upshift , downshift flag(=1,2)
ISEG	current segment number being executed (n)
ISEG1	Integer variable for the start of a debug segment
ISEG2	Integer variable for the end of a debug segment
ISEGO	REFERENCED IN COMMS
ISHIFT	REFERENCED IN inpdia
ISMODE	REFERENCED IN INPDIA
ISTART	REFERENCED IN INPBAT
IT	REFERENCED IN INPBAT
ITASK	Integer variable for new file specs.
ITERAT	SUBROUTINE

ITERR	REFERENCED IN INPBAT
ITS	The current segment type being executed
ITSAV	Type of a segment (of the drive schedule) being executed
ITTYWD	Logical flag for the printing width
ITYSEG	Array of segment types to driving cycle
IU	REFERENCED IN INPBAT
IUNIT	Integer variable for output print unit
IVAN	REFERENCED IN INPBAT
JCT	Integer const for tty unit
JENG	Array of engine numbers, one assigned for each gear
JOBDEV	Integer constant for job device
JOBNUM	Integer constant for job number
JOBPPN	Ineger constant for the project, programmer number
KEND	LINE FLAG
KPTIME	SUBROUTINE
LBMEP	Logical flag for BMEP units used with engine data
LBRAKE	Logical flag indicating whether or not brake is applied
LBSFC	Logical flag for BSFC units used with engine data
LCLTCH	Flag indicating that clutch is in or out
LCMD	Integer variable used to save command code
LDETE	Logical flag used to chack that detent units match shift line units
LDETNT	Logical flag equivalent to the word "DETENT OVERRIDE"
LDETV	Logical flag used to check that detent units match line unit
LDIES	Logical flag used with DIESEL modification
LDNSHF	Flag of downshift
LDYNA	Only 2 wheels rotate (identifier); logical flag for dynamometer simulation
LDYNOV	Logical flag for override

LEFIL	Logical flag used to identify an end of file
LENG	Logical flag
LFULL	Logical flag to identify a converter as a speed or full converter
LGALHR	-1-GAL/HR units in engine data
LGFREE	-1- FREE gear
LHP	-1- hp units used in engine data
LIDLE	-1- idle condition
LIMPRN	Logical flag indicating whether or not print is desired
LLBHR	-1- lb/hr units in engine data
LLPT	-1- line printer
LLSH	-1- for possible shifts or const gear
LMDLOK	Logical flag indicating wheter to hold a constant gear or???????????????????
LMPH	-1- drive schedule segment specified by velocity
LOCKG	REFERENCED IN INPDIA
LOCKUP	Logical array for the gears being locked up
LOOKUP	Subroutine used to determine if a particular part is on file
LPDP	Logical flag for a part dump
LPRNT	Logical print flag
LPS	Logical flag used for PISTON units in engine data
LPTDEV	Parameter for DEVICE option of OPEN or CLOSE statement
LPTDSP	Disposition of the print file
LPTFIL	REFERENCED IN INPDIA
LPTPPN	Array containing the project and programmer #

LPTPP	Protection for printer fileO
LRPM	Logical flag used for rpm units in engine data (convert piston speed to rpm)
LSAVE	-1- indicate whether or not a print limit is in effect
LSCALE	Logical flag indicating whether engine needs to be scaled
LSHIFT	Logical flag indicating whether or not shift is required
LTHR	Logical, declares that lines of switch in data sheet 12 (shift logic) are defined in degrees
LSIMDM	-1-initialization simulation in DIALOG mode
LSIMUL	-1- simulation in the DIALOG mode
LSKIP	-1- skipping to the next command record or card
LSTRTE	-1- last section of a route
LSTRUP	Logical Start flag for drive schedule
LSTSEG	Logical flag for the Last segment of a drive scedule
LTOR	-1- TORQUE units in engine data
LTRRZ	Flag of:(rear end inertia=0/front rotating inertia=0)
LVAC	Logical flag:vacuum is used as a variable
LVEMAX	-1- identify whether axle is contained in the vehicle
LVNEW	-1-new vehicle format (part being modified)
LWOT	Logical flag indicating whether or not to return only max and min throttle settings from engine routine
MAPOK	Integer variable used to determine in what area of an engine map something is taking place
MAPOLD	USED TO save MAPOK
MASDEV	Master device
MASFIL	master file
MASKS	Name of a common block which contains a mask array
MASPPN	Master project, programmer number

MDLKRP	Engine speed at which a gear is locked up
MFLG	Array used to play a modification part
MGOLD	=NGEAR
MILE	=CUMD
MILIM	Logical flag indicating a limited printout every & many miles
MISC	Name of a block common statement
MIN	Minutes of drive from the start to the current time step
MOCON	Name of a block common statement
MODCMD	The modify command changes the named component to a special value
MODE	Option in an OPEN or CLOSE statement which defines the character set of an external file or record.
MODSL	Subroutine used to modify a component to a specified value
MXCALL	=maximum (allowable) number of calls to ' /' subroutines
MXNGCN	Max # of attempts to find gear
N20	Max possible data points in a table of torque losses for a rear axle/gear
NA	Index of a data points of axle data array
NACC	Number of accessory being used (M)
NARG	Integer flag used to define a particular status for print limits.
NC	Integer variable for the number of characters in a word string.
NCOM	Integer constant for the number of commands
NCOMMD	min # of characters which uniquely defines a command
NCYCLE	Number of cycles of the engine type being used
NDEB	name of a block common statement
NDISK	Variable unit # for the data base or scratch file
NDRP	Percent of engine input dropped
NGEAR	Number of gear being used during current time step

NGEARO	Save the gear #
NGN	Gear number printer
NDRTE	Counter for route section
NDSEG	Initial driving segment
NENG	The number of a particular engine (1 or2)
NGD	Initial gear
NGCNT	Number of step in finding gear
NGEAR	Gear number
NGOCAL	Counts the number of calls to GOBACK
NGOLD	Gear # during previous time step
NGRLSS	Number of data pointing a table of torque losses for a given gear
NGSEG	Array of constant gears used during driving cycle (if any)
NHLIMT	Vector array to uniquely define the min number of characters to define the type of print limit.
NHMOD	Array used to store the min # of characters used to define the modify words
NHPART	Array used to store the min # of characters used to define the name of the parts.
NLSTCF	Logical flag to hint control file on line printer
NMOD	The # of items being modified
NNA	# of point within table of accessory torque losses
NNGS	Constant gear setting for a driving segment
NOPART	Subroutine used to find whether a part exists or not.
NORUN	Counter for fatal errors.
NPART	Array used to hold the names of the parts
NPART8	The 8-th location of the parts array when the dyno is being used.
NPARTS	Array used to determine if a part has been loaded.

NPAX	Number of data points in a table of torque losses in a given rear axle
NPOINT	Number of data points read into array
NPTS	Number of points in a particular array
NRAX	Number of rear axles
NRAX	Total number of tables of rear axle torque loss characteristics
NRDIST	Total number of route segments in the route being used (N)
NREC	Integer variable for the number of records
NRPM	Array containing number of speed points in each engine map
NRPMO	Number of points actually read into an array
NRPMP	Index used for speed points
NRTE	Pointer for the route record
NRTSEG	Current number of the route segment being executed
NSEG	The number of segments in the driving cycle
NSPSEG	Number of shifts during a particular segment
NSGEAR	Upshift and downshift accumulator
NSHIFT	Number of shifts for a particular gear
NSKIP	Number of records to skip
NSPTS	Array containing number of data points in each shift line
NTBP	Index of the data values where the torque converter break point exits
NTC	=NTORP - number of data points read in an array(s) of SR< TR
NTD	=NTORP ------------1-----------
NTOR	Array containing the number of torque points for each speed point on the engine map
NTORP	Number of data points read (into array)

NUMG	Total number of gears being used by the thansmission
NUMBSL	Number of lines read from 'shift logic' data sheet
NUMG	Total number of gears being used by the transmission (ie)
NUMPAR	Counter for the number of part types
NWRCNT	Fortran function
NXTCRD	Next card or record
NXTPG	Next page counter.
OBORE	Save the engine bore
ODISP	Save the engine displacement.
OLDMAP	Save old map
OLDVAL	Array used to hold those values which can be modified.
OSTROK	Save the engine stroke length
OUTCMD	The output command allows the user to specify the name of the output print file
PBMEP	Logical flag
PCKE	Kinetic energy
PCPE	Potential energy
PCROT	Rotating inertia
PCSEG	Array containing passing clearance
PCTWOT	% of wide open throttle for the given value of torque
PCT1	Modified value of upshift
PCT2	Modified value of downshift
PCTHR	Percent engine wide open throttle
PGALHR	Logical flag for gal/hr units used with engine

PHI	Wind angle
PHIO	Save the wind angle
PHP	Logical flag
PPHPHR	Total fuel lbs/Hp-hr used during the entire driving cycle
PNAME	=CNAME
PRINT6	Output unit
PRNDIS	Printing a specified distance in miles
PRNLIM	Name of a block common statement
PRNOUT	Subroutine used to print out engine data
PSI	Angle of wind
PTORQ	Torque corresponding to the current value of APWO (fraction of wide open throttle)
PWOT	Array of constant % wide open throttle required to drive segments
RAR	Rear axle ratio
RARSQ	(RAR)*(RAR)
RAT	Output capacity factor of the torque converter
RCOEF	Array of road coefficients for the route segments
RCOFF	
RCOM	Route comment array
RCRCNT	Function used to get the number of characters in a string
RENTIM	
RDIST	ARRAY OF LENGTHS FOR THE ROUTE SEGMENTS
RDLD	Road load
READPD	SUBROUTINE
REMAP	Command to allow the user to obtain an engine map printed out at the speed & torque points that he specifies rather than at the points as input initially.
RENAME	Command used to rename the output file

RD	Relative distance of a given segments
RDLD	Road load
RGRADE	Array of grades for the route segments
RMPCMD	Remap command loads engine data from parts data file, converts units if necessary, remaps data to specified speed and load points and print out engine data.
RMPD	Distance covered during the given time step (used for calculation of absolute distance travelled)
RNAME	Route name
RNAMEO	Saved route name
ROADC	Current road coefficients used during time step
RPM1	Speed on engine side of torque converter
RPM2	Speed in gearbox side of torque converter
RPM20	Same during previous time step
RPMAX	Max engine speed given by map
RPMC	Speed (rpm) at gear box input (when clutch is activated ? # 607 SIMCTR)
RPME	Engine speed during current time step
RPMEO	Engine speed during previous time step
RPMIN	Min engine speed given by map
RPMP	Propshaft speed
RPMW	Wheel speed during current time step
RPMWO	Wheel speed during previous time step
RPPN	Route ppn
RPROT	Route protection
RPMES	save the engine speed during a time step
RT	Relative time for the given segment
RTE	Name of a block common statement
RTHR	Rate of change for the throttle
RUNID	Common block name which holds thew.title array
RVWIND	The wind speed array associated with the route spec.

SATH Arrival accelaration to accelerate and hold throttle driving segment

SAVE1 Variable used to save the first Hollerith literal a statement.

SAVE2 Variable used to save the second hollerith literal in a statement

SCALEN Subroutine used to rescale engine values.

SCOM Shift logic comment array

SCRDEV Parameter of the device option in the OPEN or close statement indicating the line printer as a sratch device.

SCRFIL Parameter of the FILE option in the OPEN or CLOSE statement which indicates a sramble file

SCRPPN Parameter of the directory option in the open or close statement which indicates the user's project, programmers # or a scratch line printer file

SDEBUG Variable to save the debug command to determine its status

SDELAY Shift time

SEC Seconds of drive from the start to the current time step

SECLIM Logical flag indicating that print is to be limited to every so many seconds

SEQIN A parameter of the access option in the open statement which means that the data base is to be read in a sequential access mode.

SEQUOT A parameter of the required option, ACCESS, in the open statement which means that the specified data file is to be written in a sequential access mode.

SHFTIM Array of shifts times for the shift lines

SHFTL Name of a block common statement

SHFTNG Shifting flag for whether shifting in progress

SHFTPT Array of shifts points for the shift line

SHFTRP Array of shift speeds for the shift line

SHIFTS Subroutine in which vehicle shifting takes place

SIMCMD	The SIMULATE command begins the actual simulation. It first checks to see all the necessary parts have been assembled from the disk parts data file by means of use commands or new part data and allows the vehicle to reach initial conditions specified by the driving scedule.
SIMCTR	Subroutine controls the phases of simulation
SIMSTS	Subroutine used to check the characters entered by the user
SIN.	Intrinsic fortran function to take the sine of an argument
SIMODE	Type or mode of simulation, e.g. CAR
SLIMIT	Save limit status, summary is default
SLOOKP	Routine to do a table lookup
SMILE	Number of shifts per mile
SNAME	Shift logic name
SNGLBS	Logical flag for a single file data base structure
SOUT	Array of output speeds (speed ratios/) for torque converter
SPIDLE	Engine idle speed
SPIN	Array of input speeds for the torque converter
SPPN	Shift logic project, programmer number
SPROT	Shift logic protection
SQRT.	Fortran fnc which takes the square root of a real argument.
SR	Speed ratio for current time step
SRC	Array of speed ratios for the coast converter
SRD	-----------1-------------drive converter
SSTIME	Block common name
STIME	Shift time during the current shift being performed
STROKE	Engine stroke size
STSCMD	The status command prints out the status of the simulation parameters, including the parts that are loaded and all modifications.

T	End elapsed time in seconds of the current
TO	Initial condition : time
	driving segment
TCFAST	
TCOM	Transmission gear comment array
TEMTOT	Temporary total for the energy loss
TEND	Relative end time of current time step if a constant velocity driving segment is being executed
THRATE	Array of max rates of change of throttle settings for driving segment
THRMAX	Max throttle setting on engine map
THRMIN	Min --------------1---------------
TEMTOT	Total energy loss accumulator
TEND	Relative time end segment
TEST	Remaining time to reach the end of segment
TIREFF	=tire efficiency
TITER	Torque corresponding to the current value of APWO (%WOT required for entire segment)
TMINA	Save min engine torque
TOTEN	Total energy
TBL	Variable used for the total # of segments in a drive schedule.
TCOM	Tire comment array
TFRC1	Tire constant C1*1000.
TIN	Array of input torques for torque converter
TIREFF	Tire efficiency
TITFR	Taget torque
TITLE	Run title input by user
TMAP	Name of a block common statement.
TMIN	Min engine torque

TMINO	Same for the previous time step
TNAME	Transmission gear name
TOFAST	Logical flag used during a const acceleration to determine if the vehicle is not going at a constant acceleration.
TOLD	Elapsed time at beginning of current time step
TORBPK	The capacity factor value at the drive converter torque break point
TORQ	Torque at a current time step.
TORQA	Total torque needed by the accessories during current time step
TORQAO	Same at previous time step
TORQE	Required engine torque during current time step
TORQEO	Engine torque during previous time step
TORQF	Front end rotating inertias
TORQP	Propshaft torque during current time step
TORQW	Wheel hub torque during current time step
TORQ1	Torque on engine side of torque converter during current time step
TORQ2	Torque on gear box side of --------1------
TOTEN	
TOUT	Array of output torques (torques ratios) for torque converter
TPPN	Tire project, programmer #
TPROT	Tire protection
TR	Torque ratio used during current time step
TRACMD	Transmission command enables the user to specify an entire transmission by one name rather than by naming all the gear individually.
TRCOM	The comment associated with a transmission.
TRD	Array of torque ratios for the drive converter

TRNAM	Transmission name.
TRPPN	The project, programmer # associated with a transmission
TRR	Rear end rotating inertia torque
TSAVE	Time spent initializing current driving segment
TSEG	Array of relative end point times for the driving segment
TSTART	Relative start time of current time step
TTLCMD	Title command, read title and date.
TTOT	Current passing clearance time (within the given segment/)
TTY	Logical flag for the terminal.
TTYOUT	Block common name.
TWOT	Wide open throttle torque for current engine speed setting
TWOTO	Save wide open throttle setting.
TYPE	Name of command read in input routine
UALCMD	The lock - unlock commands with the capability od 'locking up' or 'unlocking' the torque converter for any of the gears in the transmission.
UN	Component name on input data card
UNITS	Array used to identify the units used with a particular engine.
UNTCMD	The UNIT command allows the user to specify units for the engine data printout resulting from a USE or ENGINE command in units rather than those in which it was originally input on the disks. The available units are the same as those described in the REMAP command.
USECMD	The USE command allows the user to retrieve data from the disk parts data file.
UT	Component type on input data card
V	Vehicle velocity at end of current time step

VO	Initial condition: speed
V2MISC	Block common name.
VAC	Vacuum at a current time step.
VALID2	Subroutine used to determine a valid part name.
VALNEW	Array used to save modified values.
VALUE	The new value for modified parameter.
VALUE2	The second variable value in a real statement.
VARNAME	Name of single variable value to be modified without reading a new component
VAVG	Average vehicle velocity over the entire driving cycle
VEHICL	Vehicle comment array
VELSEG	Array of terminal speed for drive segments
VELTAB	Upshift speed.
VERID	Version identification.
VNAME	The name of the particular vehicle.
VOLD	Absolute velocity at the beginning of the current time step
VPPN	Vehicle project, programmer #
VPROT	Vehicle protection code.
VSEG	Array of velocity end points for the driving segments
VSTART	Relative velocity since beginning of current time step
VTOT	A sum of a vehicle velocity and wind velocity
VWIND	Wind velocity
VWINDC	Wind velocities parallel to the direction of the vehicle (headwind)
VWINDS	Wind velocity component normal to the direction of the vehicle
WGT	Vehicle weight

WLSG	Number of tires in road contact
WMASK	Array used to contain octal values used to get commands.
WORD	Mode of operation (name of the current line) within the current data sheet)
WORD1	The first variable in a statement pertaining to a Hollerith literal.
WORD2	The second variable in a statement pertaining to a Hollerith literal.
WRAD	Wheel radius
X	(TORQUE-TMIN)/(TWOT-TMIN)*100
XCYC	Number of cycles.
XCYL	Number of cylinders.

XHIGH — Last or high values for engine rpm.

XLOW — First or low value for engine rpm.

Y

ZERCMD — Same as ZERO command.

ZERO — This command enables the user to zero the core memory containing all data currently stored by the program and reset all program flags and pointers.

2. SOURCE CODE

```
      SUBROUTINE ASCIZ(STRING,LWRDS,NCHAR)
C
C    THIS SUBROUTINE CONVERTS ALL TRAILING SPACES(BLANKS) CHAR'S TO NULL'S IN STRING.
C
C**********************************************************************
C
C        ENTRY POINTS: ASCIZ
C
C        SUBROUTINES CALLED: OUTSTR, NCRCNT
C
C        CALLED BY: HLFCMD, LOCKUP
C
C**********************************************************************
C
C
      DIMENSION STRING(LWRDS)
C
      INCLUDE 'COMMS/NOLIST'
C
      DO 20 N=LWRDS,1,-1                        !FIND WRD CONTAINING LAST NON-BLANK CHAR.
      IF(STRING(N).NE.HBLANK) GO TO 30          !(GOT IT?) YES.
20    STRING(N)=0.0                             !NO, SET TO NULL.
      NCHAR=0                                   !STRING ALL BLANKS. SET # OF CHAR'S ZERO.
      RETURN                                    !BYE.
C
30    NC=NCRCNT(STRING(N),1)                    !GET #OF CHAR'S IN WRD.
      STRING(N)=STRING(N).AND.WMASK(NC)         !USE MASK TO NULL TRAILING BLANKS IN WRD.
      NCHAR=5*(N-1)+NC                          !CALC#OF CHAR'S IN STRING.
      CALL OUTSTR(STRING(1),LWRDS)              !OUTPUT STRING TO TTY:
      RETURN                                    !BYE.
      END
      SUBROUTINE CONVTR
C
C      ENTRY POINTS: CCNVTR
C
C      CALLED BY: GOBACK
C
C***********************************************************************
C
C
      INCLUDE 'COMMS/KCLIST'
C
      IF(RPM2.GT.1.) GO TO 6
C
C    IF IDLE SET TO LOWEST SPEED RATIO
C
      SR=SRD(1)
      TR=TRD(1)
      GO TO 90
    6 TR=1.
      IF(COAST) GO TO 50
C
C    FOR DRIVE CONVERTER
C
      IF ( TORQ2 .LT. .000001 )  RETURN
C
C    COMPUTE TEST CAPACITY FACTOR PARAMETER
C
      RAT=RPM2/SQRT(TORQ2)
      IF (RAT.LT.TORBRK) GO TO 20
      IF (RAT.LT.AKD(NTD)) GO TO 10
      SR=SRD(NTD)
      RETURN
   10 J=NTD-1
   11 IF (RAT.GT.AKD(J)) GO TO 12
      J=J-1
      GO TO 11
   12 JP=J+1
      SR=(SRD(J)-SRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+SRD(JP)
      IF(SR.GT.1.) SR=1.
      RETURN
   20 IF (RAT.LT.AKD(1)) GO TO 30
      J=NTBP-1
   22 IF (J.LT.1) GO TO 30
      IF (RAT.GT.AKD(J)) GO TO 25
      J=J-1
      GO TO 22
   25 JP=J+1
      SR=(SRD(J)-SRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+SRD(JP)
   26 TR=(TRD(J)-TRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+TRD(JP)
      IF(SR.GT.1.) SR=1.
   27 IF (SR.GE.0.) RETURN
      SR=0.
      RAT=-(AKD(J)-AKD(JP))/(SRD(J)-SRD(JP))*SRD(JP)+AKD(JP)
      GO TO 26
   30 J=1
      GO TO 25
C
C    FOR COAST CONVERTER
C
   50 IF(RPM2.LT.SRC(1)*AKC(1)) GO TO 55
      IF(RPM2.GT.SRC(NTC)*AKC(NTC)) GO TO 60
```

```
C
C     IF BELOW LOWEST SR GIVEN AS INPUT
C
   55   J=1
        JP=2
        GO TO 75
C     IF ABOVE HIGHEST SR GIVEN AS INPUT
C
   60   JP=NTC
        J=JP-1
        GO TO 75
C
C     FIND CORRECT SEGMENT FOR CURRENT POINT
C
   65   DO 70 J=2,NTC
        IF(RPM2.GE.SRC(J-1)*AKC(J-1).AND.RPM2.LE.SRC(J)*AKC(J)) GO TO 73
   70   CONTINUE
   73   JP=J
        J=JP-1
C
C     COMPUTE SPEED RATIO BY INTERPOLATION
C
   75   SR=(SRC(J)-SRC(JP))/(AKC(J)*SRC(J)-AKC(JP)*SRC(JP))*(RPM2-
       1AKC(J)*SRC(J))+SRC(J)
        IF(SR.LT.1.) SR=1.
        RETURN
C
C     IF SR GREATER THAN MAX INPUT , SET TO MAX
C
   90   IF(SR.GT.SRD(NTD)) SR=SRD(NTD)
        RETURN
        END
```

```
      SUBROUTINE DEBUG(HIDNAM)
C
C     ENTRY POINTS: CTRLD, DEBUG
C
C     CALLED BY: GOBACK, SHIFTS, SIMCTR, SIMLPT, SIMSTS
C
C*****************************************************************
C
C
      INCLUDE 'COMMS/NOLIST'
C
      ICALL=1                                                     !FLG NORMAL SUBROUTINE ENTRY.
      GO TO(900,400,300,400),IDEBUG                               !DEBUG(OFF/SHIFTS/TIME/SEGMENT?).
      RETURN
C
 300  IF ( CUMT.LT.DBEGIN )  GO TO 900                            !(CUMULATIVE SIMTIME < BEGIN OF DEBUG TIME?)YES.
      IF ( CUMT.GE.DSTOP  )  GO TO 700                            !(CUMULATIVE SIM TIME >= END OF DEBUG TIME?)YES.
      GO TO 800                                                   !NO.
C
 400  ISEGA=ISEG+NDSEG                                            !CAL DRS SEG.
      IF ( ISEGA.LT.ISEG1  )  GO TO 900                           !(DRS SEG < BEGIN DEBUG SEG?)YES.
      IF ( ISEGA.GE.ISEG2  )  GO TO 700                           !(DRS SEG >= END DEBUG SEG?)YES.
      GO TO 805
C
 700  IF ( ISEG2.NE.0     )  GO TO 900
      GO TO 800
C
      ENTRY CTRLD(HIDNAM)
      ICALL=0                                                     !FLG ENTRY VIA CONTRL D.
C
 800  ISEGA=ISEG+NDSEG                                            !CALC DRS SEG #.
 805  PCTWOT = 100. * (TORQE-TMIN) / (TWOT-TMIN)                  !CALC % WOT.
      NRTEA=NRTSEG+NDRTE                                          !CALC RTE SEG.
      IF(.NOT.SHFTNG) GO TO 820                                   !(SHIFTING IN PROGRESS?)NO.
      HSMODE=HUP                                                  !YES, ASSUME UPSHIFT.
      IF(LDNSHF) HSMODE=HDN                                       !(DOWNSHIFT)YES
      HTMODE=HON                                                  !ASSUME GEAR UNLOCKED.
      IF(ICLTCH) HTMODE=HBLANK                                    !(GEAR LOCKED?)YES.
      DRPM1E=RPM1-RPME                                            !CALC NEEDED CHG IN RPME TO COMPLETE SHIFT.
 820  IF(ICALL.EQ.0) GO TO 850                                    !(OUTPUT TO JCT ONLY?)YES.
      IF(.NOT.LLPT)GO TO 850                                      !(LPT OFF)YES.
      LPT=IUNIT                                                   !SEND OUTPUT TO LINE PRINTER.
 840  IF(IDBUGO.EQ.1)WRITE(LPT,1700)
      WRITE (LPT,1600)                                            !DEBUG OUTPUT.
     1  CUMT,CUMD,V,ACCEL, FWHEEL,FROLL,FAERO,FACCEL,FGRADE,
     2 CT,NGEAR,ISEGA,NRTEA,TORQW,TRR,TORQP,TORQ2,TORQ1,TORQF,TORQE,
     3  PCTWOT,HIDNAM,ADR, REMW,DREMW,RPMP,RPM2,RPME,DRPME,
     4        NGOCAL,MAFOK,VAC
C
      IF(SHFTNG) WRITE(LPT,1900) HSMODE,HTMODE,CUMTLS,CSTIM,TORQA    !(SHIFTING?)YES, OUTPUT SHIFT DATA.
     1,TORQAC,DTORQF,ARPME,DRPM1E,RPMC,DRPMC,DTC,LCLTCH
      IF(LIMTTY.OR.LPT.EQ.JCT) GO TO 900                          !(JCT OFF OR JCT OUTPUT DONE?)YES.
C
 850  IF(IDBUGO.EQ.1)WRITE(JCT,1700)
      IF(ITTYWD.LT.120) GO TO 855                                 !(JCT CARRIAGE WIDTH OK FOR LPT FORMAT?)NO.
      LPT=JCT                                                     !SET UNIT TO JCT.
      GO TO 840                                                   !GO OUTPUT LPT FORMAT.
C
 855  WRITE (JCT,1950)                                            !OUTPUT TO JCT.
     1  CUMT,CUMD,V,ACCEL, FWHEEL,FROLL,FAERO,FACCEL,FGRADE,
     2 CT,NGEAR,ISEGA,NRTEA,TORQW,TRR,TORQP,TORQ2,TORQ1,TORQF,TORQE,
      3  FCTWCT,HIDNAM,ADR, RPMW,DRPMW,RPMP,RPM2,REME,DRPME,
      4       NGOCAL,MAFOK,VAC
C
      IF(SHFTNG) WRITE(JCT,1950) HSMCDE,HTMODE,CUMTLS,CSTIM,TORQA      !(SHIFTING?)YES, OUTPUT SHIFT DATA.
      1,TCROAO,ATORQF,ARPME,DEPR1E,RPMC,DRPMC,DTC,LCLTCH
C
  900 IDRUGO=IDEBUG
      RETURN                                                           !BYE.
C
C*********************************************************************
C
C    FORMAT STATEMENTS
C
1700   FORMAT(/
      1' CUMT,CUMD,V,ACCEL, FWHEEL,FROLL,FAERO,FACCEL,FGRADE',/,
      2' DT,NGEAR,ISEGA,NRTEA,TCROW,TRR,TORCP,TORQ2,TORQ1,TORQP,TORQE',/,
      3'  PCTWOT,HIDNAM,ABR, RPMW,DFEMW,REMP,RPM2,RPME,DRPME',/,
      4'      NGOCAL,MAFOK,VAC',/)
 1800 FORMAT (/' @'F9.2,F9.3,1X,F7.3,F10.3' (F)'5(1XG10.4)/
      1,'  DT'F6.3',GEAR'I3',ERS'I4',RTE'I4'   (T)'7(1XG10.4)/
      2,'  PWOT'F8.3',ID:'A5',BRK='F7.1' (R)'6(1XG10.4),/
      3,'  GOEACK',I5,5X,'MAPCK:',I2,5X,' VAC ',F6.1)

 1850 FORMAT(/' @'F9.2,F8.3,2(1XG7.3)' (F)'4(1XG9.4)/
      1,37XG10.4/
      1,'  DT'F6.3',GEAR'I3',ERS'I4',RTE'I4'   (T)'4(1XG9.4)
      1,/35X3(1XG10.4)/
      2,'  PWCT'F8.3',ID:'A5',BRK='F7.1' (R)'4(1XG9.4)
      3,/35X2(1XG10.4)/
      3,'  GOEACK',I5,5X,'MAPCK:',I2,5X,' VAC ',F6.1)
 1900 FORMAT(2XA2'SHIFT GR 'A2'LOCKED'F9.2,1XF4.2' (S)'7(1XG10.4))
 1950 FORMAT(2XA2'SHIFT GR 'A2'LOCKED'F9.2,1XF4.2' (S)'4(1XG10.4)/
      1,3(1XG10.4))
C
      END
      SUBROUTINE DSK
C
C     ENTRY POINTS: DSK
C
C     SUBROUTINES CALLED: DSKCTR, DSKRD, DSKWR, PRNOUT
C
C     CALLED BY: INPPAT, PRNTPD, SIMCTR
C
C*******************************************************************
C
C
      INCLUDE 'COMMS/BCLIST'
C
      LOGICAL NXTSEC
C
      DOUBLE PRECISION ADUM
C
      DIMENSION HPART(14),BPPN2(2)
C
      EQUIVALENCE (PPER,BPPN2(1))
C
C
C
      DATA ( HPART(I),I=1,NPART) /'ENGIN','CONVE','VEHIC','GEAR'
     2,'ACCES','DRIVI','SHIFT','ROUTE','TIRE','TRANS','AXLE'/
     3,LFT/6/,JCT/5/,HDRS/'DRS'/,HRTE/'RTE'/
C
C*********************************************************************
C
      NDISK=3                                  ! RESET IN CASE ¬C , .REENTER WAS DONE
      JPRNT  = 0                               ! ZERO JPRNT
      IF ( IPRNT.LE.100 )  GO TO 230           ! (/USE/?) NO.
      JPRNT  = IPRNT                           ! YES, SAVE TASK CODE.
      IPRNT  = IPRNT - 100                     ! SET FILE POINTER
      NXTSEC=.FALSE.                           !ASSUME NOT MULTI SECT.
      IF(IPRNT.LE.100) GO TO 230               !(MULTI SEC PART ?) NO.
      IPRNT=IPRNT-100                          !YES, RESET FILE POINTER.
      NXTSEC=.TRUE.                            ! SET NEXT SECTION FLAG.
      IF(IPRNT.NE.6) GO TO 180                 !(DRS?) NO, MUST BE ROUTE(RTE) REQUEST.
      NSECG=RSEC+1                             ! CALC DRS SECT # TO BE GOTTEN
      WRITE(JCT,1411) HDRS,NSECG               ! [DSKDRS/NSEC:I3$
      PNAME=DNAME                              !SET PART NAME TO LOOK UP.
      GO TO 230                                ! GO GET NEXT DRS SECTION
C
  180 NRTEG=NRTE+1                             ! CALC RTE SECT # TO BE GOTTEN
      WRITE(JCT,1411) HRTE,NRTEG               ! [DSKRTE/NSEC:I3$
      PNAME=RNAME                              !SET PART NAME TO LOOK UP.
      GO TO 230                                !GO GET NXT RTE SECT
C
C    IF NO PARTS DATA , GO TO STORE FIRST PART DATA
C
  230 CALL CHKFIL(BASDEV(IPRNT),BASFIL(IPRNT),BASPPN(1,IPRNT)
     1,$233)                                                        !SEE IF FILE IS ACCESSABLE
      GO TO 245                                !ERROR RETURN
C
C    SCAN PARTS DATA FILE FOR DUPLICATE PART NAME
C
233   CALL DSKCTR(IPRNT,SEQIN)                 ! NO, REOPEN FILE FOR INPUT ONLY
  235 READ  (NDISK ,END=240 )  PN,PT,NREC
      IF(PN.EQ.PNAME.AND.PT.EQ.HPART(IPRNT)) GO TO 250
      GO TO 235
```

```
C
C       ERROR SECTICN IF /USE/ COMMAND , PART NCT ON PARTS DATA FILE
C
  240   IF ( JPRNT .EC. 0 )  GO TC 300           ! (NOT/USE/?) YES, GO STORE NEW PART.
        IP=IPRNT
        IF(SNGLES) IP=12
        IF(TTY.AND.BATCH) WRITE(JCT,1240) HPART(IPRNT),PNAME
        1.EASDEV(IP),EASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        WRITE(LET,1240) HPART(IPRNT),PNAME
        1,BASDEV(IP),EASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
  243   IPRNT  = - 10
        GO TO 900
C
C  EMPTY OR NON-EXSISTENT DATA EASE FILE
C
  245   IP=IPRNT
        IF(SNGLES) IP=12
        IF(JPRNT.EO.0) GO TO 247                 ! ( PART TO STORE?) YES.
        WRITE(JCT,2450) EASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP) ! NO.
        CALL DSKCTR(NDISK,DELETE)                          ! DELETE (0) FILE CREATE BY LOOKUP
        GO TO 243                                ! WRITE ERR MESS AND GO SET ERR FLAG
  247   WRITE(JCT,2470) BASDEV(IP),EASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        1,HEART(IERNT),PNAME                     ! % CREATED FILE MESS
        GO TO 300                                ! GO STORE PART NOW!!
C
C       IF /USE/ COMMANC , GO TO LOAD PART DATA FOR SIMULATION
C
  250   IF ( JPRNT.NE.0 )  GO TC 500             ! (/USE/?) YES.
C
C    CHECK FCR MULTIPLE SECTION DRIVING SCHEDULE (50 SEGS/SECTION)
C        OR ROUTE(10 SEGS/SECTION)
C
        IF((IPRNT.EC.6.ANC.NSEG.LT.0).OR.(IPRNT.EQ.8.AND.NRDIST.LT.0))
        1 GC TO 320                                                            ! NO,(MULTI SECT?) YES.
C
C       WRITE ERROR MESSAGE AND DC NOT STORE IF DUPLICATE PART DATA
C
  255   IP=IPRNT                                 ! SET PTR TO FILE INFO
        IF(SNGLBS) IP=12
        IF(TTY) WRITE (JCT,1250)  BT,BN
        1,EASDEV(IP),EASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        WRITE (LPT,1250)  BT,BN
        1,EASDEV(IP),EASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        GO TO 900
C
C*********************************************************************
C
C       STORE PART CATA AT END OF PARTS CATA FILE
C
  300   NREC=IPRNT
        IF(IPRNT.NE.6 .CR. NSEG.GT.0) GO TO 302
        NSEG=IABS(NSEG)                          ! HERE ON 1ST SEC OF MULTI SEC DRS ONLY.
        GO TO 305
  302   IF(IPRNT.NE.8 .CR. NRDIST.GT.0) GO TO 310
        NRDIST=IAES(NRDIST)
  305   NREC=-IERNT
  310   DT=HPART(IPRNT)
        BN=PNAME
        CDATE=DATE
C
        CALL DSKCTR(IPRNT,APPENC)                ! OPEN DB FILE
```

```
C
      CALL TIME(CHOUR)                        !GET CREATION TIME OF PART ON DB
C
      CALL DSKWF                              ! STORE PART
C
      GO TO 900                               ! DONE !
C
C   SPECIALE HANDLING TO STORE MULTIPLE SECTION DRIVING SCHEDULE
C     OR ROUTE WHEN STORING OTHER THEN FIRST SECTION.
C
 320  READ(NDISK,END=305) BN,ET,NREC
      IF(BN.EQ.FNAME.AND.ET.EQ.HPART(IPRNT).AND.NREC.LT.0) GO TO 320
 325  WRITE(JCT,1409) JERNT,IERNT              ! IMPOSSIBLE ERR OCCURED IN DB FILE NOW OPN
      WRITE(LPT,1409) JERNT,IERNT
      CALL TRACE                              ! SOME HELP.
      IPRNT=-10                               ! SET ERR FLG
      GO TO 255                               ! GO GIVE DB FIL INFO
C
C*********************************************************************
C
C     LOAD PART DATA CALLED FOR BY /USE/ COMMAND
C
 500  IF(IPRNT.NE.6) GO TO 502           ! (LOAD DRS?) NO.
      IF(NREC.LT.0) GO TO 510            !YES, (MULTI SECT DRS?) YES.
      NDSEG=0                            ! ZERO SEG COUNT OFFSET
      GO TO 530                          ! GO GET DRS
C
 502  IF(IPRNT.NE.8) GO TO 530           !(LOAD RTE?) NO.
      IF(NREC.LT.0) GO TO 510            !YES,(MULTI SECT RTE?) YES.
      NDRTE=0                            !ZERO RTE CNT OFFSET
      GO TO 530                          ! GO GET RTE
C
 510  IF(NXTSEC) GO TO 515               ! (GET NEXT SECT?) YES.
      IF(IPRNT.NE.6) GO TO 512           ! (DRS?) NO.
      NDSEG=0                            !YES, SET TO GET 1ST SECT OF MULTI SECT DRS
      NSEC=1                             !POINT TO 2ND RECORD OF DRS
      GO TO 514
 512  IF(IPRNT.NE.8) GO TO 325           ! (RTE?) NO, IMPOSS ERR, GO BYE
      NDRTE=0                            ! YES, SET TO GET 1ST SECT OF RTE
      NRTE=1                             ! FOR NXT TIME PNT TO 2ND REC OF RTE
 514  NREC=IABS(NREC)                    ! SET FLAG SO 1ST SECT READ STATEMENT USED
      GO TO 530                          ! GO GET 1ST SECT
C
 515  IF(IPRNT.NE.6) GO TO 517           ! (DRS?) NO.
      NDSEG=NSEG                         !YES, SET DRS SEG OFFSET
      ADUM=DNAME                         !SAVE
      IE=NSEC-1                          !CALC DRS RECS TO SKIP
      NSEC=NSEC+1                        !INC DRS REC PTR
      GO TO 518
C
 517  IF(IPRNT.NE.8) GO TO 325           !(RTE?) NO. IMPOSS GO ERR BYE
      NDRTE=NRDIST                       !YES SET RTE SEG OFFSET
      ADUM=RNAME                         !SAVE
      IE=NRTE-1                          !CALC RTE RECS TO SKIP
      NRTE=NRTE+1                        !INC RTE REC PTR
C
 518  IF(IE.EQ.0) GO TO 520              !(FILE POSI?) YES, SKIP
      DO 519 I=1,IE                      !NO POSI FILE
 519  SKIP RECORD NDISK
C
 520  CALL DSKRD                         !READ SECT
```

```
      IF(ADUM.NE.PNAME) GO TO 125        !(ERR?) YES, GO ERR BYE
      IF(IPRNT.NE.6) GC TO 522           ! NO, (DRS?) NO.
      WRITE(JCT,1413) NSEG               !YES, "/NSEC:I3"
      NSEG=NDSEG+NSEG                    !SET DRS SEG LIM
      IF(.NOT.LSTSEC) GC TO 524          !(LAST DRS SECT?) NO.
      NSEC=0                             !ZERO DRS REC PTR
      GO TO 523
C
  522 IF(IPRNT.NE.8) GO TO 325           !(RTE?) NO, GO ERR BYE
      WRITE(JCT,1413) NRDIST             !YES,"/NSEC:I3"
      NRCIST=NCITE+NRCIST                !SET RTE SEG CNT LIM
      IF(.NOT.LSTRTE) GC TO 524          !(LAST RTE SECT?) NO.
      NRTE=0                             !YES, ZERO RTE REC PTR
  523 WRITE(JCT,1414)                    !"/LAST SECTION"
  524 WRITE(JCT,1415)                    !"]"
      GO TO 900                          !*DCNE
C
  530 BACKSPACE NDISK                    ! PCSI DB FILE
C
      CALL DSKRD                         ! LOAD PART CATA
      CALL PRNOUT                        ! PRINT IT
      GO TO 900                          ! *DONE
C
C*************************************************************************
C
  900 RETURN                             !*DCNE, BYE!!!
C
C*************************************************************************
C
C     FORMAT STATEMENTS
C
 1240 FORMAT(/' ? DSK-PART 'A5,1XA10
     1,' NCT CN PARTS CATA FILE 'A6':'A10'['O5','O3' ]')
 1250 FORMAT (' ? DSK- PART 'A5,1XA10' NOT STORED'
     1,' ALREADY CN PARTS DATA FILE 'A6':'A10'['O5','O3' ]')
 1403 FORMAT(/' ? CSK- IMPOSSIBLE FILE POSITIONING ERROR.'
     1,' (JPRNT='I3'/IPRNT='I3')')
 1411 FCRMAT(' [DSK-'A3'/NSEC:'I3$)
 1413 FORMAT('+/NSEG:'I4$)
 1414 FORMAT('+/LAST SECTION'$)
 1415 FCRMAT('+]')
 2450 FORMAT(/' ? DSK- EMPTY PARTS CATA FILE 'A6':'A10'['O5','O3' ]')
 2470 FORMAT(/' % DSK- CREATEC FILE 'A6':'A10'['O5','O3' ]'
     1,/5X'TC STCRE PART 'A5,3XA10)
C
      END
      SUBROUTINE DSKCTR(IARG1,DARG2)
C
C       ENTRY POINTS: DSKCTR
C
C       CALLED BY: DSK, DSKDEL, DSKDIR, INPEAT, SIMCTR, VSMCTR
C
C************************************************************************
C
C
        INCLUDE 'COMMS/NCLIST'
C
        DOUBLE PRECISION DARG2,ACCESS,OLDACC
C
        DIMENSION USEPPN(3)
        REAL ITMPPN(2)
C
        REAL LPTPPN
C
        EQUIVALENCE (ITMPPN(1),JOBPPN)
C
        DATA ISCR/0/,ICPEB/0/,USEPPN(3)/0/
C
C**************************************************************************
C
C   DATA BASE DISK FILE CONTROL FOR SUBROUTINE DSK
C
        IPRNTT=IARG1                                                        !ISOLATE ARG LIST.
        ACCESS=DARG2
C
        IF(ACCESS.NE.DELETE) GO TO 505                                      !(DELETE OPEN FILE?) NO.
        CLOSE (UNIT=IPRNTT,DISPOSE=DELETE)                                  !YES.
        RETURN                                                              !DONE.
C
  505   IF(ISCR.NE.0) GO TO 520                                             !(SCRATCH FIL OPN?) YES, CLOSE IT.
        IF(SNGLES) GO TO 570                                                !NO, (SINGLE FILE DB'S STRUCTURE?) YES.
        IF(IPRNTT) 510,530,550                                              !NO, (OPN SCR?/CLOSE ALL?/OPN DB FIL?).
C
  510   ISCR=IABS(IPRNTT)                                                   !ABSOLUTE VALUE.
        IF(ISCR.EQ.21) GO TO 513                                            !(LPT SCRATCH FILE FOR DSK DIR?) YES.
        IF(ITMPPN(1).NE.BASPPN(1,ISCR).OR.ITMPPN(2).NE.BASPPN(2,ISCR))      !NO, CALL FROM DSKDEL (CHECK ACCESS?).
       1  GO TO 515                                                         !USER FILE PROT FAILURE.  GO REPORT.
        USEPPN(1)=BASPPN(1,ISCR)                                            !SET DIRECTORY PATH.
        USEPPN(2)=BASPPN(2,ISCR)
        OPEN(UNIT=2,DEVICE=BASDEV(ISCR),ACCESS=ACCESS,MODE=BINARY           !OPEN DSKDEL SCR FIL.
       1,FILE=SCRFIL,DIRECTORY=USEPPN)
        IPRNTT=ISCR                                                         !SET PTR TO ORG DB FIL NAME.
        ACCESS=SEQIN                                                        !SET ACCESS FOR DBG DB FIL.
        GO TO 550                                                           !GO OPEN DB FIL.
C
  513   OPEN(UNIT=2,DEVICE=SCRDEV,ACCESS=ACCESS                             !OPEN SCR LPT FIL.
       1,MODE='ASCII',FILE=SCRFIL,DIRECTORY=SCRPPN)
        RETURN                                                              !DONE.
C
  515   WRITE(6,1245)                                                       !?FILE PROT FAILURE.
        IARG1=-12                                                           !SET ERR FLG.
        IF(ITTY) WRITE(5,1245)
        RETURN                                                              !BYE.
C
  520   IF(ISCR.EQ.21) GO TO 525                                            !(OPEN SCR IS AN LPT SCR?) YES, GO DELETE IT.
        CLOSE(UNIT=3,DISPOSE=DELETE)                                        !DELETE ORG DB FIL.
        CLOSE(UNIT=2,DISPOSE=RENAME,FILE=BASFIL(ISCR),PROTECTION="022)      !RENAME SCR FIL TO DB FIL.
      IOPEN=0                                                  !SET FLG NO OPEN DB FILE.
  523 ISCR=0                                                   !SET FLG NO OPEN SCR FILE.
  524 IF(SNGLBS) GO TO 570                                     !(SINGLE FILE DB?) YES, GO SPECIAL HANDLING.
      IF(IPRNTT) 510,530,555                                   !(OPEN SCR?/CLOSE ALL?/OPEN DB FIL?).
C
  525 IF(IPRNTT.EQ.-21) GO TO 527                              !(FILE TO OPEN ANOTHER LPT SCR?) YES.


      IF(IPRNTT.GT.0) GO TO 524                                !NO, (ANY SCR FIL BEING REQUESTED?) NO.
      CLOSE(UNIT=2,DISPOSE=DELETE)                             !YES, DELETE CURRENT SCR FILE.
      GO TO 523                                                !GO FLG NO SCR OPEN.
C
  527 CALL RELEAS(2)                                           !SAVE SCR FILE.
      GO TO 513                                                !GO OPEN SCR (IF DELETE OF OLD SCR OPEN WILL DO IT).
C
  530 IF(IOPEN.EQ.0) RETURN                                    !(DB FIL OPN?) NO, DONE.
      CALL RELEAS(3)                                           !RELEASE IT.
      IOPEN=0                                                  !SET FLG NO OPN DB FIL.
      RETURN                                                   !DONE.
C
  550 IF(IPRNTT.NE.IOPEN .OR. ACCESS.NE.OLDACC) GO TO 555      !(DB FIL ALL READY OPEN AS NEEDED?) NO, GO OPEN.
  552 REWIND 3                                                 !YES, JUST REWIND IT.
      RETURN                                                   !DONE.
  555 USEPPN(1)=BASPPN(1,IPRNTT)                               !GET DIRECTORY PATH.
      USEPPN(2)=BASPPN(2,IPRNTT)
      OPEN(UNIT=3,DEVICE=BASDEV(IPRNTT),ACCESS=ACCESS,MODE=BINARY  !OPEN DB FILE.
     1,FILE=BASFIL(IPRNTT),DIRECTORY=USEPPN)
      OLDACC=ACCESS                                            !SAVE ACCESS OF HOW DB OPENED.
      IOPEN=IPRNTT                                             !SAVE PTR TO OPEN DB FILE.
      RETURN                                                   !DONE.
C
C   SPECIAL HANDLING FOR SINGLE FILE DATA BASE
C
  570 IF(IPRNTT) 575,530,585                                   !(OPN SCR?/CLOSE ALL?/OPN DB FIL?).
  575 IF(IPRNTT.NE.-21) IPRNTT=-12                             !(LPT SCR?) NO, SET PTR TO SNGL FIL DB NAME.
      GO TO 510                                                !GO SCR FIL REQUEST.
  585 IF(IOPEN.EQ.12) GO TO 552                                !(DB FIL OPEN?) YES, GO REWIND.
      IPRNTT=12                                                !SET PTR TO SNGL FIL DB NAME.
      GO TO 555                                                !GO OPEN.
C
C*****************************************************************************
C
C   FORMAT STATEMENTS
C
 1245 FORMAT(/' ? DSKCTR-FILE PROTECTION FAILURE'//
     1,5X'DATA BASE PARTS MAYBE DROPPED ONLY WHEN JOB PPN'
     2,' IS DATA BASE PPN.')
      END
      SUPFOUTINE ESKCEL
C
C     ENTFY PCINTS: ESKCEL
C
C     CALIED FY: INPEAT
C
C*************************************************************************
C
C
      INCIUDE 'COMMS/NCIIST'
C
      LOGICAL LTEST
C
      DIMENSICN HPART(14),BPPN2(2)
C
      EQUIVALENCE (BPEN,BPPN2(1))
C
C
      DATA  (EPART(I),I=1,NPART) /'ENGIN','CONVE','VEHIC','GEAR'
      2,'ACCES','CFIVI','SHIPT','ROUTE','TIRE','TRANS','AXLE'/
      2,LPT/6/,JCT/5/,ISCR/2/
C
C*************************************************************************
C
C    /DROP/ OR /EEIETE/ COMMANE - DFCP PART FROM PABTS DATA FILE
C
      NDISK=3                                           ! RESET IN CASE ¬C , .REENTER WAS DONE
      GO 10 620
C
 615  WRITE (JCT,1240)  UT,UN                           !ERROR PRINTOUT , PART NOT ON PARTS DATA FILE
      1,EASDEV(IP),EASFIL(IP),BASPPN(1,IP),EASPPN(2,IP)
 617  IEFNT      = - 10                                 ! SET ERROR FLAG
      GO 10 900                                         ! GO DYE
C
C    COPY FARTS EATA FROM UNIT 3 TO UNIT 2 SKIPPING OVER PART TO DROP
C
 620  IPRATD=-IPRNT                                     ! SET PTR TO FILE TO OPEN & CALL FOR SCR FILE
      CALI DSKCTR(IPRNTE,SEQOET)                        ! OPEN DB & SCR FILE
      IF(IPRNTD.EO.-12) GO TO 617                       !  (FILE PROT FAILURE?) YES, GO EFR BYE
      NSECF=0                                           ! NO, INIT TC DROP
      IP=IPRNT                                          !SET FILE INFO PTR.
      IF(SNGLES) IP=12                                  !(SINGLE FILE DB?)YES, RESET PTR.
      IF(IPRNT.EO.1) IENG=1                             !(PROCESSING ENGINE?)SET PTR TO LOC TO USE.
      IF(IPRNT.EO.4) NGEAR=1                            !(PROCESSING GEAR?)SET PTR TO LOC TO USE.
      IF(IPRNT.EO.5) NACC=1                             !(PFOCESSING ACCESSORY?)SET PTR TO LOC TO USE.
      NDEI=0                                            !ZERO CNT OF PARTS DELETED.
      ASSIGN 625 TO REAEST                              !LABEL TO BRANCH TO TELL PART TO DELETE FOUND.
      LEFIL=.FALSE.                                     !SET EOF FLG.
      WRITE(JCT,1615)                                   ! DELETE HEADER TO JCT.
      WRITE(LPT,1615)
C
 625  REAE(NDISK,END=645) BN,ET,NREC,LTEST
      IPRNT=IABS(NREC)
      IF ( EN.NE.UN .CR. ET.NE.ET )   GO TO 630
      IF(NFEC.LT.0) GC TO 627                           !(MULTI SECT?) YES, FIND LAST SECT.
 626  NDEL=NDEL+1                                       !NO, GOT ENTIRE PART
      WRITE(JCT,1620)
      1BASDEV(IF),EASFIL(IP),EASPPN(1,IP),BASPPN(2,IP),BT,BN
      WRITE (LPT,1620)
      1BASDEV(IF),EASFIL(IP),EASPPN(1,IP),BASPPN(2,IP),BT,BN
      ASSIGN 629 TO FEAEST                              !FOUND PART SWITCH TO LOGIC TO GET REST OF FILE.
```

```
C
  627   NSECF=NSECF+1                                                 ! INC # OF SECTS
        IF(NSECF.EO.1) GC TC 625                                      !(1ST SECT?) YES, GO LOOK NEXT REC.
        IF(IPRNT.NE.6.ANC.IPRNT.NE.9) GO TO 617                       !(DRS OR RTE?) NO, ERR.
        IF(.NOT.LTEST) GC TO 625                                      !YES, (GOT LAST SECT?) NO, GO LOOK NEXT REC.
        NSECF=0                                                       ! YES, ZERO RECORD COUNT
        GO TO 626                                                     ! GOT LAST SECT, GO REPORT DROP
C
  629   READ(NCISK,END=645) DN,ET,NREC,LTEST                          !LOOK NEXT PART.
        IPRNT=IABS(NREC)                                              !SET NEXT PART TYPE PTR.
C
  630   BACKSPACE NCISK                                               !POSI DE FILE TO LOAD PART.
        IF(NSECF.EO.0) NREC=IPRNT                                     !(1ST SECT?) YES, FLG PROPER READ ST.
C
        CALL DSKRD                                                    !LOAD PART.
C
        IF(LEFIL) GC TO 645                                           !(EOF?)YES.
        IF(NREC.GT.0) GC TO 640                                       !(MULTI SECT?)YES.
        NSECF=NSECF+1                                                 !YES, INC PTR.
        IF(NSECF.EO.1) GO TO 640                                      !(1ST SECT?)YES.
        IF(NREC.EC.-6) NSEG=-NSEG                                     !(DRS?)YES, FLG DSKWR.
        IF(NREC.EO.-8) NRDIST=-NRDIST                                 !(RTE?)YES, FLG DSKWR.
C
  640   NDISK=2                                                       !SET PTR TO SCR FILE.
        CALL DSKWR                                                    !WRITE PART TO SCR FILE.
        NDISK=3                                                       !RESET PRT TO DB FILE.
C
        IF((IPRNT.EC.6.ANC.LSTSEC).OR.(IPRNT.EO.8.AND.LSTRTE)) NSECF=0 !(LAST SECT OF MULTI SECT PSRT?)YES,ZERO REC CNT.
        GO TO READST                                                  !NEXT PART.
C
C     RESET ALL PARTS ACCCUNTING ( ALL PARTS DATA OVERWRITTEN )
C
  645   IALL=0                                                        !ASSUME ONLY ONE PART TYPE OVERWRITTEN.
        IF(.NOT.SNGLES) GO TO 650                                     !(MULTIFILE DB?)YES.
        IALL=1                                                        !NO, FLG ALL PARTS IN CORE OVERWRITTEN.
        DO 690 IPRNT=1,NUMPAR                                         !LOOP THRU ALL PART TYPES.
  650   IF(IPRNT.NE.1) GO TO 693                                      !(ENGINE?)NO.
        NENG=0                                                        !ZERO # OF ENGINES.
        DO 692 I=1,20
         JENG(I)=0                                                    !ZERO GEAR ASSIGNMENTS.
  692   CONTINUE
        GO TO 695                                                     !GO SET ENG-NOT-LOADED FLG.
  693   IF(IPRNT.EO.4) NUMG=0                                         !(GEAR?)YES,ZERO # OF GEARS.
        IF(IPRNT.EO.5) NACC=0                                         !(ACCESSORY?)YES, ZERO # OF ACCESSORIES.
  695   NPARTS(IPRNT)=0                                               !FLG PART TYPE IPRNT NOT LOADED.
        IF(IALL.EC.0) GC TO 697                                       !(ZERO ALL PARTS?)NO.
  690   CONTINUE                                                      !NEXT.
  697   IF(NDEL.EC.0) GC TO 615                                       !(PART DELETED?) NO.
C
C*****************************************************************
C
  900   RETURN                                                        !*DONE, BYE!!!!
C
C*****************************************************************
C
C     FORMAT STATEMENTS
C
 1240   FORMAT(/' ? DSK-PART 'A5,1XA10
       1.' NOT ON PARTS DATA FILE 'A6':'A10'[ '05','03' ]')
 1615   FORMAT(/' PARTS DELETED FROM VEHSIM PARTS DATA BASE:')
 1620   FORMAT(1XA6':'A10'[ '05','C3' ]'3XA5,3XA10)
```

```
      SUBROUTINE DSKDIR
C
C     ENTRY POINTS: DSKDIR
C
C     SUBROUTINES CALLED: CHKFIL, DSKCTR, DSKRD, ICRCNT, IGET,
C                         LOOKUP, NWRCNT, PRNOUT, PUT
C
C     CALLED BY: INPEAT
C
C**********************************************************************
C
C
      INCLUDE 'COMMS/BCLIST'
C
      LOGICAL LPAGE
C
      DOUBLE PRECISION TUN,UNMASK,TTUN
C
      DIMENSION NPART(14),DIRDAT(27),BPPN2(2),INDIR(14),NHPART(14)
C
      EQUIVALENCE (BPPN,BPPN2(1))
C
C
      DATA ( NPART(I),I=1,NPART) /'ENGIN','CONVE','VEHIC','GEAR'
     2,'ACCES','DRIVI','SHIFT','RODIE','TIRE','TRANS','AXLE'/
     3,ISCR/2/,JCT/5/,IQMARK/63/,NHPART/14*2/
C
C**********************************************************************
C
C    PRINT DIRECTORY OF PARTS DATA FILE AND/OR DUMP PARTS DATA
C
      NDISK=3                                              !RESET IN CASE ~C , .REENTER WAS DONE
      JPRNT=IPRNT                                          !SAVE TASK CODE.
      IWILD=0                                              !ASSUME PART NAME NOT WILD CARD.
      IF(UN.NE.DSTAR) GO TO 401                            !(PART NAME COMPLETELY WILD?) NO.
      UNMASK=0..                                           !YES, SET MASK TO FULL NAME.
      GO TO 403
  401 UNMASK=DWMASK(10)                                    !ASSUME MASK FOR NO WILD.
      DO 402 I=1,10                                        !CHECK NAME FOR WILD CHAR "?".
      II=I
      INDIR(II)=0                                          !ZERO DIR PG#'S.
      IF(IGET(UN,II,IALI).EQ.IQMARK) CALL PUT(UNMASK,II,0) !(CHAR #II IN NAME UN WILD?) YES, MASK IT.
  402 CONTINUE                                             !NXT CHAR.
  403 IF(UNMASK.NE.DWMASK(10)) IWILD=1                     !(ANY WILD CHAR?) YES, FLG IT.
      CALL DAND(UN,UNMASK,TUN)                             !SET NAME MASK.
      IF(UT.EQ.HSTAR) GC TO 404                            !(PART TYPE WILD?) YES.
      CALL LOCKUP(UT,ICRCNT(UT,1),HPART,NHPART,NUMPAR,IPRNT,DBLANK
     2,$404)                                               !NO,LOOK IT UP, (GOT IT?) YES.
       WRITE(JCT,1404) UT                                  !NO, *ERR UNKNOWN PART TYPE.
      GO TO 900                                            !GO BYE.
C
  404 IDIRPG=1                                             !INTI DIR PG#.
      ICNTT=0                                              ! CNT BY PART TYPE
      IGTCT=0                                              ! GRAND TOTAL OF ALL PARTS
      NXTPG=1                                              !INTI DUMP PG#.
      IDIR=IUNIT                                           ! DIR UNIT
      IF(JPRNT.NE.0)GC TO 410                              !(NEED SCR FILE FOR DIR?)NO.
      IDIR=ISCR                                            !YES,SET UP FOR DIR TO SCR FILE..
      CALL DSKCTR (-21 , SECOUT )                          ! OPEN SCR FILE TO WRITE DIR ON
C
  410 IF(UT.NE.HSTAR) GC TO 415                            ! (ALL PARTS?) NO.
```

```
      DO 480 IPF=1,NUMPAR                                              ! LOOP THROUGH ALL PART TYPES
      IPRNT=IPF                                                        !SET PART TYP PTR.
  415 IPSAV=IPRNT
      CALL CHKFIL(DASDEV(IPRNT),DASFIL(IPRNT),DASPPN(1,IPRNT),
      1$417)                                                           !SEE IF FILE IS ACCESSBALE
      GO TO 470                                                        !ERROR RETURN
 417  IF(JPRNT.GT.0)GO TO 419
      IF(IPRNT.EQ.1)IENG=1
      IF(IPRNT.EQ.4)NGEART=1
      IF(IPRNT.EQ.5)NACC=1
 419  CALL DSKCTR(IPRNT,SEQIN)                                         ! NO, REOPEN NEEDED DB FILE FOR INPUT
      IP=IPRNT                                                         ! SET POINTER TO OPEN FILE INFO
      IF(SNGLES) IP=12                                                 ! (SINGLE FILE DB?) YES, RESET FILE PTR.
      WRITE(IDIR,1400) HPART(IPRNT)                                    ! DIR FILE HEADER
      1,DATE,DASDEV(IP),DASFIL(IP),DASPPN(1,IP),DASPPN(2,IP)
      IF(JPRNT.GE.0.AND.TTY.AND.(.NOT.LIMTTY))
      2         WRITE(JCT,1400) HPART(IPRNT)
      3,DATE,DASDEV(IP),DASFIL(IP),DASPPN(1,IP),DASPPN(2,IP)
      ILIN=3                                                           ! SET LINE COUNT
      INDIR(IPRNT)=IDIRPG                                              !SAVE DIR PG #.
      LPAGF=.TRUE.                                                     ! SET FLAG TO PAGE
C
  420 READ(NDISK,END=460) BN,ET,NREC,CDATE,CHOUR,BPROT,BPPN,DUMCOM
      IF ( ET.NE.HPART(IPRNT) )  GO TO 420                             ! (PART TYPE NEEDED?) NO, LOOK NXT.
      CALL DAND(BN,ONEASK,TTUN)
      IF(TTUN.NE.TUN) GO TO 420                                        ! YES,(GOOD NAME?) NO, LOOK NXT.
      ICNIT=ICNIT+1                                                    !INC CNT OF PARTS FOUND.
      IF(JPRNT.EQ.-1) GO TO 442                                        ! YES,(DUMP ONLY?) YES, GO DUMP.
C
      IF(.NOT.LPAGE.OR.IDIR.EQ.JCT) GO TO 425                          ! (PAGE?) NO.
      IF(ILIN.EQ.0) WRITE(IDIR,1408)                                   ! YES, (AT TOP OF PG?) NO.
      IF(JPRNT.EQ.0) WRITE(IDIR,1406) IDIRPG                           !(DUMP&DIR?) YES.
      IF(JPRNT.EQ.1) WRITE(IDIR,1407) IDIRPG                           !(DIR ONLY?) YES.
      ILIN=ILIN+3                                                      ! INC LINE COUNT
      IDIRPG=IDIRPG+1                                                  !INC DIR PG CNT.
      LPAGE=.FALSE.                                                    ! RESET PAGE FLAG
C
  425 IF(TTY.AND.(.NOT.LIMTTY)) WRITE(JCT,1412) BN                     ! (DIR/P ON TTY?) YES.
      IF(JPRNT.EQ.0) GO TO 440                                         !(DUMP&DIR?) YES.
C
C  DIRECTORY DUMP ONLY
C
      WRITE (IDIR,1401)  BN,CDATE,CHOUR,BPROT,BPPN2,DUMCOM
      IF(NREC.GT.0) GO TO 455                                          !(MULTI SECT?) NO.
  430 READ(NDISK,END=460) PNAME,AT                                     ! LOOK NXT PART
      IF(PNAME.EQ.BN.AND.AT.EQ.ET) GO TO 430                           ! (END PART?) NO
      BACK SPACE NDISK                                                 ! YES, POSI FILE
      GO TO 455
C
C  PARTS AND DIRECTORY DUMP
C
  440 WRITE(IDIR,1402) BN,CDATE,CHOUR,BPROT,BPPN2,NXTPG,DUMCOM
C
C    LOAD PART DATA FROM PARTS DATA FILE
C
  442 BACKSPACE NDISK                                                  !POSI DB FIL.
      NRECSV=NREC                                                      !SAVE.
      NREC=IABS(NREC)                                                  !IN CASE MULTI SECT, FLG 1ST SECT READ STATEMENT.
      IF(NREC.NE.6) GO TO 443                                          !(DRS?) NO.
      NDSEG=0                                                          !YES, ZERO DRS OFFSET.
      IF(NRECSV.LT.0) NSEC=1                                           !(MULTI SECT DRS?) YES,SET REC PTR.
      GO TO 445                               !GO LOAD.
 443  IF(NREC.NE.8) GO TO 445                 !(RTE?) NO.
      NRTE=0                                  !YES, ZERO RTE OFFSET.
      IF(NRECSV.LT.0) NRTE=1                  !(MULTI SECT RTE?) YES, SET REC PTR.
 445  CALL DSKRD                         !LOAD PART, ADDITIONAL SECTS CALLED FOR BY PRNOUT THROUGH /*USE/.
C
C    DUMP PART DATA
C
      IF(NREC.LT.0) IPRNT=IPRNT+200           ! (MULTI SECT?) YES, FLG PRNOUT NO RELOAD OF 1ST SECT
      CALL PRNOUT                             ! LIST PART DATA
      IPRNT=IPSAV                             ! RESET PART TYPE PTR
 455  IF(IWILD.EQ.0) GO TO 460                !(WILD CARD?) NO.
      IF(JPRNT.EQ.-1) GO TO 420               !YES,(DUMP ONLY?) GO NEXT.
      ILIN=ILIN+1                             ! INC LINE COUNT
      IF(ILIN.LE.60) GO TO 420                ! (SET PAGE FLAG?) NO.
      LPAGE=.TRUE.                            ! YES , PAGE.
      ILIN=0                                  ! ZERO LINE COUNT
      GO TO 420
C
 460  IF(JPRNT.EQ.-1) GO TO 467               !(DUMP ONLY?) YES.
      IP=IPRNT
      IF(SNGLES) IP=12
      WRITE(JCT,1405) ICNTT,HPART(IPRNT)
     1,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
      WRITE(IDIR,1405) ICNTT,HPART(IPRNT)
     1,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
 467  IGTCT=IGTCT+ICNTT                       ! SUB TOT GRAND TOTAL
      ICNTT=0                                 ! ZERO PRT CNT
      GO TO 475
C
 470  IP=IPRNT                                !SET DB FIL PTR.
      IF(SNGLES) IP=12                        !(SINGLE DBFIL?)YES, RESET PTR.
      WRITE(JCT,1470) HPART(IPRNT)            !NON-EXISTENT DB FIL.
C
 475  IF(UT.NE.HSTAR) GO TO 482               !(WILD PART?) NO.
 480  CONTINUE                                !YES, DO NEXT PART TYPE.
C
C  IF PARTS DATA DUMP COPY DIRECTORY FROM SCRATCH UNIT 2 TO LPT UNIT 6
C
 482  IF(JPRNT) 680,483,486                   !(TASK?) DUMP/DUMP&DIR/DIR.
 483  CALL DSKCTR(-21,SEQIN)                  !NO, CLOSE SCR FIL CONTAINING DIR & REOPN FOR INPUT
 485  READ(ISCR,1403,END=486) DIRDAT           !(READ DIR REC, (EOF?) YES.
      WRITE(IUNIT,1403) (DIRDAT(K),K=1,NWRCNT(DIRDAT,27))   !NO, WRITE THE REC.
      GO TO 485                               !NEXT.
C
 486  N=1                                     !ASUME 1 FILE FOR PART TYP NOT WILD.
      IF(UT.NE.HSTAR) GO TO 490               !(WILD PART?) NO, INDEX TO DIR NOT NEEDED.
      WRITE(IUNIT,1496) DATE,IDIRPG           !INDEX TO DIR HEADER.
      N=0                                     !INTI CNT OF FULL DB FILES.
      DO 488 I=1,NUMPAR                       !WRITE INDEX TO DIR.
      IF(INDIR(I).EQ.0) GO TO 487             !(DIR FOR PART TYPE?) NO.
      WRITE(IUNIT,1487) HPART(I),INDIR(I)     !WIRTE INDEX TO DIR ENTRY.
      N=N+1                                   !INC FIL CNT.
      GO TO 488
 487  WRITE(IUNIT,1488) HPART(I)              !REPORT NO DIR FOR PART TYPE.
 488  CONTINUE                                !NEXT.
C
 490  IF(N.GT.1) WRITE(JCT,1410) IGTOT,N      !(TTY IS JCT?) YES.
      IF(IGTCT.GT.1)WRITE(IUNIT,1410) IGTOT,N
      IF(JPRNT.EQ.1) GO TO 900                !(DIR ONLY?) YES, DONE.
C
```

```
C    RESET PARTS ACCOUNTING
C
  680           IF(UT.NE.HSTAR) GO TO 685
      DO 690 IPFNT=1,NUMPAR
  6E5 IF(IPRNT.NE.1) GO TO 693
      NENG=0
      DO 692 I=1,20
       JENG(I)=0
  692 CONTINUF
      GO TO 695
  693 IF(IPRNT.EQ.4) NUMGT=0
      IF(IPRNT.EQ.5) NACC=0
  695 NPAFTS(IPFNT)=0                                                   !SET FLG PART NOT LOADED.
      IF(UT.NF.HSTAR) GC TO 900                                         !(ALL PART TYPES?)NO.
  690 CONTINUE                                                          !YES, NEXT PART TYPE.
C
C*******************************************************************
C
  900 RETURN                                                            !*DONE, BYE!!!!
C
C*******************************************************************
C
C     FORMAT STATEMENTS
C
 1400 FORMAT ('1 'A5' PARTS DATA FILE DIRECTORY 'A9,3X
     1,A6':'A10'['O5','O3' ]'/2X,72('-')/)
 1401 FORMAT(1XA10,1XA9,1XA5' <'O3'> ['C5','O3' ] '16A5)
 1402 FORMAT(1XA10,1XA9,1XA5' <'O3'> ['C5','O3' ] 'I4,1X16A5)
 1403 FORMAT(30A5)
 1404 FORMAT(/' % DSKDIR - "'A5'" UNKNOWN PART TYPE.')
 1405 FORMAT(/' A TOTAL OF'I5' 'A5' PARTS ON FILE '
     1,A6':'A10'['C5','O3' ]')
 1406 FORMAT(' NAME'7X'CREATICN'8X'PROT  PPN'9X'PAGE COMMENT'
     1,62X'PAGE A-'I4/
     1,1X10('-') 1X15('-')' -----'1X11('-')' ----' 1X80('-'))
 1407 FORMAT(' NAME'7X'CREATICN'8X'PROT  PPN'9X'COMMENT'
     1,62X'PAGE A-'I4/
     1,1X10('-') 1X15('-')' -----'1X11('-') 1X80('-'))
 1408 FORMAT('1'$)
 1410 FORMAT(/' A GRAND TOTAL OF'I5' PARTS CN 'I2' FILES.'/'1')
 1412 FCRMAT(1XA10)
 1470 FCRMAT(/' % DSKDIR- SPECIFIED 'A5' DB FILE NCT FOUND OR'
     1,' EMPTY.'/)
 1486 FORMAT('1'A10,105X'PAGE A-'I4///10X'PART TYPE    '
     1,'DIRECTCRY PAGE NUMBER'/10X9('-') 3X21('-')//)
 1487 FORMAT(12XA5,7X'A-'I4)
 1488 FORMAT(12XA5,7X'X NO PAFTS ON FILE.')
      END
      SUBROUTINE DSKRD
C
C     ENTRY POINTS: DSKRD
C
C     CALLED BY: DSK, DSKDEL, DSKDIR
C
C************************************************************************
C
C
      INCLUDE 'COMMS/NCLIST'
C
      DIMENSION SPARE(5),DATAT(5)
C
C
C***************************************************************************
C
      GO TO (101,102,103,104,105,106,107,108,109,110,111),IPRNT         !(PART TYPE TO BE READ?)
C
C===========================================================================
C     LOAD ENGINE DATA
C
  101 IB     = (IENG-1) * 4  + 1                                        !CALC PTR TO ENG MAP LOC TO BE LOADED.
      IE     = IB + 3
      READ(NDISK,END=920)  ENAME(IENG),BT,NREC,CDATE,CHOUR,EPROT(IENG)
     1,EPPN(IENG),ECOM,(SPARE(I),I=1,4),LDIES
     2,DISP,ICYL,IMIN,IMAX,THRMAX
     2,THRMIN,EINER,BORE,STROKE,FSPGR,NCYCLE,RPMAX(IENG)
     3,RPMIN(IENG),NRPM(IENG),ENMIN(IENG),LRPM,LPS,LTOR
     4,LEMEP,LHP,LLBHR,LPSEC,LGALHR
     5,(NTOR(IENG,K),ERPM(IENG,K),K=1,20)
     6,(((EMAP(I,J,K),J=1,20),K=IB,IE),I=1,NRPM(IENG))
      GO TO 900
C
C===========================================================================
C     LOAD TORQUE CONVERTER DATA
C
  102 READ(NDISK,END=920)  CNAME,BT,NREC,CDATE,CHOUR,CDPROT,CDPPN        !TO GET CONVERTER TYPE COAST OR DRIVE.
     1,CCOM,SPARE,COAST
      BACKSPACE NDISK                                                   !POSITION DB FILE.
      IF ( COAST )  GO TO 1020                                          !(COAST CONVERTER ?)YES.
      READ(NDISK,END=920) CNAME,BT,NREC,CDATE,CHOUR,CDPROT,CDPPN         !NO,LOAD DRIVE CONVERTER.
     1,CCOM,SPARE,COAST,CONTCR,CDIAM,NTORP
     2,AI1,AI2,TIN,TOUT,SPIN,SCUT,NTC,AKC,SRD,TRD,TORDPK,NTBP
      GO TO 900
C
 1020 READ(NDISK,END=920)  CNAME,BT,NREC,CDATE,CHOUR,CCPROT,CCPPN        !LOAD COAST CONVERTER.
     1,CCOM,SPARE,COAST,CONTCR,CDIAM,NTORP
     2,AI1,AI2,TIN,TOUT,SPIN,SCUT,NTC,AKC,SRC
      GO TO 900
C
C===========================================================================
C     LOAD VEHICLE DATA
C
  103 READ(NDISK,END=920)  VNAME,BT,NREC,CDATE,CHOUR,VPROT,VPPN,VCOM
     1,LVEMAX,(SPARE(I),I=1,4),WGT,CD,CDC,AREA,RAR,WLSG,NRAX
     2,((RRAR(I),NRAX(I),AXRIM(J,I),AKTOFQ(J,I),J=1,NPAX(I))
     3,I=1,NRAX),LVNEW,ID,(DATAT(I),I=1,ID)
      LVEMAX=.NOT.LVEMAX
      IF(LVNEW) GO TO 900                                               !(NEW VEHICLE FORMAT?)YES.
      WRAD=DATAT(1)                                                     !NO, LOAD TIRE DATA.
      FRC1=DATAT(2)
      AIW=DATAT(4)
      TIREFF=DATAT(5)
      GO TO 900
C
C=====================================================================
C     LOAD GEAR DATA
C
  104 READ(NDISK,END=920) GNAME(NGEAR),BT,NREC,CDATE,CHOUR
     1,GPROT(NGEAR),GPPM(NGEAR),TCCM,SPARE,AIGIN(NGEAR)
     2,AIGOUT(NGEAR),GRAT(NGEAR),ERAT(NGEAR),NGRLSS(NGEAR)
     3,(GRPM(I,NGEAR),GRTORQ(I,NGEAR),I=1,NGRLSS(NGEAR))
      GO TO 900
C
C=====================================================================
C     LOAD ACCESSORY DATA
C
  105 READ(NDISK,END=920) ANAME(NACC),BT,NREC,CDATE,CHOUR,APROT(NACC)
     1,APPM(NACC),ACCM,SPARE,AIAS(NACC),NNA(NACC)
     2,(ACCS(I,NACC),ACCT(I,NACC),I=1,NNA(NACC))
      GO TO 900
C
C=====================================================================
C     LOAD DRIVING SCHEDULE DATA
C
  106 IF(NREC.LT.0) GO TO 1060                                         !(1ST SECT TYPE?)NO.
      READ(NDISK,END=920) DNAME,BT,NREC,CDATE,CHOUR,DPROT,DPPM
     2,DCCM,DUTCYC,(SPARE(J),J=1,4),TO,DO,VO,AO,NGO,NSEG
     3,(TSEG(I),ASEG(I),VSEG(I),PWOT(I),ATHOLD(I),NGSEG(I)
     4,THRATE(I),CSEG(I),PCSEG(I),FCSTSE(I),VELSEG(I)
     5,ITYSEG(I),I=1,NSEG)
      IF(DUTCYC.LT.1.E-4)DUTCYC=1.
      LSTSEC=.TRUE.                                                    !ASSUME LAST SECT.
      IF(NREC.LT.0) LSTSEC=.FALSE.                                     !(LAST SECT?)NO, RESET FLG.
      GO TO 900
C
 1060 READ(NDISK,END=920) DNAME,BT,NREC,LSTSEC,SPARE,NSEG,(TSEG(I)    !2ND OR GT SECT READ ST.
     1,ASEG(I),VSEG(I),PWOT(I),ATHOLD(I),NGSEG(I)
     2,THRATE(I),CSEG(I),PCSEG(I),FCSTSE(I),VELSEG(I)
     3,ITYSEG(I),I=1,NSEG)
      GO TO 900
C
C=====================================================================
C     LOAD SHIFT LOGIC DATA
C
  107 READ(NDISK,END=920) SNAME,BT,NREC,CDATE,CHOUR,SPROT,SPPM
     1,SCCM,SPARE,GOVPSI,OUTRPM,NGET,LVAC,LDTH
     2,LENG,GDAT,LCEINT,NUMG,PARAB,LDETE,LDETV,GOVLIM
     3,(IGF(I),IGT(I),SHFTIM(I),LGFREE(I),LPSHF(I)
     4,EFFST(I),ABRDET(I),NSETS(I),DETPT(I),DETRPM(I)
     5,(SHPTPT(J,I),SHPTRP(J,I),J=1,10),I=1,((NUMG-1)*2))
      GO TO 900
C
C=====================================================================
C     LOAD ROUTE DATA
C
  103 IF(NREC.LT.0) GO TO 1090                                         !(1ST SECT TYPE?)NO.
      READ(NDISK,END=920) RNAME,BT,NREC,CDATE,CHOUR,RPROT,RPPM
     1,RCCM,SPARE,NRDIST,(RDIST(I),RGRADE(I)
     2,RCOEF(I),RVWIND(I),I=1,NRDIST)
      LSTRTE=.TRUE.                                                    !ASSUME LAST SECT.
      IF(NREC.LT.0) LSTATE=.FALSE.                                     !LAST SECT?) NO, RESET FLG.
      GO TO 900
C
 1080 READ(NDISK,END=920) RNAME,BT,NREC,LSTATE,SPARE,NRDIST,(RDIST(I)  !2ND OR GT SECT READ ST.
     1,RGRADE(I),RCOEF(I),RVWIND(I),I=1,NRDIST)
9000  GO TO 900
C
C=======================================================================
C     LOAD TIRE DATA
C
  100 READ(NDISK,END=920) TNAME,BT,NREC,CDATE,CHOUR,TPROT,TPPN
     1,TCCM,SPARE,WRAD,FFC1,FRC2,TIREPF,AIW
      GO TO 900
C
C=======================================================================
C     LOAD TRANSMISSION DATA
C
  110 READ(NDISK,END=920) TRNAM,BT,NREC,CDATE,CHOUR,TRPROT,TRPPN
     2,TRCCM,SPARE,NGTR,(GEANAM(I),GEANUM(I),I=1,NGTR)
      GO TO 900
C
C=======================================================================
C     LOAD AXLE DATA
C
  111 READ(NDISK,END=920) AXNAME,BT,NREC,CDATE,CHOUR,AXPROT,AXPPN
     2,AXCOM,SPARE,RAR,NRAX,(ERAR(I),NPAX(I),(AXRPM(J,I)
     3,AXTORQ(J,I),J=1,NPAX(I)),I=1,NRAX)
C
      GO TO 900
C
C*************************************************************
C
  920 LEFIL=.TRUE.                                                     !FLG EOF ON DB READ FOUNDISK.
  900 RETURN                                                           !*DONE, BYE.
C
      END
      SUBROUTINE DSKWR
C
C        ENTRY POINTS: DSKWR
C
C        CALLED BY: DSK, DSKDEL
C
C*********************************************************************
C
C
         INCLUDE 'COMMS/NCLIST'
C
         DIMENSION SPARE(5),DATAT(5)
C
C
C*********************************************************************
C
   10    GO TO (101,102,103,104,105,106,107,108,109,110,111),IPRNT
C
C=========================================================================
C      STORE ENGINE DATA
C
  101    IB=(IENG-1)*4+1                                                 !CALC PTR TO ENG MAP TO STORE.
         IE=IB+3
C
         WRITE (NDISK )   EN,BT,NREC,CDATE,CHOUR,EPROT(IENG),EPPN(IENG)
         1,ECCM,(SPARE(I),I=1,4),LCIES
         2,DISP,ICYL,IPIB,IMAX,THRMAX
         1,TFRMIN,FINER,BORE,STROKE,FSPGR,NCYCLE,RPMAX(IENG)
         2,REMIN(IENG),NFPM(IENG),ENMIN(IENG),LRPM,LPS,LTOR
         3,LEMEP,LHP,LIBHR,LBSEC,LGALHF
         4,(NTOR(IENG,K),EFPM(IENG,K),K=1,20)
         5,(((EMAP(I,J,K),J=1,20),K= IB,IB),I=1,NRPM(IENG))
         GO TO 900
C
C=========================================================================
C      STORE TORQUE CONVERTER DATA
C
  102    IF ( COAST )  GO TO 1020
         WRITE (NDISK )   EN,BT,NREC,CDATE,CHOUR,CDPROT,CDPPN
         1,CCCM,SPARE,COAST,CONTCR,CDIAM,NTORP,AI1
         2,AI2,TIN,TOUT,SPIN,SOUT,NTD,AKD,SRD,TRD,TORBPK,NTBP
         GO TO 900
C
 1020    WRITE (NDISK )   EN,BT,NREC,CDATE,CHOUR,CCPROT,CCPPN
         1,CCCM,SPARE,COAST,CONTCR,CDIAM,NTORP,AI1
         2,AI2,TIN,TOUT,SPIN,SOUT,NTC,AKC,SRC
         GO TO 900
C
C=========================================================================
C      STORE VEHICLE DATA
C
  103    ID=1                                                            !SET PTR ON TIRE DATA ARRAY FOR NO TIRE DATA.
         IF(LVNEW) GO TO 1010                                            !(NEW VEHICLE FORMAT?)YES.
         ID=5                                                            !NO,OLD VEHICLE SET CNT OF TIRE DATA WRDS TO WRITE.
         DATAT(1)=WRAC                                                   !GET TIRE DATA.
         DATAT(2)=FRC1
         DATAT(3)=FRC2
         DATAT(4)=AIW
         DATAT(5)=TTREFF
 1030    LVEHAX=.NOT.IVEHAX
         WRITE(NDISK) EN,ET,NREC,CDATE,CHOUR,VPROT,VPPN,VCOM
```

```
     1,LVEHAX,(SPARE(I),I=1,4),WGT,CD,CDC,AREA,BAR,NLSG,NRAX
     2,(ERAR(I),NFAX(I),(AXREM(J,I),AXTORQ(J,I),J=1,NPAX(I)),I=1,NRAX)
     3,LVNEW,ID,(DATAT(I),I=1,ID)
      GO TO 900
C
C==========================================================================
C     STORE GEAR DATA
C
  104 WRITE(NDISK) EN,ET,NREC,CDATE,CHOUR,GEROT(NGEAR),GPPN(NGEAR).
     1,GCOM,SPARE,AIGIN(NGEAR),AIGCUT(NGEAR)
     2,GRAT(NGEAR),EFAT(NGEAR),NGRISS(NGEAR)
     3,(GRPM(I,NGEAR),GRTORQ(I,NGEAR),I=1,NGRLSS(NGEAR))
      GO TO 900
C
C==========================================================================
C     STORE ACCESSORY DATA
C
  105 WRITE (NDISK)  EN,ET,NREC,CDATE,CHOUR,APROT(NACC),APPN(NACC)
     1,ACCM,SPARE,AIDS(NACC),NBA(NACC)
     2,(ACCS(I,NACC),ACCT(I,NACC),I=1,NBA(NACC))
      GO TO 900
C
C==========================================================================
C     STORE DRIVING SCHEDULE DATA
C
  106 IF(NSEG.LT.0) GO TO 1060                                         !(1ST SECT TO BE WRITTEN?)NO.
      WRITE(NDISK) EN,ET,NREC,CDATE,CHOUR,DPROT,DPPN                   !YES.
     1,DCOM,DOTCYC,(SPARE(J),J=1,4),TO,DO,VO,AO,NGO,NSEG,(TSEG(I)
     2,ASEG(I),VSEG(I),PWOT(I),ATHOLD(I),NGSEG(I)
     3,THRATE(I),DSEG(I),PCSEG(I),ECSTSB(I),VELSEG(I)
     4,ITYSEG(I),I=1,NSEG)
      GO TO 900
C
 1060 NSEG=IABS(NSEG)                                                  !GET RID OF FLG.
      WRITE(NDISK) EN,ET,NREC,LSTSEC,SPARE,NSEG,(TSEG(I)
     2,ASEG(I),VSEG(I),PWOT(I),ATHOLD(I),NGSEG(I)
     3,THRATE(I),DSEG(I),PCSEG(I),ECSTSE(I),VELSEG(I)
     4,ITYSEG(I),I=1,NSEG)
      GO TO 900
C
C==========================================================================
C     STORE SHIFT LOGIC DATA
C
  107 WRITE(NDISK) EN,ET,NREC,CDATE,CHOUR,SEROT,SPPN
     1,SCOM,SPARE,GOVPSI,CUTEPE,NGET,LVAC,LDTH,LENG
     2,GIAT,LDETNT,NUMG,PARAE,LDETE,LDETV,GOVLIN
     3,(IGF(I),IGT(I),SHFTIM(I),LGFREE(I),LPSHP(I)
     4,EFEST(I),ABRDFT(I),NSFTS(I),DETPT(I),DETRPM(I)
     5,(SFETET(J,I),SFETRP(J,I),J=1,10),I=1,((NUMG-1)*2))
      GO TO 900
C
C==========================================================================
C     STORE ROUTE DATA
C
  108 IF(NRDIST.LT.0) GO TO 1080                                       !(1ST SECT TO BE WRITTEN?)NO.
      WRITE(NDISK) EN,ET,NREC,CDATE,CHOUR,RPROT,RPEN                   !YES.
     1,RCOM,SPARE,NRDIST,(RDIST(I),RGRADE(I)
     2,RCOEF(I),RVWIND(I),I=1,NRDIST)
      GO TO 900
C
 1080 NRDIST=IABS(NRDIST)                                              !GET RID OF FLG.
```

```
      WRITE(NDISK) EN,ET,NREC,LSTRTE,SPARE,NRDIST,(RDIST(I),RGRADE(I)
     1,RCOEF(I),RVWIND(I),I=1,NRDIST)
      GO TO 900
C
C=========================================================================
C     STORE TIRE DATA
C
  109 WRITE(NDISK) EN,DT,NREC,CDATE,CHOUR,VEROT,VPPM,TCOM
     1,SPARE,WRAD,FRC1,FRC2,TIREFF,AIW
      GO TO 900
C
C=============================================================================
C     STORE TRANSMISSION DATA
C
  110 WRITE(NDISK)TRNAM,BT,NREC,CDATE,CHOUR,TRPROT,TRPPM
     2,TRCOM,SPARE,NGTR,(GEANAM(I),GEANUM(I),I=1,NGTR)
      GO TO 900
C
C=============================================================================
C     STORE AXLE DATA
C
  111 WRITE(NDISK)AXNAME,DT,NREC,CDATE,CHOUR,AXPROT,AXPPM
     2,AXCOM,SPARE,RAF,NBAX,(EBAR(I),NPAX(I),(AXRPM(J,I)
     3,AXTORC(J,I),J=1,NPAX(I)),I=1,NBAX)
C
      GO TO 900
C
C*********************************************************************
C
  900 RETURN                                                    DONE, BYE..
C
      END
      SUBROUTINE ENGINE
C
C       ENTRY POINTS: ENGINE
C
C       CALLED BY: GOBACK, REMAP
C
C*****************************************************************************
C
C
      DOUBLE PRECISION   UN
C
      LOGICAL NCRPM,LWCT
      COMMON /GET/ DT,UN,NUG(20),JENG(20),IENG
      COMMON /ENGMAP/ RPMAX(2),RPMIN(2),NRPM(2),RPME,TORQE,FRATE,VAC,
     1THR,MAPOK,
     1IERRE,NTCR(2,20),EMAP(20,20,8),ERPM(2,20),ENMIN(2),SPIDLE(2)
      COMMON /TEMP/ TCBC,LWCT,TWOT,TRIM
C
C     CHECK TO SEE IF RPM IS ON THE ENGINE MAP
C
      K1=(IENG-1)*4+1
      K2=K1+1
      K3=K2+1
      K4=K3+1
      IERRE=0
      MAPCK=1
      IF(RPME.GE.RPMAX(IENG)) GO TO 10
      IF(RPME.GT.RPMIN(IENG)) GO TO 20
      IF ( ERPM(IENG,1)-ENMIN(IENG).LT.-1. )  GO TO 20
C
C     ENGINE SPEED BELOW MIN AND SET TO BOTTOM OF MAP
C
      I=2
      MAPCK=2
      GO TO 40
C
C     ENGINE SPEED OFF MAP IN UPPER DIRECTION
C
   10 I=NRPM(IENG)
      MAPCK=3
      GO TO 40
C
C     DETERMINE WHERE ENGINE SPEED IS ON MAP
C
   20 NRDUM=NRPM(IENG)
      DO 30 I=2,NRDUM
      IF(RPME.LE.ERPM(IENG,I)) GO TO 40
   30 CONTINUE
C
C     PRINT ERROR MESSAGE IF FAILURE TO FIND SPEED SETTING
C
      WRITE (6,100) TORQE,RPME
  100 FORMAT (//2X,5X,60(1H*)//2X,33H***** FAILURE TO FIND RPM SETTING,
     1          11H FOR ENGINE/2X,15H***** TORQUE = ,E15.7, 5X,6HRPM = ,
     2          E15.7/2X,25H***** EXECUTION CONTINUES//2X,5X,60(1H*)//)
      WRITE (6,300) I,NRPM(IENG),ERPM
  300 FORMAT(2I5/(10F10.2))
      IERRE=1
      RETURN
C
C     INTERPOLATE ENGINE SPEED BETWEEN TWO MAP SETTINGS
```

```
C         AND COMPUTE MAX AND MIN THROTTLE SETTINGS AT THAT SPEED
C
   40     IM=I-1
          CO=(RPME-ERPM(IENG,I))/(ERPM(IENG,IM)-ERPM(IENG,I))
          TWOT=EMAP(I,20,K1)+CO*(EMAP(IM,20,K1)-EMAP(I,20,K1))
          TMIN=EMAP(I, 1,K1)+CO*(EMAP(IM, 1,K1)-EMAP(I, 1,K1))
          IF(LWCT) RETURN
   41     CONTINUE
C
C       CHECK FOR TORQUES OFF THE MAP
C
          IF(TORQE.GE.TWOT) GO TO 99
          IF(TORQE.LE.TMIN) GO TO 97
          IF(TORQE.GE.EMAP(IM,20,K1)) GO TO 50
          IF(TORQE.GT.EMAP(IM, 1,K1)) GO TO 60
C
C       TORQUE IS OFF MAP AT LOW END FOR LOWER SPEED SETTING
C
          J=1
          GO TO 75
C
C       TORQUE IS OFF MAP AT HIGH END FOR LOWER SPEED SETTING
C
   50     J=19
          GO TO 75
C
C       SEARCH FOR TORQUE SETTING ON LOWER SPEED SETTING
C
   60     N=21-NTOR(IENG,IM)
          DO 70 J=N,19
          IF(TORQE.LE.EMAP(IM,J+1,K1)) GO TO 75
   70     CONTINUE
C
C       PRINT ERROR MESSAGE IF TORQUE SETTING NOT FOUND
C
          WRITE (6,200) TORQE,RPME
  200     FORMAT (//2X,5X,60(1H*)//2X,34H***** FAILURE TO FIND TORQUE SETTI,
          1         13HNG FOR ENGINE/2X,15H***** TORQUE = ,E15.7, 5X,6HRPM = ,
          2         E15.7/2X,25H***** EXECUTION CONTINUES//2X,5X,60(1H*)//)
          WRITE (6,400) J,N,NTOR, (EMAP(IM,J,K1),J=1,20)
  400     FORMAT(2I5/20I5/(10F10.2))
          IERRR=1
          RETURN
C
C       INTERPOLATE ENGINE PARAMETERS AT LOWER SPEED SETTING
C
   75     JP=J+1
          F1 =EMAP(IM,J ,K2)
          T1 =EMAP(IM,J ,K3)
          V1 =EMAP(IM,J ,K4)
          TO1=EMAP(IM,J ,K1)
          F2 =EMAP(IM,JP,K2)
          T2 =EMAP(IM,JP,K3)
          V2 =EMAP(IM,JP,K4)
          TO2=EMAP(IM,JP,K1)
          C1=0.
          IF(ABS(TO2-TO1).GT.1.E-5) C1=(TO2-TORQE)/(TO2-TO1)
          F6=F2-C1*(F2-F1)
          V6=V2-C1*(V2-V1)
          T6=T2-C1*(T2-T1)
          TO6=TO2-C1*(TO2-TO1)
   78   N=21-NTCR(IENG,I)
C
C     CHECK IF TORQUE IS OFF MAP FOR HIGHER SPEED SETTING
C
        IF(TORQE.GE.EMAP(I,20,K1)) GO TO 82
        IF(TORQE.LE.EMAP(I, 1,K1)) GO TO 84
C
C     LOCATE TORQUE SETTING FOR HIGHER SPEED SETTING
C
        DO 80 K=N,19
        IF(TORQE.LE.EMAP(I,K+1,K1)) GO TO 85
   80   CONTINUE
C
C     PRINT ERROR MESSAGE IF TORQUE SETTING NOT FOUND
C
        WRITE (6,200) TORQE,RPME
        WRITE (6,400) K,N,NTOR,(EMAP(I,K,K1),K=1,20)
        IERRE=1
        RETURN
C
C     INTERPOLATE ENGINE PARAMETERS AT HIGHER SPEED SETTING
C
   82   K=19
        GO TO 85
   84   K=1
   85   KP=K+1
        P3 =EMAP(I,K ,K2)
        T3 =EMAP(I,K ,K3)
        V3 =EMAP(I,K ,K4)
        TO3=EMAP(I,K ,K1)
        P4 =EMAP(I,KP,K2)
        T4 =EMAP(I,KP,K3)
        V4 =EMAP(I,KP,K4)
        TO4=EMAP(I,KP,K1)
        C2=0.
        IF(ABS(TO4-TO3).GT.1.E-5) C2=(TO4-TORQE)/(TO4-TO3)
        P5=P4-C2*(P4-P3)
        V5=V4-C2*(V4-V3)
        T5=T4-C2*(T4-T3)
        TO5=TO4-C2*(TO4-TO3)
   86   C3=C0
C
C     INTERPOLATE ENGINE PARAMETERS BETWEEN SPEED SETTINGS IF ON MAP
C
        TORQ=TO5-C3*(TO5-TO6)
        IF(TORQ.GT.TWCT) GO TO 99
        IF(TORQ.LT.TMIN) GO TO 97
        FRATE=P5-C3*(P5-P6)
        VAC=V5-C3*(V5-V6)
        THR=T5-C3*(T5-T6)
        RETURN
C
C     FOLLOWING ARE INTERPOLATIONS OF DIRECT SETTINGS OF ENGINE PARAMETE
C       FOR POINTS OFF THE MAP
C
   97   FRATE=EMAP(I, 1,K2)+C0*(EMAP(IM, 1,K2)-EMAP(I, 1,K2))
        THR  =EMAP(I, 1,K3)+C0*(EMAP(IM, 1,K3)-EMAP(I, 1,K3))
        VAC  =EMAP(I, 1,K4)+C0*(EMAP(IM, 1,K4)-EMAP(I, 1,K4))
        TORQ=TMIN
        IF(MAPOK.GT.1) GO TO 986
        MAPCK=4
```

```
      RETURN
988   MAPOK=MAPCK+4
      RETURN
 99   FRATE=EMAP(I,20,K2)+CO*(EMAP(IM,20,K2)-EMAP(I,20,K2))
      THR  =EMAP(I,20,K3)+CO*(EMAP(IM,20,K3)-EMAP(I,20,K3))
      VAC  =EMAP(I,20,K4)+CO*(EMAP(IM,20,K4)-EMAP(I,20,K4))
      TOBC=TWOT
      IF(MAPOK.GT.1) GO TO 999
      MAPCK=5
      RETURN
999   MAPOK=MAPCK+6
      RETURN
      END
      FUNCTION ENTERP(ACCT,ACCS,NPTS,NTBL,RPM,NLEN)
C
C    INTERPOLATE ALL DATA CURVES TO COMPUTE SUM OF SPECIFIED POINTS
C
C
C      ENTRY POINTS: ENTERP
C
C      CALLED BY: GOBACK
C
C***********************************************************************
C
      DIMENSION ACCT(NLEN,NTBL),ACCS(NLEN,NTBL),NPTS(NTBL)
C
      ENTERP=0.                                  ! ZERO ACCUM
      IF(NTBL.LT.1) GO TO 35                     ! (ANY TABLES?) NO.
C
      DO 30 I=1,NTBL                             ! YES, LOOP TROUGH TABLES.
      J=NPTS(I)                                  ! LOOK UP # OF DATA PTS IN CURRENT TABLE
      IF(J.LE.1) GO TO 30                        ! (ENOUGH PTS?) NO, SKIP TO NEXT TABLE
C
      IF(RPM.LE.ACCS(1,I)) GO TO 15              ! YES,(RPM OFF LOW END?) YES.
      IF(RPM.GT.ACCS(J,I)) GO TO 11              ! NO,(RPM OFF HGH END?) YES.
C                                                ! NO, RPM IN TABLE
      DO 10 K=2,J                                ! SEARCH TABLE FOR PT JUST BELOW RPM
      IF(RPM.LE.ACCS(K,I)) GO TO 20              ! (GOT IT?) YES.
   10 CONTINUE                                   ! NO.
C
   11 K=J                                        ! ABOVE MAX SPECIFIED
      GO TO 20
   15 K=2                                        ! BELOW MIN SPECIFIED
C
   20 KM=K-1                                     ! SET LOW END PT FOR ENTERP
C
C    COMPUTE TORQUE FOR EACH TABLE
C
      TOR=(ACCT(KM,I)-ACCT(K,I))/(ACCS(KM,I)-ACCS(K,I))*(RPM-ACCS(K,I))
     1+ACCT(K,I)
C
      IF(TOR.GT.0.) ENTERP=ENTERP+TOR            ! (VALID?) YES,ADD PT FROM CURRENT TABLE TO TOTAL
   30 CONTINUE                                   ! NO, ASSUME ZERO.
C
   35 RETURN                                     ! *DONE, BYE
      END
      SUBROUTINE GETACL(ITTER)
C
C     ENTRY POINTS: GETACL
C
C     CALLED BY: ITERAT
C
C**********************************************************************
C
C
      DOUBLE PRECISION TEMP,C1,C2,C3,C4,C5,C6,C7,C8,C9,C10
     1,C11,C12,C13,C14,C15,C16,C17,C18,ENAME
      COMMON /CDEBUG/  IDEBUG,DBEGIN,DSTOP,ISEG1,ISEG2,ISEG,CUMT,CUMD
      COMMON /TCRCON/ TCRBPK,TORQ2,RPM2,COAST,SR,TR,TRD(20),SRD(20),
     1AKE(20),NTD,SRC(20),AKC(20),NTC,NTEP,TIN(20),TOUT(20),SPIN(20),
     2SOLT(20),NTORP,CDIAM,CNAME,CCOM(16),CONTOR
      COMMON /MISC/ RNA(20),TIREFF
      COMMON /ACCESS/ NACC,ACCT(20,20),ACCS(20,20),TORQA,ANAME(20),
     1 AIAS(20)
      COMMON /IC/ECOM(16),ENAME(2),DISP,ICYL,IMIN,IMAX,THRMAX,THRMIN
     1,EINER,BORE,STROKE,PSPCR,NCYCLE
      COMMON /OUTP/FWHEEL,FAERO,FACCEL,TRR,TORQF,DRPMW,DRPME,FROLL,
     1   FGRADE
      COMMON /CONST/   FRC1,FRC2,FAC,CD,AREA,VWIND,WGT,FGC,WRAD,RAR,
     1  GRAT(20),NUMG,NGEAR,AIW,AIP,AI2,ERAT(20),ERAR(2),
     2  AIE,AIA,AI1,EPER,CDC,PHI,PSI,AIGIN(20),AIGOUT(20),WLSG,LTBRZ
     3,NGRISS(20),GRPM(20,20),GRTORQ(20,20),GNAME(20),GCOM(16)
      COMMON /MCCCN/   BB1,BB2,RPMWC,RPMEO,CPHI,VWINDC,VWINDS,AA1,AA2,
     1                  AA3,AA4,AA5,AA6,AA7,AA8,AA9,AA10,RARSQ
      COMMON /CNTRL/   IC,TOLC,VOLD,T,V,ACCEL,D,DT
C
      PRINT6=.FALSE.
      IF(IDEBUG.GE.6.AND.CUMT.GE.DBEGIN.AND.
     2        (CUMT.LE.DSTOP.OR.ISEG2.EQ.0)) PRINT6=.TRUE.
C
      C1=DT/1.4666666666666667D0
C
      C2=AA3*C1**2
C
      C3=(AA2*C1+2*AA3*C1*VOLD+AA4)
C
      C4=(AA1+AA2*VOLD+AA3*VOID**2+EPER*WGT)
C
      AEFFG=ERAT(NGEAR)
      AGRAT=GRAT(NGEAR)
      C5=TR*AGRAT*AEFFG*AA8/WFAC
C
      IF(SR.LT.1.E-3) SR=1.E-3
      C8=AGRAT*RAR*AA5/SR
C
      C9=C8*C1
C
      C10=C8*VOID
C
      C11=BB1*(EINER+AIA)
C
      C12=(AA6+(AA7*(AIGIN(NGEAR)+AIGOUT(NGEAR))
     1         *AGRAT*AGRAT))
C
      C13=AA5*VOLD
C
      C14=AA5*C1
```

```
C
      C15=C11*(C10-EPREC)/DT
C
      C16=C11*C9/DT
C
      C17=C12*(C13-EPMWC)/DT
C
      C18=C12*C14/DT
C
      C6=(C3+C16*C5/TR+C18/WRAD)/C2
C
      C7=((C4-((TITER-TCRQA-C15)/TR*C5))+(C17/WRAD))/C2
C
      TEMP=DSQRT(C6**2-4.*C7)
C
      ACCELM=(-C6-TEMP)/2.
      ACCELP=(-C6+TEMP)/2.
C
      IF(ERINT6)WRITE(6,3001)C1,C2,C3;C4,C5,C6,C7,C8,C9,C10
     2,C11,C12,C13,C14,C15,C16,C17,C18
9001  FORMAT(' $ GETACL C1-C18 ->'/(6G))
      IF(ERINT6)WRITE(6,9000)TITER,ACCEL,ACCELM,ACCELP
9000  FORMAT(' $ GETACL - TITER,ACCEL,ACCELM,ACCELP:',4G)
      ACCEL=ACCELP
      IF(ABS(ACCELM).LT.100.)ACCEL=ACCELM
      RETURN
      END
```

```
        SUBROUTINE GOBACK
C
C       ENTRY POINTS: GOBACK
C
C       SUBROUTINES CALLED: CONVTR, CTRLC, DEBUG, ENGINE,
C                         ENTERE, EXIT, SIMSYS, TTYINP
C
C       CALLED BY: ITERAT, SIMCTR, SIMINT
C
C       EDIT HISTORY
C
C       [607]/SS-4-10-78        CLUTCH
C       [611]/SS-6-22-78        NO BRAKES IF COMING FROM ITERAT
C       [621]/SS-5-9-79         ADD MODIFY LOCKUP GEAR IF OVER A
C                               SPECIFIED RPMS.
C
C*****************************************************************
C
C
        EXTERNAL SIMSYS
C
        INCLUDE 'COMMS/NCLIST'
C
C
        DATA HNCRE/'NCREL'/,HCOAST/'COAST'/,N20/20/,NXCALL/500/
        1,HGOERR/'GOERR'/
C
C*******************************************************************
C
        NGOCAL=NGOCAL+1                                                 !INC CNT OF CALLS THIS DT.
        IF(NGOCAL.GT.NXCALL) GO TO 999                                  !(NOF CALLS THIS DT EXCEEDED MAX?) YES.
C
        V=VOLD+ACCEL*DT/1.466667                                        !COMPUTE VELOCITY AT END OF TIME STEP.
        IF(V.GE.0.)GO TO 3
        V=0.
        ACCEL=-VOLD*1.466667/DT
C
C     SET UP VELOCITY DEPENDENT CONSTANTS FOR THIS TIME STEP
C
3       AGRAT=GRAT(NGEAR)
        AEFFG=ERAT(NGEAR)
        FROLL=AA1+AA2*V
        IF(FROLL.LT.1.E-30) FROLL=0.
        VTOT=V+VWINDC
        VTOTSQ=VTOT*VTOT
C       PSI=0.                                                          !COMMENTED OUT TO INSURE MODIFY PHI WORKS.
        IF(VTOTSQ.GT.1.E-30) PSI=ATAN(VWINDS/VTOT)
        FACCEL=ACCEL*AA4
        FGRADE=EPER*WGT
        RPMW=AA5*V                                                      !CALC RPM OF WHEELS.
        IF(IDYNA) GO TO 4                                               !(DYNO SIM?) YES.
        FAERO=AA3*VTOTSQ*(1.+CDC*PSI)                                   !NO, CALC AERO DYANMIC DRAG.
        GO TO 5
    4   CONTINUE
C       FROLL=FROLL*.5  !COMMENTED OUT PER SPECIFICATION OF HERB GOULD  !DYNO ONLY 2 WHEELS ROTATE RECALC ROLL RESIST.
        IF(RPMW.GT.1.E-3)FAERO=DYN*(V/50.)**2.5*5252./(WRAD*RPMW)       !FOR DYNO CALC AERO DRAG.
C
    5   FWHEEL=(FROLL+FAERO+FACCEL+FGRADE)                              !COMPUTE FORCE AT WHEELS
C
C     COMPUTE REAR END ROTATING INERTIA
C
```

```
      DRPMW=RPMW-RPMWC                                                  !CALC DIF BETWEEN RPM OF WHEELS AND OLD VAL.
      TRR=0.0
      IF(LTRR2) GO TO 15                                                ! (TRR ALWAYS .0?) YES.
      IF(LOCKUP(NGEAR)) GC TO 10                                        ! (GEAR LOCKEDUP ?) YES.
C
      TRR=BE1*(WLSG*AIW+RARSQ*ERAR(1)*((AIP+AIGOUT(NGEAR))              !CALC TRR FOR UNLOCKED GEAR.
     1   *AGRAT*AGRAT*AEFFG*(AI2+AIGIN(NGEAR)))))*DRPMW/DT
      GO TO 15
C
   10 TRR=(AA6+(AA7*(AIGIN(NGEAR)+AIGOUT(NGEAR)))                       !CALC TRR FOR LOCKED GEAR.
     1    *AGRAT*AGRAT)*DRPMW/DT
C
   15 TORCW=WFAC*RWHEEL*TRR                                             !CALC TORQUE AT WHEELS.
C
C    USING WHEEL TORCUE AND RPM , COMPUTE BACK THROUGH DIFFERENTIAL
C    AND REAR BOX TO THE TORQUE CONVERTER
C
      RPMP=RAR*RPMW
      RPMC=AGRAT*RPMP                                                   ![607]
      RPM2=RPMC                                                         ![607]
      IF(.NOT.LCLTCH)GO TO 16                                           ![607]
      RPM2=RPM2C+DRPMC*DT                                               ![607]
C
      IF(LDNSHF)GO TO 19                                                ![607](DOWNSHIFT?) YES.
      IF(RPM2.GT.RPMC)GC TO 16                                          ! (SHIFT OVER?) NO.
18    RPM2=RPMC                                                         !YES.
      LLSH=.FALSE.
      LCLTCH=.FALSE.
      GO TO 16
19    IF(RPM2.LT.RPMC)GC TO 16
      GO TO 18
C
16    TORCP=TORCW/AA8+ENTERP(AXTORQ(1,1),AXRPM(1,1),NPAX(1),NRAX,RPMP
      1,N20)
      TORC2=TORCP/(AGRAT*AEFFG)+ENTERP(GBTORQ(1,NGEAR),GRPM(1,NGEAR)
      1,NGRLSS(NGEAR),1,RPM2,N20)
      IF(SHFTNG) GO TC 17                                               ! (SHIFTING ?) YES, DON'T CHG COAST/DRIVE STATE.
      COAST=.FALSE.                                                     !ASSUME DRIVE.
      IF(TORQ2.LT.-1.E-6) COAST=.TRUE.                                  ! (COASTING ?) YES, SET FLG.
   17 IF ( .NOT. LOCKUP(NGEAR) )   GO TO 996                            ! (GEAR UNLOCKED ?) YES.
      SR     = 1.                                                       !SET SPEED RATIO.
      TR     = 1.                                                       !SET TORQUE RATIO.
      GO TO 20
C
C    COMPUTE SPEED AND TORQUE RATIOS IN TORQUE CONVERTER
C
C  996 CALL CCNVTR                                                      !GET SR+TR RATIO.
996   CONTINUE
      IF(.NCT.FIOD) GC TO 50
C     FOR SPLIT TORQUE CONVERTER
      IF(NGEAR.LE.2) CALL CONVTR
      IF(NGEAR.EQ.3) CALL OVRDRV
      GO TO 52
50    CONTINUE
C     REGULAR CONVERTER
      CALL CCNVTR
52    CONTINUE
      IF(SR.GT.0.) GO TC 20                                             ! (SPEED RATIO 70.?) YES.
C
C    FOR NEGATIVE SPEEC RATIO USE MIN ENGINE SPEED AND MIN SPEED
C
```

```
          RPME=ENMIN(IENG)
          IF(.NOT.LPDLOK)GO TO 200                                          ![621]
          LOCKUP(NGEAR)=.FALSE.                                             ![621]
          IF((IFIX(RPME).GE.MDLKRE) .AND. (NGEAR.EQ.MDLKGR))
          2         LCCKUP(NGEAR)=.TRUE.                                    ![621]
C         RPME=RPMIN(IENG)
C         IF(ABS(ACCEL).LT.1.E-5.AND.V.LT.1.E-5) RPME=ENMIN(IENG)
200       SR=SRD(1)
          TR=TRD(1)
          TORQ1=0.
          GO TO 25
   20     RPME=RPM2/SR
          IF(.NOT.LPDLOK)GO TO 205                                          ![621]
          LOCKUP(NGEAR)=.FALSE.                                             ![621]
          IF((IFIX(RPME).GE.MDLKRE) .AND. (NGEAR.EQ.MDLKGR))
          2         LCCKUP(NGEAR)=.TRUE.                                    ![621]
205       TORQ1=TORQ2/TR
   25     RPM1=RPME
          IF(RPME.GE.ENMIN(IENG).OR.LCLTCH) GO TO 30                        !(RPM ON MAP OR CLUTCH IN ?) YES.
          RPME=ENMIN(IENG)                                                  !SET ENG RPM TO MIN ENG RPM.
          IF(.NOT.LPDLOK)GO TO 29                                           ![621]
          LOCKUP(NGEAR)=.FALSE.                                             ![621]
          IF((IFIX(RPME).GE.MDLKRE) .AND. (NGEAR.EQ.MDLKGR))
          2         LCCKUP(NGEAR)=.TRUE.                                    ![621]
C         RPM1=RPME
C         IF((.NOT.COAST).OR.(.NOT.LOCKUP(NGEAR))) GO TO 29                 !(DRIVE OR GEAR UNLOCKED ?) YES.
C         ABR     = TORQW
C         TORQB   = TORQW - TORQ2*AGRAT*AEFFG*AAB
C         ABR     = TORQW - ABR
C         LBRAKE=.TRUE.
C         TORQP   = TORQW / AAB
C         TORQ2   = TORQP / ( AGRAT * AEFFG )
C         TORQ1   = 0.
C         GO TO 30
   29     SR      = RPM2 / REM1
   30     IF ( V.GE.1.E-5 )GO TO 32                                         !VEL 0.?
            RPME = ENMIN(IENG)                                              ! YES, SET ENG RPM TO MIN ENG RPM.
          IF(.NOT.LPDLCK)GO TO 32                                           ![621]
          LOCKUP(NGEAR)=.FALSE.                                             ![621]
          IF((IFIX(RPME).GE.MDLKRP) .AND. (NGEAR.EQ.MDLKGR))
          2         LCCKUP(NGEAR)=.TRUE.                                    ![621]
C-----------------------------------------------------------------
C
C...      FOLLOWING CODE DELETED FOR [607]
C
C         IF(.NOT.LCLTCH) GO TO 32                                          !(CLUTCH OUT?) YES.
C         IF(IDNSFP) GO TO 31                                               !(DOWN SHIFTING?) YES.
C         RPME=RPPEC-((TCFOAO-TBIBO)*DT/PB1)/(EINER+AIA)                !CALC RPM OF ENG BY SUBTRACTING SPIN DOWN RPM OF ENG DURING DT.
C         GO TO 32                                                          !DURING UP SHIFT.
C    31   RPME=RPPEC+ARFME                                              !SET ENG SPEED DURING DOWN SHIFT BY ADDING SPIN UP RPM DURING DT.
C-----------------------------------------------------------------
   32     DRPPE=RPME-RPPEC                                                  !CALC DIF RPM OF ENG.
C       COMPUTE TORQUE USED BY ACCESSORIES
C
          TORQA=0.
          IF(NACC.GT.0)
          2TORQA=(ENTERP(ACCT(1,1),ACCS(1,1),RMA(1),NACC,RPME,N20))*
          3         DUTCYC                                                  !GET ACCES TORQ LOSSES.
C
C       COMPUTE FRONT END ROTATING INERTIAS
C
```

```
      TORQF=0.0                                                  !SET.
      IF(LTRR2) GC TC 35                                         !(TORQF ALWAYS 0.) YES.
      IF(LOCKUP(NGEAR)) GO TO 33                                 !(GEAR LOCKED UP?) YES.
      TORCF=AA9*DFPME/DT                                         !CALC FCR UNLOCKED GEAR.
      GO TO 35
   33 TORCF=EE1*(EINER+AIA)*DFPME/DT                             !CALC FOR LOCKED UP GEAR.
   35 IF(ICLTCE) GO TC 40                                        !(CLUTCH IN?) YES.
      TORCE=TCRCA+TORCF+TORQ1                                    !NO, CALC TORQUE OF ENG.
      GO TO 70
   40 IF(IDNSEF) TORCF=ATORQF                                    !(DOWN SHIFTING?) YES, ADD TORQUE FOR ENG SPIN UP.
      TORCE=TCRCA+TCRCF                                          !CALC TCRQUE OF ENG.
C
C    DETERMINE STATE OF THE ENGINE
C
   70 CALL ENGINE
C
      IF(.NOT.LCLTCH.CR.LDNSHF) GO TO 75                         !(CLUTCH OUT OR DOWN SHIFTING?) YES.
      TORCE=TMIN                                                 !SET ENG TORQUE.
      TORCF=TMIN                                                 !SET FRONT END TORQUE.
   75 CALL=HHCFM
      IF(ISTROP)CALL='START'
      IF(IDEBUG.EC.1)GO TO 76
      IF((IDEBUG.GT.2.AND.IDEBUG.LT.5).CR.IDEBUG.EQ.7.OR.
     2       (IDEBUG.EC.2.AND.SHFTNG)) CALL DEBUG (CALL)         !(DEBUG PRINT OUT?) YES, DO IT.
76    IF((.NOT.COAST).OF.(TORCE.GE.TMIN).OR.LCLTCH
     2       .OR.LITER) GO TC 99                                 ![611]

      ADF=TCRCW
      LBRAKE=.TRUE.
      TORCW=TCRCW-(TORQE-TMIN)*AGRAT*ABFFG*AA8
      ABR=TORCW-AER
      TORCP=TCRCW/AA8
      TORC2=TCRCP/(AGRAT*AEFFG)
      TORC1=TORC2
      TORCE=TCRC1+TCRCA+TORQF
      MAECK=MAECK-4
      IF((IDEBUG.GT.2.AND.IDEBUG.LT.5).CR.
     2       IDEBUG.EO.7) CAIL DEBUG (HCOAST)                    !(DEBUG ?) YES, DO IT.
C
   59 SHFTNG=ILSH                                                ![60']TURN OFF WHEN LEAVING
      IF(TTY) CALL TTYINP(SIMSTS)                                !(JCT IS TTY? )) YES, IF INTERUPT FOUND GO HANDLE IN SIMSTS.
      RETURN                                                     !*DONE, BYE.
C
  999 WRITE(JCT,1999) EXCALL                                     !"?FAILURE TO CONVERGE."
      CALL CTRLC(HGOEFR)                                         !DEBUG TO JCT.
      IF(FTY) CALL EXIT                                          !(JCT IS PTY?) YES, OH WELL BETTER LUCK NEXT TIME.
      CALL RESETN                                                !NO,"*RESET" TO JCT, REINT FOROTS, + REENTER VSMCTR.
C
 1999 FORMAT(/' ? GOBACK - FAILURE TC CCNVERGE IN 'I4' ATTEMPTS'/)
      END
      SUBROUTINE HLPCMD
C
C     ENTRY POINTS: HLPCMD
C
C     SUBROUTINES CALLED: ASCIZ, CHKFIL, CRLF, ICHCNT, LOOKUP, SKPREC
C
C     CALLED BY: INEDIA
C
C*****************************************************************
C
      LOGICAL FOUND
      DOUBLE PRECISION DHLPFL,INF,HELPF
      DIMENSION LINE(16),HLPFIL(2)
      INTEGER HLPPPN(3)
      INCLUDE 'COMMS/SCLIST'
C
      EQUIVALENCE (HLPFIL,DHLPFL)
C
      DATA FOUND/.FALSE./,HELPF/'VERSIM.HLP'/, HLPFIL(2)/'D.HLP'/
C
      DATA HHELP/5HHELP /,HLPPPN(3)/0/
C
      FOUND=.FALSE.
      HLPPPN(1)=MASPPN(1)
      HLPPPN(2)=MASPPN(2)
20    WRITE(5,40)
40    FORMAT(' ENTER COMMAND OR HELP: '$)
      READ(5,60)WORD
60    FORMAT(A5)
      IF(WORD.EQ.'ALL')GO TO 500
      GO TO 100
70    WRITE(5,80) (HCOMND(I),I=1,41)
80    FORMAT(' THESE ARE THE COMMANDS',/
      1,10(1X,A5),/,10(1X,A5),/,10(1X,A5),/,10(1X,A5),/,
     2 1(1X,A5))
      GO TO 20
C
100   CALL LOOKUP(WORD,-ICHCNT(WORD,1),HCOMND,HCOMND
      1,41,ICMD,HHELP,$160)
C
120   IF(.NOT.FOUND)WRITE(5,140)WORD
140   FORMAT(' HELP UNAVAILABLE FOR ',A5,'.')
C
      RETURN
C
160   IF(ICMD.EQ.37)GO TO 70
      HLPFIL(1)=(HCOMND(ICMD).AND."777777700000) .OR. "41632
      CALL CHKFIL(MASDEV,HELPF,MASPPN,$170)
      GO TO 120
170   OPEN(UNIT=25,DEVICE=MASDEV,ACCESS='SEQIN',FILE='VERSIM.HLP'
      1,DIRECTORY=HLPPPN)
180   READ(25,200,END=320)INF,LEND
200   FORMAT(A10,G)
      IF(DHLPFL.EQ.INF)GO TO 210
      CALL SKPREC(25,LEND)
      GO TO 180
210   FOUND=.TRUE.
      DO 300 J=1,LEND
      READ(25,220)LINE
220   FORMAT(16A5)
      CALL ASCIZ(LINE,16,NDUM)
```

```
        CALL CRLF
300     CONTINUE
C
320     CLOSE(UNIT=25,DISPOSE='SAVE')
        GO TO 120
C
C       CODE FOR CYCLING THROUGH ALL HELP FILES
C
500     CALL CHKFIL(MASDEV,HELPF,MASPPN,$540)
        WRITE(JCT,520)
520     FORMAT(' % HELP FILE NOT AVAILABLE')
        RETURN
540     FOUND=.TRUE.
        OPEN(UNIT=25,DEVICE=MASDEV,ACCESS='SEQIN',FILE='VENSIM.HLP'
        1,DIRECTORY=HLPPPN)
560     READ(25,200,END=120)INP,LEND
        DO 580 J=1,LEND
        READ(25,220)LINE
        CALL ASCIZ(LINE,16,NDUM)
        CALL CRLF
580     CONTINUE
        CALL CRLF
        CALL CRLF
        GO TO 560
        END
```

```
      SUBROUTINE SKPREC(IUN,LENC)
      DO 40 I=1,LENC
      SKIP RECORD IUN
40    CONTINUE
      RETURN
      END
      SUBROUTINE INREAT  ( IRCOND , ENDD , NLSTCP )
C
C     EDIT HISTORY
C
C     [612]/SS-6-23-78        MODIFY DYNAMOMETER HORSE POWER
C     [615]/SS-10-4-78        DIESEL STUFF
C     [617]/SS-11-21-78       DUMP COMMAND TO DUMP PARTS IN FORM FOR READING BACK IN.
C     [621]/SS-5-9-79         ADD MODIFY LOCKUP GEAR IF OVER A
C                             SPECIFIED RPME.
C     [623]/SS-7-9-79         ADD MODIFY TOPSPEED TO GO TO SUMMARY
C                             IF ABS(V-VOLD)<1.E-3 AND ACCEL<1.E-3.
C     [624]/JD                DEGREES OF THROTTLE FOR SHIFT LOGIC
C
      INCLUDE 'COMMS/NOLIST'
C
      LOGICAL LSAVE,NLSTCP,ENDD,ENG1,ENG2,LSIMUL
     2,LSIMDN,IFULL,ISKIP
C
      DIMENSION CARD(16),HPART(14),CNVTYP(2),MFLG(NMOD)
     1,OLDVAL(NMOD),VALNEW(NMOD),HMOD(NMOD),NHMOD(NMOD),NHPART(14)
     2,HLIST(5),NHLIST(5)
C
      INTEGER DEDTAB(7)
C
      EQUIVALENCE (HMOD(1),HTIRE),(HMOD(2),HC1),(HMOD(3),HC2)
     2,(HMOD(4),HCD),(HMOD(5),HREAR),(HMOD(6),HWHEEL),(HMOD(7),HAREA)
     3,(HMOD(8),HWEIGH),(HMOD(9),HSHIFT),(HMOD(10),HSTEP)
     4,(HMOD(11),HTRF),(HMOD(12),HWIND),(HMOD(13),HFUEL)
     5,(HMOD(14),HIDLE),(HMOD(15),HDISPL),(HMOD(16),HSTROK)
     6,(HMOD(17),HCYLIN),(HMOD(18),HUPSHI),(HMOD(19),HDOWNS)
     7,(HMOD(23),HDIFS),(HMOD(24),HLOCKU),(HMOD(25),HTPSPD)
C
      DOUBLE PRECISION CNVNAM(2),HNLOAD,DATE1,HPVEHI
     2,HINPBA,FNAMEO,HPDYNO
C
      DATA NDEB/7/,DEBTAB/'OFF','SHIFT','TIME','SEGME',
     2'ITERA','GETAC','ALL'/
C
      DATA HACCES/'ACCES'/,HAREA/'AREA '/,HAXLE/'AXLE'/,HSINGL/'SINGL'/
     1,HEVEHI/'%VEHICLE'/,HPDYNO/'%DYNO'/
C
      DATA HBMEP/'BMEP'/,HBSFC/'BSFC'/,HINPBA/'INPBAT'/
C
      DATA   HC1   /5HC1    / , HC2   /5HC2    / , HCD   /5HCD    /,
     1        HCOAST/5HCOAST / , HCONVE/5HCONVE / , HCYLIN/5HCYLIN /
C
      DATA   HDATA /5HDATA  / , HDETEN/5HDETEN / , HDIREC/5HDIREC / ,
     1        HDISPL/5HDISPL / , HDOWNS/5HDOWNS / , HDRIVE/5HDRIVE / ,
     2 HDRIVI/'DRIVI'/,CNVTYP/'DRIVE','COAST'/,HNLOAD/'NOT LOADED'/
C
      DATA   HENGIN/5HENGIN / , HFUEL /5HFUEL  / , HON/'ON'/
C
      DATA   HGALHR/5HGAL/H / , HGEAR /5HGEAR  / , HHP   /5HHP    /
C
      DATA   HIDLE /5HIDLE  / , HINERT/5HINERT / , HINITI/5HINITI /
C
      DATA   HLBHR /5HLB/HR / , HLOAD /5HLOAD  / , HLOCKU/5HLOCKU /
C
      DATA   HM    /5HM     / , HMILE /5HMILE  / , HNUMB /5HNUMB  /
      DATA HDIES/'DIESE'/ ,HTESPD/'TOPSP'/
C
      DATA   (HEAFT(I),I=1,NPART) /5HENGIN ,5HCONVE ,5HVEHIC ,5HGEAR
     2,5HACCES , 5HDRIVE ,5HSHIFT ,5HROUTE, 5HTIRE  ,5HTRANS
     1 ,'AXLE'   /
     3,DSTAR/'*      '/
C
      DATA HPARTS/'PARTS'/,HWHEEL/'WHEEL'/
     1,HTRR/'TRR'/,NHMCD/14*2,3,7*2,3,1/,NHPART/14*2/,HMOD(20)/'PHI'/
     2,HMOD(21)/'DYNAM'/,HMOD(22)/'COTYC'/
C
      DATA   HREAR /5HREAR  / , HROUTE/5HROUTE / , HRPM  /5HRPM   /
C
      DATA   HSEC  /5FSEC   / , HSEGME/5HSEGME / , HSHIFT/5HSHIFT /,
     1        HSLASH/5H  /   / , HSPEED/5HSPEED / , HSTAR /5H*     / ,
     2        HSTEP /5HSTEP  / , HSTROK/5HSTROK / , HSUMMA/5HSUMMA /
C
      DATA   ETFROT/5HTEROT / , HTIRE /5HTIRE  / , HTORQU/5HTORQU /
      DATA   ETIME /5HTIME  /, HCONTI/'CONTI'/
C
      DATA HDTHEO/5HDTHEO /                                              ! [624]
C
      DATA   EUPSHI/5HUPSHI / , HVACUU/5HVACUU / , HVEHIC/5HVEHIC /
C
      DATA   HWEIGH/5HWEIGH / , HWIND /5HWIND  /,HMILEP/'MILEP'/
C
      DATA HEATCH/'EATCH'/,HTRUCK/'TRUCK'/,HCAR/'CAR'/,HBUS/'BUS'/
C
      DATA HLIMT/'SEGME','MILE','SECCN','OFF','SUMMA'/
      DATA NHLIMT/3,1,3,1,3/
C
C*********************************************************************
C
C    INITIALIZE LOGICAL FLAGS FOR PRINT LIMITATION AND END OF INPUT
C    DATA CN FIRST PASS THROUGH
C
20      ISTART=IECOND                                                    !GET CTRFIL STATUS.
        LSKIP=.FALSE.
        ISKIP=0
        IECCND=0                                                         !ASSUME NO ERR.
        ENDE=.FALSE.                                                     !ASSUME NO EOF ON CTRFIL.
        NXTPG=1                                                          !INTIAL PAGE CNT.
        GO TO (40,80,120),ISTART                                         !(NEW CTRFIL/REWIND CTRFIL&EXE/CONT EXE CTRFIL ?).
C
C     COPY CONTROL FILE CNTO LPT FILE
C
40      IF(LLSTCF) GO TC 100                                             !(LIST CTRFIL ON LPT?) NO.
        WRITE(6,4980)                                                    !HEADER TO LPT.
60      READ(4,5000,END=90) CARD                                         !READ CTR REC(EOF?) YES.
        WRITE(6,5040)  (CARD(I),I=1,NNBCNT(CARD,16))                     !NO, WRITE REC.
        GO TO 60                                                         !NEXT.
C
80      REWIND 4                                                         !POSI FOR EXE.
100     LSIMCM=.FALSE.                                                   !INTI FLG FOR /*SIM  DIALO/ CARD.
        GO TO 180                                                        !GO GET COMMAND.
120     IF(.NOT.LSIMDM) GO TO 140                                        !(CONTI FROM /*SIM  DIALO/& SIM NOT DONE IN DIALO?) NO.
        LSIMDE=.FALSE.                                                   !YES, RESET FLG.
        GO TO 540                                                        !GO SIM.
C
```

```
140       NSKIP=0                                                   !ERROR RETURN FROM SUBROUTINE VALID2
          LSKIP=.TRUE.                                              !SKIP TO NEXT COMMAND CARD
          GO TO 180                                                 !GO READ A CARD
C
160       IF(DIALCG) RETURN                                         !(DIALO MODE?) YES.
          IF(ENDF) GO TO 4940                                       !NO, (EOF CTRFIL?) YES.
180       READ  (4,5060,END=4900)   COL1,COMND,(CARD(I),I=1,15)     !NO, READ NEXT CARD FROM CONTROL FILE(EOF?) YES.
          IF(.NOT.LSKIP)GC TO 200                                   !ARE WE SKIPPING?
          IF(CCL1.NE.ASTAR) GO TO 180                               !GOT A CCNTROL CARD?
          ISKIP=ISKIP+1                                             !YES. INCREASE COUNTER
          IF(ISKIP.LE.NSKIP)GO TO 180                               !HAVE WE SKIPPED ENOUGH?
          LSKIP=.FALSE.                                             !YES
200       WRITE(5,5020)  CCL1,COMND,(CARD(K),K=1,NWRCNT(CARD,15))   !WRITE CTR REC ON TTY.
C
C       DETERMINE COMMAND CN COMMAND CARD (* IN COLUMN 1)
C
220       BACKSPACE 4                                               !POSI CTRFIL.
          IF(COL1.NE.ASTAR) GO TO 260                               !(IS REC COMD?) NO.
          LCMD=ICMD                                                 !YES, SAVE COMD CODE OF LAST COMD.
          CALL LCCKUP(CCMND,5,HCOMND,DUMMY,41,ICMD,DBLANK,$340)     !(KNOWN COMD?) YES.
C
C       PRINT COMMAND INVALID MESSAGE AND GO TO NEXT COMMAND
C
240       IF(DIALCG) RETURN                                         !NO, (DIALO MODE?) YES.
260       IF(CCL1.NE.'!')GO TO 270                                  !(COMMENT?) NO
          SKIP RECORD 4                                             !SKIP COMMENT
          GO TO 160                                                 !GO PROC NEXT CARD
C
270       READ(4,5000) CARD                                         !READ REC IN ERR.
          I=NWRCNT(CARD,16)                                         !CNT WRDS IN REC.
          WRITE(6,5080)  (CARD(K),K=1,I)                            !PRI BAD REC ON LPT.
280       IF(TTY) GC TO 320                                         !(JCT IS TTY?) YES.
          NORUN=NCRCN+1                                             !NO, INC FATAL ERR CNT.
          GO TO 160                                                 !NEXT REC.
C
C       BACKSPACE FOR USE BY INPUT ROUTINE WHEN OTHER ERRORS DETECTED
C
300       IF(DIALCG) RETURN                                         !(DIALOG MODE?) YES.
          BACKSPACE 4                                               !POSI CTRFIL TO BAD REC.
          GO TO 260                                                 !GO PRI IT.
C
320       IF(DIALCG) RETURN                                         !(DIALO MODE?) YES.
          WRITE(5,5040)  (CARD(K),K=1,I)                            !PRI ON TTY BAD REC.
          WRITE(5,5100)  HCCMND(ICMD)                               !"%CTR ERR SWITCH TO DIALO MODE".
          GO TO 380                                                 !GO TO USER FOR HELP.
C
C       GO TC PROCESS COMMAND
C
C           CCMMAND          LABEL           CCMMAND          LABEL
C           -------          -----           -------          -----
C
C          ACCESSCRY         2760           REMAP             3760
C          AXLE              2400
C          DEBUG              400           ROUTE             2960
C          DRIVING SCHEDULE  2860           SHIFT LOGIC       2780
C                                           SIMULATE           500
C                                           SPLIT              495
C          DUMP              1300           S.R. CONVERTER    3140
C          ENGINE            3320           STATUS            4620
C          FULL CCNVERTER    3120           TIRE              3900
C          GEAR              2700           TITLE              480
```

```
C                                              TRANSMISSION       2320
C        LIMIT PRINT        2100              UNLOCK CONVERTER2100
C        LOCKUP CONVERTER 2100                USE                4000
C        MODIFY             1400              VEHICLE            2220
C        PRINT UNITS        1260              ZERO                360
C
340     GO TO (4000,500,1400,1260,1340,1300,3760,2100,2100,480,
        1       240,4620,560, 240, 240, 380,3320,3120,3140,2220,
        2       2700,2760,2780,2660,2960,1980, 400,4900, 240, 360,
        3       1200,1200, 240, 240, 240, 240, 240, 440,2660, 2400,
        4        455) , ICMD
C
360     WRITE(5,5120)                                               !STORE ENCOUNTERED IN CTRFIL.
        IF(PTY) CALL EXIT                                           !(JCT IS PTY?)YES, *BYE RUN COMPLETED.
C
380     BATCH=.FALSE.                                               !SWITCH TO DIALOG MODE.
        DIALOG=.TRUE.
        GO TO 4920
C
C**********************************************************************
C
C     /DEBUG/ COMMAND - LOAD DEBUG PRINTOUT PARAMETERS
C
        ENTRY DEBCMD(IECCMD)                                        !DIALOG ENTRY.
        DBEGIN=DATA(1)                                              !GET DATA.
        DSTOP=DATA(2)                                               !GET DATA.
        GO TO 420
C
400     READ  (4,5140)  WORD,DBEGIN,DSTOP                           !READ DEBUG COMMAND DATA.
420     IECCMD=2                                                    !ASSUME ILLEGAL DEBUG COMMAND.
        IDEBGC=IDEBUG
        CALL SLCOMP(WORD,NDEB,DEBTAD,IDEBUG,$430)                   !LOOKUP DEBUG COMMAND.
          GO TO 300                                                 !(VALID DEBUG COMMAND?)NO.
430     IECCMD=3                                                    !ASSUME BAD DATA.
        IF (DBEGIN.NE.0..AND.DSTOP.NE.0..AND.DBEGIN.GT.DSTOP)  GO TO 300     !BAD DATA?)YES.
        ISEG1  = DBEGIN + .001                                      !CALC DEBUG START PT.
        ISEG2  = DSTOP  + .001                                      !CALC STOP PT.
        IECCMD=1                                                    !SET NO ERR FLG.
        SDEBUG=DFBTAB(IDEBUG).AND.-1                                !SAVE DEBUG COMMAND FOR ?*STATUS/.
        GO TO 160                                                   !DONE.
C
C**********************************************************************
C
C     /OUTPUT/ COMMAND - GET NEW OUTPUT FILE SPECS
C
        ENTRY OUTCMD                                                ![616]
440     ITASK=-1
        CALL CLSLFT(ITASK)                                          ![616]
        IF(BATCH)GO TO 442                                          ![616]
        WRITE(5,10400)                                              ![616]
        READ(5,5000)FSPECS                                          ![616]
        GO TO 444                                                   ![616]
C
442     READ(4,5160)FSPECS                                          ![616]
C
444     CALL FILSPC(FSPECS,LPTDEV,LPTFIL,LPTPPN)                    ![616]
        ITASK=-1
        CALL CPBLFT(ITASK)                                          ![616]
        GO TO 160                                                   ![616]
C
C**********************************************************************
```

```
C
C     /TITLE/ COMMAND - READ RUN TITLE AND DATE
C
       ENTRY TTLCMD(IECCMD)                                                  !DIALOG ENTRY.
       READ(5,5180,END=470) TITLE,DATE1                                      !GET TITLE FROM JCT.
       IECCMD=1                                                              !SET NO ERR FLG.
       GO TO 160                                                             !DONE.
C
470    IECCMD=2
       GO TO 300
C
480    READ  (4,5160)  TITLE,DATE1                                           !READ TITLE + OVERRIDE DATE.
490    IF ( DATE1.NE.HBLANK) DATE  = DATE1                                   !(OVERRIDE DATE?) YES.
       GO TO 160
C
C**************************************************************************
C
C      /SPLIT/COMMAND
C
495    CONTINUE
       READ(4,5021) SPLIT
       PIOD=.TRUE.
       GO TO 160
C
C**************************************************************************
C
C     /SIMULATE/ COMMAND - CHECK TO SEE THAT ALL PARTS REQUIRED ARE
C                          DEFINED , RESCALE ENGINE IF REQUIRED AND
C                          RETURN TO MAKE A RUN
       ENTRY SIMCMD ( IECOND )                                               !DIALOG ENTRY.
       LSIMDM=.FALSE.                                                        !FLG SIM DONE FROM DIALOG MODE.
       IECCMD=3                                                              !SET ERR FLG.
       GO TO 540
C
500    READ  (4,5260)  DATA(1),DATA(2)                                       !READ SIMULATE CARD.
       IF(DATA(1).EQ.HBLANK) GO TO 520                                       !SIMODE FIELD BLANK?) YES.
       IF(DATA(1).EQ.HCAR) GO TO 510                                         !CAR MODE?
       WRITE(JCT,5290) DATA (1)                                              !NO. REPORT, BUT GO ON ANYWAY
510    SIMCDE=DATA(1)                                                        !YES, SET MODE.
520    IF(ETY.OR.(DATA(2).EQ.HBLANK).OR.(DATA(2).EQ.HCONTI)
       1 .OR.(DATA(2).EQ.DATCH))  GO TO 540                                  !(EXECUTE ?*SIMULATE/?) YES.
       IF(DATA(2).NE.HCOMND(34)) GO TO 300                                   !NO,(DIALOG COMMAND?) NO ERR.
       LSIMDM=.TRUE.                                                         !YES, FLG ?*SIMULATE/ NOT DONE BECAUSE TRANSFER TO
       GO TO 380                                                             !DIALOG MODE.
C
540    IF ( NOFUN.GT.0 )  GO TO 940                                          !(UNRESOLVED ERRORS EXIST?) YES.
       IERROR = 0                                                            !NO,ZERO CNT OF PART NOT LOADED ERRORS.
       DO 880  I = 1,NUMPAR                                                  !LOOP THROUGH ALL PART TYPES.
       IDATA(I)=0                                                            !ZERO FLG TO IMPDIA.
       IF ( NPARTS(I).GT.0 )  GO TO 580                                      !(PARTS LOADED?) YES.
       GO TO(560,660,560,560,860,560,820,560,840,880,860),I                  !BRANCH TO APPROPRIATE PART MISSING ERROR HANDLER.
C
C     WRITE *PART MISSING* ERROR MESSAGE
C
560    IERROR = IERROR + 1                                                   !INC ER CNT.
       IF ( IERROR.EQ.1 )  WRITE (6,5200)                                    !PART MISSING HEADER.
       WRITE (6,5220)  HEAPT(I)                                              !REPORT PART MISSING.
       IDATA(I)=1                                                            !FLGPART TYPE FOR IMPDIA.
       GO TO 880                                                             !NEXT.
C
580    GO TO(600,640,680,880,880,700,880,800,880,880,880),I                  !ON PART LOADED SPECIAL HANDLING BRANCH, IF ANY.
```

```
C
C     CHECK THAT NUME OF ENGINES MATCHES GEAR REQUIREMENTS
C
600       NUME   = 1                                                     !ASSUME ONE ENG.
          NUMEG  = NPARTS(4)                                             !GET # OF GEARS.
          IF ( NUMEG.LT.1 )   GO TO 880
          DO 620  J = 1,NUMEG
          IF ( JENG(J).EQ.2 )   NUME = 2
620       CONTINUE
          IF ( NPARTS(1).NE.NUME )   GO TO 560                           !(NUMBER OF ENGS LOADED = NUMBER OF ENGS NEEDED?)NO.
          GO TO 880                                                      !YES NEXT.
C
C     CHECK THAT 2 CONVERTERS ARE DEFINED
C
640       IF(NPARTS(2).GE.2) GO TO 880                                   !(2 CONVS LOADED?)YES, NEXT.
660       DO 680 K=1,NUMEG                                               !NO LOOP THRO ALL GEARS.
          IF(.NOT.LOCKUP(K)) GO TO 560                                   !(GEAR LOCKED UP?)NO ERROR.
680       CONTINUE                                                       !YES, NEXT.
          WRITE(5,5320)                                                  !ALL GEARS LOCKED UP.
          ITERR=1
C
C  LOAD STANDARD COAST CONVERTER FOR MANUAL TRANSMISSION
          IPRNT=102                                                      !SET DSK FLG TO /*USE/ CONV.
          CNAME='COCV2 '                                                 !SET DEFAULT COAST CONVERTER NAME.
          CALL DSK                                                       !GET IT.
          IF(IPRNT.EQ.-10) ITERR=2                                       !(ERR?)YES, FLG.
          GO TO (700,760),ITERR                                          !(GOT IT/ERR MESS?)
700       CALL ERROUT                                                    !PRINT IT.
          NPARTS(2)=1                                                    !FLG OUT ONE CONV.
C
C  LOAD STANDARD DRIVE CONVERTER FOR MANUAL TRANSMISSION
720       IPRNT=102                                                      !SET DSK FLG TO /*USE/ CONV.
          CNAME='CODV2 '                                                 !SET DEFAULT DRIVE CONV NAME.
          CALL DSK                                                       !GET IT.
          IF(IPRNT.EQ.-10) GO TO 740                                     !(ERR?)YES.
          CALL ERROUT                                                    !PRINT IT.
          IF(ITERR.EQ.2) GO TO 560                                       !(ERR ON DRIVE CONV /USE/?)YES.
          NPARTS(2)=2                                                    !NO,FLG 2 CONVS LOADED.
          CNAME='STANDARD'                                               !SET CONV NAME.
          GO TO 980                                                      !NEXT.
C
C  ERROR - CAN'T FIND MANUAL COVERTER FOR MANUAL TRANSMISSION
740       ITERR=3
760       CALL NOPART(5,CNAME,2,CNVTYP(ITERR-1))
          GO TO (760,720,560),ITERR                           !(IMPOSS SO PUT INTO INF LOOP/COAST CONV ERR TRY DRIVE/TRIED BOTH-NEXT?)
C
780       IF(NDSEG.EQ.0) GO TO 880                                       !  (1ST SECT OF DRS LOADED?) YES, CONT
          IPRNT=106                                                      !  NO, SET DSK FLG TO /USE/ DRS
          PNAME=DNAME                                                    !GET DRS NAME.
          CALL VALID2(PNAME,$140)                                        !VSLID NAME?
          CALL DSK                                                       !  GO GET 1ST SEC OF DRS
          IF(IPRNT.EQ.6) GO TO 880                                       !  (GOT IT?) YES, CONT.
          WRITE(5,5240) HPART(6),DNAME                                   !NO, REPORT DSK ERR ON RELOAD
          GO TO 920
C
800       IF(NORTE.EQ.0.OR.LDYNA) GO TO 880                              !(1ST SECT OF RTE LOADED?) YES, CONT
          IPRNT=108                                                      !NO,SET DSK FLG /USE/ RTE
          PNAME=RNAME                                                    !GET RTE NAME.
          CALL VALID2(PNAME,$140)                                        !VSLID NAME?
          CALL DSK                                                       !GO GET 1ST SECT OF RTE
          IF(IPRNT.EQ.8) GO TO 980                                       !(GOT IT?) YES, CONT
```

```
      WRITE(5,5240) HPART(8),FNAME                        !NO, GO REPORT ERR
      GO TO 920                                           !GO ERR BYE
820   IF(LDYNA) GO TO 880
      GO TO 560
C
840   IF(LVNEW.OR.NPARTS(3).EQ.0) GO TO 560               ! (TIRE DATA DEFINED?) NO.
      TNAME=HEVEHI
      NPARTS(9)=1                                         ! YES, SET TIRE LOADED FLG.
C
860   IF(LVEHAX)GO TO 880                                 !(AXLE TIED TO VEHICLE?)YES, IGNORE ERROR.
      GO TO 560                                           !NO, REPORT ERROR
880   CONTINUE
C
      IF ( IERRCR.GT.0 )  GO TO 160                       !(ALL PARTS LOADED?) NO.
C
      CALL DSKCTR(0,'INEBAT')                             ! YES, RELEASE ALL DB FILES
C
      LSIMUL=.TRUE.
      GO TO 9640                                          ! DO A VEHSIM STATUS REPORT TO LPT FILE
900   LSIMUL=.FALSE.
C
      WRITE(5,5300)                                       ! "[SIMULATING]"
C
      CALL SIMCTR ( IECCND , SIMODE )                     ! GO SIMULATE
C
      IF(IECOND.EQ.0) IECCND=1                            ! SET INPDIA RETURN FLG
      NXTPG=1                                             ! RESET NEXT PAGE NUMBER
C
      GO TO 4920                                          ! * DONE , BYE
C
C     WRITE MESSAGE AND SKIP SIMULATE IF PREVIOUS CONTROL CARD ERRORS
C
920   NORUN=NORUN+1                                       ! INC FATAL ERR CNT
940   WRITE (6,5260) NORUN                                ! REPORT ERR.
      IDATA(1)=NORUN                                      ! PASS TO INPDIA.
      IECCND=3                                            ! SET INPDIA ERR FLG
      GO TO 160                                           ! BYE.
C
C*****************************************************************************
C
C     /ZERO/ COMMAND - RESET ALL PROGRAM VARIABLES TO INITIAL STATE
C
      ENTRY ZERCMD(IECCND)                                !DIALOG ENTRY.
      IECCND=1                                            !FLG DIALOG ENTRY.
      GO TO 980
C
960   READ  (4,5540)  UB,UT                               !READ BATCH COMMAND CARD.
      IECCND=0                                            !FLG BATCH ENTRY.
980   IF(UT.EQ.HPART(5)) GO TO 1040                       !(ZERO ACCESSORY?)YES.
      CALL ZERO                                           !NO, ZERO ALL SUD ZERO FOR GLOBAL DATA.
      LSIMUL=.FALSE.                                      !FLG TO ?*STATUS/ THAT /*SIMULATE/ DID CALL.
      LSIMDM=.FALSE.                         !FLG THAT ?*SIMULATE/ SWITCHED TO DIALOG. RESET IF SIM DONE FROM INPDIA.
      LDYNA=.FALSE.
      LDYNOV=.FALSE.
      LDIFS=.FALSE.
      LMDICK=.FALSE.                                      ![621]
      ENG1=.FALSE.                                        !ENG 1 NOT LOADED.
      ENG2=.FALSE.                                        !ENG 2 NOT LOADED.
      CNVNAM(1)=HNLCAD                                    !DRIVE CONV NAME.
      CNVNAM(2)=HNICAD                                    !COAST CONV NAME.
      RNAMEC=HNLOAD                                       !SAVE LCC FOR "%DYNO" OF ROUTE NAME.
```

```
        NPARTB=0                                             !SAVE LOC FOR NPARTS(8) WHEN "ADYANO".
        SDEBUG=NOFF                                          !DEBUG DEFAULT COMMAND.
        SLIMIT=NSUMMA                                        !LIMIT PRINT DEFAULT COMMAND.
        DO 1000 I=1,NMOD                                     !LOOP THRU ALL MODIFY FLGS.
1000    MFLG(I)=0                                            !FLG NO MODS.
1020    IF(IFCOND.EQ.1) RETURN                               !(ZERCMD ENTRY?) YES.
        GO TO 160                                            !NO.
C
C   ZERO LOADED ACCESSORY FROM CORE
C       BY OVER WRITING IT WITH THE ACC(NACC) AND DECREMENTING NACC BY 1
C
1040    WRITE(5,5520)                                        ! REPORT ZERO ACC.
        WRITE(6,5520)                                        ! REPORT ZERO ACC.
        IF(NACC.LE.0) GO TO 1100                             ! (ANY ACC'S LOADED?) NO.
        IF(UN.NE.ESTAR) GO TO 1060                           ! (ALL ACC'S?) NO, GO SEARCH.
        IB=1                                                 ! SET UP TO REPORT ACC'S ZEROED
        IE=NACC
        NACC=0                                               ! SET NO ACC'S LOADED
        GO TO 1180                                           ! GO REPORT.
C
1060    DO 1080 I=1,NACC                                     ! LOOK FOR ACC TO ZERO.
        IF(UN.EQ.ANAME(I)) GO TO 1120                        ! (GOT IT?) YES.
1080    CONTINUE                                             ! NO, NEXT.
        WRITE(5,5480) UN                                     ! % ACC NOT LOADED
        WRITE(6,5480) UN                                     ! % ACC NOT LOADED
        GO TO 1020                                           ! BYE.
1100    WRITE(5,5500)                                        ! % NO ACC'S LOADED
        WRITE(6,5500)                                        ! % NO ACC'S LOADED
        GO TO 1020                                           ! BYE.
C
1120    IF(NACC.EQ.1) GO TO 1160                             ! (ONLY ONE ACC LOADED?) YES.
        NNA(I)=NNA(NACC)                                     ! NO, MOVE LAST ACC LOADED DOWN ON TOP OF ACC TO ZERO.
        APRCT(I)=APROT(NACC)                                 !MOVE PROTECTION.
        APPN(I)=AEPN(NACC)                                   !MOVE PPN.
        DO 1140 K=1,NNA(NACC)                                !LOOP THRU NNA POINTS.
        ACCS(K,I)=ACCS(K,NACC)                               !MOVE RPM DATA.
1140    ACCT(K,I)=ACCT(K,NACC)                               !MOVE TORQUE DATA.
1160    IB=I                                                 ! SET UP TO REPORT ACC ZEROED
        IE=I
        NACC=NACC-1                                          ! ONE LESS ACC
1180    WRITE(5,5980) (K,ANAME(K),K=IB,IE)                   ! ACC # & NAME
        WRITE(6,5980) (K,ANAME(K),K=IB,IE)                   ! ACC # & NAME
        ANAME(IE)=ANAME(NACC+1)                              !MOVE ACC NAME.
        GO TO 1020                                           ! *DONE
C
C***********************************************************************
C
C      /DELETE/ COMMAND - DROP PART FROM PARTS DATA FILE
C
        ENTRY DRPCMD(IECCMD)                                 !DIALOG ENTRY.
        IECCMD=2                                             !ASSUME ERROR.
        NC=ECRCNT(UT,1)                                      !GET # OF CHARS IN PART TYPE.
        GO TO 1220
C
1200    READ  (4,5560)  UN,UT,UE                             !READ COMMAND CARD FROM CTRFIL.
        NC=5                                                 !MUST BE 5 CHARS.
1220    CALL LOCKUP(UT,-NC,HPART,NHPART,NUMPAR,IPRNT,DBLANK,$1240)  !LOCK UP PART TYPE.
        IECCMD=1                                             !SET UNKNOWN PART TYPE ERROR FLG.
        GO TO 300                                            !GO REPORT.
C
1240    UT=HPART(IPRNT)                                      !GET COMPLETE PART TYPE WORD FROM TABLE.
```

```
      CALL CSRDEL                                                        !GO DELETE IT.
      IF ( IPRNT .EQ. -10 )  GO TO  280                                  !(ERR?)YES, FLG IT.
      IECCND=1                                                           !NO, FLG NO ERR.
      GO TO 160                                                          !NEXT.
C
C*********************************************************************
C
C     /PRINT UNITS/ COMMAND - READ UNITS FOR ENGINE MAP PRINTOUT AND
C                             SET FLAGS IF DIFFERENT FROM ENGINE
C                             INPUT DATA UNITS
C
      ENTRY UNTCMD(IECCND)                                               !DIALOG ENTRY.
      IECCND=2                                                           !SET ERROR UNKNOWN UNITS.
      GO TO 1280
C
1260  READ  (4,5280)  WORD,UNITS                                         !READ COMMAND CARD FROM CTR FIL.
      IF ( WORD.NE.HEBGIN   )  GO TO 300                                 !(UNITS FOR ENG?)NO,ERR.
1290  IF ( UNITS(1).NE.HRPM .AND.UNITS(1).NE.HPISTO )  GO TO 300         !(ENG SPD UNITS VALID?)NO.
      IF ( UNITS(2).NE.HBMEP.AND.UNITS(2).NE.HTORQU.AND.
     1 UNITS(2).NE.HHP     )      GO TO 300                              !(ENG LOAD UNITS VALID?)NO.
      IF ( UNITS(3).NE.HLBHR.AND.UNITS(3).NE.HBSFC
     1 .AND.UNITS(3).NE.HGALHR )      GO TO 300                          !(ENG FUEL RATE UNITS VALID?)NO.
      PRPM    = .FALSE.                                                  !ALL UNITS VALID, TURN ALL UNITS OFF.
      PRPM    = .FALSE.
      PPS     = .FALSE.
      PTOR    = .FALSE.
      PBMEP   = .FALSE.
      PHP     = .FALSE.
      PLBHR   = .FALSE.
      PBSFC   = .FALSE.
      PGALHR  = .FALSE.
      IF ( UNITS(1).EQ.HPISTO )  PPS     = .TRUE.                        !TURN NEW UNITS ON.
      IF ( UNITS(1).EQ.HRPM   )  PRPM    = .TRUE.
      IF ( UNITS(2).EQ.HBMEP  )  PBMEP   = .TRUE.
      IF ( UNITS(2).EQ.HHP    )  PHP     = .TRUE.
      IF ( UNITS(2).EQ.HTORQU )  PTOR    = .TRUE.
      IF ( UNITS(3).EQ.HBSFC  )  PBSFC   = .TRUE.
      IF ( UNITS(3).EQ.HGALHR )  PGALHR  = .TRUE.
      IF ( UNITS(3).EQ.HLBHR  )  PLBHR   = .TRUE.
      IECCND=1                                                           !SET NO ERROR FLG.
      GO TO 160
C
C*********************************************************************
C
C     /COMP/ COMMAND - PRINT ALL PARTS DATA STORED ON PARTS DATA FILE
C                      AND/OR A DIRECTORY OF ALL THESE PARTS
C
      ENTRY CMPCMD(IECCND)                                               !DIALOG ENTRY.
      IPRNT=IECCND                                                       !GET PART TYPE PTR.
      IECCND=4                                                           !SET ERR FLG.
      GO TO 1320
C
1300  READ(4,5540) UN,UT
      IF(UT.NE.HDIREC .AND. UT.NE.HPARTS) GO TO 1320                     ! (ALL PART TYPES?) NO.
      IPRNT=-1                                                           ! ASSUME DIR ONLY.
      IF(UT.EQ.HPARTS) IPRNT=0                                           ! (DIR ONLY?) YES, SET FLG.
      UN=HSTAR                                                           !SET PART NAME WILD.
      UT=HSTAR                                                           !SET PART TYPE WILD.
C
1320  LSAVE=LINERN                                                       !SAVE.
      LINERN=.FALSE.                                                     ! PRINT ON.
```

```
C
      CALL DSKDIR                                                        ! EXECUTE DUMP/DIR REQUEST.
C
      LIMERN=LSAVE                                                       ! RESTORE.
      IECCND=1                                                           ! ASSUME NO ERR.
      IF(IPRNT.EQ.-10) IECOND=1                                          ! (ERR?) YES, SET ERR FLG.
      GO TO 160                                                          ! NO, *DONE.
C
C**********************************************************************.
C
C     /LIMIT PRINT/ COMMAND - SET RUN PRINT LIMIT PARAMETERS
C
      ENTRY LIMCMD(IECCND)                                               !DIALOG ENTRY.
      ALIMN=DATA(1)                                                      !GET PRINT OUT INC VAL.
      IECCND=2                                                           !ASSUME ERROR.
      GO TO 1360
C
1340  READ  (4,5340)  WORD,ALIMN                                         !READ COMMAND CARD FROM CTR FILE.
1360  NC=ICRCNT(WORD,1)
      CALL LOOKUP(WORD,-NC,HLIMT,NHLIMT,5,NARG,'LIMIT',$1370)            !(VALID LIMIT PRINT?)YES, .
      GO TO 300
1370  LIMERN = .TRUE.                                                    !SET DEFAULT.
      MILIM  = .FALSE.
      SECLIM = .FALSE.
      ENDLIM = .FALSE.
      IF ( NARG.EQ.1 )  ENDLIM = .TRUE.
      IF ( NARG.EQ.2  )  MILIM  = .TRUE.                                 !SET TO NEW STATUS.
      IF ( NARG.EQ.3   )  SECLIM = .TRUE.
      IF ( NARG.EQ.4   )  LIMERN = .FALSE.
      IF ( ALIMN.GT.1.E-10)   GO TO 1380                                 !(INC VALUE SPECIFIED?)YES.
      IF ( MILIM           )   ALIMN  = .1                               !NO,(PRINT OUT BY DIST?)YES, SET TO DEFAULT.
      IF ( SECLIM          )   ALIMN  = 10.                              !(PRINT OUT BY TIME?)YES, SET TO DEFAULT.
1380  IECCND=1                                                           !FLG NO ERR.
      SLIMIT=HLIMT(NARG)                                                 !SAVE LIMIT STATUS.
      GO TO 160                                                          !NEXT.
C
C**********************************************************************
C
C     /MODIFY/ COMMAND - CHANGE NAMED COMPONENT TO SPECIFIED VALUE
C
      ENTRY MODCMD(IECCND)                                               !DIALOG ENTRY.
      IECCND=1
      VALUE=DATA(1)
      PNAME=HINEBA                                                       !SET ROUTINE NAME FOR ERR MESS FROM LOOKUP.
      NC=ICRCNT(WORD,1)                                                  !COUNT CHARS IN VARIABLE TO BE MODIFIED.
      GO TO 1420
C
1400  READ  (4,5342)  WORD,VALUE     ,VALUE2
      PNAME=CBLANK                                                       !FLG LOOKUP NO ERR MESS.
      NC=5
1420  CALL LOOKUP(WORD,-NC,HMCD,NHMCD,NMCD,IMOD,CBLANK,$1440)            !(VALID MODIFY?)YES.
      IECCND=2                                                           ! ? UNKNOWN MODIFY COMMAND
      GO TO 300                                                          !ER.
1440  GO TO(1780,1500,1520,1540,1600,1620,1480,1000,1460,1560
     2,1640,1820,1580,1840,1860,1860,1860,1660,1680,1720
     3,1740,2080,2090,20900,20940),IMOD                                  !GO TO SAVE OLD VALUE, SET NEW VALUE, AND CHECK IF LEGAL.

C
1460  OLDVAL(9)=ISMCDE                                           ! MODIFY SHIFT NOT IN DOCUMENTATION- INTENDED FOR SUPPORT DEBUGGING
      ISMCDE = VALUE + .001
      IF(ISMCDE.LT.1 .OR. ISMCDE.GT.2)  GO TO 2040
```

```
         GO TO 1940
 1480    OLDVAL(7)=AREA
         AREA   = VALUE
         GO TO 1980
 1500    OLDVAL(2)=FRC1
         FRC1   = VALUE
         GO TO 1960
 1520    OLDVAL(3)=FRC2
         FRC2   = VALUE
         GO TO 1960
 1540    OLDVAL(4)=CD
         CD     = VALUE
         GO TO 1980
 1560    OLDVAL(10)=DEFECT
         DEFECT = VALUE
         GO TO 1940
 1580    OLDVAL(13)=FSPGR
         FSPGR  = VALUE
         GO TO 2000
 1600    OLDVAL(5)=RAR
         RAR    = VALUE
         GO TO 1980
 1620    OLDVAL(6)=WLSG
         WLSG=VALUE
         GO TO 1940
 1640    OLDVAL(11)=LTRRZ
         LTRRZ=.FALSE.
         IF(VALUE.GT..0) LTRRZ=.TRUE.
         GO TO 1940
 1660    OLDVAL(18)=PCT1
         PCT1=VALUE
         MODE   = 1
         GO TO 1700
 1680    OLDVAL(19)=PCT2
         PCT2=VALUE
         MODE   = 2
 1700    IF ( NPARTS(7).LT.1 )   GO TO 2020                       !(SHIFT LOGIC LOADED?)NO, ERROR.
C.......................................
         CALL MODSL ( MODE,VALUE )                                !YES, MODIFY IT.
         GO TO 1940
 1720    OLDVAL(20)=PHI
         PHI=VALUE
         PHIC=VALUE
         GO TO 1940
 1740    OLDVAL(21)=LDYNA
         IF(VALUE.EQ.0.) GC TO 1760
         LDYNA=.TRUE.
         RNAMEO=RNAME
         NPARTO=NPARTS(8)
         PHIC=PHI
         RNAME=HEDYNC
         NPARTS(8)=1
         LDYNCV=.FALSE.![612]
         IF(VALUE.GT.0)GC TO 1940![612]ONLY OVERIDE IF NEG VALUE.
         DYN=-VALUE![612]ASSIGN VALUE
         LDYNCV=.TRUE.![612]SET FLAG
         GO TO 1940
 1760    LDYNA=.FALSE.
         LDYNCV=.FALSE.                                           ![612]
         RNAME=RNAMEO
         NPARTS(8)=NPARTO
```

```
      PHI=PHIC
      GO TO 1940
1780  OLDVAL(1)=TIREFF
      TIREFF = VALUE
      GO TO 1960
1800  OLDVAL(8)=WGT
      WGT    = VALUE
      GO TO 1980
1820  OLDVAL(12)=VWIND
      VWIND  = VALUE
      GO TO 1940
1840  OLDVAL(14)=RPMIN(1)
      ENMIN(1) = VALUE
      ENMIN(2) = VALUE
      RPMIN(1) = VALUE
      RPMIN(2) = VALUE
      GO TO 2000
C
C     ENGINE PARAMETER MODIFICATION
C
1860  IF ( NPARTS(1) .LT.1 )  GO TO 2020                         ! (ENGINE LOADED?) NO,ERR.
      ODISP  = DISP                                              !SAVE CURRENT VALUES.
      OBORE  = BORE
      OSTROK = STROKE
      IOCYL  = ICYL
      GO TO (1880,1900),(IMOD-15)                                !MODIFY (STROKE/CYLINDER?)
      OLDVAL(15)=DISP                                            !FALL HERE FOR MODIFY DISPLACEMENT.
      DISE   = VALUE
      BORE   = SQRT ( 4*(DISP/ICYL)/(3.1415927*STROKE) )
      GO TO 1920
1880  OLDVAL(16)=STROKE
      STROKE = VALUE
      DISP   = 3.1415927 * ((BORE/2.)**2.) * STROKE * ICYL
      GO TO 1920
1900  OLDVAL(17)=ICYL
      ICYL   = VALUE + .001
      DISP   = (ODISP/ICCYL)*ICYL
C
1920  CALL SCALEN                                                !SCALE ENGINE.
C
1940  MFLG(IMCD)=1                                               ! SET MODIFIED VALUE FLG.
      VALNEW(IMCD)=VALUE                                         ! SAVE ,NEW VALUE.
      GO TO 160                                                  ! *DONE.
C
1960  IF(NPARTS(9).LT.1.AND.LVNEW) GC'TO 2020                    ! (TIRE LOADED?) NO, ERR.
      GO TO 1940                                                 ! YES, ALLOW MODIFICATION.
C
1980  IF ( NPARTS(3).LT.1 )  GO TO 2020                          ! (VEHICLE LCADED?) NO, ERR.
      GO TO 1940                                                 ! YES, ALLOW MODIFICATION.
C
2000  IF ( NPARTS(1).LT.1 )  GO TO 2020                          ! (ENGINE LCADED?) NO, ERR.
      GO TO 1940                                                 ! YES, ALLOW MODIFIFCATION.
C
2020  IERCMD=3                                                   ! ? PART TO MODIFY NOT LOADED.
      GO TO 2060
C
2040  IERCMD=4                                                   ! ? ILLEGAL MODIFY VALUE.
2060  WORD=MMCD(IMOD)                                            !SET WRD TO UNABREVIATED FORM.
      GO TO 300
C
2080  OLDVAL(22)=DUTCYC                                          !SAVE OLD ACCESSORY DUTY CYCLE.
```

```
      DUTCYC=VALUE                                                  !SET NEW DUTYCYCLE.
      GO TO 1940
C
2090  OLDVAL(23)=LDIES                                              ![615]
      LDIES=.FALSE.                                                 ![615]
      IF(VALUE.NE.0)LDIES=.TRUE.                                    ![615]

      GO TO 1940
C
20900 OLDVAL(24)=LMDLCK                                             ![621]
      LMDLOK=.FALSE.                                                ![621]
      IF(VALUE.EQ.0)GO TO 1940                                      ![621] A VALUE OF ZERO TURNS OFF FEATURE.
      LMDLOK=.TRUE.                                                 ![621] MUST BE ON.
      IF(ETY)GO TO 20920                                            ![621] IF BACTH, ALREADY GOT GEAR NUMBER.
      WRITE(JCT,7390)VALUE                                          ![621] PROMPT FOR GEAR NUMBER
      READ(JCT,7400)VALUE2                                          ![621] GET IT.
20920 MDLKGR=VALUE2                                                 ![621]
      MDLKEP=VALUE                                                  ![621]
      GO TO 1940
C
20940 OLDVAL(25)=LTPSED
      LTPSPD=VALUE
      GO TO 1940
C
C*****************************************************************
C
C     /LOCKUP/ , /UNLOCK/ COMMANDS - LOCK OR UNLOCK THE TRANSMISSION
C                                    GEARS
C
      ENTRY UALCMD(IECCND)                                          !DIALOG ENTRY.
      IECCND=-1
      COMND=WORD                                                    !GET COMMAND.
      GO TO 2120
C
2100  READ  (4,5360)  WORD,LOCKG
      IF ( WORD.NE.HGEAR )  GO TO 300
C
2120  DO 2180  I = 1,20                                             !LOOP THRU LOCKG LOOKING FOR GEAR #'S.
      IF ( LOCKG(I)) 2200,2180,2140                                 !(BAD VALUE/NEXT/POSSIBLY GOOD VALUE?)
2140  IF ( LOCKG(I).GT.20 )  GO TO 2200                             !GOOD VALUE?)NO.
      IF ( COMND.EQ.HLOCKU )  GO TO 2160                            !(LOCK UP GEAR?)YES.
      LOCKUP(LOCKG(I)) = .FALSE.                                    !FLG GEAR UNLOCKED.
      GO TO 2180                                                    !NEXT.
2160  LOCKUP(LOCKG(I)) = .TRUE.                                     !FLG GEAR LOCKED UP.
2180  CONTINUE                                                      !NEXT.
      IECCND=0                                                      !FLG NO ERR.
C
      GO TO 160                                                     !DONE.
C
2200  IECCND=I                                                      !FLG ERROR AND POINT TO BAD VALUE.
      GO TO 300
C
C*****************************************************************
C
C     /VEHICLE/ COMMAND - LOAD VEHICLE DATA AND STORE ON PARTS DATA
C                         FILE AND PRINT LISTING OF THE DATA
C
2220  READ  (4,5380)  VNAME                                         !GET VEHICLE NAME.
      PNAME=VNAME                                                   !SET NAME FOR DSK.
      CALL VALID2(PNAME,$140)                                       !VALID NAME?
      READ  (4,5000)  VCOM                                          !GET VEHICLE COMMENT.
```

```
          READ(4,5400) WORD,VALUE                                     !LOOK AT DATA CARD.
          BACKSPACE 4                                                 !REPOS DATA CARD TO READ AGAIN.
          IF(WORD.NE.HDATA) GO TO 300                                 !(IS IT DATA CARD?)NO, ERR.
          LVEHAX=.TRUE.                                               !ASSUME AXLE TIED TO VEHICLE.
          NRAX=1                                                      !ASSUME 1 AXLE.
          NPAX(1)=1                                                   !ASSUME NO SPIN LOSS DATA AXLE 1.
          NPAX(2)=1                                                   !ASSUME NO SPIN LOSS DATA AXLE 2.
          IF(VALUE.GT.100.) GO TO 2240                                !(NEW VEH DATA FORMAT?)YES.
          LVNEW=.FALSE.                                               !NO, FLG.
          READ  (4,5400)  WORD,WRAD,RAR,ERAR(1),WGT,AREA,FRC1,FRC2,CD, !READ DATA.
          1                 AIP,AIW,CDC,TIREFF
          WLSG=4.                                                     !SET # OF TIRES.
          NPARTS(9)=1                                                 !SET FLG TIRE DATA NOT LOADED.
          TNAME=HEVEHI                                                !SET TIRE NAME "%VEHICLE".
          GO TO 2280
C
2240      LVNEW=.TRUE.                                                !FLAG NEW VEHICLE FORMAT(NO TIRE DATA).
          READ(4,5400) WORD,WGT,CD,CDC,AREA,RAR,AIP,ERAR,WLSG         !READ DATA.
          IF(RAR.GT.1.E-7)GO TO 2260                                  !IS THERE AXLE DATA?)YES.
          LVEHAX=.FALSE.                                              !SET FLAG.
          GO TO 2360                                                  !GO ON.
2260      IF(ERAR(2).GT.1.E-7) NRAX=2                                 !(2 AXLES?)YES, FLG.
          IF(WLSG.LT..5) WLSG=4.                                      !(# OF WHEELS GIVEN?)NO, SET TO DEFAU.LT
C
2280      CALL NXTCRD(HAXLE,IFLAG,WORD)                               !LOOK AT NEXT CARD.
          GO TO(2300,2320,2360,2360),IFLAG                            !(?/FOUND "AXLE"/COMMAND/EOF)
2300      IF(WORD.NE.HSINGL) GO TO 2380                               !(SPIN LOSS FOR ONE AXLE ONLY?)NO, ERR.
2320      DO 2340 NA=1,NRAX                                           !LOOP FOR # OF AXLES.
          CALL READPD(2,20,10,HAXLE,NPAX(NA),IFLAG,AXRPM(1,NA)        !READ AXLE SPIN LOSS DATA.
          1,AXTORQ(1,NA),DUMMY,DUMMY)
          GO TO (2380,2340,2360,2360),IFLAG                           !WHAT HAPPENED (ERROR/MORE DATA/COMMAND/EOF?)
2340      CONTINUE
2360      IPRNT=3                                                     !FLG PART TYPE.
          GO TO 3720                                                  !GO STORE.
C
2380      NPARTS(3)=0                                                 !FLG PART NOT LOADED.
          GO TO 300                                                   !ERROR END.
C
C******************************************************************
C
C      /AXLE/ COMMAND - LOAD AXLE DATA AND STORE ON PARTS DATA FILE
C
2400      READ(4,5380)AXNAME
          PNAME=AXNAME
          CALL VALID2(PNAME,$140)
C
          READ(4,5000)AXCOM
          READ(4,5400)WORD,RAR,ERAR
          IF(ERAR(2).GT.1.E-7)NRAX=2
C
          CALL NXTCRD(HAXLE,IFLAG,WORD)
          GO TO (2420,2440,2480,2480),IFLAG
2420      IF(WORD.NE.PSINGL)GO TO 2500
2440      DO 2460 NA=1,NRAX
          SKIP RECORD 4
          CALL READPD(2,20,10,HAXLE,NPAX(NA),IFLAG,AXTORQ(1,NA)
          2,AXRPM(1,NA),DUMMY,DUMMY)
C
          GO TO (2500,2460,2480,2480),IFLAG
C
2460      CONTINUE
```

```
C
2490     IPRNT=11
         GO TO 3720
C
2500     NPARTS(11)=0
         GO TO 300
C
C***********************************************************************
C
C     /TRANS/ COMMAND - LOAD TRANSMISSION DATA AND STORE ON PARTS DATA FILE
C
2520     ENTRY TRACMD
2540     WRITE(5,6320)
         READ(5,6340)TRNAM
         PNAME=TRNAM
         CALL VALID2(PNAME,$2640)
2560     WRITE(5,6380)
         READ(5,*)NGTR
         IF(NGTR.LT.1.OR.NGTR.GT.20)GO TO 2560
C
         DO 2600 I=1,NGTR
         GEANUM(I)=I
2580     WRITE(5,6400) I
         READ(5,6340)GEANAM(I)
C
         CALL VALID2(GEANAM(I),$2580)
2600     CONTINUE
C
         WRITE(5,6420)
         READ(5,5000)TRCOM
C
2620     IPRNT=10
         CALL DSK
C
         GO TO 160
C
2640     WRITE(5,6360)TRNAM
         GO TO 2540
C
C
2660     READ(4,6440)TRNAM,NGTR
         PNAME=TRNAM
         CALL VALID2(PNAME,$140)
C
         READ(4,5000)TRCOM
C
         DO 2680 I=1,NGTR
         GEANUM(I)=I
         READ(4,6460)GEANAM(I)
         CALL VALID2(GEANAM(I),$140)
2680     CONTINUE
         GO TO 2620
C
C***********************************************************************
C
C     /GEAR/ COMMAND - LOAD GEAR DATA AND STORE ON PARTS DATA FILE
C                      AND PRINT LISTING OF THE DATA
C
2700     NGEAR=1                                                         !SET GEAR STORAGE PTR.
         READ  (4,5380)  PNAME                                           !GEAR NAME.
         CALL VALID2(PNAME,$140)                                         !VALID NAME?
```

```
          GNAME(NGEAR)=PNAME
          READ  (4,5000)  GCOM                                            !COMMENT.
          READ(4,5420) WORD,AIGIN(NGEAR),AIGOUT(NGEAR),GRAT(NGEAR)        !READ GEAR DATA.
          1,FRAT(NGEAR)
          IF ( WORD.NE.HDATA  )   GO TO 300                                       !(ERR?) YES.
          NGRISS(NGEAR)=1                                                 !ASSUME NO SPIN LOSS DATA.
          CALL NXTCRD(HTORQU,IFLAG,WORD)                                  !LOOK NEXT CARD.
          GO TO (300,2720,2740,2740),IFLAG                                      !(ERR/SPIN LOSS DATA/COMMAND/EOF ?)
2720      CALL READED(2,20,10,HBLANK,NGRISS(NGEAR),IFLAG,GRTORQ(1,NGEAR)        !READ GEAR SPIN LOSS DATA.
          1,GRPM(1,NGEAR),DUMMY,DUMMY)
2740      IPRNT=4                                                         !SET DSK FLG TO STORE GEAR.
          NPARTS(4)=0                                                     !SET NO GEAR LOADED FLG.
          GO TO 3740                                                      !GO STORE.
C
C*******************************************************************
C
C      /ACCESSORY/ COMMAND - LOAD ACCESSORY DATA AND STORE ON PARTS
C                            DATA FILE AND PRINT LISTING OF DATA
C
2760      NACC=1                                                          !SET ACCESSORY STORAGE PTR.
          READ  (4,5380)  PNAME                                           !ACCESSORY NAME.
          CALL VALID2(PNAME,$140)                                         !VALID NAME?
          ANAME(NACC)=PNAME
          READ  (4,5000)  ACOM                                            !COMMENT.
          READ  (4,5420)  WORD,AINS(NACC)                                 !READ INTERTIA DATA.
          IF ( WORD.NE.HINERT )   GO TO 300                                     !(ERR?) YES.
          CALL READED(2,20,10,HBLANK,NHA(NACC),IFLAG,ACCT(1,NACC)         !READ TORQUE LOSS DATA.
          1,ACCS(1,NACC),DUMMY,DUMMY )
          IPRNT=5                                                         !SET DSK FLG TO STORE ACCESSORY.
          GO TO 3720                                                      !GO STORE.
C
C*******************************************************************
C
C      /SHIFT LOGIC/ COMMAND - LOAD SHIFT LOGIC DATA AND STORE ON PARTS
C                              DATA FILE AND PRINT LISTING OF THE DATA
C
2780      GDAT   = .FALSE.
          READ  (4,5380)  SNAME
          PNAME=SNAME
          CALL VALID2(PNAME,$140)                                         !VALID NAME?
          READ  (4,5000)  SCOM
          READ  (4,5420)  WORD,DNUMG
          IF ( WORD.NE.HNUMG  )   GO TO 300
          NUMG   = DNUMG + .001
          IF ( NUMG.LT.1 .OR. NUMG.GT.20 )   GO TO 300
          NUMPSL = (NUMG-1) * 2
          IF ( NUMPSL.LT.1 )  GO TO 2840
C
          DO 2820  I = 1,NUMPSL
          READ  (4,5420)  WORD,DGF,DGT,SHFTIM(I)
          IF ( WORD.NE.HSHIFT )   GO TO 300
          IGF(I) = DGF + .001
          IGT(I) = DGT + .001
          IF ( SHFTIM(I).LT.1.E-10 )  SHFTIM(I) = 1.
          IF ( I.GT.1 ) GO TO 2800
          READ  (4,5000)  WORD
          IF ( WORD.NE.HVACUU .AND. WORD.NE.HTHROT .AND.WORD.NE.HDTHRO)
          1 GO TO 300
          SAVE1  = WORD
CCCC      IF ( WORD.EQ.HVACUU )  IVAC  = .TRUE.
CCCC      IF ( WORD.EQ.HTHROT )  IVAC  = .FALSE.
```

```
        LVAC=.FALSE.
        LDTH=.FALSE.
        IF(WORD.EQ.HVACUU) LVAC=.TRUE.
        IF(WORD.EQ.HDTHRO) LDTH=.TRUE.
        READ  (4,5000)  WORD
        IF ( WORD.NE.HOUTEU .AND. WORD.NE.HENGIN .AND. WORD.NE.HVEHIC )
        1   GO TO 300
        SAVE2  = WORD
        IF ( WORD.EQ.HCUTEU )  LENG  = .FALSE.
        IF ( WORD.EQ.HENGIN )  LENG  = .TRUE.
        IF ( WORD.EQ.HVEHIC )  EAFAB = .TRUE.
        IF ( WORD.NE.HVEHIC )  EAFAB = .FALSE.
        BACKSPACE 4
        BACKSPACE 4
C.................................
2800    CALL READPD ( 2,10,10,HSHIFT,NSPTS(I),IFLAG,
        1              SHFTPT(1,I),SHFTRP(1,I),DUMMY,DUMMY )
C
C     READ DETENT OVERRIDE DATA IF PRESENT (FOR 100 PCT WOT SEGS)
C
        LDETNT = .FALSE.
        READ  (4,5000,END=2820)  WORD
        BACKSPACE 4
        IF ( WORD.NE.PCETEN )  GO TO 2920
        LDETNT = .TRUE.
        READ  (4,5340)  WORD1,DETET(I),WORD2,DETRPM(I)
        IF(WORD1.NE.HVACUU.AND.WORD1.NE.HTHROT.AND.WORD1.NE.HDTHRO)
        1 GO TO 300
        IF ( WORD2.NE.HCUTPU .AND. WORD2.NE.HENGIN .AND. WORD2.NE.HVEHIC )
        1   GO TO 300
C
C     CHECK THAT DETENT UNITS MATCH SHIFT LINE UNITS
C     LOGICAL FLAGS ARE SET BUT NOT USED BY THE PROGRAM
C
        IF ( WORD1.NE.SAVE1 .OR. WORD2.NE.SAVE2 )  GO TO 300
        IF ( WORD1.EQ.HVACUU )  LDETV = .TRUE.
        IF ( WORD1.EQ.HTHROT )  LDETV = .FALSE.
        IF ( WORD2.EQ.HCUTPU )  LDETE = .FALSE.
        IF ( WORD2.EQ.HENGIN )  LDETE = .TRUE.
2820    CONTINUE
C
2840    IPRNT     = 7
        GO TO 3720
C
C*****************************************************************
C
C     /DRIVE SCHEDULE/ COMMAND - LOAD DRIVING SCHEDULE DATA AND STORE
C                                ON PARTS DATA FILE AND PRINT LISTING
C                                OF THE DATA
C
2860    READ  (4,5380)  DNAME                                         !GET DRS NAME.
        PNAME=DNAME                                                   !PASS FOR DSK.
        CALL VALID2(PNAME,$140)                                       !VALID NAME?
        READ  (4,5000)  DCOM                                          !GET DRS COMMENT.
        READ  (4,5420)  WORD,TO,DO,VO,AO,GO,DUTCYC                    !READ INITIAL CONDITIONS.
        IF ( WORD.NE.HINITI )  GO TO 300                              !(ERROR?)YES, BYE.
        NGO     = GO + .001                                           !FIX INITIAL GEAR.
        IF ( NGO.EQ.0 )  NGO = 1                                      !(INITIAL GEAR ZERO?)SET TO 1ST GEAR.
        IF(DUTCYC.EQ.0.)DUTCYC=1
        NDSEG=0                                                       !INITALIZE DRS SEG OFFET.
        MLTSEC=.FALSE.                                                !ASSUME NOT MULTI SECT.
```

```
C
2800     DO 2900  NSEG = 1)50                                          !READ UP TO 50 DRS SEGS.
         REAC  (4,5420)  WCRD,ASEG(NSEG),VSEG(NSEG),PWOT(NSEG),        !READ A SEGMENT.
         1       ATHOLD(NSEG),GSEG,THRATE(NSEG),TSEG(NSEG),
         2       CSEG(NSEG),PCSEG(NSEG),POSTSE(NSEG),VELSEG(NSEG)
         IF ( WORD.NE.HSEGME )   GO TO 300                             !(ERROR?)YES, BYE.
         NGSEG(NSEG) = GSEG + .001                                     !FIX GEAR TO HOLD.
         BACKSPACE 4                                                   !REREAD.
         READ  (4,5440)  (CARD(I),I=1,4),TBL,DBL,CBL,OBL,EBL           !READ A FORMAT TO SEE WHAT WE GOT.
C
         ITYSEG(NSEG)=NWFCNT(CARC,4)                                   !DETERMINE TYPE OF SEGMENT
C
         IF ( CDL.EO.HELANK )  PCSEG  (NSEG)  = -1.                    !DETERMINE SEGMENT END CONDITIONS
         IF ( CBI.EQ.HELANK )  DSEG   (NSEG)  = -1.
         IF ( EDL.EO.HELANK )  VELSEG(NSEG)  = -1.
         IF ( OBL.EO.HELANK )  PCS1SE(NSEG)  = -1.
         IF ( TBL.EO.HELANK )  TSEG   (NSEG)  = -1.
C
         CALI NXTCFD(HSEGME,IPLAG,WORD)                                !LOOK NEXT CARD.
         GO TO(300,2900,2940,2940),IPLAG                               !(ERR/MORE DATA/COMMAND/EOF).
C
2900     CONTINUE                                                      !NEXT CARD.
C
         NSEG=NSEG-1                                           !CALC # OF SEG'S WHEN NORMAL LOOP TERMINATION, MUST BE MULTI SECT.
         IF(PLTSEC) GO TC 2920                                         !(1ST TIME?)NO.
          NPARTS(6)=0                                                   !YES, FLG R S NOT LONE D.
         LSTSEC=.FALSE.                                                !FLG NOT LAST SECT.
         MLTSEC=.TRUE.                                                 !FLG DRS MULTI SECT.
C
2920     NSEG=-NSEG                                                    !FLG DSK MULTI SECT DRS.
         IPRNT=6                                                       !FLG DSK STORE DRS.
         LSAVE=LSTSEC                                                  !SAVE.
         CALI DSK                                                      !GO STCRE ON DB.
         NDSEG=NDSEG+NSEG                                              !CALC DRS SEG OFFSET.
         IF(.NOT.LSAVE)  GO TO 2880                                    !(LAST SECT?)NO.
         IERRT=106                                                     !YES, FLG DSK /USE/ DRS.
         GO TO 4120                                                    !RELOAD 1ST SECT.
C
2940     LSTSEC=.TRUE.                                                 !FLG LAST SECT OF DRS.
         IF(PLTSEC)  GO TC 2920                                        !(MULTI SECT?)YES.
         IPRNT=6                                                       !NO, FLG DSK TO STORE DRS.
         GO TO 3720                                                    !GO STORE.
C
C********************************************************************
C
C     /ROUTE/ CCMMAND - ICAD FOUTE DATA ANC STORE ON PARTS DATA
C                       FILE AND PRINT LISTING OF DATA
C
2960     REAC  (4,5380)   RNAME                                        ! ROUTE NAME
         PNAME=RNAME
         CALI VALIC2(PNAME,$140)                                       !VALID NAME?
         REAC  (4,5000)   RCOM                                         ! COMMENT
C
         MLTSEC=.FALSE.                                                !ASSUME NOT MULTI SECT RTE.
         DRVWNC=1000.0                                                 !SET DRS WINC SPEED TO FLG RUN TIME DEFAULT.
         READ(4,5420) WCRD,VALUE                                       !READ NXT CARD.
         IF(WORD.NE.HDATA) GC TO 2980                                  !(CRS WIND SPEED DATA CARD?)NC.
         DRVWND=VALUE                                                  !YES, RESET CRS WIND SPEED DEFAULT VALUE.
         GO TO 3000
2980     BACKSPACE 4                                                   !POSI CTRFIL.
         IF(WORD.NE.HMILEF) GO TC 240                                  !(ERR?) YES.
```

```
C
3000  DO 3020 I=1,10                                                   !INTIALIZE SEGMENT WIND SPEED TO SCHED DEFAULT.
3020  RVWIND(I)=DRVWND
      CALL READPD ( 3,10,10,HMILEP,NRDIST,IFLAG,                        ! READ RTE SECT
      1                 RDIST,RGRADE,RCOEF,DUMMY)
C
      GO TO (3040,3060,3100,3100),IFLAG                                 !(WHAT HAPPENED?) WIND SPD DATA/MORE RTE NXT/COMND NXT/EOF

C
3040  CALL READPD(1,10,10,HMILEP,I,IFLAG                                !READ SECT WIND SPEED DATA.
      1,RVWIND,DUMMY,DUMMY,DUMMY)
      IF(NRDIST.NE.I) GO TO 300                                         !(ERR?) YES.
      GO TO (300,3060,3100,3100),IFLAG                                  !(WHAT HAPPENED?) ERROR/MORE RTE NXT/COMND NXT/EOF
C
3060  IF(MLTSEC) GO TO 3080                                             ! (1ST TIME?) NO.
      MLTSEC=.TRUE.                                                     !YES SET MULTI SECT FLG
      NPARTS(8)=0                                                       ! IN CASE OF RELOAD BOMB SET NO LOAD FLG
      LSTRTE=.FALSE.                                                    !SET FLG NOT LAST RTE.
C
3080  NRDIST=-NRDIST                                                    !SET DB FLG TO MULTI REC PART
      IPRNT=9                                                           !SET DSK FLG TO STORE RTE.
      LSAVE=LSTRTE                                                      ! SAVE
      CALL DSK                                                          ! GO STORE RTE SECT
      IF(.NOT.LSAVE) GO TO 3000                                         !(LST RTE SECT?)  NO, GO GET NXT RTE SECT
      IPRNT=108                                                         !YES, SET DSK FLG TO /USE/ RTE.
      GO TO 4400                                                        !GO RELOAD 1ST SECT.
C
3100  LSTRTE=.TRUE.                                                     ! SET LST RTE SECT FLG
      IF(MLTSEC) GO TO 3080                                             ! (MULTI REC RTE?) YES.
      IPRNT=8                                                           !NO, SET DSK FLG TO STORE RTE
      GO TO 3720                                                        ! GO STORE SNGL SECT RTE
C
C*******************************************************************
C
C     /FULL CONVERTER/ COMMAND - LOAD FULL CONVERTER DATA AND STORE ON
C                                PARTS DATA FILE AND PRINT LISTING OF
C                                DATA
C
3120  LFULL=.TRUE.                                                      !FLG FULL CONVERTER DATA TO BE READ.
      GO TO 3160                                                        !GO READ.
C
C*******************************************************************
C
C     /S.R. CONVERTER/ COMMAND - LOAD S.R. CONVERTER DATA AND STORE ON
C                                PARTS DATA FILE AND PRINT LISTING OF
C                                DATA
C
3140  LFULL=.FALSE.                                                     !FLG S.R. CONVERTER DATA TO BE READ.
3160  READ  (4,5380)  CNAME,WORD                                        !GET CONVERTED NAME AND MODE.
      IF ( WORD.NE.HCOAST .AND. WORD.NE.HDRIVE )  GO TO 300             !(LEGAL MODE?)NO, ERR, BYE.
      PNAME=CNAME                                                       !PASS NAME TO DSK.
      CALL VALID2(PNAME,$140)                                           !VALID NAME?.
      COAST  = .TRUE.                                                   !ASSUME COAST CONV.
      IF ( WORD.EQ.HDRIVE )  COAST = .FALSE.                            !(DRIVE CONV?)YES, FLG.
      READ  (4,5000)  CCOM                                              !GET CONV COMMENT.
      READ  (4,5420)  WORD,CDIAM,CCNTOR,AI1,AI2                         !CONV DATA.
      IF ( WORD.NE.HDATA  )  GO TO 300                                  !(ERROR?)YES, BYE.
      IF(LFULL) GO TO 3180                                              !(FULL CONV?)YES.
      CALL READPD ( 3,20,10,HBLANK,NTORP,IFLAG,SOUT,TOUT,SPIN,DUMMY )
      GO TO 3200
3180  CALL READPD ( 4,20,10,HBLANK,NTORP,IFLAG,TIN,TOUT,SPIN,SOUT )
```

```
3200      NTD      = NTORP
          NTC      = NTORE
          NTBP     = 0
C
          DO 3280  I = 1,NTCRP
          IF ( CCNTCR.GT..001 )  TIN(I) = CONTOR
          IF(IFULL) GC TO 3220                                          !(FULL CONV?)YES.
          TOUT(I)   = TOUT(I) * TIN(I)
          SOUT(I)   = SOUT(I) * SPIN(I)
3220      IF ( COAST )   GC TO 3260
C
C        CALCULATE CAPACITY FACTCR,SPEED RATIO,TORQUE RATIO
C
          AKD(I) = SOUT(I) / SQRT ( TOUT(I) )
          SRD(I) = SOUT(I) / SPIN(I)
          TRD(I) = TOUT(I) / TIN(I)
C
C        FIND TORQUE BREAKPCINT FOR DRIVE CONVERTER
C
          IF ( NTBP.GT.0  )   GO TC 3280
          IF ( I.EQ.NTORP )   GO TC 3240
          IF ( ABS ( TRD(I) - 1. ) .GT. .0001 )   GO TO 3280
3240      NTBP    = I
          TORBPK = AKD(I)
          GO TO 3280
C
C        CALCULATE CAPACITY FACTCR,SPEED RATIO FOR COAST CONVERTER
C
3260      AKC(I) = SPIN(I)
          SRC(I) = SOUT(I) / SPIN(I)
3280      CONTINUE
C
3300      IPRNT      = 2
          NPARTS(2) = NPARTS(2) + 1
          GO TO 3740
C
C*****************************************************************************
C
C        /ENGINE/ COMMAND - LOAD ENGINE DATA AND STORE ON PARTS DATA
C                           FILE AND PRINT LISTING OF THE DATA
C
3320      READ   (4,5380)  PNAME,UNITS
          CALL VALID2(PNAME,$140)                                       !VALID NAME?
          IF( .NOT.((UNITS(1).EQ.FRPM   .OR.  UNITS(1).EQ.HPISTO )
          1    .AND. (UNITS(2).EQ.HEMEP  .OR.  UNITS(2).EQ.HTORQU  .OR.
          2           UNITS(2).EQ.HHP    )
          3    .AND. (UNITS(3).EQ.HLBHR  .OR.  UNITS(3).EQ.HBSFC   .OR.
          4           UNITS(3).EQ.HGALHR)
          5 .AND. (UNITS(4).EQ.HDIES .OR. UNITS(4).EQ.HBLANK)))GO TO 300
          READ   (4,5000)  ECOM
          READ   (4,5460)  WORD,BORE,STROKE,CYL,ENMIN(1),RPMAX(1),
          1                EINER,ESEGR,CYCLE
          IF ( WORD.NE.HDATA  )   CO TO 300
          ICYL    = CYL + .001
          DISP    = 3.1415927 * ((BORE/2.)**2.) * STROKE * ICYL
          IF ( ESEGR.LT..0001 )   ESEGR = .764
          NCYCLE = 4
          ICYCLE = CYCLE + .001
          IF ( ICYCLE .EQ. 2 )   NCYCLE = 2
C
C        DETERMINE UNITS USED IN ENGINE MAP
```

```
C
          LRPM   = .TRUE.                                                    !SET TO DEFAULT UNITS.
          LTOR   = .TRUE.
          LLBHR  = .TRUE.
          LPS    = .FALSE.
          LBMEP  = .FALSE.
          LBSFC  = .FALSE.
          LHP    = .FALSE.
          LGALHR = .FALSE.
          IF ( UNITS(1).NE.HPISTO )  GO TO 3340
          LRPM   = .FALSE.
          LPS    = .TRUE.
 3340     IF ( UNITS(2).NE.HBMEP .AND. UNITS(2).NE.HHP    )  GO TO 3380
          LTOR   = .FALSE.
          IF ( UNITS(2).EQ.HBMEP  )  GO TO 3360
          LHP    = .TRUE.
          GO TO 3380
 3360     LBMEP  = .TRUE.
 3380     IF ( UNITS(3).NE.HBSFC .AND. UNITS(3).NE.HGALHR )  GO TO 3420
          LLBHR  = .FALSE.
          IF ( UNITS(3).EQ.HBSFC  )  GO TO 3400
          LGALHR = .TRUE.
          GO TO 3420
 3400     LBSFC  = .TRUE.
 3420     CONTINUE
          LDIES=.FALSE.
          IF(UNITS(4).EQ.HDIES) LDIES=.TRUE.
C
C       ZERO ENGINE MAP
C
          DO 3440  I = 1,20
          ERPM(1,I)      = 0.
          NTOR(1,I)      = 0.
          ERPMO(I)       = 0.
          NTORO(I)       = 0.
          DO 3440  J = 1,20
          DO 3440  K = 1,4
          EMAP(I,J,K)    = 0.
 3440     EMAPO(I,J,K)   = 0.
C
C       LOAD ENGINE MAP DATA
C
          DO 3500  NRPMP = 1,20                                              !LOAD UP TO 20 SPEED POINTS OF DATA.
          READ  (4,5420)  WORD,ERPM(1,NRPMP)                                 !READ A SPEED POINT CARD.
          IF ( WORD.NE.HSPEED )  GO TO 300                                           !(ERROR?)YES, BYE.
C.................................
          CALL READPD ( 4,20,10,HSPEED,NPOINT,IFLAG,DATA( 1),DATA(21),
          1                                              DATA(41),DATA(61) )
          NTOR(1,NRPMP) = NPOINT                                             !SAVE # OF LOAD POINTS THIS SPEED POINT.
C
C       RIGHT JUSTIFY ENGINE MAP
C
          DO 3460  J = 1,NPOINT
          DO 3460  K = 1,4
 3460     EMAP(NRPMP,21-J,K)  =  DATA(20*(K-1)+NPOINT+1-J)
C
C       FILL LEFT HALF OF MAP WITH FIRST DATA POINT
C
          N        = 20 - NPOINT
          DO 3480  J = 1,N
          DO 3480  K = 1,4
```

```
C
C     CHECK FOR END OF DATA
C
        IF ( IFLAG.GT.2 )  GO TO 3520                              !(MORE DATA?)NO.
3500    CONTINUE                                                   !YES, NEXT SPEED POINT.
C
        NRPMP      = NRPMP - 1
3520    NRPM(1)    = NRPMP
C
C     CONVERT PISTON SPEED TO RPM
C
        IF ( LRPM  )  GO TO 3560
        CONST      = 6. / STROKE
        DO 3540  I = 1,NRPMP
3540    ERPM(1,I)  = CONST * ERPM(1,I)
        ERMIN(1)   = CONST * ERMIN(1)
        RPMAX(1)   = CONST * RPMAX(1)
        SPIDLE(1)  = CONST * SPIDLE(1)
C
C     CONVERT BMEP,HP TO LB-FT
C
3560    IF ( LTCR  )  GO TO 3600
        CONST      = DISP / ( 150.8 * NCYCLE/4. )
        DO 3580  I = 1,NRPMP
        IF ( LHP   )  CONST = 5252. / ERPM(1,I)
        DO 3580  J = 1,20
3580    EMAP(I,J,1)  = CONST * EMAP(I,J,1)
C
C     CONVERT GAL/HR,BSFC TO LB/HR
C
3600    IF ( LLBHR )  GO TO 3680
        IF ( LBSFC )  GO TO 3640
        CONST      = 7.480520 / ( FSPGR * 62.426134 )
        DO 3620  I = 1,NRPMP
        DO 3620  J = 1,20
3620    EMAP(I,J,2)  = CONST * EMAP(I,J,2)
        GO TO 3680
3640    DO 3660  I = 1,NRPMP
        CONST      = ERPM(1,I) / 5252.
        DO 3660  J = 1,20
3660    EMAP(I,J,2)  = CONST * EMAP(I,J,1) * EMAP(I,J,2)
C
C     COMPUTE MAX AND MIN THROTTLE ANGLES
C
3680    IMIN       = ERMIN(1) + .001
        IMAX       = RPMAX(1) + .001
        RPMIN(1)   = ERPM(1,1)
        THRMAX     = - 1000.
        THRMIN     =   1000.
        DO 3700  I = 1,NRPMP
        IF ( EMAP(I, 1, 3) .LT. THRMIN )  THRMIN = EMAP(I, 1, 3)
        IF ( EMAP(I,20, 3) .GT. THRMAX )  THRMAX = EMAP(I,20, 3)
3700    CONTINUE
C
        IENG       = 1                                             !POINT TO ENG TO STORE.
        ENAME(1)=PNAME                                             !SET ENG NAME.
        IFLAT      = 1                                             !SET DSK FLG TO STORE ENG.
        NPARTS(4)  = 0                                             !FLG ENG NOT LOADED.
        NENG       = 0                                             !FLG ENG NOT LOADED.
        GO TO 3740                                                 !GO STORE.
```

```
C
C********************************************************************
C
C     STORE PART DATA ON PARTS DATA FILE AND PRINT OUT THE DATA
C
3720    NPARTS(IPRNT)=1                                                 ! SET PART LOADED FLAG
C
3740    CALL DSK                                                        !STORE PART ON DB.
        CALL PRNOUT                                                     !GO PRINT DATA.
        GO TO 160                                                       !DONE.
C
C********************************************************************
C
C     /REMAP/ COMMAND - LOAD ENGINE DATA FROM PARTS DATA FILE , CONVERT
C                       UNITS IF NECESSARY , REMAP DATA TO SPECIFIED
C                       SPEED AND LOAD POINTS AND PRINT OUT ENGINE MAP
C
        ENTRY RMPCMD(IECCND)
        IECCND=2
C
3760    LRPM   = .TRUE.
        LTCR   = .TRUE.
        LLBHR  = .TRUE.
        LPS    = .FALSE.
        LBMEP  = .FALSE.
        LBSFC  = .FALSE.
        LHP    = .FALSE.
        LGALHR = .FALSE.
C
C     LOAD ENGINE DATA FROM PARTS DATA FILE
C
        IF(BATCH) READ  (4,5380)  PNAME,UNITS
        CALL VALID2(PNAME,$140)                                         !VALID NAME?
        PNAME(1)=ENAME
        IF ( UNITS(1).NE.HRPM  .AND.UNITS(1).NE.HPISTO )  GO TO 300
        IECCND=3
        IF ( UNITS(2).NE.HBMEP .AND.UNITS(2).NE.HTORQU.AND.
     1 UNITS(2).NE.HHP    )      GO TO 300
        IECCND=4
        IF ( UNITS(3).NE.HLBHR .AND.UNITS(3).NE.HBSFC .AND.
     1 UNITS(3).NE.FGALHR )      GO TO 300
        IECCND=1
        IENG   = 1
        IPRNT  = 101
        LSAVE  = LIMPRN
        LIMPRN = .TRUE.
C..................................
        CALL DSK
        LIMPRN = LSAVE
C
C     SET ENGINE MAP PRINTOUT UNITS FLAGS
C
        IF ( UNITS(1).NE.FPISTO )  GO TO 3780
        LRPM   = .TRUE.
        LPS    = .FALSE.
3780    IF ( UNITS(2).NE.HBMEP .AND. UNITS(2).NE.HHP    )  GO TO 3820
        LTCR   = .FALSE.
        IF ( UNITS(2).EQ.HBMEP  )  GO TO 3800
        LHP    = .TRUE.
        GO TO 3820
3800    LBMEP  = .TRUE.
```

```
3820    IF ( UNITS(3).NE.HDSFC .AND. UNITS(3).NE.HGALHR )  GO TO 3860
        LLPFR  = .FALSE.
        IF ( UNITS(3).EQ.HDSFC  )  GO TO 3840
        LGALHR = .TRUE.
        GO TO 3860
3840    LDSFC  = .TRUE.
C
3860    IF(BATCH) GO TO 3880
C
C   **CODE TO BE INSERTED FOR DIALOG MODE**
C
        GO TO 3900
C
C      SET UP ARRAY OF POINTS FOR REMAPPING ENGINE DATA MAP
C
C.................................
3880    CALL READPD ( 1,20,10,HLOAD,NRPMO,IFLAG,
        1                 RRPMO,DUMMY,DUMMY,DUMMY )
C.................................
        CALL READPD ( 1,20,10,HBLANK,NPOINT,IFLAG,
        1                 DATA,DUMMY,DUMMY,DUMMY )
C
3900    DO 3920  J = 1,NPOINT
3920    EMAPO(1,21-J,1) = DATA(NPOINT+1-J)
        DO 3940  I = 1,20
3940    NTORO(I)   = NPOINT
        DO 3960  I = 2,NRPMO
        DO 3960  J = 1,20
3960    EMAPO(I,J,1) = EMAPO(1,J,1)
        LPRNT      = .TRUE.
C
C      REMAP ENGINE DATA MAP
C
        CALL REMAP
        NENG   = 0
        GO TO 160
C
C*****************************************************************************************
C
C     /TIRE/ COMMAND- LOAD TIRE DATA AND STORE ON PARTS DATA FILE
C                        AND PRINT LISTING.
C
3980    READ(4,5380) TNAME                                              !GET TIRE NAME.
        PNAME=TNAME                                                     !PASS TO DSK.
        CALL VALID2(PNAME,$140)                                         !VALID NAME?
        READ(4,5000) TCOM                                               !GET TIRE COMMENT.
        READ(4,5400) WCFD,WRAD,FRC1,FRC2,TIREFF,AIW                     !READ TIRE DATA.
        IF(WORD.NE.HDATA) GO TO 300                                     !(ERROR?)YES, BYE.
        IFRST=9                                                         !FLG DSK TO STORE TIRE.
        GO TO 3720                                                      !STORE.
C
C************************************************************************************
C***************************************************************************
C
C      /USE/ COMMAND - LOAD PART DATA FROM PARTS DATA FILE AND PRINT DATA
C
        ENTRY USECMD(IECCMD)                                            !DIALOG ENTRY.
        LPDF=.FALSE.                                                    ![617] ASSUME NOT PDP DUMP
        IF(IECOMD.GT.0)GO TO 40000                                      ![617] IF NEG, PDP DUMP.
        LPDF=.TRUE.
        IECCMD=ABS(IECCMD)                                              !RESET FLAG
```

```
40000   IP=IECOND                                                              !GET PART TYPE PTR.
        IECCND=3                                                               !ASSUME ERROR.
        GO TO 4020
C
4000    REAL  (4,5580)  PNAME,UT,I,ICATA                                       !GET USE COMMAND.
        CALL VALIC2(PNAME,$140)                                                !VALID NAME?
C
        CALL LCCKUP(UT,5,HPAPT,LUMMY,NUMPAR,IP,DBLANK,$4020)                   !LOOK UP PART TYPE AND SET PTR,(GOT IT?)YES.
        GO TO 4600                                                             !?TYPE?
4020    IPRNT=100+IP                                                           !SET DSK FLG TO /*USE/
        GO TO (4140,4080,4460,4320,4040,4100,4420,4380,4500,4280,4440),IP        !GO SET UP & DO ANY SPECIAL HANCLING.
C
C
C=============================================================================
C     LOAD ACCESSORY CATA FROM PARTS CATA FILE
C
4040    NACC        = NACC + 1                                                 !INC # OF ACC LOADED.
        ANAME(NACC) = PNAME                                                    !SET ACC NAME.
        CALL DSK                                                               !LOAD ACC.
        IF ( IPRNT.EQ.-10 )  GO TC 4060                                        !(ERROR?)YES.
        IF(IPDP)GC TO 4070                                                     ![617]
        NPARTS(5) = NACC                                                       !FLG ALL LOADED.
        GO TO 4560
4060    NACC        = NACC - 1                                                 !ON ERR DEC # OF ACC'S LOADED.
        GO TO 4580                                                             !ERR BYE.
C
4070    WRITE(6,7000)PNAME                                                     ![617]
        WRITE(6,5000) (ACCM(IT),IT=1,NRRCNT(ACOM,16))                          ![617]
        WRITE(6,7020)AIAS(NACC)                                                ![617]
        IS=1
        IE=KNA(NACC)                                                           ![617]
        IF(IP.LE.10)GC TO 4072
        IP=10
4072    WRITE(6,7040) (ACCT(IT,NACC),IT=IS,IP)                                 ![617]
        WRITE(6,7060) (ACCS(IT,NACC),IT=IS,IP)                                 ![617]
        IF(KNA(NACC).IE.10 .OR. IE.GT.10) GO TO 4074
        IS=11
        IP=KNA(NACC)
        GO TO 4072
4074    NACC=NACC-1                                                            ![617]
        GO TO 4560                                                             ![617]
C
C================================================================================
C     LOAD CONVERTER CATA FROM PARTS CATA FILE
C
4080    CNAME      = PNAME                                                     !GET CONV NAME.
C.................................
        CALL DSK                                                               !LOAD CONV.
        IF ( IPRNT.EQ.-10 )  GO TO 4580                                        !(ERROR?)YES, BYE.
        IF(IPDP)GO TO 4560
        NPARTS(2) = NPARTS(2) + 1                                              !FLG CONV AND INC CNT.
        IF(.NOT. COAST) CBVNAM(1)=CNAME                                        !(CRIVE CONV?)YES, SAVE NAME.
        IF(COAST) CNVNAM(2)=CNAME                                              !(COAST CONV?)YES, SAVE NAME.
        GO TO 4560                                                             !DCNE.
C
C=============================================================================
C     LOAC CRIVING SCHECULE DATA FROM PARTS DATA FILE
C
4100    DNAME      = PNAME                                                     !GET DRS NAME.
4120    CALL DSK                                                               !LOAD DBS.
        IF ( IPRNT.EC.-10 )  GO TC 4580                                        !(EROR?)YES, BYE.
```

```
      IF(LPDP)GO TO 4580
      MFLG(22)=0                                                        !FLAG RESET NO MOD TO DUTY CYCLE
      GO TO 4540                                                        !DONE.
C
C==========================================================================
C     LOAD ENGINE DATA FROM PARTS DATA FILE
C
4140  IF(.NOT.LPDP)GO TO 4142                                           ![617]
      IENG=1                                                            ![617]
      GO TO 4180                                                        ![617]
4142  IF(DIALOG) GO TO 4160                                             !(DIALOG MODE?)YES.
      IENG=I
      IF ( IENG.GT.2 )  GO TO 4600
      IF ( IENG.EQ.0 )  IENG = 1
      GO TO 4180
4160  IENG=LOCNG(1)                                                     !GET ENG LOC TO USE.
4180  ENAME(IENG)=PNAME                                                 !GET ENG NAME.
      CALL DSK                                                          !LOAD ENG.
      IF ( IPRNT.EQ.-10 )  GO TO 4580                                   !(ERROR?)YES, BYE.
      IF(LPDP)GO TO 4270                                                ![617]
      DO 4200 I=13,17
4200  MFLG(I)=0                                                         !FLG NO ENG MODS.
      IF(IENG.EQ.1) ENG1=.TRUE.                                         !LOAD ENG INTO LOC 1?)YES, FLG.
      IF(IENG.EQ.2) ENG2=.TRUE.                                         !LOAD ENG INTO LOC 2?)YES, FLG.
      DO 4220  I = 1,20
      J          = IDATA(I)
      IF ( J.LT.0 .OR. J.GT.20 )  GO TO 4600
      IF ( J.GT.0 .AND. J.LE.20 )  JENG(J) = IENG
4220  CONTINUE
      IF ( IENG.EQ. 2 )  GO TO 4240
      IF ( NENG.EQ. 0 )  NENG =  1
      IF ( NENG.EQ.10 )  NENG = 11
      GO TO 4260
4240  IF ( NENG.LT.10 )  NENG = NENG + 10
4260  IF(NENG.NE.11) GO TO 4540                                         !(ENG 1?)YES, DONE.
      NPARTS(1)=2                                                       !FLG 2 ENG'S LOADED.
      GO TO 4560                                                        !DONE.
C
4270  IF(IRPM)UNITS(1)=HRPM                                             ![617]
      IF(IFS)UNITS(1)=HFS                                               ![617]
      IF(LTOR)UNITS(2)=HTORQU                                           ![617]
      IF(LBMEP)UNITS(2)=HBMEP                                           ![617]
      IF(IHP)UNITS(2)=HHP                                               ![617]
      IF(LLBHR)UNITS(3)=HLBHR                                           ![617]
      IF(LBSFC)UNITS(3)=HBSFC                                           ![617]
      IF(IGALHR)UNITS(3)=HGALHR                                         ![617]
      UNITS(4)=HBLANK
      IF(LDIES)UNITS(4)=HDIES                                           ![617]
      WRITE(6,7080) ENAME(IENG),UNITS
      WRITE(6,5000) (ECOM(IT),IT=1,NWRCNT(ECOME,16))
      XCYL=ICYL          ;          XCYC=NCYCLE                         ![617]
      WRITE(6,7100) BORE,STROKE,XCYL,RPMIN(IENG),RPMAX(IENG),EINER
     2        ,ESPGR,XCYC                                               ![617]
      IB=(IENG-1)*4+1                                                   ![617]
      DO 4274 IR=1,NREM(IENG)                                           ![617]
      WRITE(6,7120) RREM(IENG,IR)
      NPS=NTOF(IENG,IR)
      NP=20
      IS=21-NPS
      IF(NPS.LE.10)GO TO 4272
      NP=IS+9
```

```
4272    WRITE(6,7140) (EMAE(IR,IE,IB),IP=IS,NP)                          ![617]
        WRITE(6,7160) (EMAE(IR,IE,IB+1),IP=IS,NP)                        ![617]
        WRITE(6,7180) (EMAE(IR,IE,IB+2),IP=IS,NP)                        ![617]
        WRITE(6,7200) (EMAE(IR,IE,IB+3)),IP=IS,NP)                       ![617]
        IF(NPS.LE.10 .OR. IS.GT.10)GO TO 4274
        IS=NP+1
        NP=NPS
        GO TO 4272
4274    CONTINUE
        GO TO 4560
C
C=======================================================================================
C     LOAD TRANSMISSION DATA FROM PARTS DATA FILE
C
4280    CALL DSK
        IF(IPRNT.EQ.-10)GO TO 4580
        IF(LPDP)GO TO 4560
C
        DO 4300 I=1,NGTF
        NGEAR=GEARNUM(I)
        PNAME=GEARNAM(I)
        GNAME(NGEAR)=PNAME
        IPRNT=104
        CALL DSK
        IF(IPRNT.EQ.-10)GO TO 4580
        NPARTS(4)=NPARTS(4)+1
4300    CONTINUE
C
        NPARTS(10)=1
        GO TO 4560
C
C=======================================================================================
C     LOAD GEAR DATA FROM PARTS DATA FILE
C
4320    IF(LPDP)GO TO 4560
        DO 4340  I = 1,20                                                !SCAN IDATA FOR GEAR.
        IF ( IDATA(I).GT.0 .AND. IDATA(I).LE.20 )   GO TO 4360           !(GOT A VALID #?)YES.
4340    CONTINUE                                                         !NO,NEXT.
        GO TO 4600                                                       !ERR, GEAR NOT ASSIGNED # OR INVALID #.
4360    NGEAR      = IDATA(I)                                            !GET GEAR #.
        GNAME(NGEAR)=PNAME                                               !GET GEAR NAME.
        CALL DSK                                                         !LOAD GEAR.
        IF ( IPRNT.EQ.-10 )   GO TO 4580                                 !(ERROR?)YES, BYE.
        NPARTS(4) = NPARTS(4) + 1                                        !NO, FLG GEAR LOADED & INC CNT.
        NPARTS(10)=0
        GO TO 4560                                                       !DONE.
C
C=============================================================================
C     LOAD ROUTE DATA FROM PARTS DATA FILE
C
4380    RNAME      = PNAME                                               !GET ROUTE NAME.
4400    CALL DSK                                                         !LOAD ROUTE.
        IF ( IPRNT.EQ.-10 )   GO TO 4590                                 !(ERROR?)YES, BYE.
        IF(LPDP)GO TO 4560
        GO TO 4540                                                       !NO, DONE.
C
C=============================================================================
C     LOAD SHIFT LOGIC DATA FROM PARTS DATA FILE
C
4420    SNAME      = PNAME                                               !GET SHIFT LOGIC NAME.
        CALL DSK                                                         !LOAD SHIFT LOGIC.
        IF ( IPRNT.EQ.-10 )  GO TO 4580                                   !(ERROR?)YES, BYE.
        IF(LPDP)GO TO 44220
        MFLG(18)=0                                                        !SET SHIFT LOGIC NOT MODIFIED FLGS.
        MFLG(19)=0
        GO TO 4540                                                        !DONE.
C
44220   WRITE(6,7220)SNAME
        WRITE(6,5000)(SCOM(IT),IT=1,NWRCNT(SCOM,16))
        WRITE(6,7240)NUMG
        NUMESL=(NUMG-1)*2
        DO 44260 IG=1,NUMESL
        NP=NSPTS(IG)
        WRITE(6,7260)IGF(IG),IGT(IG),SHFTIM(IG)
        IF(IVAC)GO TO 44230
        IF(LDTH)WRITE(6,7270)(SHFTPT(IP,IG),IP=1,NP)
        IF(.NOT.LVAC.AND..NOT.LDTH)WRITE(6,7280)(SHFTPT(IP,IG),IP=1,NP)
CCC     WRITE(6,7280)(SHFTPT(IP,IG),IP=1,NP)             !KILLED OCT.22,1980
        GO TO 44240
44230   WRITE(6,7300)(SHFTPT(IP,IG),IP=1,NP)
44240   IF(LENG)WRITE(6,7320)(SHFTRP(IP,IG),IP=1,NP)
        IF(PARAB)WRITE(6,7340)(SHFTRP(IP,IG),IP=1,NP)
        IF(.NOT.LENG .AND. .NOT.PARAB)
        2        WRITE(6,7360)(SHFTRP(IP,IG),IP=1,NP)
        IF(.NOT.LDETNT)GO TO 44260
        WORD1=H1HFOT
        IF(LDETV)WORD1=HVACUO
        WORD2=HCOTPU
        IF(LENG)WORD2=HENGIN
        IF(PARAB)WORD2=FVEHIC
        WRITE(6,7380)WORD1,DETPT(IG),WORD2,DETRPM(IG)
44260   CONTINUE
        GO TO 4560
C
C=========================================================================
C     LOAD AXLE DATA FROM PARTS DATA FILE
C
4440    LVEMAX=.FALSE.
        CALL DSK
C
        IF(IPRNT.EQ.-10)GO TO 4580
        IF(LPDP)GO TO 4560
        NPARTS(11)=1
        GO TO 4560
C
C=========================================================================
C     LOAD VEHICLE DATA FROM PARTS DATA FILE
C
4460     VNAME      = PNAME                                               !GET VEHICLE NAME.
         LVEMAX=.TRUE.
         CALL DSK                                                         !LOAD VEHICLE NAME.
         IF ( IPRNT.EQ.-10 )  GO TO 4580                                  !(ERROR?)YES, BYE.
         IF(LPDP)GO TO 4560
         DO 4480 I=4,9
4480     MFLG(I)=0                                                        !SET FLGS VEHICLE NOT MODIFIED.
         IF(LVNEW) GO TO 4540                                             !(NEW DATA FORMAT?)YES, DONE.
         NPARTS(9)=1                                                      !NO, FLG TIRE DATA LOADED.
         TNAME=HPVEHI                                                     !SET TIRE NAME TO "%VEHICLE".
         GO TO 4520                                                       !DONE.
C
C=========================================================================
C     LOAD TIRE DATA FROM PARTS DATA FILE
```

```
C
4500    TNAME=PNAME                                                    !GET TIRE NAME.
        CALL DSK                                                       !LOAD TIRE.
        IF(IPRNT.EQ.-10) GO TO 4560                                    !(ERROR?)YES, DONE.
        IF(LPDP)GC TO 4560
4520    MFLG(1)=0                                                      !FLG TIRE NOT MODIFIED.
        MFLG(2)=0
        MFLG(3)=0                                                      !DONE.
C
C=======================================================================
4540    NPARTS(IPRNT)=1                                                !FLG PART LOADED.
C
4560    IECCND=1                                                       !FLG NO ERROR.
        GO TO 160                                                      !DONE.
C
C     ERROR RETURN WHEN PART REQUESTEC NOT CN PARTS DATA FILE
C
4580    IECCND=2                                                       !? PART NOT FOUND.
        GO TO 280
C
4600    IECCND=3                                                       !? UNKNOWN PART TYPE.
        GO TO 300
C
C**********************************************************************
C
C    /STATUS/ CCMMAND - REPORT CURRENT VEHSIM STATUS
C
        ENTRY STSCMD(IECCND)                                           !DIALOG ENTRY.
        JCT=IECCND                                                     !GET UNIT NUMBER TO SEND STATUS REPORT TO.
        IECCND=1                                                       !ASSUME ERROR.
        GO TO 4660
C
4620    SKIP RECORD 4                                                  !LOOK NEXT COMMAND.
        READ(4,5060) DUMMY,WORD
        BACKSPACE 4
        IF(WORD.EC.HCCMND(2)) GO TO 160                        !IS IT /*SIMULATE/?)YES, SKIP STATUS COMMAND SIMULATE WILL DO IT.
4640    JCT=6                                                          !SEND STATUS REPORT TO LPT.
C
4660    WRITE(JCT,5600) VERID                                          !VERSION ID.
        IF(BATCH) WRITE(JCT,5620)                                      !MODE.
        IF(DIALCG) WRITE(JCT,5640)
        IF(PTY) WRITE(JCT,5660)                                        !SUBMODE.
        IF(TTY) WRITE(JCT,5680)
        WRITE(JCT,5700) SIMODE                                         !SIMULATION MODE.
        IF(LDYNA) WRITE(JCT,5720)                                      !(DYNAMOMETER SIM?)YES, REPORT IT.
C
        I=NWRCNT(TITLE,12)                                             !GET NUMBER OF WORDS IN TITLE.
        IF(I.GT.0) GO TC 4680                                          !(ANY TITLE?)YES.
        WRITE(JCT,5740)                                                !NO,SAY SO.
        GO TO 4700
4680    WRITE(JCT,5760) (TITLE(K),K=1,I)                               !REPORT TITLE.
C
4700    IF(ENG1) WRITE(JCT,5780) ENAME(1)                              !(ENG # 1 LOADED?)YES.
        IF(ENG1.AND.LDIES)WRITE(JCT,5750)                              ![615]
        IF(ENG2) WRITE(JCT,5800) ENAME(2)                              !(ENG # 2 LOADED?)YES.
        IF(ENG2.ANC.LDIES)WRITE(JCT,5750)                              ![615]
        IF( .NOT. (ENG1 .CR. ENG2) ) WRITE(JCT,5820)                   !(NO ENGS LOADED?)YES, REPORT IT.
        WRITE(JCT,5840) ((CNVTYE(I),CFVNAM(I)),I=1,2)                  !REPORT CONVERTERS.
        WRITE(JCT,5860) VNAME                                          !REPORT VEHICLE.
        IF(.NOT.LVEHAX.ANC.NPARTS(11).EQ.1) WRITE(JCT,5960)AXNAME      !(AXLE NOT TIED TO VEHICLE AND AXLE LOADED?)YES. REPORT IT.
        IF(NPARTS(10).EQ.0)GO TO 4720
        WRITE(JCT,5880) TRNAM
C
4720    IF(NUMG.GT.0) GO TO 4740                      !(ANY GEARS?)YES.
        WRITE(JCT,5900)                               !NO, REPORT IT.
        GO TO 4820
4740    ASSIGN 4760 TO GWR                            !ASSUME NO ENGS LOADED.
        IF(ENG1 .OR. ENG2) ASSIGN 4780 TO GWR    !(ANY ENGS?)YES, USE WRITE STATEMENT THAT GIVES ENG ASSIGNMENTS.
        DO 4900 I=1,NUMG                              !LOOP THRU LOADED GEARS.
        GO TO GWR                                     !GO TO PROPER WRITE STATEMENT.
4760    WRITE(JCT,5920) I,GNAME(I)
        GO TO 4800
C4780   WRITE(JCT,5940) I,GNAME(I),ENAME(JENG(I))
4780    CONTINUE
        IF(FICD) GO TO 4785
C       NOT SPLIT TORQUE
        WRITE(JCT,5940) I,GNAME(I),ENAME(JENG(I))
        GO TO 4800
4785    CONTINUE
C       SPLIT TORQUE
        IF(I.NE.3) WRITE(JCT,5940) I,GNAME(I),ENAME(JENG(I))
        IF(I.EQ.3) WRITE(JCT,5945) I,GNAME(I),ENAME(JENG(I))
4800    CONTINUE                                      !NEXT.
C
4820    IF(NACC.GT.0) WRITE(JCT,5980) ((I,ANAME(I)),I=1,NACC)    !(ANY ACCES?)YES, REPORT THEM.
        IF(NACC.LE.0) WRITE(JCT,6000)                 !(ANY ACCES?)NO, REPORT IT.
        WRITE(JCT,6020) DNAME                         !DRIVING SCHEDULE.
        IF(ITPSPD)WRITE(JCT,7420)                     ![623]  WRITE(JCT,6040) SNAME            !SHIFT LOGIC.
        WRITE(JCT,6040) SNAME                         !SHIFT LOGIC 7-31-80 J. DOLAN
        WRITE(JCT,6060) RNAME                         !ROUTE NAME.
        WRITE(JCT,6080) TNAME                         !TIRE NAME.
C
        IL=0                                          !ASSUME NO GEARS LOCKED UP.
        IU=0                                          !ASSUME NO GEARS UNLOCKED.
        DO 4860 K=1,NUMG                              !LOOP THRU GEARS.
        IF(LOCKUP(K)) GO TO 4840                      !(GEAR LOCKED UP?)YES.
        IU=IU+1                                       !NO INC COUNT OF UNLOCKED GEARS.
        IDATA(IU)=K                                   !SAVE GEAR #.
        GO TO 4860                                    !NEXT.
4840    IL=IL+1                                       !INC CNT OF LOCKED UP GEARS.
        LOCKG(IL)=K                                   !SAVE GEAR #.
4860    CONTINUE                                      !NEXT.
C
        IF(IL.EQ.0) WRITE(JCT,6100)                   !(ANY GEARS LOCKED UP?)NO REPORT IT.
        IF(IL.GT.0) WRITE(JCT,6120) (LOCKG(I),I=1,IL) !(ANY GEARS LOCKED UP?)YES, REPORT IT.
        IF(IU.EQ.0) WRITE(JCT,6140)                   !(ANY GEARS UNLOCKED?)NO, REPORT IT.
        IF(IU.GT.0) WRITE(JCT,6160) (IDATA(I),I=1,IU) !(ANY GEARS UNLOCKED?)YES, REPORT THEM.
C
        WRITE(JCT,6180) SLIMIT,ALIMN                  !REPORT LIMIT PRINT.
        WRITE(JCT,6200) SDEBUG,DBEGIN,DSTOP           !REPORT DEBUG STATUS.
C
        WORD=HON                                      !ASSUME TTY OUTPUT ON.
        IF(LIMTTY) WORD=HOFF                          !(TTY OUTPUT OFF?)YES.
        WRITE(JCT,6220) WORD                          !REPORT TTY OUTPUT STATUS.
        WORD=HOFF                                     !ASSUME LPT OUTPUT OFF.
        IF(LLPT)WORD=HON                              !(LPT OUTPUT ON?)YES.
        WRITE(JCT,6240)WORD                           !REPORT LPT OUTPUT STATUS.
C
        IMCNT=0                                       !ASSUME NO MODIFICATIONS.
        DO 4880 I=1,NMCD                              !LOOP THRU ALL MODS EXCEPT DYNA.
```

```
         IF(I.EQ.21.OR. (I.GE.23.AND.I.LE.25))GO TO 4880            !SKIP DYNA. [615] [621] SKIP MODI-LOCKUP [623]TOPSPEED
         IF(MFLG(I).EQ.0) GO TO 4880                                 !(MODIFIED VALUE?)NO.
         IMCNT=IMCNT+1                                               !YES, INC CNT OF MODIFIED VALUES.
         WRITE(JCT,6260) MPOD(I),CIDVAL(I),VALNEW(I)                 !REPORT MODIFICATION.
4880     CONTINUE                                                    !NEXT.
         IF(IMCNT.EQ.0) WRITE(JCT,6280)                              !(ANY MODIFICATIONS?)NO, REPORT IT.
         IF(.NOT.IEDLOK)GO TO 4890                                   ![621]
         WRITE(JCT,7410)MDLKGH,MDLKBP                                ![621]
C
4890     IF(LSIMUL) GO TO 900                                        !/*SIMULATE/ CALL TO /*STATUS/?)YES, GO SIMULATE.
         IECCND=0                                                    !NO, FLG NO ERR.
         GO TO 160                                                   !DONE.
C
C*******************************************************************
C
C      SET PROPER *ENDE* FLAG AND RETURN
C
4900     ENDE   = .TRUE.                                             !FLG EOF FOR VEHSIM CONTROL FILE.
         GO TO 4940
C
4920     ENDE   = .FALSE.                                            !FLG NO EOF FOR VEHSIM CONTROL FILE.
C
4940     ENDC=ENDE                                                   !PASS EOF COND BACK THROUGH ARG BLOCK.
         IF(.NOT.ENDE) GO TO 4960                                    !(EOF CTR FILE?)NO.
         WRITE(5,6300) CTRDEV,CTRFIL,CTRPPN(1),CTRPPN(2)             !YES REPORT IT.
         WRITE(6,6300) CTRDEV,CTRFIL,CTRPPN(1),CTRPPN(2)             !REPORT LPT.
4960     RETURN                                                      !DONE, BYE.
C
C*******************************************************************
C
C      FORMAT STATEMENTS
C
4980     FORMAT(/' VEHSIM CONTROL FILE'1X19(1H*)//)
5000     FORMAT (16A5)
5020     FORMAT(1XA1,16A5)
5021     FORMAT(1X,A5)
5040     FORMAT(1X16A5)
5060     FORMAT (A1,16A5)
5090     FORMAT(/' ? INPEAT-COMMAND OR DATA CARD IN ERROR ---'16A5/)
5100     FORMAT(/' ? INPEAT-DUE TO VEHSIM CONTROL FILE ERRORS DETECTED'
         1/10X'DURING EXECUTION OF "'A5'" COMMAND SWITCHING TO'
         1,' DIALOGUE MODE.'
         2/10X'GIVE "CONTINUE" COMMAND TO RESUME PROCESSINGOF VEHSIM'
         3,' CONTROL FILE.'/)
5120     FORMAT(/' /*STOP/ COMMAND ENCOUNTERED IN VEHSIM CONTROL FILE.')
5140     FORMAT (18XA5,7X2F10.1)
5160     FORMAT (12X12A5,A8)
5180     FORMAT (12A5,A8)
5200     FORMAT ('1? INPEAT-/*SIMULATE/ COMMAND IGNORED  - '
         1,' THE FOLLOWING PARTS ARE UNDEFINED'//)
5220     FORMAT (/52XA5)
5240     FORMAT(/' ? INPEAT- FIRST SECTION OF 'A5,3XA10' WAS'
         1,' NOT LOADED'/10X'AND RELOAD ATTEMPT FAILED.')
5260     FORMAT (' ?INPEAT-/*SIMULATE COMMAND BYPASSED DUE TO PREV'
         1,'IOUS CONTROL CARD ERRORS'/'     ERROR COUNT='I//)
5280     FORMAT (18XA5,3(7XA5))
5290     FORMAT(' %Simulating a ',A5,' with the CAR version of VEHSIM')
5300     FORMAT(/' [SIMULATING]')
5320     FORMAT(/' % INPEAT - NO TORQUE CONVERTERS LOADED BUT'
         1,/13X'ARE ALL GEARS LOCKED UP.'//' INPEAT-ATTEMPING TO LOAD'
         2,' DEFAULT TORQUE CONVETERS CCDY2 AND COCY2')
```

```
5340      FORMAT (18XA5,7XF6.1,6XA5,7XF6.1)
5342      FORMAT(18X,A5,7X,2F6.1)
5360      FORMAT (18XA5,7X20I2)
5380      FORMAT (18XA10,2XA5,7X,A5,7X,A5,7X,A5)
5400      FORMAT (A5,1X12F6.1)
5420      FORMAT (A5,7X11F6.1)
5440      FORMAT (12X4(A5,1X),12X5(A5,1X))
5460      FORMAT (A5,7X8F6.1)
5490      FORMAT(/' % INPFAT-ACCESSORY 'A10' NOT LOADED.')
5500      FORMAT(/' % INPFAT-NO ACCESSORIES LOADED.')
5520      FORMAT(/' PARTS ZERCED:')
5540      FORMAT (6XA10,2XA5,17X20I2)
5560      FORMAT(6XA10,2XA5,1XA5)
5590      FORMAT (6XA10,2XA5,11X,I1,5X,20I2)
5600      FORMAT('1 VEHSIM 'A5,2XC2'('02') STATUS REPORT'/2X,34('-'))
5620      FORMAT(/' BATCH MCDE'$)
5640      FORMAT(/' DIALOGUE MODE')
5660      FORMAT('+'14X'/PTY')
5680      FORMAT('+'14X'/TTY')
5700      FORMAT(' SIMULATICN MCDE-'A5)
5720      FORMAT('+'21X'/DYNAMOMETER')
5740      FORMAT(' NO RUN TITLE')
5750      FORMAT('+',23X,'/DIESEL')
5760      FORMAT(' RUN TITLE-'12A5)
5780      FORMAT(' ENGINE(1)-'A10)
5800      FORMAT(' ENGINE(2)-'A10)
5820      FORMAT(' ENGINE-NCT LOADED')
5840      FORMAT(1XA5' CONVERTER-'A10)
5860      FORMAT(' VEHICLE-'A10)
5880      FORMAT(' TRANSMISSION - 'A10)
5900      FORMAT(' GEARS-NCT LOADED')
5920      FORMAT(' GEAR #'I2'-'A10' UNASSIGNED')
5940      FORMAT(' GEAR #'I2'-'A10' ASSIGNED TO ENGINE-'A10)
5945      FORMAT(' GEAR #'I2'-'A10' ASSIGNED TO ENGINE-'A10,
      1   ' SPLIT TORQUE')
5960      FORMAT(' AXLE - ',A10)
5980      FORMAT(' ACCESSORY #'I3'-'A10)
6000      FORMAT(' ACCESSORIES-NOT LOADED')
6020      FORMAT(' DRIVING SCHEDULE-'A10)
6040      FORMAT(' SHIFT LOGIC-'A10)
6060      FORMAT(' ROUTE-'A10)
6080      FORMAT(' TIRE-'A10)
6100      FORMAT(' NO GEARS LOCKED UP')
6120      FORMAT(' GEARS LOCKED UP-'20I3)
6140      FORMAT(' NO GEARS UNLOCKED')
6160      FORMAT(' GEARS UNLOCKED-'20I3)
6180      FORMAT(' LIMIT PRINT-'A5,G)
6200      FORMAT(' DEBUG-'A5,2G)
6220      FORMAT(' TTY OUTPUT-'A3)
6240      FORMAT(' LPT OUTPUT-'A3)
6260      FORMAT(1XA5' MODIFIED FROM 'G10.4' TO 'G10.4)
6280      FORMAT(' NO MODIFICATIONS')
6300      FORMAT(/' END OF VEHSIM CONTROL FILE 'A6':'A10'['05','03']')
6320      FORMAT(' ENTER TRANSMISSION NAME: '$)
6340      FORMAT(A10)
6360      FORMAT(' ? ',A10,' NOT A VALID NAME.')
6380      FORMAT(' HOW MANY GEARS? (MAX. 20) '$)
6400      FORMAT(' ENTER NAME FOR GEAR #',I2': '$)
6420      FORMAT(' ENTER COMMENT: (1 LINE) '$)
6440      FORMAT(18X,A10,2X,I)
6460      FORMAT(12X,A10)
```

```
7000    FORMAT('*ACCESSORY',T19,A10)
7020    FORMAT('INERTIA',T13,F6.3)
7040    FORMAT('TORQUE',T13,10F6.3)
7060    FORMAT('ENGINE RPM',T13,10F6.1)
7080    FORMAT('*ENGINE',T13,A10,T31,A5,T43,A5,T55,A5,T67,A5)
7100    FORMAT('DATA',T13,2F6.3,F6.0,2F6.1,2F6.3,F6.0)
7120    FORMAT('SPEED POINT',T13,F6.1)
7140    FORMAT('LOAD POINT',T13,10F6.2)
7160    FORMAT('FUEL RATE',T13,10F6.2)
7180    FORMAT('THROTTLE',T13,10F6.2)
7200    FORMAT('MANIFOLD',T13,10F6.2)
7220    FORMAT('*SHIFT LOGIC',T19,A10)
7240    FORMAT('NUMB GEARS',T13,I5,'.')
7260    FORMAT('SHIFT LINE',T13,I5,'.',I5,'.',F6.3)
7270    FORMAT('DTHROTTLE',T13,10F6.2)
7280    FORMAT('THROTTLE',T13,10F6.2)
7300    FORMAT('VACUUM',T13,10F6.2)
7320    FORMAT('ENGINE RPM',T13,10F6.1)
7340    FORMAT('VEHICLE MPH',T13,10F6.2)
7360    FORMAT('OUTPUT RPM',T13,10F6.1)
7380    FORMAT('DETENT OVERRIDE',T19,A5,T31,F6.2,T43,A5,T55,F6.1)
7390    FORMAT(' Enter gear # to be locked at',F6.0,':'$)
7400    FORMAT(10G)
7410    FORMAT(' LOCKUP GEAR',I3,' if engine RPM >=',I5)
7420    FORMAT('+',27X,'/TOP SPEED')
10400   FORMAT(' Enter new output file: '$)
C
        END
```

```
C
      INCLUDE 'COMMS/NOLIST'
C
      DOUBLE PRECISION HINPDI
C
      LOGICAL ENDET,LPAUSE,LTRANS
C
      DIMENSION HPART(14),CNVTYP(3),IPCRS(14),NHPART(14)
C
      DATA (HPART(I),I=1,NPART)/'ACCES','CONVE','DRIVI','ENGIN','GEAR '
     1,'ROUTE','SHIFT','VEHIC','TIRE','TRANS','AXLE'/,NHPART/14*2/
C
      DATA CNVTYP/'DRIVICOASTTORQU'/,YES/'Y'/,HNO/'N'/,DIA/'D'/
     1,HON/'ON'/,HOFF/'OFF'/
C
      DATA   HCCMND/ 5HUSE   ,5HSIMUL ,5HMODIF ,5HPRINT ,5HLIMIT ,
     1               5HDUMP  ,5HREMAP ,5HLOCKD ,5HUNLOC ,5HTITLE ,
     2               5HDIREC ,5HSTATU ,5HZERO  ,5HASK   ,5HTELL  ,
     3               5HBATCH ,5HENGIN ,5HFULL  ,5HS.R.  ,5HVEHIC ,
     4               5HGEAR  ,5HACCES ,5HSHIFT ,5HDRIVI ,5HROUTE ,
     5               5HTIRE  ,5HDEBUG ,5HRESET ,5HTTY   ,5HDDT   ,
     6               5HEDUMP ,5HDELET ,5HEXIT  ,5HDIALO ,5HCONTI ,
     7               5HIPT   ,5HHELP  ,5HOUTPU ,5HTRANS ,5HAXLE  ,
     8               5HSPLIT /
     9,NCCMND/2*2,4,3*2,3,2,2,3*3,2,3,9*2,3,2
     1,3*2,2,3,2,3,2,3,3,2,2,1,2,2,3/
     2,(IPCRS(I),I=1,NPART)/5,2,6,1,4,8,7,3,9,10,11/
     3,HINPDI/'INPDIA'/
C
C*********************************************************************
C
      LPAUSE=.FALSE.                                                  !INITIALIZE NOT IN PAUSE MODE.
      GO TO 9
C
      ENTRY INPESE(ISEGT)                                             !ENTRY FOR ¬P FROM SIMSTS.
      LPAUSE=.TRUE.                                                   !FLG IN ¬P(PAUSE) MODE AND INHIBIT CERTAIN COMMANDS.
      WRITE(5,1047) ISEGT                                             !REPORT IN ¬P MODE AND WHAT DRS SEG.
    9 IECCMD=0                                                        !ASSUME NO ERROR.
      LASK=.FALSE.                                                    !INITIAL IN TELL MODE.
   10 ICMD=0                                                          !SET WAIT COMMAND.
   11 WRITE(5,1041)                                                   !PROMPT.
      READ(5,1023,END=920,ERR=990) CCMND,FSPECS                       !GET COMMAND.
      IF(CCMND.EQ.HBLANK) GO TO 11                                    !(EXTRANEOUS "<CR>"?)YES,TRY AGAIN.
      IF((CCMND.AND.WMASK(1)) .NE. ('@'.AND.WMASK(1)))GO TO 13        !(COMMAND FILE?)NO
      REREAD 1021, FSPECS
      ICMD=16
      GO TO 206                                                       !HANDLE BATCH COMMAND FILE
C
13    CALL LOOKUP(CCMND,-ICHCNT(CCMND,1),HCOMND,NCCMND                !(VALID COMMAND?)
     1,41,ICMD,HINPDI,$20)                                            !YES.
      GO TO 10                                                        !NO, GO PROMPT.
C
   17 WRITE(5,1043) HCCMND(ICMD)                                      !%BATCH COMMAND NOT IMPLEMENTED FOR DIALOGUE MODE.
      GO TO 10                                                        !GO PROMPT.
C
   20 PNAME=FSPECS(1)
      GO TO ( 100,203,231, 17,241,300, 17,211,221,400
     1,      310,251,270, 21, 22,206, 17, 17, 17, 17
     2,       17, 17, 17, 17, 17, 17,261, 40,280, 60
     3,      320,290, 30, 10, 50,500,550,700,600, 17,800) ,ICMD       !GO HANDLE COMMAND.
```

```
C
  21    LASK=.TRUE.                                                     !SET TO ASK MODE.
        GO TO 100                                                       !GO ASK QUESTION FOR SIMULATION.
C
  22    LASK=.FALSE.                                                    !SET TO TELL MODE.
        GO TO 10                                                        !GO PROMPT.
C
  30    CALL EXIT                                                       !EXIT TO MONITER.
        GO TO 900                                                       !IMPOSSIBLE ERR.
C
  40    CALL RESET                                                      !RE-INITIALIZE FOROTS.
        GO TO 900                                                       !IMPOSSIBLE ERR.
C
  50    IF(LPAUSE) GO TO 207                                            !(IN PAUSE MODE?)YES,DISALLOW /*CONTINUE/ COMMAND.
        IECOND=3                                                        !NO,FLG INPBAT TO CONT PROCESSING BATCH CTR FILE.
        RETURN                                                          !DONE, BYE.
C
60      IF(.NOT.PTY)CALL GETDDT                                         !ENTER DDT IF LOADED AND NOT BATCH
        GO TO 10
C
C*****************************************************************************
C
C  *USE
C
  100   IF(LASK) GO TO 101                                              !(ASK MODE?)YES.
  103   WRITE(5,1044)                                                   !ASK PART TYPE.
        READ(5,1023,END=920,ERR=990) UT                                 !GET PART TYPE.
        IF(UT.EQ.HBLANK) GO TO 10                                       !(DONE?)YES, GO PROMPT.
        CALL LOCKUP(UT,-ICRCNT(UT,1),HPART,NHPART,NUMPAR
        2,IPRNTT,EINPDI,$102)                                           !LOOK UP PART TYPE AND SET PTR(VALID?)YES.
        WRITE(5,1042) UT                                                !NO,?UNKNOWN PRT TYPE.
        GO TO 103                                                       !GO TRY AGAIN.
C
  101   DO 199 IP=1,NUMPAR
        IF(IP.EQ.5)GO TO 199
        ICMD=1                                                          !SET COMMAND PTR.
        IPRNTT=IP                                                       !SET PART TYPE PTR.
C
C  NORMAL BRANCH
C
  102   GO TO (110,120,130,140,1510,160,170,180,190,1500,1520),IPRNTT   !GO PROCESS /*USE/ FOR SPECIALIZED PART TYPE.
C
C  FORMAT I/O ERROR BRANCH RETURN
C
  104   GO TO (118,122,131,141,151,161,171,181,191),IPRNTT              !ON "ERR=" GET BACK TO WHERE WE WERE.
C
C========================================================================
C
C  *USE ACCESSORY
C
  110   IPRNTT=1                                                                !FLG ACCES.
  111   IR=1                                                            !PTR TO NEXT STATEMENT IN CASE OF ERR=.
  115   WRITE(5,1001)                                                   !ASK # OF ACCES TO BE USED.
        READ(5,1002,END=920,ERR=990) NUM                                !GET #.
        IF(NUM.LT.0 .OR. NUM.GT.20) GO TO 111                           !(LEGAL?)NO,ERR.
        IF(NUM.EQ.0) GO TO 118                                          !(ANY ACCES TO LOAD?)NO, DONE.
C
        IR=2                                                            !SET PTR TO ST. # 112.
        DO 117 I=1,NUM                                                  !ASK FOR NEEDED # OF ACCES.
  112   WRITE(5,1003) I                                                 !ASK ACCES NAME.
        READ(5,1004,END=920,ERR=990) ENAME                              !GET NAME.
        GO TO 182                                                        !GO GET IT.
  117   CONTINUE                                                         !NEXT.
        GO TO 198                                                        !DONE.
C
  118   GO TO (115,112),IR                                               !ON FORMAT ERR= RECOVERY COMPLETE.
C
C=========================================================================
C
C  *USE CONVERTER
C
  120   IPRNTT=2                                                                 !SET COMMAND PTR.
        DO 126 M=1,2                                                     !NEED 2 CONVERTERS COAST AND DRIVE.
  122   WRITE(5,1006) CNVTYP(M)                                          !ASK CONV NAME.
        READ(5,1004,END=920,ERR=950) FNAME                               !GET NAME.
        GO TO 182                                                        !GO GET IT.
  126   CONTINUE                                                         !(DONE?)NO.
        GO TO 198                                                        !YES.
C
C=====================================================================
C
C  *USE DRIVING SCHEDULE
C
  130   IPRNTT=3                                                                 !SET COMMAND PTR.
  131   WRITE(5,1008)                                                    !ASK FOR DRS NAME.
        READ(5,1004,END=920,ERR=990) FNAME                               !GET NAME.
        GO TO 182                                                        !GO GET IT.
C
C=====================================================================
C
C  *USE ENGINE
C
  140   IPRNTT=4                                                                 !SET COMMAND PTR.
  141   WRITE(5,1010)                                                    !ASK FOR ENG #,NAME,AND GEAR ASSIGNMENTS.
        READ(5,1011,END=920,ERR=950) IENG,FNAME,IDATA                    !GET THEM.
        IF(IENG.EQ.0) GO TO 148
        IF(IENG.LT.1 .OR. IENG.GT.2) GO TO 141
C
  142   LOCKG(1)=IENG                                                    !PASS ENG #.
        GO TO 182                                                        !GO GET IT.
C
  144   WRITE(5,1012)                                                    !ENG NOT FOUND. ASK FOR CORRECT NAME.
        READ(5,1004) FNAME                                               !GET NAME.
        IF(FNAME.EQ.DBLANK) GO TO 141                    !("<CR>"?)YES, GO ASK FOR ALL INFO ON ENG.[TO PROVIDE EXIT FROM LOOP].
        GO TO 142                                                        !GO TRY IT.
C
  146   NENGI=NENGI+1                                                    !INC ENG LOADED CNT.
        GO TO 141                                                        !SEE IF WE WANT A SECOND ENG.
C
  148   IF(NENGI.EQ.0) GO TO 141                                         !(GOT 1 ENG LOADED?)NO,GO ASK AGAIN.
        GO TO 198                                                        !DONE.
C
C=====================================================================
C
C  *USE TRANS
C
 1500   IPRNTT=10                                                               !SET POINTER.
C
        WRITE(5,1080)
        READ(5,1004,END=920,ERR=990)FNAME                                !GET NAME.
C
        IF(FNAME.EQ.DBLANK)GO TO 150                                     !(BLANK?)YES, GO TO GEARS, OLD METHOD.
```

```
C
      GO TO 182                                                        !NO, GOT NAME, SO GO LOAD IT.
C
C=========================================================================
C
C   *USE GEAR
C
 1510 WRITE(5,1082)
      READ(5,1002,END=920,ERR=950) IDATA(1)
C
      IF(IDATA(1).LT.1 .OR. IDATA(1).GT.20)GO TO 1510
C
      WRITE(5,1014) IDATA(1)
      READ(5,1004,END=920,ERR=990) PNAME
C
      IMOD=1
      GO TO 182
C
C
  150 IPRNTT=5                                                                  !FLG GEAR USE.
  151 IR=1                                                             !FLG ERR= BRANCH TO NXT ST.
  152 WRITE(5,1013)                                                    !ASK # OF GEARS TO BE USED.
      READ(5,1002,END=920,ERR=950)  NUM                                !GET #.
      IF(NUM.LT.1 .OR. NUM.GT.20)  GO TO 152                           !(LEGAL #?) NO, TRY AGAIN TURKEY.
C
      IMOD=0                                                           !FLG DO LOOP IN CONTROL.
      IR=2                                                             !FLG ERR= BR BACK INTO LOOP.
      DO 157 I=1,NUM                                                   !LOAD GEARS.
      IDATA(1)=I                                                       !ASK GEAR #.
  154 WRITE(5,1014)  I                                                 !ASK NAME FOR GEAR # I.
      READ(5,1004,END=920,ERR=990)  PNAME                              !GET NAME.
      GO TO 192                                                        !GO GET IT.
C
  156 IF(IMOD.EQ.1)  GO TO 155                                         !(DO LOOP IN CTRL?) NO.
      GO TO 154                                                        !YES.
C
 153  IF(IMOD.EQ.1)  GO TO 155                                         !(DO LOOP IN CTRL?) NO.
  157 CONTINUE                                                         !YES, NEXT.
C
      IMOD=1                                                           !FLG DO LOOP NOT IN CONTROL,SO BELOW CODE CAN BRANCH INTO IT
.
      IR=3                                                             !FLG NXT ST ON FOR ERR= RECOVERY.
  155 WRITE(5,1039)                                                    !ASK FOR GEAR # AND NAME.
  158 READ(5,1011,END=920,ERR=990)  IDATA(1),PNAME                     !GET # AND NAME.
      IF(IDATA(1).EQ.0 .AND. PNAME.EQ.BLANK)  GO TO 198                !(DONE?) YES.
      GO TO 182                                                        !NO, GOT REQUEST. GO GET IT.
C
  159 GO TO (152,154,155),IR                                           !WHICH READ STATEMENT CAUSED ERR?
C
C=========================================================================
C
C   *USE AXLE
C
 1520 IF(LVEHAX)GO TO 158
      IPRNTT=11
C
      WRITE(5,1084)
      READ(5,1004,END=920,ERR=990) PNAME                               !GET NAME
C
      GO TO 182
C
```

```
C=========================================================================
C
C  *USE ROUTE
C
  160   IPRNTT=6                                                          !FLG ROUTE.
  161   WRITE(5,1016)                                                 !ASK ROUTE NAME.
        READ(5,1004,END=920,ERR=990) PNAME                            !GET ROUTE NAME (EOF OR ERR?)YES.
        GO TO 182                                                     !GO LOAD IT.
C
C=========================================================================
C
C  *USE SHIFT LOGIC
C
  170   IPRNTT=7                                                          !FLG SHIFT LOGIC.
  171   WRITE(5,1018)                                                 !ASK SHIFT LOGIC NAME.
        READ(5,1004,END=920,ERR=990) PNAME                            !GET NAME (EOF OR ERR?) YES.
        GO TO 182                                                     !GO LOAD IT.
C
C=========================================================================
C
C  *USE VEHICLE
C
  180   IPRNTT=8                                                          !FLG VEHICLE.
  181   WRITE(5,1020)                                                 !ASK VEHICLE NAME.
        READ(5,1004,END=920,ERR=990) PNAME                            !GET NAME.(EOF OR ERR?)YES.
  182           IF(PNAME.EQ.DBLANK) GO TO 184                         !(PART NAME BLANK?)YES, ERR ILLEGAL NAME.
        IECCND=IPCRS(IPRNTT)                                          !SET PART TYPE PTR AS USED BY OTHER ROUTINE.
        CALL USECMD(IECCND)                                           !GO LOAD PART.
        GO TO (185,184,900),IECCND                                    !GOT IT /DIDN'T/OOPS.
  184   CALL NOPART(5,PNAME,IPCRS(IPRNTT),CNVTYP(N))                  !?NO SUCH PART ON DB.
        GO TO (112,122,131,144,156,161,171,181,191,1500,1520),IPRNTT  !OH WELL, LETS TRY AGAIN.  BACK TO WHERE WE WERE.
C
  185   GO TO(117,126,198,146,153,198,198,198,198,198,198),IPRNTT     !GOT IT NEXT.
C
C=========================================================================
C
C  *USE TIRE
C
  190   IF(LASK.AND. .NOT.LVNEW)GO TO 199
        IPRNTT=9                                                          !FLG TIRE.
  191   WRITE(5,1049)                                                 !ASK TIRE NAME.
        READ(5,1004,END=920,ERR=990) PNAME                            !GET NAME (EOF OR ERR?) YES.
        GO TO 182                                                     !GO LOAD IT.
C
  198   IF(ICMD.EQ.2) GO TO 205                                       !(SIMULATE ERR CALL?)YES.
        IF(.NOT. LASK) GO TO 103                                      !TELL MODE?)YES.
  199   CONTINUE                                                      !NO, ASK PART TYPE.
C
C*************************************************************************
C
C   *SIMULATE COMMAND
C
  200   IF(LPAUSE) RETURN                                             !(IN ¬P MODE?)YES, RETURN TO SIMSTS AND CONTINUE SIMULATION.
        ICMD=2                                                        !SET COMMAND PTR.
        GO TO 251                                                     !GO SEND STATUS TO JCT.
  201   WRITE(5,1022)                                                 !IN ASK MODE, WHAT NEXT. SIM "Y" OR "N" OR GO BATCH.
        READ(5,1023,END=920,ERR=990) ANS
        IF(ANS.EQ.ANO) GO TO 209                                      !NO SIM. ASK ADDITIONAL QUEST.
        IF(ANS.EQ.YES) GO TO 203                                      !YES SIM, GO DO IT
        IF(ANS.NE.DIA) GO TO 201                                      !UNKNOWN ANS TRY. AGAIN.
        BATCH=.FALSE.                                                 !FLAG NO BATCH.
```

```
      DIALOG=.TRUE.                                                     !FLAG DIALOG TRUE.
      GO TO 9
C
 203  IF(LPAUSE) RETURN                                                 !ANS YES.
      LTYOLD=LIMTTY                                                     !SAVE TTY PRT STATUS.
      CALL SIMCMD ( IECOND )                                            !PERFORM SIMULATION.
      LASK=.FALSE.
      LPAUSE=.FALSE.                                                    !RESET ¬P FLG.
      LIMTTY=LTYOLD                                                     !RESTORE ORIGINAL TTY PRI STATUS.
      GO TO (10,208,204),IECOND                          !WHAT HAPPENED?(NO ERR SIM DONE/NO SIM BECAUSE OF ERR/NO SIM PARTS MISSING).
C
C  PARTS MISSING ON SIMULATE COMMAND
C
 204  WRITE(5,1035)                                                     !?PARTS MISSING.
      DO 205 I=1,NUMPAR                                                 !SEARCH ERR TABLE.
      IF(I.EQ.10)GO TO 205                                              !WHO CARES ABOUT TRANSMISSION - ONLY GEARS COUNT.
      IF(I.EQ.11.AND.LVEHAX)GO TO 205                                   !NO AXLE BUT AXLE TIED TO VEHICLE, GO ON.
      IF(IDATA(I).EQ.0) GO TO 205                                       !(PART LOADED?)YES.
      IPRNTT=I                                                          !NO, GET PART MISSING PTR.
      GO TO 102                                                         !GO ASK FOR IT.
 205  CONTINUE                                                          !NEXT.
      GO TO 201                                                         !DONE, LETS GIVE IT ANOTHER TRY.
C
 206  IF(LPAUSE) GO TO 207                                              !(IN ¬P MODE?)YES.
      IF(ICMD.EQ.16) IECOND=2
      BATCH=.TRUE.                                                      !FLG BATCH.
      DIALOG=.FALSE.                                                    !TURN OFF DIALOGUE.
      RETURN                                                            !DONE HERE FOR NEW, BYE.
C
 207  WRITE(5,1046) CCMND                                               !?ILL COMMAND IN ¬P MODE.
      GO TO 10                                                          !GO PROMPT.
C
C  UNRESOLVED "NORUN" ERRORS DETECTED
C
 208  WRITE(5,1031)                                                     !REPORT FATAL ERROR ON TRY TO SIMULATE.
      GO TO 275                                                         !BEYOND HELP, GO ZERO AND TRY AGAIN.
C
 209  IF(.NOT. LASK) GO TO 10                                           !(TELL MODE?)YES, GO PROMPT.
C
C**********************************************************************
C
C  *LOCKUP CONVERTER GEAR
C
 210  ICMD=8                                                                    !FLG LOCKUP COMMAND.
 211  WRITE(5,1024)                                                     !ASK GEAR #'S TO UNLOCK.
      READ(5,1002,END=920,ERR=990) LOCKG                                !GET #'S (EOF OR ERR?)YES.
      WORD=HCCMND(8)                                                    !PASS COMMAND NAME.
      CALL UAICMD(IECOND)                                               !TRY IT.
      IF(IECOND) 900,219,224                                            !(IMPOSS ERR/NOERR/ILL GEAR#?)
 214  WRITE(5,1025) IDATA(IECOND)                                       !?ILL GEAR#.
      GO TO 211                                                         !TRY IT AGAIN.
C
 219  IF(.NOT. LASK) GO TO 10                                           !(TELL MODE?)YES, GO PROMPT.
C
C**********************************************************************
C
C  *UNLOCK CONVERTER GEAR
C
 220  ICMD=9                                                                    !FLG UNLOCK COMMAND.
 221  WRITE(5,1026)                                                     !ASK GEAR #'S TO UNLOCK.
      READ(5,1002,END=920,ERR=990) LOCKG                                !GET #'S (EOF OR ERR?)YES.
      WORD=MCCMND(9)                                                  !PASS COMMAND NAME.
      CALL UALCMD(IECCND)                                             !TRY IT.
      IF(IECCND) 900,229,224                                          !(IMPOSS ERR/NO ERR/ILL GEAR #?).
  224 WRITE(5,1025) IDATA(IECCND)                                     !?ILL GEAR #.
      GO TO 220                                                       !TRY AGAIN.
C
  229 IF(.NOT. IASK) GO TO 10                                         !(TELL MODE?)YES, GO PROMPT.
C
C*************************************************************************
C
C  *MODIFY
C
  230 ICMD=3                                                          !FLG MODIFY COMMAND.
  231 WRITE(5,1027)                                                   !"*MODIFY:"
      READ(5,1040,END=920,ERR=990) DATA(1),WORD                       !GET VALUE AND VARIABLE.
      IF(WORD.EQ.MBLANK) GO TO 239                                    !(ANY MODIFY COMND?)NO, GET OUT.
      CALL MODCMD(IECCND)                                             !YES, LET'S TRY IT.
      GO TO (230,230,234,236),IECCND                                  !WHAT HAPPENED? $230 GOOD, ALL OTHERS ERR.
C
  234 WRITE(5,1030) WORD                                              !?PART TO MODIFY NOT LOADED.
      GO TO 230                                                       !TRY AGAIN.
C
  236 WRITE(5,1051) DATA(1),WORD                                      !?ILL VALUE.
      GO TO 230                                                       !TRY AGAIN.
C
  239 IF(.NOT. IASK) GO TO 10                                         !(TELL MODE?)YES, GO PROMPT.
C
C**********************************************************************
C
C  *LIMIT PRINT
C
  240 ICMD=5                                                                  !FLG LIM CMD.
  241 WRITE(5,1032)                                                   !ASK FOR OPTION.
      READ(5,1028,END=920,ERR=990) WORD,DATA(1)                       !GET OPTION AND ANY VALUE.
      IF(WORD.EQ.MBLANK) GO TO 249                                    !(ANY ANS?)NO,LEAVE CURRENT OPTION IN EFFECT.
      CALL LIMCMD(IECCND)                                             !YES, GO SET IT.
      IF(IECCND.EQ.1) GO TO 249                                       !(ERR?) NO.
      WRITE(5,1033) WORD                                              !YES, REPORT IT.
      GO TO 241                                                       !TRY AGAIN.
C
  249 IF(.NOT. IASK) GO TO 10                                         !(TELL MODE?)YES, GO PROMPT.
      GO TO 260                                                       !NO, NEXT.
C
C*************************************************************************
C
C  *STATUS
C
  250 ICMD=12                                                                 !FLG STATUS COMMAND.
  251 IECCND=5                                                        !PASS UNIT # TO SEND JCT.
      CALL STSCMD(IECCND)                                             !GO DO ?*STATUS/.
      IF(IPAOSP) WRITE(5,1047) ISEG1                                  !(¬P MODE?)YES, REPORT IT.
      IF(IECCND.EQ.1) GO TO 900                                       !(ERR?)YES, IMPOSSIBL.
      IF(.NOT. IASK) GO TO 10                                         !(TELL MODE?)YES, GO PROMPT.
      GO TO 201                                                       !GO ASK SIM?
C
C**************************************************************************
C
C  *DEBUG
C
  260 ICMD=27                                                                 !SET COMMAND PTR.
  261 WRITE(5,1036)                                                   !ASK FOR DEBUG OPTION.
```

```
        READ(5,1028,END=920,ERR=950)  WORD,DATA(1),DATA(2)                 !GET OPT AND LIMITS.
        CALL DEFCMD(IECCND)                                                !GO SET OPTION.
        GO TO (269,262,264),IECCND                                         !(OK/?UNKNOWN OPTION/?ILL VALUE ON LIMITS.)
C
  262   WRITE(5,1037) WORD                                                 !?UNKNOWN WORD.
        GO TO 261
C
  264   WRITE(5,1038)                                                      !?ILL VALUES ON LIMITS.
        GO TO 261
C
  269   IF(.NOT. IASK) GO TO 10                                            !(TELL MODE?)YES, GO PROMPT.
        GO TO 250                                                          !NO, NEXT IN ASK SEQ.
C
C***********************************************************************************
C
C  *ZERO
C
  270   ICMD=30                                                                     !SET COMMAND PTR.
        IF(LPAUSE) GO TO 207                                               !(¬P MODE?)YES, A DEF NO NO.
        IF(FNAME.EQ.HBLANK) GO TO 275                                      !)ZERO ACCES?)NO.
        UT=HPART(1)                                                        !YES, SET PART TYPE TO ACCES.
        UN=FNAME                                                           !GET ACCES NAME.
        GO TO 277                                                          !GO ZERO ACCES FROM CORE.
  275   UT=HBLANK                                                          !FLG ZERO ALL.
        NENGT=0                                                            !ENG LOADED CNT.
  277   CALL ZERCMD(IECCND)                                                !GO ZERO.
        GO TO 10                                                           !GO PROMPT.
C
C********************************************************************************
C  *TTY
C
  280   ICMD=29                                                                     !SET COMMAND PTR.
  281   WRITE(5,1045)                                                      !ASK FOR TTY MODE.
        READ(5,1028,END=920,ERR=990)  WORD                                 !GET MODE.
        IF(WORD.EQ.HON) GO TO 283                                          !(TTY ON?)YES.
        IF(WORD.NE.HOFF) GO TO 281                                         !(TTY OFF?)NO,?BAD MODE, GO ASK AGAIN.
        LIMTTY=.TRUE.                                                      !YES, SET TTY PRINT OFF.
        LTYCLD=.TRUE.                                                      !SAVE FOR RESET.
        GO TO 289
  283   LIMTTY=.FALSE.                                                     !SET TTY PRINT ON.
        LTYCLD=.FALSE.                                                     !SAVE FOR RESET.
        GO TO 289
  289   IF(.NOT. IASK) GO TO 10                                            !(TELL MODE?)YES, GO PROMPT.
        GO TO 260                                                          !NXT Q IN ASK SEQ.
C
C********************************************************************************
C
C  *DELETE
C
  290   ICMD=32                                                                     !SET COMMAND PTR.

  291   WRITE(5,1044)
        READ(5,1028,END=920,ERR=990)  OT
        IF(OT.EQ.HBLANK) GO TO 10
C
  295   WRITE(5,1048)
        READ(5,1004,END=920,ERR=990)  UN
        IF(UN.EQ.HBLANK) GO TO 295
        CALL DRPCMD(IECCND)
        GO TO (291,297),IECCND
C
```

```
  297    CALL NOPART(5,UN,IPCRS(IPRNTT),CNVTYP(J))
         GO TO 291
C*****************************************************************************
C
C   *DUMP
C
  300    ICMD=6                                                                          !FLG DUMP COMMAND.
         IECCND=0                                                             !ASSUME DIR WITH DUMP.
  307    DMPTTY=.FALSE.
         WRITE(5,11052)
         READ(5,1023)ANS
         IF(ANS.EQ.YES)DMPTTY=.TRUE.
         WRITE(5,1052)                                                        !ASK IF DIR WANTED.
         READ(5,1023,END=920,ERR=990)  ANS                                    !GET ANS.(EOF OR ERR?)YES.
         IF(ANS.EQ.YES)  GO TO 315                                            !ANS YES?)YES, GO DUMP NO DIRECT.
         IF(ANS.NE.HNO)  GO TO 307                                            !(ANS NO?)NO, BAD ANS, GO ASK AGAIN.
         IECCND=-1                                                            !ANS=NO, FLG DUMP ONLY.
         GO TO 315
C
C*****************************************************************************
C
C   *DIRECT
C
  310    ICMD=11                                                                 !FLG COMMAND.
         IECCND=1                                                             !FLG DIR ONLY.
  315    UN=HSTAR                                                             !ASSUME WILD NAME.
         UT=HSTAR                                                             !ASSUME WILD PART TYPE.
         WRITE(5,1044)                                                        !ASK PART TYPE.
         READ(5,1023,END=920,ERR=990)  WORD                                   !GET ANS.  (EOF OR ERR?)YES.
         IF(WORD.NE.HBLANK) UT=WORD                                           !(ANY ANS?)YES, OVERRIDE ASSUMPTION.
         WRITE(5,1050)                                                        !ASK PART NAME.
         READ(5,1004,END=920,ERR=990)  FNAME                                  !GET NAME (EOF OR ERR?)YES.
         IF(FNAME.NE.HBLANK)  UN=FNAME                                        !(ANY ANS?)YES, OVERRIDE ASSUMPTION.
  318    CALL DMECPD(IECCND)                                                  !GO DUMP/DIRECT.
         DMPTTY=.FALSE.
         GO TO 10
C
C*****************************************************************************
C
C   *PDUMP
C
320      ICMD=31
         IECCND=31
         WRITE(5,1044)
         READ(5,1023)UT
         IF(UT.EQ.HBLANK)GO TO 10
         CALL LOOKUP(UT,-ICRCNT(UT,1),HPART,NHPART,NPART,IPRNTT,HINPDI,
         2$340)
         WRITE(5,1042)UT
         GO TO 320
C
340      WRITE(5,1050)
         READ(5,1004)FNAME
         CALL USECMD(-IPCRS(IPRNTT))
         GO TO 320
C
C*****************************************************************************
C
C   *TITLE
C
 400     IECCND=1                                                             !DUMMY TO FULFIL ARGUMENT REQUIREMENTS.
```

```
          WRITE(5,1060)
          CALL TTICMD(IECCMD)
C
          GO TO 10
C
C
C
C*****************************************************************************
C
C   *LET
C
  500     ICMD=36
  502     WRITE(5,1062)
          READ(5,1028,END=920,ERR=990) WORD
          IF(WORD.EQ.HON)GO TO 506
          IF(WORD.NE.HOFF)GO TO 502
  504     LLP1=.FALSE.
          GO TO 508
  506     LLP1=.TRUE.
  508     GO TO 10
C
C*****************************************************************************
C
C   *HELP
C
   550    CALL HLPCMD
C
          GO TO 10
C
C*****************************************************************************
C
C   *TRANS
C
  600     CALL TRACMD
C
          GO TO 10
C
C*****************************************************************************
C
C   *OUTPUT
C
  700     CALL OUTCMD
C
          GO TO 10
C
C************************************************************************************
C
C *SPLIT
C
  800     CONTINUE
          ICMD=41
          FIOE=.TRUE.
          GO TO 10
C
C************************************************************************************
C
C
C*****************************************************************************
C
C   ERROR CONTROL SECTION
C
```

```
C               BUT NO RECOVERY METHOD YET IMPLEMENTED OR IMPOSSIBLE
C               ERROR.
C
  900   WRITE(5,901)  HCCMND(ICMD),ICMD,IPRMPT,IECOND                    !SOME HELP TO JCT.
        WRITE(6,901)  HCCMND(ICMD),ICMD,IPRMPT,IECOND                    !SOME HELP TO LPT.
        CALL TRACE                                                       !TO JCT.
  901   FORMAT(//' ? INPDIA-IMPOSSIBLE ERROR HAS OCCURED'///
        1,' PLEASE CONTACT SID SHAPIRO/KHL TEL. 617-494-2272.'//
        2,'**SAVE ALL OUTPUT**'5XA5,5X,3I3' /HC,IC,IP,IE/'/'1')
C
        IF(ITTY) GO TO 905                                               !(JCT IS TTY?)YES.
        CALL EXIT                                                        !NO, BEYOND REPAIR, BYE.
        STOP '? INPDIA-IMPOSSIBLE STOP POINT IN INPDIA'                  !UGH, GOD HELP US.
C
  905   UT=HBLANK                                                        !FLG ZERO ALL.
        CALL ZERCMD(IECCMD)                                              !GO ZERO.
        CALL TTYCLR                                                      !CLR TTY INP BUF.
        RETURN                                                           !BYE.
C
C  HERE ONLY IN PTY SUBPROC- END OF BATCH CONTROL FILE
C
  920   WRITE(5,921)                                                     !TO JCT.
        WRITE(6,921)                                                     !TO LPT.
  921   FORMAT(//' ? INPDIA-UNEXPECTED END OF BATCH CONTROL FILE.')
C
        ENDET=.TRUE.                                                     !FLG EOF.
        RETURN                                                           !BYE.
C
C  HERE ON TTY FORMAT I/O ERROR
C
  990   WRITE (5,1034)                                                   !?FORMAT ERROR.
        CALL TTYCLR                                                      !CLEAR TTY INP BUFFER.
        IF(ICMD.EQ.0) GO TO 10                                     !ANY COMMAND BEING PROCESSED?)NO, COMMAND READ ERR, GO PROMPT.
  995   GO TO (104,201,231,900,241,900,900,211,221,900
        1,      900,900,900, 10, 10,900,900,900,900,900
        2,      900,900,900,900,900,900,261,900,281,900
        3,      291,291,900,900,900,900,900,900,900,900,900),ICMD        !GET BACK TO WHERE WE WERE.
        GO TO 900                                                        !IMPOSSIBLE, BUT --- ZAP WERE LOST.
C
C***********************************************************************
C
C    ***FORMAT STATEMENTS***
C
 1001   FORMAT(/' ENTER NUMBER OF ACCESSORIES TO BE USED: '$)
 1002   FORMAT(20I)
 1003   FORMAT(/' ENTER PART NAME FOR ACCESSORY #'I3': '$)
 1004   FORMAT(A10)
 1006   FORMAT(/' ENTER PART NAME OF 'A5' CONVERTER TO BE USED: '$)
 1008   FORMAT(/' ENTER PART NAME OF DRIVING SCHEDULE TO BE USED: '$)
 1010   FORMAT(/' ENTER ENGINE NUMBER(1 OR 2), PART NAME, AND GEAR '
        1,'ASSIGNMENTS'/' : '$)
 1011   FORMAT(I,A10,20I)
 1012   FORMAT(/' ENTER CORRECT ENGINE NAME: '$)
 1013   FORMAT(/' ENTER NUMBER OF GEARS TO BE USED: '$)
 1014   FORMAT(/' ENTER PART NAME FOR GEAR #'I3': '$)
 1016   FORMAT(/' ENTER PART NAME OF ROUTE TO BE USED: '$)
 1018   FORMAT(/' ENTER PART NAME OF SHIFT LOGIC TO BE USED: '$)
 1020   FORMAT(/' ENTER PART NAME OF VEHICLE TO BE USED: '$)
 1021   FORMAT(1X,10A10)
 1022   FORMAT(/' SIMULATE ? (ANS Y/N/E): '$)
```

```
1023    FORMAT(A5,1X10A10)
1024    FORMAT(/' ENTER GEAR NUMBERS TC BE LOCKED UP: '$)
1025    FORMAT(/' ? INPCIA-'I3' IS ILIEGAL GEAR NUMBER.')
1026    FORMAT(/' ENTER GEAR NUMBERS TC BE UNLOCKED: '$)
1027    FORMAT(/' *MODIFY(NEW VALUE,ITEM): '$)
1028    FORMAT(A5,10F)
1029    FORMAT(/' ? INPCIA- 'A5' UNKNOWN MODIFY COMMAND.')
1030    FORMAT(/' ? INPCIA- NO PART LCADEC FOR 'A5' MODIFY.')
1031    FORMAT(/' ? INPCIA- INPEAT BEFORE PLEASE RELOAD PARTS DATA.')
1032    FORMAT(/' *LIMIT PRINT: '$)
1033    FORMAT(/' ? INPCIA- 'A5' UNKNCWN LIMIT PRINT COMMAND.')
1034    FORMAT(/' ? INPCIA- FCRMAT I/C ERROR.')
1035    FORMAT(/' ? INPCIA- PARTS MISSING.')
1036    FORMAT(/' DEBUG: '$)
1037    FORMAT(/' ? INPCIA- "'A5'" UNKNOWN DEBUG COMMAND.')
1038    FORMAT(/' ? INPCIA- DEBUG START VALUE LARGER THAN STOP VALUE.')
1039    FORMAT(/' ENTER GEAR # , NAME: '$)
1040    FCRMAT(F,A5)
1041    FORMAT(/' *'$)
1042    FORMAT(/' ?'A5'?')
1043    FORMAT(/' ?INPCIA- "'A5'" UNIMPLEMENTED VERSIN COMMAND FOR'
        1,' CIALOGUE MOCE.')
1044    FORMAT(/' PART TYPE: '$)
1045    FORMAT(/' TTY OUTPUT(CN/OFF): '$)
1046    FORMAT(/' ?INPCIA-"'A5'" ILLEGAL COMMAND WHEN IN ¬P(PAUSE)'
        1,' MCDE.')
1047    FORMAT(/' %INPCIA- IN ¬P(PAUSE) MODE ISEG:'I4' **BE CAREFUL**')
1048    FORMAT(/' ENTER NAME OF PART TC BE CROPPED: '$)
1049    FCRMAT(/' ENTER PART NAME OF TIRE TO BE USED: '$)
1050    FORMAT(/' ENTER PART NAME: '$)
1051    FORMAT(/' ? INPCIA- 'F' IS ILLEGAL VALUE FOR 'A5' MODIFY.')
11052   FORMAT(/' DUMP TO TTY?(ANS Y/N) '$)
1052    FORMAT(/' CO YOU WANT A DIRECTORY OF PARTS DUMPED?(ANS Y/N): '$)
1060    FORMAT(/' ENTER RUN TITLE: '$)
1062    FORMAT(/' LPT OUTPUT(CN/OFF): '$)
1080    FORMAT(/' ENTER PART NAME OF TRANSMISSION (OR <CR> FOR GEARS): '
        2,      $)
1082    FORMAT(/' ENTER GEAR NUMBER: '$)
1084    FORMAT(/' ENTER AXLE NAME: '$)
        END
```

```
      SUBROUTINE ITERAT(TITER)
C
C       ENTRY POINTS:    ITERAT
C
C       SUBROUTINES CALLED:     GETACL, GOBACK
C
C       CALLED BY:      SIMCTR
C
C*****************************************************************
C
C       EDIT HISTORY
C
C       [601]/SS-1-27-78        IN DECMAL PORTION, IF NOT STRADDLING TARGET,
C                    BUT STILL GOING IN WRONG DIRECTION, REVERSE.
C       [602]/SS-1-31-78        ONLY DO LIN.INTERP. IF STRADDLING.
C       [604]/SS-3-16-78        OUTPUT DEBUG INFO DEPENDING ON DEBUG FLAG
C       [611]/SS-6-22-78        SET FLAG FOR GOBACK IF IN ITERAT
C       [622]/SS-6-12-79        IF WE HAVE BRACKETED THE SOLUTION,
C                               INDICATED BY A SWITCHING OF SIGNS OF TORD,
C                               THEN WE ASSUME WE CAN FIND THE SOLUTION.
C                               BASED ON THAT, WE LOWER THE DACC TOLERANCE
C                               AND IGNORE TORD POSSIBLY NOT MOVING.
C
C*****************************************************************
C
C
      COMMON /CDEBUG/  IDEBUG,DBEGIN,DSTOP,ISEG1,ISEG2,ISEG,CUMT,CUMD
      COMMON /ENGMAP/ RPMAX(2),RPMIN(2),NRPM(2),RPME,TORQE,FRATE,VAC,
     1THR,MAPOR,
     2IFFRE,MTOR(2,20),EMAP(20,20,8),ERPM(2,20),EMMIN(2),SPIDLE(2)
      COMMON /CNTRL/ IC,TOLC,VOID,T,V,ACCEL,D,DT,LITER
      COMMON /TMAP/ TCRC,LWOT,TWOT,THIN
      COMMON /DSCRED/ DNAME,DCOM(16),TO,VO,DO,AO,NGO,NSEG,ASEG(50),
     1VSEG(50),PWOT(50),ATHOLD(50),NGSEG(50),THRATE(50),TSEG(50),
     2DSEG(50),PCSEG(50),POSTSE(50),VELSEG(50),ITYSEG(50)
     3,LSTSEC,KDSEG,NSEC
      COMMON /OUTP/FREEEL,FAERO,FACCEL,TRR,TORQF,DRPMW,DRPME,FROLL,
     1   FGRADE
      LOGICAL DECMAL,LITER,LFLIE
      LITER=.TRUE.                                                 ![611]

      PRINT6=.FALSE.
      IF(IDEBUG.GE.5.AND.CUMT.GE.DBEGIN.AND.
     2        (CUMT.LE.DSTOP.OR.ISEG2.EQ.0))PRINT6=.TRUE.
      IF(PRINT6)WRITE(6,9001) TITER
9001  FORMAT(' $ITERAT - TITER ->',F)
C
C    ACCELERATE TO DESIRED VELOCITY
C
      DACC=0.                                                      !INIT DELTA ACCEL TO 0.
      NITER=0                                                      !INIT ITERATION COUNTER.
      DECMAL=.FALSE.                                               !INIT LINEAR EXTRAPOLATION METHOD FLAG.
      LFLIP=.FALSE.                                                ![622]
      DACTOL=1.E-3
      TORDO=2.E+10                                                 !SET OLD DELAT TORQE TO SOME HIGH CONSTANT.
      TORD=1.E10                                                   !SET DELTA TORQE TO SOME HIGH CONSTANT.
      TOL=(TWOT-THIN)*.5E-2                                        !COMPUTE TOLERANCE AS A FUNCTION OF THROW.
20    ACCO=ACCEL                                                   !SAVE ACCELERATION.
      CALL GETACL(TITER)                                           !GET NEW ACCELERATION.
40    CALL GOBACK
      NITER=NITER+1                                                !INC ITERATICN COUNTER.
      TORDO=TORD                                !SAVE OLD DELTA TORQE.
      TORD=TITER-TORQE                          !GET A DELTA TORQE.
      IF(ERIN16)WRITE(6,9000)TORD,TOL,ACCEL,DACC
9000  FORMAT(' $ ITERAT - TORD,TOL,ACCEL,DACC',4G)
      IF(ABS(TORD).LT.TOL)GO TO 500             !(WITHIN TOLERANCE)YES, GO TO 500.
      IF(DECMAL)GO TO 100                       !(DECIMAL METHOD?)YES.
      IF(ABS(TORDO-TORD).LT.1.E-3)GO TO 60      !(ARE WE MOVING?)NO, SWITCH TO DECIMAL METHOD.
      IF(TORDO.GT.1.E9)GO TO 20                 !(FIRST TIME THRU?)YES, GO BACK AND DO AGAIN.
      IF(NITER.GT.20)GO TO 60              !(HAVE WE CONVERGED IN 20 TRIES?)NO, SWITCH TO DECIMAL METHOD.
      IF(TORD*TORDO.GT.0.)GO TO 50              ![602]
      ANEW=ACCEL-TORDO*((ACCEL-ACCOO)/(TORD-TORDO))  !GET NEW ACCELERATION, BY LINEAR EXTRAPOLATION METHOD.
45    ACCOO=ACCEL                               !SAVE OLD ACCEL.
      ACCEL=ANEW                                !GET NEW ACCEL.
      GO TO 40
C
50    DACC=ACCEL-ACCOO                          ![602]
      IF(ABS(TORD).GT.ABS(TORDO))DACC=-DACC     !IF MOVING AWAY, SWITCH.SIGN OF DELTA ACC.[CSID]REMOVED FOR AUTOMATIC
      ANEW=ACCEL+DACC                           ![602]
      GO TO 45                                  ![602]
C
60    DECMAL=.TRUE.                             !SET DECIMAL FLAG.
      IF(ERIN16)WRITE(6,9002)
9002  FORMAT(' $ITERAT - DECMAL ---')
      DACC=-(ACCOO-ACCEL)/2.                    !GET DELTA ACCEL.
      IF(DACC.LT.1.E-3)DACC=-ACCEL/2.
80    ACCOO=ACCEL                               !SAVE OLD ACCEL.
      ACCEL=ACCEL+DACC                          !GET NEW ACCEL.
      GO TO 40
C
100   IF((TORD*TORDO).GT.0.)GO TO 120           !(ARE WE APPROACHING FROM SAME SIDE?)YES.
      LFLIP=.TRUE.                              ![622]
      DACTOL=1.E-5                              ![622]
      DACC=-DACC*0.5                            !NO, CUT IN HALF AND SWITCH SIGNS.
      IF(ABS(DACC).LT.DACTOL)GO TO 500          !(IS DELTA ACCEL WITHIN TOLERANCE?)YES.
      GO TO 140
120   IF(TORD.NE.TORDO)GO TO 130           ![622]IF TORD NOT CHANGING, KEEP GOING WITH SAME DACC UNTIL IT DOES CHANGE
      IF(DACC.LT.0.)DACC=-DACC    ![622]MAKE SURE ACCEL WILL INCREASE RATHER THEN DECREASE (CUZ IT WILL GO TO ZERO REAL SOON
      GO TO 80                                  ![622]TRY IT
130   IF(ABS(TORD).GT.ABS(TORDO))GO TO 160      ![601] MOVING AWAY?)YES.
      IF((ABS(TORD-TORDO).LT.1.E-5) .AND. .NOT.LFLIP)GO TO 160     ![622]
      DACC=DACC*0.5                             !NO, CUT IN HALF.
140   IF(ABS(DACC).LT.DACTOL)GO TO 500          !(IS DELTA ACCEL WITHIN TOLERANCE?)YES.
      GO TO 80
C
160   IF(ABS(DACC).GE.ABS(ACCEL))DACC=DACC/2.   ![SID]IF MAG OF DELTA IS GRATER THAN MAG OF ACCEL, BAD MOVE
      DACC=-DACC                                ![601] REVERSE
      IF(ABS(DACC).LT.DACTOL)GO TO 500          !(IS DELTA ACCEL WITHIN TOLERANCE?)YES.
      ACCEL=ACCEL+2.*DACC                       ![601] GET BACK TO LAST STEP
      CALL GOBACK                               ![601]
      TORD=TITER-TORQUE                         ![601]
      GO TO 80                                  ![601]
C
500   LITER=.FALSE.                             ![611]
      CALL GOBACK                               ![611]
      RETURN
      END
      SUBROUTINE KPTIME(ITASK)
C
C     ENTRY POINTS:   KPTIME
C
C     CALLED BY:      SIMCTR, SIMINT, SIMSTS
C
C*************************************************************************
C
C
      COMMON /SSTIME/ CPUS,CPUT
C
      CALL SECNDS(RUNTIM)               !GET JOB RUN TIME(SECS)
C
      GO TO (10,20,30),ITASK            !(BEGIN TIME OF-SIMULATION/DRS SEG/CALC ELAPSED TIME)
C
   10 BEGSIM=RUNTIM                     !SAVE BEGIN SIM TIME.
      RETURN                            !BYE
C
   20 BEGSEG=RUNTIM                     !SAVE BEGIN DRS SEG TIME
      RETURN                            !BYE
C
   30 CPUS=RUNTIM-BEGSEG                !CALC CURRENT DRS SEG CPU TIME.
      CPUT=RUNTIM-BEGSIM                !CALC CURRENT CPU TIME.

      RETURN                            !BYE
      END
      SUBROUTINE LOOKUP(WORD,NCHAR,TABLE,NMIN,NLEN,NWRD,HCALL,*)
C
C     ENTRY POINTS:   LOOKUP
C
C     SUBROUTINES CALLED:     ASCIZ,  RESETM, TRACEL
C
C     CALLED BY:      DSKDIR, HLPCMD, INPRAT, INPDIA
C
C     *******************************************
C
C     ARGUMENT DEFINITIONS
C
C     WORD    - WORD YOU WISH TO LOOKUP
C     NCHAR   - NUMBER OF CHARACTERS IN WORD
C               IF NCHAR IS NEGATIVE, LOOKUP WILL OUTPUT ANY ERROR MESSAGES
C     TABLE   - VECTOR OF WORDS
C     NMIN    - VECTOR WITH MINIMUM CHARS TO UNIQUELY DEFINE
C               WORD IN CORRESPONDING TABLE.
C     NLEN    - NUMBER OF ENTRIES IN TABLE
C     NWRD    - RETURNED VALUE.  POSITION OF WORD IN TABLE
C     HCALL   - HNAME OF CALLER (TO BE USED IN ERROR MESSAGES)
C     $STATEMENT TO RETURN ON NORMAL RETURN.  LOOKUP WILL
C             DROP THROUGH CALL ON ERROR RETURN.
C
C     **************************************
C
      LOGICAL LERMSG
C
      DOUBLE PRECISION HCALL,HUNKNO,HAMBIG,DWORD,DBLANK,HTRACE
C
      DIMENSION TABLE(NLEN),NMIN(NLEN)
C
      COMMON /MASKS/ WMASK(5)
C
      DATA HUNKNO/'UNKNOWN'/,HAMBIG/'AMBIGUOUS'/,DBLANK/' '/,JCT/5/
C
      IF(NCHAR.GE.0) GO TO 10                                 !(DO WE PRINT OUR OWN ERROR MESSAGE?) NO.
      NC=-NCHAR                                               !YES, MAKE # OF CHAR'S (+).
      LERMSG=.TRUE.                                           !REMEMBER TO PRINT ERR MSG IF ANY.
      GO TO 15
   10 NC=NCHAR
      LERMSG=.FALSE.                                          !IF ERR JUST FLG CALLER AND RETURN.
   15 N=NC                                                    !SET MASK PTR.
      IF(N.GT.5) N=5                                          !(# OF CHAR> # OF MASK?) YES, RESET PTR.
C
      DO 20 NWRD=1,NLEN                                       !SEARCH TABLE FOR WORD.
      IF((TABLE(NWRD).AND.WMASK(N)).EQ.(WORD.AND.WMASK(N))) GO TO 30   !(GOT IT?) YES.
   20 CONTINUE                                                !NO, NEXT.
C
      NWRD=0                                                  !SET ERR, WORD NOT IN TABLE.
      DWORD=HUNKNO
   25 IF(.NOT.LERMSG) RETURN                                  !(PRINT ERR MSG?) NO, GET OUT.
      IF(HCALL.NE.DBLANK) GO TO 17
      CALL TRACEL(0,NARG,DUMMY,IERR,HTRACE)                   !GET CALLER'S NAME. (NON-VERSIM USERS, KILL THIS LINE.)
      IF(IERR.GE.0) GO TO 19                                  !(FATAL ERR?) NO.
      WRITE(JCT,1005)                                         !YES, REPORT IT.
      CALL RESETM                                             !TAKE THE EASY WAY OUT!(NON-VERSIM USERS KILL THIS LINE)
C
   17 HTRACE=HCALL
   19 WRITE(JCT,1000) HTRACE
      CALL ASCIZ(WORD,1,N)                                    !NULL TRAILING BLANKS.
```

```
          WRITE(JCT,1001) DWORD                                       !REPORT IT.
          RETURN                                                      !BYE.
C
   30     IF(NC.GE.KMIN(NWRD).OR.NCHAR.EQ.5) RETURN 1     !(WORD HAS#OF CHAR'S TO UNIQUILY DEFINE IT IN TABLE?) YES, BYE.
          DWORD=BAMBIG                                                !NO,FLG WORD AMBIGUOUS.
          NWRD=-NWRD
          GO TO 25
C
 1000     FORMAT(/' ? 'A6'- "'$)
 1001     FORMAT('+" 'A9' WORD.')
 1005     FORMAT(/' ? LOOKUP - TRACE ERROR.')
          END
```

```
      SUBROUTINE MODSL    ( MODE,PCT )
C
C       ENTRY POINTS:    MODSL
C
C       SUBROUTINES CALLED:      PRNTPD
C
C       CALLED BY:       INPEAT
C
C*****************************************************************
C
C
      LOGICAL      ISCALE,LINERD,LTRRZ
C
      DOUBLE PRECISION   SNAME,GNAME
C
      COMMON /PRNLIM/  LIMPRN,BILIN,SECLIN,ENDLIN,ALINN,LSCALE
      COMMON /CONST/   FRC1,FRC2,FAC,CD,AREA,VWIND,WGT,FGC,WRAD,RAR,
     1        GRAT(20),RDMG,NGEAR,AIN,AIP,AI2,ERAT(20),ERAR(2),
     2AIE,AIA,AI1,EPER,CDC,PHI,PSI,AIGIN(20),AIGOUT(20),WLSG,LTRRZ
     3,NGELSS(20),GRPM(20,20),GRTORC(20,20),GNAME(20),GCOM(16)
      COMMON /SHPTL/   SNAME,SCOM(16),GCVPSI(4),OUTRPM(4),NGPT,IGP(38),
     1                  IGT(38),SHPTIN(38),LGPREB(38),LPSHP(38),
     2                  EFFST(38),ABRUPT(38),SHFTPT(10,38),
     3                  SHFTRP(10,38),LVAC,LENG,GDAT,NSPTS(38),LDETNT,
     4                  DETPT(38),DETRPM(38),PARAB,LDETP,LDETV,
     5                  GOVLIN,LDTH
      COMMON /V2MISC/  LOCKUP(20),DATE(2),NPARTS(11),DEFDT,NORUN,NENG,
     1                  IUNIT,IEART,IPMODE,IBMODE,ISMODE,MSGBAR(20,2)
C
C
C     MODE   = 1 FOR UPSHIFT MODS , 2 FOR DOWNSHIFT MODS
C     PCT    = PERCENT CHANGE (+ OR -) OF SHIFT LEVEL (VACU OR THROT)
C
C
C*****************************************************************
C
      NUMESL = ( NENG - 1 ) * 2
      DO 200  I = 1,NUMESL
C
C     SELECT UP/DOWN SHIFT LINE TO MODIFY
C
      GO TO ( 110,120 ) , MODE
  110 IF ( IGP(I).GE.IGT(I) )  GO TO 200
      GO TO 140
  120 IF ( IGP(I).LE.IGT(I) )  GO TO 200
C
C     CHANGE SHIFT LEVEL , NO CHANGE TO DETENT OVERRIDE LEVEL
C
  140 NPOINT = NSPTS(I)
      DO 160  N = 1,NPOINT
      SHFTPT(N,I) = SHFTPT(N,I) + (PCT/100.)*SHFTPT(N,I)
C
C     CHECK FOR SHIFT LINE MODIFIED BELOW ZERO
C
       IF ( SHFTPT(N,I).LT.0. )  SHFTPT(N,I) = 0.
  160 CONTINUE
C
  200 CONTINUE
C
C     PRINT OUT MODIFIED SHIFT LOGIC
C
```

```
        IF ( LIPPEN )  GO TO 900
        GO TO ( 310,320 ) , MODE
  310   WRITE (IUNIT,1310)  PCT
        GO TO 340
  320   WRITE (IUNIT,1320)  PCT
  340   IPAFT  = 7
        CALL ERNTPD
C
  900   RETURN
C
C
C**********************************************************************
C
C     FORMAT STATEMENTS
C
 1310   FORMAT (1H1/////10X,34HTHE UPSHIFT LINES OF THE FOLLOWING,
        1          29H SHIFT LOGIC WERE MODIFIED BY,F6.1,8H PERCENT///)
 1320   FORMAT (1H1/////10X,36HTHE DOWNSHIFT LINES OF THE FOLLOWING,
        1          29H SHIFT LOGIC WERE MODIFIED BY,F6.1,8H PERCENT///)
C
        END
```

```
      SUBROUTINE NOPART(IUNIT,PNAME,IPRNT,IFLG)
C
C     ENTRY POINTS:   NCPART
C
C     CALLED BY:      INPBAT, INPDIA
C
C*********************************************************************
C
C
      DOUBLE PRECISION ENAME
C
C*****************************************************************************
C
      GO TO (10,20,30,40,50,60,70,80,90,100,110),IPRNT                          !WHAT PART TYPE NOT FOUND.
C
   10 WRITE(IUNIT,1012) PNAME                                         !ENG NOT FOUND.
      GO TO 900
C
   20 WRITE(IUNIT,1007) IFLG,ENAME                                    !TORQUE CONVERTER NOT FOUND.
      GO TO 900
C
   30 WRITE(IUNIT,1021) PNAME                                         !VEH NOT FOUND.
      GO TO 900
C
   40 WRITE(IUNIT,1015) PNAME                                         !GEAR NOT FOUND.
      GO TO 900
C
   50 WRITE(IUNIT,1005) PNAME                                         !ACCESSORY NOT FOUND.
      GO TO 900
C
   60 WRITE(IUNIT,1009) PNAME                                         !DRI SCHED NOT FOUND.
      GO TO 900
C
   70 WRITE(IUNIT,1019) PNAME                                         !SHIFT LOGIC NOT FOUND.
      GO TO 900
C
   80 WRITE(IUNIT,1017) PNAME                                         !ROUTE NOT FOUND.
      GO TO 900
C
   90 WRITE(IUNIT,1001) PNAME                                         !TIRE NOT FOUND.
      GO TO 900
C
  100 WRITE(IUNIT,1022)ENAME                                          !TRANSMISSION NOT FOUND.
      GO TO 900
C
  110 WRITE(IUNIT,1024)ENAME                                          !AXLE NOT FOUND.
      GO TO 900
C
  900 RETURN
C
C****************************************************************************
C
C  FORMAT STATEMENTS
C
 1005 FORMAT(/' ? ACESSORY 'A10' NOT IN PARTS DATA FILE.')
 1007 FORMAT(/' ? 'A5' CONVERTER 'A10' NOT IN PARTS DATA FILE.')
 1009 FORMAT(/' ? DRIVING SCHEDULE 'A10' NOT IN PARTS DATA FILE.')
 1012 FORMAT(/' ? ENGINE 'A10' NOT IN PARTS DATA FILE.')
 1015 FORMAT(/' ? GEAR 'A10' NOT IN PARTS DATA FILE.')
 1017 FORMAT(/' ? ROUTE 'A10' NOT IN PARTS DATA FILE.')
 1019 FORMAT(/' ? SHIFT LOGIC 'A10' NOT IN PARTS DATA FILE.')
 1021    FORMAT(/' ? VEHICLE 'A10' NOT ON PARTS DATA FILE.')
 1001    FORMAT(/' ? TIRE 'A10' NOT IN PARTS DATA FILE.')
 1022    FORMAT(/' ? TRANSMISSION ',A10,' NCT FOUND IN PARTS DATA BASE')
 1024    FORMAT(/' ? AXLE ',A10,' NOT FOUND IN PARTS DATA BASE')
C
         END
```

```
      SUBROUTINE PRNOUT
C
C       ENTRY POINTS:   PRNOUT
C
C       SUBROUTINES CALLED:     PRNTPD
C
C       CALLED BY:      DSK,    DSKDIR, INPEAT, REMAP,  SCALEM
C
C*******************************************************************
C
C
        INCLUDE 'COMMS/NCLIST'
C
        DIMENSION  ARR(20),SOUZ(20),TOUZ(20)
C
C
C
C**********************************************************************
C
C
        IF (LIMERR) RETURN                                          !(LIMIT PRINT OUT?)YES, A REAL QUICKY.
C
        IPART  = IPRNT                                              !SAVE.
        IF(IPRNT.GT.200) IPART=IPRNT-200                            !(FLG TO CALL FOR RELOADING OF 1ST SECT?)YES, SUB FLG.
C
        IF(IPRNT.GT.1) GO TO 2000
C
C      ENGINE DATA TO BE PRINTED SO DO ANY NEEDED UNITS CONVERSION FIRST.
C
        KENG=(IENG-1)*4
        NRDUM=NEPP(IENG)
C
C     CHECK UNITS FOR ENGINE DATA TO BE PRINTED
C
        IF(PRPM.OR.PPS) GO TO 2
        IF(LRPM) PRPM=.TRUE.
        IF(LPS) PPS=.TRUE.
    2   IF(PTOR.OR.PBMEP.OR.PHP) GO TO 3
        IF(LTOR) PTOR=.TRUE.
        IF(LBMEP) PBMEP=.TRUE.
        IF(LHP) PHP=.TRUE.
        IF(LLBHR) PLBHR=.TRUE.
    3   IF(PLBHR.OR.PBSFC.OR.PGALHR) GO TO 1
        IF(LLBHR) PLBHR=.TRUE.
        IF(LBSFC) PBSFC=.TRUE.
        IF(LGALHR) PGALHR=.TRUE.
C
C     CONVERT UNITS IF NECESSARY
 1      IF(PLBHR) GO TO 7
        IF(PBSFC) GO TO 5
C
C     CONVERT LB/HR TO GAL/HR
C
        DUM=FSPGR*62.426134/7.480520
        NRDUM=NAPP(IENG)
        DO 4 I=1,NRDUM
        DO 4 J=1,20
    4   EMAP(I,J,2+KENG)=DUM*EMAP(I,J,2+KENG)
        GO TO 7
C
C     CONVERT LB/HR TO BSFC
```

```
C
    5   DO 6 I=1,NRDUM
        DUM=5252./ERPP(IENG,I)
        DO 6 J=1,20
        IF(ABS(EMAP(I,J,1+KENG)).LT.1.E-20) GO TO 666
        EMAP(I,J,2+KENG)=DUM*EMAP(I,J,2+KENG)/EMAP(I,J,1+KENG)
        GO TO 6
  666   EMAP(I,J,2+KENG)=EMAP(I,J,2+KENG)*1.E20
    6   CONTINUE
    7   IF(ETOR) GO TO 1002
        IF(EDMEP) GO TO 9
C
C     CONVERT LB-FT TO HP
C
        DO 8 I=1,NRDUM
        DUM=ERPP(IENG,I)/5252.
        DO 8 J=1,20
    8   EMAP(I,J,1+KENG)=DUM*EMAP(I,J,1+KENG)
        GO TO 1002
C
C     CONVERT LB-FT TO BMEP
C
    9   AK=150.8
        IF(NCYCLE.EQ.2) AK=75.4
        DUM=AK/DISP
        DO 1001 I=1,NRDUM
        DO 1001 J=1,20
 1001   EMAP(I,J,1+KENG)=DUM*EMAP(I,J,1+KENG)
 1002   IF(ERPM) GO TO 1004
C
C     CONVERT RPM TO PISTON SPEED
C
        DUM=STROKE/6.
        DO 1003 I=1,NRDUM
 1003   ERPP(IENG,I)=DUM*ERPM(IENG,I)
        ENMIN(IENG)=DUM*ENMIN(IENG)
        RPMAX(IENG)=DUM*RPMAX(IENG)
        SPIDLE(IENG)=DUM*SPIDLE(IENG)
 1004   CONTINUE
C
C     PRINT ENGINE DESCRIPTION
C
C....................................
        CALL PRNTED
C
C     CONVERT UNITS BACK
C
        IF(ERPM) GO TO 15
C
C     PISTON SPEED TO RPM
C
        DUM=6./STROKE
        DO 12 I=1,NRDUM
   12   ERPP(IENG,I)=DUM*ERPM(IENG,I)
        ENMIN(IENG)=DUM*ENMIN(IENG)
        RPMAX(IENG)=DUM*RPMAX(IENG)
        SPIDLE(IENG)=DUM*SPIDLE(IENG)
   15   IF(ETOR) GO TO 20
        IF(EBMEP) GO TO 17
C
C     HP TO LB-FT
```

```
C
         DO 16 I=1,NRDUM
         DUM=5252./ERPP(IENG,I)
         DO 16 J=1,20
   16    EMAP(I,J,1+KENG)=DUM*EMAP(I,J,1+KENG)
         GO TO 20
C
C     BMEP TO LB-FT
C
   17    AK=150.8
         IF(NCYCLE.EQ.2) AK=75.4
         DUM=DISP/AK
         DO 18 I=1,NRDUM
         DO 18 J=1,20
   18    EMAP(I,J,1+KENG)=DUM*EMAP(I,J,1+KENG)
   20    IF(ELBHR) GO TO 25
         IF(EBSFC) GO TO 23
C
C     GAL/HR TO LB/HR
C
         DUM=7.480520/(ESPGR*62.426134)
         DO 22 I=1,NRDUM
         DO 22 J=1,20
   22    EMAP(I,J,2+KENG)=DUM*EMAP(I,J,2+KENG)
         GO TO 25
C
C     BSFC TO LB/HR
C
   23    DO 24 I=1,NRDUM
         DUM=ERPP(IENG,I)/5252.
         DO 24 J=1,20
         IF(ABS(EMAP(I,J,2+KENG)).GT.1.E10) GO TO 244
         EMAP(I,J,2+KENG)=EMAP(I,J,2+KENG)*EMAP(I,J,1+KENG)*DUM
         GO TO 24
  244    EMAP(I,J,2+KENG)=EMAP(I,J,2+KENG)/1.E20
   24    CONTINUE
C
 25      GO TO 9000
C
C
C*****************************************************************
C
 2000    CALL ERRTBD                                            !GO DO ACTUAL PRINTING.
C
C*****************************************************************
C
 9000    RETURN                                                 !DONE, BYE.
C
         END
```

```
      SUBROUTINE PRNTPD
C
C        ENTRY POINTS:    PRNTPD
C
C        SUBROUTINES CALLED:      DSK
C
C        CALLED BY:       PRNCUT
C
C*****************************************************************
C
C...     EDIT HISTORY
C
C        [615]/SS-10-4-78         ADD DIESEL PRINTOUT
C
         INCLUDE 'COMNS/NCLIST'
C
         LOGICAL FIRST,ISAVE,LTRANS
C
         DIMENSION AKK(20),SOUZ(20),TOUZ(20),HPART(9)
C
         DOUBLE PRECISION HNA,BOUT(10)
C
         DATA HPART/3*'     ','GEAR ACCESDRIVISHIFTBOUTETIRE'/
         1,HNA/' **N/A**'/,HAXLE/'AXLE'/
C
C*******************************************************************
C
C
         FIRST=.TRUE.
         ISUNIT=IUNIT
         IF(NXTPG.LE.0) NXTPG=1                                  !(PG CNT SET?)NO, SET IT.
         WRITE (IUNIT,1000)  DATE,NXTPG                          !TOP OF PAGE.
         NXTPG=NXTPG+1                                           !INC PAGE COUNT.
         JPRNT=IPRNT                                             !SAVE.
C
10       GO TO (100,200,300,400,500,600,700,800,900,2000,2100),IPART    !WHAT PART TYPE TO PRINT.
C
C*******************************************************************
C
C      PRINT ENGINE DATA
C
  100    KENG    = ( IENG - 1 ) * 4
         FPGAL   = FSPGB * 8.3452
         IF(.NOT.LDIES)WRITE(IUNIT,1100) ENAME(IENG),ECOM,ICYL,            ![615]
         2FPGAL,EORE,EINER,STROKE,DISP,NRPM(IENG),THRMIN,THRMAX
         IF(LDIES)WRITE(IUNIT,1101) ENAME(IENG),ECOM,ICYL,                 ![615]
         2FPGAL,EORE,EINER,STROKE,DISP,NRPM(IENG),THRMIN,THRMAX
         IF ( PRPM    )   WRITE (IUNIT,1102)   ENMIN(IENG),RPMAX(IENG)
         IF ( PPS     )   WRITE (IUNIT,1104)   ENMIN(IENG),RPMAX(IENG)
         NRCOM   = NREP(IENG)                                    !GET # OF SPEED PTS ON ENG MAP.
         IPAGE   = 2                                             !SET LINE COUNT.
C
         DO 160   I = 1,NRCOM                                    !LOOP THRU ALL SPEED PTS.
         IF ( PRPM    )   WRITE (IUNIT,1110)   ERPM(IENG,I)
         IF ( PPS     )   WRITE (IUNIT,1112)   ERPM(IENG,I)
         IF ( .NOT.LPRNT )   GO TO 114
         N       = 21 - NTOR(IENG,I)
         IB      = N
         IE      = 20
         IF ( NTOR(IENG,I).GT.10 )   IE = IB + 9
         GO TO 120
  114 IB      = 1
      IE      = 10
  120 IF ( ETCR   )  WRITE(IUNIT,1120)  (EMAP(I,J,1+KENG),J=IB,IE)
      IF ( EBMEP  )  WRITE(IUNIT,1121)  (EMAP(I,J,1+KENG),J=IB,IE)
      IF ( PHE    )  WRITE(IUNIT,1122)  (EMAP(I,J,1+KENG),J=IB,IE)
      IF ( PLEHK  )  WRITE(IUNIT,1123)  (EMAP(I,J,2+KENG),J=IB,IE)
      IF( .NCT. PESEC )  GO TC 130
C
C    SPECTAL HANCLING FOF FRINTING OF ESFC UNITS , BSFC NOT APPLICABLE
C    WHEN TORQUE IS ZERO - INSERT  **N/A**  IN FORMAT FOR PRINTOUT
C
      NP=0                                                              !ZERO BSFC DATA PT COUNTER.
      DO 125 J=IB,IE                                                    !LOOP THROUGH CURRENT BSFC TO OUTPUT.
      NP=NP+1                                                           !INC BSFC DATA PT CNTR.
      IF(ABS(EMAP(I,J,1+KENG)).GT..009) GO TO 123                       !(TORQUE ZERO?) NO, BSFC VALID.
      EOUT(NP)=HNA                                                      !YES, BSFC N/A.
      GO TO 125                                                         !NEXT.
  123 ENCCDE(8,11124,EOUT(NP)) EMAP(I,J,2+KENG)                         !LOAD EOUT WITH BSFC VAL FOR OUTPUT.
  125 CONTINUE                                                          !NEXT.
      WRITE(IUNIT,1124) (EOUT(J),J=1,NP)                                !PRI BSFC DATA.
C
  130 IF( PGALHR )  WRITE(IUNIT,1125)  (EMAP(I,J,2+KENG),J=IB,IB)
      WRITE (IUNIT,1130)  (EMAP(I,J,3+KENG),J=IB,IB)
      WRITE (IUNIT,1132)  (EMAP(I,J,4+KENG),J=IB,IE)
      IPAGE  = IPAGE + 1                                                !INC LINE CNT.
      IF ( IPAGE.NE.10 .OR. I.EC.NRIUN )  GO TO 140                     !(PAGE?)NO.
      WRITE (IUNIT,1000) CATE,NXTPG                                     !(YES.
      NXTPG=NXTPG+1                                                     !INC PAGE CNT.
      IF(NTOR(IENG,I).LE.10) WRITE(ICNIT,1002)                          !(MORE THAN 10 LOAD PTS?)NO,OUTPUT A BLANK LINE.
      IPAGE  = 0                                                        !ZERO LINE CNT.
  140 IF ( .NCT.LPRNT          )  GO TO 150
      IF ( NTCR(IENG,1).LE.10 )  GO TO 160                              !(MORE THAN 10 LOAD PTS?)NO.
      WRITE (IUNIT,1002)                                                !YES,OUTPUT BLANK LINE.
      IB      = N + 10
      GO TO 152
  150 IB      = 11
  152 IF ( IE.EC.20           )  GO TO 160
      IE      = 20
      GO TO 120
  160 CONTINUE
      GO TO 999                                                         !BYE.
C
C********************************************************************
C
C    PRINT TORQUE CCNVERTER DATA
C
  200 IF ( COAST       )  WRITE(IUNIT,1200)  CNAME                      !(COAST CONVERTER?)YES.
      IF ( .NCT.CCAST )  WRITE(IUNIT,1202)  CNAME                       !(DRIVE CONVERTER?)YES.
      WRITE (IUNIT,1210) CCOM,CCIAM,AI1,AI2
      IF ( ABS(CONTOF).GT..001 )  WFITE (IUNIT,1212)  CONTOB
      DO  220  I = 1,NTCRP
      SOUZ(I)      = SOUT(I) / SPIN(I)
  220 TOUZ(I)      = TOUT(I) / TIN(I)
      IB      = 1                                                       !SET PTR TO 1ST DATA FLD.
      IE      = NTCRE                                                   !SET PTR TO LAST DATA FLD.
      IF ( NTCRE.GT.10 )  IE = 10                                       !(MCRE THAN 10 FIELDS?)YES,SET PTR TO 10TH NO REST NXT PAGE.
  230 WRITE (IUNIT,1230)  (SOUZ(I),I=IB,IE)
      WRITE (IUNIT,1232)  (TOUZ(I),I=IB,IE)
      WFITE (IUNIT,1234)  (SPIN(I),I=IB,IE)
      IF ( .NCT.COAST  )  WRITE (IONIT,1236)  (AKD(I),I=IB,IE)          !(CRIVE CCNVERTER?)YES.
      IF ( IE.EQ.NTCRE )  GC TO 999                                     !(MORE THAN 10 SPD PTS?)NO.
```

```
        WRITE (IUNIT,1002)                                                !YES, OUTPUT BLANK LINE.
        IL     = 11                                                       !SET PTR TO NXT DATA FLD TO BE PRINTED.
        IE     = NTOFF                                                    !SET PTR TO LAST DATA FLD.
        GO TO 230
C
C*******************************************************************
C
C     PRINT VEHICIF DATA
C
  300   IF(LVNEW) GO TO 320                                               !!(NEW VEH DATA FORMAT?)YES.
        WRITE(IUNIT,1300) VNAME,VCOM,WGT,BAR,AREA,ERAR(1),WBAD,           !NO.
        1                       CD,CEC,AIW,FRC1,AIP,FRC2
        GO TO 330
  320   WRITE(IUNIT,1320) VNAPE,VCCM,WGT,RAB,ABEA,WLSG,CD,CDC,AIP
        1,NRAX,(ERAR(I),I=1,NRAX)
        IF(.NCT.LVEHAX)GO TO 999                                          !IF AXLE NOT TIED TO VEHICLE?)GO ON
        IF(NRAX.EC.2) GC TO 340                                           !(2 AXLES?)YES.
  330   IF(NPAX(1).GT.1) GO TC 335                                        !(ANY AXLE SPIN LOSS DATA?)YES.
  333   WRITE(IUNIT,1410) HAXLE                                           !REPORT NO DATA FOR SINGLE AXLE.
        GO TO 999                                                         !DCNE.
  335   WRITE(IUNIT,1415) HAXLE                                           !SINGLE AXLE SPIN LOSS DATA HEADER.
        GO TO 350                                                         !GO PRINT.
  340   IF(NPAX(1).LE.1.AND.NPAX(2).LE.1) GO TO 333                       !(FOR 2 AXLES ANY SPIN LOSS DATA?)NO.
        WRITE(IUNIT,1340)                                                 !YES, HEADER.
        IF(NPAX(1).LE.1) WRITE(IUNIT,1341)                                !(CATA AXLE 1?)NO, REPORT.
        IF(NPAX(2).LE.1) WRITE(IUNIT,1342)                                !(CATA AXLE 2?)NO, REPORT.
  350   IE=NPAX(1)                                                        !ASSUME AXLE 1 DATA LONGER,SET END DATA PTR.
        IF(NPAX(2).GT.IE) IE=NPAX(2)                                      !(AXLE 2 DATA LONGER?)YES,RESET PTR.
C
        DO 360 I=1,IE                                                     !LOOP THRU ALL AXLE DATA POINTS.
        L1=0                                                              !ASSUME NO DATA AXLE 1 TO PRINT.
        IF(RPAX(1).LE.1.OF.NPAX(1).GT.IE) GO TO 355                       !(CATA TO PRINT?)NO.
        L1=1                                                              !YES,FLG IT.
        WRITE(IUNIT,1420) AXRPM(I,1),AXTORQ(I,1)                                  !PRINT AXLE 1.
  355   IF(NRAX.EC.1 .OF. NPAX(2).LE.1 .OB. NPAX(2).GT.IE) GO TO 360
C
        IF(L1.EC.0) WRITE(IUNIT,1002)                                     !(WAS AXLE 1 DATA PRINTED?)NO,"<CR><LF>">
        WRITE(IUNIT,1355) AXRPM(I,2),AXTORQ(I,2)                                  !PRINT AXLE 2 DATA.
  360   CONTINUE                                                          !NEXT.
        GO TO 999
C
C*******************************************************************
C
C     PRINT GEAR CATA
C
  400   LTRANS=.FALSE.
C
  410   WRITE(IUNIT,1400) GNAME(NGEAR),TCOM,GBAT(NGEAR),AIGIN(NGEAR)
        1,EFAT(NGEAR),AIGCUT(NGEAR)                                       !GEAR DATA.
        IF(NGRLSS(NGEAR).GT.1) GO TO 420                                  !(ANY SPINN LOSS DATA?)YES.
        WRITE(IUNIT,1410) HPART(4)                                        !NO, REPORT IT.
        IF(ITRANS)GO TO 2002
        GO TO 999                                                         !DYE.
  420   WRITE(IUNIT,1415) HPART(4)                                        !SPIN LOSS DATA HEADER.
        WRITE(IUNIT,1420) (GRPM(I,NGEAR),GRTORQ(I,NGEAR)                          !PRINT DATA.
        1,I=1,NGRLSS(NGEAR))
        GO TO 999                                                         !DYE.
C
C*******************************************************************
C
C     PRINT ACCESSORY LCSS CATA
```

```
C
  500   WRITE (IUNIT,1500)  ANAME(NACC),ACON,ATAS(NACC)              !ACCES HEADER.
        WRITE(IUNIT,1520)  (ACCS(I,NACC),ACCT(I,NACC),I=1,NNA(NACC))  !ACCES TORQUE LOSS DATA.
        GO TO 999                                                     !BYE.
C
C*************************************************************************
C
C     PRINT DRIVING SCHEDULE
C
  600   WRITE (IUNIT,1600)  DNAME,DCON,TO,DO,VO,AO,NGO                !DRIVE SCHD HEADE AND INITIAL CONDITION.
        WRITE (IUNIT,1602)                                            !SEGMENT HEADER.
        IPAGE=25                                                      !SET LINE USED CNT.
        NSEGA=NSEG                                                    !GET SEG CNT.
  620   DO 680  I = 1,NSEGA
        ISEGA=NDSEG+I                                                 !CALL SEG#.
        GO TO (621,622,623,624),ITYSEG(I)
  621   WRITE (IUNIT,1621)  ISEGA,ASEG(I)
        GO TO 630
  622   WRITE (IUNIT,1622)  ISEGA,VSEG(I)
        GO TO 630
  623   WRITE (IUNIT,1623)  ISEGA,PWOT(I)
        GO TO 630
  624   WRITE (IUNIT,1624)  ISEGA,ATHCLD(I)
  630   IF ( NGSEG (I).GT.0    )  WRITE (IUNIT,1630)  NGSEG(I)
        IF ( THRATE(I).GT..01 )  WRITE (IUNIT,1631)  THRATE(I)
        IF ( TSEG   (I).GT.-.5 )  WRITE (IUNIT,1632)  TSEG(I),ISEGA
        IF ( DSEG   (I).GT.-.5 )  WRITE (IUNIT,1633)  DSEG(I),ISEGA
        IF ( PCSEG (I).GT.-.5 )  WRITE (IUNIT,1634)  PCSEG(I),ISEGA
        IF (POSTSE(I).GT.-.5 )  WRITE (IUNIT,1635)  POSTSE(I),ISEGA
        IF ( VELSEG(I).GT.-.5 )  WRITE (IUNIT,1636)  VELSEG(I),ISEGA
C
        IPAGE=IPAGE+1                                                 !INC LINE USED CNT.
        IF(IPAGE.LE.59 .OR. (I.EQ.NSEGA.AND.LSTSEC)) GO TO 680        !(LINES LEFT OR END CF LOOP?)YES.
        WRITE(IUNIT,1000) DATE,NXTPG                                  !NO, PAGE.
        NXTPG=NXTPG+1                                                 !INC PAGE #.
        WRITE(IUNIT,1602)                                                     !WRITE SEG HEADER.
        IPAGE=7                                                       !SET LINE USED CNT.
C
  680   CONTINUE                                                      !NEXT.
        IF(LSTSEC) GO TC 690                                          !(LAST SECTION?)YES.
        IPRNT=206                                                     !NO, SET DSK FLG TO LOAD NEXT SECTION.
        CALL DSK                                                      !LOAD NEXT SECTION.
        NSEGA=NSEG-NDSEG                                              !CALC SEG'S TO DO.
        GO TO 620                                                     !GO PRINT.
C
  690   IF(NDSEG.EQ.0 .CR. JPRNT.GT.200) GO TO 999                    !(RELOAD 1ST DRS SECT?) NO,BYE.
        IPRNT=106                                                     !YES, SET DSK FLG /USE/ DRS.
        LSAVE=LIMPRN                                                  !SAVE.
        LIMPRN=.TRUE.                                                 !SET TO NO PRI OUT.
        CALL DSK                                                      !RELOAD 1ST SECT OF DRS.
        LIMPRN=LSAVE                                                  !RESTORE.
        IF(IPRNT.EQ.6) GO TO 999                                      !(DSK ERR?) NO, BYE.
        WRITE(JCT,1640) HPART(6),CNAME                                !YES, *IMPOSS, BUT REPORT.
        IPRNT=-10                                                     ! SET ERR FLG
        CALL TRACE                                                    ! SOME HELP?
        GO TO 999                                                     ! GOOD LUCK! BYE.
C
C*************************************************************************
C
C     PRINT SHIFT LOGIC
C
```

```
 700    WRITE (IUNIT,1700)  SHAPE,SCCH,NUMG                              !HEADER.
        NSL     = ( NUMG - 1 ) * 2                                        !CALC # OF SHIFT LINES.
        IPAGE=2                                                           !SET LINE USED CNT.
        DO 770  I = 1,NSL                                                 !LOOP THRU ALL SHIFT LINES.
        IF ( IGF(I).LT.IGT(I) )  WRITE (IUNIT,1710)  IGF(I),IGT(I)
        IF ( IGF(I).GT.IGT(I) )  WRITE (IUNIT,1712)  IGF(I),IGT(I)
        WRITE (IUNIT,1714)  SHFTIM(I)
        N       = NSPTS(I)
        IF ( LVAC       )  WRITE (IUNIT,1720)  (SHFTPT(J,I),J=1,N)
        IF ( LDTH       )  WRITE (IUNIT,1721)  (SHFTPT(J,I),J=1,N)
CCC     IF ( .NCT.LVAC  )  WRITE (IUNIT,1722)  (SHFTPT(J,I),J=1,N)
        IF(.NCT.LVAC.ANC..NOT.LITE)WRITE(IUNIT,1722)  (SHFTPT(J,I),J=1,N)
        IF ( .NCT.PARAE )  GO TC 730
        WRITE (IUNIT,1724)  (SHFTEP(J,I),J=1,N)
        GO TO 740
 730    IF ( LENG       )  WRITE (IUNIT,1730)  (SHFTRP(J,I),J=1,N)
        IF ( .NCT.LENG  )  WRITE (IUNIT,1732)  (SHFTRP(J,I),J=1,N)
 740    IF ( .NCT.LDETNT)  GO TC 760
        WRITE (IUNIT,1740)
        IPAGE=IPAGE+1                                                     !INC LINE CNT.
        IF ( LDETV    )  WRITE (IUNIT,1720)  DETPT(I)
        IF (.NCT.IDETV)  WRITE (IUNIT,1722)  DETPT(I)
        IF ( .NOT.PARAD )  GO TC 750
        WRITE (IUNIT,1724)  DETEPE(I)
        GO TO 760
 750    IF ( LDETE    )  WRITE (IUNIT,1730)  DETRPM(I)
        IF (.NOT.IDETE)  WRITE (IUNIT,1732)  DETRPM(I)
C
 760    IPAGE=IPAGE+1                                                     !INC LINE CNT.
        IF(IPAGE.LE.10 .OR. I.EC.NSL) GO TO 770                           !NC PAGE OR END OF LOOP?)YES.
        WRITE(IUNIT,1000) DATE,NXTPG                                      !PAGE.
        NXTEG=NXTEG+1                                                     !INC PAGE #.
        IPAGE=0                                                           !ZERO.
C
 770    CONTINUE                                                          !NEXT SHIFT LINE.
        GO TO 999                                                         !BYE.
C
C**********************************************************************
C
C     PRINT ROUTE SPECIFICATION
C
 800    WRITE (IUNIT,1800)  RNAME,RCCE                                    !ROUTE HEADER.
        IPAGE=45                                                          !SET LINE LEFT CNT.
 803    IK=NCRTE+1                                                        !SET PTR TO 1ST SEG IN SECT.
 805    IB=IK                                                             !SET PTR TO 1ST SEG TO PRINT.
        IE=IK+IPAGE-1                                                     !SET PTR TO LAST LINE TO PRINT.
        IF(IE.GT.NRDIST) IE=NRDIST                                        !(ENOUGH DATA TO FILL ALL LINES?)NO,RESET TO LAST SEG.
        WRITE(IUNIT,1902) (I,RDIST(I-NCRTE),RGRADE(I-NDRTE)               !PRINT ROUTE SECTION.
     1,RCCEF(I-NCRTE),RVWIND(I-NDRTE),I=IB,IE)
        IF(IE.EC.NRDIST) GO TO 810                                        !(END OF RTE SECT?)YES.
        IK=IE+1                                                           !POINT TO NEXT RTE SEG.
        IPAGE=IPAGE-(IE-IE+1)                                             !DEC LINE LEFT CNT.
        IF(IPAGE.CT.0) GO TO 805                                          !(ANY LEFT?)YES.
        WRITE(IUNIT,1000) DATE,NXTPG                                      !NO, PAGE.
        NXTEG=NXTPG+1                                                     !INC PAGE #.
        IPAGE=55                                                          !SET LINE LEFT CNT.
        GO TO 805                                                         !GO PRINT.
C
 810    IF(LSTRTE) GO TC 820                                              !(LST RTE SECT?) YES,.
        IPRNT=20P                                                         !NO, SET DSK FLG TO GET NXT SECT
        CALL DSK                                                          !GET NXT SECT
```

```
      GO TO 803                                                   ! GO PRINT
C
 820  IF(NDRTE.EQ.0 .OR. JPRNT.GT.200) GO TO 999                   !(RELOAD 1ST SECT?) NO, BYE
      IPRNT=108                                                   !YES, SET DSK FLG /USE/ RTE
      LSAVE=LIMPRN                                                !SAVE
      LIMPRN=.TRUE.                                               !SET TO NO PRNOUT
      CALL DSK                                                    ! RELOAD 1ST SECT OF RTE
      LIMPRN=LSAVE                                                !RESTORE
      IF(IPRNT.EQ.5) GO TO 999                                    !(ERR?) NO, BYE
      WRITE(JCT,1640) HEART(8),RNAME                              !YES,*IMPOSS BUT REPORT IT
      IPRNT=-10                                                   !SET ERR FLG
      CALL TRACE                                                  ! SOME HELP?
      GO TO 999                                                   ! GOOD LUCK! BYE.
C
C*****************************************************************************
C
C     PRINT TIRE DATA
C
 900  WRITE(IUNIT,1900) TNAME,TCON,HEAD,FRC1,FRC2,TIREFF,AIN
C
      GO TO 999
C
C*****************************************************************************
C
C     PRINT TRANSMISSION DATA
C
 2000 WRITE(IUNIT,2500)TRNAM,TRCOM,(GEARNUM(I),GEARAM(I)
     2,       I=1,NGTR)
C
CCCC  DO 2002 NGEAR=1,NGTR
CCCC  GO TO 410
 2002 CONTINUE
C
      GO TO 999
C
C*****************************************************************************
C
C     PRINT AXLE DATA
C
 2100 WRITE(IUNIT,1330)AXNAME,AXCON,RAB,NRAX,(ERAR(I),I=1,NRAX)
      IF(NRAX.EQ.2) GO TO 2340                                             !(2 AXLES?)YES.
 2330 IF(NPAX(1).GT.1) GO TO 2335                                 !(ANY AXLE SPIN LOSS DATA?)YES.
 2333 WRITE(IUNIT,1410) HAXLE                                     !REPORT NO DATA FOR SINGLE AXLE.
      GO TO 999                                                   !DONE.
 2335 WRITE(IUNIT,1415) HAXLE                                     !SINGLE AXLE SPIN LOSS DATA HEADER.
      GO TO 2350                                                  !GO PRINT.
 2340 IF(NPAX(1).LE.1.AND.NPAX(2).LE.1) GO TO 2333                !(FOR 2 AXLES ANY SPIN LOSS DATA?)NO.
      WRITE(IUNIT,1340)                                                   !YES, HEADER.
      IF(NPAX(1).LE.1) WRITE(IUNIT,1341)                          !(DATA AXLE 1?)NO, REPORT.
      IF(NPAX(2).LE.1) WRITE(IUNIT,1342)                          !(DATA AXLE 2?)NO, REPORT.
 2350 IE=NPAX(1)                                                  !ASSUME AXLE 1 DATA LONGER,SET END DATA PTR.
      IF(NPAX(2).GT.IE) IE=NPAX(2)                                !(AXLE 2 DATA LONGER?)YES,RESET PTR.
C
      DO 2360 I=1,IE                                              !LOOP THRU ALL AXLE DATA POINTS.
      L1=0                                                        !ASSUME NO DATA AXLE 1 TO PRINT.
      IF(NPAX(1).LE.1.OR.NPAX(1).GT.IE) GO TO 2355                !(DATA TO PRINT?)NO.
      L1=1                                                        !YES,FLG IT.
      WRITE(IUNIT,1420) AXRPM(I,1),AXTORQ(I,1)                            !PRINT AXLE 1.
 2355 IF(NRAX.EQ.1 .OR. NPAX(2).LE.1 .OR. NPAX(2).GT.IE) GO TO 2360
C
      IF(L1.EQ.0) WRITE(IUNIT,1002)                               !(WAS AXLE 1 DATA PRINTED?)NO,"<CR><LF>">
 2360   CONTINUE                                                              !NEXT.
C
C
C*********************************************************************
C
  999   IF(.NOT.DMPTTI)GO TO 9999
        IF(.NOT.FIRST)GO TO 9999
        FIRST=.FALSE.
        IUNIT=JCT
        GO TO 10
C
9999    IUNIT=ISUNIT
        RETURN                                                                !* DONE HERE BYE
C
C*********************************************************************
C
C      FORMAT STATEMENTS
C
 1000   FORMAT (1H1,1XA5,110X'PAGE'I5)
 1002   FORMAT (1X)
 1100   FORMAT('+'20X'ENGINE DATA  ( 'A10,2H )/20X,27(1H-)//2X,16A5/
        1        /2X,14HCYLINDERS    =,I5,12X,12HFUEL DENSITY,6X,1H=,
        2        F8.3,2X,6HLB/GAL/2X,4HBORE,9X,1H=,F9.3,8X,
        3        19HROTATING INERTIA  =,F8.3,2X,12HFT-LB-SEC**2/2X,
        4        6HSTROKE,7X,1H=,F9.3/2X,14HDISPLACEMENT =,F7.1,28X,
        5        17HMINIMUM   MAXIMUM//2X,I2,13H SPEED POINTS,16X,
        6        17HTHROTTLE ANGLE  =,F8.2,F10.2,2X,7HDEGREES)
 1101   FORMAT('+'20X'ENGINE DATA  ( 'A10,2H )/20X,27(1H-)//2X,16A5/
        1       /2X,'DIESEL'
        1        /2X,14HCYLINDERS    =,I5,12X,12HFUEL DENSITY,6X,1H=,
        2        F8.3,2X,6HLB/GAL/2X,4HBORE,9X,1H=,F9.3,8X,
        3        19HROTATING INERTIA  =,F8.3,2X,12HFT-LB-SEC**2/2X,
        4        6HSTROKE,7X,1H=,F9.3/2X,14HDISPLACEMENT =,F7.1,28X,
        5        17HMINIMUM   MAXIMUM//2X,I2,13H SPEED POINTS,16X,
        6        17HTHROTTLE ANGLE  =,F8.2,F10.2,2X,7HDEGREES)
 1102   FORMAT (33X,12HENGINE SPEED,4X,1H=,F7.1,F10.1,3X,6HRPM   )
 1104   FORMAT (33X,12HENGINE SPEED,4X,1H=,F7.1,F10.1,3X,6HFT/MIN)
 1110   FORMAT (2X,13HSPEED (RPM) =,F8.2/2X,21(1H-))
 1112   FORMAT (2X,23HPISTON SPEED (FT/MIN) =,F8.2/2X,31(1H-))
 1120   FORMAT ( 5X,17HTORQUE(FT-LB)      , 8X,10F8.2)
 1121   FORMAT ( 5X,17HBMEP(PSI)          , 8X,10F8.2)
 1122   FORMAT ( 5X,17HPOWER(HP)          , 8X,10F8.2)
 1123   FORMAT ( 5X,17HFUEL RATE(LB/HR) , 8X,10F8.2)
11124   FORMAT(F8.2,2X)
 1124   FORMAT(5X'BSFC(LB/HP-HR)'11X10A8)
 1125   FORMAT ( 5X,17HFUEL RATE(GAL/HR), 8X,10F8.2)
 1130   FORMAT ( 5X,17HTHROTTLE(DEGREES), 8X,10F8.2)
 1132   FORMAT ( 5X,22HMANIFOLD VACUUM(IN-HG) ,3X,10F8.2)
 1200   FORMAT (/20X,24HCOAST CONVERTER DATA  ( ,A10,2H )/20X,36(1H-)//)
 1202   FORMAT (/20X,24HDRIVE CONVERTER DATA  ( ,A10,2H )/20X,32(1H-)//)
 1210   FORMAT (2X,16A5///2X,11HDIAMETER  =,F6.1,14X,12HPUMP INERTIA,
        1        5X,1H=,F7.3,2X,12HFT-LB-SEC**2//33X,15HTURBINE INERTIA,
        2        2X,1H=,F7.3,2X,12HFT-LB-SEC**2//)
 1212   FORMAT (2X,24HCONSTANT INPUT TORQUE  =,F8.2,2X,5HLB-FT//)
 1230   FORMAT (2X,12HSPEED RATIO ,6X,10F10.3)
 1232   FORMAT (2X,12HTORQUE RATIO,6X,10F10.3)
 1234   FORMAT (2X,12HINPUT SPEED ,6X,10F10.3)
 1236   FORMAT (2X,12HK-FACTOR    ,6X,10F10.3)
 1300   FORMAT (/20X'VEHICLE DATA  ( 'A10' )  % OLD VEHICLE DATA '
        1,'FORMAT THAT INCLUDES TIRE DATA.'/20X,28(1H-)///2X,16A5//
```

```
     1          /2X,6HWEIGHT,9X,1H=,F8.1,3X,3HLBS,17X,10HAXLE RATIO,
     2          7X,1H=,F7.2//2X,12HFFONTAL AREA,3X,1H=,F9.2,2X,5HSQ FT,
     3          15X,18HAXLE EFFICIENCY  =,F7.2//2X,
     4          16HRCLLING RADIUS =,F9.2,2X,2HFT//2X,
     5          16HDRAG COEFFICIENT,6X,1H=,F11.6//44X,
     6          23HCD SENSITIVITY COEFF  =,F11.6//
     7          2X,20HFEF WHFEL INERTIA  =,F6.3,2X,12HFT-LB-SEC**2,17X,
     8          2HC1,5X,1H=,F11.6//2X,20HPFOPSHAFT INERTIA  =,F6.3,
     9          2X,12HFT-LB-SEC**2,17X,2HC2,5X,1H=,F11.6//)
1330   FORMAT(/20X'AXLE DATA   ( 'A10' )'/20X,25('-')///2X,16A5//
     2         2X,15HREAR AXLE RATIO
     3         ,7X,'=',F7.2//2X'NUMBER OF AXLES = ',I2,/
     4         ,'  AXLE EFFICIENCIES = ',2F7.2)
1320   FORMAT (/20X'VEHICLE DATA   ( 'A10' )'/20X,28(1H-)///2X,16A5//
     1          /2X,6HWEIGHT,9X,1H=,F8.1,3X,3HLBS,12X,15HREAR AXLE RATIO,
     2          7X,1H=,F7.2//2X,12HFFCNTAL AREA,3X,1H=,F9.2,2X,5HSQ FT,
     3          10X,'NUMEER OF TIRES'8X'=',F7.0//2X,
     4          16HDRAG COEFFICIENT,6X,1H=,F11.6,
     5          '  CD SENSITIVITY COEFF  ='F11.6//
     6          //2X,20HPROPSHAFT INERTIA  =,F6.3,
     7          2X,12HFT-LB-SEC**2//
    8           2X,'THERE ARE ',I2,' AXLE(S)'//
    9           2X,'FEAR AXLE EFFICIENCY IS ',2(F6.1,2X))
1340   FORMAT(48X'AXLE SFIN LOSS DATA'/48X19('-')//35X'AXLE 1'
       1,34X'AXLE 2'/35X6('-')34X('-')///7X2(20X'SPEED'10X'TORQUE')
       2,/7X2(20X'(RPM)'10X'(LE-FT)')/7X2(20X6('-')10X7('-'))//)
1341   FCRMAT(25X'NO SFIN LOSS DATA SPECIFIED')
1342   FORMAT(65X'NC SFIN LOSS DATA SPECIFIED')
1355   FCRMAT('+'59XF10.1F16.3)
1400   FORMAT (/20X,13HGEAR DATA   ( ,A10,2H )/20X,25(1H-)///2X,16A5//
     1          /2X,13HGEAR RATIC  =,F7.3,11X,17HINPUT INERTIA   =,F7.3,
     2          2X,12HFT-LB-SEC**2//2X,13HEFFICIENCY  =,F7.3,11X,
     3          17HOUTPUT INERTIA  =,F7.3,2X,12HFT-LB-SEC**2///)
1415   FCRMAT(25XA4' SFIN LOSS DATA'/25X19('-')///
       1,27X'SPEED'10X'TCRQUE'/27X'(RPM)'10X'(LB-FT)'/27X6('-')
       1,10X7('-')//)
1420   FCRMAT(22XF10.1F16.3)
1410   FORMAT(19X'NO 'A4' SPIN LCSS DATA SPECIFIED'19X32('-')/)
1500   FORMAT (/20X,23HACCESSORY LOSS DATA   ( ,A10,2H )/20X,35(1H-)///
     1          10X,16A5///10X,10HINERTIA  =,F6.3,2X,12HFT-LB-SEC**2///
     2          20X,5HSPEED,10X,6HTORQUE/20X,5H(RPM),10X,7H(LB-FT)/
     3          20X,5(1H-),10X,7(1H-)/)
1520   FORMAT (16X,F10.1,F16.3)
1600   FCRMAT (/20X,20HDRIVING SCHEDULE   ( ,A10,2H )/20X,32(1H-)///2X,
     1          16A5///2X,18HINITIAL CONDITICNS,10X,4HTIME,11X,1H=,F6.2,
     2          2X,3HSEC/30X,8HDISTANCE,7X,1H=,F6.2,2X,2HFT/30X,
     3          16HVEHICLE SPEED  =,F6.2,2X,3HMPH/30X,12HACCELERATION,
     4          3X,1H=,F6.2,2X,9HFT/SEC**2/30X,16HSTARTING GEAR  =,I3,
     5          ///2X,7HSEGMENT,11X,
     6          19HDESIRED PERFORMANCE/2X,7(1H-),11X,19(1H-))
1602   FORMAT (/13X,12(1H*),14H SEGMENT TYPE ,12(1H*),18X,15(1H*),
     1          18H SEGMENT ENDPCINT ,15(1H*)/33X,18HCONSTANT  ACCEL TO,
     2          8X,8HTHROTTLE/2X,7HSEGMENT,4X,2(8HCONSTANT,2X),7HPERCENT,
     3          3X,23HAND HOLD  HOLD  RATE OF,3X,2(8HRELATIVE,2X),
       4'PASSING   ABSCLUTE  ABSCLUTE  SEGMENT'/'  NUMBER'5X'ACCELER'
       5,3X'SPEED'5X'WOT'7X'THROTTLE  GEAR  CHANGE    TIME'
       6,6X'DISTANCE  CLEARANCE MILEPOST  VELOCITY  NUMBER'//)
1621   FORMAT (1XI6,4XF10.2)
1622   FCRMAT (1XI6,14XF10.2)
1623   FORMAT (1XI6,24XF10.2)
1624   FORMAT (1XI6,34XF10.2)
```

```
1630    FORMAT
1631    FORMAT  ('+'56XF10:2)
1632    FORMAT  ('+'66XF10.2,41XI6)
1633    FCRPAT  ('+'76XF10.2,31XI6)
1634    PORMAT  ('+'86XF10.2,21XI6)
1635    FORMAT  ('+'96XF10.2,11XI6)
1636    POPMAT  ('+'106XF10.2,1XI6)
1640    FCREAT(/' ? PRNTPC- RELCAL ERF ON 'A5,3XA10/)
1700    FORPAT  (20X,15HSHIPT LOGIC   ( ,A10,2H )/20X,27(1H-)///2X,16A5//
        1        /2X,21HTHIS TRANSMISSION HAS,I4,7H  GEARS)
1710    PORPAT  (//2X,10HUP SHIPT  ,I3,2H -,I3)
1712    POFPAT  (//2X,10HDCWN SHIPT,I3,2H -,I3)
1714    FORMAT  (1H+,21X,12HSHIFT TIME =,F6.3,4H SEC)
1720    PORPAT  (22X,21HVACUUM (IN-HG)          ,10F8.2)
1721    PORPAT  (22X,21HTHFOTTLE (DEGRFES)      ,10PA.2)
1722    FORPAT  (22X,21HTHFOTTLE (FCT SOT)      ,10P8.2)
1724    FORPAT  (22X,21HVFHICLE SPEED (EPH)     ,10F8.2)
1730    PORMAT  (22X,21HENGINE SEEED (FPM)      ,10F8.2)
1732    PCREAT  (22X,21HFROPSHAFT SPEED (RPM),10P8.2)
1740    FCRPAT  (/22X,26HDETENT CVERIOE DESCRIPTION/)
1800    FORMAT  (/20X,23HFCUTE SEECIFICATICN   ( ,A10,2H )/20X,35(1H-)/
        1,//10X,16A5/
        1/23X'DISTANCE'8X'PERCENT'9X'FOAD'9X'WIND SPEED'/
        1,10X'POINT'9X'(MILES)'10X'GRACE'7X'COEFFICIENT'7X'(MPH)'/
        2,10X5(1H-)8X7(1H-)10X5(1H-)7X11(1H-)4X10('-')/)
1802    PORMAT(10XI4,1X4F15.3)
1900    PORMAT(/20X'TIFE DATA  ( 'A10' )'/20X25('-')//2X16A5//
        1,'  ROLLING RACIUS   ='F9.3'    FT'12X'C1 ='F11.6//
        2,'  C2 ='F11.6//
        3,'  TIFE EFFICIENCY ='4XF6.3
        4,'  WHEEL INERTIA   ='4XF6.3'  FT-LB-SEC**2')
2500    FORMAT(/20X,'TRANSMISSICN DATA  ( ',A10,' )',/
        2,20X,34('-')///2X,16A5//
        3,20(30X,I2,' - ',A10/))
C
        END
```

```
      FUNCTION RCRCNT(STRING,IWRDS)
C
C     ENTRY POINTS:   RCRCNT
C
C     SUBROUTINES CALLED:     NWRCNT
C
C     CALLED BY:      ASCIZ,  ICRCNT, INPEAT
C
C****************************************************************
C
C
      DIMENSION STRING(IWRDS),CHARS(4)
C
C    FUNCTION RCRCNT COUNTS THE # OF CHAR'S IN ALPHA-NUMERIC STRING
C     INCLUDING IMBEDDED BLANKS AND EXCLUDING TRAILING BLANKS.
C
      NW=NWRCNT(STRING,IWRDS)                           !SET PTR TO LAST NON-BLANK WORD IN STRING.
      IF(NW.GT.0) GO TO 30                              !(STRING ALL BLANKS?) NO.
      RCRCNT=0.                                         !YES,SET CHAR CNT ZERO.
      RETURN                                            !BYE.
C
   30 DECODE(5,31,STRING(NW)) CHARS                     !PUT LAST 4 CHAR'S OF WRD INTO CHARS ARRAY.
   31 FORMAT(1X,4A1)
C
      RCRCNT=(NW-1)*5+NWRCNT(CHARS,4)+1                 !CALC # OF CHAR'S IN STRING.
      RETURN                                            !BYE.
      END
      FUNCTION NWRCNT(STRING,LWRDS)
C
C     ENTRY POINTS:   NWRCNT
C
C     CALLED BY:      DSKDIR, INPEAT, RCRCNT, READPD
C
C**************************************************************
C
C
      DIMENSION STRING(LWRDS)
      DATA HBLANK/' '/
C
      DO 20 NWRCNT=LWRDS,1,-1                         !FIND LST NON-BLANK WRD IN STRING.
      WORD=STRING(NWRCNT)
      IF(WORD.NE.HBLANK .AND. WORD.NE.0) GO TO 30     !(GOT IT?) YES.
   20 CONTINUE                                        !NO, NEXT.
C
      NWRCNT=0                                        !ALL WORDS IN STRING ARE BLANK.
   30 RETURN                                          !*DONE, BYE.
      END
      FUNCTION ICRCNT(STRING,LWRDS)
C
C     ENTRY POINTS:   ICRCNT
C
C     SUBROUTINES CALLED:     RCRCNT
C
C     CALLED BY:      DSKDIR, HLPCMD, INPRAT, INPDIA, VALID2
C
C************************************************************************
C
      DIMENSION STRING(LWRDS)
C
      ICRCNT=RCRCNT(STRING,LWRDS)                         !INTEGER CALL OF RCRCNT.
      RETURN                                              !BYE.
      END
      SUBROUTINE READEC
     1         ( NARRAY,LENGTH,NECARD,HWORD,NPOINT,IFLAG,A1,A2,A3,A4 )
C
C     ENTRY POINTS:   NXTCRD, READPD
C
C     SUBROUTINES CALLED:     NWRCNT
C
C     CALLED BY:      INPEAT
C
C*******************************************************************
C
C
      LOGICAL  ENDE,LEFIL
C
      DIMENSION  A1(1),A2(1),A3(1),A4(1),ALPHA(11)
C
      COMMON /ENDOF/   ENDE,IERST,LEFIL
C
      DATA HBLANK/' '/,HSTAR /'*'/
C
C    NARRAY = NUMBER CF ARRAYS TC BE FILLED
C    LENGTH = LENGTH CF ARRAYS TO BE FILLED
C    NPCARD = NUMBER OF DATA PCINTS TC BE READ OFF EACH CARD
C    HWORD  = ALPHA WCRD WHEN FOUND ON DATA CARD STOPS INPUT
C    NPOINT = NUMBER OF DATA PCINTS ACTUALLY READ INTO EACH ARRAY
C    IFLAG  = RETURN STATUS FLAG WHERE 1-NC END , 2-HWORD , 3-* , 4-EOF
C    A1 , A2 , A3 , A4  = ARRAYS TO BE FILLED WITH DATA
C
C********************************************************************
C
      NPOINT = 0                                                ! ZERO COUNT OF DATA POINTS/RECORD
      ENDE   = .FALSE.                                          ! ASSUME NOT EOF
      IENT=0                                                    !FLG NORMAL ENTRY.
      IF ( NARRAY.LT.1 )  GO TO 80                              !(CLD CALL JUST TO LOOK AT NEXT CARD?)YES, SWITCH.
      GO TO 90                                                  !NO,GO READ IN DATA.
C
      ENTRY NXTCRD(HWCRD,IFLAG,WCRD)                            !ENTRY TO LOOK AT NEXT CARD ONLY.
   80 IENT=1                                                    !FLG NXT CRD ENTRY.
      GO TO 120                                                 !GO LOOK NEXT CARD.
C
   90 DO 200  IB = 1,LENGTH,NECARD
      IE     = IB + NECARD - 1                                  !CALC PTR TO LAST ELEMENT TO BE READ THIS PASS.
      IF ( IE.GT.LENGTH )  IE = LENGTH                          !(WILL WE READ PAST END OF ARRAY?)YES, POINT TO END OF ARRAY
.
C
C    READ DATA CARD SET
C
      READ  (4,1000)  (A1(I),I=IB,IE)
      NPFLG=1                                                   !FLG 1ST CARD OF SET BEING SCANNED FOR # OF DATA PTS/REC.
   95 BACKSPACE 4                                               !POST CTR FIL.
      READ(4,1100) (ALPHA(I),I=1,NPCARD)                        !REREAD USED LATER TO CALC NPOINT
C
      J=NWRCNT(ALPHA,NPCARD)                                    !COUNT DATA PTS ON CARD.
      IF(J.EQ.0) GC TC 97                                       !(CARD DATA FIELD BLANK?) YES, GO FLG.
      NECINT = IB + J - 1                                       !CALC # OF DATA PTS./REC
C                                                               !SAME # OF PTS./REC ASSUMED
      GO TO (98,95),NPFLG                                       ! (READ REC 2 OR 3 NXT?)
   97 NPFLG=2                                                   ! FLG 1ST REC BLANK, SCAN 2ND REC TO CALC DATA PTS/REC.
C
   98 IF ( NARRAY.LT.2 )  GC TO 120                             !(READ RECORD?) NO.
      READ  (4,1000)  (A2(I),I=IB,IE)                           ! YES.
      IF(RPFLG.EQ.2) GO TO 95                                                 !(FIRST DATA REC BLANK?) YES, GO SCAN ON 2ND REC.
  95  IF ( NARRAY.LT.3 )  GO TO 120                                           !(READ RECORD?) NO.
      READ  (4,1000)  (A3(I),I=ID,IE)                                         ! YES.
      IF ( NARRAY.LT.4 )  GO TO 120                                           !(READ RECORD?) NO.
      READ  (4,1000)  (A4(I),I=IB,IE)                                         ! YES.
C
C     CHECK FOR EOF OR COMMAND CARD (* IN COLUMN 1) OR SPECIFIED
C     ALPHAMERICS (IN COLUMNS 1-5) ON NEXT DATA CARD
C
 120  READ  (4,1120,END=840)  COL1                                            !READ NEXT CARD,(EOF?) YES.
      BACKSPACE 4                                                             !NO, POSI CTRFIL.
 140  IF ( COL1.EQ.HSTAR )  GO TO 830                                         !(NXT REC COMND?) YES.
      READ  (4,1140)  WORD                                                    ! NO.
      BACKSPACE 4                                                             !POSI CTRFIL.
      IF ( WORD.EQ.HWORD )  GO TO 810                                         !(NXT REC CONT DATA?) YES.
C
      IF(IENT.EQ.1) GO TO 800                                                 ! NO,(LOOK NXT CARD ONLY?) YES.
 200  CONTINUE                                                                ! NO.
C
C**************************************************************************
C
C     SET PROPER RETURN STATUS FLAG AND RETURN
C
 800  IFLAG  = 1                                                              ! ERROR OR UNKNOWN NXT REC TYPE
      GO TO 900
 810  IFLAG  = 2                                                              ! FOUND HWORD ON NXT REC, DATA CONT.!
      GO TO 900
 830  IFLAG  = 3                                                              !NXT REC IS COMND.
      GO TO 900
 840  IFLAG  = 4                                                              ! FOUND EOF ON DATA FIL
      ENDE   = .TRUE.                                                         ! SET EOF FLAG
C
 900  RETURN                                                                  ! DONE, BYE.
C
C**************************************************************************
C
C     FORMAT STATEMENTS
C
1000  FORMAT (12X,11F6.1)
1100  FORMAT (12X,11(A5,1X))
1120  FORMAT (A1)
1140  FORMAT (A5)
C
      END
      SUBROUTINE REMAP
C
C        ENTRY POINTS:    REMAP
C
C        SUBROUTINES CALLED:      ENGINE, PRNOUT
C
C        CALLED BY:       INPDAT
C
C*****************************************************************
C
C
      DOUBLE PRECISION   ENAME,DN
C
      LOGICAL SRPM,SBS,STOR,SBMEP,SHP,SLBHR,SBSFC,SGALHR,LWOT
     1,LRPM,PRPM,LBS,PBS,LTOR,PTOR,LBMEP,PBMEP,LHP,PHP,LLBHR
     2,PLBHR,LBSFC,PBSFC,LGALHR,PGALHR,LPRNT
      COMMON /TMAP/ TCFC,LWOT,TWOT,TRIM
      COMMON /GET/ DT,DN,NUG(20),JENG(20),IENG
      COMMON /ENDOF/ ENDE,IPRNT
      COMMON /ENGMAP/ RPMAX(2),RPMIN(2),NRPM(2),RPMB,TORQE,FRATE,VAC,
     1THR,MAPOR,
     1IEFRE,NTOR(2,20),EMAP(20,20,8),ERPM(2,20),EMMIN(2),SPIDLE(2)
      COMMON /LDIMEN/ LRPM,PRPM,LPS,PPS,LTOR,PTOR,LBMEP,PBMEP,LHP,PHP,
     1LLBHR,PLBHR,LBSFC,PBSFC,LGALHR,PGALHR,LPRNT
      COMMON /IC/ECOM(16),ENAME(2),DISP,ICYL,IMIN,IMAX,THRMAX,THRMIN
     1,EINER,ECRE,STROKE,FSPCR,NCYCLE
      COMMON /CINMAP/ EMAPO(20,20,4),ERPMO(20),NTORO(20),NRPMO
      IF(LRPM) GO TO 12
C
C    CONVERT PISTON SPEED TO RPM
C
      DUM=6./STROKE
      DO 11 I=1,NRPMO
   11 ERPMO(I)=DUM*ERPMO(I)
   12 IF(LTOR) GO TO 17
      IF(LBMEP) GO TO 14
C
C    CONVERT HP TO LB-FT
C
      DO 13 I=1,NRPMO
      DUM=5252./ERPMO(I)
      DO 13 J=1,20
   13 EMAPO(I,J,1)=DUM*EMAPO(I,J,1)
      GO TO 17
C
C    CONVERT BMEP TO LB-FT
C
   14 AK=150.8
      IF(NCYCLE.EQ.2) AK=75.4
      DUM=DISP/AK
C
C    STORE ENGINE DATA IN TEMPORARY LOCATIONS
C
      DO 16 I=1,NRPMO
      DO 16 J=1,20
   16 EMAPO(I,J,1)=DUM*EMAPO(I,J,1)
   17 CONTINUE
C
C    COMPUTE NEW ENGINE MAP
C
      DO 30 I=1,NRPMO
      DO 30 J=1,20
      TORQE=EMAPO(I,J,1)
      RPME=ERPMO(I)
      LWOT=.FALSE.
      CALL ENGINE
      EMAPO(I,J,2)=FRATE
      EMAPO(I,J,3)=THR
   30 EMAPO(I,J,4)=VAC
C
C    RETRIEVE DATA FROM DUMMY LOCATIONS
C
      NDUM=NRPMO
      NRPMO=NRPM(IENG)
      NRPM(IENG)=NDUM
      DO 35 I=1,20
      DUM=ERPMO(I)
      ERPMO(I)=ERPM(IENG,I)
      ERPM(IENG,I)=DUM
      NDUM=NTORO(I)
      NTORO(I)=NTOR(IENG,I)
      NTOR(IENG,I)=NDUM
      DO 35 J=1,20
      DO 35 K=1,4
      DUM=EMAPO(I,J,K)
      EMAPO(I,J,K)=EMAP(I,J,K)
   35 EMAP(I,J,K)=DUM
C
C    SET UNITS FLAGS FOR PRINT
C
      SRPM=PRPM
      PRPM=LRPM
      SPS=PPS
      PPS=LPS
      STOR=PTOR
      PTOR=LTOR
      SBMEP=PBMEP
      PBMEP=LBMEP
      SHP=PHP
      PHP=LHP
      SLBHR=PLBHR
      PLBHR=LLBHR
      SBSFC=PBSFC
      PBSFC=LBSFC
      SGALHR=PGALHR
      PGALHR=LGALHR
      IPRNT=1
C
C    PRINT REMAPPED ENGINE DATA
C
      CALL ERNOUT
C
C    RESET UNITS FLAGS FOR REST OF RUN
C
      PRPM=SRPM
      PPS=SPS
      PTOR=STOR
      PBMEP=SBMEP
      PHP=SHP
      PLBHR=SLBHR
      PBSFC=SBSFC
      PGALHR=SGALHR
      NDUM=NRTMC
      NRPMC=NRPM(IENG)
      NRPM(IENG)=NDUM
      DO 40 I=1,20
      DUM=ERPMO(I)
      ERPMO(I)=ERPM(IENG,I)
      ERPM(IENG,I)=DUM
      NDUM=NTCRO(I)
      NTCRO(I)=NTOR(IENG,I)
      NTOR(IENG,I)=NDUM
      DO 40 J=1,20
      DO 40 K=1,4
      DUM=EMAPO(I,J,K)
      EMAPO(I,J,K)=EMAP(I,J,K)
40    EMAP(I,J,K)=DUM
      RETURN
      END
      SUBROUTINE SCALEN
C
C        ENTRY POINTS:    SCALEN
C
C        SUBROUTINES CALLED:      PRNOUT
C
C        CALLED BY:       INPDAT
C
C*********************************************************************
C
C
      LOGICAL            ISCALE,LINPRN,LTRRZ
C
      DOUBLE PRECISION    ENAME,ON,GNAME
C
      COMMON /PRNLIN/  LINPRN,MILIN,SECLIN,ENDLIN,ALINN,LSCALE
      COMMON /GET/     CT,UN,NOG(20),JENG(20),IENG
      COMMON /ENGMAP/  RPMAX(2),RPMIN(2),NRPM(2),RPME,TORQE,FRATE,VAC,
     1                  THR,MAPOK,IEBRE,BTOB(2,20),EMAP(20,20,8),
     2                  ERPM(2,20),ENMIN(2),SPIDLE(2)
      COMMON /IC/ ECOM(16),ENAME(2),DISP,ICYL,EMIN,IMAX,THRMAX,
     1                  THRMIN,EINER,BORE,STROKE,FSPGR,NCYCLE
      COMMON /CCNST/   FRC1,FRC2,FAC,CD,AREA,VWIND,WGT,FGC,RRAD,BAR,
     1         GRAT(20),NUMG,NGEAR,AIW,AIP,AI2,ERAT(20),ERAB(2),
     2AIE,AIA,AI1,EPER,CDC,PHI,PSI,AIGIN(20),AIGOUT(20),WLSG,LTRRZ
     3,NGRLSS(20),GRPM(20,20),GRTORC(20,20),GNAME(20),GCOM(16)
      COMMON /CIDIO/   CDISP,CBORE,CSTROK,IOCYL
      COMMON /EBDOF/   ENDE,IFRST
C
C
C*********************************************************************
C
C     DEFINE SCALING RATIOS
C
      DRATIO = DISP/CDISP
      SRATIO = CSTROK/STROKE
C
C     DETERMINE NUMBER OF ENGINE MAPS TO RESCALE
C
      NUME   = 1
      DO 100  I = 1,NOMG
      IF ( JENG(I).EQ.2 )  NUME = 2
  100 CONTINUE
C
C     DEFINE CONTROL LOOP PARAMETERS
C
      DO 400  NENG = 1,NUME
      KENG    = (NENG-1) * 4
      NRPMP   = NRPM(NENG)
      DO 200  I = 1,NRPMP
C
C     RESCALE RPM
C
      ERPM(NENG,I)      = SRATIO * ERPM(NENG,I)
      DO 200  J = 1,20
C
C     RESCALE TORQUE
C
      EMAP(I,J,1+KENG)  = DRATIO * EMAP(I,J,1+KENG)
C
C     RESCALE FUEL RATE
```

```
  200   EMAP(I,J,2*KENG) = DRATIO * SRATIO * EMAP(I,J,2*KENG)
C
C     REDEFINE ENGINE MAP BOUNDARIES
C
        RPMAX(NENG)   = EFEM(NENG,NRPMF)
  400   CONTINUE
C
C     PRINT RESCALED ENGINE MAP
C
        IF ( LIPPEN )  GO TO 900
        WRITE (6,1000)  NONE,ODISE,OBCRE,OSTROK,IOCYL,
        1                       CISP, BORE,STROKE,ICYL
        IPRNT  = 1
        DO 600  IENG = 1,NONE
  600   CALL PRNOUT
C
  900   RETURN
C
C
C**********************************************************************
C
C     FORMAT STATEMENTS
C
 1000   FORMAT (1H1///10X,13HTHE FOLLOWING,I3,22H ENGINE MAPS HAVE BEEN,
        1         37H RESCALED WITH THE FOLLOWING CHANGES /10X,74(1H-)////
        2         /25X,5HDISPL,5X,5HBORE ,5X,6HSTROKE,4X,6HNO.CYL
        3         /25X,5H-----,5X,5H---- ,5X,6H------,4X,6H------
        4         ///15X,3HOLD,F12.1,2F10.3,I8/15X,3H---
        5         ///15X,3HNEW,F12.1,2F10.3,I8/15X,3H---////)
C
        END
```

```
      SUBROUTINE SHIFTS
C
C     ENTRY POINTS:   SHIFTS
C
C     SUBROUTINES CALLED:     DEBUG
C
C     CALLED BY:      SIMCTR, SIMINT
C
C     EDIT HISTORY
C
C     [621]/SS-5-9-79         ADD MODIFY LOCKUP GEAR IF OVER A
C                             SPECIFIED RPME.
C
C*******************************************************************
C
C
      DIMENSION IGRS(2)
C
      DOUBLE PRECISION   CNAME,SNAME,UN,GNAME
C
      LOGICAL LISH,PARAE,LGFREE,LPSHP,LVAC,LENG,GDAT,LDETV,LDETE
     1,LDETNT,LTBRZ,LOCKUP,LRPE
      LOGICAL LETH
C
      COMMON /DSHIFT/ LRPH,AVEL
      COMMON /V2MISC/  IOCKUP(20)
      COMMON /CDEBUG/ IDEBUG
      COMMON /GET/ UT,UB,NUG(20),JENG(20),IENG
      COMMON /CONSHE/ LISH,ISEPT,STIME
      COMMON /TFORPE/ TORQP,REME,TORQW,RPMW,TORQ1,RPM1
      COMMON /SFPTL/ SNAME,SCOM(16),GOVEST(4),OUTRPM(4),NGPT,IGF(38),
     1IGT(38),SHFTIN(38),LGFREE(38),LPSHP(38),EPPST(38),ABRUPT(38),
     2SHFTPT(10,38),SHFTRP(10,38),LVAC,LENG,GDAT,NSPTS(38),LDETNT,
     3DETPT(38),DETREM(38),PARAE,LDETE,LDETV,GOVLIN,LDTH
      COMMON /TORCON/ TORBPK,TORQ2,RPM2,CCAST,SR,TR,TRD(20),SRD(20),
     1AKE(20),NTC,SRC(20),AKC(20),NTC,NTBP,TIN(20),TOUT(20),SPIN(20),
     2SOUT(20),NTORP,CDIAM,CNAME,CCOM(16),CONTOR
      COMMON /ENGMAP/ EMAX(2),EPMIN(2),NEPM(2),BPME,TORQE,FRATE,VAC,
     1THE,NAEOR,
     1IEFRE,NTCR(2,20),EMAP(20,20,8),ERPM(2,20),EMMIN(2),SPIDLE(2)
      COMMON /THAF/ TORC,LWOT,TWOT,TMIN
      COMMON /CONST/ FRC1,FRC2,EAC,CD,AREA,VWIND,WGT,EGC,WRAD,RAR,
     1GRAT(20),NUMG,NGEAR,AIW,AIP,AI2,ERAT(20),ERAR(2),AIE,AIA,AI1
     2,BFPR,CDC,FH1,PSI,AIGIN(20),AIGOUT(20),WLSG,LTBRZ
     3,NGRLSS(20),GREN(20,20),GRTORC(20,20),GNAME(20),GCOM(16)
      COMMON /CNTRL/ IC,TCLD,VCID,T,V,ACCEL,D,DT
      COMMON /SEGNO/ITS
C
      DATA HBEFOR/'EEEOF'/
C
C     TURN SHIFT FLAG OFF AND DETERMINE WHAT GEARS TO UP/DOWN SHIFT TO
C     IF REQUIRED
C
      LISH=.FALSE.                                               !ASSUME NO SHIFT.
      J=0                                                        !SET TO NO PTRS SET.
      DO 5 I=1,((NUMG-1)*2)                                      !LOOP THRU ALL SHIFT PTS SET PTRS TO DATA.
      IF(IGF(I).NE.NGEAR) GO TO 5                                !(GOT DATA FOR GEAR WE ARE IN?)NO.
      J=J+1                                                      !YES, INC CNT OF PTRS SET.
      IF(IGT(I).GT.NGEAR) IGRS(1)=I                              !(UP?)YES, SET PTR TO UPSHIFT DATA.
      IF(IGT(I).LT.NGEAR) IGRS(2)=I                              !(DOWN?)YES, SET PTR TO DOWNSHIFT DATA.
      IF(J.EQ.2) GO TO 7                                         !(HAVE WE GOT UP & DOWN SHIFT PTR'S)YES.
```

```
    5   CONTINUE                                                                   !TRY NEXT DATA SET.
C
C       SET CURRENT VALUES OF SHIFT PARAMETERS TO BE MONITORED
C
CCC 7   X=VAC
CCC     IF(.NOT.LVAC) X=(TORQE-TMIN)/(TWOT-TMIN)*100.
    7   X=(TORQE-TMIN)/(TWOT-TMIN)*100.
        IF(LVAC) X=VAC
        IF(LDTH) X=THR
        Y=REMP
        IF(LENG) Y=RPEE
        IF(EARAE) Y=V
        IF(.NOT.LDETNT.OR.ITS.NE.3) GO TO 10
C
C       IF IN CONSTANT PERCENT WOT SEGMENT AND THROTTLE IS WIDE OPEN USE
C       DETENT  OVERRIDE SHIFT CRITERIA
C
        PCTHR  = X
CCC     IF(LVAC) PCTHR=(TORQE-TMIN)/(TWOT-TMIN)*100.
        IF(LVAC.OR.LDTH) PCTHR= (TORQE-TMIN)/(TWOT-TMIN)*100.
        IF(PCTHR.LT.99.) GO TO 10
        IF(NGEAR.EQ.NUMG) GO TO 8
        IGR=IGRS(1)
        IF(Y.GE.DETRPE(IGR)) GO TO 62
        IF(NGEAR.EQ.1) RETURN
    8   IGR=IGRS(2)
        IF(Y.LE.DETRPE(IGR)) GO TO 61
        RETURN
C
   10   DO 50 IS=1,2                                                               !TRY UP AND DOWN SHIFT.
        IGR=IGRS(IS)                                                               !SET PTR TO DATA TO LOOK AT.
        IF (IS.EQ.1.AND. NGEAR.EQ.NUMG )  GO TO 50                                 !(TRYING UPSHI & IN LAST GEAR?)YES. CAN'T UPSHIFT.
        IF(IS.EQ.2.AND.NGEAR.EQ.1) RETURN                                          !(TRYING DOWN SHIFT & IN FIRST GEAR?)YES - CAN'T.
        IF ( Y.LT.SHFTRE(1,IGR) )  GO TO (50,60),IS
        IF ( Y.GT.SHFTRE(NSPTS(IGR),IGR) ) GO TO (60,64),IS
CCC     DO 40  I = 2,(NSETS(IGR)-1)
        DO 40  I = 2,NSETS(IGR)                                                  !JD FIX TO GET BETWEEN LAST 2 PTS.
        IF ( Y.GT.SHFTRE(I,IGR) )  GO TO 40
        GO     =  (SHFTET(I-1,IGR)-SHFTET(I,IGR)) * (Y-SHFTRP(I,IGR))
        1            / (SHFTEP(I-1,IGR)-SHFTEP(I,IGR))  +  SHFTPT(I,IGR)
        IF ( LVAC )  GO TO (30,20),IS
        IF(IS.EQ.2) GO TO 30
   20   IF ( X.LE.GO )  GO TO 60
        GO TO 50
   30   IF ( X.GE.GO )  GO TO 60
        GO TO 50
   40   CONTINUE
        GO TO 60
   50   CONTINUE
        RETURN                                                                     !NO SHIFT.
C
   60   CONTINUE
        IF(IS.EQ.1) GO TO 62                                                       !(UP SHIFT?)YES.
   61   IF(ITS.EQ.2.AND.BERR.LT.0..AND.LOCKUP(NGEAR)) RETURN                       !(CON VEL DRS SEG AND DOWN GRADE?) YES.
   62   IF(ITS.NE.1) GO TO 65                                                      !(CONST ACCEL?)NO.
        IF(.NOT.LEPH) GO TO 65                                                     !(END DRS SEG SPEC BY VEL?)NO.
        IF(ABS(ACCEL).LT.1.E-3) GO TO 65                                           !(ZERO ACCEL?)YES.
        IF((AVEL-VOLD)*1.466667/ACCEL.GT.SHFTIM(IGR).OR.LOCKUP(NGEAR))             !(WILL WE GO PAST END DRS SEG DURING SHIFT?)
        1 GO TO 65                                                                 !NO.
   64   RETURN                                                                     !YES, IF WE GET TO HERE DON'T SHIFT. BYE.
C
```

```
65    IF(IDEBUG.EQ.2) CALL DEBUG(HBEFOR)                                    !DEBUG SHIFTS) YES.
      NGEAR=IGT(IGR)                                                        !GET NEW GEAR #.
      IENG=JENG(NGEAR)                                                      !GET NEW ENG MAP #.
      STIME=SHFTIM(IGF)                                                     !GET SHIFT TIME.
      LLSF=.TRUE.                                                           !FLAG THAT WE SHIFTED.
      IF(.NOT.LPDLOK) GO TO 70                                              ![621]
      LOCKUP(NGEAR)=.FALSE.                                                 ![621]
      IF((IFIX(RPME).GE.MDLKRE) .AND. (NGEAR.EQ.MDLKGR))
     2         LOCKUP(NGEAR)=.TRUE.                                         ![621]
70    RETURN                                                                !SO LONG.
      END
```

```
C
C       ENTFY ŔCINTS:    SIMCTR
C
C       SUBROUTINES CALIEC:     CTRLD,  DEEUG,  DSK,    DSKCTR, EXIT,
C                       GCBACK, ITERAT, KETIME, RESETM, SHIFTS, SIMINT,
C                       SIMLPT, SIMSTS
C
C       CALIEC EY:      INPBAT
C
C****************************************************************
C
C       EDIT HISTORY
C
C       [601]/SS-01-26-78       IN CONST THROT DOWNSHIFT GIVE A REASONABLE START ACCELERATION
C       [602]/SS-1-31-78        SAVE GEAR AT BEGIN OF EACH TIME STEP.
C       [603]/SS-2-22-78        REPLACE CCNS THOT SECTION WITH CALL TO ITERAT
C       [604]/SS-2-22-78        SKIP THIS TYPE OF TESTING AHEAD IF
C                               NCT TRUCK CAUSE NOT TRUCKS HAVE MORE
C                               THAN 2 RPMS PER SHIFT LINE.
C       [606]/SS-3-28-78        IF CCNST ACCEL SEG TO AB VEL, AND VEL HIGHER THAN
C                               DESIREC V, SWITCH TO CONST VEL INORDER TO
C                               DECCEL TO REQ VEL THEN ENDSEG.
C       [607]/SS-4-10-78        CIUTCH
C       [610]/SS-6-19-78        ALLOW TO SHIFT IN FIRST TIME STEP EVEN IF WITHIN DELAY.
C       [613]/SS-6-19-78        NEW STARTUP PROCEEDURE
C       [614]/SS-7-5-78         HISTOGRAM OUTPUT
C       [615]/SS-10-4-78        TCRN CFF FUEL FLOW IF DIESEL AND DECCELERATION AND NOT
C                               NCT STOPPED.
C       [620]/SS-4-3-79         TURN CFF BRAKES IF UP SHIFTING.
C       [621]/SS-5-9-79         AED MODIFY LOCKUP GEAR IF OVER A
C                               SPECIFIED RPME.
C       [623]/SS-7-9-79         AED A MODIFY TOPSPEED TO GO TO SUMMARY IF HIT TOPSPEED.
C       [625]/JC-10-7-80        SHIFT LOGIC IN DEGREES OF THROTTLE ROTATION
C       [626]/JC-03-23-81       UPSHIFT AND DOWNSHIFT ON A GRADE TAKING INTO
C                               ACCOUNT THE DIFFERENCE BETWEEN ENGINE RPM,
C                               CUTPUT OR PROPSHAFT RPM AND MPH IN THE
C                               SHIFT LOGIC
C       [627]/JC-02-28-81       GET CCRRECT TCTALS FOR ACCESSORY LOSSES
C****************************************************************
C
        INCLUDE 'COPES/BCLIST'
C
C
        LOGICAL LSEC,LMILE,LPASS,IMP,LTHR,ENDSEG,LSHFT,LSTRUP
        1,ARRIVE
C
        DATA HAITIP/'AFTER'/
C
C****************************************************************************
C
        PCTNOT(I)=100.*(TCRQE-TMIN)/(TNOT-TMIN)
D       PRINTG=-1
        SIMCDE=SIMODT
        CALL SIMINT                                             ! BRING VEHICLE UPTO INITIAL CCNDITIONS
C
        SDELAY=.8
C       SDELAY=SHITIM(1)
        IF(IPER.GT.0.)SDELAY=SDELAY*EPER
        BPEFO=BEFF                                              !SAVE ROAD GRADE.
        IERCND=0                                                        !ASSUME NO ERR.
        KEND=0                                                          !SET PRINT LINE FLG.
        GO TO 6
C
    5   IF(ISEG+NDSEG.LE.NSEG) GO TO 6                                  !(END DRS SECT?) NO.
        IF(ISTSEC) GO TO 111                                            !YES,(LAST DRS SECT?) YES.
        IPRNT=206                                                       !NO, SET DSK FLG TO NXT DRS SECT
        CALL DSK                                                        !GET NXT DRS SECT
        IF(IPRNT.NE.6) RETURN                                           !(DSK ERR?) YES.
        CALL DSKCTR(0,'SIMCTR')                                         !RELEASE DB FILES
        ISEG=1                                                          !SET FOR 1ST SEG OF DRS SECT
        ISEGO=ISEG                                                      ![610]SET OLD SEG CNTR.
C
C     INITIALIZE ALL PARAMETERS FOR SEGMENT OF DRIVING
C     SCHEDULE TO BE EXECUTED
C
    6   CALL KPTIME(2)                                                  !SAVE START OF DRS SEG RUNTIM
        ITS=ITYSEG(ISEG)                                                !SET SEG TYPE FLG.
        ITSAV=ITS
        NSPSEG(ISEG)=0
        ENDSEG=.FALSE.                                                  !FLG NOT END DRS SEG.
        DT=DEFDT                                                        !SET TIME STEP.
        AASE=ASEG(ISEG)                                                 !GET IT.
        AVSE=VSEG(ISEG)
        ATSE=TSEG(ISEG)
        APWC=PWCT(ISEG)
        APWCO=APWC*.01
        SATH=ATHOLD(ISEG)
        NNGS=NGSEG(ISEG)
        ATHR=THRATE(ISEG)
        ADSE=DSEG(ISEG)
        APCS=PCSEG(ISEG)
        APOS=POSTSE(ISEG)
        AVEL=VELSEG(ISEG)
        ARRIVE=.FALSE.
        DSTART=CUMD*5280.
        TSTART=T
        VSTART=V
        LSTFUP=.FALSE.                                                  ![613]
CSIE    IF(V.LT.1.E-5)LSTFUP=.TRUE.                                     ![613]
        ASTART=ACCEL
        RT=0.
        RD=0.
        LSEC=.FALSE.
        LMILE=.FALSE.
        LPASS=.FALSE.
        LMP=.FALSE.
        LMPH=.FALSE.
        LTHR=.FALSE.
        LSHFT=.FALSE.
        ACCC=0.
C
C     SET END OF SEGMENT FLAGS
C
        IF(ATSE.GT.-.5)  LSEC=.TRUE.
        IF(ADSE.GT.-.5)  LMILE=.TRUE.
        IF(APCS.GT.-.5)  LPASS=.TRUE.
        IF(APOS.GT.-.5)  LMP=.TRUE.
        IF(AVEL.GT.-.5)  LMPH=.TRUE.
        IF(ABS(ATHR).GT.1.E-20)  LTHR=.TRUE.
        IF(LMILE) ADSE=ADSE*5280.
```

```
          IF ( LME )  AEOS = APOS * 5280.
          TSAVE=0.
          DSAVE=0.
          IF(NNGS.EC.0) GC TO 9                                            !(HOLD A GEAR NO MATTER WHAT?) NO,
          LSHFT=.TRUE.                                                     !YES, SET FLG TO PREVENT SHIFTING.
          NGEAR=NNGS                                                       !GET GEAR # TO HOLD.
    9     IENG=JENG(NGEAR)                                                 !GET ENG MAP # TO USE.
          IENG=JENG(NGEAR)
          NGCNT=0                                                          !ZERO CNT TO FIND GEAR FOR CONST VEL SEG.
C
C        GO TO PROPER CONTROL LOGIC DEPENDING ON TYPE OF SEGMENT
C
   10     NGEARO=NGEAR                                                     !(602)
          GC TO (40,20,60,80),ITS                                   ![ACCEL/VEL/%WOT/ACCEL TO CON ACCEL & HOLD %WOT] (SEG TYPE LOGIC?).
C
C        CONSTANT VELOCITY SEGMENT
C
   20     DT=DEFDT
          IF(.NCT.LCCKUP(NGEAR))DT=.25
          TOLD=T
          T=T+DT
          VOLD=V
          LWCT=.FALSE.
          ACCEL=0.
          IF(ISTRUP.AND.AVSE.NE.0.)CALL SIMLPT(12)                         ![613]ERROR
          IF(ISTRUP.AND.AVSE.EQ.0.)ISTRUP=.FALSE.                          ![613]IF IDLE STEP, TURN OFF START FLAG
          CALL GOBACK
C
          IF(NGCNT.LT.MXNGCN) GO TO 205                                    !(REACHED LIMIT ON FIND GR?)NO.
          IF(NSFSEG(ISEG).LT.MXNGCN)GO TO 205
C
          CALL SIMLPT(3)                                                   ! ? SIMCTR- SHIFT STUTTER DETECTED.
          RETURN
C
  205     IF(EPER.GT.0.) GO TO 206                                         !(UPHILL?)YES.
          IF(NGCNT.LE.2) GO TO 204                                         !(HERE FOR 3RD GEAR OR GEATER TIME THID DRS SEG?) NO.
  206     IF(NGEAR.EQ.NGOLD) GO TO 201                                     !YES,(VEH SETTLE INTO COAST VEL?)YES.
C
C        ALLOW CAR TC REACH CONSTANT VELOCITY REQUIRED BY SEGMENT IF NOT
C        ALREADY AT THAT VELCCITY AFTER LAST SEGMENT
C
  204     IF(ABS(EPME-REMEO).LT..01.AND.(ABS(AVSE-V).LT.1.E-5))GO TO 201
 2040     NGOLD=NGEAR                                                      !SAVE GEAR #.
          NGCNT=NGCNT+1                                                    !INC CNT OF PASSES.
          TOLD=T
          T=T+DT
          VOLD=V
          LWCT=.FALSE.
          ENDSEG=.FALSE.                                                   !FLG NOT END DRS SEG.
C
C        ACCELERATE TO DESIRED VELCCITY
C
          ACCEL=(AVSE-V)*1.46667/DT                                        !COMPUTE DESIRED ACCELERATION.
          CALL GOBACK
          IF(MAPOK.LT.4)GC TO 15                                           !(ON THE MAP)YES, DONE, RETURN.
          LOWMAP=(MAPOK.EC.4.OR.MAPOK.EQ.6.OR.MAPOK.EQ.7)
          IF(.NCT.LOWMAP)GC TO 16
          AGRAT=GRAT(NGEAR)
          AEFFG=EFAT(NGEAR)
          ABR=TCECW
          LBRAKE=.TRUE.
```

```
      TORQW=TORQW-(TORQE-TMIN)*AGRAT*AEFFG*AAB
      ABR=TORQW-AER
      TORQP=TORQW/AAB
      TORQ2=TORQP/(AGRAT*AEFFG)
      TORQ1=TORQ2
      TORQE=TORQ1+TORQA+TORQF
      GO TO 15
16    CONTINUE
      RPME=RPMEC
      IF(.NOT.LPDLOK)GO TO 160                                       ![621]
      LOCKUP(NGEAR)=.FALSE.                                          ![621]
      IF((IFIX(RPME).GE.MCLKRE) .AND. (NGEAR.EQ.MDLKGR))
      2        LOCKUP(NGEAR)=.TRUE.                                  ![621]
160   LWOT=.TRUE.
      CALL ENGINE
      LWOT=.FALSE.
      TRR=0.                                                         !ZERO FRONT END ROTATING INERTIA.
      TORQF=0.
      TITER=TWOT                                                     !ASSUME TARGET TORQE TO BE MAX.
      IF(MAPOK.GT.3 .AND. LOWEAF)TITER=TMIN                          !(POINT TO MIN)YES, SET TO MIN.
      CALL ITERAT(TITER)
C
   15 TSAVE=TSAVE+DT
   17 PCTHR=PCTWOT(I)
      DSAVE=DSAVE+DT*V*1.46667
CSIC  GO TO 99
      GO TO 45                                                       !IF NOT SETTLED, BETTER MAKE ENDSEG CHECKS
  201 IF(ABS(RPME-RPMEC).GT..01) GO TO 2040
C
C    COMPUTE END POINT OF SEGMENT IF NOT ON GRADE
C
      IF(ABS(TORQE-TORQEO).GT. .01)GO TO 2040
   21 TEND=1.E20                                                     !YES.
      DD=CUMD*5280.
      IF(.NOT.LSEC) GO TO 22
C
C    IF RELATIVE TIME END POINT GIVEN
C
      TEND=ATSE-TSAVE
   22 IF(.NOT.LRILE) GO TO 24
C
C    IF RELATIVE DISTANCE END POINT GIVEN
C
      TEST=(ADSE-DSAVE)/(V*1.46667)
      IF(TEST.LT.TEND) TEND=TEST
   24 IF(.NOT.LPASS) GO TO 26
C
C    IF PASSING CLEARANCE END POINT GIVEN
C
      IF(ABS(V-VO).LT..00001) GO TO 26
      TEST=(APCS-CUMD*5280.+DSTART+T*VO*1.466667)/((V-VO)*1.466667)
      IF(TEST.LT.TEND) TEND=TEST
   26 IF(.NOT.LEP) GO TO 28
C
C    IF ABSOLUTE MILE POST END POINT GIVEN
C
      TEST=(APOS-DD)/(V*1.46667)
      IF(TEST.LT.TEND) TEND=TEST
C
C    COMPUTE REST OF SEGMENT IN ONE TIME STEP
C
```

```
   28    IF(ITSAV.NE.1)GO TO 280                                               ![606]
         ENDSEG=.TRUE.
         GO TO 99
  280    TOLD=T
         IF(TEND.GE. 1.E-10) GO TO 281                                         !OVERSHOOT?)NO.
C
         CALL SIMLET(6)                                                        ! ? ERROR - OVER SHOT END OF CONSTANT VEL SEG.
         RETURN                                                                !ERROR, BYE.
C
  281    ENDSEG=.TRUE.                                                         !ASSUME END OF DRS SEG.
         IF(ENDRSG.GT.CUMD+V*TEND/3600.) GO TO 29                              !(WILL THIS TAKE US PAST END OF RTE SEG?)NO.
         ENDSEG=.FALSE.                                                        !YES, FLG NOT END OF DRS SEG.
         TEND=(ENDRSG-CUMD)*3600./V                                            !WELL CALC HOW FAR WE CAN GET.
         TSAVE=TSAVE+TEND
         NGCNT=0
   29    T=T+TEND
         ACCEL=0.
         VOLD=V
         LWOT=.FALSE.
         DT=TEND                                                               !SET TIME STEP TO GET US TO END OF DRS/RTE SEG.
         CALL GOBACK                                                           !ZAP - WERE THERE.
         PCTHR=PCTWOT(1)                                                       ! % THROTTLE.
         GO TO 99                                                              !GO UPDATE ACCUMULATORS.
C
   30    ITS=1                                                      !WE'RE ON A HILL AND REACHED CONST VEL SW TO CONST ACCELL SEG.
         ACCEL=0.                                                              !SET TO ZERO ACCEL. THAT = CONST VEL.
         GO TO 45                                                              !GO END SEG CHECKS.
C
C     CONSTANT ACCELERATION SEGMENT
C
   40    LWOT=.FALSE.
         TOFAST=.FALSE.                                                        ![606]
         IF(IMPH.AND.VSTART.GT.AVEL.AND.AASE.GT.0.)TOFAST=.TRUE.               ![606]
         IF(.NOT.TOFAST)GO TO 42                                               ![606]
         ITS=1                                                                 ![606]
         AVSE=AVEL                                                             ![606]
         GO TO 20                                                              ![606]
  42     IF(ITSAV.EQ.2 .AND. ABS(BEER).LE. 1.-3)ITS=2
         TOLD=T
         DT=CPFDT
         T=T+DT
         ACCEL=AASE
         VOLD=V
         PCOLD=PCTHR
         IF(ISTEOP)GO TO 960                                                   ![613]
C
C     TRY REQUIRED ACCELERATION
C
         CALL GOBACK
         PCTHR=PCTWOT(1)
         IF(MAPOK.GT.1) GO TO 428
         IF(ARRIVE) GO TO 45
C
C     CHECK FOR RATE OF THROTTLE CHANGE
C
         PITER=RTHR*DT+PCOLD
         IF(PCTHR.GT.PITER-.01) GO TO 434
         GO TO 45
C
C     IF OFF ENGINE MAP ITERATE TO GET BACK ON
C
```

```
  428   IF(MAPOK.EQ.5.OR.MAPOK.GT.7) GO TO 431
        ARRIVE=.TRUE.
        GO TO 44
  431   PITER=RTHR*DT+PCCID
        IF(100..GT.PITER-.01) GO TO 434
        CALL ITERAT(TWOT)
        GO TO 44
  434   TITER=TMIN+PITER*(TWOT-TMIN)*0.01
        IF(LOCKUP(NGEAR)) GO TO 435
        ACCEL=ACCC+0.01
        CALL GOBACK
  435   CALL ITERAT(TITER)
C
C     BEGIN CHECKS TO SEE IF AT THE END OF A SEGMENT
C
   44   PCTHR=PCTWOT(I)                                                      ! % THROTTLE.
   45   ISEGO=ISEG                                                           ![610]
        DD=CUMD*5280.
        IF(ACCEL.GT.0.) GO TO 451                                            !(+ ACCEL?) YES.
        IF(AASE.LE.0.) GO TO 451
        IF(ABS(EPER).GT..01) GO TO 451                                       !LEVEL GROUND?) NO.
        ACCEL=0.
  452   DT=DT+DEFCT
        CALL GOBACK
        PCTHR=PCTWOT(I)
        IF ( PCTHR.GE.100.1 )  GO TO 452
        IF(.NOT.LIMPRN) CALL SIELPT(5)
  451   IF(V.GE.0.) GO TO 455
C
C     CORRECT SMALL NEGATIVE VELOCITY IF IT OCCURS
C
        DT=-VOLD*1.46667/ACCEL
        CALL GOBACK
        ENDSEG=.TRUE.
        IF(TORQE.LT.TMIN) TORQE=TMIN
        PCTHR=PCTWOT(I)
        GO TO 99                                                             !GO UPDATE ACCUMULATORS.
C
C     DETERMINE IF CAR HAS ARRIVED AT REQUIRED ACCELERATION
C
  455   IF(ABS(ACCEL-AASE).LT..1) ARRIVE=.TRUE.                              !REACHED DESIRED ACCEL?) YES, FLAG IT.
        IF(.NOT.LSEC) GO TO 47
C
C     IF RELATIVE TIME END POINT GIVEN
C
        RT=RT+DT
        IF(ATSE-RT.GT..001) GO TO 47
        ENDSEG=.TRUE.
        GO TO 99
   47   IF(.NOT.LFILE) GO TO 49
C
C     IF RELATIVE DISTANCE END POINT GIVEN
C
        RD=RD+1.466667*VOLD*DT+ACCEL*DT*DT/2.
        IF(ADSE-RD.GT..001) GO TO 49
        ENDSEG=.TRUE.
        GO TO 99
   49   IF(.NOT.LPASS) GO TO 51
C
C     IF PASSING CLEARANCE END POINT GIVEN
C
```

```
      DTC1=DTCT+1.466667*VOLD*D1+ACCEL*DT*DT/2.
      TTO1=TTOT+DT
      IF(APCS-(FTOT-VO*1TOT*1.466667).GT..001) GO TO 51
      ENDSEG=.TRUE.
      GO TO 99
   51 IF(.NOT.LMP) GC TC 53
C
C    IF ABSOLUTE MILE ECST END PCINT GIVEN
C
      RMPD=1.466667*VCLD*DT+ACCEL*D1*DT/2.
      IF ( (AEOS-(DD+RMED)).GT..001 )  GO TO 53
      ENDSEG=.TRUE.
      GO TO 99
   53 IF(.NCT.LMPH) GC TO 99
      IF ((T-TSTART).LT.100.) GC TO 52
      IF (AES(ACCEL).GT.1.E-3) GO TC 52
      IF(LOCKUP(NGEAR).AND.CURTIS.EC.CURT+DT) GO TO 52
C
      CALL SIELET(4)                                            ! ? SIMCTR- FAILURE TO REACH TERMINAL VELOCITY
      RETCRN
C
C    IF TERMINAL VELCCITY END EOINT GIVEN
C
   52 IF((VSTART-AVEL)*(AVEL-V).LT.0.) GO TO 99
      ENDSEG=.TRUE.
   55 IF(.NCT.LLSH) ET=(AVEL-VOLD)*1.466667/ACCEL
      CALL GOEACK
      IF(TORQE.LT.TEIN) TORQE=TEIN
      PCTER=PCTWOT(I)
      IF (PCTER.LT.100.1) GO TO 99
      ACCEL=ACCEL-.001
      GO TO 55
C
C    CONSTANT PERCENT WOT SEGMENT
C
   60 DT=DEFDT
      VOLD=V
      PCCID=PCTHR
      TCLD=T
      T=T+DEFDT
      LWCT=.FALSE.
      TITER=TEIN+APWOC*(TWOT-THIN)                              I[603]
      IF(LSTROP)GO TC 860                                       I[613]
      CALL ITERAT(TITER)                                        I[603]
      PCTHR=PCTWOT(I)                                           I[603]FOR PRINT-OUT
      GO TO 45                                                  I[603]
      IF(ABS(ACCEL).LT. 1.E-5) ACCEL=5.0
C
C    SEE IF ALREADY AT REQUIRED WOT SETTING
C
      CALL GCEACK
      IF(.NCT. LOCKUP(NGEAR)) GC TO 621
      FTORQ=TMIN+APWOC*(TWOT-TMIN)
      DIF=PTORQ-TORCE
      IF(DIF*DIF.LE.1.E-5) GO TO 45
C
C    IF NOT AT REQUIRED SETTING, ITERATE TO GET THERE
C
      DIFC=DIF
      DACC=ACCEL * DIF /PTCHC
      IF(ACCEL.GT.0..AND.(DACC.LT.-ACCEL.AND.DPER.LE.0.) DACC=-ACCEL+.1
   62   ACCEL=ACCEL+DACC
        CALL GOBACK
        PTORQ=TMIN + APWOC * (TWOT-TMIN)
        DIF=PTORQ-TORQE
        IF(DIF*DIF.LE. 1.E-5) GO TO 63
        DACC=ACCEL *  DIF /PTORQ
        IF(ABS(DACC).LT.1.E-3) GO TO 63
        GO TO 62
  621   PCTHR=PCTWOT(X)
        DIF=APWO-PCTHR
        IF(DIF*DIF .LE. 0.1) GO TO 45
        DIFO=DIF
        DACC=1.
        IF(DIF.LT.0.) DACC=-DACC
  622   ACCEL=ACCEL+DACC
        CALL GOBACK
        PCTHR=PCTWOT(X)
        DIF=APWO-PCTHR
        IF(DIF*DIF .LE. .01) GO TO 63
        IF(DIF*DIFO .GT. 0.) GO TO 622
        DACC=-DACC*.1
        IF(ABS(DACC) .LT. 1.E-3) GO TO 63
        DIFO=DIF
        GO TO 622
C
C     CHECK FOR RATE OF CHANGE OF THROTTLE AND ITERATE IF NECESSARY
C
   63   PCTHR=100.0 *( TORQE-TMIN )/( TWOT-TMIN)
        IF((PCTHR-PCOLD)/DT.LT.RTHR) GO TO 45
        IF(.NOT.LOCRUE(NGEAR) .AND. NGEAR.EQ.1) GO TO 64
        PCR=(PCOLD+RTHR*DT)*0.01
  631   PTORQ=TMIN+PCR*(TWOT-TMIN)
        DIF=PTORQ-TORQE
        IF(DIF*DIF .LE. 1.E-5) GO TO 45
        DACC=ACCEL *  DIF /PTORQ
        IF(ABS(DACC) .LE. 1.E-3) GO TO 45
        ACCEL=ACCEL+DACC
        IF(ACCEL.LT. 0.0) ACCEL=.01
        CALL GOBACK
        PCTHR=PCTWOT(X)
        GO TO 631
   64   DACC=-.1
  641   ACCEL=ACCEL+DACC
        CALL GOBACK
        PCTHR=PCTWOT(X)
        IF((PCTHR-PCOLD)/DT .LT. RTHR) GO TO 45
        GO TO 641
C
C     ACCELERATE TO REQUIRED ACCELERATION AND HOLD CONSTANT
C     THROTTLE FROM THEN ON SEGMENT
C
   80   DT=DELDT
        VOLD=V
        TOLD=T
        T=T+DT
        ACCEL=SATR
        PCOLD=PCTHR
        LWOT=.FALSE.
        IF(LSTRUP)GO TO 860                                  1[613]
C
C     SEE IF POSSIBLE TO DO IT ON THE MAPS
```

```
      CALL GOBACK
      PCTHR=PCTWOT(X)
      IF(MAPOK.GT.3) GO TO 809
C
C     CHECK FOR RATE OF CHANGE OF THROTTLE
C
      IF(ARRIVE) GO TO 851
      IF((PCTHR-PCOLD)/DT.GT.RTHR) GO TO 848
      GO TO 851
 809  IF(.NOT.(MAPOK.EQ.5.OR.MAPOK.GT.7)) GO TO 85
C
C     ITERATE TO GET BACK ON MAP IF TORQUE TOO HIGH
C
  82  DACC=-1.
 821  MAPOLD=MAPOK
  83  ACCEL=ACCEL+DACC
      CALL GOBACK
      PCTHR=PCTWOT(X)
      IF(MAPOK.EQ.MAPOLD) GO TO 83
      IF(MAPOK.GT.3.AND.MAPOLD.GT.3) GO TO 83
      DACC=-DACC*.1
      IF(DACC*DACC.LT.1.E-6.AND.MAPOK.LT.4) GO TO 848
      GO TO 821
  85  PCTHR=PCTWOT(X)
 848  DACC=-1.
C
C     CHECK FOR RATE OF CHANGE OF THROTTLE AND ITERATE IF NECESSARY
C
 849  IF(V.LT.0.) GO TO 8511
      IF((PCTHR-PCOLD)/DT.LT.RTHR) GO TO 852
      IF(PCTHR.LT.0.) GO TO 852
8459  ACCEL=ACCEL+DACC
      IF(ACCEL.LT.0.) GOTO 8512
      CALL GOBACK
      PCTHR=PCTWOT(X)
      GO TO 849
8511  DACC=-DACC
      GO TO 853
8512  ACCEL=ACCEL-DACC
      DACC=DACC*.1
      IF(DACC*DACC.LT.1.E-6) GO TO 851
      GO TO 8459
 852  DACC=-.1*DACC
      IF(DACC*DACC.LT.1.E-6) GO TO 851
C
C     ITERATE FOR THROTTLE RATE OF CHANGE
C
 853  ACCEL=ACCEL+DACC
      CALL GOBACK
      PCTHR=PCTWOT(X)
      IF((PCTHR-PCOLD)/DT.GT.RTHR) GO TO 854
      GO TO 853
 854  DACC=-.1*DACC
      IF(DACC*DACC.LT.1.E-6) GO TO 951
      GO TO 8499
C
C     SEE IF REACHED REQUIRED ACCELERATION
C
 851  IF(ACCEL.GT.0.) GO TO 855
      IF(SATH.LE.0.) GO TO 855
      IF(ABS(EPER).GT.0.01) GO TO 855
      ACCEL=0.
  856 DT=DT+DELDT
      CALL GOBACK
      PCTHB=PCTHOT(X)
      IF(PCTHB.LT.100.1) GO TO 855
      GO TO 856
  855 IF(SATH-ACCEL.GT.0.) GO TO 45
C
C    CONTINUE THE SEGMENT AS IF IT WERE A CONST PERCENT HOT SEGMENT
C
      ARRIVE=.TRUE.
      ITS=3
      APHC=PCTHB
      GO TO 45
C
C    COMPUTATIONS PERFORMED AT END OF EACH TIME STEP
C
   59 D=1.466667*VOLD*DT+ACCEL*DT*DT/2.
      ACCC=ACCEL
C
C...  UPDATE ACCUMULATORS
C
      IF(V.LT.0.) V=0.
      DHR=DT/3600.
      DDIS=D/5280.
      DHRD=DHR*.5
      IF((LDIES) .AND. (ACCEL.LT.0.) .AND.
     2      (V.GT.1.E-4)) FRATE=0.0          ![615]
      DFU=(FRATE+FRATEO)*DHRD
      IF(CUMD.GE.ENDRTE.AND.LSTRTE) ENDSEG=.TRUE.
      CUMD=CUMD+DDIS
      CUMT=CUMT+DT
      NGCCAL=0                                               !ZERO CNT OF CALLS TO GOBACK.
      VAVG=3600.*CUMD/CUMT
C
C    FUEL ECONOMY COMPUTATIONS
C
      CUMFU=CUMFU+DFU
      HPE=TORQE*RPME*EB2
      BSFC=99.99
      IF(HPE.GT.0.) BSFC=FRATE/HPE
      FRATG=FRATE*AA10
      FEINST=999.99
      IF(FRATG.NE.0)FEINST=V/FRATG
      CUMG=CUMG+(FRATG+FRATGO)*DHRD
      RDLD=(FWHEEL-(FACCEL))*V/375.
      GALPR=0.
      IF(CUMT.GT.0.) GALHR=3600.*CUMG/CUMT
      CUMFE=CUMD/CUMG
      MILE=CUMD
      FT=(CUMD-FLOAT(MILE))*5280.
      MIN=CUMT/60.
      SEC=(CUMT-FLOAT(MIN)*60.)
C
C    HORSEPOWER AND EFFICIENCY COMPUTATIONS
C
      HP2=TORQ2*(RPM2*EB2)
      HPCL=TORQ2*(RPMC*EB2)                                  ![607]
      HPE=TORQP*RPME*EB2
      HPW=TORQW*RPMW*EB2
```

```
         ALOSSF=ABS(HPE-HPW)
         ALOSSG=ABS(HP2-HPP)
         ALOSSL=ABS(HP2-HPCL)
C        IF(PRINT6)WRITE(6,9001) RPM2,REMC,HP2,HPCL,ALOSSL
C9001     FORMAT(' $SIMCTR-RPM2,REMC,HP2,HPCL,ALOSSL->',5G)
C        IF(PRINT6)WRITE(6,9002) RPM2,RPMC,HP2,HPCL,ALOSSL
C9002     FORMAT(5G)
C        IF(PRINT6)WRITE(6,9003) TORQ2,TORQ2,BB2,BB2
C9003     FORMAT('$SIMCTR-TORQ2,BB2->',G,O,G,O)
         HP1=TORQ1*RPM1*BB2
         HPA=TORQA*RPME*BB2
         DEN=(HPE+HPEO)*CHRD
         IF(DEN.LT.0.)  CUMENR=CUMENR+DEN
         IF(DEN.GE.0.)  CUMEN=CUMEN+DEN
         EFFC=TR*SR
         IF(COAST.AND.EFFC.GT.1.) EFFC=1./EFFC
         ALOSSC=ABS(HP2-HP1)
         IF ( LOCKUP(NGEAR) .AND. SHFTNG )  ALOSSC = ABS(HP2)
         IF(ABS(ACCEL).GT.1.E-5) GO TO 302
         IF(V.GT.1.E-5) GO TO 301
C
C      UPDATE ACCUMULATORS ACCORDING TO TYPE OF DRIVING BEING DONE
C
         CTI=CTI+DT
C
C      IDLE STEP
C
         CFI=CFI+DFU
         IF(DEN.LE.0.) GO TO 310
         CEI=CEI+DEN
         GO TO 310
C
C      CRUISE STEP
C
  301    CTCR=CTCR+DT
         CDCR=CDCR+DDIS
         CECR=CECR+DEN
         CFCR=CFCR+DFU
         GO TO 310
C
C      DECELERATION STEP
C
  302    IF(ACCEL.GT.0.) GO TO 303
         CTD=CTD+DT
         CDD=CDD+DDIS
         IF(DEN.LT.0.) CEDM=CEDM+DEN
         IF(DEN.GE.0.) CED=CED+DEN
         CFD=CFD+DFU
         GO TO 310
C
C      ACCELERATION STEP
C
  303    CTA=CTA+DT
         CDA=CDA+DDIS
         CEA=CEA+DEN
         CFA=CFA+DFU
C
C      COMPUTE ENERGY AND COMPONENT LOSSES
C
  310    EDUM=WRAD*BB2*CHRD
         EP1=EDUM*RPMW
```

```
      FF1C=FDOF*RPMWO
      CAC=CAC+(HPA+HPAC)*DHRD
      CTR=CTR+(ALCSSC+ALOSCC)*DHRD
      CRC=CFO+FROLL*FF1+FROLLC*FF10
      CPE=CPE+EGRADE*FF1+EGRADO*FF10
      CGB=CGB+(ALCSSG+ALOSGO)*DHRD
      CCL=CCI+(ALCSSI+ALCSLC)*DHRD                                  ![607]
      CDIF=CDIF+(ALCSSR+ALOSRC)*DHRD
      DTIRE=ABS(HPW*TTT1)
      CTIRE=CTIRE+(CTIRE+DTIREO)*DHRD
      CBR=CBR+(ABR*RPEW+ABRC*RPEWO)*EB2*DHRD
      CAE=CAE+EAERO*EF1+PAEROC*EF10
      TEMTOT=CAC+CTR+CRC+CGB+CDIE+CUMENM+CBR+CAE+CCL
      DCKE=5.05E-7*(WGT/(2.*32.17))*((V*1.46667)**
     2        2-(VOLD*1.46667)**2)
      DCRCT=.5*( 4.*AIW + RARSQ*(AIP+GRAT(NGEARO)*GRAT(NGEARO)*AI2
     1             +AIGIN(NGEARO)+AIGOUT(NGEARO)) ) * (BB1*RPMWO)**2
     2           - .5 * ( 4.*AIW + RARSQ*(AIP+GRAT(NGEAR)*GRAT(NGEAR)*AI2
     3             +AIGIN(NGEAR)+AIGOUT(NGEAR)) ) * (BB1*RPMW )**2
     4           + .5 * (EINER+AIA+AI1) * (BB1*RPMEO)**2
     5           - .5 * (EINER+AIA+AI1) * (BB1*RPME )**2
      CRCT=CRCT+DCRCT
      CKE=CKE+DCKE
      IF(.NOT.PRINT6)GO TO 700
      OTHER=CPE+DCKE-DCTOT*5.05E-7
      TOTEN=CUMEN
      IF( OTHER.LT.0.) TOTEN=CUMEN-OTHER
      IF(OTHER.GT.0.)TEMTOT=TEMTOT+OTHER
      WRITE(6,9000)CKE,CROT,CTO,CGB,CCL,
     2CDIF,CTIRE,CBR,CAE,TEMTOT,TOTEN
9000  FORMAT(1X,10E13.7)
C
C     UPDATE GEAR TIME DISTRIBUTION ARRAY
C
700   GTIMAC(NGEAR)=GTIMAC(NGEAR)+DT
C
C
C     DO HISTOGRAM ACCUMULATIONS   [614]
C
      XLOW=FIRREM
      XHIGH=XLOW+DELREM
      DO 7020 IHR=1,20
      IF(RPME.GE.XLOW .AND. RPME.LT.XHIGH)GO TO 7040
      XLOW=XHIGH
      XHIGH=XLOW+DELREM
7020  CONTINUE
      IHR=20
C
7040  XLOW=FIRTQR
      XHIGH=XLOW+DELTQR
      DO 7060 IHT=1,20
      IF(TORQE.GE.XLOW .AND. TORQE.LT.XHIGH)GO TO 7080
      XLOW=XHIGH
      XHIGH=XLOW+DELTQR
7060  CONTINUE
      IHT=20
C
7080  HIST(IHT,IHR,1)=HIST(IHT,IHR,1)+TORQE*DT
      HIST(IHT,IHR,2)=HIST(IHT,IHR,2)+RPME*DT
      HIST(IHT,IHR,3)=HIST(IHT,IHR,3)+DT
C
```

```
      IF(IBRAKE) CFE=CFE+DFU
      IF(DEN.LT.0) CENG=CENG-DEN*DHR
      IF(PCTHR.LT.0.) PCTHR=0.
      HPAC=HPA
      HPRO=HPR
      HP1C=HP1
      HP2C=HP2
      HPFC=HPF
      HPWC=HPW
      FRATEC=FRATE
      FRATGO=FRATG
      DTIREO=DTIRE
      FROILC=FROIL
      FGRADC=FGRADE
      FWHEEC=FWHEEL
      FAEROO=FAERO
      ALCSCC=ALCSSC
      ALCSGC=ALCSSG
      ALCSRC=ALCSSR
      ALCSLC=ALCSSL
      RPM2O=RPM2                                                 !(607)
      RPMWO=RPMW
      RPMEO=RPME
      TORQAC=TORQA                                               !SAVE ACC TORQ.
      TORQEC=TORQE
      TMINO=TMIN                                                 !SAVE MIN ENG TORQ.
      IF(ISTRUP.AND.IDEBUG.EQ.3)CALL DEBUG('START')              !TEMP PRINT
      IF(ISTRUP)GO TO 667                                        !(613)
      IF(SHFTNG.AND.LOCKUP(NGEAR)) GO TO 915                     !(MANUAL SHIFTING IN PROGRESS?) YES.
C
C     DETERMINE IF LIMIT PRINT SPECIFIED
C
      IF(.NOT.LIMPRD) GO TO 715
      IF(MILIM) GO TO 705
      IF(SECLIM) GO TO 710
      GO TO 998
C
C     PRINT EVERY *ALIMN* MILES
C
  705 PRNDIS=PRNDIS+DDIS
      IF (ALIMN-PRNDIS.GT..0001) GO TO 998
      PRNDIS=0.
      GO TO 715
C
C     PRINT EVERY *ALIMN* SECONDS
C
  710 PRNTIM=PRNTIM+DT
      IF ( (ALIMN-PRNTIM).GT..001 )  GO TO 998
      PRNTIM=0.
C
C     SET BRAKE FLAG
C
  715 IF(.NOT.((ITS.GT.1).OR.(ACCEL.GE.0.).OR.(V.GT.1.E-4)))
     1 AER=ABRC
C
      CALL SIPLET(5)                                             !PRINT LINE FOR THIS TIME STEP IF REQUIRED
C
      IF(.NOT.ENDSEG) GO TO 999                                  !(END DRS SEG?) NO.
      KEND=1                                                     !FLG DATA LINE FOR LST TIME STEP TO LPT.
      GO TO 110
C
```

```
  958   IF(ENDSEG) GO TO 110                                                 !(END DRS SEG?) YES.
C
C     CHECK FOR SHIFT POSSIBILITY
C
  959   NGOLD=NGEAR                                                          !SAVE GEARS 0.
        IF(.NOT.LLSH) GO TO 988


        DT=ETO
        IF(LOCKUP(NGEAR)) ACCEL=0.
        LLSH=.FALSE.
         CLUTCH=.FALSE.                                                        ![607]
        GO TO 102
  988   IF((CUMT-CUMTLS.LT.SDELAY.OR.LSHFT).AND.
        2        ISEG.EQ.ISEGO) GO TO 102                                    ![610](SHIFT POSSIBLE?) NO.
        CALL SHIFTS                                                          !YES, TRY IT.
        IF(.NOT.LLSH) GO TO 102                                              !(DID WE?) NO.
C
C     COUNT UPSHIFTS AND DOWNSHIFTS INTO GEAR
C
        ACCEL=-(FAERO+FRCIL+FGRADE)/AA4
        IF ( NGOLD.LT.NGEAR )   GO TO 740                                    !(WAS IT DOWNSHIFT?) NO.
        IF(EPER.LE.0.)GO TO 730
        VNEW=VOLD+ACCEL*DT/1.46667
        IF(.NOT.PARAB) GO TO 722                                             ![626]
        VELTAR=SHFTRP(2,(NGOLD-NGEAR-1)+NUMG)                                ![626]
        GO TO 724                                                            ![626]
 722    CONTINUE                                                             ![626]
        IF(LENG) VELTAR=SHFTRP(2,NGEAR)/(GRAT(NGEAR)*RAR*AA5)               ![626]
        IF(.NOT.LENG) VELTAR=SHFTRP(2,NGEAR)/(RAR*AA5)                       ![626]
 724    CONTINUE                                                             ![626]
        IF(.NOT.(VNEW.GT.VELTAR))GO TO 730
 725    NGEAR=NGOLD
        LLSH=.FALSE.
        DT=.05
        ACCEL=ACCC
        CALL GOBACK
        GO TO 102
 730    NSGEAR(NGEAR,2) = NSGEAR(NGEAR,2) + 1
        LDNSHF=.TRUE.
        GO TO 750
  740   IF(EPER.LE. 0.)GO TO 745
        NGN=(NUPG-NGEAR)+NUMG
        RPMES=RPME
        RPME=SHFTRP(1,NGN)
        TORQA=0.
        IF(NACC.GT.0)TORQA=(ENTERP(ACCT(1,1),ACCS(1,1),NNA(1)
        2,NACC,RPME,N20))*DUTCYC
        CALL ENGINE
        RPME=RPMES
        DT=(EINFR+AIA)*(RPME-SHFTRP(1,NGN))*BB1/(TORQA-TMIN)
        VNEW=VOLD+ACCEL*DT/1.46667
        IF(.NOT.PARAB) GO TO 742                                             ![626]
        VELTAR=SHFTRP(1,NGN)                                                 ![626]
        GO TO 744                                                            ![626]
 742    CONTINUE                                                             ![626]
        IF(LENG) VELTAR=SHFTRP(1,NGN)/(GRAT(NGEAR)*RAR*AA5)                 ![626]
        IF(.NOT.LENG) VELTAR=SHFTRP(1,NGN)/(RAR*AA5)                         ![626]
 744    CONTINUE                                                             ![626]
        IF(.NOT.(VNEW.LT.VELTAR))GO TO 745
        GO TO 725
  745     NSGEAR(NGEAR,1) = NSGEAR(NGEAR,1) + 1
          LDNSHF=.FALSE.
          ABR=0.                                                          ![620]
  750     NSFSEG(ISEG)=NSFSEG(ISEG)+1
          DRPME=0.                                                        ![607]
          LCLTCH=.TRUE.                                                   ![607]
          SHFTNG=.TRUE.                                                   !FLAG SHIFT IN PROGRESS.
          CALL GOBACK                                                     ![607]
          DTC=(3.*ABS(RPM2-RPMC)/RPMAX(IENG))**(1./3.)
     2             *(.4+.7*(1.-1./EXP(ABS(TORQ2/TWOT))))                  ![607]
          DRPMC=ABS(RPM2-RPMC)/DTC                                        ![607]
          IF(.NOT.LDNSHF) DRPMC=-DRPMC                                    ![607]
C
          DTO=DT
          ARRIVE=.FALSE.
          LWOT=.FALSE.
          CSTIM=0.0                                                       !INTI TIME ACUM OF SHIFT.
          IF(LOCKUP(NGEAR)) GO TO 900                                     !(GEAR LOCKED UP?) YES.
C
C       PERFORM SHIFT FOR UNLOCKED GEARS.
C
          TSAVE=TSAVE+STIME
          DSAVE=DSAVE+STIME*V*1.46667
          DT=STIME
          TOLD=T
          T=T+DT
          VOLD=V
          CSTIM=STIME
          HPCTHR=PCTHR
          IF(TORQE.GE.(TMIN-.001)) GO TO 982
          CALL ITERAT(TMIN)
C
  982     CALL GOBACK
          IF(.NOT.SHFTNG) GO TO 975                                       ![SID]
          IF(ISMODE.EQ.2) GO TO 997
C
C       PERFORM CONSTANT THROTTLE SHIFT
C
          IF(ITS.EQ.1 .AND. RASE.GE.0. .AND. LDNSHF .AND.
     2          ACCEL.LT.0.)CALL ITERAT(TWOT)                             ![601]
          CALL GOBACK                                                     ![601]
          PCTHR=PCTWOT(X)
          DIF=HPCTHR-PCTHR
          DACC=1.
          IF(DIF.LT.0.) DACC=-DACC
 9950     ACCEL=ACCEL+DACC
          CALL GOBACK
          PCTHR=PCTWOT(X)
          DIFO=DIF
          DIF=HPCTHR-PCTHR
          IF(DIF*DIF.LE. 0.01) GO TO 992
          IF((DIF*DIFO.GT.0.).AND.(ABS(DIF).LT.ABS(DIFO))) GO TO 9950
          DACC=-DACC*.1
          IF(ABS(DACC) .GT. 1.E-3) GO TO 9950
          GO TO 992
C
C       PERFORM CONSTANT ACCELERATION SHIFT
C
  997     PCTHR  = 100. * (TORQE-TMIN) / (TWOT-TMIN)
          IF ( PCTHR.LT.0.    )  GO TO 995
          IF ( PCTHR.LE.100.49999)  GO TO  45
```

```
      DACC    = - 1.
  993 MAPOLD = MAPOK
  991 ACCEL   = ACCEL + DACC
      CALL GOBACK
      IF ( MAPOLD.EQ.MAPOK )  GO TO 991
      IF ( MAPOK.GT.3 .AND. MAPOLD.GT.3 )   GO TO 991
      DACC    = - DACC * .1
      IF ( DACC*DACC.LT.1.E-6 .AND. MAPOK.LT.4 )   GO TO 1011
      GO TO 993
  995 TORQE   = TORQ
 1011 PCTHR   = 100. * (TORQE-TMIN) / (TWOT-TMIN)
      GO TO 992
C
C   PERFORM COASTING SHIFT FOR LOCKED UP GEARS.
C
  900 IF(.NOT.LDNSHF) GO TO 910                                   !DOWN SHIFTING?) NO.
      ACCEL=-(FAERO+FROLL+FGRADE)/AA4                             !LINEAR GUESS OF ACCEL DURING COAST.
      VNEW=VOLD+ACCEL*DT/1.46667                                  !COMPUTE NEW VELOCITY IF THIS ACCEL IS USED.
      IF(VNEW.LT.0.)DT=1.46667*(.01-VOLD)/ACCEL                   !(NEG VELOCITY) YES
      IF(SHFTRP(3,NGEAR).NE.0.)GO TO 910                          ![604]
      IF(.NOT.PARAB) GO TO 906                                    ![626]
      VELTAB=SHFTRP(2,NGEAR+1)                                    ![626]
      GO TO 908                                                   ![626]
906   CONTINUE                                                    ![626]
      IF(LENG) VELTAB=SHFTRP(2,NGEAR)/(GRAT(NGEAR)*RAR*AA5)       ![626]
      IF(.NOT.LENG) VELTAB=SHFTRP(2,NGEAR)/(RAR*AA5)              ![626]
908   CONTINUE                                                    ![626]
      IF(VNEW.GT.VELTAB)DT=1.46667*(VELTAB+(.1E-4)-VOLD)/ACCEL    !(IS IT ABOVE UPSHFT SPEED) YES
  910 LCLTCH=.TRUE.                                               !FLAG ENGINE SEPERATED FROM POWER TRAIN.
      DT=.05                                                      !SET TIME STEP DURING SHIFT.
      GO TO 917
C
  915 IF(CSTIM.GT.STIME) GO TO 977                                !(ALLOWED SHIFT TIME EXCEEDED?) YES, ERR.
      CSTIM=CSTIM+DT                                              !NO, INC TIME SPENT SHIFTING ACUM.
  917 TSAVE=TSAVE+DT
      DSAVE=DSAVE+DT*V*1.46667
      TOLD=T
      T=T+DT
      VOLD=V
      ACCEL=-(FAERO+FROLL+FGRADE)/AA4                             !CALC ACCEL OF COASTING VEHICLE DURING CURRENT DT.
      IF(LDNSHF)GO TO 920
      VNEW=VOLD+ACCEL*DT/1.46667                                  !COMPUTE NEW VELOCITY IF THIS ACCEL IS USED.
      IF(VNEW.LT.0.)DT=1.46667*(.01-VOLD)/ACCEL                   !(NEG VELOCITY) YES
      IF(.NOT.PARAB) GO TO 918                                    ![626]
      VELTAB=SHFTRP(1,(NUMG-NGEAR)+NUMG)                          ![626]
      GO TO 919                                                   ![626]
918   CONTINUE                                                    ![626]
      IF(LENG)VELTAB=SHFTRP(1,(NUMG-NGEAR)+NUMG)/(GRAT(NGEAR)*RAR*AA5)    ![626]
      IF(.NOT.LENG) VELTAB=SHFTRP(1,(NUMG-NGEAR)+NUMG)/(RAR*AA5)  ![626]
919   CONTINUE                                                    ![626]
      IF(VNEW.LT.VELTAB)DT=1.46667*(VELTAB-(.1E-4)-VOLD)/ACCEL    !(IS IT BELOW DNSHFT SPEED) YES.
      IF(DT.LT..01)DT=.05
920   CALL GOBACK
      IF(SHFTNG)GO TO 99                                          ![607]
C
  975 LCLTCH=.FALSE.                                              !POP, CLUTCH OUT,
CSID  ACCEL=0.                                                    !ZERO ACCEL.
      IF(.NOT.LDNSHF) GO TO 992                                   !(DOWN SHIFTING?) NO.
      ATORQF=0.                                                   !FOR LOCKS ONLY.
      ARPPE=0.                                                    !FOR LOOKS ONLY.
C
```

```
  992     CUMTLS=CUMT+DT                                        !SAVE SIM TIME AT END OF SHIFT.
          SHFTNG=.FALSE.                                        !FLG NO LONGER SHIFTING.
          LCLTCH=.FALSE.
          IF ( IDEBUG.NE.2 )  GO TO 45                          !(DEBUG SHIFTS?)NO.
          CUMTO  = CUMT                                         !SAVE SIM TIME.
          CUMT=CUMTLS                                       !FOR DEBUG CALC CUMT AS THIS NORMALLY DONE DURING ACCUM UPDATE.
          CALL DEBUG  ( HAFTER )                                !DEBUG OUTPUT AFTER SHIFT.
          CUMT   = CUMTC                                        !RESTORE CUMT.
          GO TO 45                                              !GO END OF SEGMENT CHECKS.
C
  977     WRITE(5,1977)
 1977     FORMAT(/' ? SIMCTR - SHIFT TIME EXCEEDED ALLOWED TIME.'/)
          CALL CTRLC('SIMER')                                   !DEBUG TO TTY.
          RETURN
C
C         STARTUP CODE [613]
C
960       ACCEL=.1                                              !GIVE SOME SMALL ACCEL TO GET MOVING.
          VOLD=V
          DT=DEEDT
          LWCT=.TRUE.
          TWOTO=TWOT
          TMINA=TMIN
          CALL GOBACK
 861      IF(TWOT.GE.TORQE*1.1)GO TC 865                        !(GOT ENOUGH TORQUE TO MOVE?)YES
          RPME=RPME+10.                                         !NO, INC RPM AND TRY AGAIN
          IF(.NOT.LPDLOK)GO TO 8611                             ![621]
          LOCKUP(NGEAR)=.FALSE.                                 ![621]
          IF((IFIX(RPME).GE.MCLKRE) .AND. (NGEAR.EQ.NDLKGR))
     2          LOCKUP(NGEAR)=.TRUE.                            ![621]
8611      CALL ENGINE
          IF(RPME.LE.RPMAX(IENG))GO TO 861
          CALL SIMLET(13)                                       !CAN'T MOVE)C
          RETURN
C
865       NDRE=IFIX((RPME-RPMIN(IENG))/100.)+1
          IDRP=0
          ACCEL=0.
867       IDRP=IDRP+1
          T=T+DT
          RPME=RPMIN(IENG)+100*(IDRE-1)
          IF(.NOT.LPDLOK)GO TO 8670                             ![621]
          LOCKUP(NGEAR)=.FALSE.                                 ![621]
          IF((IFIX(RPME).GE.MCLKRE) .AND. (NGEAR.EQ.NDLKGR))
     2          LOCKUP(NGEAR)=.TRUE.                            ![621]
8670      CALL ENGINE
          IF(IDRP.NE.NDRE)GC TO 95
          RPM1=RPME
          RPM2=SR*RPM1
          LSTRUP=.FALSE.
          TWOT=TWOTO
          TMIN=TMINA
          GO TO (40,20,60,80),ITS
          CALL SIMLET(14)
          RETURN
C
C        CHECK FOR END OF ROUTE OR ROUTE SEGMENT
C
  102     ISEGO=ISEG                                            ![610]RESET OLD SEG CNTR.
          IF(CUMD.LT.ENDRSG) GO TC 105                          !(END RTE SEG?) NO.
          NRTSEG=NRTSEG+1                                       !YES, INC RTE SEG PTR
```

```
        IF(NRTSEG+NCRTE.LE.NRDIST) GO TO 104                            !(END OF RTE SECT?) NO.
        IF(LSTRTE) GO TO 1101                                           !YES, (LAST RTE SECT?) YES.
  103   IPRNT=208                                                       !NO, SET DSK FLG TO NXT RTE SECT.
        CALL DSK                                                        !GET NXT SECT.
        IF(IPRNT.NE.8) RETURN                                           !(DSK ERR?) YES.
        CALL DSKCTR(0,'SIMCTR')                                         !RELEASE DB FILES.
        NRTSEG=1                                                        !SET SEG PTR.
        ENDRTE=RDIST(NRDIST-NDRTE)                                      !LOOKUP MILEPOST END OF RTE SECT.
C
C     BEGIN NEXT ROUTE SEGMENT
C
  104   BPERO=BPER
        DPER=RGRADE(NRTSEG)*.01                                         !GET NEW ROUTE GRADE.
        ENDRSG=RDIST(NRTSEG)                                            !GET NEW END ROUTE MILE POST.
        ROADC=RCOEF(NRTSEG)                                             !GET NEW ROAD COEF.
        VSWIND=RVWIND(NRTSEG)                                           !ASSUME USE RTE SEG WIND SPEED.
        IF(VSWIND.GT.300.) VSWIND=VWIND                         !(RUN TIME VAL FLAGED?) YES, SET TO VALUE GIVEN BY /*MODIFY/ CMD.
        VWINDS=VSWIND*SIN(PHI)                                          !CALC SIDE WIND COMPONENT.
        CPHI=COS(PHI)                                                   !CALC COS OF WIND ANGLE.
        VWINDC=VSWIND*CPHI                                              !CALC HEAD WIND COMPONENT.
        IF(.NOT.((ITYSEG(ISEG).EQ.2).AND.(BPER.NE.BPERO)
     1 .AND.(ABS(V-AVSP).GE.0.1))) GO TO 105
        ITS=2
        SDELAY=.8
C       SDELAY=SHFTIM(1)
        IF(EPER.GT.0.)SDELAY=SDELAY*EPER
        NGCNT=0
  105   IF(ENDSEG) GO TO 112                                            !(END OF DRSSEG?) YES.
        IF(.NOT.LTPSPD)GO TO 106                                        ![623]
        IF((ABS(V-VCLD).LT.1.E-3) .AND. (ACCEL.LT.1.E-3))GO TO 111      ![623]
C
C     RESET BRAKE FLAG FOR NEXT TIME STEP
C
106     ABRCC=ABRC                                                      !SAVE OLD BRAKE.
        ABRC=ABR                                                        !SAVE BRAKE.
        ABR=0.                                                          !ZERO BRAKE.
        LBRAKE=.FALSE.                                                  !BRAKES OFF.
        GO TO 10                                                        !NEXT TIME STEP PLEASE.
C
  110   IF(CUMD.LT.ENDRTE) GO TO 112                                    !(END OF RTE SECT?) NO.
        IF(.NOT.LSTRTE) GO TO 103                                       !(LAST RTE SECT?) NO.
 1101   CALL SIMSTS("36)                                                !YES, REPORT IT.
        GO TO 1111
C
C     COMPUTE FINAL ACCUM PERCENTS OF TOTAL AVAIL ENERGY AND OTHER TOTAL
C
  111   CALL SIMSTS("35)                                                !REPORT END OF DRS SCHED.
 1111   IF((.NOT.SECLIM.AND..NOT.RILIM) .OR.
        2       (KEND.GT.0) .OR. LTPSPD) GO TO 120                      ![623]
        LIDLE=.FALSE.                                                   ![607]
        IF((ITS.GT.1).OR.(ACCEL.GE.0.).OR.(V.GT.1.E-4)) GO TO 122
        ABR=ABRC
        IF(ABR.LT.ABRCO-10.) ABR=ABROO
C
  122   CALL SIMPLT(5)
        LIDLE=.TRUE.                                                    ![607]
C
C     IDLE CONDITION
C
  120   IC=-1
        CTI=100.*CTI/CUMT
```

```
      CFI=100.*CFI/CUMFU
      CEI=100.*CEI/CUMEN
C
C     ACCELERATION CONDITION
C
      CTA=100.*CTA/CUMT
      CEA=100.*CEA/CUMEN
      CFA=100.*CFA/CUMFU
C
C     DECELERATION CONDITION
C
      CTD=100.*CTD/CUMT
      CED=100.*CED/CUMEN
      CFD=100.*CFD/CUMFU
      CDA=100.*CDA/CUMD
      CDD=100.*CDD/CUMD
C
C     CRUISE CONDITION
C
      CTCR=100.*CTCR/CUMT
      CDCR=100.*CDCR/CUMD
      CECR=100.*CECR/CUMEN
      CFCR=100.*CFCR/CUMFU
      CFB=100.*CFB/CUMFU
C
C     LOSS ACCUMULATORS
C
      CKE=5.05E-7*(WG1/(2.*32.17))*((V*1.46667)**2-(V0*1.46667)**2)
      CROT    =   .5 * ( 4.*AIB + RARSQ*(AIP+GRAT(NGO  )*GRAT(NGO  )*AI2
     1              +AIGIN(NGO  )+AIGCUT(NGO  )) ) * (DB1*RPMWI)**2
     2            - .5 * ( 4.*AIW + RARSQ*(AIP+GRAT(NGEAR)*GRAT(NGEAR)*AI2
     3              +AIGIN(NGEAR)+AIGCUT(NGEAR)) ) * (BB1*RPMW )**2
     4            + .5 * (EINER+AIA+AI1) * (EB1*RPMEI)**2
     5            - .5 * (EINER+AIA+AI1) * (DB1*RPME )**2
      CROT=-CROT*5.05E-7
      OTHER=CEE+CKE+CECT
      TOTEN=CUMEN
      IF(CTHER.LT.0.) TOTEN=CUMEN-CTHER
      CAC=100.*CAC/TOTEN
      CTR=100.*CTR/TOTEN
      CRO=100.*CRO/TOTEN
      CAE=100.*CAE/TOTEN
      CGE=100.*CGE/TOTEN
      CCL=100.*CCL/TOTEN                                                1[607]
      CDIF=100.*CDIF/TOTEN
      CTIRE=100.*CTIRE/TOTEN
      CUMENM=-100.*CUMENM/TOTEN
C
C     COMPUTE SUBTOTALS OF ENERGY LOSSES
C
      C345   = CGE + CDIF + CTIRE + CCL                                 1(627)
      C2345  = CTR + C345                                               1[607] [627]
      C1TC7  = CAE + CFC + CAC + C2345
C
      CENERG=100.*CTHER/TOTEN
      IF(CENERG.LT.0.)  CENERG=0.
      CDR=100.*CDR/TOTEN
C
C     TOTAL INCLUDING BRAKE AND RETURN TO ENGINE
C
      CTOTAL = C1TO7 + CDR + CUMENM
      PCKE=-CKE
      PCPE=-CPE
      PCROT=-CROT
C
C     ENERGY LOSSES
C
      CALIT=CTOTAL+CENERG
      PPHEHR=CUMFU/CUMEN
      HPMI=CUMEN/CUMD
C
C     PRINT UPSHIFT/DOWNSHIFT GEAR DATA
C
      IF(.NOT.(FILIN.OR.SECLIN.OR.ENDLIN.OR..NOT.LIMPBN) )   GO TO 770
      NSHIFT     = 0
      DO 760  I = 1,20
      DO 760  J = 1,2
  760 NSHIFT     = NSHIFT + NSGEAR(I,J)
      SMILE      = NSHIFT / CUMD
      ASFESG= NSHIFT / NSEG
C
      CALL SIMLET(8)                                          !OUTPUT SUMMARY OF SHIFTING.
C
  770 IF(ABS(PCKE).LT.1.E-3) PCKE=0.
      IF(ABS(PCPE).LT.1.E-3) PCPE=0.
      IF(ABS(PCROT).LT.1.E-3) PCROT=0.
      FPGAL  = FSPGR * 62.4261 / 7.48052
      TFRC1  = FRC1 * 1000.
      TFRC2  = FRC2 * 1000.
C
      CALL SIMLET(9)                                          !FINAL SUMMARY PAGE.
      IECOND=1                                                !FLG NO ERROR.
      RETURN                                                  !BYE, WE LANED SAFELY.
C
C     PRINT OUT A LINE ONLY IF LAST TIME STEP IN SEGMENT HAS BEEN SPECIF
C
  112 IF(.NOT.ENDLIN) GO TO 114
      IF((ITS.GT.1).OR.(ACCEL.GE.0.).OR.(V.GT.1.E-4))  GO TO 123
      ABR=ABRO
      IF(ABR.LT.ABROO-10.) ABR=ABROO
C
  123 CALL SIMLET(5)                                          !OUTPUT DATA LINE.
C
  114 CALL SIMSTS("34)                                        !REPORT END OF DRS SEG.
      ISEGO=ISEG                                              ![610]RESET OLD SEG CNTR.
      ISEG=ISEG+1                                             !INC DRS SEG #.
      LBRAKE=.FALSE.                                          !TURN BRAKES OFF.
      ABRO=ABR                                                !SAVE BRAKE VALUE.
      ABR=0.                                                  !ZERO BRAKE.
      IF(.NOT.(ACCEL.LT.1.E-3.AND.V.LT.1.E-5))GO TO 5         !(IDLE?)NO, GO TO NEXT SEGMENT.
      IF(NGEAR.NE.1)NSGEAR(NGEAR,2)=NSGEAR(NGEAR,2)+1         !YES, COUNT DOWNSHIFT IF NOT ALREADY IN GEAR 1.
      NGEAR=1                                                 !GO INTO GEAR 1.
      GO TO 5                                                 !NEXT DRS SEG PLEASE.
C
 5000 CALL SIMLET(10)
      GO TO 5040
 5020 CALL SIMLET(11)
 5040 IF(ETY)CALL EXIT                                        !(PSEUDO-TTY)YES, EXIT.
      CALL RESETM                                             !NO, RESET.
C
      END
```

```
C
C       ENTFY PCINTS:   SIMINT
C
C       SUBFOUTINES CALIEL:     GCEACK, KPTIME, BESETM, SHIFTS,
C                       SIMLPT, TTYCLR
C
C       CALLEC EY:      SINCTR
C
C       EDIT HISTCRY
C
C       [607]/SS-4-10-78        CIUTCB
C       [612]/SS-6-23-78        MCDIFY DYN HP
C       [614]/SS-7-5-78         HISTOGRAM INITIALIZATION
C
C********************************************************
C
C
        INCLUDE 'COMMS/NCLIST'
C
        DIMENSICN DYNVAL(11),WGTLIM(0/11)
C
C
        DATA DYNVAL/7.8,8.3,8.8,9.4,9.9,10.3,11.2,12.,12.7,13.4,13.9/
        1,WGTLIM/1625.,1876.,2126.,2376.,2626.,2876.,3251.,3751.,4251.
        2,4751.,5251.,5751./,JCT/5/
C
C**********************************************************************
        JCT=5
C
C       ZERC OUT GEAR TIME DISTFIEUTICN ARRAY
C
        CALL ZERCP(GTIMAC,20)
C
C
C...    HISTOGFAM INIT
C
        CALL ZEFOF(HIST,1600)
        FIFTOR=0.
        FIBFPM=FRPIN(1)
        TOPFPM=FPMAX(1)
        IF(KENG.EC.1)GO TC 10
        IF(FNMIN(2).NE.0..AND.ENMIN(2).LT.FIRRPM)FIRRPM=ENMIN(2)
        IF(PPMAX(2).GT.TCFRPM)TCPFPM=EPMAX(2)
C
10      IF(FIRREM.LT.100. .CR. TOERPM.GT.10000.)WRITE(JCT,12)FIRRPM,TOPRPM
12      FORMAT(' ?SIMINT - Warning, MIN & MAX RPMS are',2F10.2,/,
        2       ' Better check engine map for faulty data!!')
        DO 20 IE=1,NENG
        IM=1
        IF(IE.EC.2)IM=5
        DO 20 IF=1,NFPM(IE)
        DO 20 IT=1,20
        TEMF=EMAP(IE,IT,IP)
        IF(TEMP.LT.FIFTCR)FIRTOF=TEMP
        IF(TEMP.GT.TOFTCB)TCPTOF=TEMP
20      CONTINUE
C
28      DO 10 I=1,10
        IF(FIFFPM+I*2000.GT.TCPFPP)GO TO 32
10      CONTINUF
```

```
      I=10
32    DELRPM=I*100
      FIRRPM=IFIX(FIRRPM/DELRPM)*DELRPM
      IF(FIRRPM+DELRPM*20. .LT.TOPRPM)GO TO 28
C
38    DO 40 I=1,10
      IF(FIRTCR+I*200.GT.TOPTCR)GO TC 42
40    CONTINUE
      I=10
42    DELTOR=I*10
      FIFTOR=(IFIX(FIFTCR/DELTOR)-1.)*DELTOR
      IF(FIRTCR+DELTOR*20..LT.TCPTOR)GO TO 38
C
C     INITIALIZE CONSTANTS TO BE USED DURING A PARTICULAR SIMULATION
C
      CALL KPTIME(1)                                        ! SAVE START OF SIM RUN TIME
      CALL KPTIME(2)                                        ! SAVE START OF INIT COND RUN TIME
      IF(TTY) CALL TTYCIR                                   ! (PHYSICAL TTY IS JCT?) YES, CLEAR JCT INPUT BUFF
      AIA=0                                                 !ZERO ACCESSORY INERTIA.
      DO 100  I = 1,20
      IF(I.LE.NACC) AIA=AIA+AIAS(I)                         ! SUM ACCESSORY INERTIA'S
      DO 100  J = 1,2
      NSGEAR(I,1)=0
100   NSGEAR(I,2)=0                                         ! ZERO UP&DOWN SHIFT ACCUMS
C     PHI=0.                                                !COMMENTED OUT TO INSURE MODIFY PHI WORKS
      FAC=.0025168
      AA1=FRC1*WGT
      AA2=FRC2*WGT
      AA3=FAC*CD*AREA
      AA4=WGT/32.17
      AA5=14./WRAD
      RARSQ=RAR*RAR
      AA6=DD1*(NLSG*AIW+RARSQ*AIP)
      AA7=BB1*RARSQ
      AA8=RAR*EFAR(1)                                       ! ASSUME 1 AXLE
      IF(NRAX.EQ.2) AA8=1./(.5/AA8+.5/(RAR*EBAR(2)))        ! (2 AXLES?) YES.
      AA9=BE1*(EINER+AIA+AI1)
      AA10=7.480520/(FSPGR*62.426134)
      IF(.NOT.LDYNA) GO TO 200                              !(DYNA SIM?)NO.
      IF(LDYNCV)GO TO 200                                   ![612](OVERRIDE?)NO
      IF(WGT.LT.WGTLIM(0)) GO TC 125                        !YES,(WGT ABOVE MIN?)NO.
      DO 120 I=1,11                                         !SCAN WGT TABLE.
      IF(WGT.LT.WGTLIM(I)) GO TC 130                        !(GOT WGT CLASS?)YES.
120   CONTINUE                                              !NC, TRY NEXT CLASS.
125   WRITE(JCT,1040)                                       !? VEH WGT TO LOW OR HGH FOR DYNA SIMULATION.
      CALL RESETM                                           ! *RESET, TURKEY, YOU BLEW IT.
130   DYN=DYNVAL(I)                                         !GET FAC FOR AERO DRAG CALC.
C
C     INITIALIZE ALL ACCUMLATORS TO ZERO
C
200   RTHF=4000.                                            !RATE OF CHG FOR THROTTLE.[400 FOR 20%,4000FOR 100%]
      LLSH=.FALSE.
      CUMEN=0.
      CUMFU=0.
      CUMENR=0.
      CEDR=0.
      LBRAKE=.FALSE.
      CLUTCH=.FALSE.                                        ![607]
      CCL=0.                                                ![607]
      CTI=0.
      CDI=0.
      CEI=0.
      CFI=0.
      CFD=0.
      CTA=0.
      CDA=0.
      CEA=0.
      CFA=0.
      CTD=0.
      CDD=0.
      CED=0.
      CFD=0.
      CTCE=0.
      CDCR=0.
      CECE=0.
      CFCR=0.
      CAC=0.
      CTR=0.
      CRC=0.
      CAE=0.
      CBR=0.
      CGB=0.
      CDIF=0.
      CTIRE=0.
      CENG=0.
      CPE=0.
      CKE=0.
      CROT=0.
      ABR=0.
      ABRC=0.
      NGCCAL=0
C
      T=T0                                       !SET INITIAL CONDITIONS AS SPECIFIED BY DRIVING SCHEDULE

      HPEC=0.
      D=D0/5280.
      TTOT=0.
      DTOT=0.
      IC=0
C
      V=V0
      NDRTE=0                                    ! # OF RTE SEGS EX. IN PAST RTE SECTS
      NRTSEG=1                                   !PTR TO 1ST RTE SEG.
      IF(IDYNA) GO TO 300                        !(DYNA SIM?)YES.
      VSWIND=RVWIND(1)                           !GET WIND SPEED.
      IF(VSWIND.GT.300.) VSWIND=VWIND            !(RUN TIME DEFAULT FLAGGED?)YES, GET IT.
      BPER=RGRADE(1)*.01                         !ROAD GRADE.
      ENDFSG=RDIST(1)                            !END OF RTE SEG.
      ENDFTE=RDIST(NRDIST)                       !END OF RTE OR RTE SECT, TO BE FOUND OUT LATER.
      ROADC=RCOEF(1)                             !ROAD COEFF.
      GO TO 310                                  !AVOID.
C
C    DYNAMOMETER SPECS.
C
  300 ENDFSG=1.E20                               !INFINITE ROAD. MOST ROADS THAT GO IN CIRCLES ARE.
      ENDFTE=1.E20                               !INFINTE ROAD.
      ROADC=1.0                                  !ROAD COEFF. PERFECT.
      BPER=0.                                    !GRADE.
      VSWIND=0.                                  !NO WIND.
      PHI=0.                                     !WIND ANGLE.
      FAERO=D.
C
```

```
 310   CPHI=COS(PHI)
       VWINDS=VSWIND*SIN(PHI)
       VWINDC=VSWIND*CPHI
       NGEAR=NGO                                                  !GET INITIAL COND GEAR.
       IF ( NGEAR.EQ.0 )   NGEAR = 1                              !(INITIAL GEAR GIVEN?)NO, CAN'T HAVE THAT, LET'S TRY 1ST.
       MXNGCH=NUMG*2+5                                            !CALC MAX ATTEMPTS TO FIND GEAR OR WE HAVE STUTTERS.
       IENG=JENG(NGEAR)                                           !GET INITIAL ENG # TO USE 1 OR 2.
       RPMEO=ENMIN(IENG)                                          !SET OLD RPME TO MIN ENG RPM.
       DRPMC=0.
       RPMC=0.
       DTC=0.
       RPM2C=0.
       RPMWO=0.                                                   !OLD VAL OF WHEEL ROM.
       ACCEL=A0                                                   !GET INITIAL ACCEL.
       TTT1=0.
       CUMC=0.                                                    !DISTANCE ACCUM.
       CUMT=0.                                                    !TIME ACCUM.
       CUMG=0.
       CUMTLS=0.                                                  !CUMT AT END OF LAST SHIFT.
       CSTIM=0.                                                   !SHIFT TIME ACCUM.
C
C     GO BACK FROM WHEELS TO ALLOW CAB TO REACH INITIAL CONDITIONS
C     BEFORE SIMULATION BEGINS
C
       TOLD=T
       VOLD=V
       ISEG=1
       NDSEG=0                                                    ! # OF DRS SEGS EX. IN PAST DRS SECTS
       DT      = .1
       ING     = 0
       NGOLD   = NGEAR
       LCLTCH=.FALSE.                                             !FLG CLUTCH OUT.
       SHFTNG=.TRUE.                                              !FLG WE ARE SHIFTING.
       ITS=0                                                      !FLG IN INITIAL COND.
 401   CALL GOBACK                                                !DETERMINE VEH STATE.
       CALL SHIFTS                                                !GEAR?
       IF ( NGEAR.EQ.NGOLD )   GO TO 403                          !(SETTLED INTO GEAR?)YES.
       ING     = ING + 1                                          !NO, INC ATTEMPTS TO FIND GEAR.
       IF ( ING.GT.MXNGCH )   GO TO 402                           !(ARE WE EVER GOING TO FIND IT?)NO.
       RPMEO   = RPME                                             !WELL MAYBE, SET SOME OLD VALUES.
       RPMWO   = RPMW
       NGOLD   = NGEAR
       GO TO 401                                                  !TRY AGAIN.
C
 402   CALL SIMLFT(2)                                     !% CAN'T FIND INITIAL GEAR, BUT WE'LL TRY SEE IF SIM FLIES ANYWAY.
C
 403   SHFTNG=.FALSE.                                             !FLG NOT SHIFTING.
       RPMEO=RPME
       RPMWO=RPMW
       VOLD=V
       DT=1000.                                                   !LARGE DT TO ALLOW THINGS TO SETTLE DOWN.
       CALL GOBACK                                                !ZAP.
       RPMWO=RPMW
       RPMEO=RPME
       VOLD=V
       CALL GOBACK                                                !AND ZAP AGAIN.
       RPMWO=RPMW                                                 !MORE OLD VALUES.
       RPMEO=RPME
       HPAC=TORQA*RPME*RE2                                        !HORSE POWER.
       HPEC=TORQE*RPME*RE2
       HP1C=TORQ1*RPM1*RE2
```

```
      HP2C=TORQ2*RPM2*EE2
      HPC1C=TORC2*RPMC*EB2
      HEPC=TORQP*RPMP*EE2
      HPWC=TORQW*RPMW*EE2
      FRATEO=FRATE                                                    !FORCES.
      FRATGO=FRATE*AA10
      FROILC=FROLL
      FGRADC=FGRADE
      FWHEEC=FWHEEL
      FAEROO=FAERO
      DTIRE=0.                                                        !LOSSES.
      DTIREO=ABS(HPWO*TTT1)
      ALOSGO=ABS(HP2C-HEPC)
      ALOSRO=ABS(HPEO-HEWO)
      ALOSCC=ABS(HP2O-HE1O)
      ALOSLC=ABS(HP2C-HECLO)
      PCTHR=100.*(TORCE-TMIN)/(TMOT-TMIN)                             !% THROTTLE.
      RPMEI  = RPME
      RPMWI  = RPMW
      NGO    = NGEAR
      TORCEO=TORQE
      TORCAO=TORQA
      TMINO=TMIN
      ATORQP=0.
      ARPME=0.
      WRITE(JCT,1050)                                                 ![REACHED INIT COND].
      IF(LIMPRN.AND.(.NOT.(ENDLIM.OR.SECLIM.OR.MILIM))) GO TO 5       !(WE DOING ANY DATA OUTPUT?)NO.
      FEINST=V/(FRATE*AA10)                                           !YES. CALC INSTANT MPG.
      HPE=TORCE*RPME*EE2
      BSFC=99.99                                                      !FOR LOOKS.
      IF(HPE.GT.0.) BSFC=FRATE/HPE                                    !(ENG GOT POWER?)YES, CALC A REAL VALUE.
      CUMFE=0.                                                        !GREAT MPG.
      RDLD=(FWHEEL-FACCEL)*V/375.
      HPW=TORQW*RPMW*EE2
      HPE=TORCE*RPME*EE2
      EFFC=SR*TR
      IF(COAST.AND.EFFC.GT.1.) EFFC=1./EFFC
      IF(PCTHR.LT.0.) PCTHR=0.
C
      CALL SIMLET(1)                                                  !INIT SIM PRINT ROUTINE.
      CALL SIMLET(5)                                                  !OUTPUT VEH STATUS AT END OF INIT COND.
C
    5 RETURN                                                          !TAXI RIDE OVER, LET'S SEE IF IT FLIES, GOOD LUCK.
C
 1040 FORMAT(/' ? SIMINT - VEHICLE WEIGHT OUT OF RANGE FOR DYNAMOMETER'
     1,' SIMULATION.'/)
 1050 FORMAT(' [REACHED INITIAL CONDITIONS]')
      END
      SUBROUTINE SIMLPT(ILPTT)
C
C     ENTRY POINTS:   SIMLPT
C
C     SUBROUTINES CALLED:     CTRLD,  TTYSET
C
C     CALLED BY:      SIMCTR, SIMINT
C
C     EDIT HISTORY
C     [607]/SS-4-10-78          CLUTCH
C     [615]/SS-10-4-78          DIESEL PRINTOUT
C     [628]/JC-02-25-81         REQUEST TO BE INSERTED BY B. ZUB
C     [625]/JC-03-16-81         REQUEST BY B. ZUB
C
C*********************************************************************
C
C
      DIMENSION TEMER(21),TEMET(21)
      INCLUDE 'COMMS/NCLIST'
C
      DATA JCT/5/
C
C******************************************************************************
C
      JCT=5
      LPT=IUNIT                                                   !GET LPT UNIT #.
      ISEGA=ISEG+NDSEG                                            !CALC CURRENT DRS SEG.
      GO TO (10,20,20,20,30,20,20,800,900,20
     2 ,20,1200,1300,1400),ILPTT                                  !INIT DEPENDING ON TASK.
C
   10 NPAGE=0                                                     !ZERO PAGE CNT START SIMULATION.
      GO TO 40
C
   20 IF(NLINE.LT.50) GO TO 50                                    !NEED AT LEAST 10 LINES. (GO 'EM?)YES.
      GO TO 40
C
   30 IF(NLINE.LT.60) GO TO 50                                    !NEED AT LEAST 1 LINE.(GO IT?)YES.
C
   40 NPAGE=NPAGE+1                                               !NO, INC PAGE #.
      NLINE=15                                                    !HEADER TAKES 15 LINES. SET LINE CNT.
      IF(TTY .AND. (.NOT. LIMTTY))
     1 WRITE(JCT,2010) DATE,NPAGE,TITLE,DNAME,RNAME
      IF(LPT.EQ.IUNIT.AND.(.NOT.LLPT))GO TO 9000
      WRITE(LPT,2000)  DATE,NPAGE,TITLE,DNAME,RNAME               !WRITE HEADER TO LPT.
C
   50 IF(LPT.EQ.IUNIT.AND.(.NOT.ILPT))GO TO 9000
      GO TO (9999,200,300,400,500,600,700,800,900,1100
     2 ,1120,1200,1300,1400),ILPTT                                !GO OUTPUT.
C
C
C*****************************************************************************
C
C  ERROR MESSAGE - FAILURE TO REACH INTIAL CONDITIONS
C
  200 CALL TTYSET                                                 !RESET ¬O AND CLR TTY INP BUFFER.
      WRITE(LPT,204)  NGEAR                                       !# ERR MESS.
      IF(LPT.EQ.6) NLINE=NLINE+10                                 !(LPT OUTPUT?)YES, INC LINE CNT.
      GO TO 9000
C
C****************************************************************************
C
```

```
C    ERROR MESSAGE - SHIFT STUTTER
C
  300   CALL TTYSET                                                   !RESET -O AND CLR TTY INP BUFFER.
        WRITE(LPT,1990) ISEGA,CUMT,NGCNT,NSFSEG(ISEG),DT              !?ERR MESS.
        GO TO 9000
C
C******************************************************************************
C
C    ERROR MESSAGE - FAILURE TO REACH TERMINAL VELOCITY
C
  400   CALL TTYSET                                                   !RESET -O AND CLEAR TTY BUFFER.
        WRITE(LPT,404) ISEGA                                          !?ERR MESS.
        GO TO 9000
C
C******************************************************************************
C
C    OUTPUT DATA
C
  500   TSR=SR
        TEFFC=EFFC
C
C   GRADEABILITY COMPUTATION ADDED 9/5/79 BY TIM COLLINS/KII
C   NO VALID ANSWER CAN BE HAD IF HPW < 0, SO A DUMMY ANSWER
C   IS INSERTED TO GIVE ******** ON OUTPUT
C   NOTE: TAN(X)=SIN(X)/COS(X)
        GRDELT=10E20
CCC     HRDIF=HPW-RDLD                                                 ![625]OCT.21,1980
        HRDIF=(HPW+(TCRCP*REME/5252.0))-RDLD                            ![625]OCT.21,1980
        IF(HRDIF.LT.0)  GO TO 501
        GRDELT= ASIN(  (HRDIF*374.15) / (V*WGT)    )
        GRDELT = 100.0  *  ( SIN(GRDELT)/COS(GRDELT)  )
501     IF(LOCKOP(NGEAR))TSR=1.
        IF(LOCKOP(NGEAR))TEFFC=1.
        IF(LPT.EQ.IUNIT)                                               !(LPT ON?)YES.SEND DATA TO LPT.
        1WRITE(LPT,1002) CUMT,COMD,V,ACCEL,FRINST,BSFC,CUMFB,NGEAR,RDLD
        1.HEW,HEE,RPME,TORQE,VAC,TSR,TEFFC,PCTHR,ISEGA,ABR,GRDELT
        IF(TTY .AND. (.ACT. LIMTTY) )                                  !(JCT IS TTY & TTY ON?)
        1 WRITE(JCT,1012) CUMT,CUMD,V,ACCEL,NGEAR,HPW,HPE              !YES. SEND DATA TO TTY.
        1.REME,TORQE,ISEGA
        NLINE=NLINE+1                                                  !INC LIN CNT.
        GO TO 9999
C
C******************************************************************************
C
C    WARNING MESSAGE - OVER SHOT END OF CONSTANT VELOCITY SEGMENT
C
  600   CALL TTYSET                                                   !RESET -O AND CLR JCT INP BUFFER.
        WRITE(LPT,602) ISEGA                                          !% WARNING MESSAGE.
        IF(LPT.EQ.JCT) CALL CTRID('ERROR')                            !(DOING OUTPUT TO JCT?)YES.
        NLINE=NLINE+10                                                !INC LINE CNT.
        GO TO 9000
C
C******************************************************************************
C
C    ERROR MESSAGE - INSUFFICIENT TORQUE TO SHIFT
C
  700   CALL TTYSET                                                   !RESET -O AND CLR JCT INP BUFFER.
        WRITE(LPT,1980)  TORQE,TMIN,ISEGA,CUMT                         !?ERR MESS.
        GO TO 9000
C
C******************************************************************************
```

```
C
C   PRINT SHIFTING DATA
C
  800   IF(LPT.EQ.5.AND.LIMTTY.AND.IDEBUG.NE.2) GO TO 780                  !OUTPUT SHIFTING DATA?) NO.
        NPAGE=NPAGE+1                                                      !INC PAGE #.
        WRITE(LPT,1010)  DATE,NPAGE,NSHIFT,SMILE,NUMG,NSGEAR               !OUTPUT SHIFTS INTO AND OUT OF GEAR.
        IF(IDEBUG.NE.2) GO TO 780                                          !DEBUG SHIFTS?) NO.
C
        NLINE=20                                                           !INC LINE CNT.
        J=0
        WRITE(LPT,1770) ASFPSG                                             !AUG SHIFTS/DRS SEG.
  772   K=0                                                                !ASSUME LAST PASS.
        IB=(16*J)+1                                                        !CALC PTR TO DATA FOR CURRENT LINE.
        IE=NSEG                                                            !SET END PTR TO END OF DATA.
        ITE=16*(J+1)                                                       !(WILL REST OF OUTPUT FIT THIS LINE?) YES.
        IF(IE.LE.ITE) GO TO 775                                            !(WILL REST OF OUTPUT FIT THIS LINE?) YES.
        IE=ITE                                                             !NO, SET END PTR TO WHAT WILL FIT.
        J=J+1                                                              !INC PASS CNT.
        K=1                                                                !FLG NOT LAST PASS.
  775   WRITE(LPT,1771)  (I,I=IB,IE)                                       !OUTPUT DRS SEG #.
        WRITE(LPT,1772)  (ITYSEG(I),I=IB,IE)                               !OUTPUT DRS SEG TYPE.
        WRITE(LPT,1773)  (NSPSEG(I),I=IB,IE)                               !OUTPUT SHIFTS THAT DRS SEG.
        NLINE=NLINE+7                                                      !INC LINE CNT.
        IF(K.EQ.0) GO TO 780                                               !(DONE?) (LAST PASS?) YES.
        IF(NLINE.LE.52) GO TO 772                                          !NO,(ENOUGH LINES LEFT?) YES.
        NPAGE=NPAGE+1                                                      !NO, INC PAGE CNT.
        NLINE=4                                                            !SET LINE CNT.
        WRITE(LPT,1774) NPAGE                                              !PAGE AND WRITE HEADER.
        GO TO 772                                                          !CONTINUE.
780     IF(IDEBUG.EQ.2) NPAGE=NPAGE+1                                      ![628]
        IF(IDEBUG.EQ.2) WRITE(LPT,1774) NPAGE                              ![628]
        DO 781 I = 1,20                                                    ![628]
781     GTIMAC(I)=GTIMAC(I) / CONT * 100.                                  ![628]
        IL=0                                                               ![629]
        IU=0                                                               ![629]
        DO 783 K=1,NUMG                                                    ![629]
        IF(LOCKUP(K)) GO TO 782                                            ![629]
        IU=IU+1                                                            ![629]
        IDATA(IU)=K                                                        ![629]
        GO TO 783                                                          ![629]
782     CONTINUE                                                           ![629]
        IL=IL+1                                                            ![629]
        LOCKG(IL)=K                                                        ![629]
783     CONTINUE                                                           ![629]
C                                                                          ![628]
C       WRITE OUT TRANSMISSION NAME, GEARS LOCKED, GEAR NAME, RATIO        ![628]
C       AND %TIME SPENT IN EACH                                            ![628]
        WRITE(LPT,11011)                                                   ![629]
        IF(IL.GT.0) WRITE(LPT,11010) TRNAM, (LOCKG(I),I=1,IL)              ![629]
        IF(IU.GT.0) WRITE(LPT,11009) TRNAM, (IDATA(I),I=1,IU)              ![629]
        WRITE(LPT,11013)                                                   ![628]
        WRITE(LPT,11012) (I,GNAME(I),GRAT(I),GTIMAC(I), I=1, NGTR)         ![628]
        GO TO 9000
C
C*****************************************************************************
C
C     WRITE TOTALS FOR THE ENTIRE SIMULATION
C
  900   NPAGE=NPAGE+1                                                      !INC PAGE CNT.
        ANSV=(14.0/WRAD)*GRAT(NGTR)*RAR                                     !ADDITION TO TIRE OUTPUT
        WRITE(LPT,1005)  DATE,NPAGE,TITLE,CUMFE,HPMI,PPHPHR,VAVG
```

```
C
C      CAPSULE SUMMARY OF DATA USED FOR THIS SIMULATION
C
        DIECCM=0                                                    ![615]
        IF(IDIES)DIECCM='[D]'                                        ![615]
        IF(.NOT.IDYNOV)GO TO 920                                    ![612]
        WRITE(LPT,1108) CNAME,RNAME,DYN,VNAME,ENAME(1),DIECOM
        1,CNAME,SNAME,WGT
        1,STROKE,DISP,RAR,VWIND,FEGAL,AREA,CD,TFRC1,TFRC2,TIREFF,
     1    ANSV
        GO TO 940
C
920     WRITE(LPT,1008) CNAME,RNAME,VNAME,ENAME(1),DIECOM
        1,CNAME,SNAME,WGT
        1,STROKE,DISP,RAR,VWIND,FEGAL,AREA,CD,TFRC1,TFRC2,TIREFF,
     1    ANSV
C
C      ACCUMULATORS BROKEN OUT BY IDLE,CRUISE,ACCEL,DECEL
C
940     WRITE(LPT,1006)  CUMT,CTCR,CTA,CTD,CTI,CUMD,CDCR,CDA,CDD,CDI,
        1 CUMEN,CECR,CEA,CED,CEI,CUMFU,CFCR,CFA,CFD,CFI,CFB
C
C      ENERGY SOURCES AND SINKS
C
        WRITE(LPT,1009)  CUMEN,ECKE,PCPE,PCROT
C
C      LOSSES AS PERCENT AVAILABLE TOTAL ENERGY
C
        WRITE(LPT,1007) CAC,CTR,CCL,CGE,CDIF,CTIRE,C345,C2345,CAB
        2,CRO,C1TC7,CBR,CUMENB,CTCTAL,CENERG,CALLT                  ![607]
C
C...    HISTOGRAM OUTPUT
C
        IF(LPT.EQ.JCT)GO TO 9000
        NPAGE=NPAGE+1
        TEMFR(1)=FIFRPM
        TEMFT(1)=FIFTOR
        DO 9420 I=2,21
        TEMFR(I)=TEMFR(I-1)+DELRPM
        TEMFT(I)=TEMFT(I-1)+DELTOR
9420    CONTINUE
C
        TOTIME=0.
        DO 9440 I=1,20
        DO 9430 J=1,20
        TIME=HIST(J,I,3)
        TOTIME=TOTIME+TIME
        IF(TIME.EQ.0. .OR. HIST(J,I,1).EQ.0)GO TO 9430
        HIST(J,I,1)=HIST(J,I,1)/TIME
        IF(HIST(J.I,1).EQ.0)GO TO 9440
        HIST(J.I,2)=HIST(J,I,2)/TIME
9430    CONTINUE
9440    CONTINUE
        DO 9460 I=1,20
        DO 9460 J=1,20
        HIST(J,I,3)=(HIST(J,I,3)/TOTIME)*100.
9460    CONTINUE
C
C...    GET INDICIES FOR BOUNDARY CUT BACK ON HISTOGRAM OUTPUT
C
        DO 9463 J=1,20
        DO 9462 I=1,20
        DO 9461 K=1,3
        IF(HIST(I,J,K).EQ.0)GO TO 9461
        ISR=J
        GO TO 9465
9461    CONTINUE
9462    CONTINUE
9463    CONTINUE
C
9465    DO 9468 J=20,1,-1
        DO 9467 I=1,20
        DO 9466 K=1,3
        IF(HIST(I,J,K).EQ.0)GO TO 9466
        IER=J
        GO TO 9470
9466    CONTINUE
9467    CONTINUE
9468    CONTINUE
C
9470    DO 9473 J=1,20
        DO 9472 I=1,20
        DO 9471 K=1,3
        IF(HIST(J,I,K).EQ.0)GO TO 9471
        IST=J
        GO TO 9475
9471    CONTINUE
9472    CONTINUE
9473    CONTINUE
C
9475    DO 9478 J=20,1,-1
        DO 9477 I=1,20
        DO 9476 K=1,3
        IF(HIST(J,I,K).EQ.0)GO TO 9476
        IET=J
        GO TO 9480
9476    CONTINUE
9477    CONTINUE
9478    CONTINUE
C
C...      DO HISTOGRAM OUTPUT
C
C
C
C REV TC-07-30-79  TO OUTPUT HP  (=RPM*TORQUE/5252.0)           [    ]
C
9480    DO 9481 I = 1,20
        DO 9481 J = 1,20
9481    IF(HIST(I,J,1).NE.0)HIST(I,J,0)=HIST(I,J,2) * HIST(I,J,1) / 5252.0
C
C SEE ALSO FORMAT LINES 3020 AND 3080
C
        IF(IET-IST.GT.13)GO TO 9490
        IF(IET.GE.14)GO TO 9482
        IET=IST+13
        GO TO 9484
9482    IST=IET-13
9484    WRITE(LPT,3060)DATE,NPAGE,(TEMPT(I),I=IST,IET+1)
        WRITE(LPT,3080)
        2(TEMPR(J),((HIST(I,J,K),I=IST,IET),K=0,3),J=ISR,IER)
        WRITE(LPT,3100)TEMPR(IER+1)
C
```

```
         GO TO 9000
C
9490     WRITE(LPT,3000) DATE,NPAGE,(TEMPT(I),I=1,11)
         WRITE(LPT,3020) (TEMPR(J),((HIST(I,J,K),I=1,10),K=0,3),J=1,20)
         WRITE(LPT,3040) TEMPR(21)
C
         NPAGE=NPAGE+2
         WRITE(LPT,3000) DATE,NPAGE,(TEMPT(I),I=11,21)
         WRITE(LPT,3020) (TEMPR(J),((HIST(I,J,K),I=11,20),K=0,3),J=1,20)
         WRITE(LPT,3040) TEMPR(21)
C
         GO TO 9000
C
C*************************************************************************
C
C    ERROR MESSAGE - STALL CONDITION
C
 1100    CALL TTYSET
         WRITE(LPT,2100) ISEGA,CUPT,RPM1,ENMIN(1)
         GO TO 1140
 1120    WRITE(LPT,2120) ISEGA,CUMT,RPM1,RPMAX(IENG)
 1140    IF(LPT.EQ.JCT)CALL CTRLE('STALL')
         GO TO 9000
C
C*************************************************************************
C
C    ERROR MESSAGE - BAD DRIVING SCHEDULE
C
1200     CALL TTYSET
         WRITE(LPT,2200) ISEGA,CUPT
         CALL RESETM
         GO TO 9000
C
C*************************************************************************
C
C    ERROR MESSAGE - NO TORQUE
C
1300     CALL TTYSET
         WRITE(LPT,2300) ISEGA,CUMT
         CALL CTRLD('MOTOR')
         GO TO 9000
C
C*************************************************************************
C
C    ERROR MESSAGE - UNDEFINED SEGMENT TYPE
C
1400     CALL TTYSET
         WRITE(LPT,2400) ISEGA
         GO TO 9000
C
C**********************************************************************
C
 9000    IF( LPT.EQ.JCT ) GO TO 9999                                  !(OUTPUT TO JCT?)NO.
         LPT=JCT                                                      !YES, GET JCT UNIT #.
         GO TO 50                                                     !GO OUTPUT.
C
 9999    RETURN                                                       !DONE, BYE.
C
C********************************************************************
C********************************************************************
C
```

```
C   FORMAT STATEMENTS
C
C===========================================================================
 2000   FORMAT (1H1/6XA9,31X'VEHICLE PERFORMANCE SIMULATION'33X,
        1         5HPAGE ,I3/6X,9(1H-),31X,30(1H-),33X,8(1H-)//
        1,26X,12HRUN TITLE ( ,12A5,2H ),/,26X,74(1H-),//
        1         29X'DRIVING SCHEDULE ( '
        2A10' )'10X'USING ROUTE ( 'A10' )'/29X31(1H-)10X26(1H-)//
        1,32X'INST'8X'COM'65X'DES',8X,'GRADE-',/
        35X,4RSEC.,3X,5HMILES,3X,3HMPH,3X,3HACC,4X,3HMPG,2X,4HBSFC,2X,
        43HPPG,1X,4HGEAR,3X,4HRILE,4X,3HHPW,4X,3HHPE,3X,3HRPM,2X,4HTORQ,
        53X,3HVAC,4X,2HSR,3X,3HETA,1X,7HPCT.WCT,1X,3HSEG,1X,6HBRAKES,
        1  8H ABILITY,/
        62X,128(1H-)/)
C
C===========================================================================
  204   FORMAT (//2X,5X,60(1H*)//2X,30H***** FAILURE TO REACH INITIAL,
        1         23H CONDITION GEAR SETTING/2X,16H***** EXECUTION ,
        2         26HCONTINUES STARTING IN GEAR,I4//2X,5X,60(1H*)//)
C
C===========================================================================
 1990   FORMAT(//,7X,65(1R*),//,2X
        1,62H***** SHIFT STUTTER DETECTED DURING CONSTANT VELOCITY SEGE
        2MENT ,/,43H  ***** UNABLE TO ATTAIN CONSTANT VELOCITY   ,/
        3,48H  *****       POSSIBLY DUE TO FAULTY SHIFT LOGIC  ,/
        4,13H  ***** SEG = ,I5,10H   AT TIME ,F9.2,17H SECS.   PASSES =
        5,I4,/
        6,25H  ***** SHIFTS THIS SEG = ,I3,7H   DT =,E11.5,/
        6,29H  ***** SIMULATION,TERMINATED ,//,7X,60(1H*),//)
C
C===========================================================================
  404   FORMAT (//2X,5X,60(1H*)//2X,31H***** FAILURE TO REACH TERMINAL,
        1         21R VELOCITY FOR SEGMENT,I5/2X,16H***** SIMULATION,
        2         11H TERMINATED//2X,5X,60(1H*)//)
C
C===========================================================================
 1002   FORMAT (1X,F8.2,F8.3,F6.1,F6.2,F7.2,F6.2,F5.1,1X,I3,1X,3F7.1,
        1         2F6.0,F6.1,2F6.3,F6.0,2X,I4,F7.1,F8.2)             ! GRADEABLILITY ADDED 9-05-79
C
C===========================================================================
 1980   FORMAT (//2X,5X,60(1H*)//2X,31H***** TORQUE REQUIRED BY ENGINE,
        1   32H BELOW MINIMUM FFICR TC SHIFTING/2X,15H***** TORQUE = ,
        2   E10.3,5X,10HMINIMUM = ,E10.3/2X,17H***** FOR SEGMENT,I5,
        3   2X, 8E AT TIME,F9.2/2X,16H***** SIMULATION,
        4         11H TERMINATED//2X,5X,60(1H*)//)
C
C===========================================================================
 1010   FORMAT('1SHIFT FREQUENCY DATA BY GEAR'32XA9
        1,20X,5HPAGE ,I3,/1X,28(1H-),60X,8(1H-)
        1,///,3X,15HTOTAL SHIFTS  =,I6,5X,18HSHIFTS PER MILE  =,
        2         F6.1,5X,13HNUME GEARS  =,I3///3X,9HGEAR INTO,4X,
        3         37H1  2  3  4  5  6  7  8  9 10 11 12 13,
        4         21H 14 15 16 17 18 19 20//3X,8HUPSHIFTS,3X,20I3//
        5         3X,11HDCWNSHIFTS ,20I3//)
C
C
11011   FORMAT('2GEAR TIME DISTRIBUTION',/,1X,22('-'),//)
11010   FORMAT(' TRANSMISSION: ',A10,10X,' GEARS LOCKED UP: ', 20I4,/)
11009   FORMAT(' TRANSMISSION: ',A10,10X,' GEARS UNLOCKED: ', 20I4,/)
11013   FORMAT
        1 (/,1X,' NO.',4X,'     NAME:',4X,'    RATIO:',4X,'    %TIME:',
C
11012   FORMAT(1X,I4,4X,A10,4X,F10.3,8X,F6.2)
C
C=============================================================================
 1770   FORMAT(32H SHIFT FREQUENCY DATA BY SEGMENT,/,1X,31(1H-),//
        1,23H    SHIFTS PER SEGMENT =,F10.3)
 1771   FORMAT(//,11H    SEGMENT ,40I4)
C
C=============================================================================
 1772   FORMAT(/,11H    SEG TYPE,40I4)
C
C=============================================================================
 1773   FORMAT(/,11H    # SHIFTS,40I4)
C
C=============================================================================
 1774   FORMAT(33H1 SHIFT FREQUENCY DATA BY SEGMENT,50X,5HPAGE ,I3
        1,/,1X,31(1H-),//)
C
C=============================================================================
 1005   FORMAT (1H1/10X'VEHICLE PERFORMANCE SIMULATION'3XA9
        1,40X,5HPAGE ,I3,/10X,
        1          42(1H*)///1X'RUN TITLE = '12A5//' SCHEDULE AVERAGES'
        2          ,5X,18HFUEL ECONOMY      =,F6.2, 4H MPG/
        2          23X,18HWORK PER MILE     =,F6.2, 9H HP-HR/MI/
        3          23X,18HAVG SP FUEL CONS =,F6.2,10H LBS/HP-HR/
        4          23X,18HAVG SPEED         =,F5.1, 5H  MPH)
C
C=============================================================================
 1008   FORMAT (// 1X,19HADDITIONAL RUN DATA//
        1          3X,23HDRIVING SCHEDULE NAME =,1X,A10,
        2          3X,23HROUTE NAME              =,1X,A10/
        3          3X,23HVEHICLE NAME            =,1X,A10,
        4          3X,23HENGINE NAME             =,1X,A10,A5/
        5          3X,23HCONVERTER NAME          =,1X,A10,
        6          3X,23HSHIFT LOGIC NAME        =,1X,A10/
        7          3X,23HWEIGHT (LBS)            =,F7.0,
        8          7X,23HSTROKE (INCHES)         =,F7.2/
        9          3X,23HDISPLACEMENT (CU IN)    =,F7.1,
        1          7X,23HREAR AXLE RATIO         =,F7.2/
        1          3X,23HWIND VELOCITY (MPH)     =,F7.1,
        2          7X,23HFUEL DENSITY (LB/GAL) =,F7.2/
        3          3X,12HAERO DRAG  =,F6.2,2H ,F5.2,5X,
        4          7X, 8HTIRES  =,F7.2,2(2H ,F5.2)/
     5             40X,8H N/V    =,F7.2)
C
 1108   FORMAT (// 1X,19HADDITIONAL RUN DATA//
        1          3X,23HDRIVING SCHEDULE NAME =,1X,A10,
        2          3X,23HROUTE NAME              =,1X,A10,
        3          3X,'(',F5.1,')'/
        3          3X,23HVEHICLE NAME            =,1X,A10,
        4          3X,23HENGINE NAME             =,1X,A10,A5/
        5          3X,23HCONVERTER NAME          =,1X,A10,
        6          3X,23HSHIFT LOGIC NAME        =,1X,A10/
        7          3X,23HWEIGHT (LBS)            =,F7.0,
        8          7X,23HSTROKE (INCHES)         =,F7.2/
        9          3X,23HDISPLACEMENT (CU IN)    =,F7.1,
        1          7X,23HREAR AXLE RATIO         =,F7.2/
        1          3X,23HWIND VELOCITY (MPH)     =,F7.1,
        2          7X,23HFUEL DENSITY (LB/GAL) =,F7.2/
        3          3X,12HAERO DRAG  =,F6.2,2H ,F5.2,5X,
     4          7X, 8HTIRES  =,F7.2,2(2H ,F5.2)/
    5           40X,8H M/V   =,F7.2)
C
C=========================================================================
 1006   FORMAT (//1X, 6HTOTALS,20X,5HTOTAL,8X,16HPERCENT OF TOTAL/
     1          10X,44HVARIABLE (UNITS) AMOUNT (CRUISE ACCEL  DECEL,
     2          18H   IDLE ) (BRAKES)/
     3          10X,45H-------- ------- ------  ------ ------ ------,
     4          16H ------   ------
     5,/10X'TIME      (SECS)'F7.1,1X,4F7.1/
     6          10X,16HDISTANCE (MILES),F6.1,1X,4F7.1/
     7          10X,16HENERGY   (HP-HR),F7.2,  4F7.1/
     8          10X,16HFUEL      (LBS)  ,F7.2,  4F7.1,F9.1)
C
C=========================================================================
 1009   FORMAT (//1X,13HENERGY SUPPLY,28X,5HHP-HR/42X,5H-----/
     1          19X,22H(1) ENGINE              =,F8.2/
     2          19X,22H(2) KINETIC ENERGY   =,F8.2/
     3          19X,22H(3) POTENTIAL ENERGY =,F8.2/
     4          19X,22H(4) ROTATING INERTIA =,F8.2)
C
C=========================================================================
 1007   FORMAT(//1X'BREAKDOWN'22X'PERCENT ENGINE HP-HR'/32X,20(1H-)/
     1          18X,23H (1) ACCESSORIES       =,F7.2/
     2          18X,23H (2) TORQUE CONVERTER =,F7.2/
     2          19X,23H (3) CLUTCH            =,F7.2/
     3          18X,23H (4) GEAR BOX          =,F7.2/
     4          18X,23H (5) DIFFERENTIAL      =,F7.2/
     5          18X,23H (6) TIRE SLIP         =,F7.2/
     6          18X,23H           3+4+5+6  =,F7.2/
     7          18X,23H         2+3+4+5+6  =,F7.2/
     8          18X,23H (7) AERODYNAMIC DRAG =,F7.2/
     9          18X,23H (8) ROLLING RESIST    =,F7.2/
     1          18X,23H       SUBTOTAL 1- 8  =,F7.2/
     1          18X,23H (9) BRAKES            =,F7.2/
     2          18X,23H(10) ENGINE MOTORING  =,F7.2/
     3          18X,23H       SUBTOTAL 1-10  =,F7.2/
     4          18X,23H(11) OTHER ENERGY     =,F7.2/
     5          18X,23H           TOTAL 1-11  =,F7.2)
C
C=========================================================================
 1012   FORMAT(1XF8.2,1XF8.3,1XF6.1,1XF7.2,1XI3,4(1XF7.1) 1XI3)
C
C=========================================================================
 2010   FORMAT('1   'A9,9X'VEHICLE PERFORMANCE SIMULATION'9X'PAGE'I4//
        1,3X'RUN TITLE : '12A5' )'//
        2,8X'DRIVING SCHEDULE ( 'A10' )'6X'USING ROUTE ( 'A10' )'//
        3,T6'SEC.'T14'MILES'T23'MPH'T31'ACC'T35'GEAR'T43'HPB'
        4,T51'HPE'T59'RPM'T66'TORQ'T71'SEG'/1X72('-')/)
C
C=======================================================================
  602   FORMAT(' ? SIMCTR -  OVER SHOT END OF SEGMENT'I5
        1,/10X'THIS A CONSTANT VELOCITY SEGMENT AND OVER SHOOT'
        2,/10X'CAUSED BY SHIFT STUTTER OR SEGMENT TIME TOO SHORT'
        3,//' % SIMCTR - SIMULATION TERMINATED'/)
C
C=======================================================================
 2100   FORMAT(//7X,65('*'),
     1         //2X,'***** STALL CONDITION *****'
     2         /2X,'***** AT SEGMENT',I5,2X,' AT TIME',F9.2,
     2         /2X,'***** RPM1 (',F6.0,') IS LESS THAN RPM MINIMUM ('
     3,        F6.0,') ******'
     4         //7X,65('*'),//)
C
 2120   FORMAT(//7X,65('*'),
     1         //2X,'****** STALL CONDITION ******'
     2         /2X,'****** AT SEGMENT',I5,2X,' AT TIME',F9.2,
     2         /2X,'****** RPM1 (',F6.0,
     3         ') IS GREATER THAN RPM MAXIMUM ('
     3,        F6.0,') ******'
     4         //7X,65('*'),//)
C
C==================================================================
2200    FORMAT(' ? SIMCTR - SEGMENT ENDING WITH ZERO VELOCITY FOLLOWED',/,
     2         '         BY CONSTANT VELOCITY SEGMENT. (BAD DRIVING SCHEDULE)',/,
     3         '         AT SEG',I5,'      TIME',F10.2)
C
C==================================================================
2300    FORMAT(' ? SIMCTR - NOT ENOUGH TORQUE TO MOVE VEHICLE',/,
     2         '         AT SEG',I5,'    TIME',F10.2)
C
C==================================================================
2400    FORMAT(' ? SIMCTR - BAD DRIVING SCHEDULE',/,
     2 '       SEG',I5,' IS UNDEFINED TYPE')
C
C==================================================================
3000    FORMAT(1H1/10X'VEHICLE PERFORMANCE SIMULATION'3XA9
       1,40X,5HPAGE ,I3,/,10X,42('*')///
       210X,'BREAKDOWN OF % TIME SPENT ON VARIOUS PARTS OF ENGINE MAP',///
       3,3X,11F8.1,/,1X,11(7(' '),'I'))
C
C REV TC-07-30-79 [     ] TO GIVE F6.1 ON TORQUE OUTPUT AND ADD ROOM FOR HP
C
3020    FORMAT(20('*'F6.0,'-+',10('--------+'),/,'*'
     2          11(7(' '),'I'),/,'*'
     3          7(' '),'I',10(F6.1,' I'),/,'*'
     4          7(' '),'I',10(F6.1,' I'),/,'*'
     4          7(' '),'I',10(F6.0,' I'),/,'*'
     5          7(' '),'I',10(F6.1,' I')/))
3040    FORMAT('+',F6.0,92('-'))
C
3060    FORMAT(1H1/10X'VEHICLE PERFORMANCE SIMULATION'3XA9
       1,40X,5HPAGE ,I3,/,10X,42('*')///
       210X,'BREAKDOWN OF % TIME SPENT ON VARIOUS PARTS OF ENGINE MAP',///
       3,3X,15F8.1,/,1X,15(7(' '),'I'))
C
C REV TC-07-30-79 [     ] TO GIVE F6.1 ON TORQUE OUTPUT AND AND HP
C
3080    FORMAT(20('*'F6.0,'-+',14('--------+'),/,'*'
     2          15(7(' '),'I'),/,'*'
     3          7(' '),'I',14(F6.1,' I'),/,'*'
     4          7(' '),'I',14(F6.1,' I'),/,'*'
     4          7(' '),'I',14(F6.0,' I'),/,'*'
     6          7(' '),'I',14(F6.1,' I')/))
3100    FORMAT('+',F6.0,114('-'))
        END
        SUBROUTINE SIMSTS(CHAR)
C
C       ENTRY POINTS:    SIMSTS
C
C       SUBROUTINES CALLED:     CTFLD,  EXIT,   KPTIME,
C                         INPPSE, TTYCLR, TTYSET
```

```
C
C       CALLED BY:        GCBACK, SIMCTR
C
C************************************************************************
C
C
        INCLUDE 'COMMS/NCLIST'
C
        INTEGER CHAR
        LOGICAL LIMESG
C
        DATA HISEG/'ISEG'/,HESEG/'ESEG'/,HESCD/'ESCD'/,ILL/0/
        1,LIMESG/.FALSE./,HERTE/'ERTE'/
C
        SVCHAR=CHAR                                                     !SAVE CHARACTER FROM TTY(HAVE OCTAL VALUE)>
        ISEGA=ISEG+NDSEG                                                !CALC DRS SEG #.
C
   10   GO TO ( 50, 50, 50,200,700, 50, 50,600
        1,        50, 50, 50, 50, 50,450,800,300
        2,        50, 50,100, 50, 50, 50, 50,820
        3,        50, 50, 50,400,500,550, 50, 50),CHAR
C
C    HERE ON ILLEGAL INTERUPT CHARACTER
        IF(SVCHAR.NE.CHAR) GO TO 50                                     !(HERE BEFORE THIS CALL?)YES, MUST BE ILL CHAR.
        CHAR=CHAR-64
        GO TO 10
C
   50   IF(ILL.GT.0) GO TO 900                                          !(1ST ILL CHAR?)NO, IGNORE.
        WRITE(JCT,1000)                                                 !YES % MESS ¬H HELP.
        ILL=1                                                           !FLG HERE BEFORE.
        GO TO 900                                                       !BYE.
C
C    HANDLE CONTROL-S
  100   CALL KPTIME(3)                                                  !UPDATE SIM RUN TIME.
        WRITE(JCT,1100) HISEG,ISEGA,CUMT,CUMD,V,CPUS,CPUT               !STATUS TO JCT.
        GO TO 900
C
C    HANDLE CONTROL-DEBUG TO JCT.
  200   CALL CTRLD('CTRID')
        GO TO 900
C
C    HANDLE CONTROL-P
  300   CALL INPPSE(ISEGA)                                              !CALL ¬P MODE ENTRY TO INPDIA.
        WRITE(5,1300)                                                   ![SIM CONT] TO JCT.
        GO TO 900
C
C    HANDLE SIMCTR END OF SEGMENT CALL
  400   IF(LIMESG) GO TO 900                                            !NO OUTPUT ESEG FLG ON?)YES, BYE.
        CALL KPTIME(3)                                                  !UPDATE SIM RUN TIME.
        WRITE(JCT,1100) HESEG,ISEGA,CUMT,CUMD,V,CPUS,CPUT               !ESEG MESS TO JCT.
        GO TO 900                                                       !BYE.
  450   LIMESG=.NOT.LIMESG                                              !FLIP NO ESEG FLG.
        GO TO 900                                                       !BYE.
C
C    HANDLE SIMCTR END OF DRIVING SCHEDULE CALL
  500   WORDT=HESCD                                                     !SET TO END DRS.
        PNAME=DNAME                                                     !GET DRS NAME.
  510   CALL KPTIME(3)                                                  !UPDATE SIM RUN TIME.
        CALL TTYSET                                                     !RESET ¬O.
        WRITE(JCT,1500) WORDT,PNAME,CUMT,CUMD,CUMPE,CPUT                !TO JCT.
        GO TO 900                                                       !BYE.
  550     WORDT=WRDTE                                                                   !GET RTE NAME.
          PNAME=RNAME
          GO TO 510
C
C    HANDLE CONTROL-H
  600     WRITE(JCT,1600)                                                               !OUTPUT INTERRUPT(RUN TIME) COMMANDS.
          GO TO 900                                                                     !BYE.
C
C    HANDLE CONTROL-E
  700     CALL TTYCLR
          WRITE(5,1700)                                                                 !"ARE YOU SURE?"
          READ(5,1701) ANS                                                              !GET ANSWER.
          IF(ANS.EQ.'Y') CALL EXIT                                                      !(ANS YES?) YES, GOOD BYE FOR THIS RUN.
          GO TO 900                                                                     !BYE.
C
C  HANDLE O INTERUPT
  800     LINTTY=.NOT. LINTTY                                                           !FLOP INTERNAL ¬O FLG.
          WRITE(5,1800)                                                                 !"<CR-LF>"
          GO TO 900
C
C...      HANDLE CONTROL-X (ENTER DDT IF LOADED)
C
 820      CALL GETDDT
          GO TO 900
C
  900     RETURN                                                                        !DONE, BYE.
C
C    FORMAT STATEMENTS
C
 1000     FORMAT(' % SIMSYS-ILLEGAL INTERUPT CHARACTER.'
          1,' TYPE ¬H FOR HELP'/)
 1100     FORMAT(1XA4':'I3' CUMT:'F8.2' CUMD:'F8.3' MPH:'F5.1
          1,' CPUS:'F6.2' CPUT:'F8.2/)
 1300     FORMAT(/' [SIMULATION CONTINUING]')
 1500     FORMAT(1XA4':'A10'  CUMT:'F8.2'  CUMD:'F8.3'  MPG:'F5.1
          1,'  CPUT:'F6.2/)
 1600     FORMAT(/' VEHSIM INTERUPT CONTROL CHARACTERS'
          1,/'    ¬D - TYPE OUT DEBUG INFORMATION'
          3,/'    ¬E - CLOSE ALL FILES AND EXIT TO MONITOR'
          2,/'    ¬H - TYPE OUT THIS HELP MESSAGE'
          3,/'    ¬N - SUPERESS END OF DRIVING SEGMENT MESSAGES'
          3,/'    ¬P - PAUSE TO MODIFY AND THEN CONTINUE SIMULATION'
          4,/'    ¬S - TYPE OUT CURRENT STATUS OF SIMULATION'/
          5,/'     O - TURN OFF SIMULATION OUTPUT EXCEPT FOR ERROR'/
          6,/'    ¬X - ENTER DDT IF LOADED'/)
 1700     FORMAT(' ARE YOU SURE? '$)
 1701     FORMAT(A1)
 1800     FORMAT(/$)
C
          END
          SUBROUTINE SLOOKP(WORD,NTAB,TABLE,POSIT,*)
C
C         ROUTINE TO DO A TABLE LOOKUP
C
          IMPLICIT INTEGER (A-Z)
C
          DIMENSION MASK(5),TABLE(1)
          DATA MASK/"774000000000,"777760000000,"777777700000
          2,"777777777400,-1/
C
          FOUND=.FALSE.                                                                 !ASSUME NO MATCH.
```

```
         NCHS=ICRCNT(WORD,1)                                                  !GET # OF CHARS IN WORD.
         TMASK=MASK(NCHS)                                                     !GET CORRECT MASK.
         MWORD=WORD.AND.TMASK                                                 !MASK IT.
C
         DO 20 IT=1,NTAB                                                      !LOOP THRU ALL ENTRIES IN TABLE.
         MTABLE=TABLE(IT).AND.TMASK                                           !MASK TABLE ENTRY.
         IF(MWORD.NE.MTABLE)GO TO 20                                          !(MATCH?)NO.
         IF(FOUND)GO TO 60                                                    !YES. (SECOND MATCH?)YES.
         FOUND=.TRUE.                                                         !NO, SET MATCH FLAG.
         POSIT=IT                                                             !SAVE INDEX OF MATCH.
20       CONTINUE
C
         IF(FOUND)RETURN 1                                                    !(GOT A MATCH?)YES.
C
         WRITE(5,40)MWORD                                                     !REPORT NO MATCH.
40       FORMAT(' %',A5,' illegal command')
         RETURN
C
60       WRITE(5,80)MWORD                                                     !REPORT 2 MATCHS.
80       FORMAT(' %',A5,' ambiguous command')
         RETURN
         END
         SUBROUTINE VALID2(PNAME,*)
C
C        ENTRY POINTS:   VALID2
C
C        SUBROUTINES CALLED:     ICRCNT, IGET,   PUT
C
C        CALLED BY:      INPEAT
C
C**********************************************************************
C
         DOUBLE PRECISION PNAME,POINT,BLANK,HQ
         LOGICAL BATCH,DIALOG,PTY,TTY
         COMMON /MODE/BATCH,DIALOG,PTY,TTY
         DATA JCT/5/,DBLANK/' '/, PTR/'¬'/, HQ/'?'/
C
C
C
C
20       N=ICRCNT(PNAME,2)                                                    !GET NUMBER OF CHARACTERS IN NAME
         IF(N.EQ.0)GO TO 60
         DO 40 I=1,N
         IT=I
         CALL IGET(PNAME,IT,CHAR)                                             !GET A CHARACTER
C
C                                            TEST FOR LETTERSS OR NUMBERS
C
         IF((CHAR.GE.'A' .AND. CHAR.LE.'Z') .OR.
         1         (CHAR.GE.'0' .AND. CHAR.LE.'9')
         2         .OR. (CHAR.NE.'.') .OR. (CHAR.NE.'-'))GO TO 40
         GO TO 80                                                             !ILLEGAL. GO TELL.
40       CONTINUE
         RETURN                                                               !LEGAL. RETURN
60       I=1
80       POINT=DBLANK                                                         !BLANK OUT WORD
         CALL PUT(POINT,I,PTR)                                                !PUT IN ARROW
         WRITE(JCT,1020)PNAME,POINT
         IF(PTY)RETURN 1
100      WRITE(JCT,1040)
         READ(JCT,1060)PNAME
```

```
      IF(FNAME.EQ.DBLANK) GO TO 100
      IF(FNAME.EQ.HQ) RETURN 1
      GO TO 20
C
C
1020  FORMAT(/,' ? VALID - ILLEGAL CHARACTER IN PART NAME.'/
     1      //5X,A10/5XX,A10//)
1040  FORMAT(' ENTER NEW PART NAME OR ? TO SKIP TO NEXT COMMAND: '$)
1060  FORMAT(A10)
      END
      BLOCK DATA VSBBLK
C
C
      INCLUDE 'COMMS'
C
C
      DATA HBLANK/' '/,CSTAR/'*'/,HSTAR/'*'/,DBLANK/' '/
      DATA HUP/'UP'/,HDN/'DN'/,RUN/'UN'/,HQ/'?'/
      DATA SEQIN/'SEQIN'/,SEQOUT/'SEQOUT'/,SEQIO/'SEQINOUT'/
      DATA APPEND/'APPEND'/,DELETE/'DELETE'/,BINARY/'BINARY'/
      DATA RENAME/'RENAME'/
      DATA SNGLES/.FALSE./,MASDEV/'DSKB  '/,MASPPN/"4132,"402/
     1,(MASFIL(I),I=1,12)/'VSMENG.BIN','VSMTCV.BIN','VSMVEH.BIN'
     3,'VSMGRS.BIN','VSMACC.BIN','VSMDRS.BIN','VSBSLG.BIN'
     4,'VSMRCO.BIN','VSMTIR.BIN','VSMTRA.BIN','VSMAXL.BIN'
     5,'VSMEAS.BIN'/
C
      DATA IDEBUG/1/,IDBUGO/1/
C
      DATA CTRFIL/'VSMCTR.DAT'/,LPTFIL/'000VSM.DAT'/
     1,CTRPPN(3)/0/,LPTPPN(3)/0/,SCRPPN(3)/0/,LPTPRO/"100/
C
      DATA (BASFIL(I),I=1,12)/'VSMENG.BIN','VSMTCV.BIN','VSMVEH.BIN'
     2,'VSMGRS.BIN','VSMACC.BIN','VSMDRS.BIN','VSBSLG.BIN'
     3,'VSMRCO.BIN','VSMTIR.BIN'
     4,'VSMTRA.BIN','VSMAXL.BIN'
     6,'VSMEAS.BIN'/
C
      DATA BASDEV/15*'DSKB      '/
C
      DATA DATPEN/15*"000000004132000000000402/
      DATA LPTDSP/'APPEND'/,BATCH/.TRUE./,DIALOG/.FALSE./
     1,NCSEG/0/
C
C**********************************************************************
C
C
      DATA BB1/0.104720/,BB2/1.50404E-4/
     1,MASK/"774000000000,"777760000000,"777777700000,"777777777400
     2,"777777777776/,LWMASK(10)/"7777777777767777777777776/
      DATA NOPPAR/NPAR1/
C
      END
      SUBROUTINE VSPCTR
C
C     ENTRY POINTS:   CLOSE,  RENTER, VSMCTR
C
C     SUBROUTINES CALLED:     CHKFIL, DSKCTR, EXIT,   FILSPC,
C                     INPPAT, INPCIA, JCPEPN, TTYSTS, ZERCMD
C
C     CALLED BY:      VEHSIM
```

```
C
C*****************************************************************
C
C
      EXTERNAL RESETM
      DOUBLE PRECISION BASFIL,LPTFIL,CTRFIL,DBLANK,LPTDSP
     2,SCRDEV,SCRFIL,MASFIL,JOBDEV,CTRDEV
     3,BASDEV,MASDEV,LPTDEV,DATE,FILNAM,DISP,LPTORG,DELETE
     4,RENAME,APPEND,LPTACC,SAVE,SEQOUT,SEQIN,PNAME,FSPECS
     5,IERGNA
C
      DIMENSION IDEV(2)
C
      LOGICAL SNGLBS,BATCH,ENDE,NLSTCF,DIALOG,PTY,TTY,LPTOPN,CTROPN
     1,LOCKUP,LAPEND,LINTTY,LLPT
C
      REAL LPTLEN
C
      EQUIVALENCE (IDEV(1),JOBDEV)
C
      COMMON /TTYOUT/ LINTTY,JCTWD,LLPT
      COMMON /GET/ OT
      COMMON /INPCOM/ SIMCDE,FNAME,WORD,DUMMY(124),FSPECS(9)
      COMMON /MODE/ BATCH,DIALOG,PTY,TTY
      COMMON /RUNID/ TITLE(12),VERID(3)
      COMMON /FILES/ JOBNUM,JOBDEV,PPNJOB(2),SNGLBS,MASDEV,MASFIL(15)
     1,MASPPN(2),BASDEV(15),BASFIL(15),BASPPN(2,15)
     2,CTRDEV,CTRFIL,CTRPPN(3)
     3,LPTDEV,LPTFIL,LPTPPN(3),LPTDSP
     4,SCRDEV,SCRFIL,SCRPPN(3),LPTPRO
      COMMON /V2MISC/ LOCKUP(20),DATE
C
      DATA DBLANK/' '/,HZERO/'00000'/
     1,NLSTCF/.FALSE./,SCRFIL/'000VSM.SCR'/,NZ/1/,IDEV/-1,0/
     2,ENDE/.FALSE./,BELL/"034000000000/,BLANK/' '/
     3,APPEND/'APPEND'/,RENAME/'RENAME'/,SAVE/'SAVE'/,SEQOUT/'SEQOUT'/
     4,SEQIN/'SEQIN'/,LAPEND/.FALSE./,DELETE/'DELETE'/,LLPT/.TRUE./
C
C*****************************************************************
C
      CALL SETC(RESETM)
      CALL ERRSET(0)
      CALL GETNAM(IERGNA)
C
C     VSMCTR(VEHSIM CONTROL) DOES DISK PROGRAM MODE & FILE CONTROL ONLY
C
C   INTIALIZE
C
      OT=BLANK                                                   !FLG ZERO ALL.
      CALL ZERCOM(INPCOM)                                        !INITIALIZE VEHSIM.
      IF(NZ.EQ.0) GO TO  45                                      !(RESTART?) YES.
      TTY=.TRUE.                                                 !ASSUME JCT IS TTY.
      CALL TTYSTS(PTY,TTYNUM,JCTWD)                              !GET TTY STATUS (PTY T OR F) & TTY#.
      IF(PTY) TTY=.FALSE.                                        !(PSEUDO TTY I.E. BATCON) YES SET TTY MODE.
      IF(TTY) DIALOG=.TRUE.                                      !(TTY)SET TO DIALOG MODE.
C
C    GENERATE JOB NUMBER DEPENDENT FILE NAMES
C
      IDEV(1)=-1                                                 !FLG JOBPPN FOR 1ST STRUCTURE IN JOB SEARCH LIST.
      CALL JOBFEN(JOBDEV,JOBNUM,PPNJOB(1),PPNJOB(2))             !GET JOB INFO.
      ENCODE(3,1003,TEMP) JOBNUM                                 !CONVERT JOBNUM TO ASCII.
```

```
        IF(JOBNUM.GT.99) GO TO 40                                 !(3 SIGNIFICANT DIGITS IN JOBNUM?) YES.
        IF(JOBNUM.LE.9) NZ=2                                      !NO, (1 SIGNIFICANT DIGIT IN JOBNUM?) YES.
        ENCODE(NZ,1005,TEMP) NZERO                                !NO, LEFT FILL ZERO'S IN ASCII JOBNUM.
   40   ENCODE(3,1005,LPTFIL) TEMP                                !ADD JOB NUM TO LPT FILENAME.
        ENCODE(3,1005,SCRFIL) TEMP                                !ADD JOB NUM TO SCRATCH FILE NAME.
        NZ=0                                                      !FLG 0 ONE IN CASE OF RESTART.
C
45      LPTDEV=JOBDEV                                             !SET DEFAULT LPT FILE STRUCTURE.
        LPTCBG=LPTFIL
        LPTPPN(1)=PPNJOB(1)
        LPTPPN(2)=PPNJOB(2)
        LPTCEN=.FALSE.
        LPTCSP=CBLANK
C
        CTRDEV=JOBDEV                                             !SET DEFAULT BATCH CONTROL FILE STRUCTURE.
        CTRPPN(1)=PPNJOB(1)
        CTRPPN(2)=PPNJOB(2)
        CTRCEN=.FALSE.
C
        SCRDEV=JOBDEV                                             !SET SCRATCH FILE STRUCTURE.
        SCRPPN(1)=PPNJOB(1)
        SCRPPN(2)=PPNJOB(2)
C
C   REENTER ENTRY POINT
C
        ENTRY RENTER
C
   50   CALL TIME(HOUR)                                           !GET TIME "HH:MM".
        WRITE(5,1000)IPRGNA, VERID,DATE,HOUR,JOBNUM,TTYNUM,JOBDEV,PPNJOB     !REPORT JOB INFO TO TTY.
C
        CALL ERRSET(0)
C PROMPT FOR COMMAND INPUT
  100   IF(DIALOG)GO TO 110
        IF(TTY) WRITE(5,1007) BELL                                !(TTY?) YES.
  101   WRITE(5,1001)                                             !PROMPT "*"
        READ (5,1002) CCMND,FSPECS                                !READ COMMAND.
        IF((COMND.AND."774000000000) .NE. ('@'.AND."774000000000))
       2GO TO 103
        REREAD 1030,FSPECS
        GO TO 117
C
103     FILNAM=FSPECS(1)
        DLSTCE=FSPECS(2)
C
        IF(CCMND.EQ.BLANK) GO TO 101                              !(EXTRANEOUS "CR"?) YES, GO PROMPT.
        IF(CCMND.EQ.'BATCH') GO TO 115                            !(BATCH?) YES.
        IF(CCMND.EQ.'DIALO') GO TO 110                            !(DIALOGUE?) YES.
        IF(CCMND.EQ.'STOP' .OR. CCMND.EQ.'EXIT') CALL EXIT        !(STOP OR EXIT VEHSIN RUN COMPLETE?) YES.
        IF(CCMND.EQ.'CONTI') GO TO 105                            !(CONTINUE EXECUTION OF CONTROL FILE?) YES.
        WRITE(5,1006) CCMND                                       !?COMND? UNKNOWN.
        GO TO 100                                                 !GO PROMPT.
C
C  *CONTINUE
  105   IF(CTROPN) GO TO 107                                      !(CONTROL FILE OPEN?) YES.
        WRITE(5,1008)                                             !NO, ERR REPORT.
        IF(DIALOG.AND.IECCNE.EQ.1) GO TO 210                      !(DIALOG &       ?).
        GO TO 100                                                 !GO PROMPT.
  107   MCTR=1                                                    !FLG INPUT FOR *CONTINUE.
  109   BATCH=.TRUE.                                              !SET BATCH MODE.
        DIALOG=.FALSE.
        IF(MCTR.EQ.1)GO TO 200                                    !*CONTINUE COMMAND
       CALL FILSEC(FSPECS,CTRDEV,FILNAM,CTRPPN)
       CALL CHKFIL(CTRDEV,FILNAM,CTRPPN,$200)
       WRITE(5,1020)CTRDEV,FILNAM,CTRPPN
C
C  *DIALOGUE
  110  BATCH=.FALSE.                                                    !SET DIALOGUE MODE.
       DIALOG=.TRUE.
       GO TO 200                                                        !GO CHECK LPT OPEN.
C
C  *BATCH
  115  IF(FILNAM.NE.'REWIND') GO TO 117                                 !(REWIND CTR FIL & PROCESS?) NO.
       MCTR=2                                                           !YES, SET REWIND FLG TO INPBAT.
       GO TO 109                                                        !GO SET BATCH MODE.
  117  MCTR=1                                                           !SET NEW CTR FIL FLG TO INPBAT.
       IF(.NOT.CTROPN) GO TO 109                                        !(CTR FILE OPEN?) NO, GO SET BATCH MODE.
       CALL RELEAS(4)                                                   !YES, RELEASE IT.
       CTROPN=.FALSE.                                                   !SET FLG CTR FIL NOT OPEN.
       GO TO 109                                                        !GO SET BATCH MODE.
C
C  MAKE SURE LPT FILE OPEN
       ENTRY   OPNLPT(ITASK)
  200  IF(LPTOPN) GO TO 205                                             !(LPT FILE OPEN?) YES, SKIP.
       LPTACC=SEQOUT                                                    !ASSUME NEW FILE FOR LPT.
       IF(LAPEND) LPTACC=APPEND                                         !(APPEND OLD LPT FILE?) YES, SET FLG.
       OPEN(UNIT=6,DEVICE=LPTDEV,ACCESS=LPTACC,FILE=LPTFIL              !OPEN LPT FILE.
       1,DIRECTORY=LPTPPN,PROTECTION=LPTPRO)
C
       LPTOPN=.TRUE.                                                    !SET FLG LPT OPEN..
       IF(.NOT.ITASK)GO TO 202
       ITASK=.FALSE.
       RETURN
C
202    CALL TIME(HOUR)                                                  !GET TIME "HH:MM".
       WRITE(6,1000)IPRGNA,VERID,DATE,HOUR,JOBNUM,TTYNUM,JOBDEV,PPNJOB !REPORT JOB INFO TO LPT.
       IF(.NOT.LAPEND) GO TO 205                                        !(APPENDING OLD LPT FILE?) NO.
       LAPEND=.FALSE.                                                   !RESET LPT APPEND FLG..
       WRITE(5,1013) LPTDEV,LPTFIL,LPTPPN(1),LPTPPN(2)                  !"% APPENDING OUTPUT TO LPT FILE:"
C
C  HANDLE DIALOGUE MODE
  205  IF(ITASK)RETURN
       IF(BATCH) GO TO 215                                              !(VEHSIM CONTROL FILE?) YES.
  210  CALL INPDIA  ( IECOND , ENDE )                                   !NO CALL DIALOGUE ROUTINE.
       GO TO (115,115,107),IECOND                                       !(BATCH/FUTURE USE/CONTINUE?) YES.
       GO TO 100                                                        !NO, GO PROMPT.
C
C  HANDLE BATCH MODE
C
C    MAKE SURE VEHSIM CONTROL FILE IS OPEN.
C
  215  IF(CTROPN) GO TO 218                                             !(VEHSIM CONTROL FILE OPEN?) YES, SKIP.
       IF(MCTR.NE.2) CTRFIL=FILNAM                                      !(*REWIND?) NO, GET VEHSIM CTR FIL NAME TO OPEN.
       OPEN(UNIT=4,DEVICE=CTRDEV,ACCESS=SEQIN,FILE=CTRFIL               !OPN VEHSIM CNTRL FIL.
       1,DIRECTORY=CTRPPN)                                              !EFFECTIVE AS REWIND IF SAME FIL NAME.
       CTROPN=.TRUE.                                                    !FLG VEHSIM CONTROL FILE OPEN.
       NLSTCF=.FALSE.                                                   !ASSUME LIST CONTROL FILE ON LPT.
       IF(LLSTCF.EQ.BLANK) NLSTCF=.TRUE.                                !(LIST?) NO. SET FLG.
       WRITE(6,1004) CTRDEV,CTRFIL,CTRPPN(1),CTRPPN(2)                  !REPORT CONTROL FILE INFO TO LPT.
C
  218  IECOND=MCTR                                                      !PASS TASK FLG TO INPBAT.
219    CALL INPBAT  ( IECOND , ENDE , NLSTCF )                          !GO PROCESS VEHSIM CONTROL FILE AS REQUESTED.
       IF(.NOT. ENDE) GO TO 225                                         !(EOF ON CTR FIL?) NO.
```

```
        CLOSE(UNIT=4)                                        !RELEASE CONTROL FILE.
        CTRCPN=.FALSE.                                       !SET FLG CONTROL FILE NOT OPEN.
C
  225   IF(DIALCG) GO TO 210                                 !(DIALOGUE COMMAND FROM CTR FIL?) YES, GO INPDIA.
        IF(.NOT.CTRCPN)GO TO 100                             !IF CONTROL FILE NOT OPEN, GO PROMPT
        MCTR=3                                               !SET FLAG TO CONTINUE CONTROL FILE
        GO TO 218                                            !GO CONTINUE.
C
C*****************************************************************************
C
        ENTRY CLOSE                                          !CLOSE ALL OPEN I/O UNITS EXCEPT JCT
C
        IF(.NOT.CTRCPN) GO TO 320                            !(CTR FIL OPEN?) NO.
        CALL RELEAS(4)                                       !YES, RELEASE IT.
        CTRCPN=.FALSE.                                       !SET FLG CTR NOT OPEN.
  320   CONTINUE
C
        ENTRY CLSLPT(ITASK)
C
        IF(.NOT. LPTOPN) GO TO 330                           !(LPT FIL OPEN?) NO.
        IF(LPTDSP.EQ.DBLANK) LPTDSP=APPEND                   !YES, (LPT DISP GIVEN?) NO, DEFAULT TO "SAVE".
        IF(ETY) GOTO 323
        WRITE(5,1011) LPTFIL                                 !REPORT LPTFIL NAME, DISPOSE:?.
        READ(5,1010) DISP                                    !ANS IS DISP.
        IF(DISP.NE.DBLANK) LPTDSP=DISP                       !(ANY ANS?) YES - GET ANS. NO - USE CURRENT DISPOSE.
  323   IF(LPTDSP.NE.APPEND) GO TO 324                       !LPT DISP=APPEND?) NO.
        LPTDSP=SAVE                                          !YES, SAVE LPT FIL.
        LAPEND=.TRUE.                                        !SET FLG SO LPT FIL WILL BE APPENDED TO ON REENTER.
        GO TO 325                                            !GO CLOSE LPT.
  324   IF(LPTDSP.NE.RENAME) GO TO 325                       !(RENAME LPT FIL?) NO. GO CLOSE.
  327   WRITE(5,1012)                                        !LPT FIL RENAME=?.
        READ(5,1031) FSPECS                                  !GET ANS.
        IF(FSPECS(1).EQ.DBLANK) GO TO 327                    !(ANY ANS?) NO, GO ASK AGAIN.
        CALL FILSEC(FSPECS,LPTDEV,LPTFIL,LPTPPN)
  325   WRITE(6,1009) LPTDEV,LPTFIL,LPTPPN(1),LPTPPN(2),LPTPRO,LPTDSP   !REPORT LPT DISP TO LPT.
        CLOSE(UNIT=6,DEVICE=LPTDEV,FILE=LPTFIL,DISPOSE=LPTDSP
       2,DIRECTORY=LPTPPN,PROTECTION=LPTPRC,ERR=3250)
        GO TO 3255
 3250   CALL ERRSNS(ITERP,JTEMP)
        WRITE(5,1035)LPTDEV,LPTFIL,LPTPPN(1),LPTPPN(2)
        GO TO 327
C
 3255   WRITE(5,1009) LPTDEV,LPTFIL,LPTPPN(1),LPTPPN(2),LPTPRO,LPTDSP   !REPORT CLOSE TO TTY.
        LPTCPN=.FALSE.                                       !SET FILG LPT CLOSED.
CSIL    IF(.NOT.ITASK)LPTFIL=LPTORG                          !RESET LPT FIL NAME.   - 6/13/79 TAKE THIS OUT TO SEE IF
C                                          !IT FIXES THE RESET CHANGING THE NAME ALREADY CHENGED BY THE "OUTPUT".

        LPTDSP=DBLANK                                        !DE INITIALIZE
        LPTPPN(1)=PENJOE(1)
        LPTPPN(2)=PENJOE(2)
  330   IF(.NOT.ITASK)GO TO 332
        ITASK=0
        RETURN
C
 332    CALL DSKCTR(0,'VSPCTR')                              !FLG DSKCTR TO CLOSE ALL IT'S FILES.
        RETURN                                               !*DONE ,BYE.
C
C*****************************************************************************
C
C   FORMAT STATEMENTS
C
```

```
 1000   FORMAT(1X,A6,1XA5,1XO2'('C2') 'A10,2XA5' JOB'I3' TTY'O3
        1,2XA6':['O5,',''C3']')
 1001   FORMAT(' *'$)
 1002   FORMAT(A5,1X10A10)
 1003   FORMAT(I3)
 1004   FORMAT('1 VEHSIM CONTROL FILE 'A6':'A10'['O5','O3']'/)
 1005   FORMAT(1A3)
 1006   FORMAT(' ?'A5'?')
 1007   FORMAT(1XA1$)
 1008   FORMAT(/' ? VSMCTR-NO VEHSIM CONTROL FILE CURRENTLY OPEN TO '
        1,'CONTINUE.')
 1009   FORMAT(/' LPT FILE 'A6':'A10'['O5','O3']<',O3,'>/DISPOSE:'A10/)
 1010   FORMAT(A10)
 1011   FORMAT(/' LPT FILE 'A10'/DISP: '$)
 1012   FORMAT(/' NEW LPT FILE NAME = '$)
 1013   FORMAT(/' % VSMCTR-APPENDING LPT OUTPUT TO FILE 'A6':'A10'['O5
        1,',','O3']')
 1020   FORMAT(' ? VEHSIM CONTROL FILE 'A6':'A10'['O5','O3'] NOT FOUND'/
        2,       ' SWITCHING TO DIALOGUE MODE')
1030    FORMAT(1X,10A10)
1031    FORMAT(10A10)
1035    FORMAT(' ?VSMCTR - Cannct rename output file to '
        2,A5,':',A10,'['O4,',',C3,']')
        END
        SUBROUTINE ZERO
C
C       ENTRY POINTS:    ZERO
C
C       CALLED BY:       INPDAT
C
C**************************************************************************
C
C
        LOGICAL  LPRNT,LIMPRN,MILIM,SECLIM,ENDLIM,PREM,PPS,PTOR,PBMEP,
        1 PBE,PLEHR,PESFC,PGALHP,LGFREE,LPSHF,LOCKOP,LTRRZ,LVNEW,LDYNA
C
        DOUBLE PRECISION   ANAME,CNAME,DNAME,ENAME,RNAME,SNAME,TNAME,
        1                   VNAME,BN,UN,CDATE,GNAME,HNLOAD
C
        COMMON /DYNAM/ LDYNA,DYN
        COMMON /INPCOM/ SIMODE
        COMMON /TTYCUT/ LIMTTY
        COMMON /ACCESS/  NACC,ACCT(20,20),ACCS(20,20),TORQA,ANAME(20)
        1,AIAS(20)
        COMMON /CNTRL/   IC,TOLD,VOLD,T,V,ACCEL
        COMMON /CONST/   FRC1,FRC2,FAC,CD,AREA,VWIND,WGT,FGC,WRAD,RAR,
        1         GRAT(20),NUMG,NGEAR,AIN,AIP,AI2,ERAT(20),ERAR(2),
        2AIE,AID,AI1,EEFR,CCC,PHI,PSI,AIGIN(20),AIGOUT(20),WLSG,LTRRZ
        3,NGFLSS(20),GREM(20,20),GRTCFC(20,20),GNAME(20),GCOM(16)
        COMMON /CURVFT/  X(100),Y(100),NPTS,COEF(4)
        COMMON /DISK/ EN,ET,NREC,DUMCOM(16),NDISK,MLTSEC,CDATE,CHOUR
        COMMON /DSCHED/  DNAME,DCOM(16),T0,V0,D0,A0,NGO,NSEG,ASEG(50),
        1                 VSEG(50),PWCT(50),ATHOLD(50),NGSEG(50),
        2                 THRATE(50),TSEG(50),DSEG(50),PCSEG(50),
        3                 POSTSE(50),VELSEG(50),ITISEG(50)
        4,LSTSEC,NDSEG,NSEC
        COMMON /ENDCF/   ENDE,IERNT
        COMMON /ENGMAP/  EPMAX(2),RPMIN(2),NRPM(2),REME,TORQE,FRATE,VAC,
        1                 THR,MAPOK,IERRE,NTOR(2,20),EMAP(20,20,8),
        2                 ERPM(2,20),EMMIN(2),SPIDLE(2)
        COMMON /GET/     OT,UN,NUG(20),JENG(20),IENG
```

```
      COMMON /IC/ ECOM(16),ENAME(2),DISP,ICYL,IMIN,IMAX,THRMAX,
     1             THRMIN,FINER,BORE,STROKE,FSPGR,NCYCLE
      COMMON /LCIPFR/  LFFM,PFFM,LPS,PPS,LTOR,PTOR,LBMEP,PBMEP,LHP,PHP,
     1                  LLBHR,PLBHR,LBSFC,PBSFC,LGALHR,PGALHR,LPRNT
      COMMON /MISC/    XNA(20),TIREFF
      COMMON /CIDIO/   CDISP,CBORE,CSTRCK,IOCYL
      COMMON /CIDMAP/  FMAPO(20,20,4),ERPMO(20),NTOBO(20),NRPMO
      COMMON /RTE/     FNAME,NRDIST,RDIST(10),RGRADE(10),RCOM(16),
     1                  RCOEF(10),RVWINC(10),LSTRTE,NDRTE,NRTE
      COMMON /PRNLIM/  LIMPRN,MILIM,SECLIM,ENDLIM,ALIMN,LSCALE
      COMMON /RUNID/   TITLE(12)
      COMMON /SHPTL/   SNAME,SCOM(16),GCVPSI(4),OUTRPM(4),NGPT,IGP(38),
     1                  IGT(38),SHPTIM(38),LGFREE(38),LPSHF(38),
     2                  EFFST(38),AERUPT(38),SHFTPT(10,38),SHFTSP(10,38)
     3                  ,LVAC,LENG,GDAT,NSPTS(38),LDETNT,DETPT(38),
     4                  DETRPM(38),PARAB,LDETE,LDETV,GOVLIM,LDTH
      COMMON /TCRCON/  TORBPK,TCRQ2,RPM2,COAST,SR,TR,TRD(20),SRD(20),
     1                  AKD(20),NTD,SRC(20),AKC(20),NTC,NTBP,TIN(20),
     2                  TCUT(20),SEIN(20),SCUT(20),NTORP,CDIAM,CNAME,
     3                  CCOM(16),CCNTOR
      COMMON /V2MISC/ LCCKUP(20),SDATE(2),NPARTS(11),DEPDT,NORUN,NENG,
     1          ICNIT,IPART,IPMCDE,IRECDE,ISMODE,NSGEAR(20,2),NXTPG
     3          ,NUMPAR
      COMMON /VEHICL/ VNAME,VCOM(16),TNAME,TCOM(16),ACOM(16)
     1,NRAX,NPAX(2),AXRPM(20,2),AXTORQ(20,2),LVNEW
      COMMON /CDEBUG/  IDEBUG,DEBGIN,DSTOP,ISEG1,ISEG2,ISEG,CUMT,CUMD
C
      DATA HBLANK/' '/,IFIRST/0/,HNLOAD/'NOT LOADED'/,HCAR/'CAR'/
C
C*******************************************************************
C
      IUNIT  = 6                                                         !LPT UNIT 6.
      IPMCDE = 1
      IDEBUG = 1
      IRECDE = 1
      ISMCDE = 1
      NENG   = 0
      NOFON  = 0
      NUMG   = 0
      VWIND  = 0.
      NACC   = 0
      DEFLT  = .05
C
      SIMCDE=HCAR                                                        !SIMULATION MODE DEFAULT.
      ENAME(1)=HNLOAD                                                    !SET PART NAMES TO NOT LOADED.
      ENAME(2)=HNLOAD                                                    !   FOR /*STATUS/.
      VNAME=HNLOAD
      DNAME=HNLOAD
      SNAME=HNLOAD
      RNAME=HNLOAD
      TNAME=HNLOAD
C
      LDYNA=.FALSE.                                                      !NOT DYNA SIM.
      LTRRZ=.FALSE.                                                      !CALC TRR.
      LIMTTY=.TRUE.                                                      !TTY OFF.
      LPRAT  = .TRUE.                                                    !PRINT FLAGS.
      LIMIRN = .TRUE.
      LVRMAX=.TRUE.
      LVNEW=.TRUE.                                                       !MORE PRINT FLGS.
      MILIM  = .FALSE.
      SECLIM = .FALSE.
      ENDLIN = .FALSE.
      PRPM   = .FALSE.                                                  !ENG UNITS FLGS.
      PPS    = .FALSE.
      PTOR   = .FALSE.
      PBMEP  = .FALSE.
      PHP    = .FALSE.
      PLBHR  = .FALSE.
      PBSEC  = .FALSE.
      PGALHR = .FALSE.
C
      DO 100  I = 1,38
      LGEFFE(I) = .TRUE.                                                !SHIFT LOGIC FLGS.
      LPSBF (I) = .TRUE.
      IF ( I.GT.20 )  GO TO 100
      JENG(I)    = 0                                                    !ENG TO GEAR ASSIGNMENTS.
      LOCKUP(I) = .FALSE.                                               !ASSUME GEARS NOT LOCKED UP.
      IF ( I.GT. NUMPAR )  GO TO 100
      NPARTS(I) = 0                                                     !PARTS LODED FLGS.
  100 CONTINUE
C
      DO 120  I = 1,12
  120 TITLE(I)  = HBLANK                                                !BLANK RUN TITLE.
C
      IF(IFIRST.GT.0) GO TO 959                                         !(1ST CALL?) NO.
C
      CALL DATE(SDATE)                                                  !YES,GET DATE FROM SYSTEM
      IFIRST=1                                                          !FLG HERE BEFORE.
C
  959 RETURN                                                            !DONE, BYE.
C
      END
      SUBROUTINE OVRDRV
C
C     ENTRY POINTS: OVRDRV
C
C     CALLED BY: GOBACK
C
C*********************************************************************
C
C     USED FOR THE SPLIT TORQUE CONVERTER IN RESPONSE TO THE
C     OVERDRIVE TRANSMISSION
C
C     DICTIONARY OF VARIABLES USED IN THIS ROUTINE
C
C     RPM2  - INPUT VARIABLE INTO SPLIT TORQUE SEGMENT
C     TCRC2 - INPUT VARIABLE INTO SPLIT TORQUE SEGMENT
C     ATSIN - CONTRACTION COEFFICIENT OF INCREMENT FOR
C             CONVERTER SPEED
C     A1    - COEFFICIENT PROPORTIONAL TO THE NUMBER OF
C             TEETH ON RING GEAR OF SPLIT TORQUE PLANETARY
C             MECHANISM
C     A2    - COEFFICIENT PROPORTIONAL TO THE NUMBER OF
C             TEETH ON RING GEAR OF SPLIT TORQUE PLANETARY
C             MECHANISM
C     DVV   - TOLERANCE OF DEVIATION OF CALCULATED RPM2
C             FROM INPUT VARIABLE
C     DLL   - THEORETICAL INFINITY, TAKEN FOR COMPARISON
C     DL    - CURRENT DEVIATION OF CALCULATED RPM2 FROM THE
C             INPUT VARIABLE
C
      INCLUDE 'COMMS/ACLIST'
```

```
C
C
C         INITIALIZE VARIABLES
          A1=1.0
          A2=2.4
          DVV=5.0
          DILRPM=100.0
          RPMI=0.0
          RPMR=0.0
          DLL=0.0
          DLR=0.0
          IPASS=0
          IF(RPM2.GT.1.0) GC TO 9
C
C         IF IDLE SET TC LOWEST SPEED RATIO
C         IDLE MODE
          SR=SRD(1)
          TR=TRD(1)
          GO TO 90
9         CONTINUE
          TR=1.0
          TORQ11=(A1/A2)*TORQ2
          RPM11=RPM2
110       RAT=RPM11/SQRT(TORQ11)
          IF(COAST) GO TO 100
C
C         FOR DRIVE MODE
C
          IF(TORQ2.GE.0.000001) GC TO 8
          SR=1.0
          RETURN
8         CONTINUE
          RAT=RPM2/SQRT(TORQ2)
          IF(RAT.LT.TCRPPM) GC TO 20
          IF(RAT.LT.AKD(NTD)) GO TO 10
          SR=SRD(NTD)
          GO TO 140
10        J=NTD-1
11        IF(RAT.GT.AKD(J)) GO TO 12
          J=J-1
          GO TO 11
12        JP=J+1
          SR=(SRD(J)-SRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+SRD(JP)
          IF(SR.GT.1.0) SR=1.0
          GO TO 140
20        IF(RAT.LT.AKD(1)) GO TO 30
          J=NTDP-1
22        IF(J.LT.1) GO TC 30
          IF(RAT.GT.AKD(J)) GO TO 25
          J=J-1
          GO TO 22
25        JP=J+1
          SR=(SRD(J)-SRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+SRD(JP)
26        TR=(TRD(J)-TRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+TRD(JP)
          IF(SR.GT.1.0) SR=1.0
27        IF(SR.GE.0.0) GC TO 140
          SR=0.0
          RAT=-(AKD(J)-AKD(JP))/(SRD(J)-SRD(JP))*SRD(JP)+AKD(JP)
          GO TO 26
30        J=1
          GO TO 25
```

```
C
C         FOR COAST MODE
C
100       CONTINUE
          IF(RPM2.LT.SRC(1)*AKC(1)) GO TO 55
          IF(RPM2.GT.SRC(NTC)*AKC(NTC)) GO TO 60
          GO TO 65
C
C         IF BELOW LOWEST SR GIVEN AS INPUT
C
55        J=1
          JP=2
          GO TO 75
C
C         IF ABOVE HIGHEST SR GIVEN AS INPUT
C
60        JP=NTC
          J=JP-1
          GO TO 75
C
C         FIND CORRECT SEGMENT FOR CURRENT POINT
C
65        DO 70 J=2,NTC
          IF(RPM2.GE.SRC(J-1)*AKC(J-1).AND.RPM2.LE.SRC(J)*AKC(J))
        1 GO TO 73
70        CONTINUE
73        JP=J
          J=JP-1
C
C         COMPUTE SPEED RATIO BY INTERPOLATION
C
75        SR=(SRC(J)-SRC(JP))/(AKC(J)*SRC(J)-AKC(JP)*SRC(JP)) *
        1 (RPM2-AKC(J)*SRC(J))+SRC(J)
          IF(SR.LT.1.0) SR=1.0
          GO TO 140
C
C         IF SR GREATER THAN MAX INPUT , SET TO MAX
C
90        CONTINUE
          IF(SR.GT.SRD(NTD)) SR=SRD(NTD)
          RETURN
140       CONTINUE
          RPM1=RPM11/SR
150       DL=((A1/A2)*RPM11+((A2-A1)/A2)*RPM1)-RPM2
          IPASS=IPASS+1
          IF(IPASS.GE.100) STOP
C         WITHIN TOLERANCE? IF YES,END OF ITERATIVE PROCESS
160       IF(ABS(DL).LE.DVV) GO TO 310
C         CONTINUE ITERATION ALONG SAME DIRECTION
170       IF(DL.LT.0.0) GO TO 210
180       RPMR=RPM11
190       DLR=DL
          GO TO 230
210       RPML=RPM11
220       DLL=DL
230       IF(ABS(DLR*DLL).NE.0.0) GO TO 290
          IF(COAST) GO TO 270
C         DRIVE MODE
          RPM11=RPM11-DLLREP
          GO TO 110
270       RPM11=RPM11+DLLREP
```

```
        GO TO 110
290     RPM11=RPML+(((RPMP-RPML)/(ABS(DL)+DLR))*ABS(DL))
        GO TO 110
310     SR=RPM2/RPM1
        IF (CCAST) GO TC 360
        TORC1=TORQ2*((A2-A1)/A2)
        TR=TORQ2/(TCRC1+TCRQ11/TR)
        RETURN
360     CONTINUE
C       COAST MCDE
        TR=1.0
        RETURN
        END
        TITLE VEHSIM
;
;*****************************************************************
;
;        VEHSIM PROGRAM  ===  MAY , 1976  ( BATCH/INERACTIVE VERSION )
;                             ORIGINATOR - H.GOULD/E.WITHJACK,DOT-TSC
;                             DESIGNERS  - S.MOFFATT/D.CRUZE,KHL
;                             DEC-SYSTEM-10 MODIFICATIONS J.GOODRIDGE,KHL
;                                                AND S.SHAPIRO/KHL
;
;       ENTRY POINTS:   EXIT,   RESET,  RESETM, TAMPER, VEHSIM
;
;       SUBROUTINE CALLED:      VSMCTR
;
;       CALLED BY:      GCDACK, INPEAT, INPDIA, LOOKUP, SIMCTR,
;                       SIMINT, SIMSIS, VSMCTR
;
;       RUNS UNDER MACRO-10/FORTRAN-10/LINK-10
;
;*****************************************************************
;
;
        VERMAJ=600
        VERMIN=23
        .JBSA==120
        .JBREN==124
        .JBCPC==130
        ENTRY VEHSIM,EXIT,RESET,RESETM,TAMPER
        EXTERNAL RESET.,VSMCTR,CLCSE,STOP.,EXIT.,RENTER
;
.COMMON RUNID   [¬D15]
;
        LOC     137
        VERMAJ+VERMIN
        RELOC
;
VEHSIM: JFCL 0,0        ;CLEAR FLAGS
        JSP 16,RESET.   ;INTIALIZE FOROTS (FORtran Object Time System)
        0,,0
        MOVEI 1,RESET   ;GET REENTER ADDRESS
        HRRZM 1,.JBREN  ;STORE REENTER ADDRESS IN JOBDAT AREA
        MOVE    1,VERASC;GET ASCII VERSION NAME
        MOVEM   1,RUNID+¬D12    ;STORE IT IN COMMON
        MOVEI   1,VERMAJ        ;GET MAJOR EDIT NUMBER
        IDIVI   1,100
        MOVEM   1,RUNID+¬D13    ;STORE IT
        MOVEI   1,VERMIN        ;GET MINOR EDIT NUMBER
        MOVEM   1,RUNID+¬D14    ;STORE IT
        MOVEI 16,M1
        PUSHJ 17,VSMCTR ;CALL VEHSIM CONTROL ROUTINE
;
BADSTK: OUTSTR [ASCIZ
?
? VSMMAC - BAD PUSHDOWN STACK

*EXIT VIA VEHSIM/FOROTS

 ]      ;BAD PUSHDOWN STACK PROBABLY DUE TO VEHSIM MISHANDLING NOT FOROTS
TAMPER: SETZM 0,.JBSA   ;PREVENT RESTART OF VEHSIM
        SETZM 0,.JBREN
        SETZM 0,.JBCPC
```

```
;
EXIT:   MOVEI 16,M1       ;NORMAL PROGRAM EXIT IS A CALL TO EXIT
        PUSHJ 17,CLOSE    ;MAKE SURE ALL FILES CLOSED AND DISPOSED OF AS REQUESTED
        MOVEI 16,M1       ;FOROTS TERMINATION OF PROGRAM VERSIM
        PUSHJ 17,STOP.
TEXIT.: MOVEI 16,M1
        PUSHJ 17,EXIT.
;
RESETM: OUTSTR [ASCIZ/

*RESET
/]
RESET:  PUSHJ 17,CLOSE    ;ON RESET CLOSE ALL FILES FIRST
        JFCL 0,0          ;CLEAR FLAGS
        JSP 16,RESET.     ;REINITIALIZE FOROTS
        0,,0
        PUSH 17,BADSTK    ;PUT BADSTK ADDRESS ON BOTTOM OF STACK
        JRST RENTER       ;GO TO REENTRY POINT IN VSMCTR
;
;       ARGUMENT BLOCKS
;
        0,,0
M1:     0,,0              ;DUMMY ARG BLK
;
VERASC: ASCII  /CAR  /
;
;
        PRGEND VERSIM
        TITLE   CHKFIL
;***********************************************************
;
;       SUBROUTINE CHKFIL
;               AUTHOR/SID SHAPIRO/KHL
;               9/1/76
;
;       SUBROUTINE CHKFIL WILL CHECK FOR A FILE'S EXISTENCE
;       THE CALLING SEQUENCE IS AS FOLLOWS (CHKFIL.MAC IS CALLABLE  .
;       ONLY FROM FORTRAN-10):
;
;       CALL CHKFIL(DEV,FILE,PPN,$CKLABEL), OR
;       CALL CHKFIL(DEV,FILE,PPN,$OKLABEL,IPROT), OR
;       CALL CHKFIL(DEV,FILE,PPN,$OKLABEL,IPROT,LIB)
;       ERROR RETURN
;
;       WHERE   DEV=    LITERAL CONSTANT OR LITERAL STRING.
;               FILE=   LITERAL STRING OR DOUBLE PRECISION WORD.
;               PPN=    2 WORD ARRAY WITH PPN (OCTAL CONSTANT) RIGHT JUSTIFIED.
;               $CKLABEL=       THE STATEMENT LABEL THAT THE USER
;                       WISHES CONTROL TO PASS TO ON A NORMAL RETURN.
;                       CHKFIL WILL PASS CONTROL TO THE STATEMENT
;                       IMMEDIATELY FOLLOWING THE CALL ON AN ERROR RETURN.
;
;
;
;       ENTRY POINTS: CHKFIL
;
;
;***********************************************************
;
        SEARCH  C
        ENTRY   CHKFIL
        SALL
;;
;;---------------------------------------------------------
;;
;;      THIS MACRO GETS A FREE CHANNEL
;;
        DEFINE  CHANEL
<       PUSHJ   17,GETCHN##
        JUMPGE  .+4
        OUTSTR  [ASCIZ/
?CHANEL - no free channels)
/]
        EXIT    1,
        EXIT
        LSH     5
        HRLZM   CHNC.
        ARRAY   CHNC.[1]>
;;
;;---------------------------------------------------------
;;
        DEFINE  ERMES(STRING)<
        JRST    [       OUTSTR  [ASCIZ/
?STRING
/]
        EXIT    1,
        EXIT    ]
>
;;
```

```
;;------------------------------------------------------------
CHKFIL: CHANEL                          ;GET A CHANNEL
        MOVE    1,@(16)                 ;GET DEVICE
        JUMPN   1,GOTDEV                ;GOT ONE?
        MOVSI   2,'DSK'                 ;NO. GET DEFAULT.
        JRST    SETOPN                  ;GO OPEN

GOTDEV: SETZM   2
        MOVE    3,[POINT 7,1]           ;GET INPUT PNTR.
        MOVE    4,[POINT 6,2]           ;GET OUTPUT PNTR
GTDV1:  ILDB    3                       ;GET A CHAR
        JUMPE   SETOPN                  ;DONE?
        CAIN    " "                     ;DONE?
          JRST  SETOPN
        SUBI    40                      ;MAKE SIXBIT
        IDPB    4                       ;PUT IT AWAY
        JRST    GTDV1                   ;GO DO MORE

SETOPN: MOVEI   1,.IOASC                ;GET MODE
        MOVEI   3,BUFF                  ;GET A BLOCK FOR BUFFERS
        MOVE    CHNO.
        IOR     [OPEN 0,1]
        XCT
          ERRMES         (CHKFIL - cannot open device or no channels)
        MOVEI   5,@1(16)                ;GET ADR OF FILE NAME
        MOVE    6,[POINT 7,(5)]         ;GET INPUT PNTR
        MOVE    7,[POINT 6,1]           ;GET OUTPUT PNTR.
GETFIL: ILDB    6                       ;GET A CHAR
        JUMPE   DONFIL                  ;DONE?
        CAIN    " "                     ;DONE?
          JRST  DONFIL                  ;YES
        CAIN    "."                     ;GOT EXT?
          JRST  [MOVE    7,[POINT 6,2]
                 JRST    GETFIL]        ;YES
        SUBI    40                      ;MAKE IT SIXBIT
        IDPB    7                       ;PUT IT AWAY
        JRST    GETFIL

DONFIL: MOVEI   5,@2(16)                ;GET ADR OF PPN
        HRL     4,(5)                   ;GET PROJ #
        HRR     4,1(5)                  ;GET PROG #
        JUMPN   4,GOTPPN                ;GOT ONE?
        MOVE    11,[-1,,.PTFRD]         ;NO GET PATH
        MOVE    10,[3,,11]
        PATH.   10,
          SETZM 11+.PTPPN               ;ERROR - ZERO TO GET DEF DIR.
        MOVE    4,11+.PTPPN             ;GET IT
GOTPPN: SETZM   3
        MOVE    CHNO.
        IOR     [LOOKUP 0,1]
        XCT
        JRST    TRYLIT                   ;LOOKUP ERROR
;
OKRET:  MOVE    3(16)                   ;GET RETURN ADDR
        HRRM    (17)                    ;PUT IT INTO STACK
        HLRZ    -1(16)                  ;GET # OF ARGS
        CAILE   -5
        JRST    ERROR                   ;NOT WANT PROT, RETURN
        LSH     3,-^D27
        MOVEM   3,@4(16)
ERROR:  MOVE    1,CHNO.
```

```
        IOR     1,[RELEAS 0,]
        XCT     1
        POPJ    17,             ;OKAY RETURN
;
TRYLIB: HLRZ    -1(16)                  ;GET # OF ARGS
        CAIE    -6                      ;TEST FOR LIB PPN?
        JRST    ERROR                   ;NO
        SETZM   @5(16)                  ;YES, ASSUME NOT LIB PPN
        MOVE    11,[-1,,-4]
        MOVE    [4,,11]
        PATH.   10,
          JRST  ERROR
        JUMPE   13,ERROR
        MOVEM   13,4
        SETZM   3
        MOVE    CHNC.
        IOR     [LOOKUP 0,1]
        XCT
          JRST  ERROR
        MOVEI   1,@2(16)
        HLRZ    13
        MOVEM   (1)
        HRR     13
        MOVEM   1(1)
        SETCM   @5(15)
        JRST    ORRET

BUFF:   BLOCK   3
        PRGEND
        TITLE   CNTRLC
;
;       THIS ROUTINE SETS UP AN INTERUPT
;       SERVICE ROUTINE TO HANDLE CONTROL-C INTERUPTS,
;
;       USE BY CALLING SETC IN YOUR MAIN ROUTINE.
;
;               EXTERNAL SUBA
;               CALL SETC(SUBA)
;
;       THEN WHEN THE PROGRAM IS EXECUTING, TWO CONTROL-C'S
;       WILL CAUSE A BRANCH TO SUBA.
;       SUBA IS, OF COURSE, THE NAME OF YOUR SERVICE ROUTINE.
;
        ENTRY   SETC
        LOC 134
        EXP INTBLK
        RELOC

INTBLK: XWD 4,INTRTN            ;CONTROL BLOCK FOR ¬C TRAP
        XWD 0,2                 ;¬C TRAP
PC:     BLOCK 2                 ;PC WORD, SPARE

INTRTN: PUSH    17,PC           ;SAVE INTERUPTED PC
        SETZM   ,PC             ;ENABLE FOR NEXT ¬C
        PUSH    17,[-1]         ;GET CUR LINE
        GETLCH  (17)            ;CHARACTERISTICS
        SETLCH  [2]             ;SET TTY NORMAL!!
XPUSH:  JRST    SIOF            ;BRANCH TO SUBROUTINE
        SETLCH  (17)            ;RESTORE LINE CHARACTER AS BEFORE
        POP     17,(17)         ;FIX STACK
        POPJ    17,             ;RETURN IF CONTINUE
```

```
;
SETC:   MOVE    (16)            ;GET ADDR OF SUBROUTINE
        HRRM    XPUSH           ;PUT ADDR INTO PUSHJ INSTRUCTION.
        POPJ    17,             ;RETURN
;
STOP:   EXIT    1               ;DEFAULT ROUTINE IF THIS ROUTINE GETS LOADED
                                INADVERTANTLY.
;
        PRGEND
        TITLE   DAND
;
;       THIS SUBROUTINE ANDS TWO DOUBLE PRECISION WORDS
;
;       CALL DAND(WORD1,WORD2,WORD3)
;       WHERE WORD3=WORD1.AND.WORD2
;
        ENTRY DAND
DAND:   DMOVE   @(16)                   ;GET FIRST WORD.
        DMOVE   2,@1(16)                ;GET SECOND WORD.
        AND     2                       ;.AND. FIRST PART.
        AND     1,3                     ;.AND SECOND PART.
        DMOVEM  @2(16)                  ;PUT WORD INTO RETURN.
        POPJ    17,                     ;RETURN
        PRGEND
```

```
          TITLE    FILSPC
;         DESCRIPTION AND PARSES OUT INTO SEPERATE VARIABLES
;         THE DEVICE, FILE NAME, AND DIRECTORY.
;
;         THE CALLING SEQUENCE  IS AS FOLLOWS;
;
;         CALL FILSEC(ISPECS,IDEV,IFILE,IPPN)
;         WHERE
;                  ISPECS  - AN ALPHA ARRAY OF 10 WORDS CONTAINING A
;                            GENERAL FILE DESCRIPTION.
;                  IDEV    - RETURNED DEVICE.  IF DEVICE IS NOT SPECIFIED,
;                            DEVICE RETURNED IS DEFAULT "DSK".
;                  IFILE   - A DOUBLE PRECISION VARIABLE CONTAINING,
;                            IN ALPHA FORM, THE COMPLETE FILE NAME AND EXTENSION.
;                  IPPN    - A 3-5 WORD VARIABLE RETURNED CONTAING THE THE
;                            DIRECTORY.  MUST BE DIMENSIONED TO 3
;                            IF NO SFD'S PRESENT.  IF SFD'S ARE PRESENT,
;                            MUST BE DIMENSICNED TO 5.  IF NO DIRECTORY
;                            IS GIVEN, DEFAULT OF PPN FOR JOB IS RETURNED.
;
;         ENTRY POINTS: FILSPC
;
;         CALLED BY: VSMC7R
;
;         ****************************************************************
;
          SUBTTL     WRITTEN BY SID SHAPIRO/KHL  12/20/76
          ENTRY    FILSPC
FILSPC:   MOVE     [XWD ZBUFF,NCHAR]
          BLT      DEVFLG
          MOVE     [XWD SAVETR,LODPTR]
          BLT      LODPTR+3         ;RESTORE ALL FLAGS AND POINTERS
          MOVE     [XWD ZBUFF,BUFF]
          BLT      BUFF+7
          MOVE     4,(16)           ;GET ADDR OF FLE SPECS
          MOVEI    5,BUFF           ;GET ADDR OF BUFFER
          MOVEI    10,11            ;GET MAX NUMBER OF WORDS
          SETZ     9,               ;ZERO WORD COUNTER
 GETWRD:  SOJE      10,SETBYT
          MOVE     @4               ;GET WORD
          AOJ      4,
          CAMN     BLANK            ;NULL WORD? YES, GO GET ANOTHER
          JRST     GETWRD
          AOJ      9,               ;INC WORD COUNTER
          MOVEM    @5               ;PUT IT AWAY
          AOJA     5,GETWRD
SETBYT:   MOVEI    10,5             ;GET NUMBER OF CHARS PER WORD
          IMUL     10,9             ;MULT BY # OF WORDS TO GET MAX # OF CHARS
          AOJ      10,
          SETZ     9,               ;ZERO CHAR COUNTER
GETBYT:   SOJE     10,DONE          ;DONE YET?
          ILDB     BUFPTR           ;GET A BYTE
          CAIN     .40              ;NULL?
          JRST     GETBYT           ;YES,GO GET ANOTHER BYTE
          AOJ      5,               ;INC CHAR COUNTER
          CAIN     72               ;IS IT A COLON?
          SETCM    DEVFLG           ;YES, SET DEVICE FLAG
          CAIN     133              ;IS IT A LEFT BRACKET?
          SETCM    DIRFLG           ;YES, SET DIRECTORY FLAG
          CAIE     135              ;IS IT A RIGHT BRACKET?
          JRST     GETBYT           ;NO, GET ANOTHER BYTE
```

```
DCNE:    MOVEM    9,NCHAR
         MOVE     10,NCRAF           ;GET EDMBER OF CHARS
         AOJ      10,
         MOVE     SEFPTR
         MOVEM    EDFPTR
         SKIFL    DFVFLG             ;CEVICE?
         JRST     ECNDEV             ;NO. CO FILE
DODFV:   SOJF     10,ECNDEV
         ILDE     UUFPTR             ;GET EYTE
         CAIN     "2                 ;IS IT COLON
         JRST     ECNDEV             ;YES, CONE
         IDPE     DEVPTR             ;FUT IT AWAY
         JRST     DCDFV
ECNDEV:  MOVE     DEV                ;GET DFVICE
         MOVEM    @1(16)             ;RETURN IT
         SETCM    DEFEXT             ;ASSOME USE DEFAULT EXTENSION
DOFIL:   SOJF     10,DCBFIL
         ILDE     BDFPTR             ;GET EYTE
         CAIN     133                ;IS IT LEFT BRACKET?
           JFST   DCNFIL             ;YES, CONE
         CAIN     ".""                ;GOT A PERIOD?
           SETZM  DFFFXT             ;YOF, CAN'T USE DEFAULT EXT.
         IDPE     EILFTR             ;FUT IT AWAY
         JRST     DCFIL
DCNFIL:  SKIFL    DFFFXT             ;USE DEFAULT?
           JFST   DNFL3              ;NO
         MOVE     14,[ASCIZ/.VSM/]           ;GFT DEFAULT
         MOVF     15,[PCINT 7,14] ;GET FCINTER
DNFI2:   ILDE     15
         JUMEE    DNFL3              ;GOT NDLL, DONE WITH DEFAULT
         IDPE     FILPTR             ;FDT IT AWAY
         JRST     DNFI2
DNFI3:   DMCVE    FILE               ;GET FILE NAME
         DMCVEM   @2(16)             ;FUT IT AWAY
         SKIFL    DIRFLG             ;IS THFRE A DIRECTORY?
         JRST     DFFPPN             ;NO DO DEFADLT PPN
         JRST     SFTPPN             ; YES, DC PPN STUFF
NXTWRD:  MCVFM    7,@5               ;FUT FFN AWAY
         AOJ      5,                 ;INC FFN WORD ADDR
         SETZ     7,                 ;ZERO OUT 7
         JRST     DCPPN
SETPPN:  MOVE     5,3(16)            ;GET ACDR OF PPN RETDRN
         SETZ     7,
DOPEN:   SOJE     10,DCNPPN
         ILDE     EOPPTR             ;GET EYTE
         CAIN     54                 ;IS IT A COMMA
         JRST     NXTWRD             ;YES, GO GFT ANOTHER WORD
         CAIN     135                ;IS IT RIGHT ERAKET
         JFST     DCNPPN             ;YES, CONE
         SUBI     60                 ;CCNVEET TO OCTAL
         IMULI    7,10               ;SHIFTRECEIVING WORD
         ADD      7,0                ;ADD IN NEW DIGIT
         JRST     DCPPN
DCNPPN:  MOVEM    7,@5               ;FUT AWAY LAST WORD
         POPJ     17,
;
DEFPPN:  GETPPN
         JFCL     0,0                ;NO-OF
         MOVE     5,3(16)            ;GFT ACDR OF RETURN PPN ARG
         HLRM     @5                 ;PUT FROJ NUMBER AWAY
         AOJ      5,
        HRRE    ▪5              ;EUT EBOG NUMEER AWAY
        POFJ    17,
;
        HALT
        PAGE
SAVETR:SDFPTR:  PCINT   7,BUFF
SDVETR: PCINT   7,DEV
SELETR: POINT   7,FILE
SDEV:   422471,,300000
        Z
;
LCDETR:EUFPTR:  Z
DEVETR: Z
FILETF: Z
DEV:    BLOCK 2
;
NCHAR:  BLOCK   1
DEFEXT: BLOCK   1
FILE:   BLOCK   2
DIRFLG: Z
DEVFLG: Z
BUFF:   BLOCK 20
ZBUFF:  BLOCK 20
;
BLANK:  MOVEI   20100(4)        ;ELANK WOED
        PRGEND
        TITLE   GETCHN
;               ;
;       RETURNS IN REG 0 A FREE CHANNEL #.
;
        ENTRY   GETCHN
GETCHN: PUSH    17,1
        MOVEI   17
CHNSCN: MOVEM   1
        DEVTYP  1,
        JFCL
        CAIE    1,0
        SOJG    CHNSCN
        POP     17,1
        POPJ    17,
        PRGEND
        TITLE   GETDDT
        SEARCH  UUOSYM
        ENTRY   GETDDT
GETDDT: HRRZ    .JBDDT
        JUMPE   NODDT
        OUTSTR  [ASCIZ/[DDT]
/]
        JRST    @
        POPJ    17,
;
NODDT:  OUTSTR  [ASCIZ/?DDT not loaded
/]
        POPJ    17,
        PRGEND
        TITLE   GETNAM
;
;       THIS ROUTINE IS USED BY SECURE TO RETURN
;       THE PROGRAM NAME WITH AN ".EXE" TACKED ONTO
;       THE END
;
        ENTRY   GETNAM
GETNAM: MOVE    [-1,,3]                 ;GET NAME FROM MONITOR.
        GETTAB
;
        JFCL
        MOVEM   SIXNAM                  ;SAVE IT.
;
        MOVE    [POINT 6,SIXNAM]        ;GET POINTER.
        MOVE    1,[POINT 7,SEVNAM]
;
LOOP:   ILDB    3,0                     ;GET CHAR.
        JUMPE   3,EXT                   ;NULL?
        ADDI    3,40                    ;MAKE SEVEN BIT.
        IDPB    3,1                     ;PUT IT AWAY.
        JRST    LOOP                    ;DO MORE.
;
EXT:    MOVE    [POINT 7,EXE]           ;GET POINTER.
        MOVEI   6,4                     ;GET A CNTR.
;
LOOP2:  ILDB    3,0                     ;GET A CHAR FROM EXTENSION.
        IDPB    3,1                     ;PUT IT AWAY.
        SOJG    6,LOOP2                 ;DONE YET?
;
        DMOVE   SEVNAM                  ;YES, GET NNAME.
        DMOVEM  @(16)                   ;PUT IT INTO RETURN PLACE.
        POPJ    17,
;
EXE:    ASCII   /.EXE/
SEVNAM: BLOCK   2
SIXNAM: Z
        Z
        PRGEND
```

```
TITLE GETPUT
;
;       SUBROUTINE/FUNCTION [I]GET/[I]PUT(S,I,T)
;
;       SUBROUTINE/FUNCTION BYTPTR(S,I,WPT,CPT)
;
;
;       ENTRY POINTS:   BYTPTR, IGET,   IPUT,   PUT
;
;       CALLED BY: DSKDIR, JOBPEN, VALID2
;
;*******************************************************************
;
;       AC USAGE.
        F=0
        PPT=1
        WPT=2
        CPT=3
        TMP=4
;
        ENTRY IGET,PUT,IPUT,BYTPTR
        IGET=GET
        IPUT=PUT
;
GET:    PUSHJ 17,SETPT.-1         ;SET BYTE PTR INDEX'S
        LDB F,POINT(CPT)          ;GET CHAR FROM STRING.
        DPB F,GPOINT              ;DEPOSIT CHAR IN CHAR:
        MOVE TMP,CHAR             ;GET CHAR:
        MOVEM TMP,@2(16)          ;STORE.
        JRST BYE                  ;DONE
;
PUT:    PUSHJ 17,SETPT.-1         ;SET BYTE PTR INDEX'S.
        MOVE PPT,2(16)            ;GET ADDRESS OF T
        LDB F,PPOINT              ;GET CHAR TO INSERT IN S.
        DPB F,POINT(CPT)          ;INSERT CHAR IN S.
        JRST BYE                  ;DONE
;
BYTPTR: MOVE F,@4(16)             ;GET BYTES/WORD
        PUSHJ 17,SETPT.           ;CALC INDEX'S TO POINT TO BYTE
        MOVEM WPT,@2(16)          ;RETURN WORD PTR INDEX.
        MOVEM CPT,@3(16)          ;RETURN CHAR PTR INDEX.
        JRST BYE                  ;DONE.
;
BYE:    MOVS CPT,FSAVE            ;SET FOR BLT TO RESTORE USED AC'S.
        BLT CPT,CPT               ;RESTORE USED AC'S.
        POPJ 17,                  ;BYE.
;
;       SET PTR'S TO CHAR IN STRING S TO BE REFERENCED.
;
        MOVEI F,5                 ;SET BYTES/WORD FOR GET & PUT.
SETPT.: MOVEM F,SAVE+4            ;SAVE BYTES/WORD.
        MOVE F,FSAVE              ;SET FOR BLT TO SAVE AC'S TO BE USED.
        BLT F,SAVE+3              ;SAVE AC'S.
        MOVE WPT,@1(16)           ;GET CHAR POS TO BE WORKED ON.
        IDIV WPT,SAVE+4           ;CALC # WORDS AND PUT REMAINDER IN CPT.
        SKIPN CPT                 ;REMAINDER? YES,SKIP.
        SUBI WPT,1                ;CALC INDEX FOR S.
        SKIPN CPT                 ;CHAR POS IN WRD SET? YES,SKIP.
        MOVE CPT,SAVE+4           ;NO, HAS TO BE POS#5.
        SUBI CPT,1                ;CALC POINT INDEX.
        ADDI WPT,@0(16)           ;ADD S INDEX TO S ADDRESS FOR BYTE PTR.
        POPJ 17,                ;BYTE PTR INDEX'S SET RETURN GET/PUT.
;
; ARGUMENT BLOCKS.
;
PSAVE:  1,,SAVE
SAVE:   BLOCK 5
CHAR:   001004020100
POINT:  POINT 7,0(WPT),6
        POINT 7,0(WPT),13
        POINT 7,0(WPT),20
        POINT 7,0(WPT),27
        POINT 7,0(WPT),34
PPOINT: POINT 7,0(PPT),6
GPOINT: POINT 7,CHAR,6
;
        PRGEND
        TITLE JOBPPN
;
;       WRITTEN BY J.GOODRIDGE DOT/TSC/KHL
;
;       CALL JOBPPN(IDEV,JOB,P,PN)
;
;       CALL DSKSTR(IDEV)
;
;       IDEV-ASCII FILE STRUCTURE NAME REQUESTED(SEE NOTE BELOW)
;       JOB-DECIMAL JOB NUMBER
;       P  -OCTAL PROJECT NUMBER
;       PN -OCTAL PROGRAMER NUMBER
;
;       ENTRY POINTS:   DSKSTR, JOBPPN
;
;       SUBROUTINES CALLED:     SEVBIT
;
;       CALLED BY:      VSMCTR
;
;*********************************************************************
;
        ENTRY JOBPPN,DSKSTR
JOBPPN: PJOB 1,         ;UUO TO REURN JOB NUMBER TO AC
        MOVEM 1,@1(16)  ;STORE JOB NUMBER IN JOB
        GETPPN 1,       ;UUO TO RETURN PPN TO AC
        HLRZM 1,@2(16)  ;STORE PROJECT NUMBER IN P
        HRRZM 1,@3(16)  ;STORE PROGRAMER NUMBER IN PN
DSKSTR: SETZ 10,
        SETZ 11,
        SETZ 12,
        SETZM 0,STOR
        HRLI 0,1        ;PUT 1 IN LH OF AC 0
        HRR 0,0(16)     ;PUT IDEV ADDRESS IN RH OF AC 0
        JOBSTR 0,       ;UUO TO RETURN SPECIFIED FILE STRUCTURE NAME
        JRST STRERR     ;ERROR RETURN
        SKIPN 7,@0(16)  ;FENCE(END OF FILE STRUCTURE SEARCH LIST)
        JRST FENCE      ;YES,GO PUT "FENCE" IN IDEV
        CAMN 7,[-1]     ;NOT END OF LIST?
        JRST ENDLST     ;YES
        MOVEM 16,AC16   ;SAVE AC 16
        MOVEM 7,STOR
        MOVEI 16,ARGBLK ;LOAD INDEX TO SIXSEV ARGUMENT BLOCK
        PUSHJ 17,SEVBIT##         ;CONVERT FILE STRUCTURE NAME TO ASCII
        MOVE 16,AC16    ;RESTORE INDEX POINTER TO JOBPPN ARGUEMENT BLOCK
        MOVE 11,STOR+2
        MOVE 12,STOR+3
RETURN: MOVEM 11,@0(16) ;STORE AC 11&12 IN IDEV
        MOVE 15,0(16)
        AOS 15
        MOVEM 12,@0(15)
        POPJ 17,        ;RETURN
;
STRERR: MOVE 11,[ASCII/ERROR/]  ;HANDLE JOBSTR UUO ERROR RETURN
        JRST RETURN
;
FENCE:  MOVE 11,[ASCII/FENCE/]
        JRST RETURN
;
ENDLST: MOVE 11,[ASCII/EMPTY/]
        MOVE 12,[ASCII /LIST /]
        JRST RETURN
```

```
;
AC16:   Z
        XWD -5,0
ARGBLK: XWD 0,STOR
        [XWD 0,1]
        XWD 0,STOR+2
        [XWD 0,1]
        [XWD 0,12]
STOR:   BLOCK 4
;
        PRGEND
        TITLE   OUTSTR
;
;       ENTRY POINTS:   OUTSTR
;
;       CALLED BY:      ASCIZ
;
;**************************************************************************
;
;
        ENTRY OUTSTR
;
OUTSTR: MOVE 15,@1(16)
        ADDI 15,@0(16)
        SETZ 0,             ;ZERO
        EXCH 0,0(15)        ;SAVE WRD AFTER END OF STRING AND ZERO IT.
        TTCALL 3,@0(16)     ;OUTPUT ASCIZ STRING FROM PASSED ADDRESSED
        MOVEM 0,0(15)       ;RESTORE WRD AFTER END OF STRING.
        POPJ 17,0           ;RETURN
;
        PRGEND
```

```
;
;********************************************************************
;
        TITLE   SECNDS
;
;       SECNDS IS A FORTRAN-10 CALLABLE SUBROUTINE TO RETURN JOB RUN TIME
;               IN SECONDS.
;       ENTRY POINTS:   SECNDS
;
;       CALLED BY:      KETIME
;
;********************************************************************
;
;
ENTRY   SECNDS
SECNDS: SETZ 2, ;INTIALIZE
        CALLI 2,27      ;UUO TO GET JOB RUNTIM IN MILLI SECONDS
        FSC 2,233       ;FLOAT RUNTIM
        FDV 2,CONS      ;CONVERT RUNTIM FROM MILLI SEC. TO SECONDS
        MOVEM 2,@0(16)  ;RETURN RUNTIM TO CALLING PROGRAM
        POPJ 17,0       ;RETURN
;
;       ARGUMENT BLOCKS
;
CONS:   1.E3
        PRGEND
```

```
TITLE SIXSEV
;
;       WRITTEN BY J.GOODRIDGE DOT/TSC/KHL
;
;       CALL SIXBIT ( SIX , SEV , NBYTES )
;
;       CALL SIXBIT ( SIX , IBSIX , SEV , IBSEV , NBYTES )
;
;       CALL SEVBIT ( SIX , SEV , NBYTES )
;
;       CALL SEVBIT ( SIX , IBSIX , SEV , IBSEV , NBYTES )
;
;               SIX - STARTING ADDRESS(FIRST WORD) OF ARRAY CONTAINING
;                       SIXBIT WORD(S).
;
;               SEV - STARTING ADDRESS(FIRST WORD) OF ARRAY CONTAINING
;                       SEVEN BIT(ASCII) WORD(S).
;
;               IBSIX & IBSEV - STARTING BYTE NUMBER OF STRING TO BE
;                               USED FOR INPUT/OUTPUT. WHEN CALL HAS 3
;                               ARGUMENTS 1 IS ASSUMED FOR IBSIX & IBSEV.
;
;               NBYTES - NUMBER OF BYTES(CHARACTERS) IN THE INPUT ARRAY.
;
;       CALLING THE SIXBIT ENTRY POINT TELLS SIXSEV TO CONVERT ASCII TO
;               SIXBIT, AND THAT SEV IS THE INPUT ADDRESS AND SIX THE
;               OUTPUT ADDRESS. CALLING THE SEVBIT ENTRY POINT TELLS
;               SIXSEV THE OPPISITE.
;
;       THE USER SHOULD KEEP IN MIND THE FOLLOWING POINTS:
;
;               1) THE SIZE OF THE NEEDED OUTPUT ARRAY WHEN CALLING SEVBIT,
;                       AS MORE OUTPUT WORDS ARE GENERATED THEN INPUT WORDS
;                       DUE TO THE DIFFERENT NUMBER OF CHAR'S/WORD
;
;               2) THE INPUT ARRAY IS UNCHANGED.
;
;               3) IN THE OUTPUT ARRAY ONLY ENOUGH WORDS ARE CHANGED TO
;                       ACCOMADATE THE CONVERTED INPUT ARRAY. ANY UNUSED
;                       BYTES IN A WORD AND ANY UNUSED WORDS ARE UNCHANGED.
;
;               4) WHEN GOING FROM SEVEN TO SIX BITS THE SAME STRING CAN BE
;                       USED FOR OUTPUT AS INPUT, OR ANY SUCH ARRANGEMENT
;                       THAT DOES NOT CAUSE THE OUTPUT TO OVER WRITE THE INPUT
;                       BYTES.
;       ENTRY POINTS:   SEVBIT,SIXBIT
;
;       SUBROUTINES CALLED:     BYTPTR
;
;       CALLED BY:      CHKAC
;
;*****************************************************************
;
;
;       AC DEF FOR CONVERSION ROUTINES
        BYTE=0          ;BYTE BEING CONVERTED
        WRDS6P=1        ;STARTING ADDRESS OF SIXBIT WORD(S)
        WRDS7P=2        ;STARTING ADDRESS OF ASCII(7 BIT) WORD(S)
        COUNT=3         ;NUMBER OF INPUT BYTES
        INB=4           ;BYTE POINTER FOR INPUT
        OUTB=5          ;BYTE POINTER FOR OUTPUT
        FAC=6           ;CONVERSION FACTOR
        PT6=7           ;INDEX TO SIXBIT BYTE POINTER.
        PT7=10          ;INDEX TO SEVBIT BYTE POINTER.
;
        ENTRY SIXBIT,SEVBIT
;
SIXBIT: PUSHJ 17,GETARG ;PICKUP ARGBLK AND PUT IN AC'S
        MOVE INB,BITS7(PT7)     ;SET UP FOR ASCII TO SIXBIT CONVERSION
        MOVE OUTB,BITS6(PT6)
        MOVEI FAC,40
        JRST LOOP
SEVBIT: PUSHJ 17,GETARG ;PICKUP ARGBLK AND PUT IN AC'S
        MOVE INB,BITS6(PT6)     ;SET UP FOR SIXBIT TO ASCII CONVERSION
        MOVE OUTB,BITS7(PT7)
        MOVNI FAC,40
LOOP:   ILDB BYTE,INB
        SUB BYTE,FAC
        IDPB BYTE,OUTB
        SOJG COUNT,LOOP
        MOVE 16,AC16
        POPJ 17,
;
GETARG: MOVEM 16,AC16           ;SAVE ARG BLK PTR.
        MOVE 10,16              ;GET ADD OF ARG BLK.
        SUBI 10,1               ;#OF ARG'S IN BLK AT ADD-1.
        MOVE 11,0(10)           ;GET #OF ARG'S.
        CAME 11,[-3,,0]         ;NEW ARG BLK FORMAT?
        JRST NEWARG             ;YES.
        MOVE WRDS6B,0(16)       ;GET WRDS6B
        MOVE WRDS7B,1(16)       ;GET WRDS7B
        MOVE COUNT,@2(16)       ;GET COUNT
        SETZI PT6,              ;SET INDEX TO SIXBIT BYTE PTR.
        SETZI PT7,              ;SET INDEX TO SEVBIT BYTE PTR.
        POPJ 17,
;
NEWARG: MOVE 15,16              ;SAVE INDEX FOR USE BY BYTPTR ARG BLK.
        MOVEI 16,PTRSIX         ;GET ARG BLK ADD.
        PUSHJ 17,BYTPTR##       ;SET UP SIXBIT BYTE POINTER.
        MOVE WRDS6B,TEMP
        MOVE PT6,TEMP+1
        MOVEI 16,PTRSEV         ;GET ARG BLK ADD.
        PUSHJ 17,BYTPTR         ;SET UP SEVBIT BYTE POINTERS.
        MOVE WRDS7B,TEMP
        MOVE PT7,TEMP+1
        MOVE COUNT,@4(16)
        POPJ 17,0               ;DONE.
;
BITS6:  POINT 6,0(WRDS6B),      ;SIXBIT BYTE POINTERS
        POINT 6,0(WRDS6B),5
        POINT 6,0(WRDS6B),11
        POINT 6,0(WRDS6B),17
        POINT 6,0(WRDS6B),23
        POINT 6,0(WRDS6B),29
;
BITS7:  POINT 7,0(WRDS7B),      ;ASCII(7 BIT) BYTE POINTERS
        POINT 7,0(WRDS7B),6
        POINT 7,0(WRDS7B),13
        POINT 7,0(WRDS7B),20
        POINT 7,0(WRDS7B),27
;
AC16:   Z
```

```
;
        -5,,0
PTRSIX: @0(15)
        @1(15)
        0,,TEMP
        0,,TEMP+1
        [0,,6]
;
        -5,,0
PTRSEV: @2(15)
        @3(15)
        0,,TEMP
        0,,TEMP+1
        [0,,5]
;
TEMP:   BLOCK 2
        PRGEND
        TITLE   TRACEL
        ENTRY   TRACEL
;
;       WRITTEN BY J.S.GOODRIDGE/KHL
;
;       CALL TRACEL(MAX,NARG,ITYPE,IERR,CNAME)
;
;       TRACEL IS A FORTRAN-10 CALLABLE ROUTINE THAT RETURNS INFORMATION
;               ABOUT THE ARGUMENT BLOCK FOR THE ROUTINE IN WHICH IT IS
;               CONTAINED.
;
;               MAX     - THE MAXIUM NUMBER OF ARGUMENTS THAT INFORMATION CAN
;                               RETURN ABOUT. (I.E. THE LENGTH OF ARRAY
;                               ITYPE.)
;
;               NARG    - THE NUMBER OF ARGUMENTS FOUND IN THE BLOCK IS
;                               RETURNED HERE. IF NOT ENOUGH SPACE IS ALLOCAT-
;                               TED FOR ITYPE THE # OF ARGUMENTS IS STILL
;                               RETURNED.
;
;               ITYPE   - AN ARRAY OF LENGTH MAX INTO WHICH THE VARIABLE
;                               TYPE CODES OF THE ARGUMENTS IS RETURNED.
;                               ITYPE(1) CORRESPONDS TO ARG1... ITYPE(N)=ARG(N)
;                               THESE CODES ARE THE ONES GIVEN IN APPENDIX D
;                               OF THE FORTRAN-10 REFERNCE MANNUAL.
;
;               IERR    - AN ITEGER VARIABLE CONTAINING AN ERROR RETURN CODE.
;                               FOR MACRO USERS THIS CODE IS ALSO RETURN
;                               AC 0.
;
;                               -1 - TRACEL CAN NOT FIND THE ARGUMENT BLOCK
;                                       IN QUESTION. IF CALLED FROM FORTRAN-10
;                                       ROUTINE THE ONLY POSSIBLE ERROR IS
;                                       THAT TRACEL WAS CALLED FROM THE MAIN
;                                       PROGRAM. IF CALLED BY MACRO ROUTINE
;                                       THE FORTRAN-10 CALLING SEQUENCE HAS
;                                       NOT BEEN USED OR TRACEL IS BEING CALLED
;                                       FROM THE MAIN MODULE.
;
;                                0 - NO ERRORS DETECTED. NORMAL RETURN CODE.
;
;                                1 - NOT ENOUGH SPACE ALLOCATED FOR ITYPE.
;                                       HOWEVER ITYPE CONTAINS INFORMATION FOR
;                                       ARGUMENTS THERE IS SPACE FOR.
;
;               CNAME   - A DOUBLE PRECISION VARIABLE TO WHICH THE 6 CHARACTER
;                               NAME OF THE CALLING ROUTINE IS RETURNNED
;                               LEFT JUSTIFIED. IF CALLING ROUTINE IS THE MAIN
;                               PROGRAM THE JOB'S NAME IS RETURNED.
;
;
;       CALLED BY:      LOOKUP
;
;*************************************************
;
        PAGE
        SIXBIT  /TRACEL/        ;NAME FOR TRACE.

TRACEL: SETZM   0,@3(16)        ;CLEAR ERROR RETURN CODE.
        MOVE    10,-2(17)       ;GET THE ADDRESS.
        JUMEN   10,SRCAIL       ;END OF STACK. NO MUST BE CALLED FROM MAIN.
        MOVE    12,[XWD -1,3]       ;SET UP AC FOR GETTAB.
        GETTAB  12,                 ;GET CURRENT JOD'S NAME.
        MOVE    12,MAINO            ;IF UUO FAILS USE /MAIN./
        MOVEI   14,13               ;FCINT TO  JOE NAME+1.
        JRST    NAME                ;GO RETURN JOB NAME AS CALLER'S NAME.
SECAIL: MOVE    0,-1(10)            ;GET PUSHJ 17, CALLING INSTRUCTION.
        MOVE    14,0                ;REMEMBER ADDRESS.
        AND     0,ACMASK            ;CLEAR ADCRESS FIELD.
        CAME    0,PUSJ17            ;IS IT A PUSHJ 17,* INSTRUCTION?
        JRST    NCGO                ;NO, ECOOCOOOH! LETS GET OUT OF HERE.
NAME:   MOVE    3,4(16)             ;GET ACDRES OF CNAME
        MOVE    4,CNPTR             ;GET CNAME BYTE POINTER.
        MOVE    5,NMPTR             ;GET FCUTINE NAME BYTE POINTER.
        HRLZI   6,-6                ;SET UP LOOP CONTROL AC.
NLOCP:  ILDE    7,5                 ;GET A SIXBIT BYTE.
        ADDI    7,40                ; SIXEIT ===> SEVENBIT.
        IDPE    7,4                 ;PUT IT IN CNAME.
        AODJN   6,NLCOP             ;ANY MCBE? YES.

        MOVE    10,-1(17)           ;GET THE ADCRESS.
        MOVE    0,-2(10)            ;GET THE INSTRUCTION.
        MOVE    14,0                ;REMEMEER IT FOR WHEN ADDRESS NEEDED.
        ANC     0,ACMASK            ;MASK CUT THE ADDRESS FIELDS.
        CAME    0,MCVI16            ;IS IT A MOVEI 16,  INSTRUCTION ?
        JRST    NCGO                ;NO, ECCOCOCOH! LETS GET OUT OF HERE.
        SOJ     14,                 ;FOINT TO ARG ELOCK CCUNT OF ARG'S.
        SKIPN   1,0(14)             ;ANY ARG'S ?
        JRST    NCARG               ;NC TCCTLES.
        HLRE    2,1                 ;CONVERT IT TO FULL WORD #.
        MOVEM   2,@1(16)            ;RETURN ARG COUNT
        MOVE    2,@1(16)            ;AND GET IT EACK.
        CAMLE   2,@0(16)            ;ENOUGH RCOM ALLOCATED IN ITYPE.
        JRST    TCARG               ;NO, GC SET ERR CODE AND SOFORTH.
        SETZI   0,                  ;NO EFRORS CLEAR AC 0 FOR MACRO USERS.
        HRR     1,2(16)             ;GET ITYPE ADCRESS AND PUT IT LOOP CTRL AC.

LCOE:   AOJ     14,                 ;FOINT AN ARG BLK ENTRY
        LDB     2,TYPTR             ;LOAD TYPE CODE FRCM ARG BLOCK.
        MOVEM   2,0(1)              ;AND RETURN IT IN ITYPE.
        AOBJN   1,LCOP              ;EUMP! ARE WE DONE? NO- TO LOOP.
        POPJ    17,                 ;YES-EYE!

NCARG:  ;HERE WHAN NO ARGUMENTS IN ARG DLK LIST.
        SETZM   @1(16)              ;ZERO ARG CCUNT IN CALLER(NARG).
        POPJ    17,                 ;AND RETURN TC CALLER.

TOARG:  ;HERE WHEN NOT ENCUGH SPACE ALLOCATED FOR ITYPE.
        AOS     0,@3(16)            ;SET ERROR CODE.
        MOVN    1,@0(16)            ;LETS GET # WE CAN DO. MAKE (-).
        HRL     1,1                 ;SET UP LCCP CONTROL AC.
        JRST    LCOE-1              ;GO PCINT TO ITYPE AND ENTER LOOP.

NCGC:   ;HERE ON FATAL ERROR.
        SOS     0,@3(16)            ;SET ERROR CODE.
        POPJ    17,                 ;AND DISAPPEAR!!!

;       MASK AND FOINTERS.

ACMASK: 777740M                     ;WORD TO ZERO ADDRESS FIELD.
POSJ17: 260740M             ;MACHINE INST. PUSHJ 17,0.
CAPTB:  POINT   7,0(3),
NEPTR:  POINT   6,-1(14),
MCVI16: 2017000                 ;MACHINE INST. MOVEI 16,0
TYPTR:  POINT   4,0(14),12      ;BYTE PTR TO ARGUMENT TYPE CODE ARG BLK ENTRY
MAINO:  SIXBIT/MAIN./

        PRGEND
        TITLE TTYINP
;
;       FORTRAN-10 CALLABLE ROUTINE TO ALLOW A FORTRAN-10
;           PROGRAM AN INTERUPT STRUCTURE WITH LIMITATIONS
;
;
;       CALLED BY:      GCBACK, INPDIA, SIMINT, SIMLPT, SIMSTS
;
        ENTRY TTYINP,TTYCLR,TTYSET
;
;
;*************************************************************
;
TTYINP: INCHRS .CHAR    ;CHARACTER INPUT?
        POPJ 17,0       ; NO! RETURN.
        MOVEM 16,AC16   ; YES! SAVE AC16.
        MOVE 15,16      ; GET ADDRESS OF ROUTINE TO HANDLE INTERUPT
        MOVEI 16,ARGBLK ; GET POINTER TO INTERUPT CHARACTER
        PUSHJ 17,@0(15) ;CALL INTERUPT HANDLER
        MOVE 16,AC16    ;PROGRAM INTERUPT COMPLETED! RESTORE AC 16
;
TTYCLR: CLRBFI          ;CLEAR INPUT BUFFER
        POPJ 17,0       ;RETURN
;
TTYSET: INCHRS  .CHAR   ;BUFFER EMPTY? ALSO NULLIFIES ^O !
        POPJ 17;0       ; YES ! RETURN.
        CLRBFI          ; NO! CLEAR BUFFER
        POPJ 17,0       ; RETURN
;
;       ARGUMENT BLOCKS
;
AC16:   Z
        -1,,0
ARGBLK: .+1
.CHAR:  Z
        PRGEND
        TITLE TTYSTS
;
;       WRITTEN BY J.GOODRIDGE/KHL
;       MODIFIED BY S SHAPIRO/KHL
;
;       CALL TTYSTS(PTY,TTYNUM,JCTWD)
;
;               PTY- ASSUMED LOGICAL FORTRAN VARIABLE THAT IS SET .FALSE.
;                       IF JOB TTY IS PHSYICAL TTY AND SET .TRUE. IF IT
;                       IS A PSEUDO TTY.
;
;               TTYNUM- THE NUMBER OF THE JOB CONTROLLING TERMINAL
;                       IS RETURNED TO THIS LOCATION
;
;               JCTWD   - IS RETURNED AS THE WIDTH OF THE CONTROLLING JOB'S TERMINAL
;
        ENTRY TTYSTS
;
;
;       CALLED BY:      VSMCTR
;
;
;*****************************************************************
;
AC1=1
N=2
.TOWID=1012             ;READ WIDTH FUNCTION NUMBER
;
TTYSTS: SETOM TTYWRD    ;SET TTYWRD (-) SO JOB CONTROLLING TTY INFO RETURNED
        SETOM @0(16)    ;ASSUME PTY .FALSE.
        TTCALL 6,TTYWRD ;UUO TO RETURN TTY INFO
        MOVE 1,TTYWRD   ;PUT RETURNED INFO IN AC1
        TDNN 1,PTYBIT   ; PTY?
        SETZM @0(16)    ; YES! SET PTY .TRUE.
        SUBI 1,200000   ; NO! CALCULATE TTYNUM.
        HRRZM 1,@1(16)  ; STORE RESULT IN TTYNUM
WIDTH:  PJOB    AC1,            ;GET JOB NUMBER
        TRMNO.  AC1,            ;GET LINE NUMBER
;
        JRST    [OUTSTR [ASCIZ / /]
        JRST    RETRN]                  ;GO TO ERROR RETURN
;
        MOVEM   AC1,ADRWID+1            ;STORE LINE #
        MOVE    AC1,[XWD N,ADRWID]      ;SET UP FOR READ WIDTH UUO
        TRMOP.  AC1,                    ;READ TTY WIDTH
;
        JRST    [OUTSTR [ASCIZ / /]
        JRST    RETRN]                  ;GO TO ERROR RETURN
;
        MOVEM   AC1,@2(16)      ;STORE RESULT IN ARGUMENT
        POPJ 17,0       ; RETURN TO CALLING PROGRAM
;
;                       ERROR RETURN
RETRN:  MOVEI   AC1,^D80        ;MOVE DEFAULT WIDTH
        MOVEM   AC1,@2(16)
        POPJ    17,
;
TTYWRD: Z
;
PTYBIT: BYTE (1)1
;
```

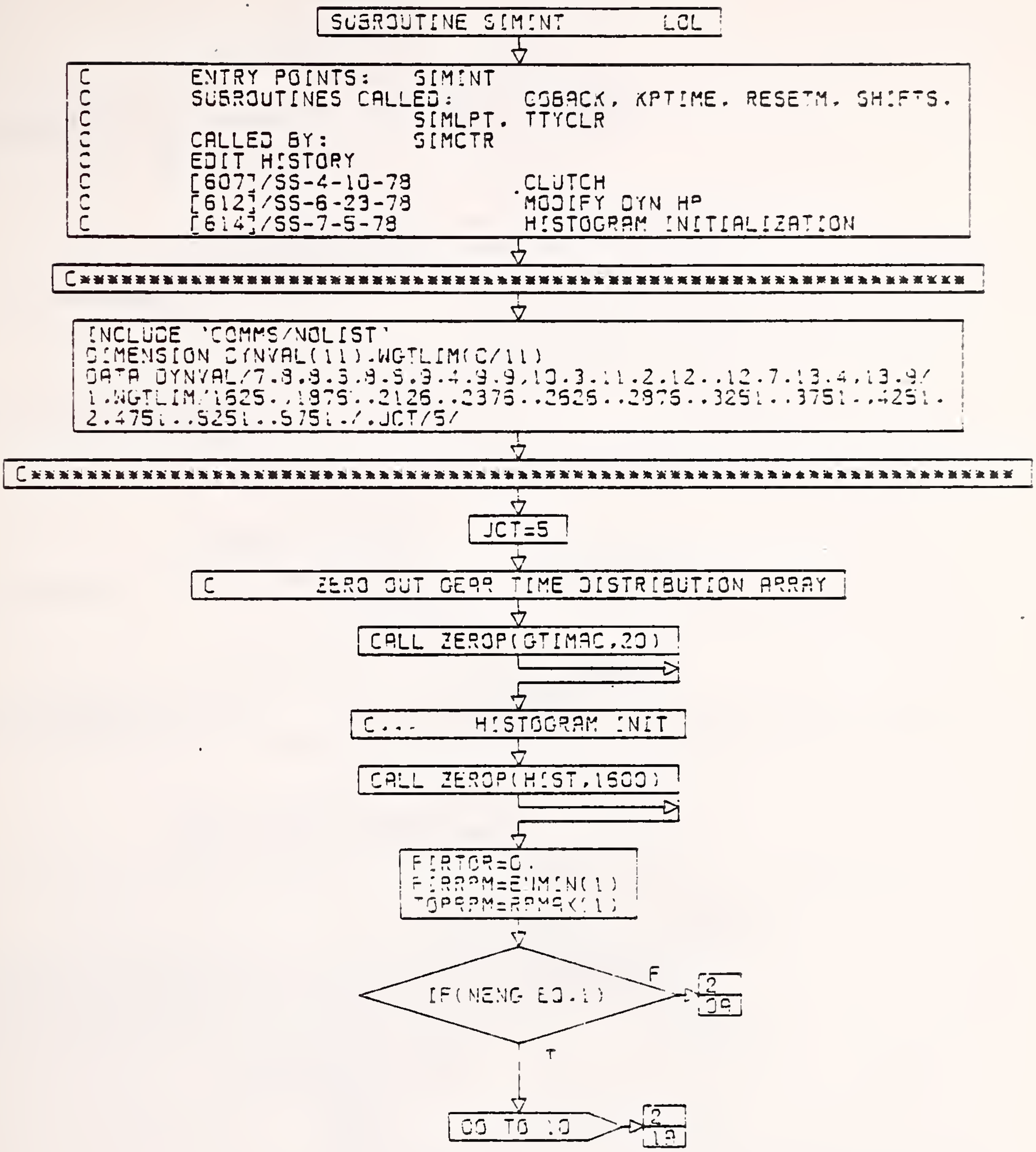
```
SCENIC: .TOPIC
         2
:
         PRGEND
```

```
        TITLE   VALID3
        ENTRY   VALID3
VALID3:  MOVE    SIPTR
        MOVEM   IPTR
        MOVE    SCPTR
        MOVEM   OPTR
        SETZ    1,
        MOVE    @2(16)          ;GET NUMBER OF BYTES
        MOVE    4,@(16)         ;GET ADDR OF SIXBIT NAME
LOOP:   ILDB    2,IPTR          ;GET A BYTE
;...    DON'T LOOP ON NULL BYTE ANY MORE.   TESTING
;       JUMPE   2,LOOP          ;IF NULL BYTE, SKIP THE BYTE
        MOVEI   12,72           ;GET VALUE OF "Z"
        CAMGE   12,2            ;IF("Z" GE BYTE)SKIP
        TDZA    12,12           ;SET 12 TO FALSE AND SKIP
        SETO    12,10           ;SET 12 TO TRUE
        MOVEI   13,41           ;GET VALUE OF "A"
        CAMLE   13,2            ;IF("A" LE BYTE) SKIP
        TDZA    13,13           ;SET 13 TO FALSE AND SKIP
        SETC    13,10           ;SET 13 TO TRUE
        AND     12,13
        MOVEI   13,31           ;GET VALUE FOR "9"
        CAMGE   13,2            ;IF("9" GE BYTE) SKIP
        TDZA    13,13           ;SETKIP
        SETO    13,10           ;SET 13 TRUE
        MOVEI   14,20           ;GET VALUE OF "0"
        CAMLE   14,2            ;IF("0" LE BYTE) SKIP
        TDZA    14,14           ;SET 14 FALSE AND SKIP
        SETC    14,10           ;SET 14 TRUE
        AND     13,14
        OR      12,13
        JUMPL   12,AHEAD        ;JUMP IF TRUE
        JRST    RETURN          ;DONE
AHEAD:  IDPB    2,OPTR          ;PUT BYTE AWAY
        SOJG    LOOP            ;GO GET ANOTHER BYTE
RETURN: MOVEM   1,@0(16)        ;PUT INTO ARGUMENT
        POPJ    17,             ;RETURN
;
SIPTR:  POINT   6,3,35
SCPTR:  POINT   6,0,35
;
IPTR:   Z
OPTR:   Z
;
        PRGEND
        TITLE   ZEROF
        SUBTTL  PROGRAM TO ZERO ARRAY.
        SEARCH  FORPRM

COMMENT %

        WRITTEN BY NORMAN GRANT. CMU. JANUARY 6,1971.
        USAGE           CALL ZEROF(A,N)
        WHERE           A:      IS VECTOR TO BE ZEROED
                        N:      IS NUMBER OF ELEMENTS TO ZERO

%

        A=1
        N=2

        HELLO   (ZEROF, )               ;ZEROF ENTRY
        MOVEI   A,@0(16)                ;GET ADDRESS OF ARRAY A.
        MOVE    N,@1(16)                ;GET VALUE OF N.
        SETZM   0(A)
        CAIG    N,1
        GOODBY  (2)
        HRLZ    0,A
        HRRI    0,1(A)
        ADD     A,N
        BLT     0,-1(A)
        POPJ    P,
        END
```

```
**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


                     * * * L P T S P L   R u n   L o g * * *

17:13:14 LPDAT  [LPTLSJ  LPTSPL version 102(2364) running on LPT012, 22-Apr-81 17:13:14]
17:13:14 LPDAT  [LPTSJS  Starting Job VSMWRK, Seq #1650, request created at 22-Apr-81 16:15:48]
17:14:29 LPMSG  [LPTSTF  Starting File DSKB:VSMWRK.POR<057>[4132,402]]
17:31:52 LPMSG  [LPTFPF  Finished Printing File DSKB:VSMWRK.POR<057>[4132,402]]
17:31:53 LPMSG  [LPTSTF  Starting File DSKB:VEHMAC.MAC<055>[4132,402]]
17:33:20 LPMSG  [LPTCPT  Checkpoint Taken during file DSKB:VEHMAC.MAC<055>[4132,402], copy 1, page 16]
17:34:12 LPMSG  [LPTFPF  Finished Printing File DSKB:VEHMAC.MAC<055>[4132,402]]
17:34:12 LPSUM  Spooler runtime 4 Seconds, 31 KCS, 538 disk reads, 198 pages printed


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


**END** User DCLAN [4132,402]  Job  VSMWRK Seq. 1650 Date 22-Apr-81 17:34:12 Monitor TSC DECsystem10A v8 **END**


0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111
0000000000111111111122222222223333333333444444444455555555556666666666777777777788888888889999999999000000000011111111112222222222333
```

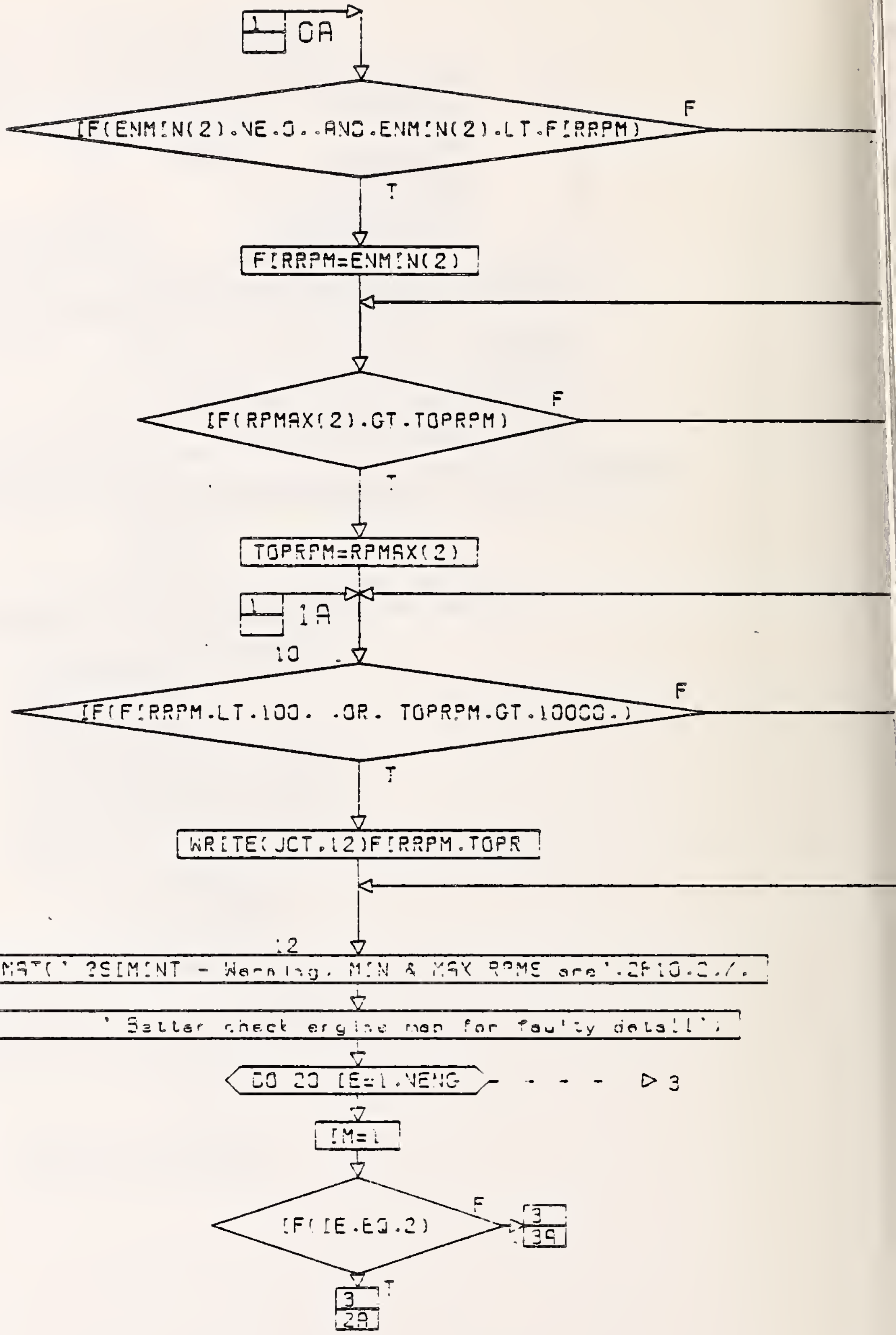
3. FLOW CHARTS

3.1 Subroutine SIMINT

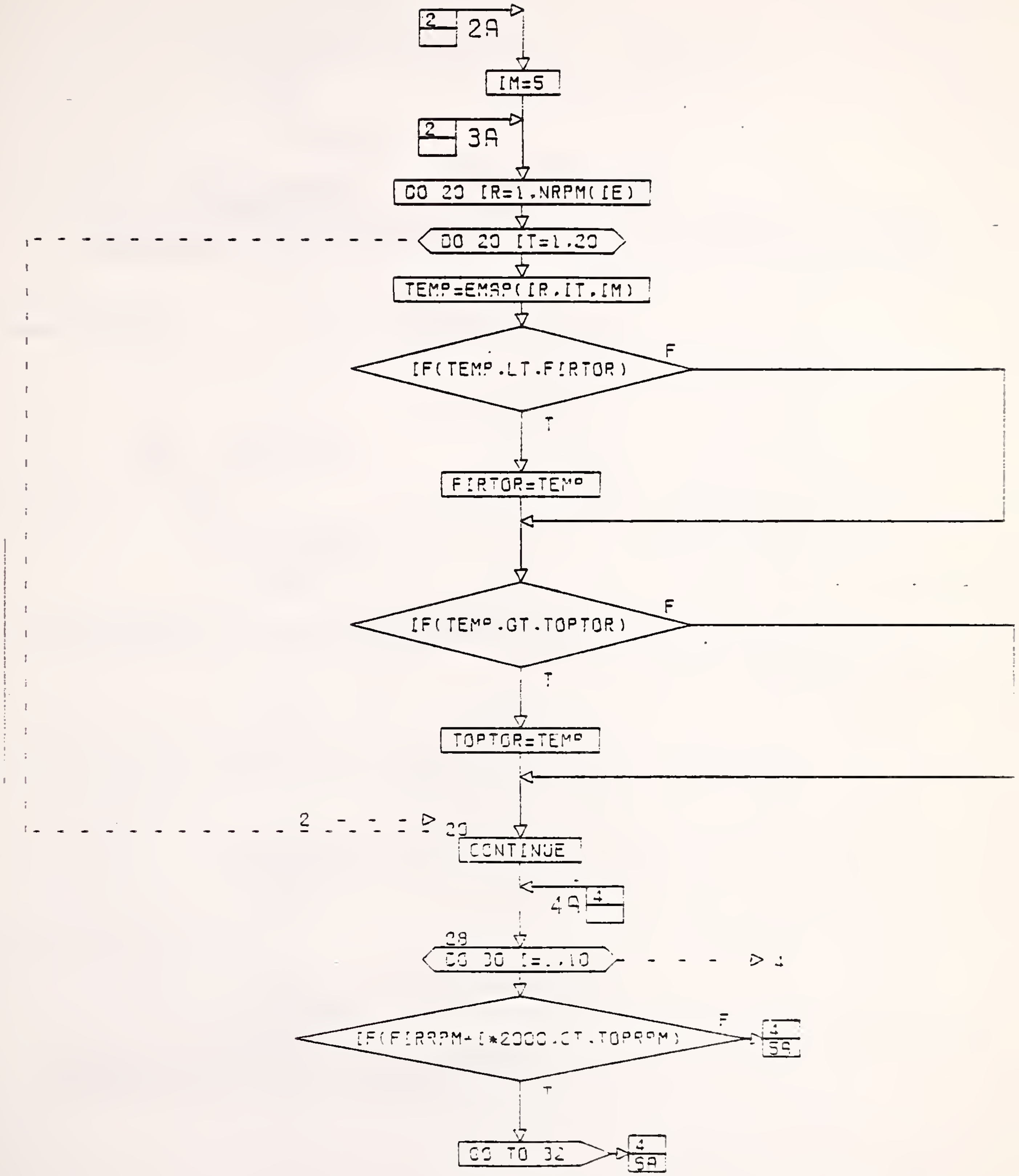


```
SUBROUTINE SIMINT        LCL

C        ENTRY POINTS:    SIMINT
C        SUBROUTINES CALLED:        GOBACK, KPTIME, RESETM, SHIFTS,
C                         SIMLPT, TTYCLR
C        CALLED BY:       SIMCTR
C        EDIT HISTORY
C        [607]/SS-4-10-78          CLUTCH
C        [612]/SS-6-23-78          MODIFY DYN HP
C        [614]/SS-7-5-78           HISTOGRAM INITIALIZATION

C*********************************************************************

INCLUDE 'COMMS/NOLIST'
DIMENSION DYNVAL(11),WGTLIM(0/11)
DATA DYNVAL/7.8,8.3,8.8,9.4,9.9,10.3,11.2,12.,12.7,13.4,13.9/
1,WGTLIM/1625.,1875.,2126.,2376.,2626.,2876.,3251.,3751.,4251.,
2,4751.,5251.,5751./,JCT/5/

C*****************************************************************************

JCT=5

C          ZERO OUT GEAR TIME DISTRIBUTION ARRAY

CALL ZEROP(GTIMAC,20)

C...     HISTOGRAM INIT

CALL ZEROP(HIST,1500)

FIRTOR=0.
FIRRPM=ENMIN(1)
TOPRPM=RPMAX(1)

IF(NENG EQ.1)

GO TO 10
```

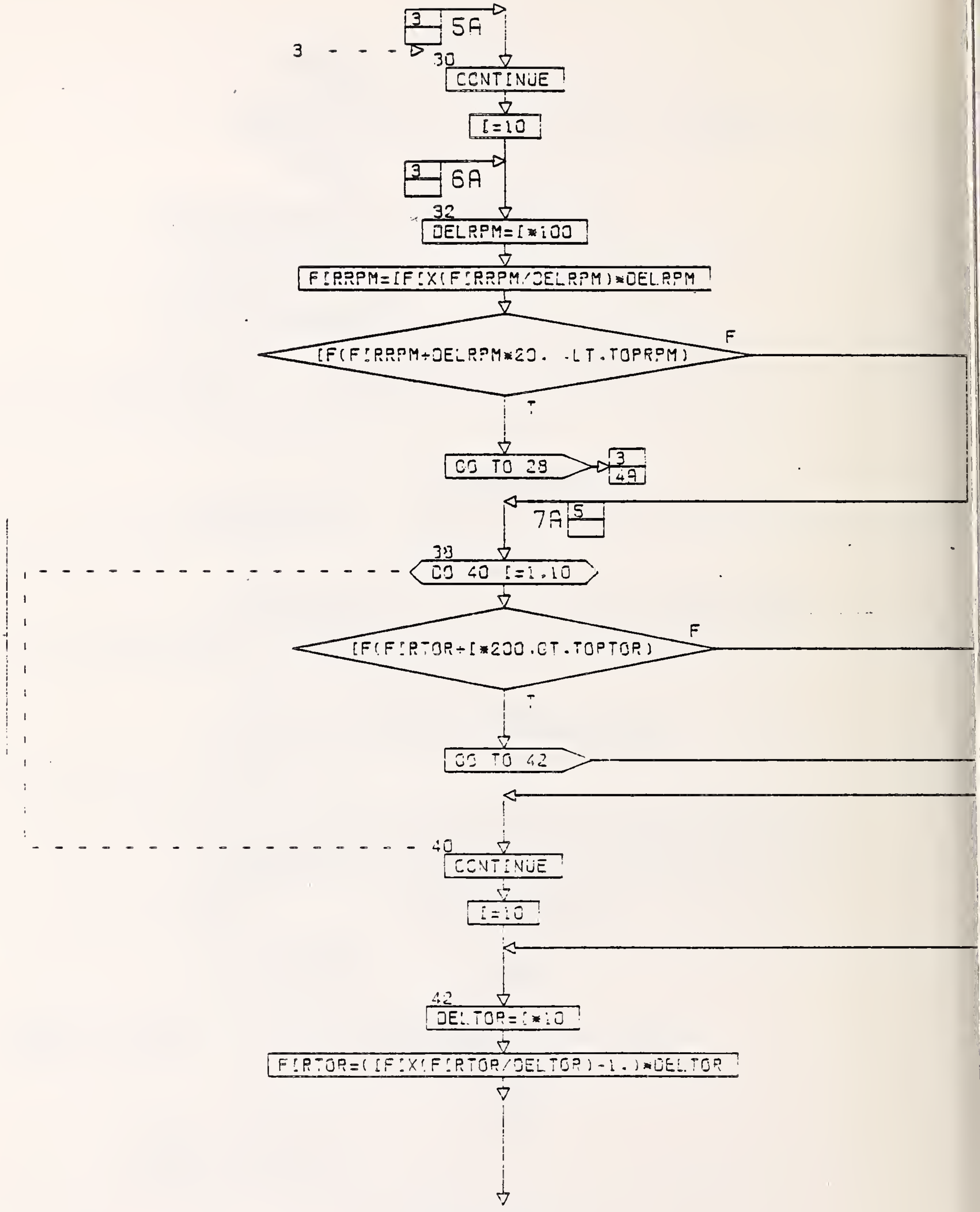
CONT. ON PG 2


1
0A
IF(ENMIN(2).NE.0..AND.ENMIN(2).LT.FIRRPM)
F
T
FIRRPM=ENMIN(2)
IF(RPMAX(2).GT.TOPRPM)
F
T
TOPRPM=RPMAX(2)
1
1A
10
IF(FIRRPM.LT.100. .OR. TOPRPM.GT.10000.)
F
T
WRITE(JCT,12)FIRRPM,TOPR
12
FORMAT(' ?SIMINT - Warning, MIN & MAX RPMS are',2F10.2,/,
2 ' Better check engine map for faulty data!')
DO 20 IE=1,NENG
3
IM=1
IF(IE.EQ.2)
F
3
3B
T
3
2A

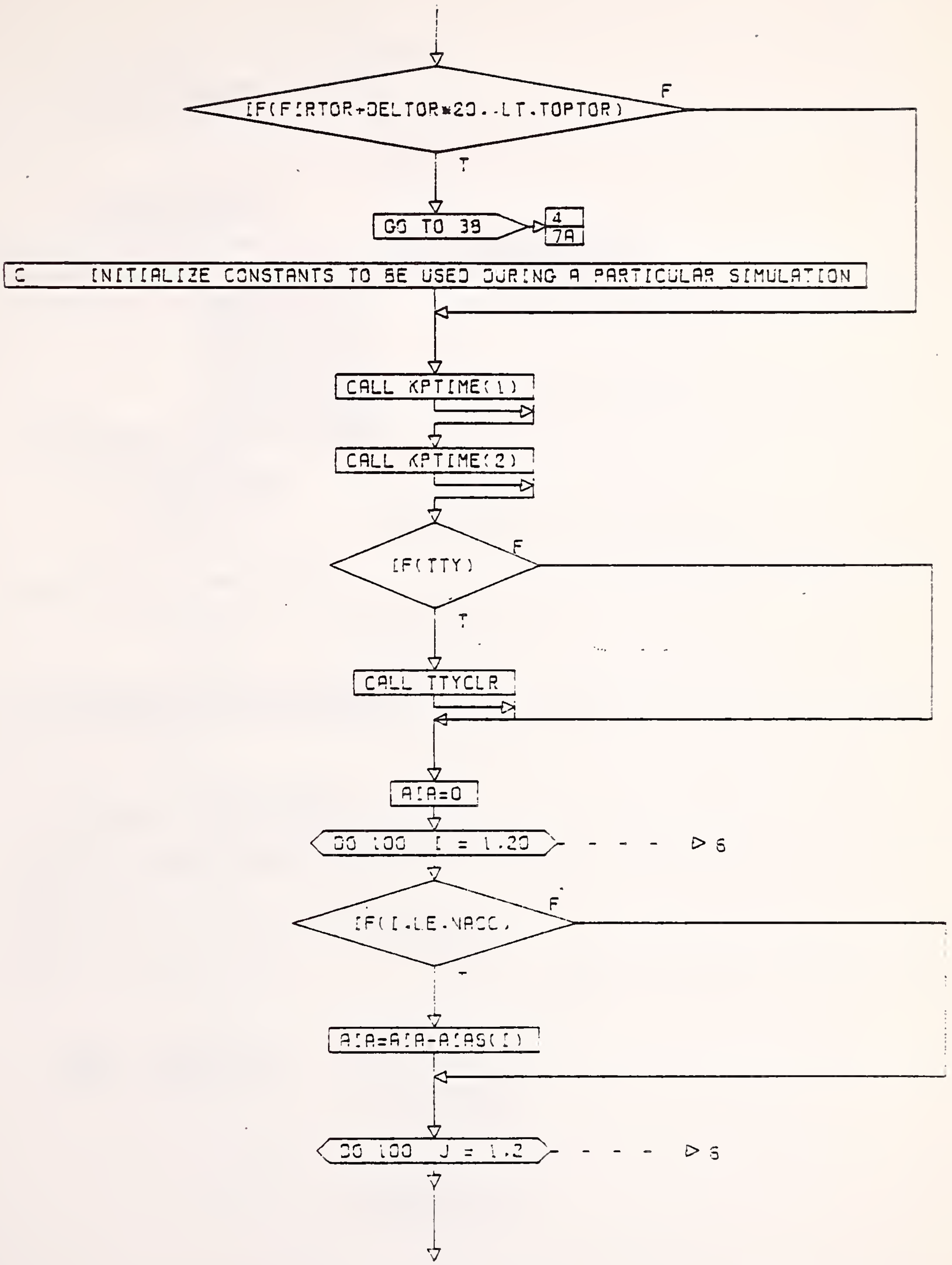
CONT. ON PG 3


2
2A
IM=5
2
3A
DO 20 IR=1,NRPM(IE)
DO 20 IT=1,20
TEMP=EMAP(IR,IT,IM)
IF(TEMP.LT.FIRTOR)
F
T
FIRTOR=TEMP
IF(TEMP.GT.TOPTOR)
F
T
TOPTOR=TEMP
2
20
CONTINUE
4A
4
28
DO 30 I=1,10
4
IF(FIRRPM+I*2000.GT.TOPRPM)
F
4
5A
T
GO TO 32
4
5A

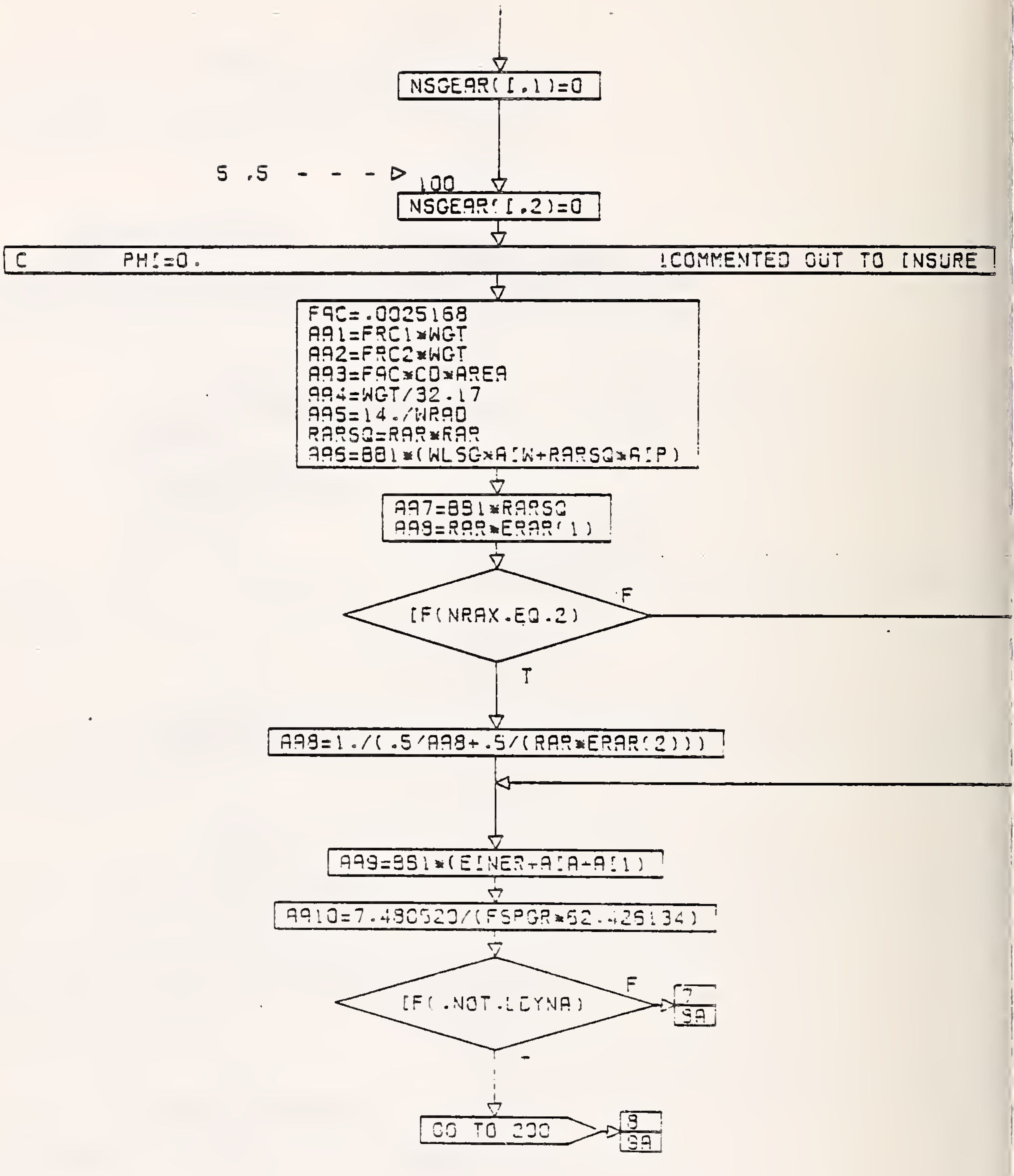
CONT. ON PG 4


3
5A
3
30
CONTINUE
I=10
3
6A
32
DELRPM=I*100
FIRRPM=IFIX(FIRRPM/DELRPM)*DELRPM
IF(FIRRPM+DELRPM*20. .LT.TOPRPM)
F
T
GO TO 28
3
4A
7A
5
38
DO 40 I=1,10
IF(FIRTOR+I*200.GT.TOPTOR)
F
T
GO TO 42
40
CONTINUE
I=10
42
DELTOR=I*10
FIRTOR=(IFIX(FIRTOR/DELTOR)-1.)*DELTOR

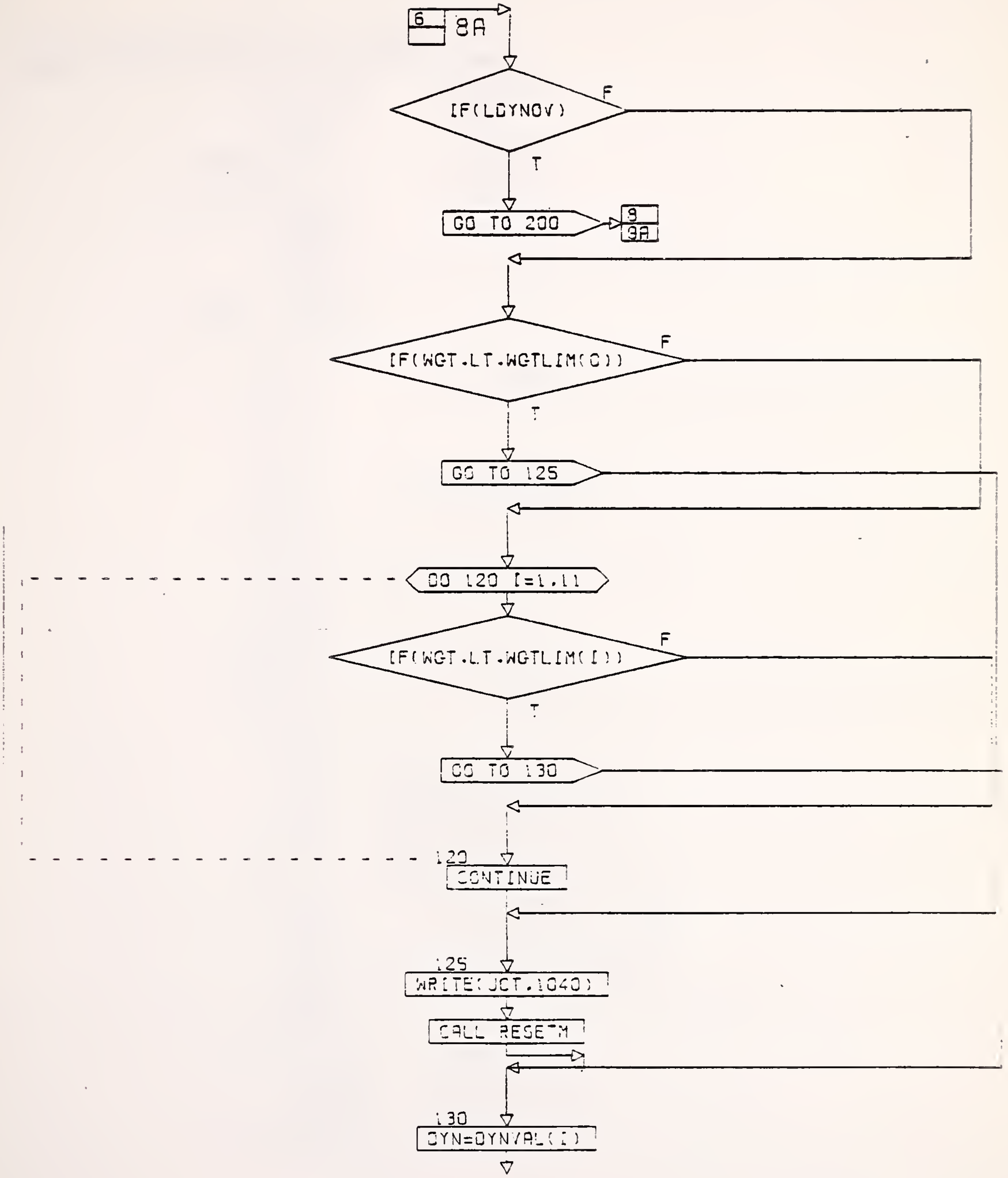
CONT. ON PG 5


IF(FIRTOR+DELTOR*20..LT.TOPTOR)
F
T
GO TO 38
4
7A
C INITIALIZE CONSTANTS TO BE USED DURING A PARTICULAR SIMULATION
CALL KPTIME(1)
CALL KPTIME(2)
IF(TTY)
F
T
CALL TTYCLR
AIA=0
DO 100 I = 1.20
6
IF(I.LE.NACC)
F
T
AIA=AIA-AIAS(I)
DO 100 J = 1.2
6

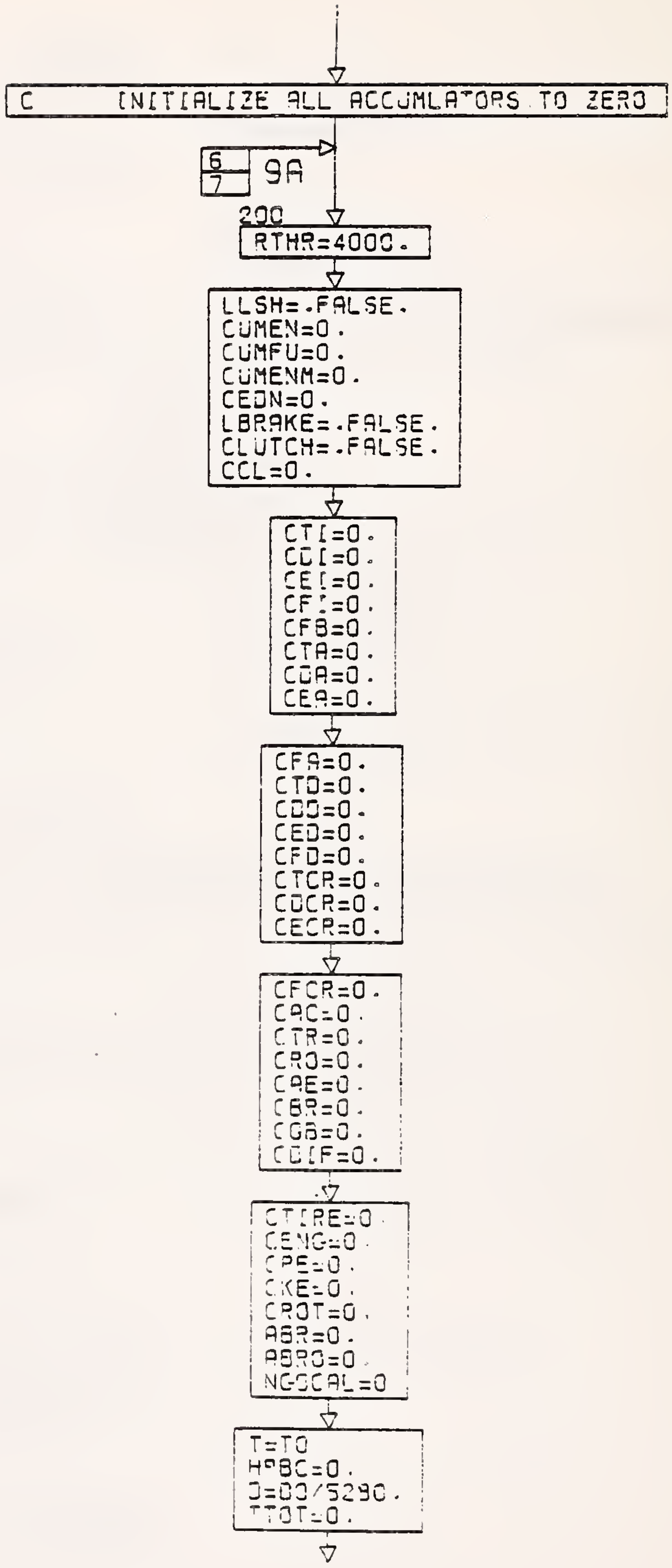
CONT. ON PG 6


NSGEAR(I,1)=0
5 ,5
100
NSGEAR(I,2)=0
C PHI=0. !COMMENTED OUT TO INSURE
FAC=.0025168
AA1=FRC1*WGT
AA2=FRC2*WGT
AA3=FAC*CD*AREA
AA4=WGT/32.17
AA5=14./WRAD
RARSQ=RAR*RAR
AA6=881*(WLSG*AIW+RARSQ*AIP)
AA7=881*RARSQ
AA8=RAR*ERAR(1)
IF(NRAX.EQ.2)
F
T
AA8=1./(.5/AA8+.5/(RAR*ERAR(2)))
AA9=881*(EINER+AIA-AI1)
AA10=7.480520/(FSPGR*62.426134)
IF(.NOT.LDYNA)
F
7
9A
GO TO 200
8
9A

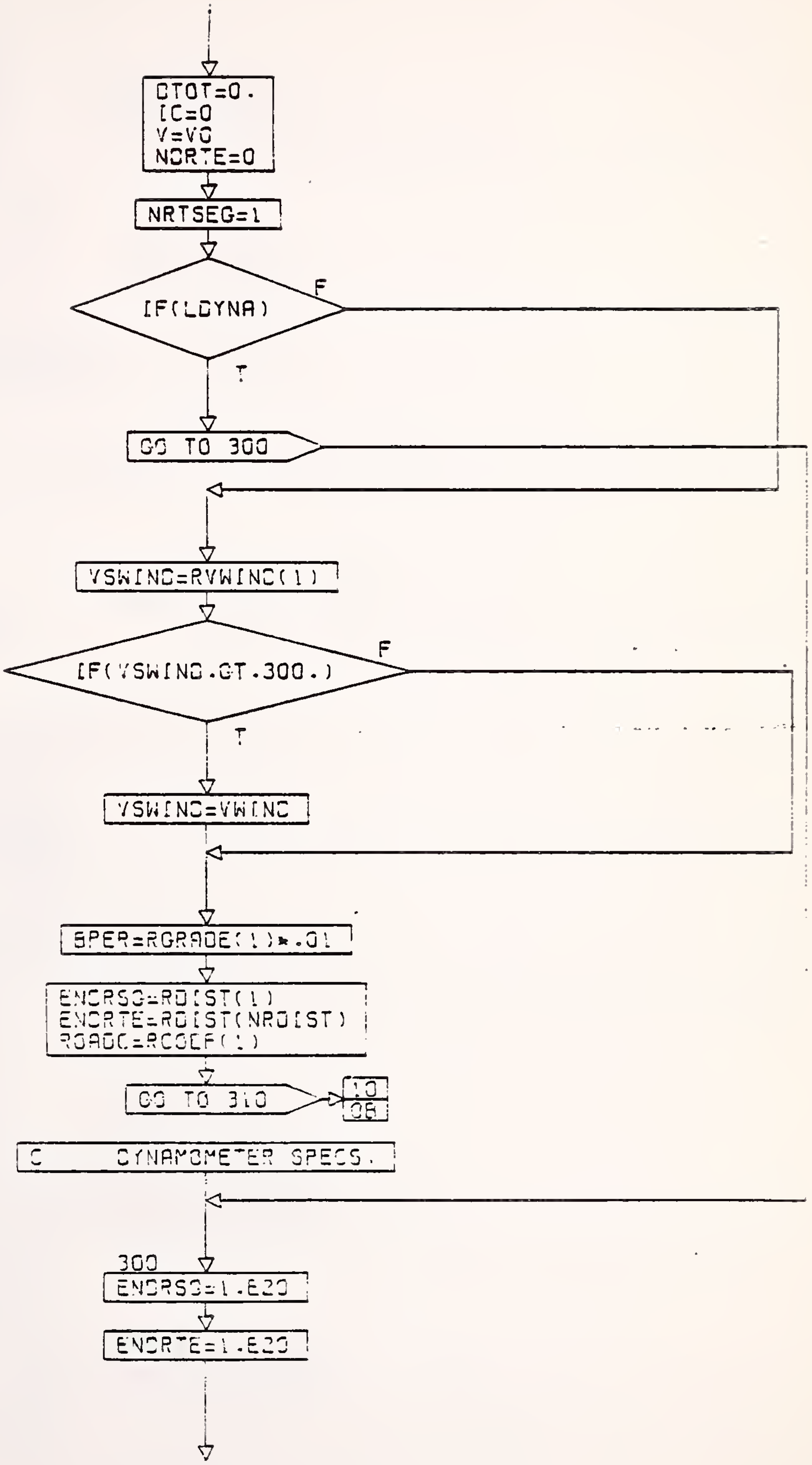
CONT. ON PG 7


6
8A
IF(LDYNOV)
F
T
GO TO 200
9
9A
IF(WGT.LT.WGTLIM(0))
F
T
GO TO 125
DO 120 I=1,11
IF(WGT.LT.WGTLIM(I))
F
T
GO TO 130
120
CONTINUE
125
WRITE(JCT,1040)
CALL RESETM
130
DYN=DYNVAL(I)

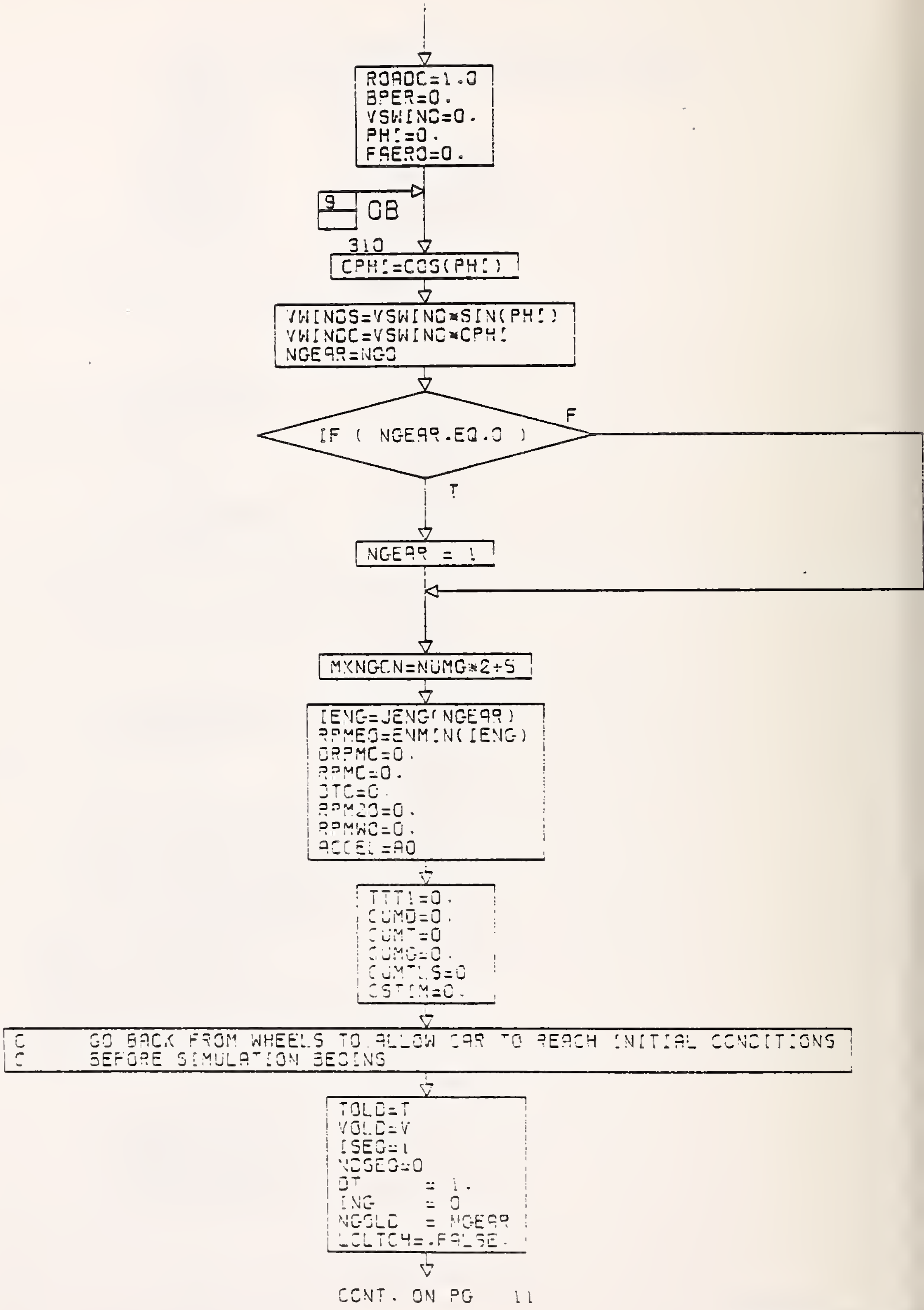
CONT. ON PG 9


C INITIALIZE ALL ACCUMLATORS TO ZERO
6
7
9A
200
RTHR=4000.
LLSH=.FALSE.
CUMEN=0.
CUMFU=0.
CUMENM=0.
CEDN=0.
LBRAKE=.FALSE.
CLUTCH=.FALSE.
CCL=0.
CTI=0.
CDI=0.
CEI=0.
CFI=0.
CFB=0.
CTA=0.
CDA=0.
CEA=0.
CFA=0.
CTD=0.
CDD=0.
CED=0.
CFD=0.
CTCR=0.
CDCR=0.
CECR=0.
CFCR=0.
CAC=0.
CTR=0.
CRO=0.
CAE=0.
CBR=0.
CGB=0.
CDIF=0.
CTIRE=0.
CENG=0.
CPE=0.
CKE=0.
CROT=0.
ABR=0.
ABRO=0.
NGSCAL=0
T=TO
HPBC=0.
D=DD/5280.
TTOT=0.

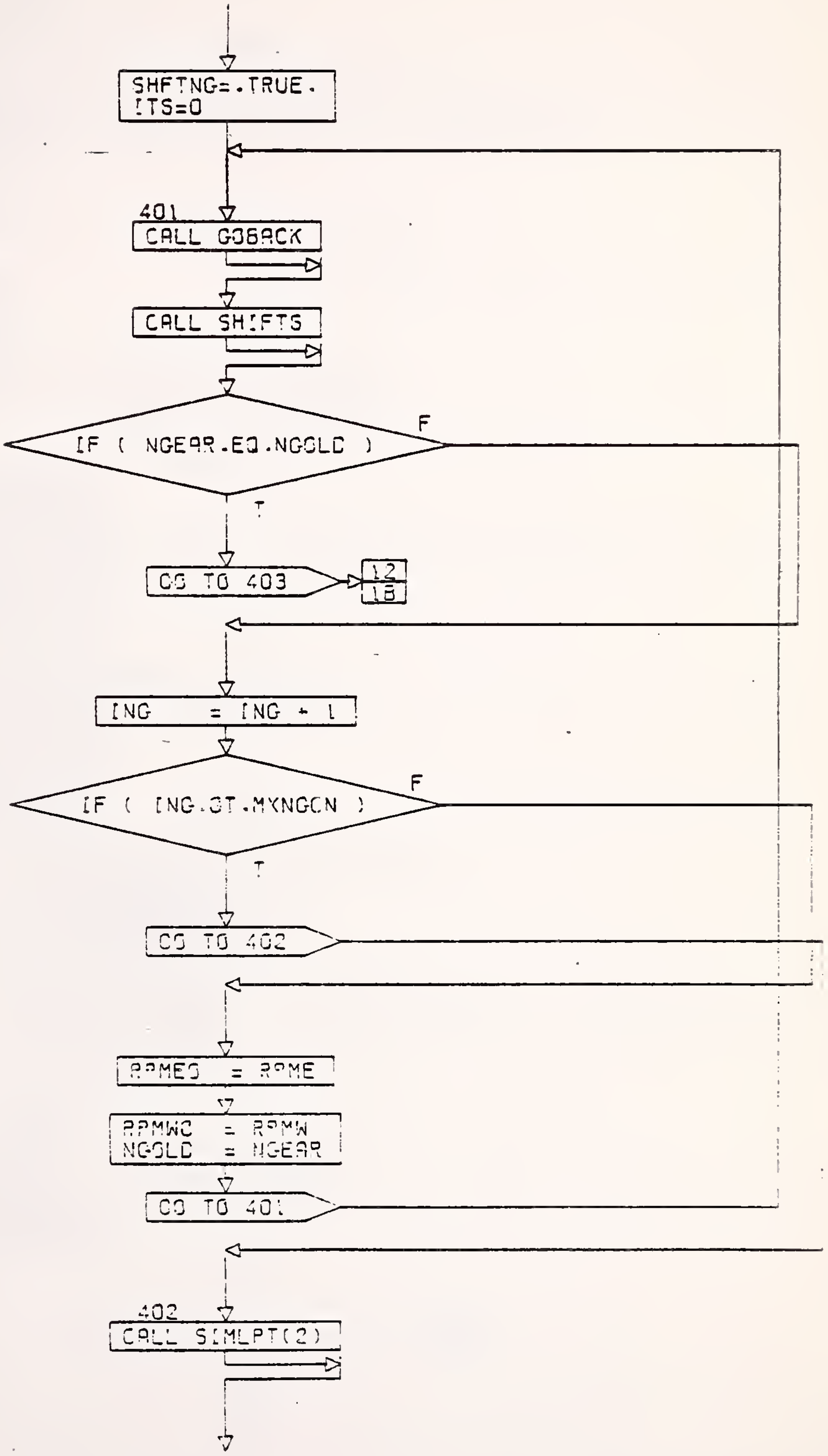
CONT. ON PG 9


CTOT=0.
IC=0
V=VO
NORTE=0
NRTSEG=1
IF(LDYNA)
F
T
GO TO 300
VSWIND=RVWIND(1)
IF(VSWIND.GT.300.)
F
T
VSWIND=VWIND
GPER=RGRADE(1)*.01
ENDRSG=RDIST(1)
ENDRTE=RDIST(NRDIST)
ROADC=RCOEF(1)
GO TO 310
10
06
C DYNAMOMETER SPECS.
300
ENDRSG=1.E20
ENDRTE=1.E20

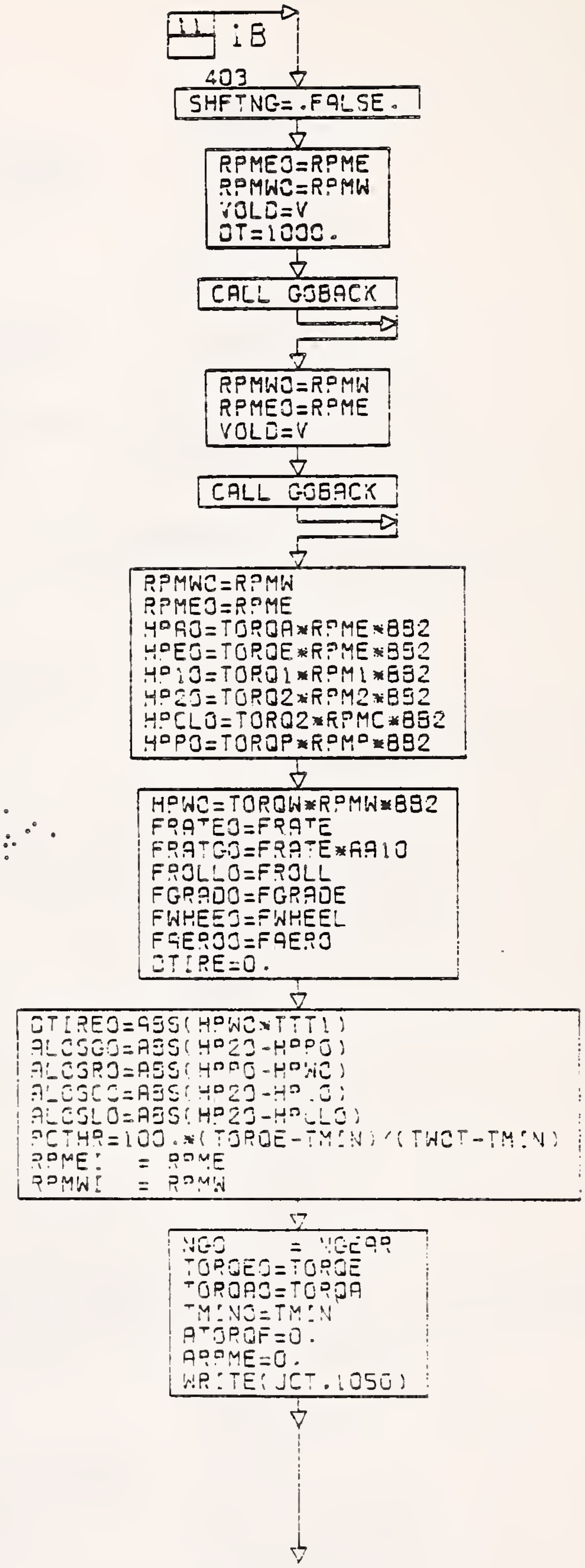
CONT. ON PG 10


ROADC=1.0
BPER=0.
VSWIND=0.
PHI=0.
FAERO=0.
9
OB
310
CPHI=COS(PHI)
VWINDS=VSWIND*SIN(PHI)
VWINDC=VSWIND*CPHI
NGEAR=NGO
F
IF (NGEAR.EQ.0)
T
NGEAR = 1
MXNGCN=NUMG*2+5
IENG=JENG(NGEAR)
RPMEO=ENMIN(IENG)
DRPMC=0.
RPMC=0.
DTC=0.
RPM20=0.
RPMWC=0.
ACCEL=A0
TTT1=0.
CUMD=0.
CUMT=0
CUMG=0.
CUMTLS=0
CSTIM=0.
C GO BACK FROM WHEELS TO ALLOW CAR TO REACH INITIAL CONDITIONS
C BEFORE SIMULATION BEGINS
TOLD=T
VOLD=V
ISEG=1
NOSEG=0
DT = 1.
ING = 0
NGOLD = NGEAR
LCLTCH=.FALSE.

CONT. ON PG 11

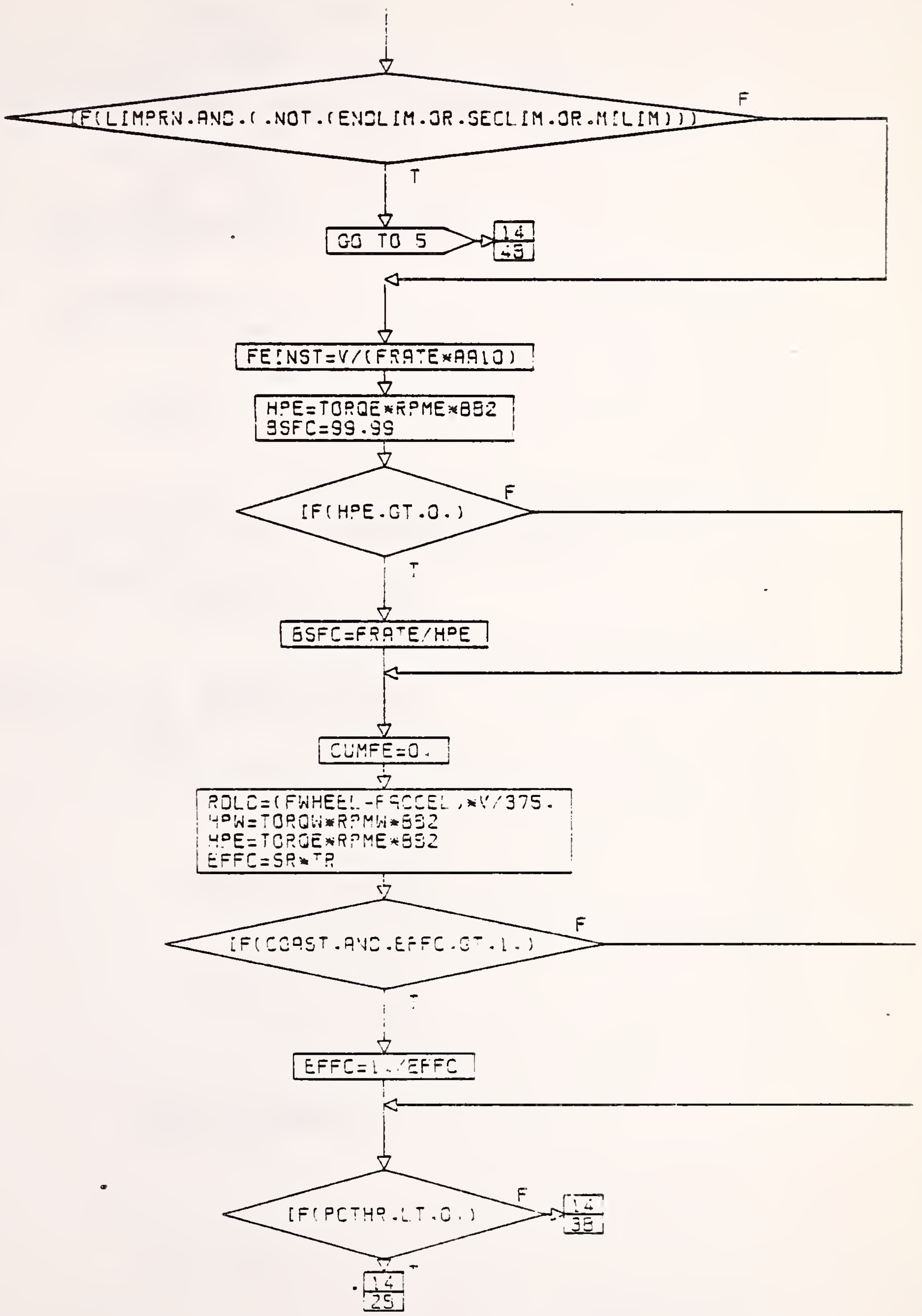

SHFTNG=.TRUE.
ITS=0
401
CALL GOBACK
CALL SHIFTS
IF (NGEAR.EQ.NGOLD)
F
T
GO TO 403
12
1B
ING = ING + 1
IF (ING.GT.MXNGON)
F
T
GO TO 402
RPMEO = RPME
RPMWO = RPMW
NGOLD = NGEAR
GO TO 401
402
CALL SIMLPT(2)

CONT. ON PG 12

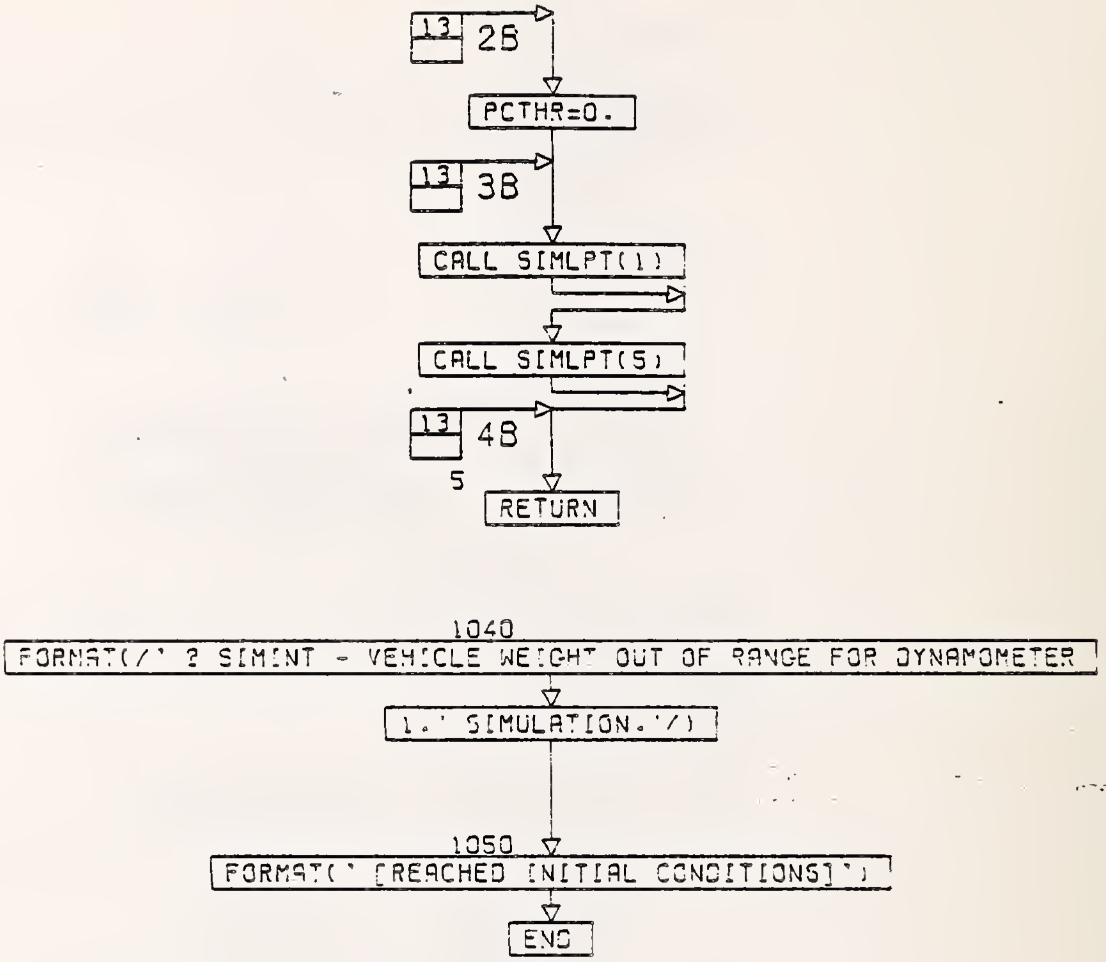


```
SHFTNG=.FALSE.

RPMEO=RPME
RPMWO=RPMW
VOLD=V
DT=1000.

CALL GOBACK

RPMWO=RPMW
RPMEO=RPME
VOLD=V

CALL GOBACK

RPMWO=RPMW
RPMEO=RPME
HPAO=TORQA*RPME*882
HPEO=TORQE*RPME*882
HP1O=TORQ1*RPM1*882
HP2O=TORQ2*RPM2*882
HPCLO=TORQ2*RPMC*882
HPPO=TORQP*RPMP*882

HPWO=TORQW*RPMW*882
FRATEO=FRATE
FRATGO=FRATE*AA10
FROLLO=FROLL
FGRADO=FGRADE
FWHEEO=FWHEEL
FAEROO=FAERO
OTIRE=0.

OTIREO=ABS(HPWO*TTT1)
ALOSGO=ABS(HP2O-HPPO)
ALOSRO=ABS(HPPO-HPWO)
ALOSCO=ABS(HP2O-HP.O)
ALOSLO=ABS(HP2O-HPCLO)
PCTHR=100.*(TORQE-TMIN)/(TWOT-TMIN)
RPMEI   = RPME
RPMWI   = RPMW

NGO      = NGEAR
TORQEO=TORQE
TORQAO=TORQA
TMINO=TMIN
ATORQF=0.
ARPME=0.
WRITE(JCT.1050)
```

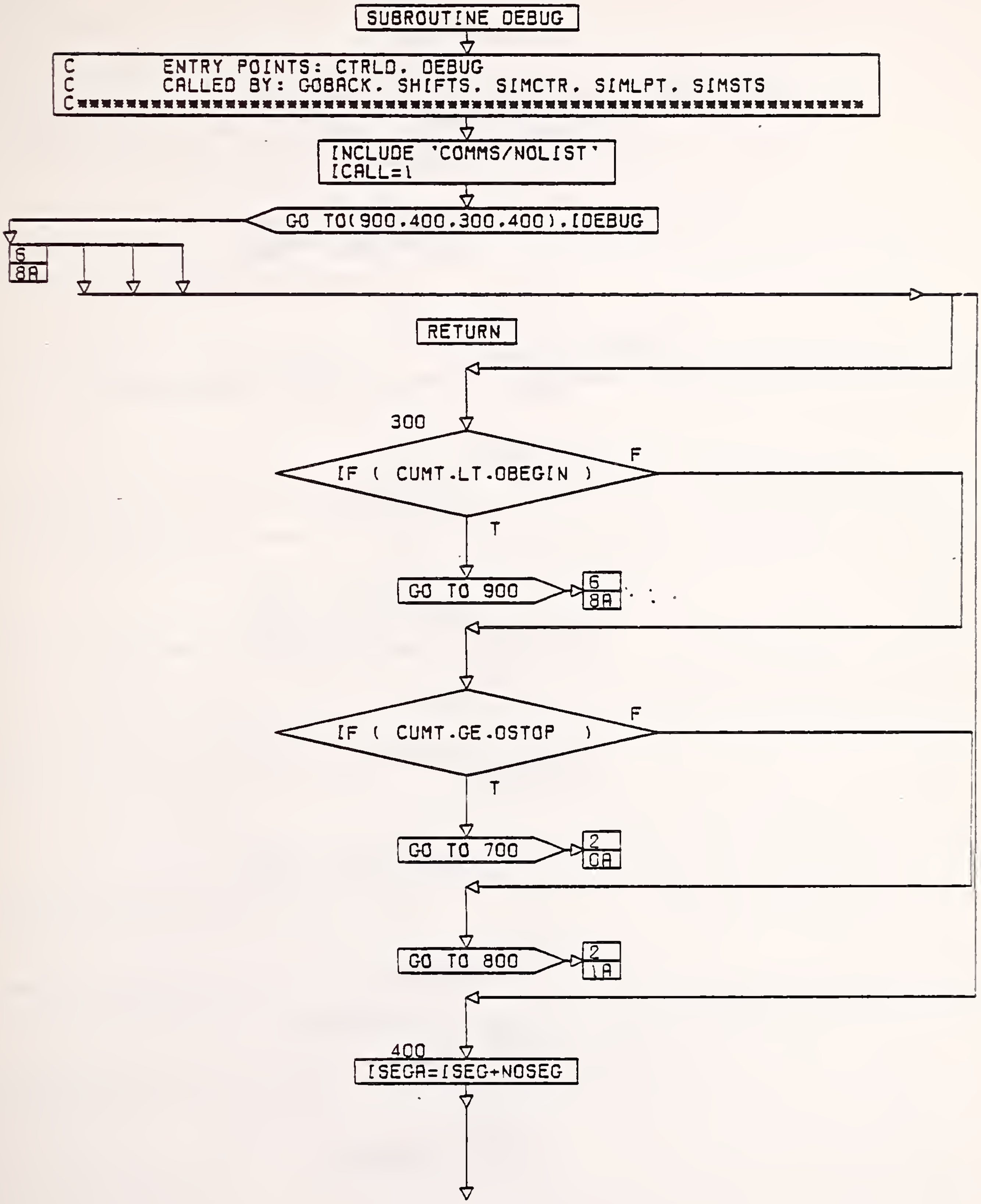
CONT. ON PG 13


IF(LIMPRN.AND.(.NOT.(ENDLIM.OR.SECLIM.OR.MILIM)))
F
T
GO TO 5
14
45
FEINST=V/(FRATE*AA10)
HPE=TORQE*RPME*BB2
BSFC=99.99
IF(HPE.GT.0.)
F
T
BSFC=FRATE/HPE
CUMFE=0.
ROLC=(FWHEEL-FACCEL)*V/375.
HPW=TORQW*RPMW*BB2
HPE=TORQE*RPME*BB2
EFFC=SR*TR
IF(COAST.AND.EFFC.GT.1.)
F
T
EFFC=1./EFFC
IF(PCTHR.LT.0.)
F
14
38
T
14
25

CONT. ON PG 14

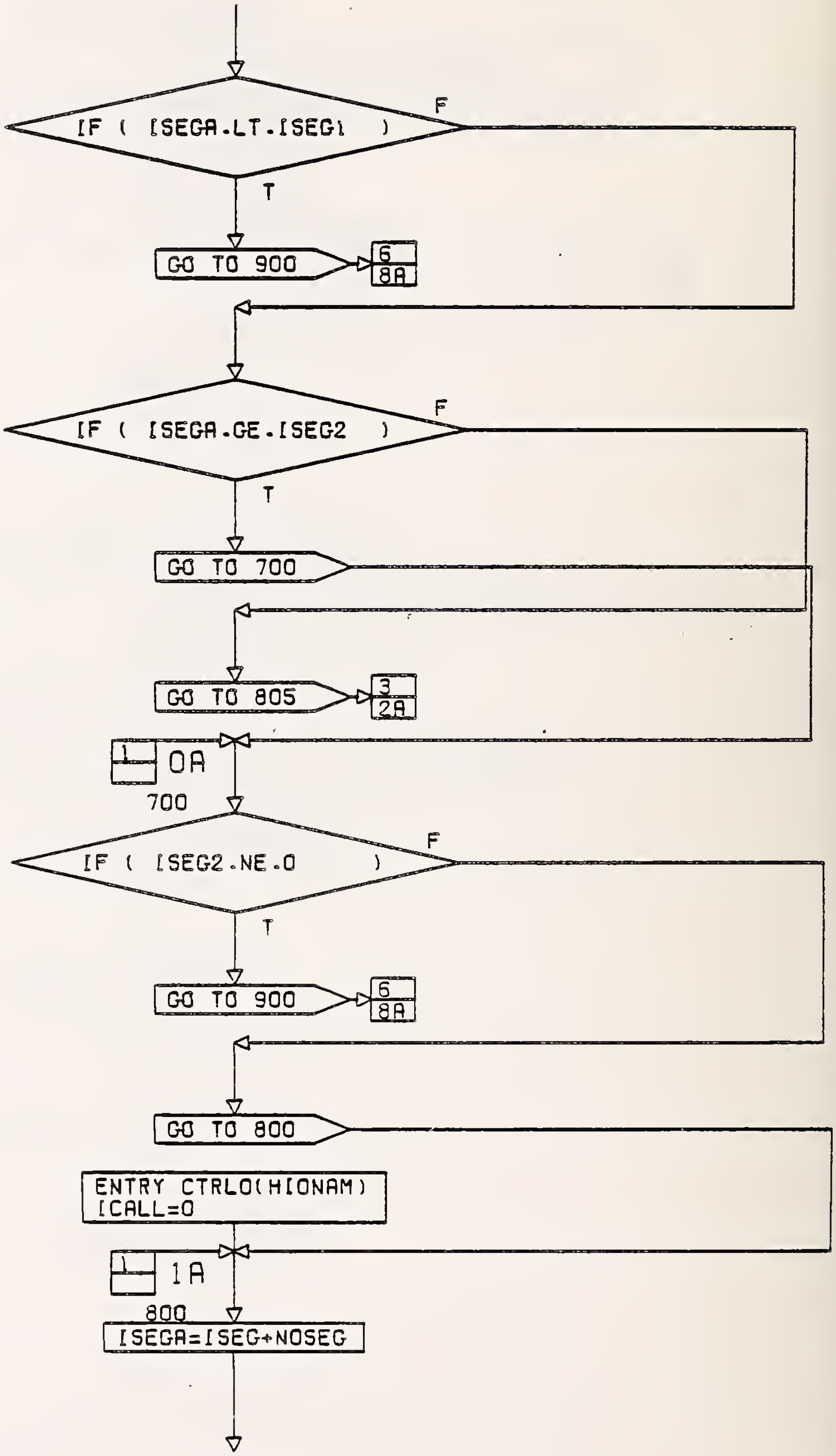

13
2B
PCTHR=0.
13
3B
CALL SIMLPT(1)
CALL SIMLPT(5)
13
4B
5
RETURN
1040
FORMAT(/' 2 SIMINT - VEHICLE WEIGHT OUT OF RANGE FOR DYNAMOMETER
1.' SIMULATION.'/)
1050
FORMAT(' [REACHED INITIAL CONDITIONS]')
END

3.2 Subroutine DEBUG

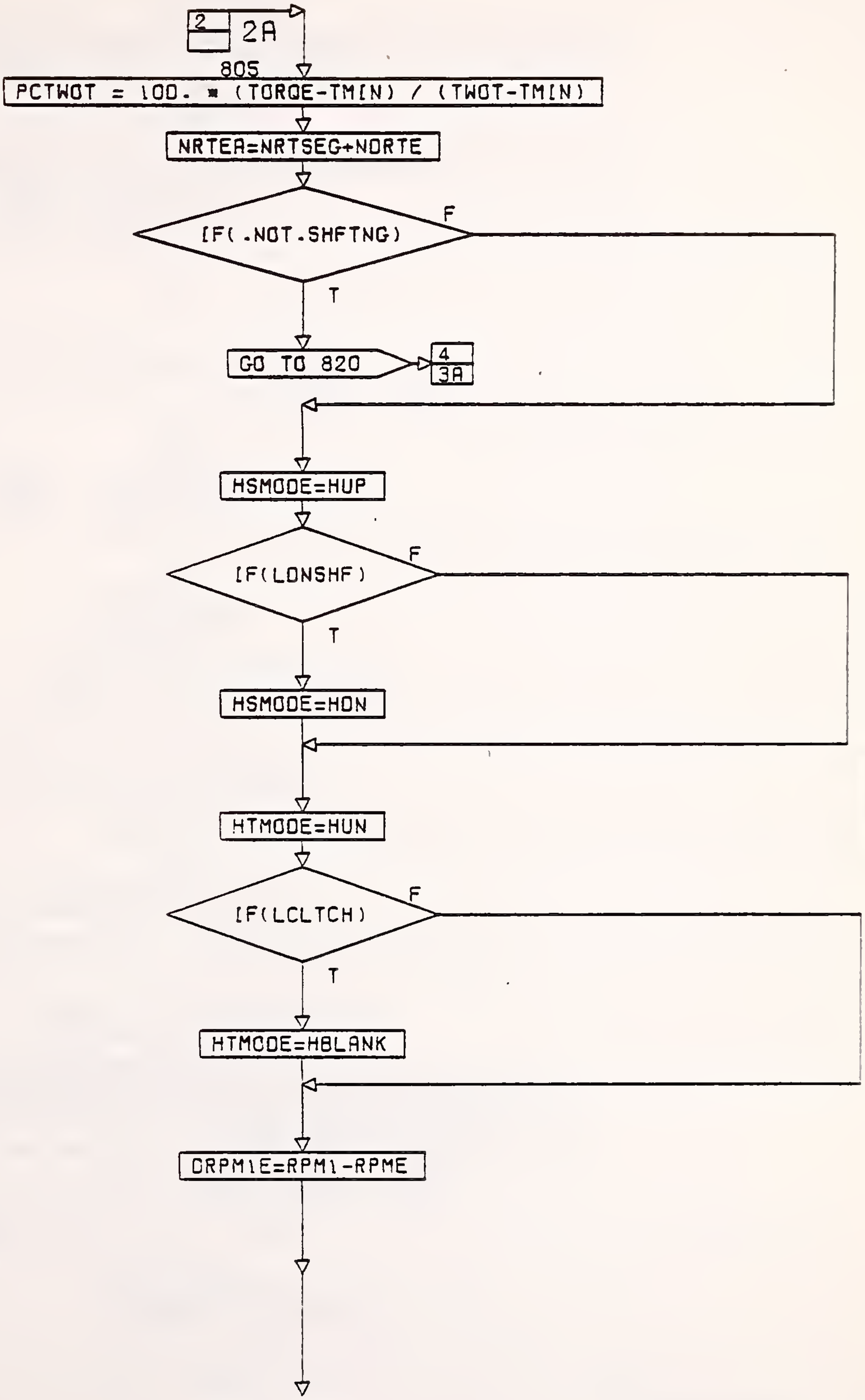

SUBROUTINE DEBUG
C ENTRY POINTS: CTRLD. DEBUG
C CALLED BY: GOBACK. SHIFTS. SIMCTR. SIMLPT. SIMSTS
C***
INCLUDE 'COMMS/NOLIST'
ICALL=1
GO TO(900.400.300.400).IDEBUG
6
8A
RETURN
300
IF (CUMT.LT.OBEGIN)
F
T
GO TO 900
6
8A
IF (CUMT.GE.OSTOP)
F
T
GO TO 700
2
0A
GO TO 800
2
1A
400
ISEGA=ISEG+NOSEG

CONT. ON PG 2

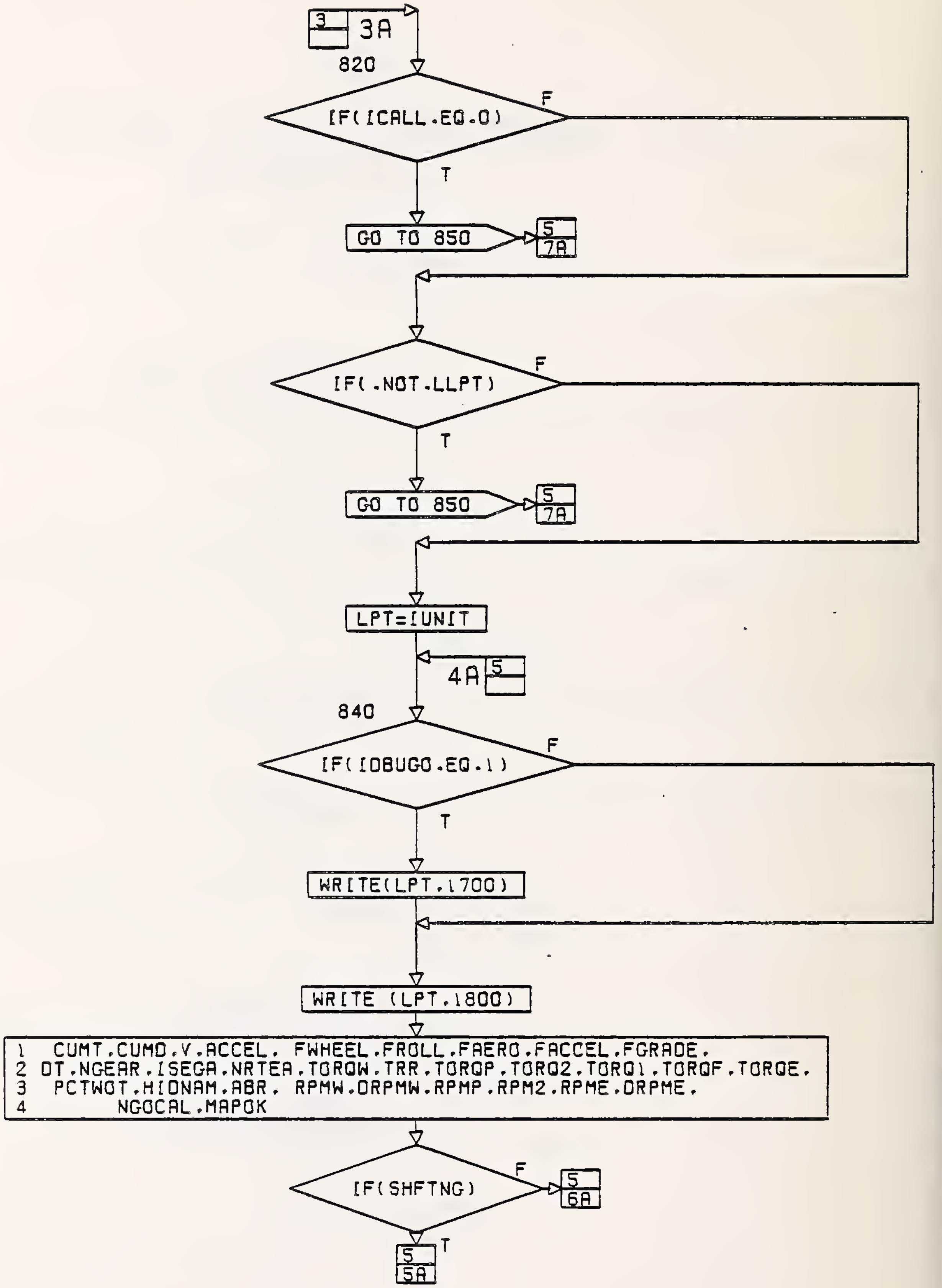
PG 1 OF 7


IF (ISEGA.LT.ISEG1)
F
T
GO TO 900
6
8A
IF (ISEGA.GE.ISEG2)
F
T
GO TO 700
GO TO 805
3
2A
1
0A
700
IF (ISEG2.NE.0)
F
T
GO TO 900
6
8A
GO TO 800
ENTRY CTRLO(HIONAM)
ICALL=0
1
1A
800
ISEGA=ISEG+NOSEG

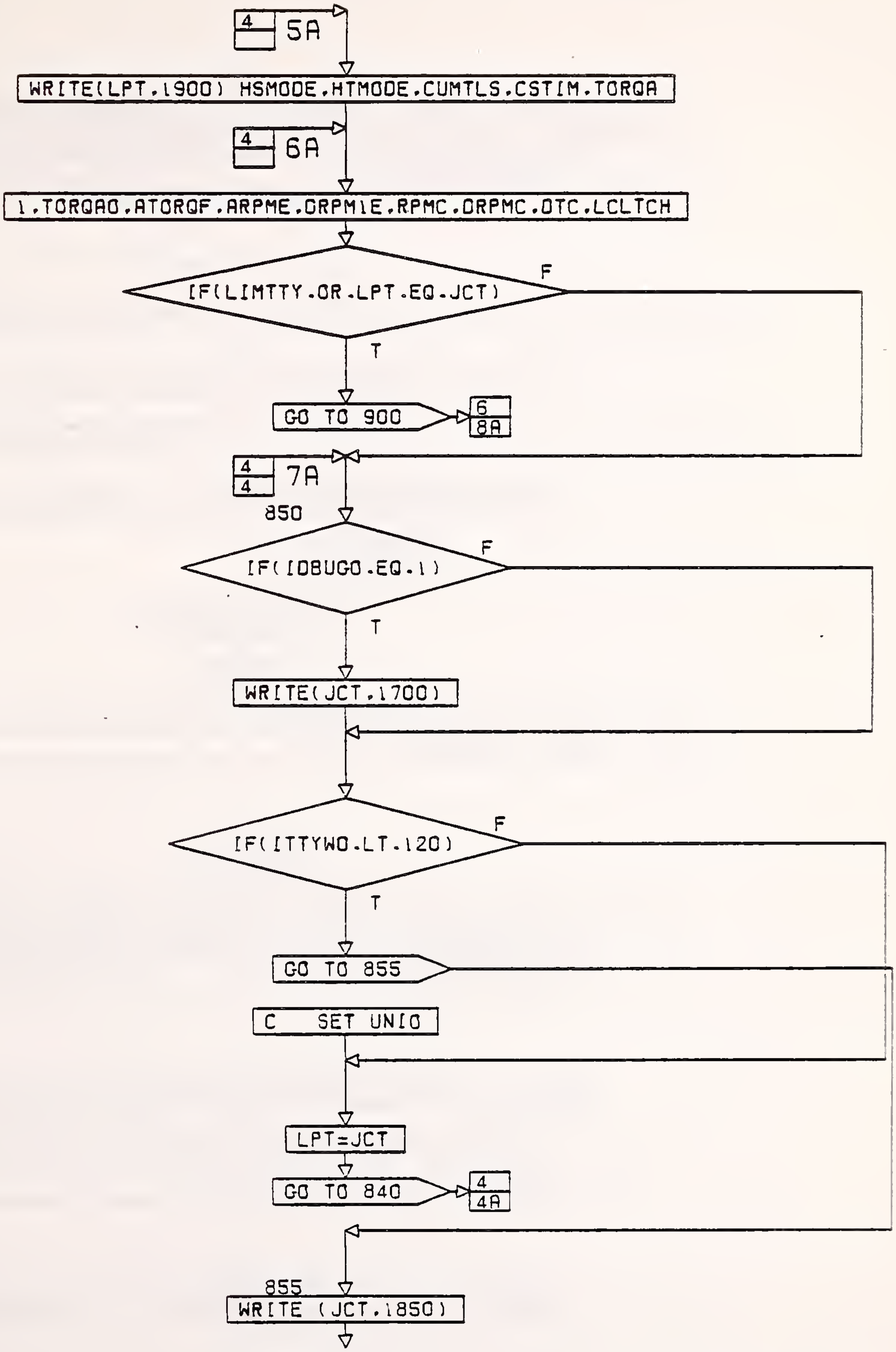
CONT. ON PG 3


2
2A
805
PCTWOT = 100. * (TORQE-TMIN) / (TWOT-TMIN)
NRTEA=NRTSEG+NORTE
IF(.NOT.SHFTNG)
F
T
GO TO 820
4
3A
HSMODE=HUP
IF(LONSHF)
F
T
HSMODE=HON
HTMODE=HUN
IF(LCLTCH)
F
T
HTMODE=HBLANK
DRPM1E=RPM1-RPME

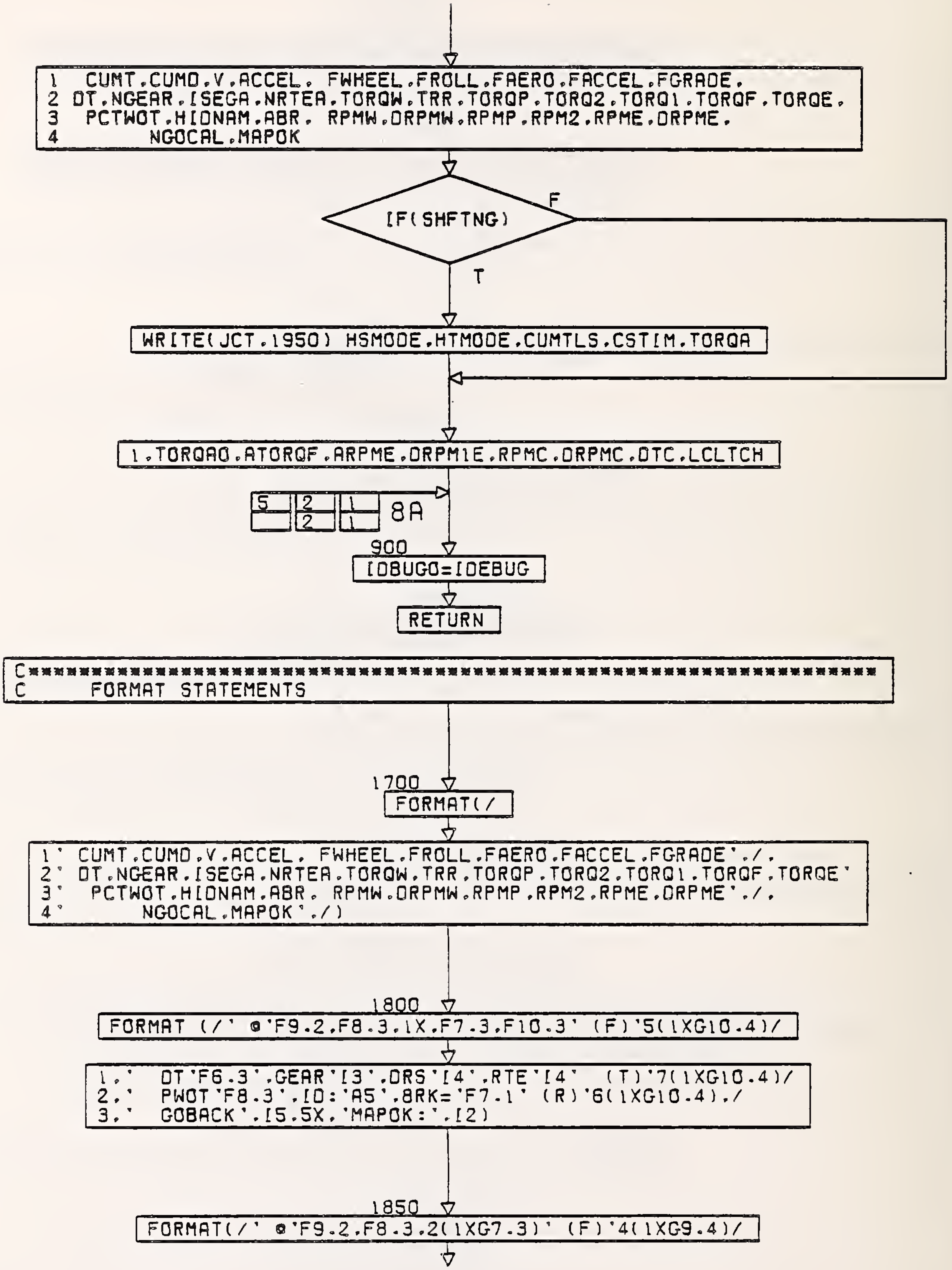
CONT. ON PG 4


3
3A
820
IF(ICALL.EQ.0)
F
T
GO TO 850
5
7A
IF(.NOT.LLPT)
F
T
GO TO 850
5
7A
LPT=IUNIT
4A
5
840
IF(IDBUGO.EQ.1)
F
T
WRITE(LPT,1700)
WRITE (LPT,1800)
1 CUMT.CUMD.V.ACCEL. FWHEEL.FROLL.FAERO.FACCEL.FGRADE.
2 DT.NGEAR.ISEGA.NRTEA.TORQW.TRR.TORQP.TORQ2.TORQ1.TORQF.TORQE.
3 PCTWOT.HIDNAM.ABR. RPMW.DRPMW.RPMP.RPM2.RPME.DRPME.
4 NGOCAL.MAPOK
IF(SHFTNG)
F
5
6A
T
5
5A

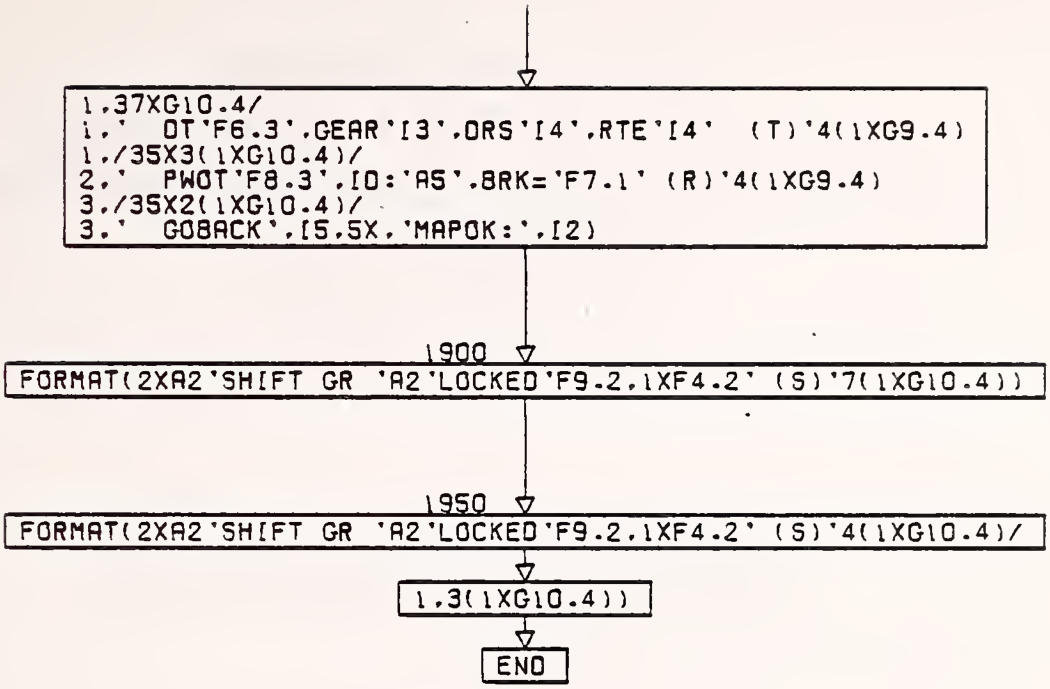
CONT. ON PG 5


4
5A
WRITE(LPT.1900) HSMODE.HTMODE.CUMTLS.CSTIM.TORQA
4
6A
1.TORQAO.ATORQF.ARPME.DRPM1E.RPMC.DRPMC.DTC.LCLTCH
IF(LIMTTY.OR.LPT.EQ.JCT)
F
T
GO TO 900
6
8A
4
4
7A
850
IF(IDBUGO.EQ.1)
F
T
WRITE(JCT.1700)
IF(ITTYWO.LT.120)
F
T
GO TO 855
C SET UNIO
LPT=JCT
GO TO 840
4
4A
855
WRITE (JCT.1850)

CONT. ON PG 6

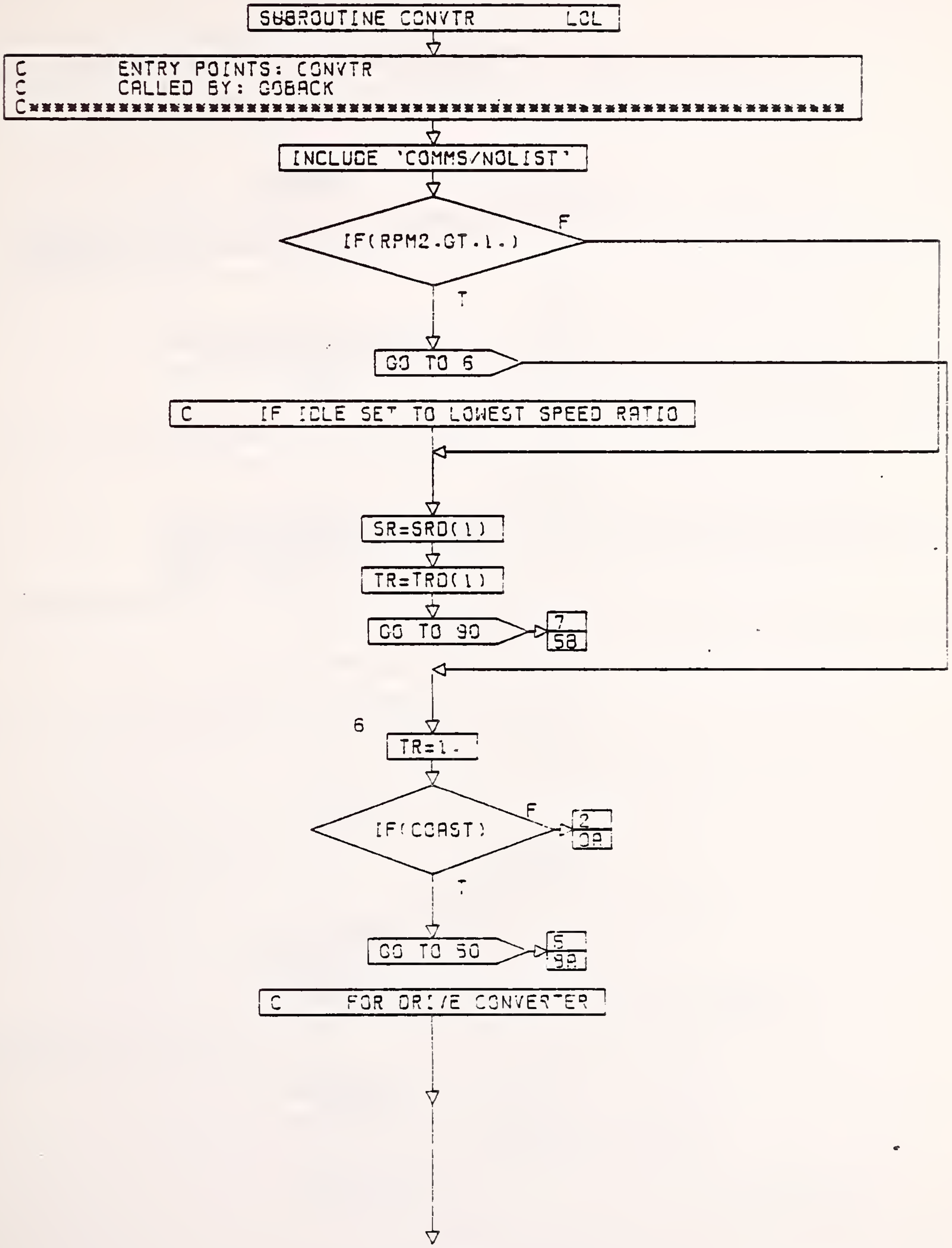

1 CUMT.CUMD.V.ACCEL. FWHEEL.FROLL.FAERO.FACCEL.FGRADE.
2 DT.NGEAR.ISEGA.NRTEA.TORQW.TRR.TORQP.TORQ2.TORQ1.TORQF.TORQE.
3 PCTWOT.HIDNAM.ABR. RPMW.DRPMW.RPMP.RPM2.RPME.DRPME.
4   NGOCAL.MAPOK
IF(SHFTNG)
F
T
WRITE(JCT.1950) HSMODE.HTMODE.CUMTLS.CSTIM.TORQA
1.TORQAO.ATORQF.ARPME.DRPM1E.RPMC.DRPMC.DTC.LCLTCH
5
2
2
1
1
8A
900
IDBUGO=IDEBUG
RETURN
C***
C   FORMAT STATEMENTS
1700
FORMAT(/
1' CUMT.CUMD.V.ACCEL. FWHEEL.FROLL.FAERO.FACCEL.FGRADE'./.
2' DT.NGEAR.ISEGA.NRTEA.TORQW.TRR.TORQP.TORQ2.TORQ1.TORQF.TORQE'
3' PCTWOT.HIDNAM.ABR. RPMW.DRPMW.RPMP.RPM2.RPME.DRPME'./.
4'   NGOCAL.MAPOK'./)
1800
FORMAT (/' @'F9.2,F8.3.1X,F7.3,F10.3' (F)'5(1XG10.4)/
1.' DT'F6.3'.GEAR'I3'.DRS'I4'.RTE'I4' (T)'7(1XG10.4)/
2.' PWOT'F8.3'.ID:'A5'.8RK='F7.1' (R)'6(1XG10.4)./
3.' GOBACK'.I5.5X.'MAPOK:'.I2)
1850
FORMAT(/' @'F9.2.F8.3.2(1XG7.3)' (F)'4(1XG9.4)/

CONT. ON PG 7



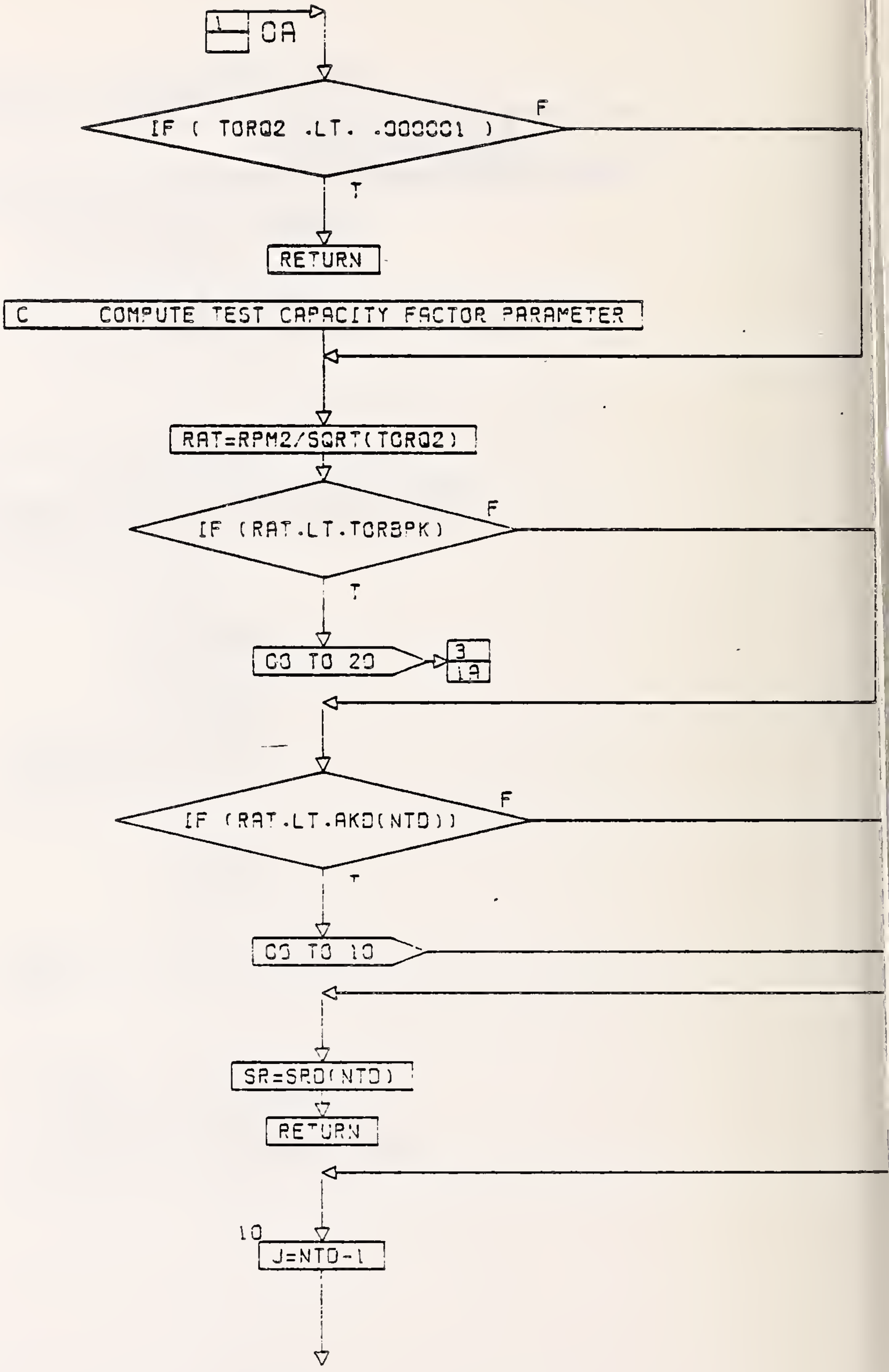
```
1,37XG10.4/
1,'   DT'F6.3'.GEAR'I3'.DRS'I4'.RTE'I4'  (T)'4(1XG9.4)
1,/35X3(1XG10.4)/
2,'   PWOT'F8.3'.ID:'AS'.BRK='F7.1' (R)'4(1XG9.4)
3,/35X2(1XG10.4)/
3.'   GOBACK',I5,5X,'MAPOK:',I2)

1900
FORMAT(2XA2'SHIFT GR 'A2'LOCKED'F9.2,1XF4.2' (S)'7(1XG10.4))

1950
FORMAT(2XA2'SHIFT GR 'A2'LOCKED'F9.2,1XF4.2' (S)'4(1XG10.4)/

1,3(1XG10.4))

END
```

3.3 Subroutine CONVTR

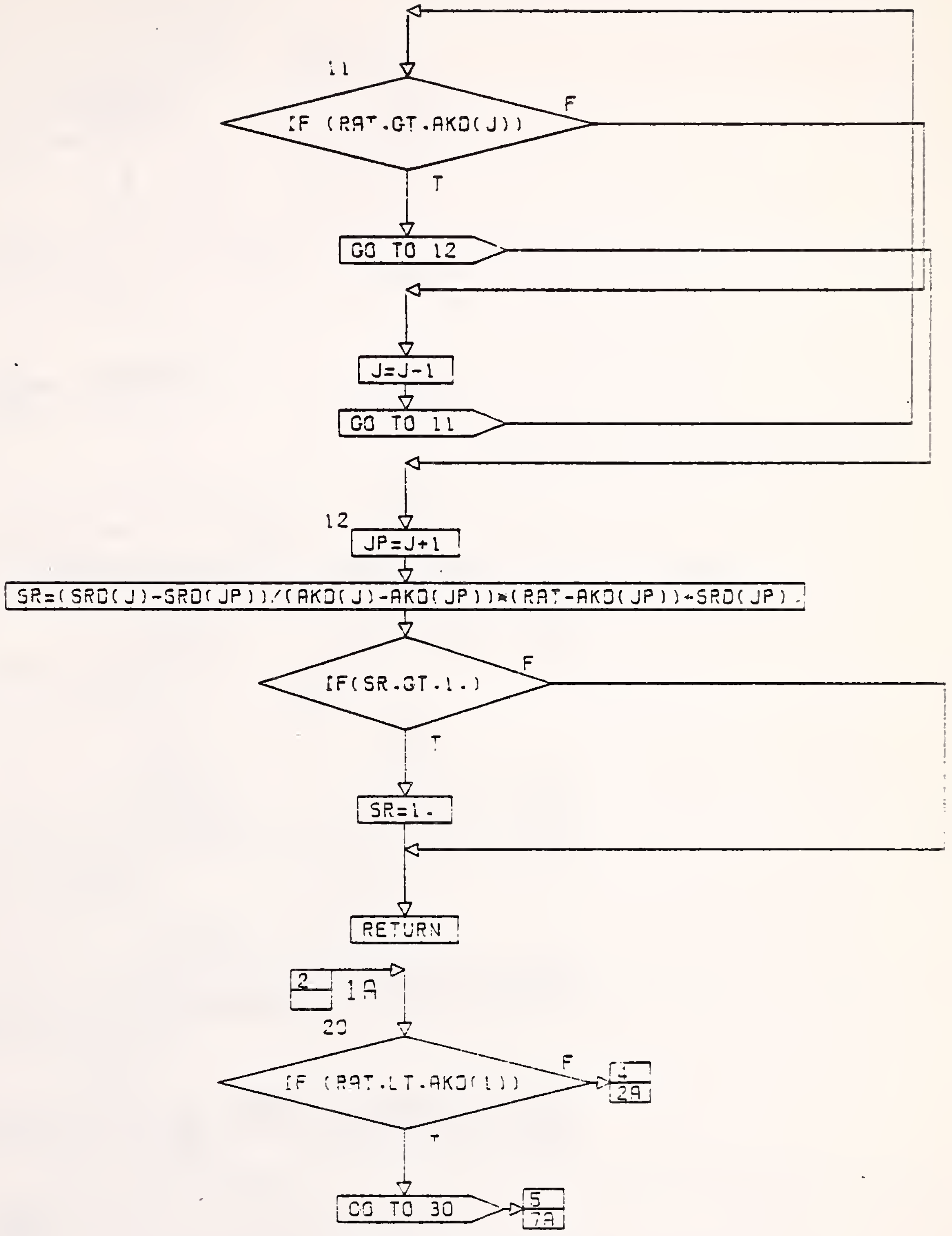


CONT. ON PG 2

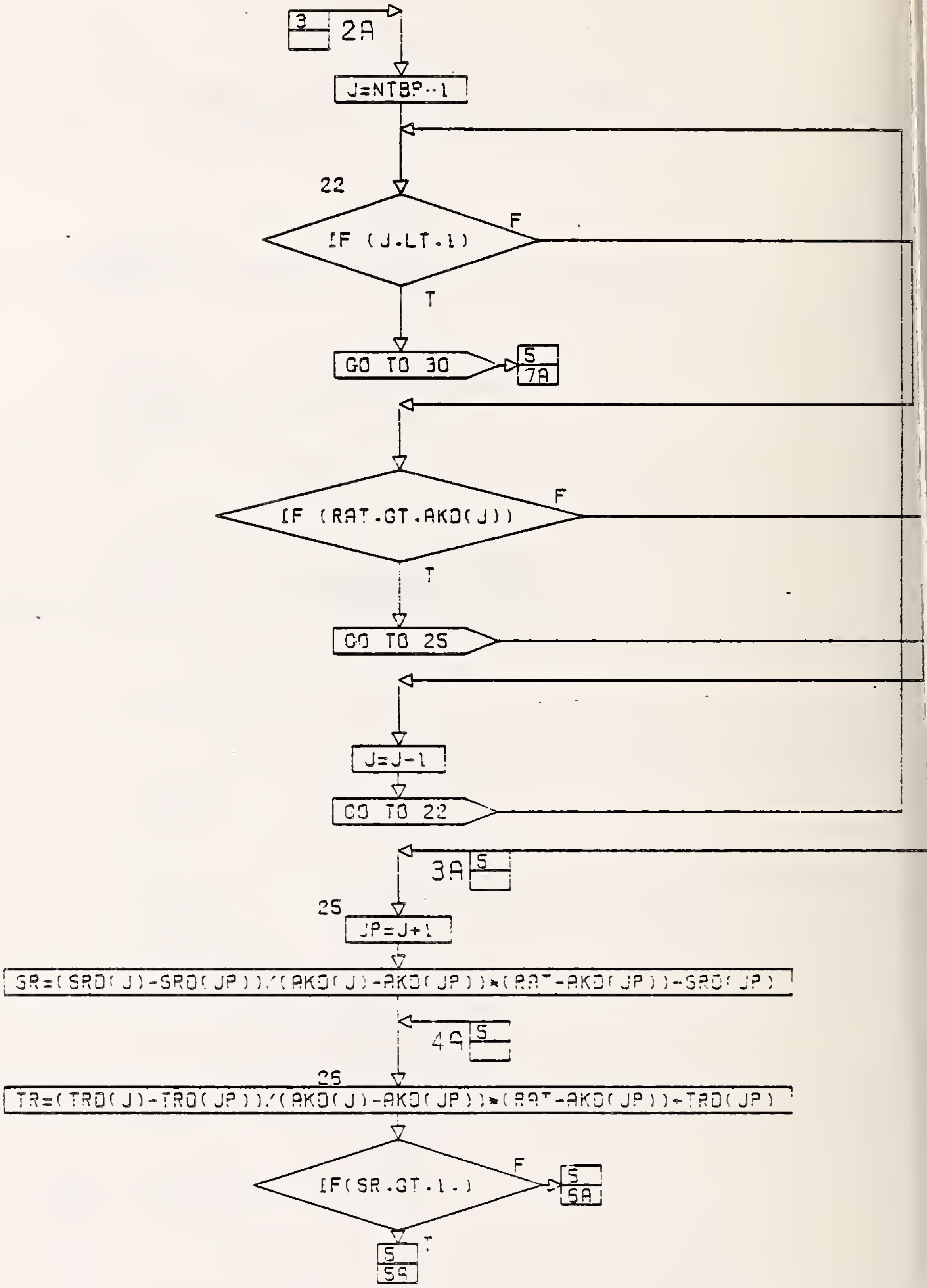
PG 1 OF 7


1
OA
IF (TORQ2 .LT. .000001)
F
T
RETURN
C COMPUTE TEST CAPACITY FACTOR PARAMETER
RAT=RPM2/SQRT(TORQ2)
IF (RAT.LT.TORBPK)
F
T
GO TO 20
3
1A
IF (RAT.LT.AKD(NTD))
F
T
GO TO 10
SR=SRD(NTD)
RETURN
10
J=NTD-1

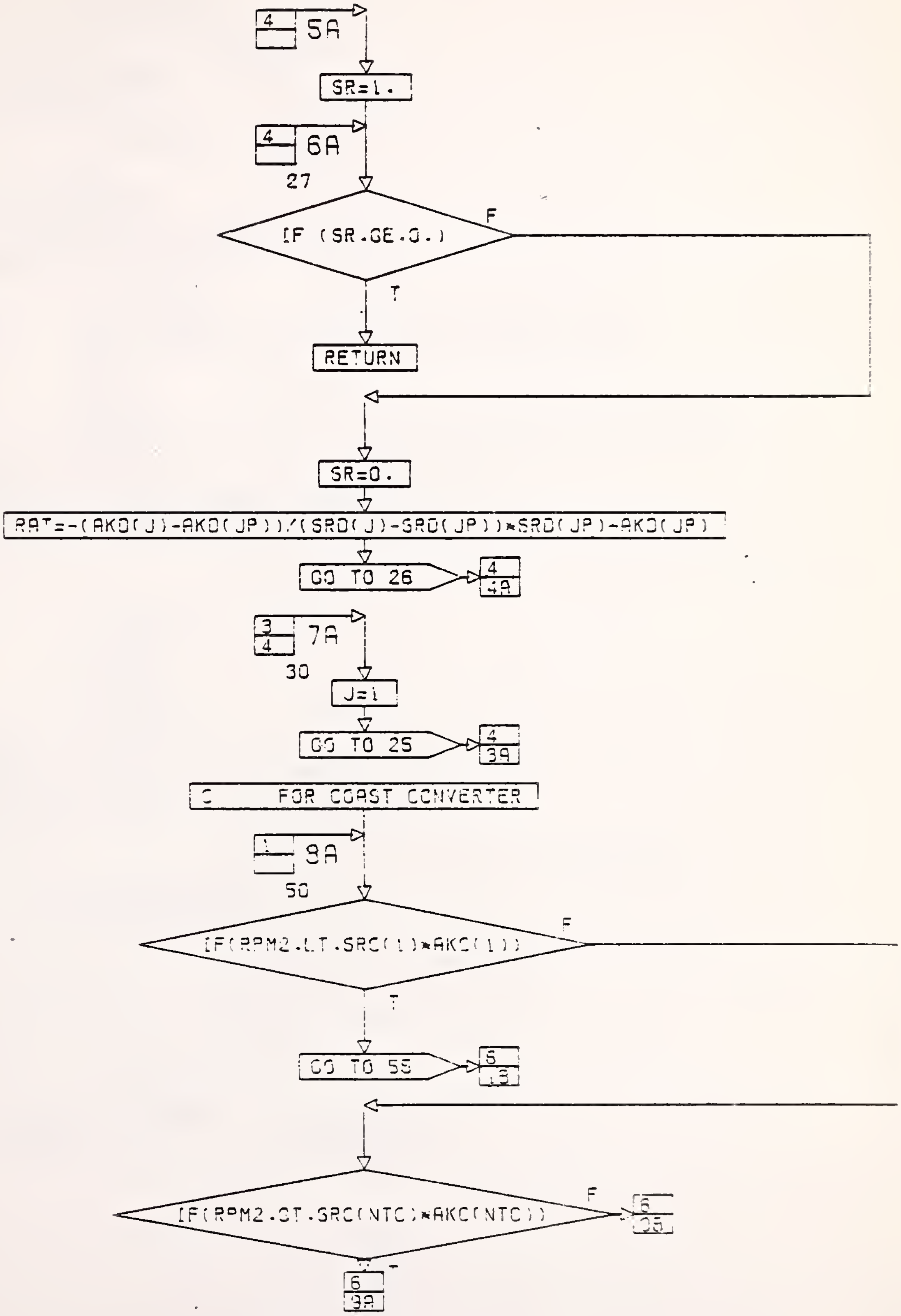
CONT. ON PG 3


11
IF (RAT.GT.AKD(J))
F
T
GO TO 12
J=J-1
GO TO 11
12
JP=J+1
SR=(SRD(J)-SRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+SRD(JP)
IF(SR.GT.1.)
F
T
SR=1.
RETURN
2
1A
20
IF (RAT.LT.AKD(1))
F
T
4
2A
GO TO 30
5
7A

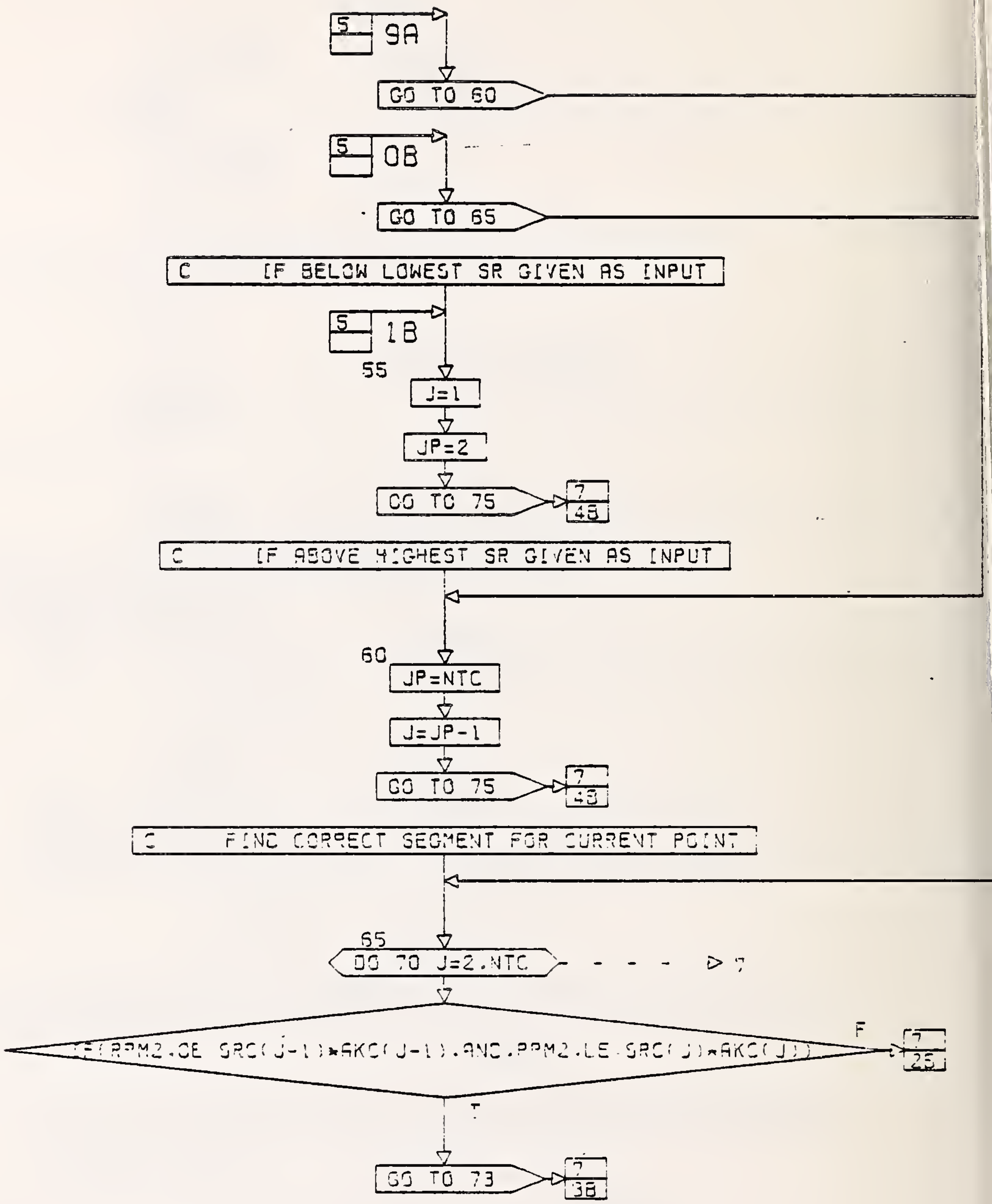
CONT. ON PG 4


3
2A
J=NTBP-1
22
IF (J.LT.1)
F
T
GO TO 30
5
7A
IF (RAT.GT.AKD(J))
F
T
GO TO 25
J=J-1
GO TO 22
3A
5
25
JP=J+1
SR=(SRD(J)-SRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+SRD(JP)
4A
5
26
TR=(TRD(J)-TRD(JP))/(AKD(J)-AKD(JP))*(RAT-AKD(JP))+TRD(JP)
IF(SR.GT.1.)
F
5
5A
T
5
5A

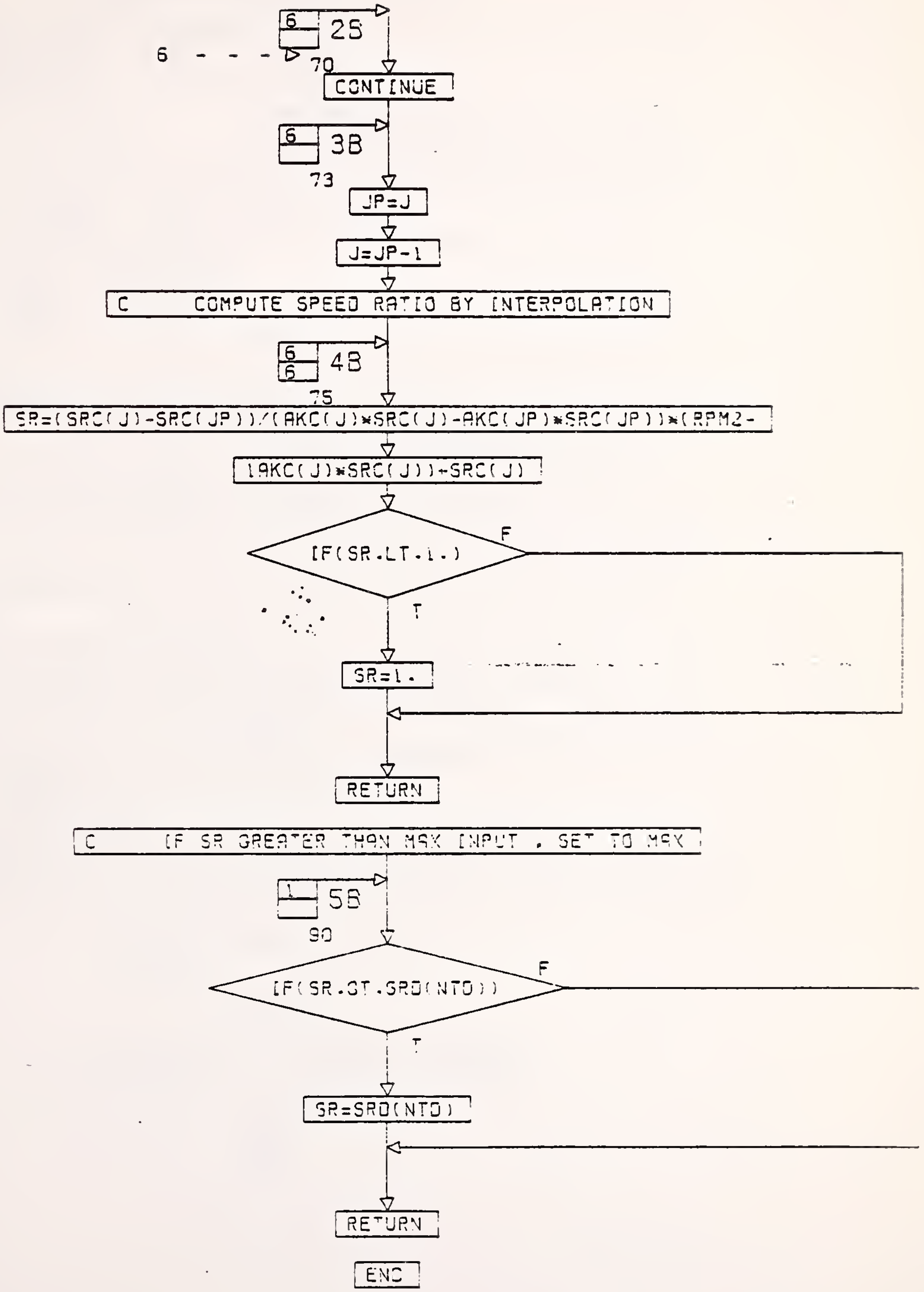
CONT. ON PG 5


4
5A
SR=1.
4
6A
27
IF (SR.GE.0.)
F
T
RETURN
SR=0.
RAT=-(AKO(J)-AKO(JP))/(SRO(J)-SRO(JP))*SRO(JP)-AKO(JP)
GO TO 26
4
4A
3
4
7A
30
J=1
GO TO 25
4
3A
C FOR COAST CONVERTER
1
8A
50
IF(RPM2.LT.SRC(1)*AKC(1))
F
T
GO TO 55
5
1B
IF(RPM2.GT.SRC(NTC)*AKC(NTC))
F
6
3B
T
6
9A

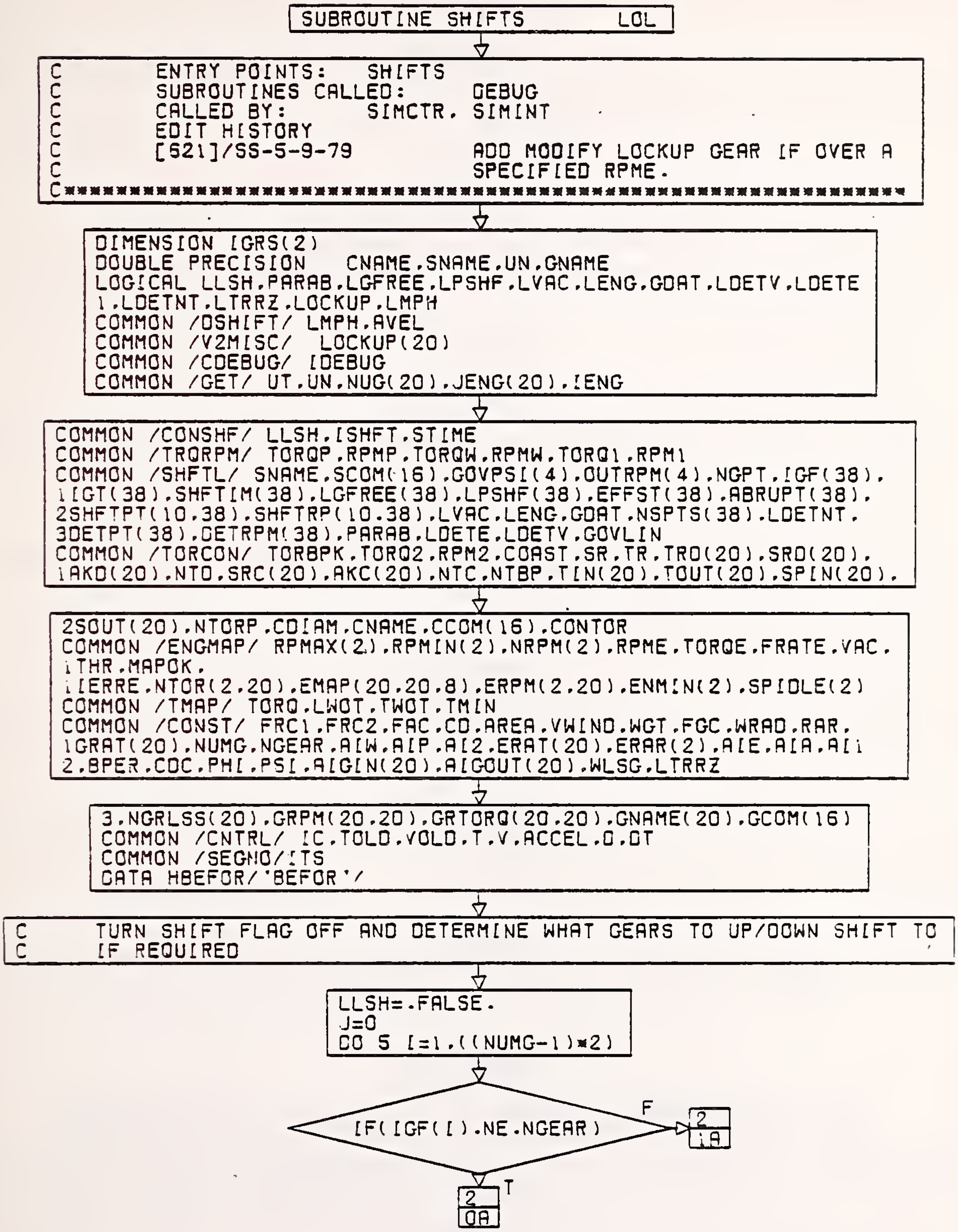
CONT. ON PG 5


5 9A
GO TO 60
5 0B
GO TO 65
C IF BELOW LOWEST SR GIVEN AS INPUT
5 1B
55
J=1
JP=2
GO TO 75
7 48
C IF ABOVE HIGHEST SR GIVEN AS INPUT
60
JP=NTC
J=JP-1
GO TO 75
7 48
C FIND CORRECT SEGMENT FOR CURRENT POINT
65
DO 70 J=2,NTC
7
IF(RPM2.GE.SRC(J-1)*AKC(J-1).AND.RPM2.LE.SRC(J)*AKC(J))
F
7 25
T
GO TO 73
7 38

CONT. ON PG 7


6
2B
6
70
CONTINUE
6
3B
73
JP=J
J=JP-1
C COMPUTE SPEED RATIO BY INTERPOLATION
6
6
4B
75
SR=(SRC(J)-SRC(JP))/(AKC(J)*SRC(J)-AKC(JP)*SRC(JP))*(RPM2-
1AKC(J)*SRC(J))+SRC(J)
IF(SR.LT.1.)
F
T
SR=1.
RETURN
C IF SR GREATER THAN MAX INPUT , SET TO MAX
1
5B
90
IF(SR.GT.SRD(NTO))
F
T
SR=SRD(NTO)
RETURN
END

3.4 Subroutine SHIFTS



```
SUBROUTINE SHIFTS        LOL
```

```
C        ENTRY POINTS:    SHIFTS
C        SUBROUTINES CALLED:       DEBUG
C        CALLED BY:       SIMCTR. SIMINT
C        EDIT HISTORY
C        [521]/SS-5-9-79           ADD MODIFY LOCKUP GEAR IF OVER A
C                                  SPECIFIED RPME.
C*****************************************************************************
```

```
DIMENSION IGRS(2)
DOUBLE PRECISION   CNAME.SNAME.UN.GNAME
LOGICAL LLSH.PARAB.LGFREE.LPSHF.LVAC.LENG.GDAT.LDETV.LDETE
1.LDETNT.LTRRZ.LOCKUP.LMPH
COMMON /DSHIFT/ LMPH.AVEL
COMMON /V2MISC/  LOCKUP(20)
COMMON /CDEBUG/ IDEBUG
COMMON /GET/ UT.UN.NUG(20).JENG(20).IENG
```

```
COMMON /CONSHF/ LLSH.ISHFT.STIME
COMMON /TRQRPM/ TORQP.RPMP.TORQW.RPMW.TORQ1.RPM1
COMMON /SHFTL/ SNAME.SCOM(16).GOVPSI(4).OUTRPM(4).NGPT.IGF(38).
1IGT(38).SHFTIM(38).LGFREE(38).LPSHF(38).EFFST(38).ABRUPT(38).
2SHFTPT(10.38).SHFTRP(10.38).LVAC.LENG.GDAT.NSPTS(38).LDETNT.
3DETPT(38).DETRPM(38).PARAB.LDETE.LDETV.GOVLIN
COMMON /TORCON/ TORBPK.TORQ2.RPM2.COAST.SR.TR.TRO(20).SRD(20).
1AKD(20).NTO.SRC(20).AKC(20).NTC.NTBP.TIN(20).TOUT(20).SPIN(20).
```

```
2SOUT(20).NTORP.CDIAM.CNAME.CCOM(16).CONTOR
COMMON /ENGMAP/ RPMAX(2).RPMIN(2).NRPM(2).RPME.TORQE.FRATE.VAC.
1THR.MAPOK.
1IERRE.NTOR(2.20).EMAP(20.20.8).ERPM(2.20).ENMIN(2).SPIDLE(2)
COMMON /TMAP/ TORQ.LWOT.TWOT.TMIN
COMMON /CONST/ FRC1.FRC2.FAC.CD.AREA.VWIND.WGT.FGC.WRAD.RAR.
1GRAT(20).NUMG.NGEAR.AIW.AIP.AI2.ERAT(20).ERAR(2).AIE.AIA.AI1
2.BPER.CDC.PHI.PSI.AIGIN(20).AIGOUT(20).WLSG.LTRRZ
```

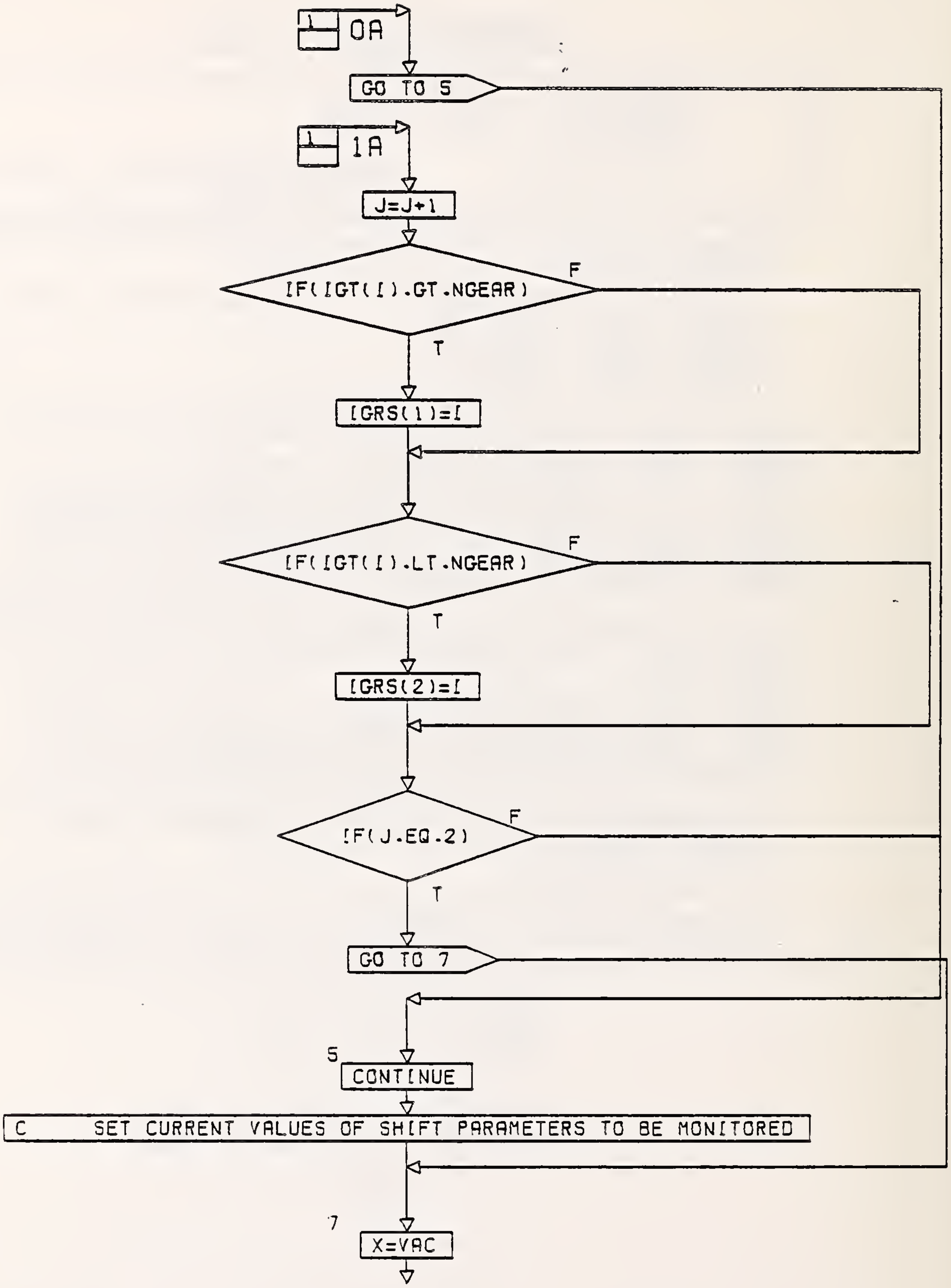
```
3.NGRLSS(20).GRPM(20.20).GRTORQ(20.20).GNAME(20).GCOM(16)
COMMON /CNTRL/ IC.TOLD.VOLD.T.V.ACCEL.D.DT
COMMON /SEGNO/ITS
DATA HBEFOR/'BEFOR'/
```

```
C      TURN SHIFT FLAG OFF AND DETERMINE WHAT GEARS TO UP/DOWN SHIFT TO
C      IF REQUIRED
```

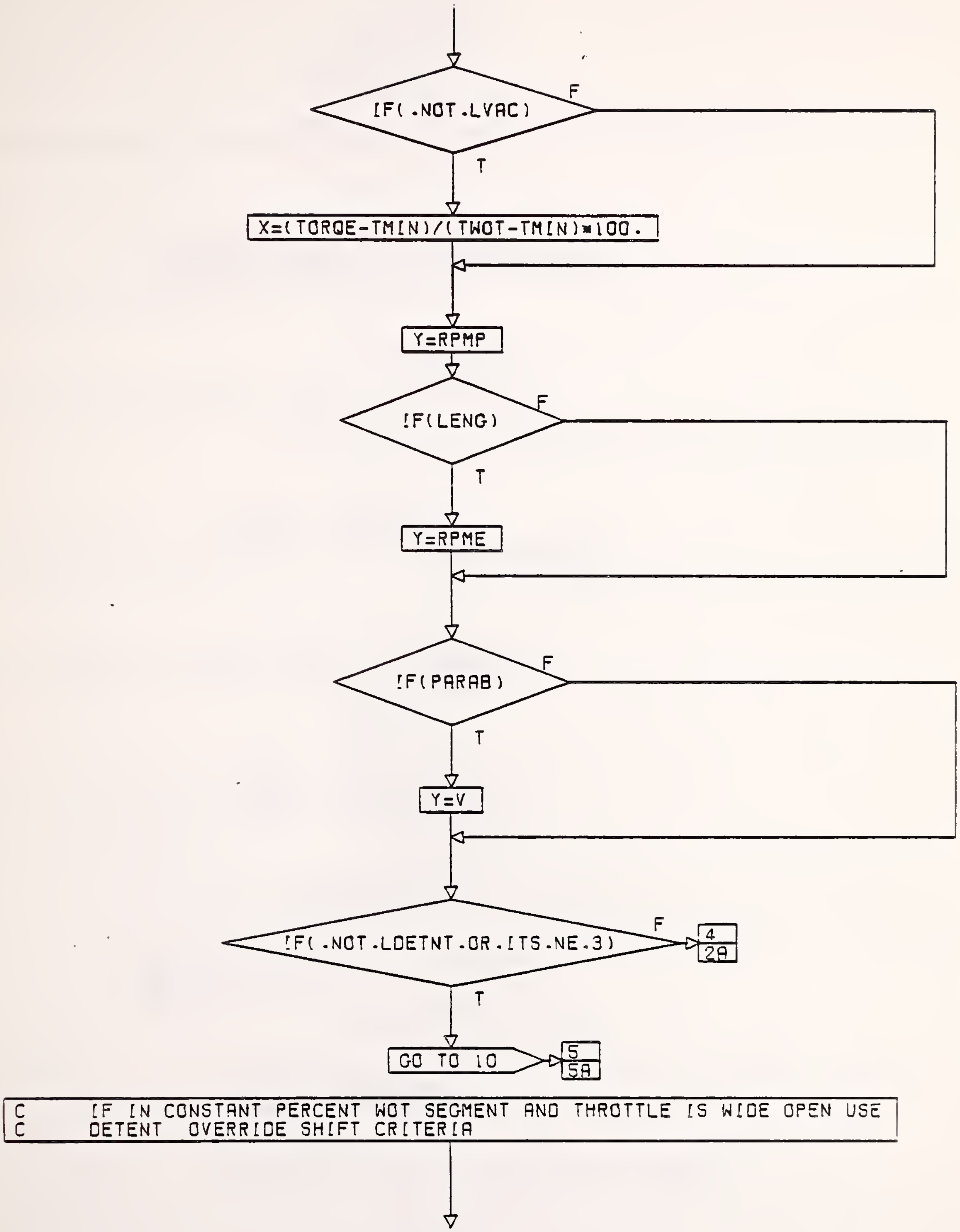
```
LLSH=.FALSE.
J=0
DO 5 I=1.((NUMG-1)*2)
```

```
IF(IGF(I).NE.NGEAR)
```

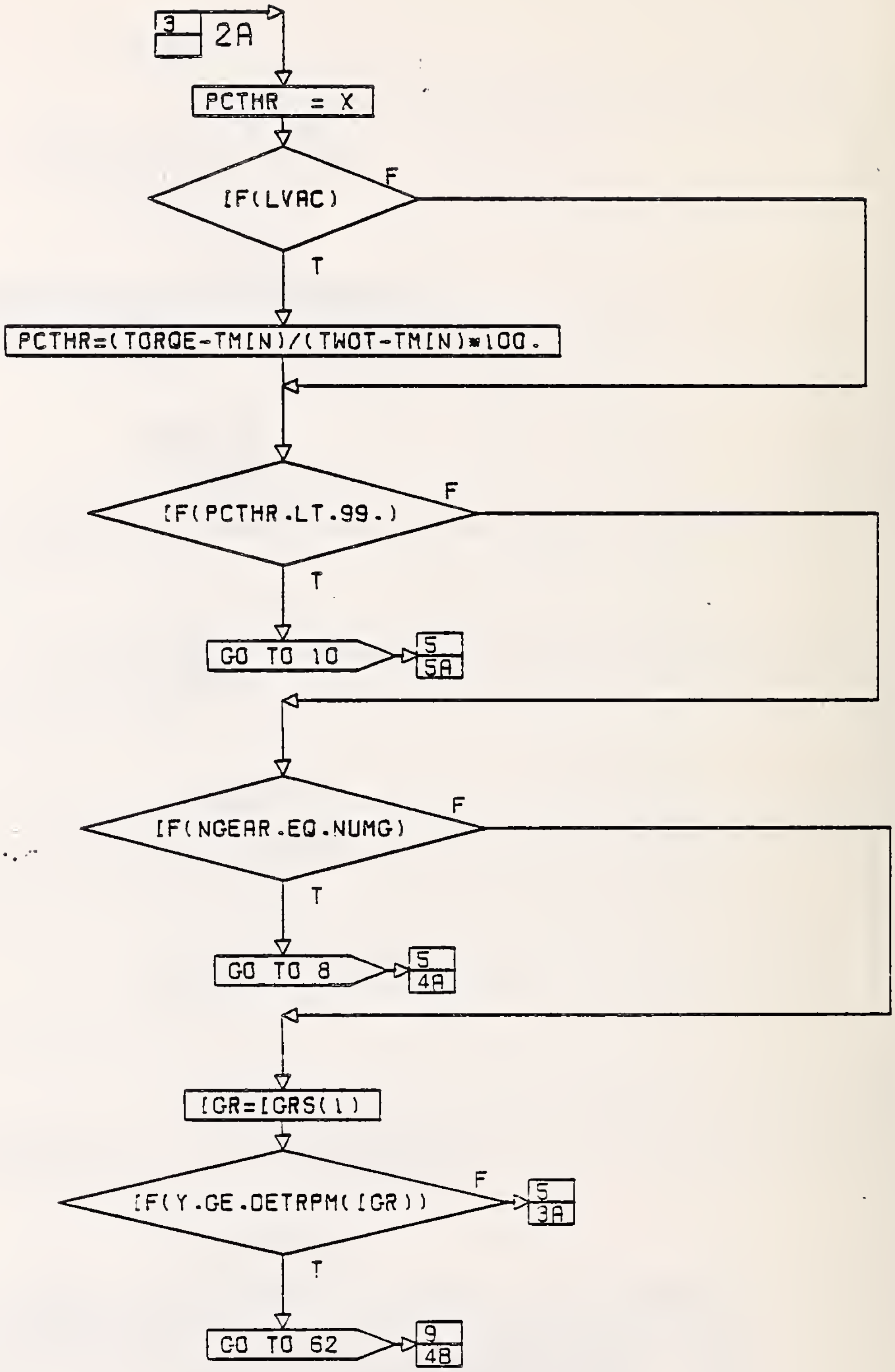
CONT. ON PG   2


1
0A
GO TO 5
1
1A
J=J+1
IF(IGT(I).GT.NGEAR)
F
T
IGRS(1)=I
IF(IGT(I).LT.NGEAR)
F
T
IGRS(2)=I
IF(J.EQ.2)
F
T
GO TO 7
5
CONTINUE
C SET CURRENT VALUES OF SHIFT PARAMETERS TO BE MONITORED
7
X=VAC

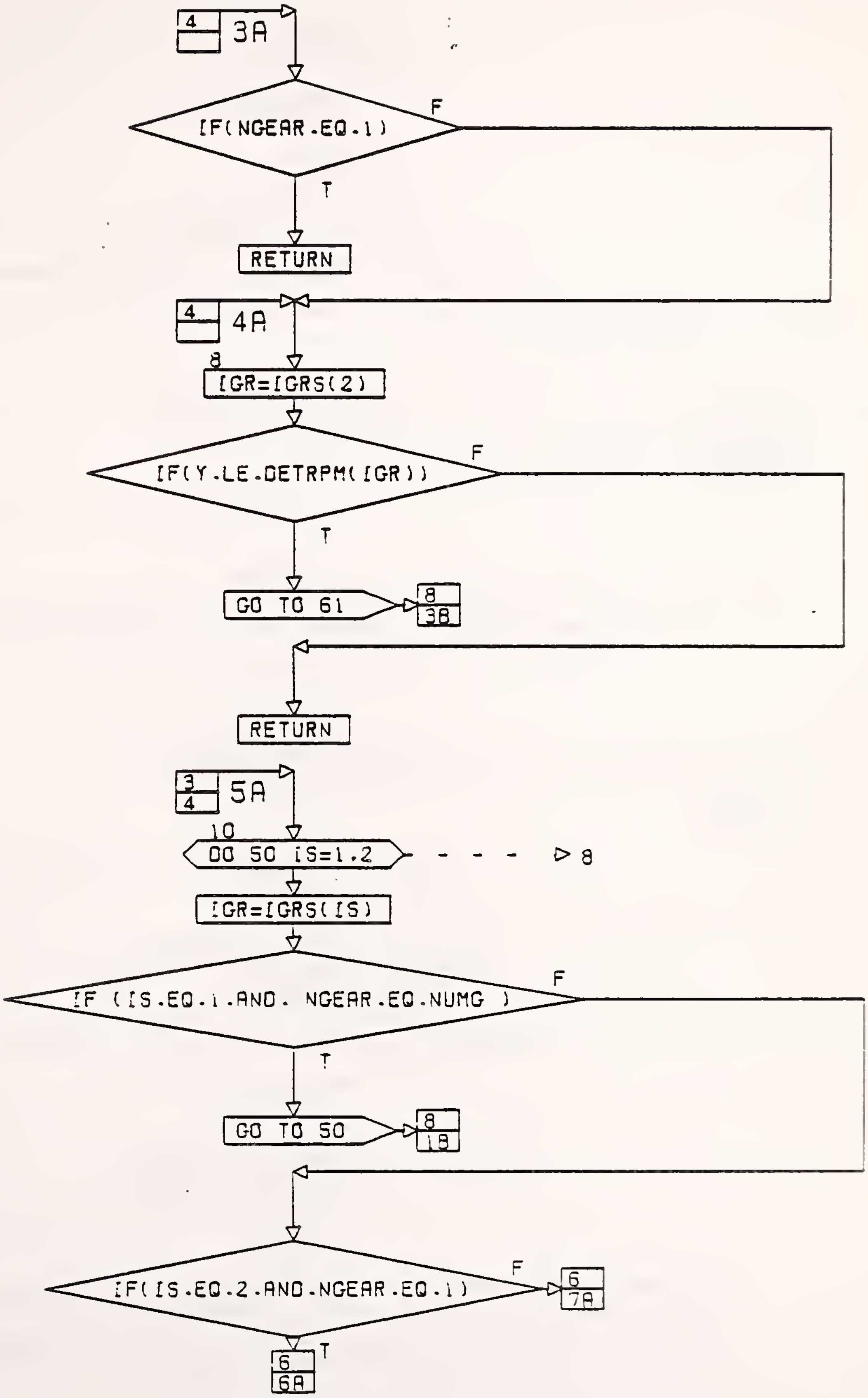
CONT. ON PG 3


IF(.NOT.LVAC)
F
T
X=(TORQE-TMIN)/(TWOT-TMIN)*100.
Y=RPMP
IF(LENG)
F
T
Y=RPME
IF(PARAB)
F
T
Y=V
IF(.NOT.LDETNT.OR.ITS.NE.3)
F
4
2A
T
GO TO 10
5
5A
C IF IN CONSTANT PERCENT WOT SEGMENT AND THROTTLE IS WIDE OPEN USE
C DETENT OVERRIDE SHIFT CRITERIA

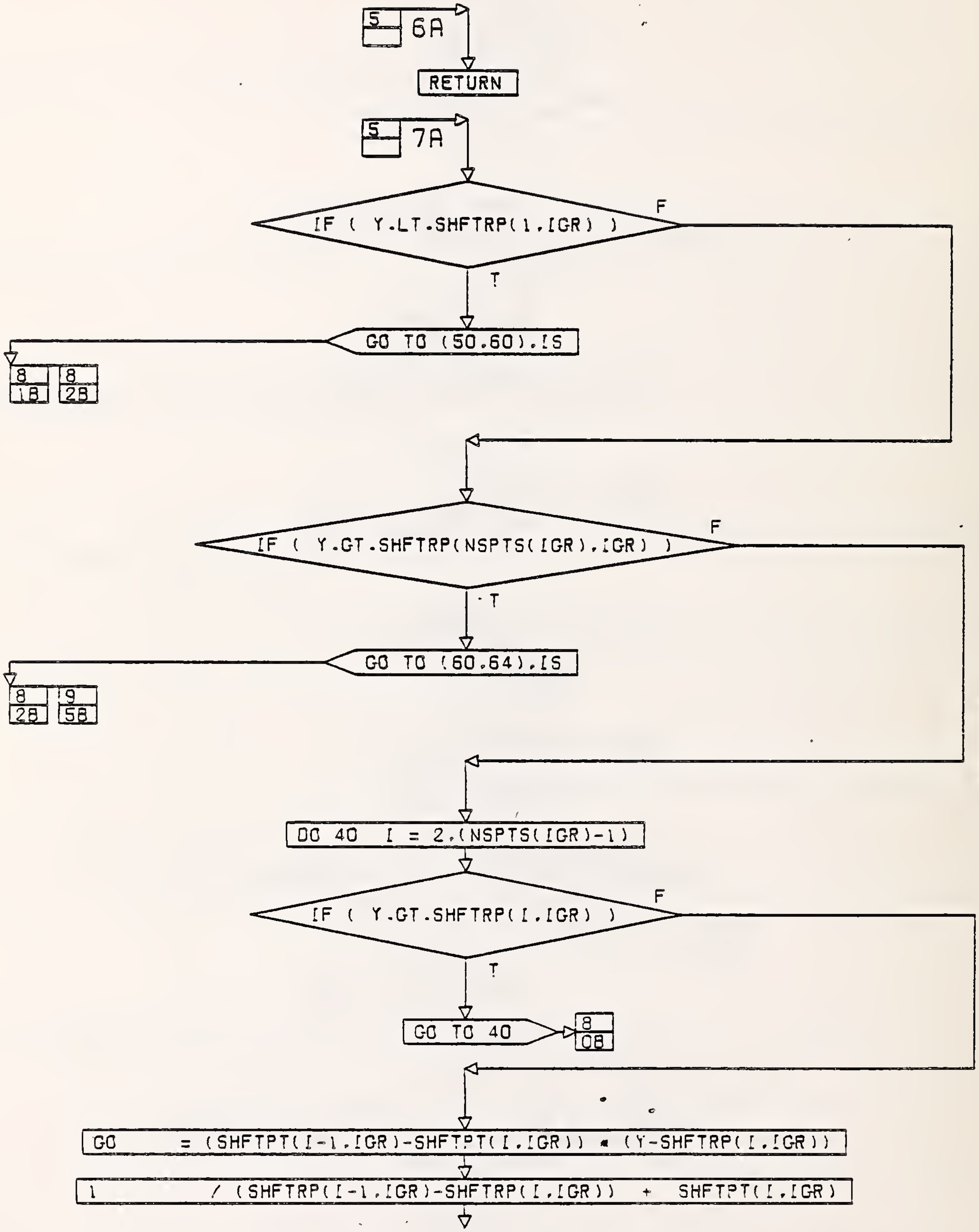
CONT. ON PG 4


3
2A
PCTHR = X
IF(LVAC)
F
T
PCTHR=(TORQE-TMIN)/(TWOT-TMIN)*100.
IF(PCTHR.LT.99.)
F
T
GO TO 10
5
5A
IF(NGEAR.EQ.NUMG)
F
T
GO TO 8
5
4A
IGR=IGRS(1)
IF(Y.GE.DETRPM(IGR))
F
5
3A
T
GO TO 62
9
4B

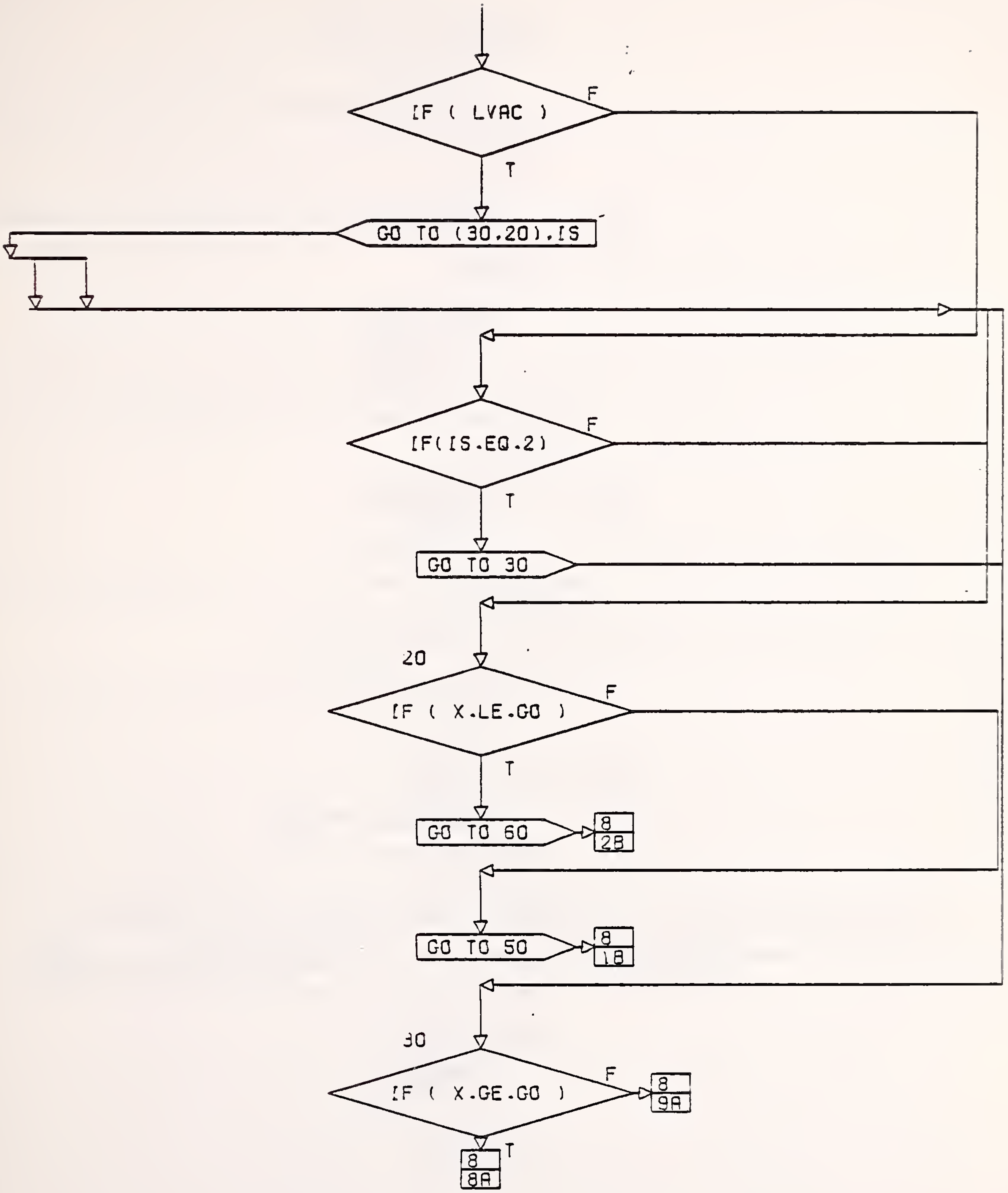
CONT. ON PG 5


4
3A
IF(NGEAR.EQ.1)
F
T
RETURN
4
4A
8
IGR=IGRS(2)
IF(Y.LE.DETRPM(IGR))
F
T
GO TO 61
8
3B
RETURN
3
4
5A
10
DO 50 IS=1,2
8
IGR=IGRS(IS)
IF (IS.EQ.1.AND. NGEAR.EQ.NUMG)
F
T
GO TO 50
8
1B
IF(IS.EQ.2.AND.NGEAR.EQ.1)
F
6
7A
T
6
6A

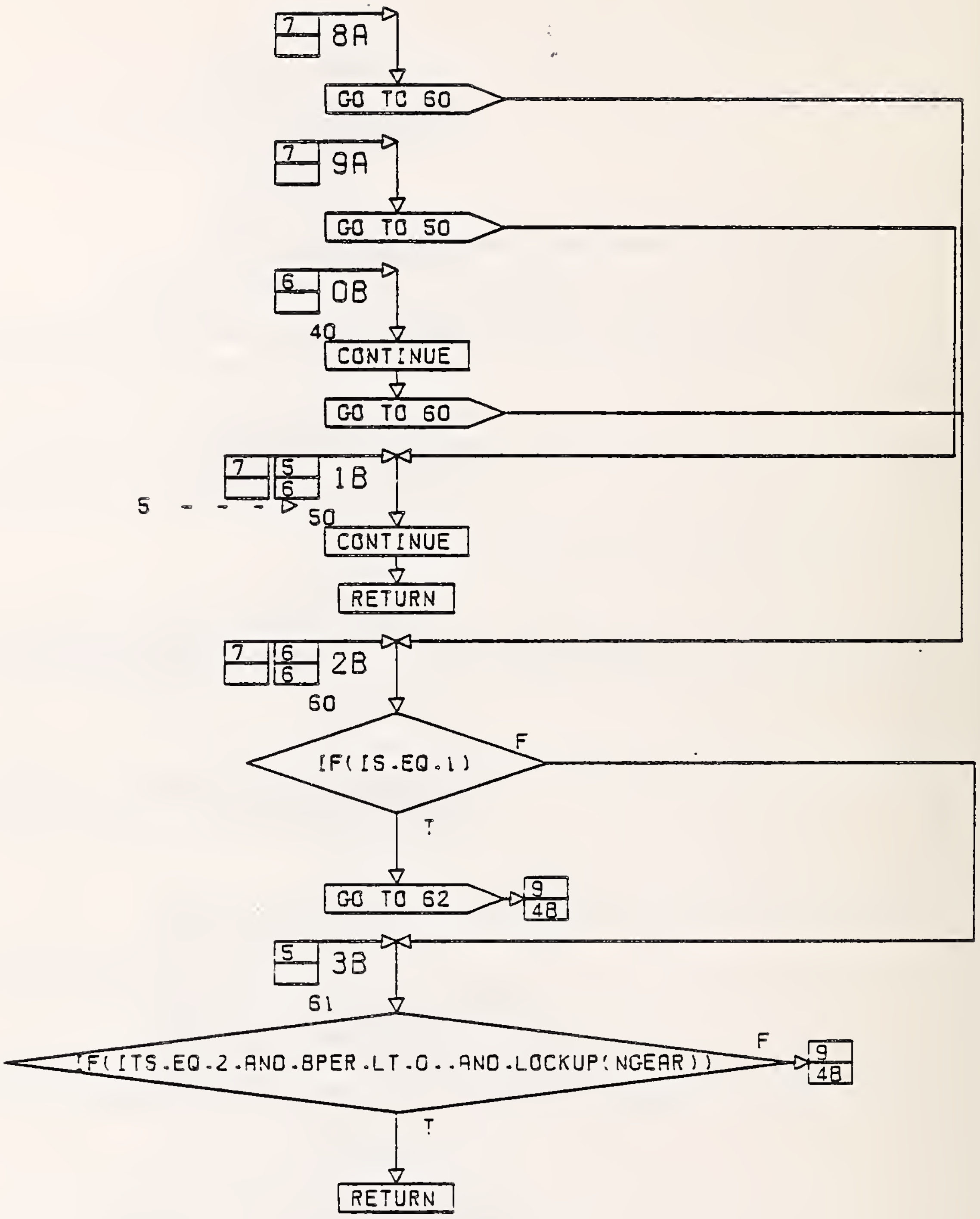
CONT. ON PG 6


5
6A
RETURN
5
7A
IF (Y.LT.SHFTRP(1.IGR))
F
T
GO TO (50.60).IS
8
1B
8
2B
IF (Y.GT.SHFTRP(NSPTS(IGR).IGR))
F
T
GO TO (60.64).IS
8
2B
9
5B
DO 40 I = 2.(NSPTS(IGR)-1)
IF (Y.GT.SHFTRP(I.IGR))
F
T
GO TO 40
8
0B
GO = (SHFTPT(I-1.IGR)-SHFTPT(I.IGR)) * (Y-SHFTRP(I.IGR))
1 / (SHFTRP(I-1.IGR)-SHFTRP(I.IGR)) + SHFTPT(I.IGR)

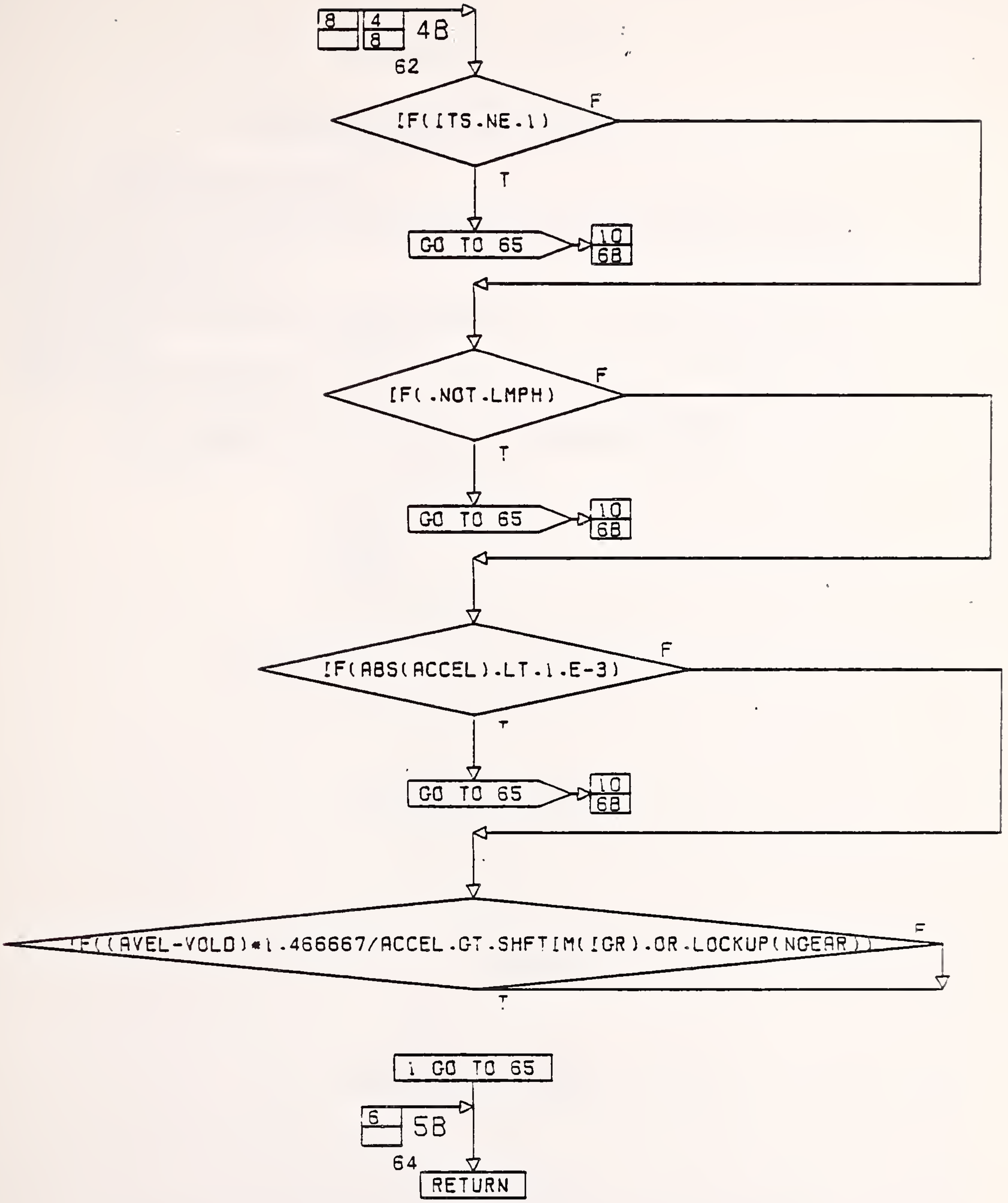
CONT. ON PG 7


IF (LVAC)
F
T
GO TO (30,20),IS
IF(IS.EQ.2)
F
T
GO TO 30
20
IF (X.LE.GO)
F
T
GO TO 60
8
2B
GO TO 50
8
1B
30
IF (X.GE.GO)
F
8
9A
T
8
8A

CONT. ON PG 8

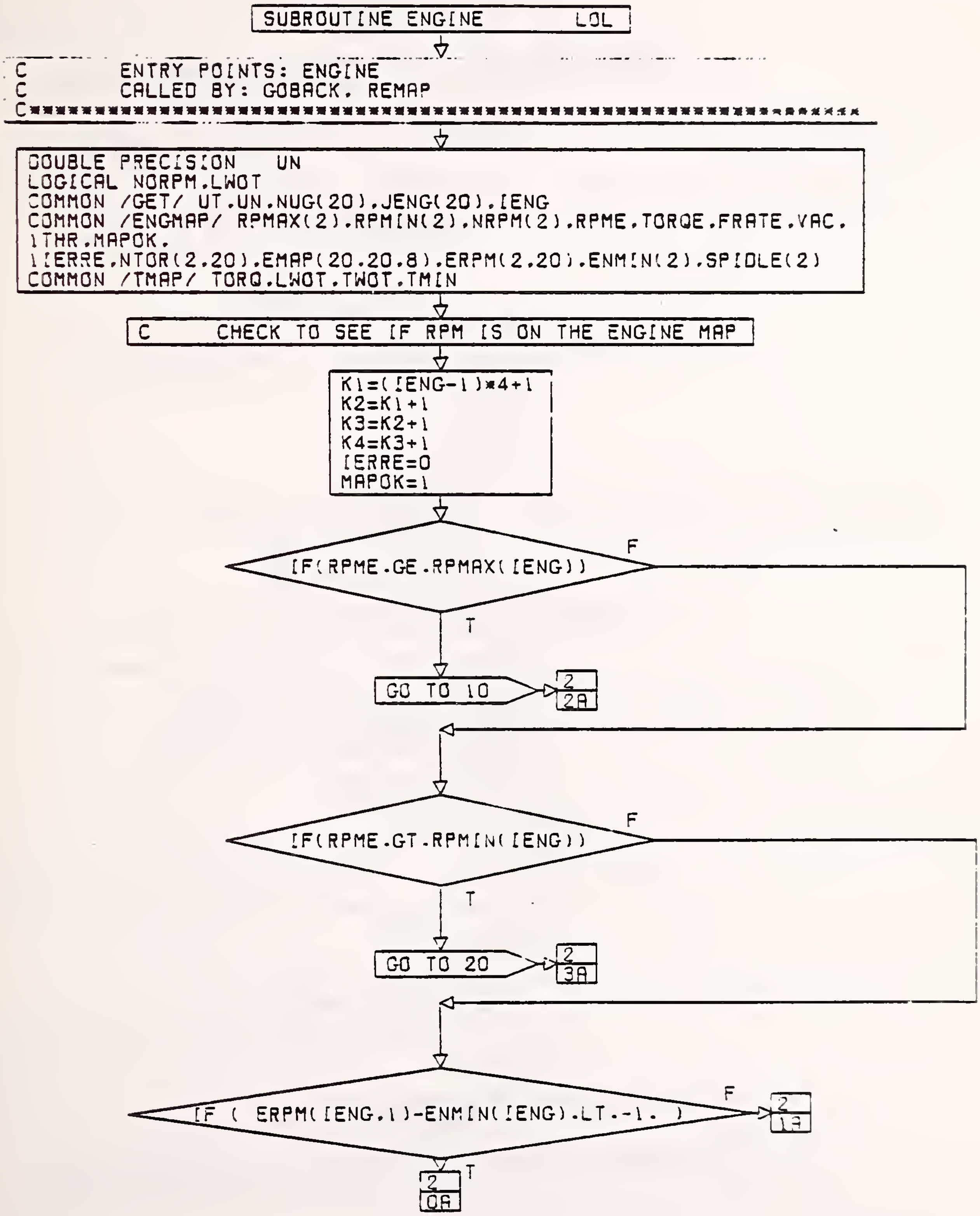

7
8A
GO TO 60
7
9A
GO TO 50
6
0B
40
CONTINUE
GO TO 60
7
5
6
1B
5
50
CONTINUE
RETURN
7
6
6
2B
60
IF(IS.EQ.1)
F
T
GO TO 62
9
48
5
3B
61
IF(ITS.EQ.2.AND.BPER.LT.0..AND.LOCKUP(NGEAR))
F
9
48
T
RETURN

CONT. ON PG 9


8
4
8
4B
62
IF(ITS.NE.1)
F
T
GO TO 65
10
6B
IF(.NOT.LMPH)
F
T
GO TO 65
10
6B
IF(ABS(ACCEL).LT.1.E-3)
F
T
GO TO 65
10
6B
IF((AVEL-VOLD)*1.466667/ACCEL.GT.SHFTIM(IGR).OR.LOCKUP(NGEAR))
F
T
1 GO TO 65
6
5B
64
RETURN

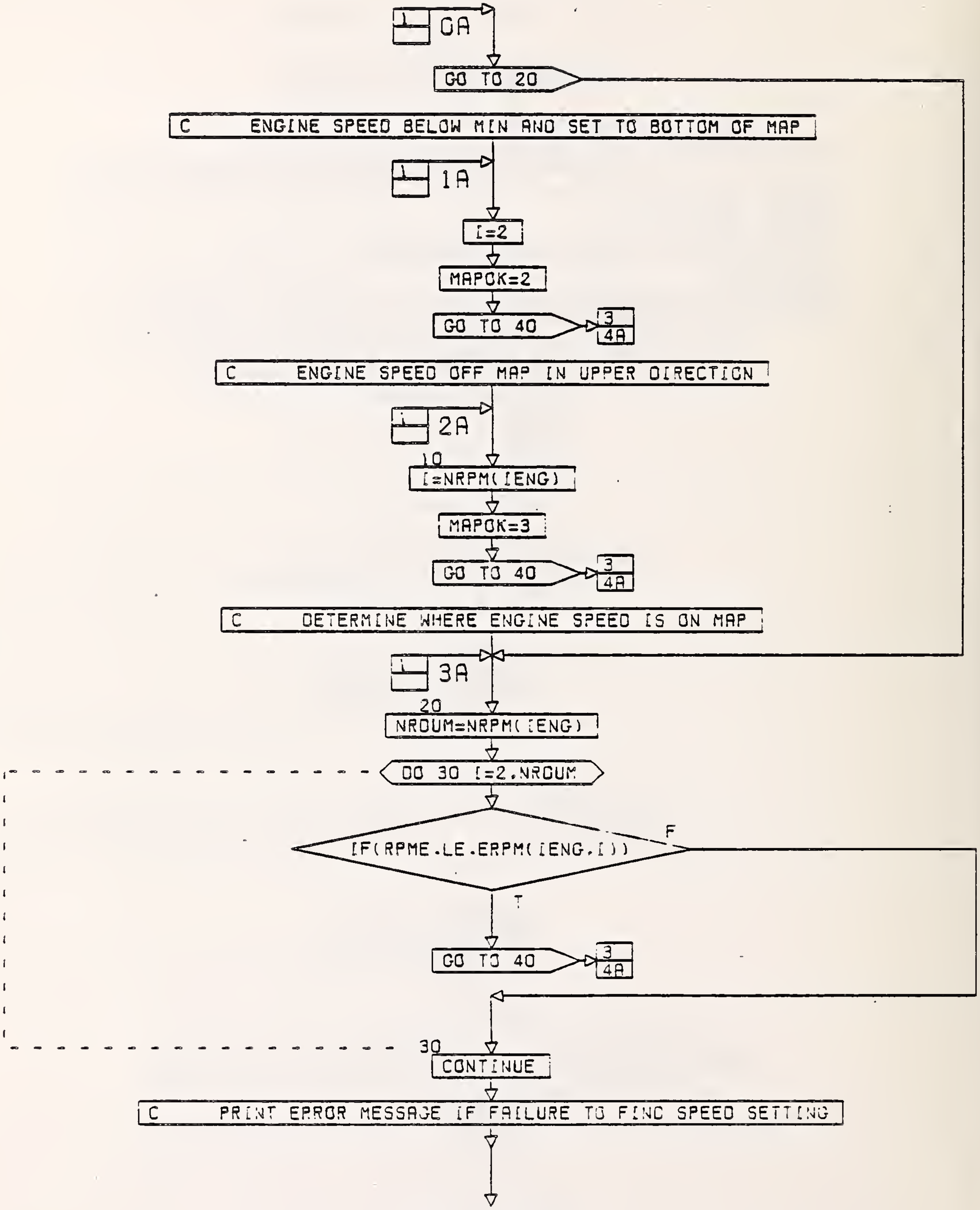
CONT. ON PG 10

3.5 Subroutine ENGINE

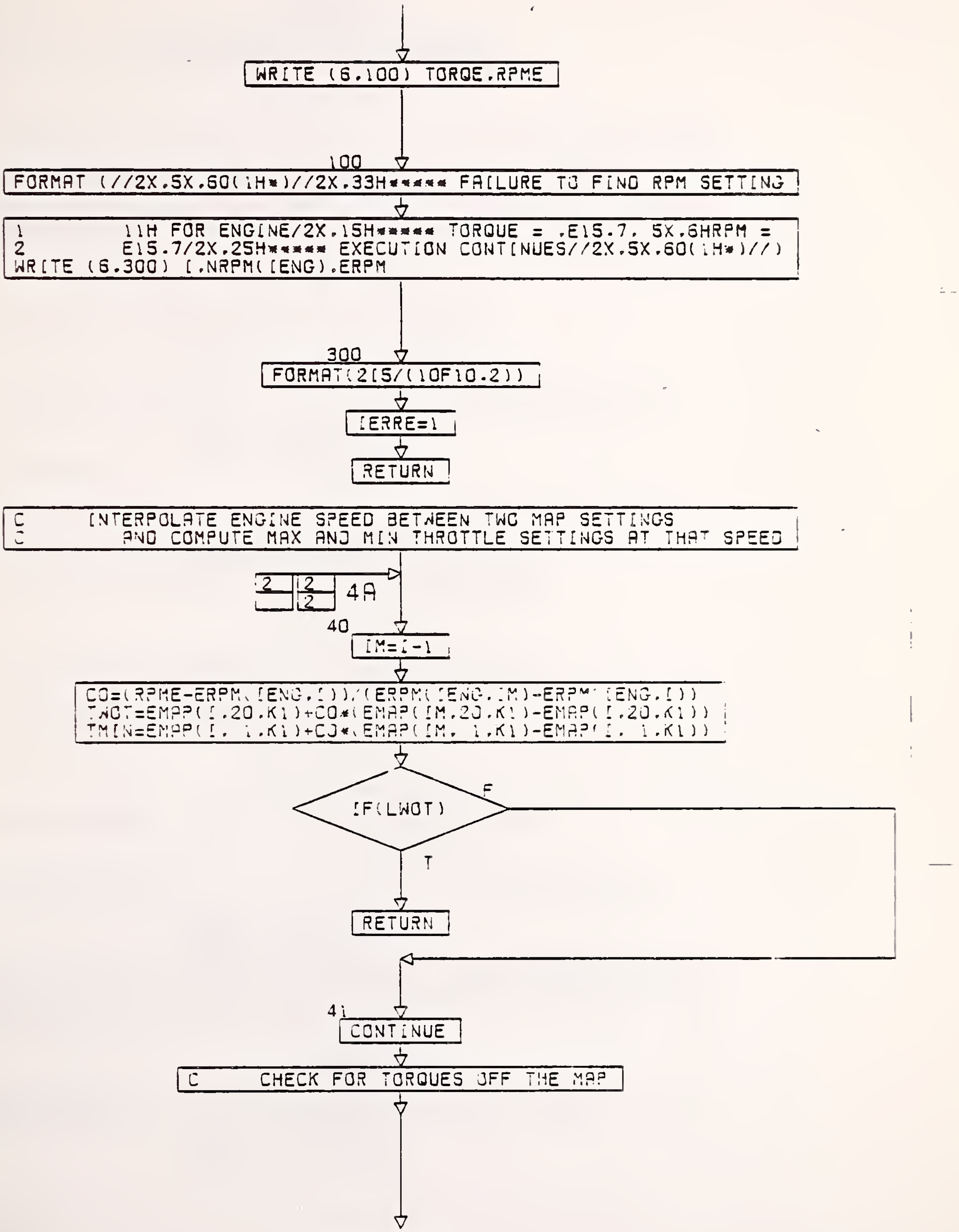


```
SUBROUTINE ENGINE          LOL
C       ENTRY POINTS: ENGINE
C       CALLED BY: GOBACK. REMAP
C*********************************************************************

DOUBLE PRECISION    UN
LOGICAL NORPM.LWOT
COMMON /GET/ UT.UN.NUG(20).JENG(20).IENG
COMMON /ENGMAP/ RPMAX(2).RPMIN(2).NRPM(2).RPME.TORQE.FRATE.VAC.
1THR.MAPOK.
1IERRE.NTOR(2.20).EMAP(20.20.8).ERPM(2.20).ENMIN(2).SPIDLE(2)
COMMON /TMAP/ TORQ.LWOT.TWOT.TMIN

C      CHECK TO SEE IF RPM IS ON THE ENGINE MAP

K1=(IENG-1)*4+1
K2=K1+1
K3=K2+1
K4=K3+1
IERRE=0
MAPOK=1

IF(RPME.GE.RPMAX(IENG))
GO TO 10

IF(RPME.GT.RPMIN(IENG))
GO TO 20

IF ( ERPM(IENG.1)-ENMIN(IENG).LT.-1. )
```

CONT. ON PG 2

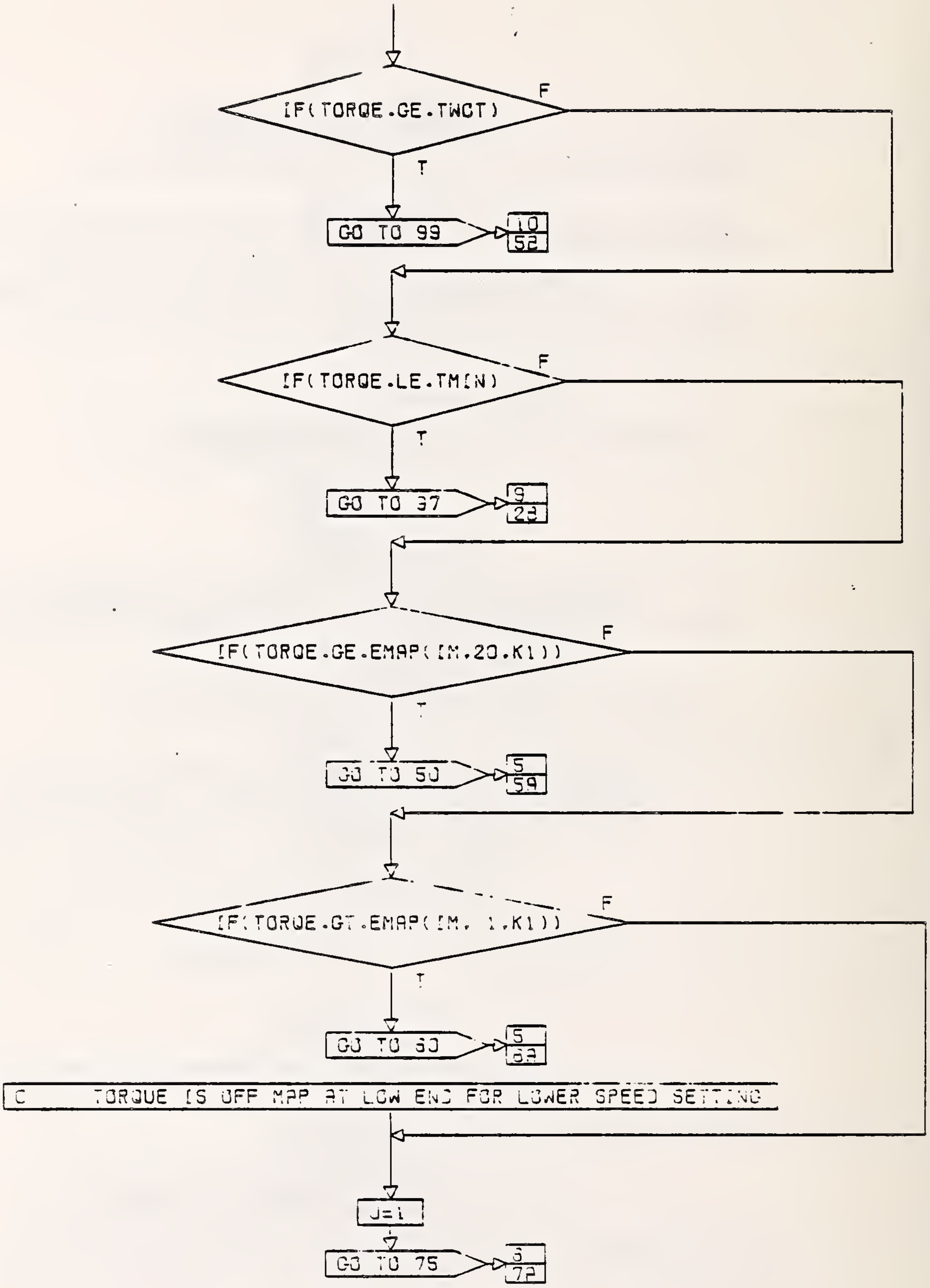

1
0A
GO TO 20
C ENGINE SPEED BELOW MIN AND SET TO BOTTOM OF MAP
1
1A
I=2
MAPOK=2
GO TO 40
3
4A
C ENGINE SPEED OFF MAP IN UPPER DIRECTION
1
2A
10
I=NRPM(IENG)
MAPOK=3
GO TO 40
3
4A
C DETERMINE WHERE ENGINE SPEED IS ON MAP
1
3A
20
NRDUM=NRPM(IENG)
DO 30 I=2,NRDUM
IF(RPME.LE.ERPM(IENG,I))
F
T
GO TO 40
3
4A
30
CONTINUE
C PRINT ERROR MESSAGE IF FAILURE TO FINC SPEED SETTING

CONT. ON PG 3

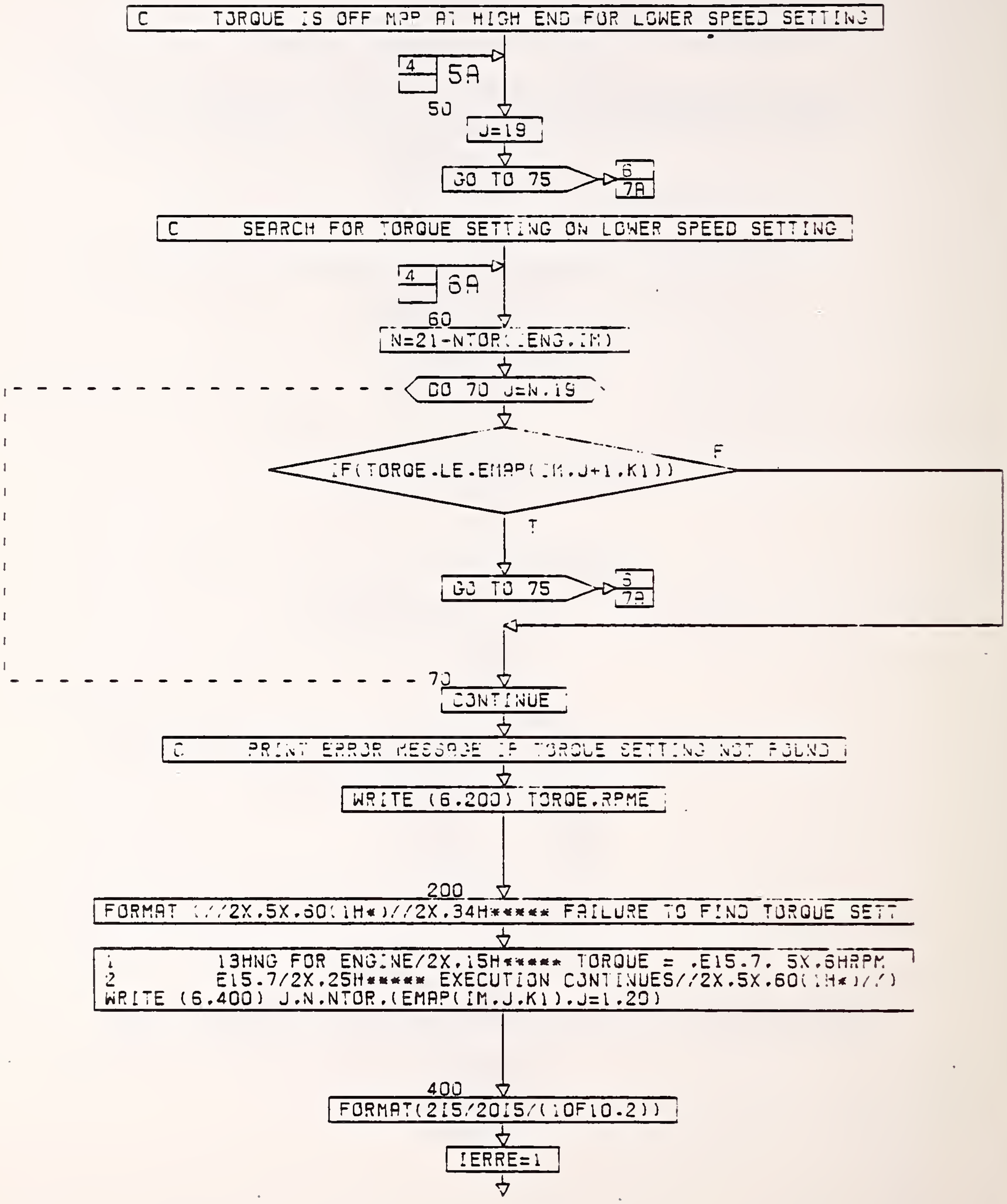


```
      WRITE (6.100) TORQE.RPME
100   FORMAT (//2X.5X.60(1H*)//2X.33H***** FAILURE TO FIND RPM SETTING
     1        11H FOR ENGINE/2X.15H***** TORQUE = .E15.7. 5X.6HRPM =
     2        E15.7/2X.25H***** EXECUTION CONTINUES//2X.5X.60(1H*)//)
      WRITE (6.300) I.NRPM(IENG).ERPM
300   FORMAT(2I5/(10F10.2))
      IERRE=1
      RETURN
C       INTERPOLATE ENGINE SPEED BETWEEN TWO MAP SETTINGS
C          AND COMPUTE MAX AND MIN THROTTLE SETTINGS AT THAT SPEED
40    IM=I-1
      CO=(RPME-ERPM(IENG.I))/(ERPM(IENG.IM)-ERPM(IENG.I))
      TWOT=EMAP(I.20.K1)+CO*(EMAP(IM.20.K1)-EMAP(I.20.K1))
      TMIN=EMAP(I. 1.K1)+CO*(EMAP(IM. 1.K1)-EMAP(I. 1.K1))
      IF(LWOT)
      RETURN
41    CONTINUE
C       CHECK FOR TORQUES OFF THE MAP
```

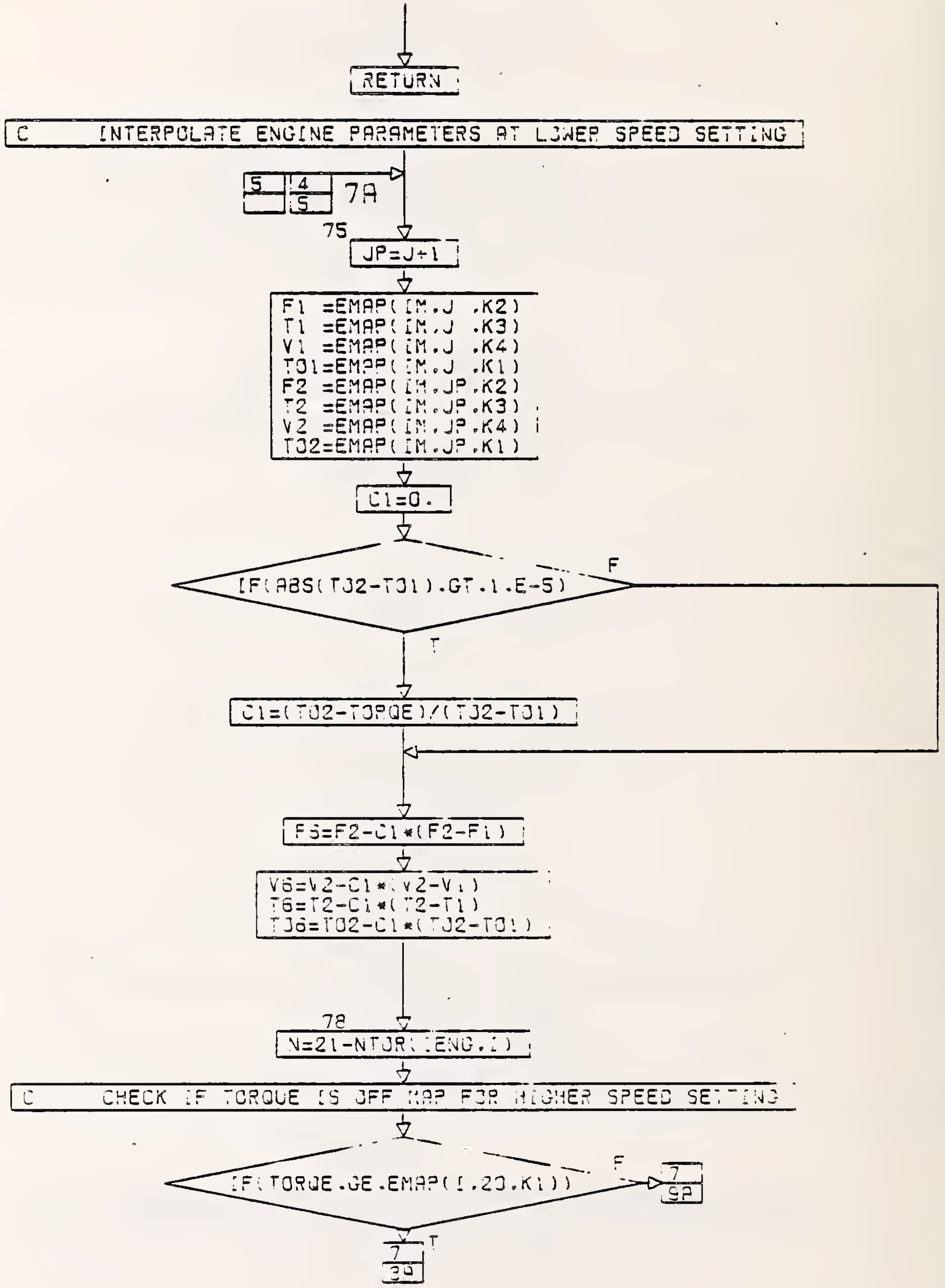
CONT. ON PG 4


IF(TORQE.GE.TWCT)
F
T
GO TO 99
10
52
IF(TORQE.LE.TMIN)
F
T
GO TO 37
9
23
IF(TORQE.GE.EMAP(IM,20,K1))
F
T
GO TO 50
5
59
IF(TORQE.GT.EMAP(IM, 1,K1))
F
T
GO TO 60
5
62
C TORQUE IS OFF MAP AT LOW END FOR LOWER SPEED SETTING
J=1
GO TO 75
3
72

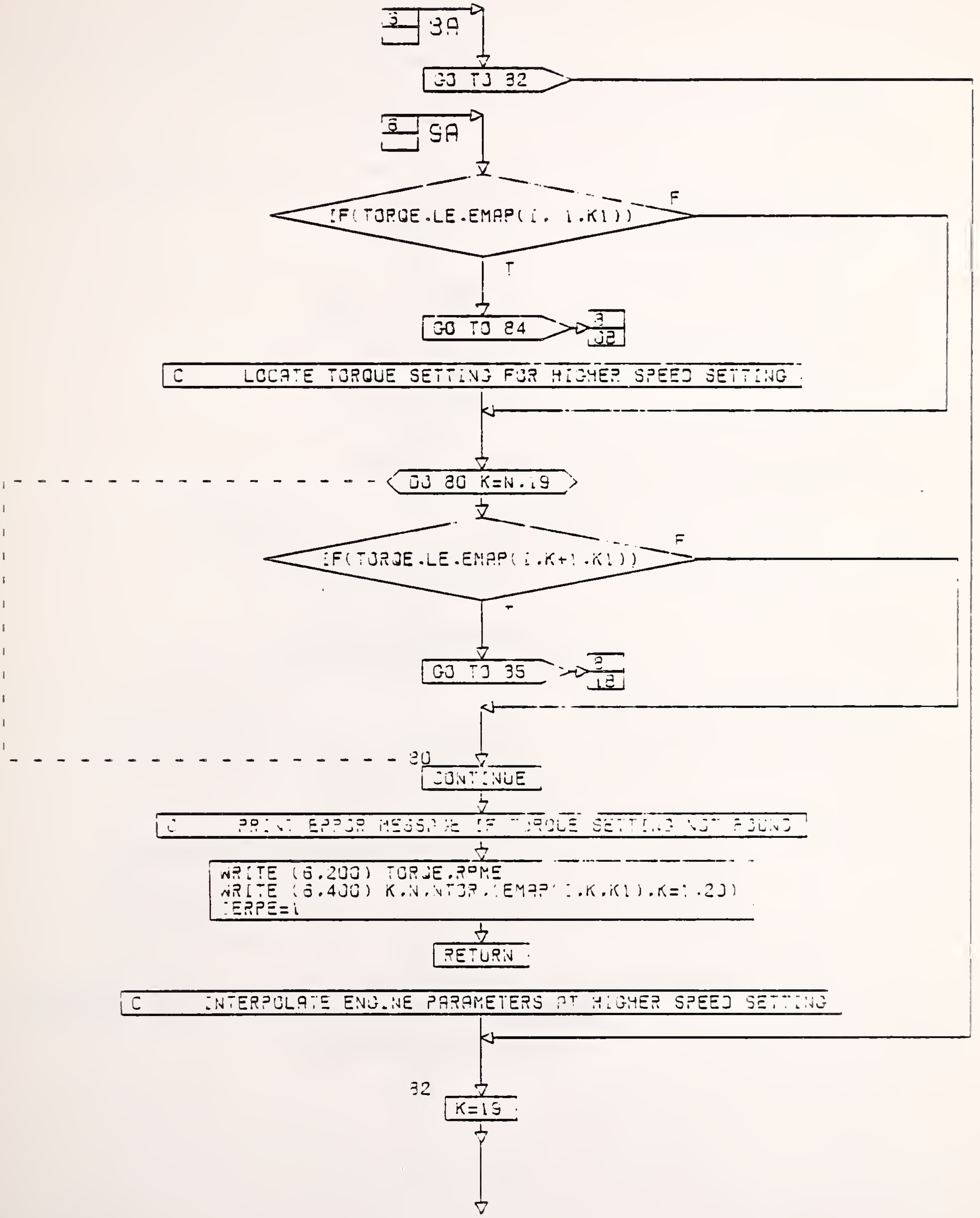
CONT. ON PG 5


C TORQUE IS OFF MAP AT HIGH END FOR LOWER SPEED SETTING
4
5A
50
J=19
GO TO 75
6
7A
C SEARCH FOR TORQUE SETTING ON LOWER SPEED SETTING
4
6A
60
N=21-NTOR(IENG.IM)
DO 70 J=N.19
IF(TORQE.LE.EMAP(IM.J+1.K1))
F
T
GO TO 75
5
7A
70
CONTINUE
C PRINT ERROR MESSAGE IF TORQUE SETTING NOT FOUND
WRITE (6.200) TORQE.RPME
200
FORMAT (//2X.5X.60(1H*)//2X.34H***** FAILURE TO FIND TORQUE SETT
1 13HNG FOR ENGINE/2X.15H***** TORQUE = .E15.7. 5X.6HRPM
2 E15.7/2X.25H***** EXECUTION CONTINUES//2X.5X.60(1H*)//)
WRITE (6.400) J.N.NTOR.(EMAP(IM.J.K1).J=1.20)
400
FORMAT(2I5/20I5/(10F10.2))
IERRE=1

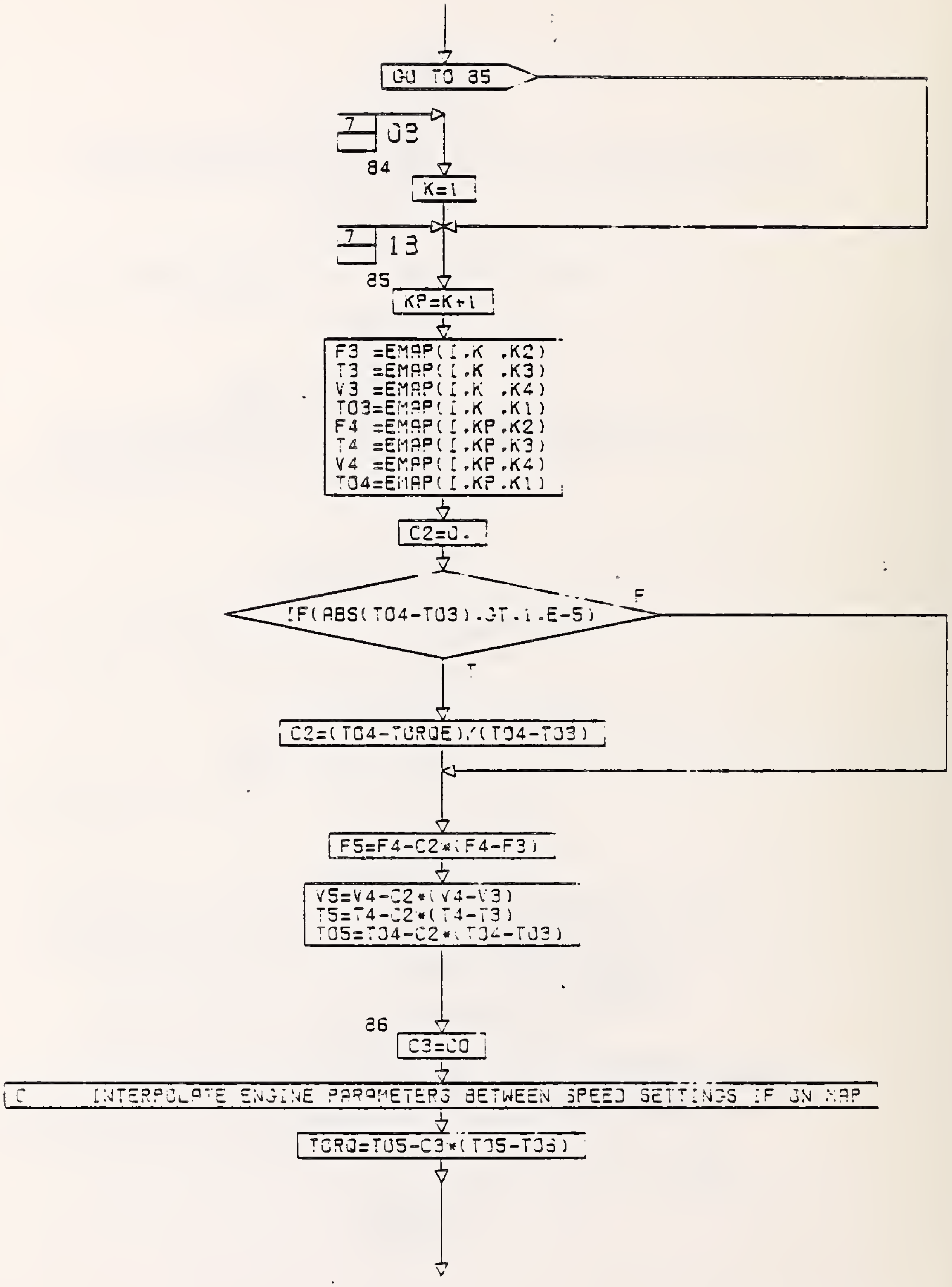
CONT. ON PG 6


RETURN
C INTERPOLATE ENGINE PARAMETERS AT LOWER SPEED SETTING
5
4
5
7A
75
JP=J+1
F1 =EMAP(IM,J ,K2)
T1 =EMAP(IM,J ,K3)
V1 =EMAP(IM,J ,K4)
TO1=EMAP(IM,J ,K1)
F2 =EMAP(IM,JP,K2)
T2 =EMAP(IM,JP,K3)
V2 =EMAP(IM,JP,K4)
TO2=EMAP(IM,JP,K1)
C1=0.
IF(ABS(TO2-TO1).GT.1.E-5)
F
T
C1=(TO2-TORQE)/(TO2-TO1)
F6=F2-C1*(F2-F1)
V6=V2-C1*(V2-V1)
T6=T2-C1*(T2-T1)
TO6=TO2-C1*(TO2-TO1)
78
N=21-NTOR(IENG,I)
C CHECK IF TORQUE IS OFF MAP FOR HIGHER SPEED SETTING
IF(TORQE.GE.EMAP(I,20,K1))
F
7
9P
T
7
9A

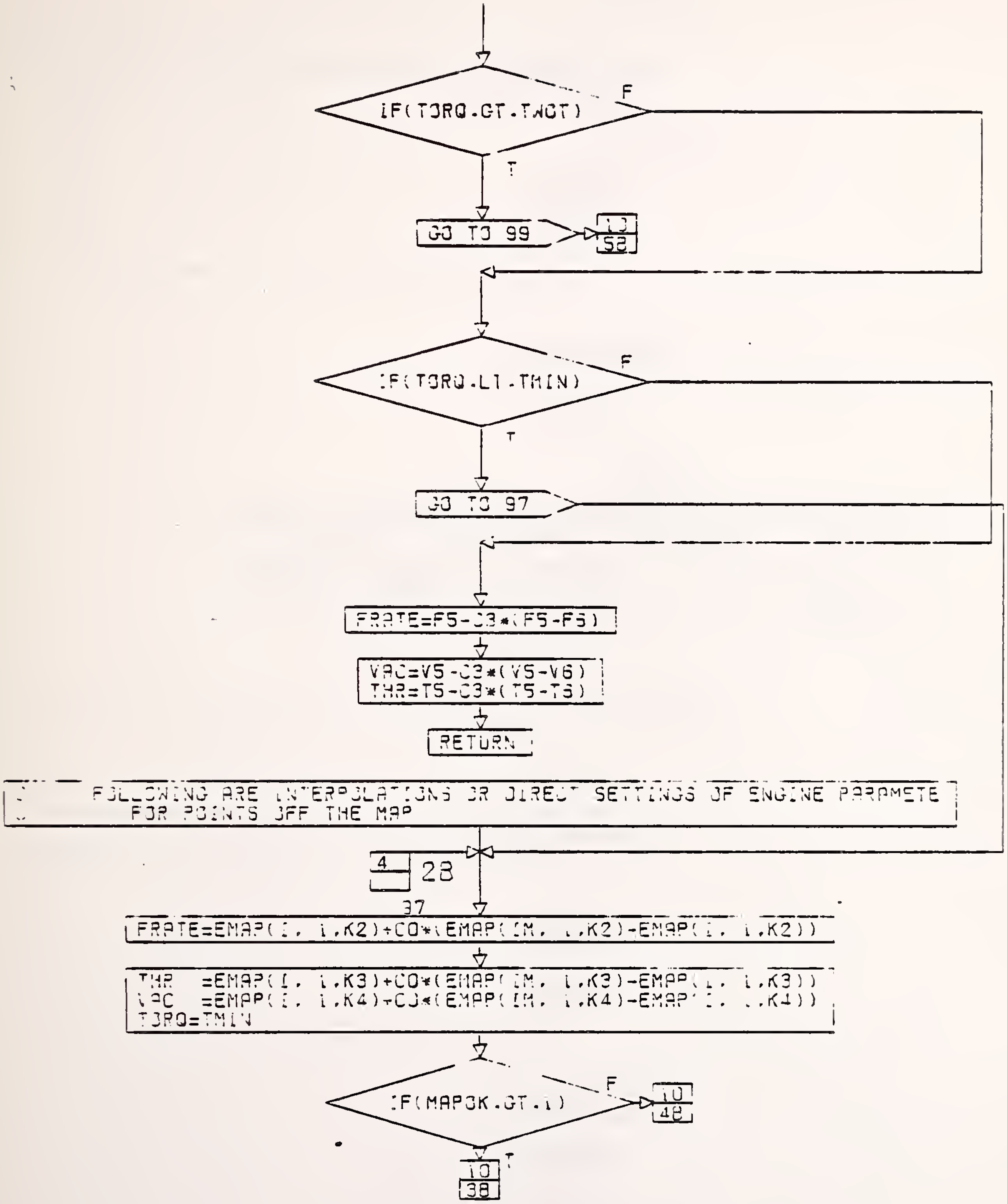
CONT. ON PG 7


3
3A
GO TO 32
6
9A
IF(TORQE.LE.EMAP(I, 1,K1))
F
T
GO TO 84
3
12
C LOCATE TORQUE SETTING FOR HIGHER SPEED SETTING
DO 80 K=N,19
IF(TORQE.LE.EMAP(I,K+1,K1))
F
T
GO TO 35
8
12
80
CONTINUE
C PRINT ERROR MESSAGE IF TORQUE SETTING NOT FOUND
WRITE (6,200) TORQE,RPME
WRITE (6,400) K,N,NTOR,(EMAP(I,K,K1),K=1,20)
IERRE=1
RETURN
C INTERPOLATE ENGINE PARAMETERS AT HIGHER SPEED SETTING
32
K=19

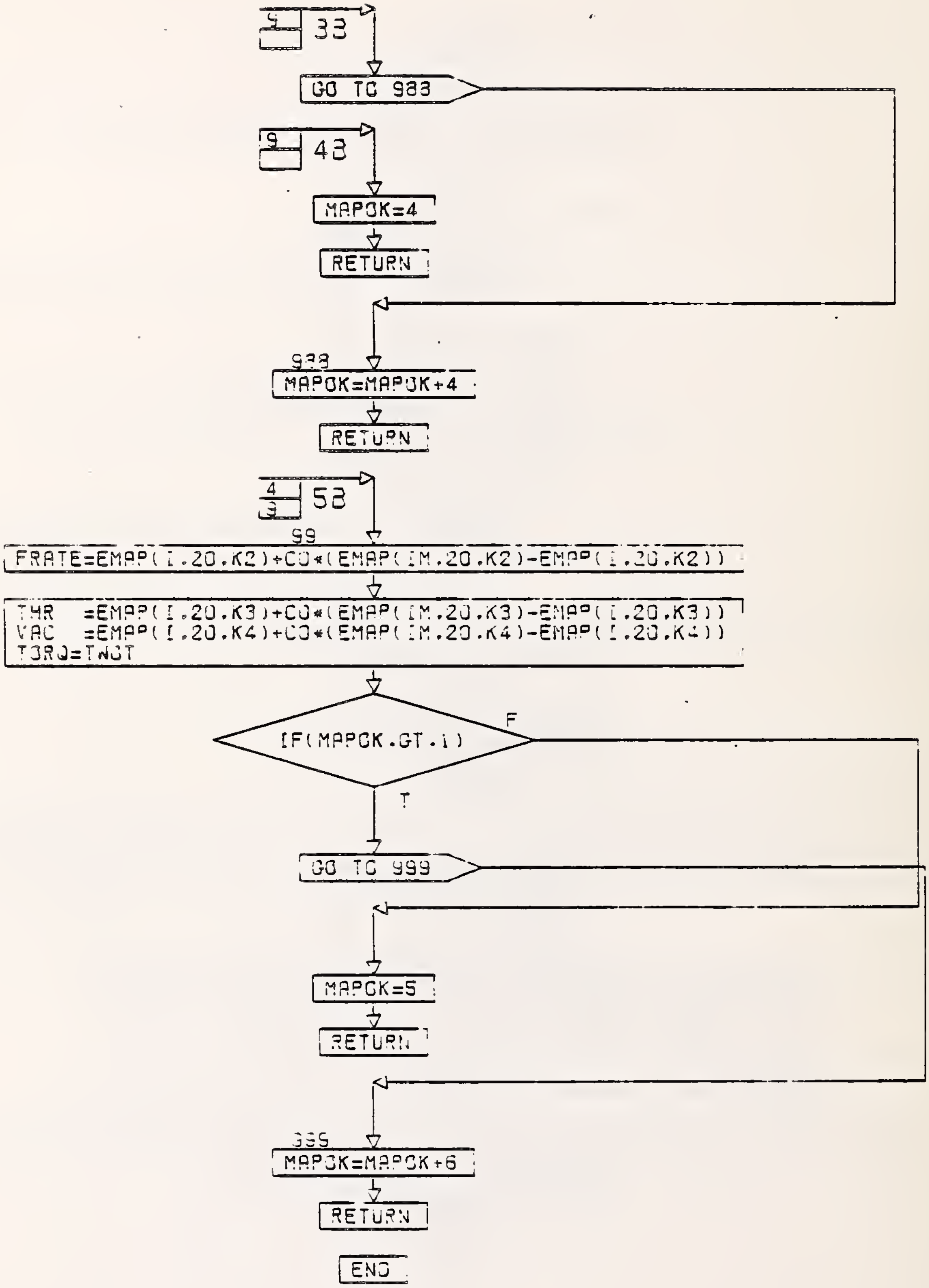
CONT. ON PG 8


GO TO 85
7
03
84
K=1
7
13
85
KP=K+1
F3 =EMAP(I,K ,K2)
T3 =EMAP(I,K ,K3)
V3 =EMAP(I,K ,K4)
TO3=EMAP(I,K ,K1)
F4 =EMAP(I,KP,K2)
T4 =EMAP(I,KP,K3)
V4 =EMAP(I,KP,K4)
TO4=EMAP(I,KP,K1)
C2=0.
IF(ABS(TO4-TO3).GT.1.E-5)
F
T
C2=(TO4-TORQE)/(TO4-TO3)
F5=F4-C2*(F4-F3)
V5=V4-C2*(V4-V3)
T5=T4-C2*(T4-T3)
TO5=TO4-C2*(TO4-TO3)
86
C3=C0
C INTERPOLATE ENGINE PARAMETERS BETWEEN SPEED SETTINGS IF ON MAP
TORQ=TO5-C3*(TO5-TO6)

CONT ON PG 9


IF(TORQ.GT.TMOT)
F
T
GO TO 99
10
52
IF(TORQ.LT.TMIN)
F
T
GO TO 97
FRATE=F5-C3*(F5-F6)
VAC=V5-C3*(V5-V6)
THR=T5-C3*(T5-T6)
RETURN
C FOLLOWING ARE INTERPOLATIONS OR DIRECT SETTINGS OF ENGINE PARAMETE
C FOR POINTS OFF THE MAP
4
28
97
FRATE=EMAP(I, 1,K2)+CO*(EMAP(IM, 1,K2)-EMAP(I, 1,K2))
THR =EMAP(I, 1,K3)+CO*(EMAP(IM, 1,K3)-EMAP(I, 1,K3))
VAC =EMAP(I, 1,K4)+CO*(EMAP(IM, 1,K4)-EMAP(I, 1,K4))
TORQ=TMIN
IF(MAPOK.GT.1)
F
10
48
T
10
38

CONT. ON PG 10


9
33
GO TO 988
9
48
MAPOK=4
RETURN
988
MAPOK=MAPOK+4
RETURN
4
9
58
99
FRATE=EMAP(I.20.K2)+CO*(EMAP(IM.20.K2)-EMAP(I.20.K2))
THR =EMAP(I.20.K3)+CO*(EMAP(IM.20.K3)-EMAP(I.20.K3))
VAC =EMAP(I.20.K4)+CO*(EMAP(IM.20.K4)-EMAP(I.20.K4))
TORQ=TWOT
IF(MAPOK.GT.1)
F
T
GO TO 999
MAPOK=5
RETURN
999
MAPOK=MAPOK+6
RETURN
END

PG 10 FINAL

4. GRAPHICS PACKAGE

```
      PROGRAM SURF3D
C
C  THIS PROGRAM BUILDS A 3D PICTURE FROM YOUR DATA
C    THE DATA YOU SUPPLY SHALL BE  IN MATRIX  FORM  IN (10x,3(f8.2,5x))FORMAT AND STORED ON DATA FILE 'vas.dat'
C      ON DISK.
C  IF YOU ARE   WORKING WITH ENGINE  DATA THERE IS A PROGRAM MB0001.FOR THAT WILL
C  TRANSFORM  YOUR  STANDARD ENGINE DATA INTO   AN ARRAY
C  OF 5-DIMENSIONAL VECTORS AND STORE'EM ON DISK.
C  LEFT COLUMN OF THIS MATRIX IS FOR X, MIDDLE - FOR Y,  & RIGHT  COLUMN IS FOR  Z .
C   YOU CAN RUN THE PROGRAM ON TEXTRONIX TERMINAL, OR YOU CAN USE TAPE. JUST KILL  UNEEDED LINE
C  'CALL UDEV( )'
C  YOUR DATA NEED NOT BE NECESSERELY 'A GOOD DATA'. BY THIS WE MEAN THAT  XX & YY  MAY NOT BE A
C  PERFECT  2D  NETWORK. WE DO THE FIXING - BY MEANS OF ITERATION WE BRING THE MATRIX TO THE
C   ACCEPTABLE SHAPE.
C  THE PROGRAM REQUIRES INPUT FROM THE TTY  THE NUMBERS NX AND NY WHICH
C  CORRESPOND TO THE NUMBER OF COLUMNS AND ROWS IN YOUR X-Y MATRIX.
      DIMENSION X(200),Y(200),Z(200)
      DIMENSION  ADAT(2500),ZDAT(25,25)                      !ZDAT(XX,YY),ADAT>2000
      DIMENSION A(25,25),B(500)                                             !A(XX,YY)
      EQUIVALENCE (A(1,1),B(1))
      COMMON XL(4000)                                        !IN() SHOULD BE  .EQ. OR >4000
      EQUIVALENCE(XL(1), ADAT(1)),(Z(1), ZDAT(1,1))

C     SELECT THE PLOTTING DEVICE

c     CALL UDEV('TTY','TK4015')
      CALL UDEV('16','OFFLINE')

C     INPUT VALUES NX AND NY.
C     THE DEFAULT VALUES FOR NX IS 20 AND FOR NY IS 10.

      TYPE 20
 20   FORMAT(' INPUT NX=',$)
      ACCEPT 30, NX
 30   FORMAT(I2)
      TYPE 40
 40   FORMAT(' INPUT NY=',$)
      ACCEPT 30, NY
      NNX=NX
      IF(NX.NE.0)GO TO 88
      NNX=20
88    NNY=NY
      IF(NY.NE.0)GO TO 89
      NNY=10
89    CONTINUE
      CALL BGNPL(0)

      CALL INTAXS
      CALL ZAXANG(90.)
      CALL COMPLX
      CALL BASALF('L/CST')
      CALL MIXALF('STAND')

C     DEFINE 3D WORKING AREA AND AXES

      CALL TITL3D('(ENGINE MAP)$',-100,9.5,6.5)
      CALL AXES3D('RPM$',100,'TORQUE$',100,'FUEL RATE$',    !NAMES OF THE AXES
      1      100,10.,10.,8.)

C     YOU MAY CHANGE THE POINT OF VIEW

      CALL VIEW(-200.,450.,400)
      CALL GRAF3D(550.,400.,3200.,-100.,40.,300.,0.,10.,70.)
```

```
C       FILE VAS.DAT  IS GIVEN A LOGICAL #33

        OPEN(UNIT=33,FILE='VAS.DAT')
        CALL BGNMAT(20,10)


C       DON'T WORRY ABOUT THE NUMBER OF INPUT DATA TO BE READ.
C       PROGRAM WILL TAKE CARE OF THIS.

        DO 5 I=1,54321
        READ(33,4,END=201),X(I),Y(I),Z(I)

C       GO TO THE I/O WINDOW AND GET AN OUTPUT.
C       SEE WETHER YOU SUPPLY THE RIGHT DATA.

        WRITE(3,4),X(I),Y(I),Z(I)
 4      FORMAT(10X,3(F8.2,5X))
 5      CALL GETMAT(X,Y,ZDAT,1,B)

 201    CONTINUE

        CALL ENDMAT(A,B)

C       NOW WE GONNA DRAW THE SURFACE. BE PREPARED!

        CALL SURMAT(ZDAT,1,NNX,1,NNY,ADAT)
        CALL DASH
        XDEL=0.
        DO 10J=1,3
        CALL BSHIFT(XDEL,0)
        CALL RLVEC3(-100.,4400.,ZDAT(1,1),-100.,4400.,0.,0)
        CALL RLVEC3(3000.,500.,ZDAT(NNY,1),3000.,500.,0.,0)
        CALL RLVEC3(3000.,2000.,160.0,3000.,2000.,0.,0)
        XDEL=XDEL+0.01
 10     CONTINUE
        CALL RESET('DOT')
        CALL RESET('BSHIFT')

         CALL GRFITI(0.,0.,0.0,1.,0.,0.0,0.,1.,0.)
        CALL INTAXS
        CALL TITLE(1H ,1,1H ,0,1H ,0,10.,10.)
        CALL GRAF(100.,200.,300.,800.,400.,4000.)

        CALL END3GR(0)
        CALL ENDPL(0)
        CALL DONEPL
        END
```

```
      PROGRAM SURF2D
C  THIS PROGRAM BUILDS A 2D PICTURE FROM YOUR DATA ON A THREE DIMENSIONAL AXIS SYSTEM.
C  IT DRAWS  THE CONTOUR LINES ON THE X-Y PLANE (SO-CALLED EQUIPOTENTIAL LINES, OR GEODESICAL LINES).
C  THE DATA YOU SUPPLY SHALL BE  IN MATRIX  FORM  IN (10X,3(F8.2,5X))FORMAT AND STORED ON DATA FILE 'VAS.DAT'
C  ON DISKB.
C  IF YOU ARE   WORKING WITH ENGINE  DATA THERE IS A PROGRAM MBDOD1.FOR THAT WILL
C  TRANSFORM  YOUR  STANDARD ENGINE DATA INTO   AN ARRAY
C  OF 5-DIMENSIONAL VECTORS AND STORE'EM ON DISK.
C  LEFT COLUMN OF THIS MATRIX IS FOR X, MIDDLE - FOR Y,  & RIGHT  COLUMN IS FOR  Z .
C   YOU CAN RUN THE PROGRAM ON TEXTRONIX TERMINAL, OR YOU CAN USE TAPE. JUST KILL  UNEEDED LINE
C  'CALL UOEV( )'
C  YOUR DATA NEED NOT BE NECESSERELY 'A GOOD DATA'. BY THIS WE MEAN THAT  XX & YY  MAY NOT BE A
C  PERFECT  2D  NETWORK. WE DO THE FIXING - BY MEANS OF ITERATION WE BRING THE MATRIX TO THE
C   ACCEPTIBLE SHAPE.
C  THE PROGRAM REQUIRES INPUT FROM THE TTY  THE NUMBERS NX AND NY WHICH
C  CORRESPOND TO THE NUMBER OF COLUMNS AND ROWS IN YOUR X-Y MATRIX.
      DIMENSION X(200),Y(200),Z(200)
      DIMENSION  ADAT(2500),ZDAT(25,25)                        !ZDAT(XX,YY),ADAT>2000
      DIMENSION A(25,25),B(500)                                                   !A(XX,YY)
      EQUIVALENCE (A(1,1),B(1))
      COMMON XL(4030)                                         !IN() SHOULD BE  .EQ. OR >4000
      EQUIVALENCE(XL(1), ADAT(1)),(Z(1), ZDAT(1,1))

C       SELECT THE PLOTTING DEVICE

C       CALL UOEV('TTY','TK4015')
        CALL UDEV('16','OFFLINE')

        TYPE 20
 20     FORMAT(' INPUT NX=',$)
        ACCEPT 30, NX
 30     FORMAT(I2)
        TYPE 40
 40     FORMAT(' INPUT NY=',$)
        ACCEPT 30, NY
        CALL BGNPL(0)

        CALL INTAXS
        CALL ZAXANG(90.)
        CALL COMPLX
        CALL BASALF('L/CST')
        CALL MIXALF('STAND')

C       DEFINE 3-D WORKING AREA AND  AXIS

        CALL TITL3D('(ENGINE MAP)$',-100,9.5,6.5)
        CALL AXES3D('RPM$',100,'TORQUE$',100,'FUEL RATE$',     !NAMES OF THE AXES
        1       100,10.,10.,8.)

        CALL VIEW(-200.,450.,400)                          !CHANGE THE POINT OF VIEW
        CALL GRAF3D(550.,400.,3200.,-100.,40.,300.,0.,10.,70.)
        OPEN(UNIT=33,FILE='VAS.DAT')
        CALL BGNMAT(NX,NY)

C       DO NOT WORRY ABOUT THE # OF DATA TO BE READ.
C       THE PROGRAM WILL TAKE CARE OF IT.

        DO 5 I=1,500                              !NO LESS THAN THE NO. OF POINTS

C       READ IN DATA FROM  FILE
```

```
      READ(33,4,END=201),X(I),Y(I),Z(I)
C     IF YOU GET FUNNY PICTURES GO TO THE I/O WINDOW AND
C     SEE WHAT KIND OF GARBADGE YOU SUPPLY.
      WRITE(3,4),X(I),Y(I),Z(I)
4     FORMAT(10X,3(F8.2,5X))
5     CALL GETMAT(X,Y,ZDAT,I,8)
201   CONTINUE
      CALL ENOMAT(A,B)
      CALL DASH
      XDEL=0.
      DO 10J=1,3
      CALL BSHIFT(XDEL,0)
      CALL RLVEC3(-100.,4400.,ZDAT(1,1),-100.,4400.,0.,0)
      CALL RLVEC3(3000.,500.,ZDAT(NY,1),3000.,500.,0.,0)
      CALL RLVEC3(3000.,2000.,160.0,3000.,2000.,0.,0)
      XDEL=XDEL+0.01
10    CONTINUE
      CALL RESET('DOT')
      CALL RESET('BSHIFT')

       CALL GRFITI(0.,0.,0.0,1.,0.,0.0,0.,1.,0.)
      CALL INTAXS
      CALL TITLE(1H ,1,1H ,0,1H ,0,10.,10.)
      CALL GRAF(100.,200.,300.,800.,400.,4000.)
      CALL BCOMON(4000)
C     SET CONTURE PARAMETERS
      CALL CONMAK(ZDAT,NX,NY,1.)
      CALL CONLIN(4,'SOLID','LABELS',2,10)
      CALL CONANG(90.)
      CALL FRAME
      CALL HEIGHT(.3)
      CALL CONTUR(5,'LABELS','DRAW')
      CALL END3GR(0)
      CALL ENOPL(0)
      CALL DONEPL
      END
      PROGRAM HPOWER
C
C     THIS PROGRAM  DRAWS THE FAMILY OF CURVES
C     WHICH ARE DESCRIBED BY THE SAME ALGORITHM.
C     IN THIS PROGRAM  WE CALCULATE HORSEPOWER
C     OF THE ENGINE AS A FUNCTION OF TORQUE
C     AND RPM. WE DRAW  THE FAMILY OF CURVES
C     WHICH ARE HYPERBOLAS AND THEY REPRESENT
C     HORSEPOWERS  WITH THE INCREMENT OF 2 H.P.

      DIMENSION X(12),Y(12)
C     CALL UDEV('TTY','TK4015')
      CALL UDEV('16','OFFLINE')
      CALL BGNPL(0)
C
C     NN STANDS FOR THE NUMBER OF CURVES YOU WANT
C     TO BE DRAWN. THEY WILL BE DISTINGUISHED BY
C     THE MARKS.
C     INCR STAND  FOR THE STEP INCREMENT BETWEEN
C     THE CURVES

      TYPE 10
 10   FORMAT('  INPUT NN=',$)
      ACCEPT 20,NN
 20   FORMAT(I2)

      TYPE 30
 30   FORMAT(' INPUT INCR=',$)
      ACCEPT 40,INCR
 40   FORMAT(I2)

      CALL TITLE('HORSEPOWER AS A FUNCTION OF T AND RPM$',
     1        100,'RPM$',100,'TORQUE',6,7.,8.)
      CALL GRAF(1000.,200.,3200.,0.,30.,240.)
      CALL FRAME
      A=5250.
      B=2.
      DO 1 I=1,12
      X(1)=1000.
      X(I)=X(I-1)+200.
      Y(I)=A*(B)/X(I)
 1    CONTINUE
      CALL CURVE(X,Y,12,1)
      NN=NN-1
      DO 77 J=1,NN
      B=B+INCR
      DO 50 I=1,12
      Y(I)=A*B/X(I)
 50   CONTINUE
      CALL CURVE(X,Y,12,1)
 77   CONTINUE
      CALL ENDPL(0)
      CALL DONEPL
      STOP
      END
```

```
              PROGRAM GEO

C     THIS PROGRAM  DRAWS  CONTUR  LINES ON THE
C     2-D  GRAPH.  YOU SUPPLY Z POINTS THAT
C     CORRESPOND TO THE X-Y MATRIX.
C     Z POINTS ARE STORED IN THE DATA FILE 'VAS1.DAT'
C     ON DSKB
C     MAKE SURE THE BOUNDARIES IN CALL GRAF
C     STATEMENT CORRESPOND TO YOUR DATA IN THIS
C     SEQUENCE (BX,SX,EX,BY,SY,EY)
C     WHERE B - BEGINNING
C           S - STEP
C           E - END
C     PROGRAM REQUIRE INPUT FROM THE TTY NUMBERS
C     NX AND NY. THEY SHOULD CORRESPOND TO THE
C     NUMBERS OF COLUMNS AND ROWS IN YOUR MATRIX X-Y.
C     GOOD LUCK!
C     DEFAULT VALUES FOR NX IS 20 AND FOR NY IS 10
              COMMON WORK(5000)

C             YOUR X-Y MATRIX  MAY BE OF ANY SIZE BUT
C             NO LARGER THAN  W

              DIMENSION W(20,10)
              DIMENSION Z(200)
              EQUIVALENCE(Z(1),W(1,1))
              TYPE 10
     10       FORMAT(' INPUT NX=',$)
              ACCEPT 20,NX
     20       FORMAT(I2)

              TYPE 30
     30       FORMAT(' INPUT NY=',$)
              ACCEPT 40, NY
     40       FORMAT(I2)

              NNX=NX
              IF(NX.NE.0) GOTO 88
              NNX=20
     88       NNY=NY
              IF(NY.NE.0) GOTO 89
              NNY=10
     89       CONTINUE
C             SELECT PLOTTING DEVICE
              CALL UDEV('TTY','TK4015')
C             CALL UDEV('16','OFFLINE')
              OPEN(UNIT=33,FILE='VAS1.DAT')

C             DO NOT WORRY  ABOUT THE NUMBER OF DATA
C             TO BE READ. THE PROGRAM WILL TAKE CARE OF IT.
              DO 5 I=1,12345
              READ(33,4,END=201),Z(I)
     4        FORMAT(F5.1)
     5        CONTINUE
     201      CONTINUE
              CALL BGNPL(0)

C             DEFINE 2-D WORK AREA AND AXES
              CALL TITLE('FUEL RATES$',100,'RPM$',100,
         1          'TORQUE$',6,7.,8.)
              CALL GRAF(1000.,200.,3200.,0.,10.,240.)
```

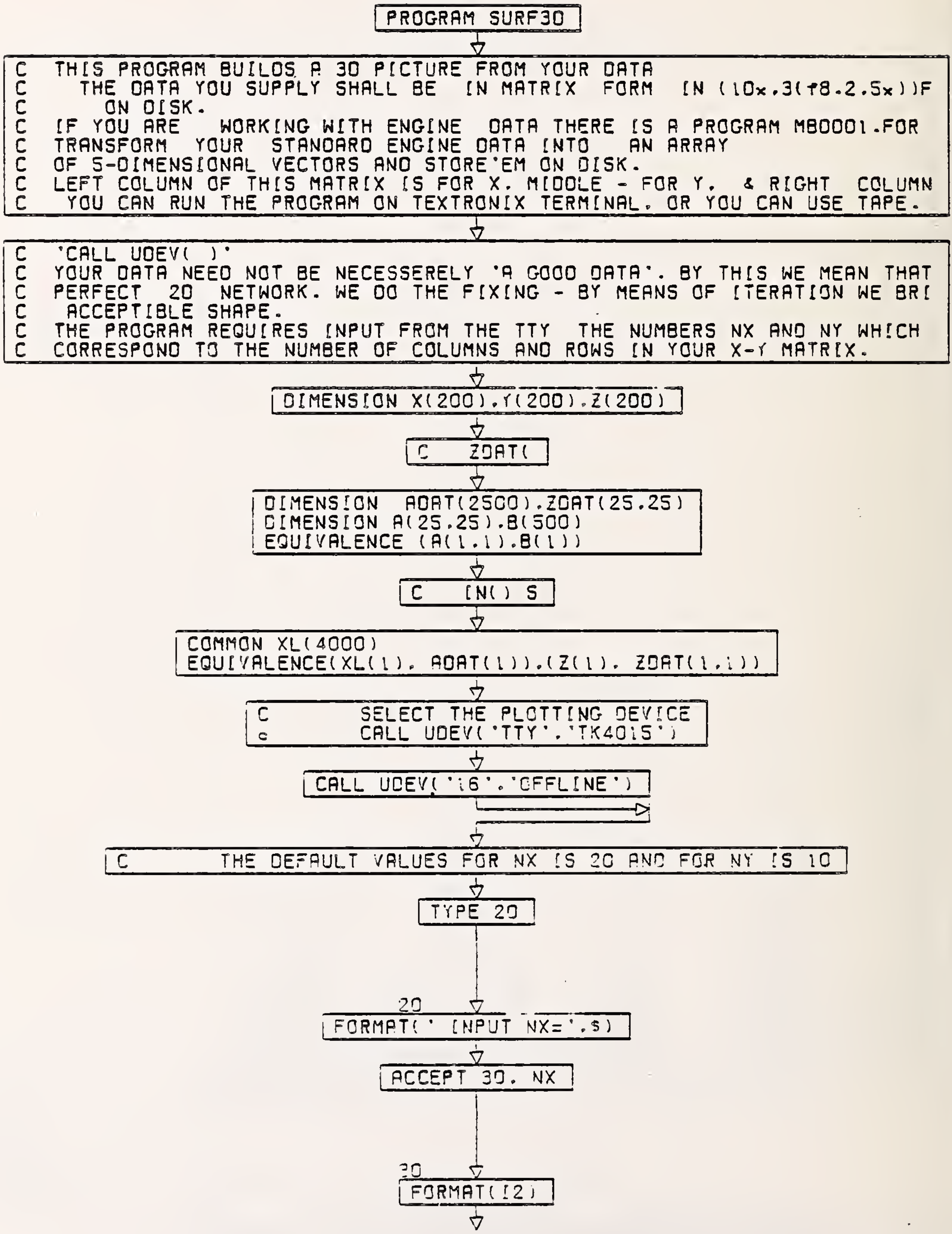
```
      CALL FRAME
      CALL  BCOMON(5000)

C     SET CONTURE PARAMETERS

      CALL CONMAK(W,NNX,NNY,5.)
      CALL CONLIN(0.,'SOLID','LABELS',2,3)
      CALL CONANG(90.)

C     ROUNDING OFF THE CURVES

      CALL PARA3
      CALL CONTUR(2,'LABELS','DRAW')
      CALL ENDPL(0)
      STOP
      END
      PROGRAM DIABOR
C     LINEAR INTERPOLATION PLUS GRID
C     THIS PROGRAM BUILDS A 2D PICTURE FROM
C     THE DATA YOU SUPPLY.
C     THE DATA YOU SUPPLY SHALL BE ON DSKB
C     IN THE FORMAT (10X,2(F8.2,5X))
C     IT IDENTIFIES THE CURVES AND
C     DRAWS THE GRID

      DIMENSION X1(200),Y1(200),Y2(200)

C     CHOOSE  THE DEVICE.
C     WE SUGGEST  TEXTRONICS TERMINAL
C     AND CALCOMP. THE LATER SHALL BE USED VIA
C     THE TAPE WHERE YOU  RECORDYOUR RUN.

CC    CALL UDEV('TTY','TK4015')
      CALL UDEV('16','OFFLINE')

c     FILE 'VAS.DAT' WAS GIVEN LOGICAL NAME 22

      OPEN(UNIT=22,FILE='VAS.DAT')

      IM= 1

C     INPUT NUMBER IM WHICH STANDS FOR MARKER
C     SPECIFICATIONS:
C     +1   -  POINTS CONNECTED AND SYMBOL AT EVERY POINT
C     -1   -  POINTS NOT CONNECTED AND A SYMBOL AT EVERY POINT
C      0   -  POINTS CONNECTED WITH NO SYMBOLS DRAWN
C     A SYMBOL WILL ALWAYS APPEAR AT THE FIRST AND LAST
C     POINTS IF IM.NE.0  REGARDLESS OF THE VALUE IM


      TYPE 10
10    FORMAT('  INPUT IM=',$)
      ACCEPT 20, IM
20    FORMAT (I2)

C     IF YOU WANT TO DRAW MORE THAN ONE CURVE
C     OPEN UP FEW MORE FILES IN EXACTLY THE SAME
C     FASHION AND READ THEIR CONTENTS. THAT WILL DO.

      DO 5 I=1,200
      READ(22,100,END=201) X1(I),Y1(I)

C     HIGH SPEED PRINTOUT WILL SHOW LATER WHETHER
C     YOU FEED APPROPRIATE DATA

      WRITE(3,100,END=201) X1(I),Y1(I)
 5    CONTINUE
 100  FORMAT(10X,2(F8.2,5X))
 201  CONTINUE

C     'ENGINE STATISTICS', 'RPM' AND 'TORQUE'
C     ARE  TITLE, X  AND Y AXES CORRESPONDINGLY.
C     YOU MAY CHANGE THOSE NAMES

      CALL TITLE('ENGINE  STATISTICS$',100, 'RPM$', 100,
      1       'TORQUE',7,6., 8.)

C     DRAW THE AXIS SYSTEM

      CALL GRAF(0.,500.,4000., -100.,50.,250.)
```

```
C       DRAW THE  GRID.

        CALL GRID(1,2)
        CALL FRAME

C       DRAW THE CURVE THRU THE GIVEN POINTS.
C       CONNECT THEM  WITH  STRAIGHT LINES.
C       OPTIONALLY IT  IS POSSIBLE TO  CONNECT
C       POINTS WITH  NON-LINEAR LINES.

        CALL CURVE(X1, Y1, 200,IM)
C
C       IF YOU WANT TO DRAW MORE THAN ONE CURVE
C
C       CALL CURVE(X2, Y2, 200, -1)

C       A MESSAGE WILL APPEAR IN THE RIGHT CORNER OF
C       YOUR DRAWING.

        CALL RLMESS ('0-19$',4, 6.0 ,0.90)
        CALL RLMESS('20-39$' , 5, 6.0, 0.85)

        CALL ENDPL(60)
        CALL DONEPL
        CLOSE(UNIT=22)
        STOP
        END
```

```
      PROGRAM  TRANSF
C     THIS PROGRAM  TRANSFORMS  THE VEHSIM OUTPUT
C     DATA INTO AN ARRAY OF 5-DIMENSIONAL VECTORS.
C     THIS PROGRAM CALLS ONE SUBROUTINE MB0002.
C     THIS PROGRAM IS MACHINE-INDEPENDENT AND
C     CAN BE RUN ON ANY COMPUTER.
C     AS AN EXAMPLE  LETS CONSIDER AN OUTPUT
C     'ENGINE DATA'. THIS PROGRAM WILL TRANSFORM IT
C     INTO AN  ARRAY OF VECTORS WHERE COLUMN ONE
C     REPRESENTS SPEED(RPM), TWO - TORQUE(FT-LB),
C     THREE - FUEL RATE(LB/HR), FOUR - THROTTLE(DEGREES),
C     FIVE - MANIFOLD VACUUM(IN-HG). WITH THIS
C     KIND OF OUTPUT IT IS  VERY EASY TO AUTOMATE
C     THE  INPUT PROCESS FOR A WIDE RANGE OF
C     PROGRAMS (NOT NECESSARILY CONNECTED WITH
C     VEHSIM).
C     AS AN  INPUT FOR THIS PROGRAM  WE USE
C     FILE NAMED 'MB0001.DAT' ON DISKB.
C     OUTPUT WILL HAVE  NAME 'VAS.DAT' AND
C     WILL BE STORED ALSO ON DISKB.

      COMMON P,F
      DIMENSION ENG(200,5)
      INTEGER P(26),F(80),KK,SS
      INTEGER CC,H,H1,H4,MM,RR
      REAL F(10),A(1000),SP(10)
      DATA SP /550.00,800.00,1000.00,1200.00,1400.00,
     1 1600.00,2000.00,2400.00,2800.00,3200.00/
      OPEN(UNIT=21,FILE='MB0001.DAT')
      OPEN(UNIT=22,FILE='VAS.DAT')
      KK=1
      SS=0
      CC=20;H=12;H1=0;H4=4
1     READ(21,10,END=1000)P
10    FORMAT(26A5)
      SS=SS+1
      IF(SS.GE.139)GO TO 1
      IF(SS.EQ.65)GO TO 35
      IF(SS.EQ.126)GO TO 35
      GO TO 40
35    CC=CC+1;GO TO 1
40    CONTINUE
      IF(SS.LE.17)GO TO 1
      IF(SS.NE.CC)GO TO 1
      H1=H1+1
      CC=CC+1
      IF(H1.EQ.4)CC=CC+1
      IF(H1.EQ.4)H1=H1+1
      IF(H1.EQ.9)CC=CC+3
      IF(H1.EQ.9)H1=0
      CALL MB0002
C     TYPE 20,P
20    FORMAT(1X,26A5)
C     TYPE 45,F
45    FORMAT(35X,10(F8.2,2X))
      DO 30 I=1,10
      A(KK+I-1)=F(I)
30    CONTINUE
      KK=KK+10
      GO TO 1
```

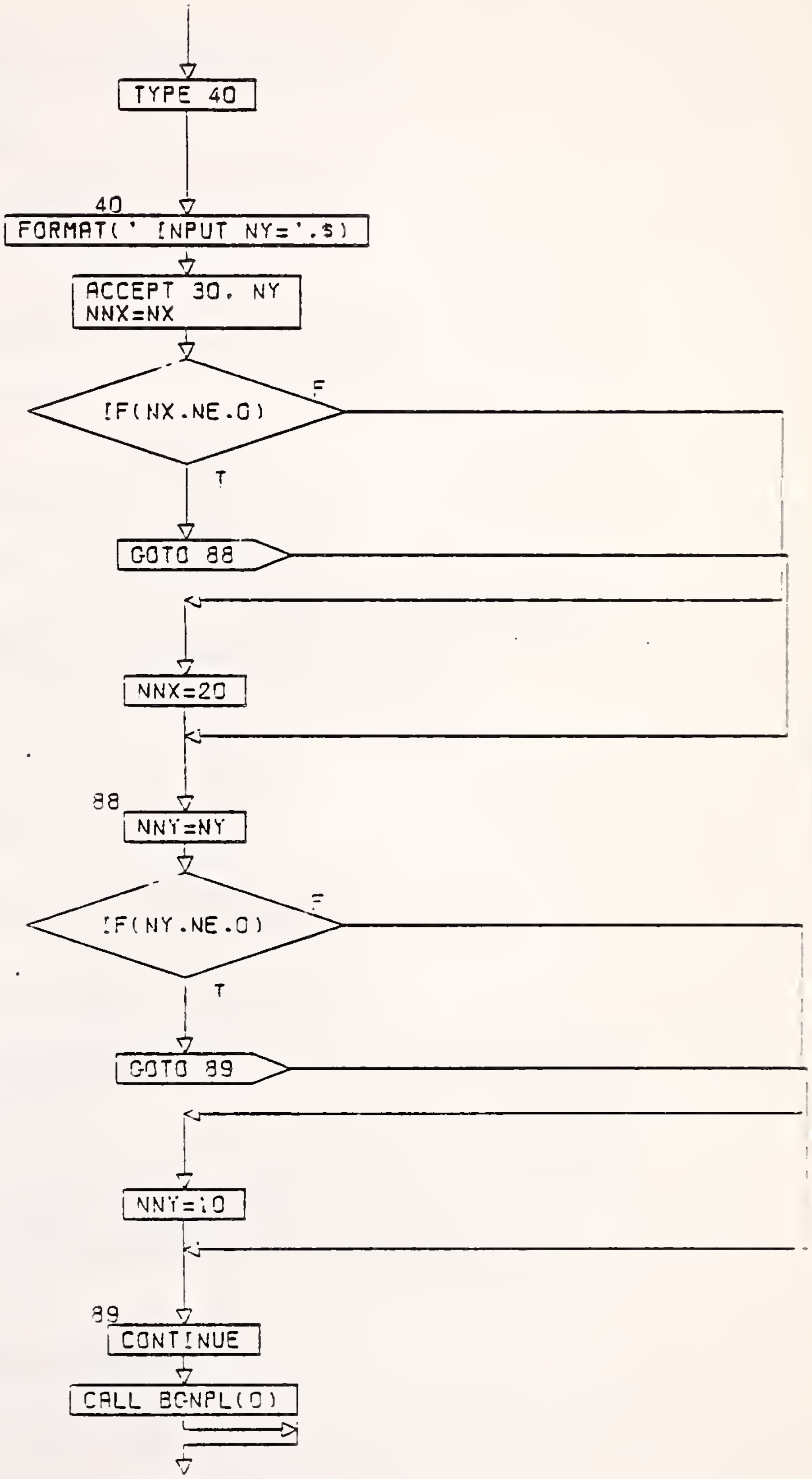
```
1000    CONTINUE
        DO 60 I=1,10
        DO 60 J=1,2
        DO 60 K=1,10
        RR=I*20-20+J*10-10+K
        ENG(RR,1)=SP(I)
        DO 60 L=1,4
        MM=I*80-80+J*40-40+L*10-10+K
        ENG(RR,L+1)=A(MM)
60      CONTINUE
        DO 110 I=1,200
C       TYPE 100,(ENG(I,J),J=1,5)
        WRITE(22,100)(ENG(I,J),J=1,5)
100     FORMAT(10X,5(F8.2,5X))
110     CONTINUE
        CLOSE(UNIT=21)
        CLOSE(UNIT=22)
        STOP
        END
```

```
      SUBROUTINE MB0002

C     THIS SUBROUTINE IS A PART OF A PROGRAM THAT
C     TRANSFORMS  VEHSIM'S OUTPUT DATA INTO
C     WORKABLE SHAPE. IT  IS BEING CALLED BY
C     TRANS.FOR.

      COMMON P,F
      INTEGER R(130),MM,P(26),S,NS,H(10)
      REAL F(10)
      DOUBLE PRECISION RE
      DECODE(130,10,P)R
   10 FORMAT(130A1)
      I=0
      NS=0
      K=0
   20 I=I+1
      IF(R(I).GE.'0'.AND.R(I).LE.'9')GO TO 40
      GO TO 20
   40 DO 50 J=1,10
   50 H(J)=' '
      S=R(I-1)
      K=0
   30 K=K+1
      H(K+1)=R(I)
      I=I+1
      IF(R(I).NE.' ')GO TO 30
      IF(S.EQ.' ')GO TO 60
      H(1)=S
      GO TO 70
   60 DO 70 J=2,10
      H(J-1)=H(J)
   70 CONTINUE
      NS=NS+1
      IF(NS.GE.11)GO TO 1000
      ENCODE(10,80,RE)H
   80 FORMAT(10A1)
      DECODE(10,90,RE)F(NS)
   90 FORMAT(F8.2)
      GO TO 20
 1000 RETURN
      END
```

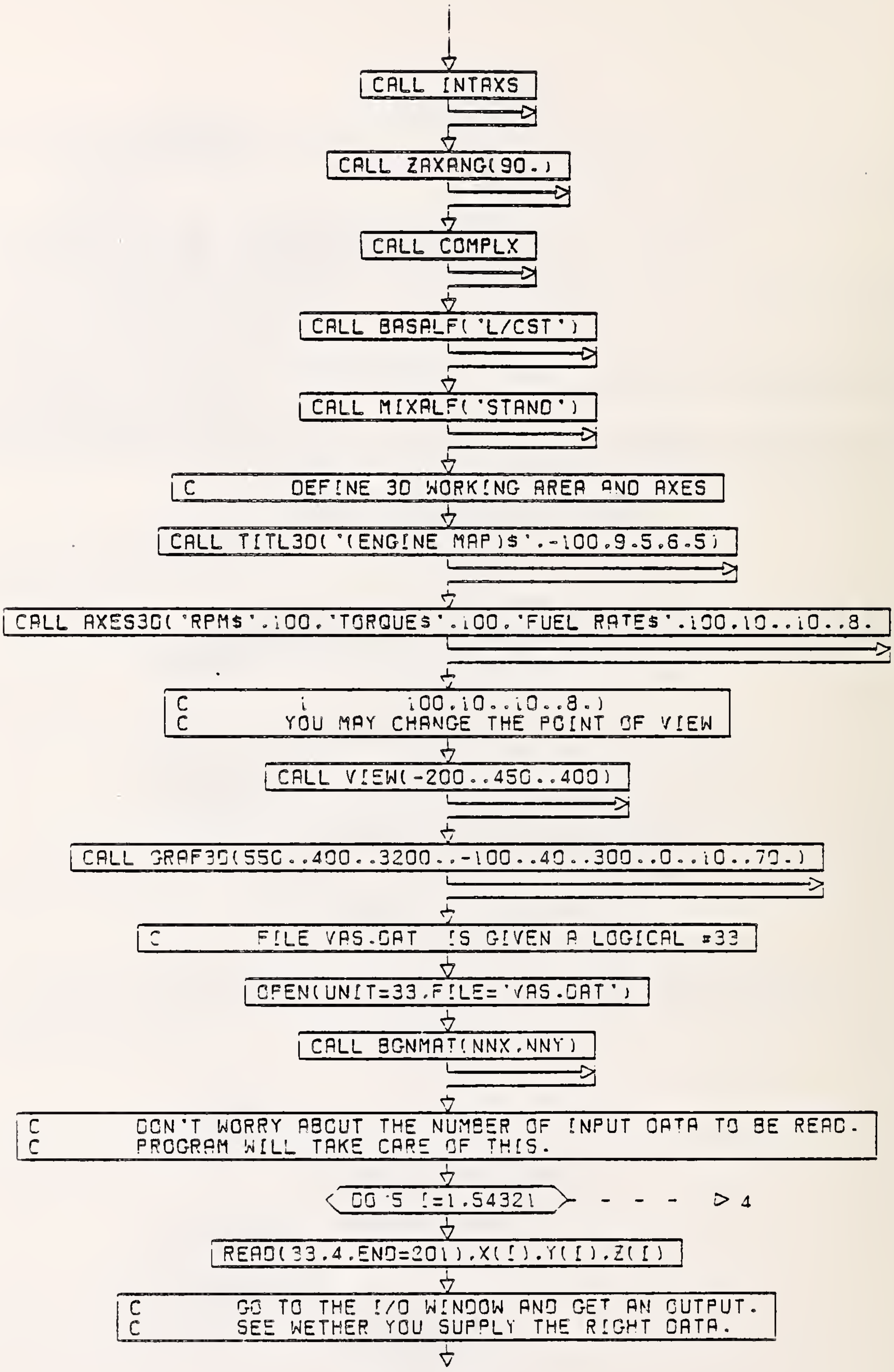


```
PROGRAM SURF3D
C  THIS PROGRAM BUILDS A 3D PICTURE FROM YOUR DATA
C    THE DATA YOU SUPPLY SHALL BE  IN MATRIX  FORM  IN (10x.3(F8.2.5x))F
C      ON DISK.
C  IF YOU ARE   WORKING WITH ENGINE  DATA THERE IS A PROGRAM M80001.FOR
C  TRANSFORM  YOUR  STANDARD ENGINE DATA INTO   AN ARRAY
C  OF 5-DIMENSIONAL VECTORS AND STORE'EM ON DISK.
C  LEFT COLUMN OF THIS MATRIX IS FOR X. MIDDLE - FOR Y.  & RIGHT  COLUMN
C   YOU CAN RUN THE PROGRAM ON TEXTRONIX TERMINAL. OR YOU CAN USE TAPE.

C  'CALL UDEV( )'
C  YOUR DATA NEED NOT BE NECESSERELY 'A GOOD DATA'. BY THIS WE MEAN THAT
C  PERFECT  2D  NETWORK. WE DO THE FIXING - BY MEANS OF ITERATION WE BRI
C   ACCEPTIBLE SHAPE.
C  THE PROGRAM REQUIRES INPUT FROM THE TTY  THE NUMBERS NX AND NY WHICH
C  CORRESPOND TO THE NUMBER OF COLUMNS AND ROWS IN YOUR X-Y MATRIX.

DIMENSION X(200).Y(200).Z(200)

C   ZDAT(

DIMENSION  ADAT(2500).ZDAT(25.25)
DIMENSION A(25.25).B(500)
EQUIVALENCE (A(1.1).B(1))

C   IN() S

COMMON XL(4000)
EQUIVALENCE(XL(1). ADAT(1)).(Z(1). ZDAT(1.1))

C        SELECT THE PLOTTING DEVICE
c        CALL UDEV('TTY'.'TK4015')

CALL UDEV('16'.'OFFLINE')

C          THE DEFAULT VALUES FOR NX IS 20 AND FOR NY IS 10

TYPE 20

20
FORMAT(' INPUT NX='.$)

ACCEPT 30. NX

30
FORMAT(I2)
```

CONT. ON PG 2

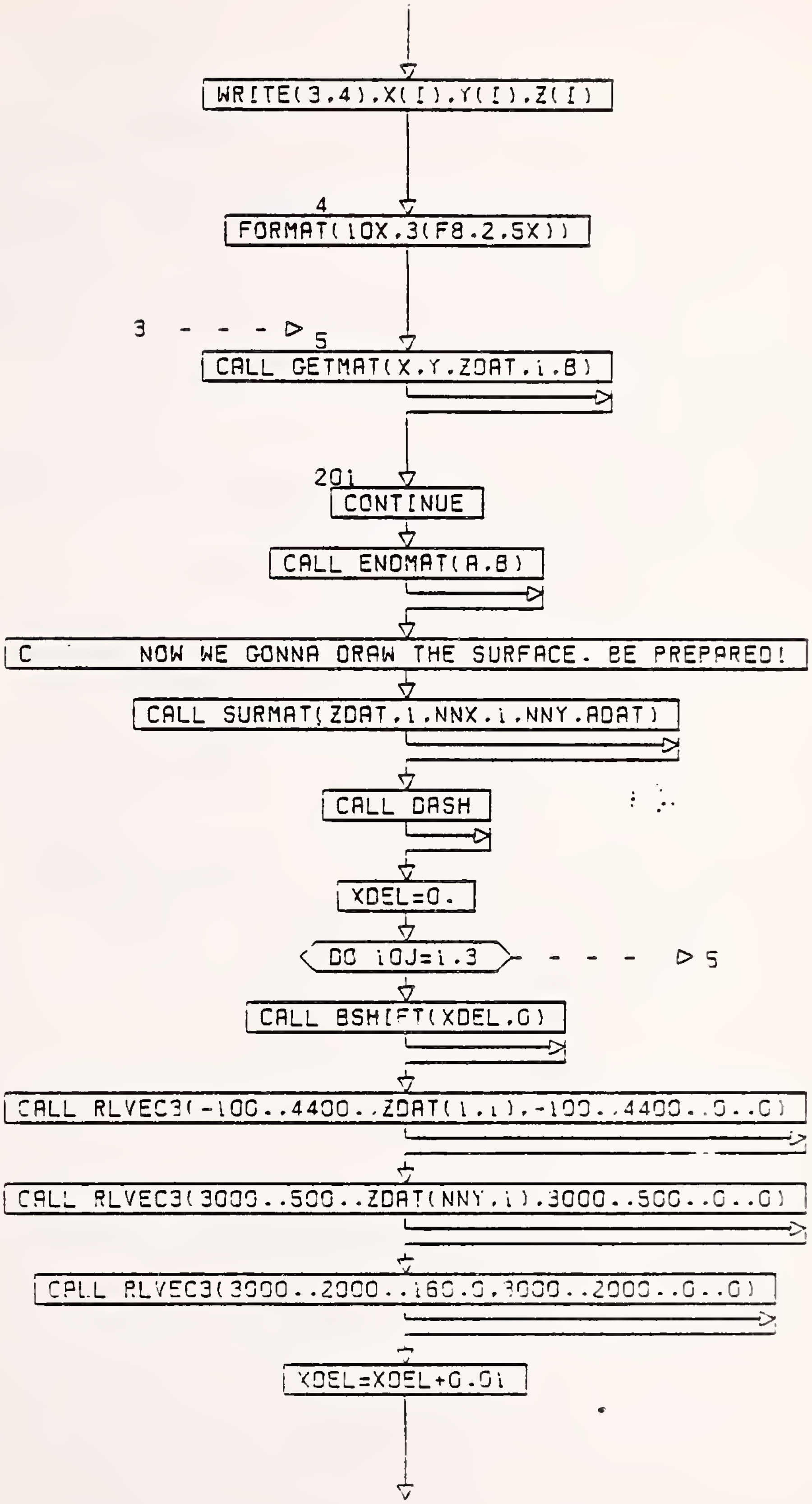

TYPE 40
40
FORMAT(' INPUT NY='.$)
ACCEPT 30, NY
NNX=NX
IF(NX.NE.0)
F
T
GOTO 88
NNX=20
88
NNY=NY
IF(NY.NE.0)
F
T
GOTO 89
NNY=10
89
CONTINUE
CALL BGNPL(0)

CONT. ON PG 3



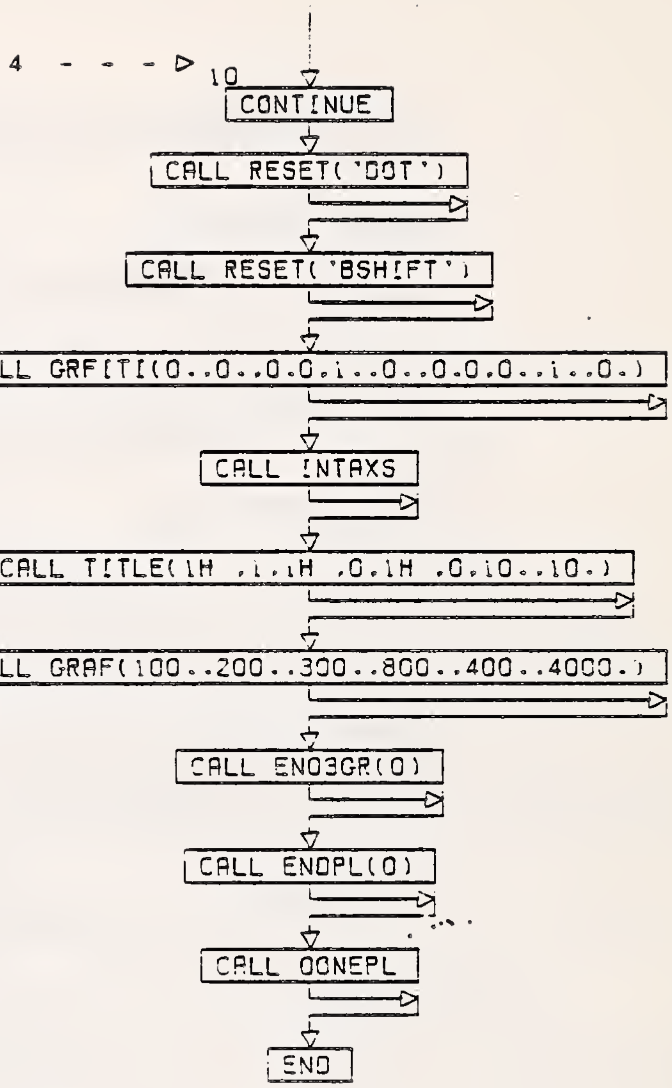
```
CALL INTAXS
CALL ZAXANG(90.)
CALL COMPLX
CALL BASALF('L/CST')
CALL MIXALF('STAND')
C         DEFINE 3D WORKING AREA AND AXES
CALL TITL3D('(ENGINE MAP)$'.-100.9.5.6.5)
CALL AXES3D('RPM$'.100.'TORQUE$'.100.'FUEL RATE$'.100.10..10..8.
C            (          100.10..10..8.)
C         YOU MAY CHANGE THE POINT OF VIEW
CALL VIEW(-200..450..400)
CALL GRAF3D(550..400..3200..-100..40..300..0..10..70.)
C         FILE VAS.DAT  IS GIVEN A LOGICAL #33
OPEN(UNIT=33.FILE='VAS.DAT')
CALL BGNMAT(NNX.NNY)
C         DON'T WORRY ABOUT THE NUMBER OF INPUT DATA TO BE READ.
C         PROGRAM WILL TAKE CARE OF THIS.
DO 5 I=1.54321   - - - > 4
READ(33.4.END=201).X(I).Y(I).Z(I)
C         GO TO THE I/O WINDOW AND GET AN OUTPUT.
C         SEE WETHER YOU SUPPLY THE RIGHT DATA.
```

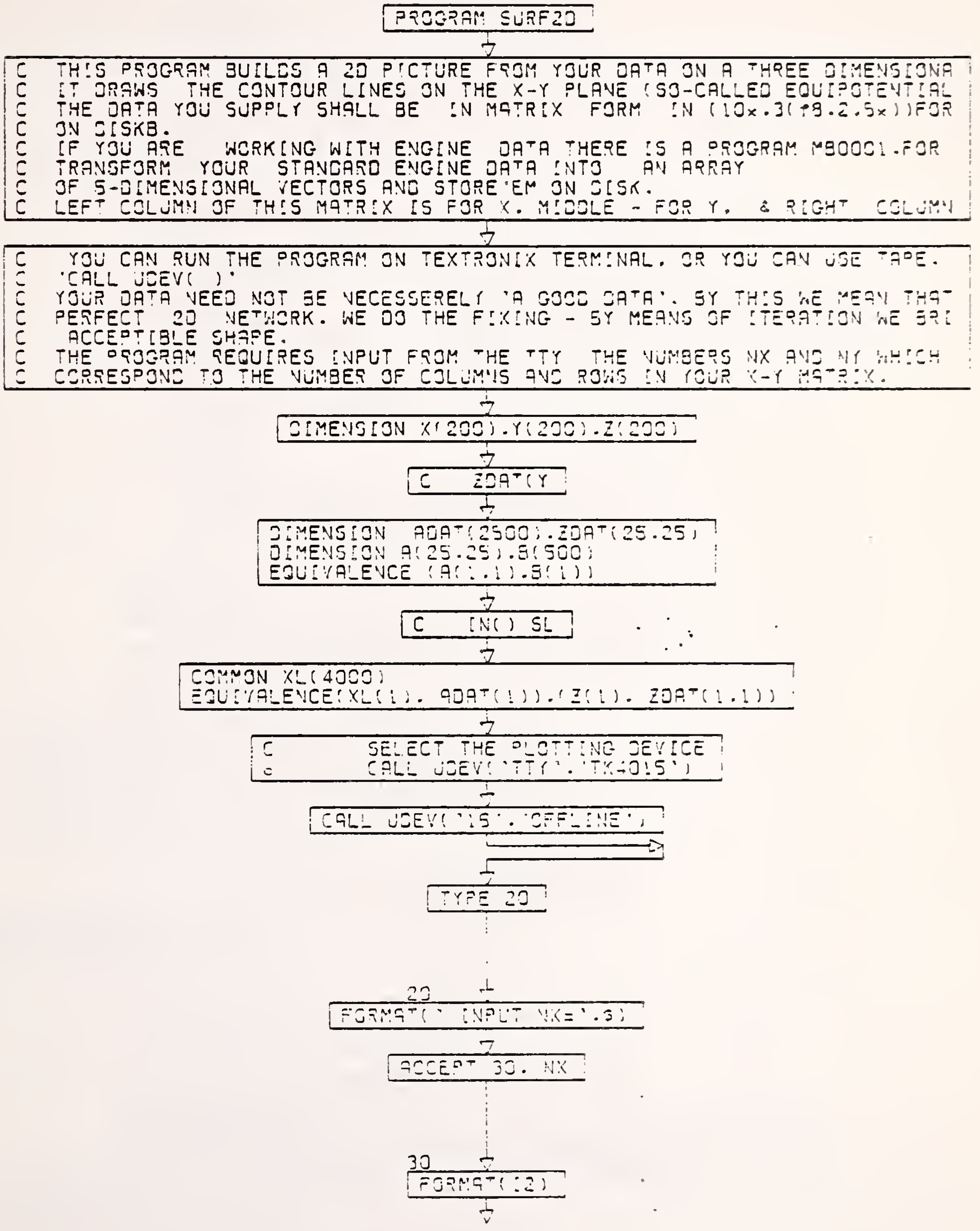
CONT. ON PG 4



```
      WRITE(3.4).X(I).Y(I).Z(I)
4     FORMAT(10X.3(F8.2.5X))
5     CALL GETMAT(X.Y.ZDAT.1.8)
201   CONTINUE
      CALL ENDMAT(A.8)
C          NOW WE GONNA DRAW THE SURFACE. BE PREPARED!
      CALL SURMAT(ZDAT.1.NNX.1.NNY.ADAT)
      CALL DASH
      XDEL=0.
      DO 10J=1.3
      CALL BSHIFT(XDEL.0)
      CALL RLVEC3(-100..4400..ZDAT(1.1).-100..4400..0..0)
      CALL RLVEC3(3000..500..ZDAT(NNY.1).3000..500..0..0)
      CALL RLVEC3(3000..2000..160.0.3000..2000..0..0)
      XDEL=XDEL+0.01
```

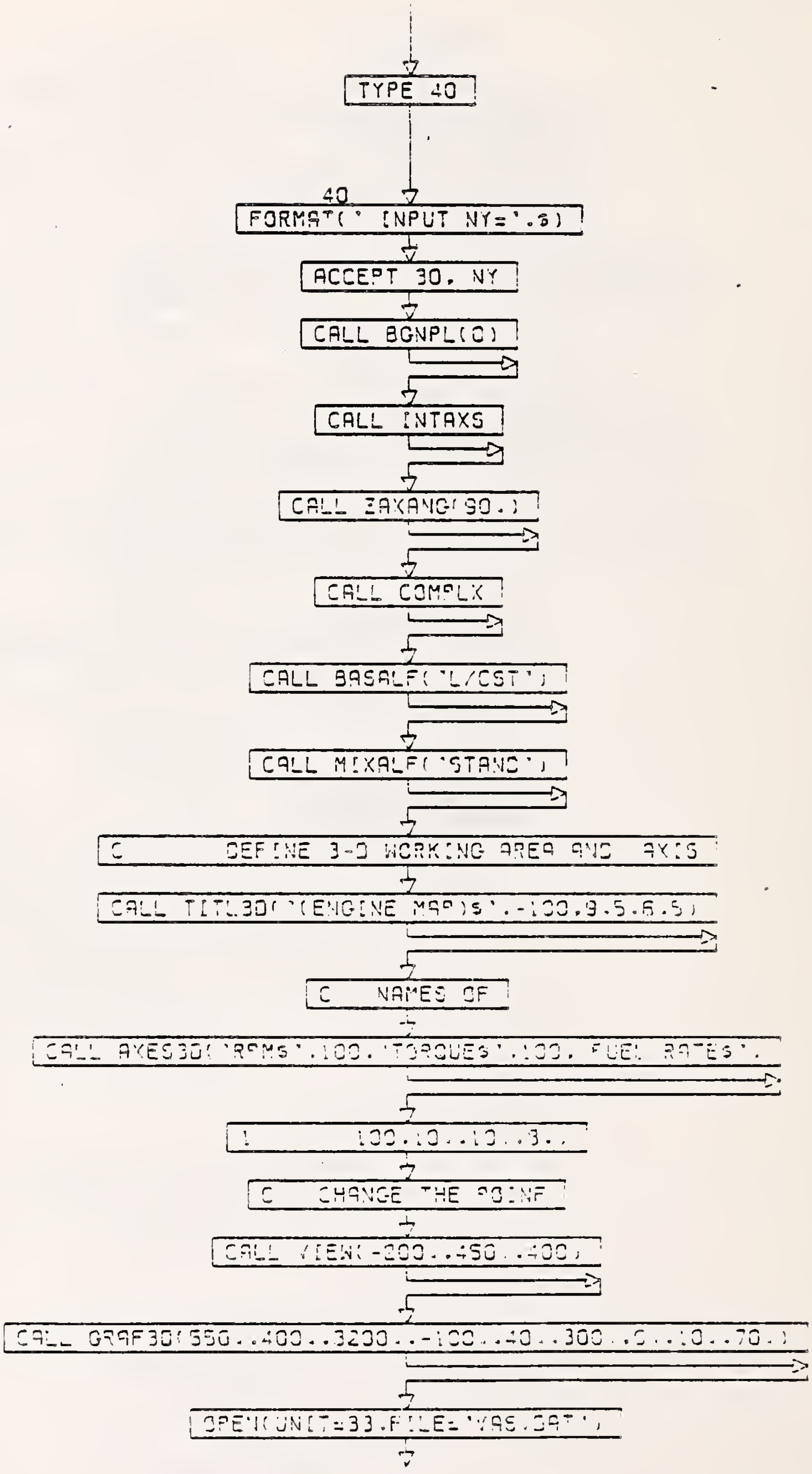
CONT. ON PG 5


4
10
CONTINUE
CALL RESET('DOT')
CALL RESET('BSHIFT')
CALL GRFITI(0.,0.,0.,0.,1.,0.,0.,0.,0.,1.,0.)
CALL INTAXS
CALL TITLE(1H ,1,1H ,0,1H ,0,10.,10.)
CALL GRAF(100.,200.,300.,800.,400.,4000.)
CALL END3GR(0)
CALL ENDPL(0)
CALL DONEPL
END

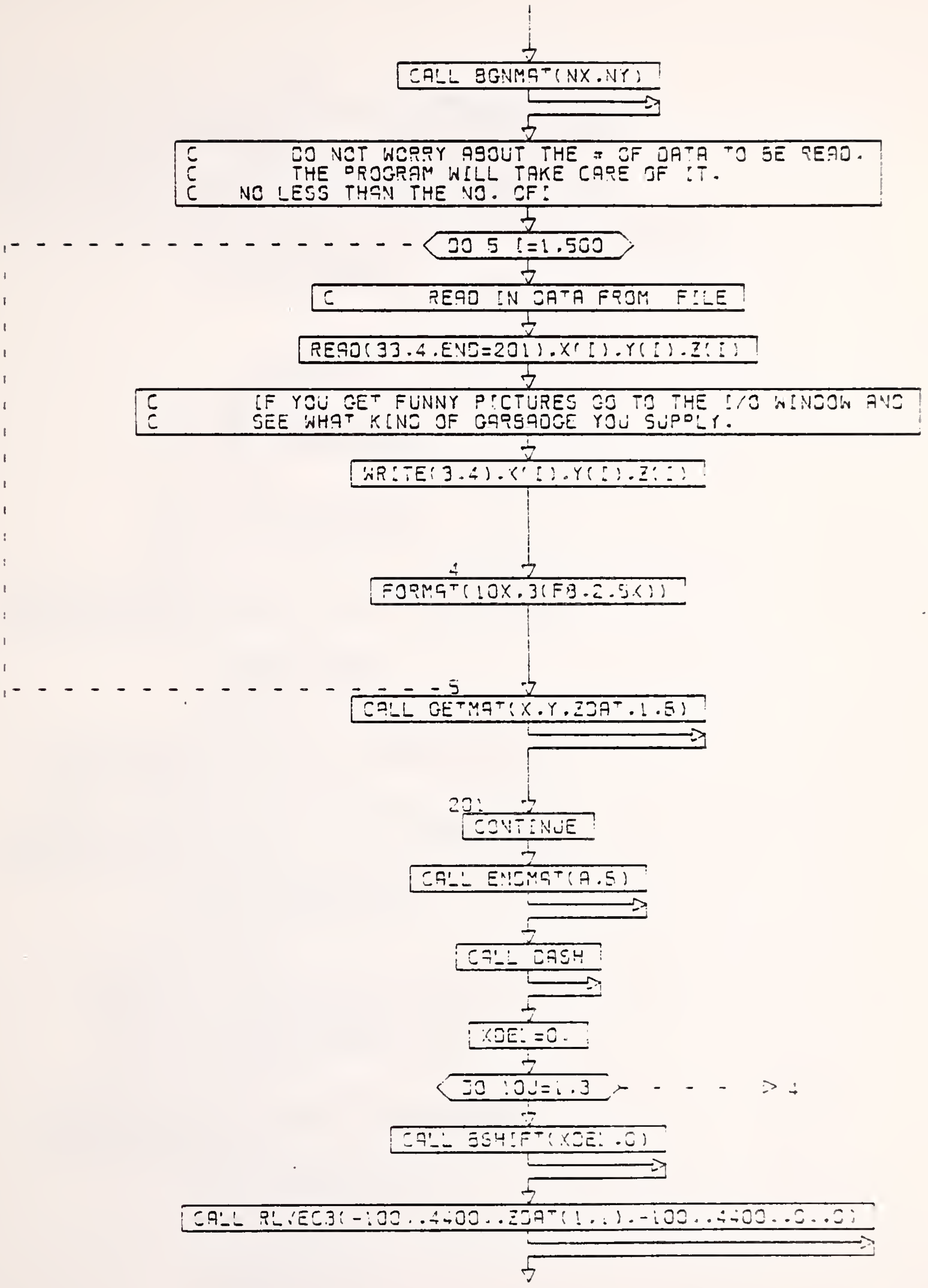


```
PROGRAM SURF2D

C  THIS PROGRAM BUILDS A 2D PICTURE FROM YOUR DATA ON A THREE DIMENSIONA
C  IT DRAWS  THE CONTOUR LINES ON THE X-Y PLANE (SO-CALLED EQUIPOTENTIAL
C  THE DATA YOU SUPPLY SHALL BE  IN MATRIX  FORM  IN (10X.3(F8.2.5X))FOR
C  ON DISKB.
C  IF YOU ARE   WORKING WITH ENGINE  DATA THERE IS A PROGRAM M8OOC1.FOR
C  TRANSFORM  YOUR  STANDARD ENGINE DATA INTO   AN ARRAY
C  OF 5-DIMENSIONAL VECTORS AND STORE'EM ON DISK.
C  LEFT COLUMN OF THIS MATRIX IS FOR X. MIDDLE - FOR Y.  & RIGHT  COLUMN

C   YOU CAN RUN THE PROGRAM ON TEXTRONIX TERMINAL. OR YOU CAN USE TAPE.
C  'CALL UDEV(  )'
C  YOUR DATA NEED NOT BE NECESSERELY 'A GOOD DATA'. BY THIS WE MEAN THAT
C  PERFECT  2D  NETWORK. WE DO THE FIXING - BY MEANS OF ITERATION WE BRI
C   ACCEPTIBLE SHAPE.
C  THE PROGRAM REQUIRES INPUT FROM THE TTY  THE NUMBERS NX AND NY WHICH
C  CORRESPOND TO THE NUMBER OF COLUMNS AND ROWS IN YOUR X-Y MATRIX.

DIMENSION X(200).Y(200).Z(200)

C   ZDAT(Y

DIMENSION  ADAT(2500).ZDAT(25.25)
DIMENSION A(25.25).B(500)
EQUIVALENCE (A(1.1).B(1))

C   IN() SL

COMMON XL(4000)
EQUIVALENCE(XL(1). ADAT(1)).(Z(1). ZDAT(1.1))

C        SELECT THE PLOTTING DEVICE
C        CALL UDEV('TTY'.'TK4015')

CALL UDEV('1S'.'OFFLINE')

TYPE 20

20
FORMAT(' INPUT NX='.$)

ACCEPT 30. NX

30
FORMAT(I2)
```

CONT. ON PG 2

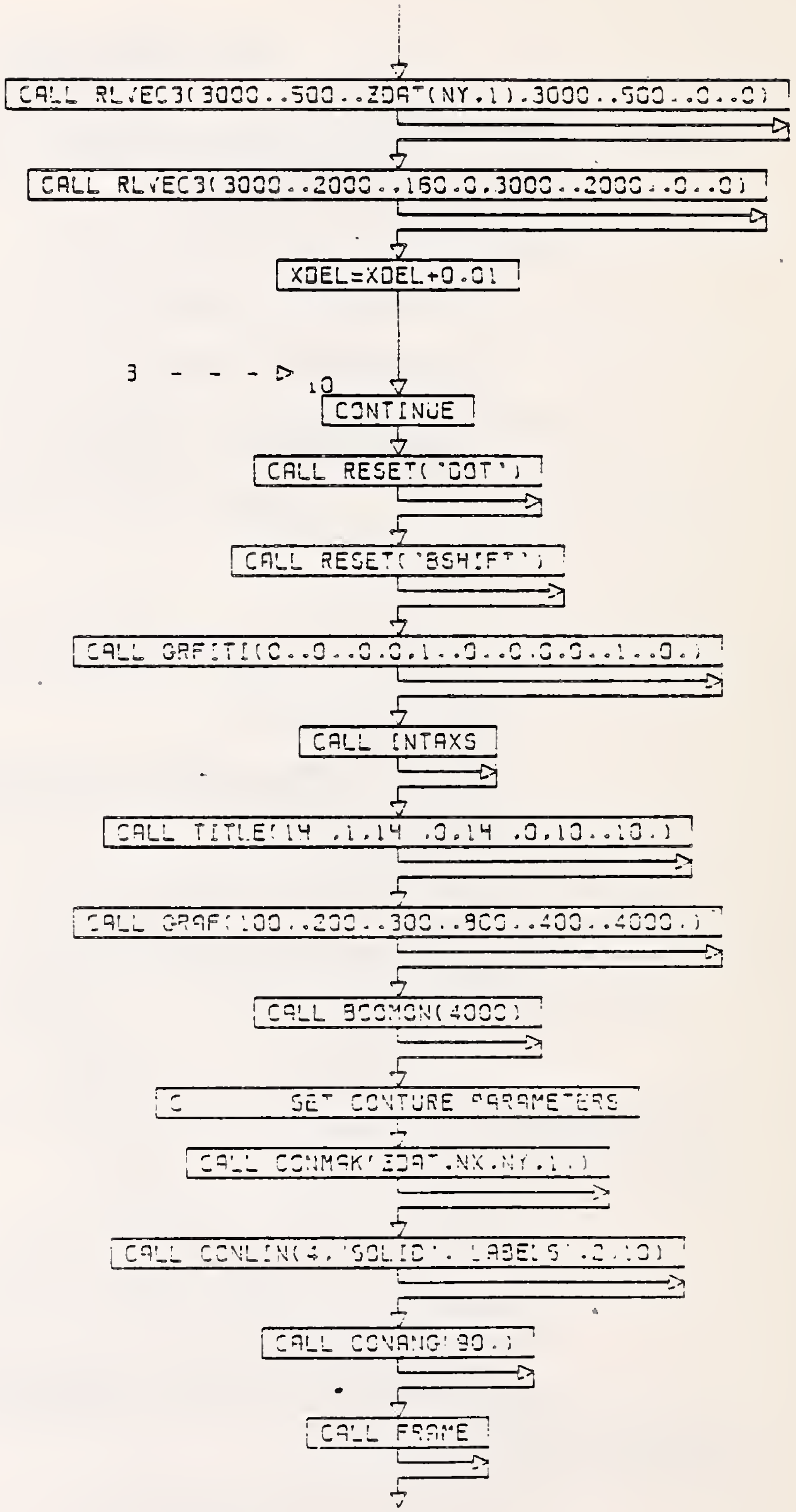

TYPE 40
40
FORMAT(' INPUT NY='.$)
ACCEPT 30, NY
CALL BGNPL(0)
CALL INTAXS
CALL ZAXANG(90.)
CALL COMPLX
CALL BASALF('L/CST')
CALL MIXALF('STAND')
C DEFINE 3-D WORKING AREA AND AXIS
CALL TITL3D('(ENGINE MAP)$'.-100.9.5.6.5)
C NAMES OF
CALL AXES3D('RPMs'.100.'TORQUEs'.100. FUEL RATEs'.
1 100.10..10..8..
C CHANGE THE POINT
CALL VIEW(-200..450..400)
CALL GRAF3D(550..400..3200..-100..40..300..0..10..70.)
OPEN(UNIT=33.FILE='VAS.DAT')

CONT. ON PG 3



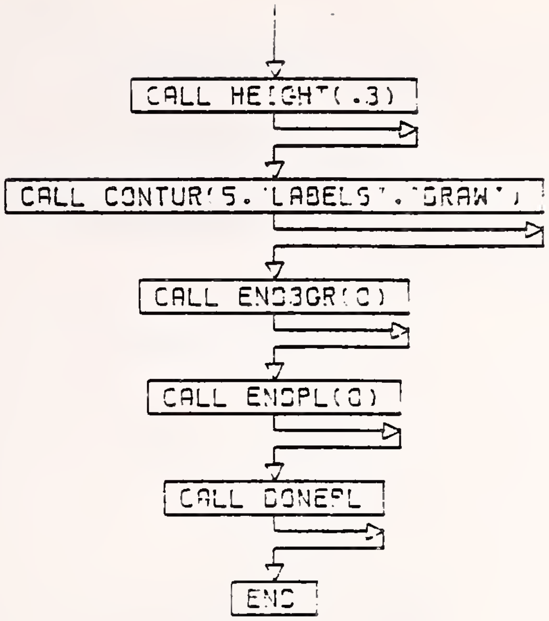
```
CALL BGNMAT(NX.NY)
C        DO NOT WORRY ABOUT THE # OF DATA TO BE READ.
C        THE PROGRAM WILL TAKE CARE OF IT.
C    NO LESS THAN THE NO. OF:
DO 5 I=1.500
C        READ IN DATA FROM  FILE
READ(33.4.END=201).X(I).Y(I).Z(I)
C        IF YOU GET FUNNY PICTURES GO TO THE I/O WINDOW AND
C        SEE WHAT KIND OF GARBADGE YOU SUPPLY.
WRITE(3.4).X(I).Y(I).Z(I)
4 FORMAT(10X.3(F8.2.5X))
5 CALL GETMAT(X.Y.ZDAT.1.5)
201 CONTINUE
CALL ENDMAT(A.5)
CALL DASH
XDEL=0.
DO 10J=1.3
CALL BSHIFT(XDEL.0)
CALL RLVEC3(-100..4400..ZDAT(1.I).-100..4400..0..0)
```

CONT. ON PG 4

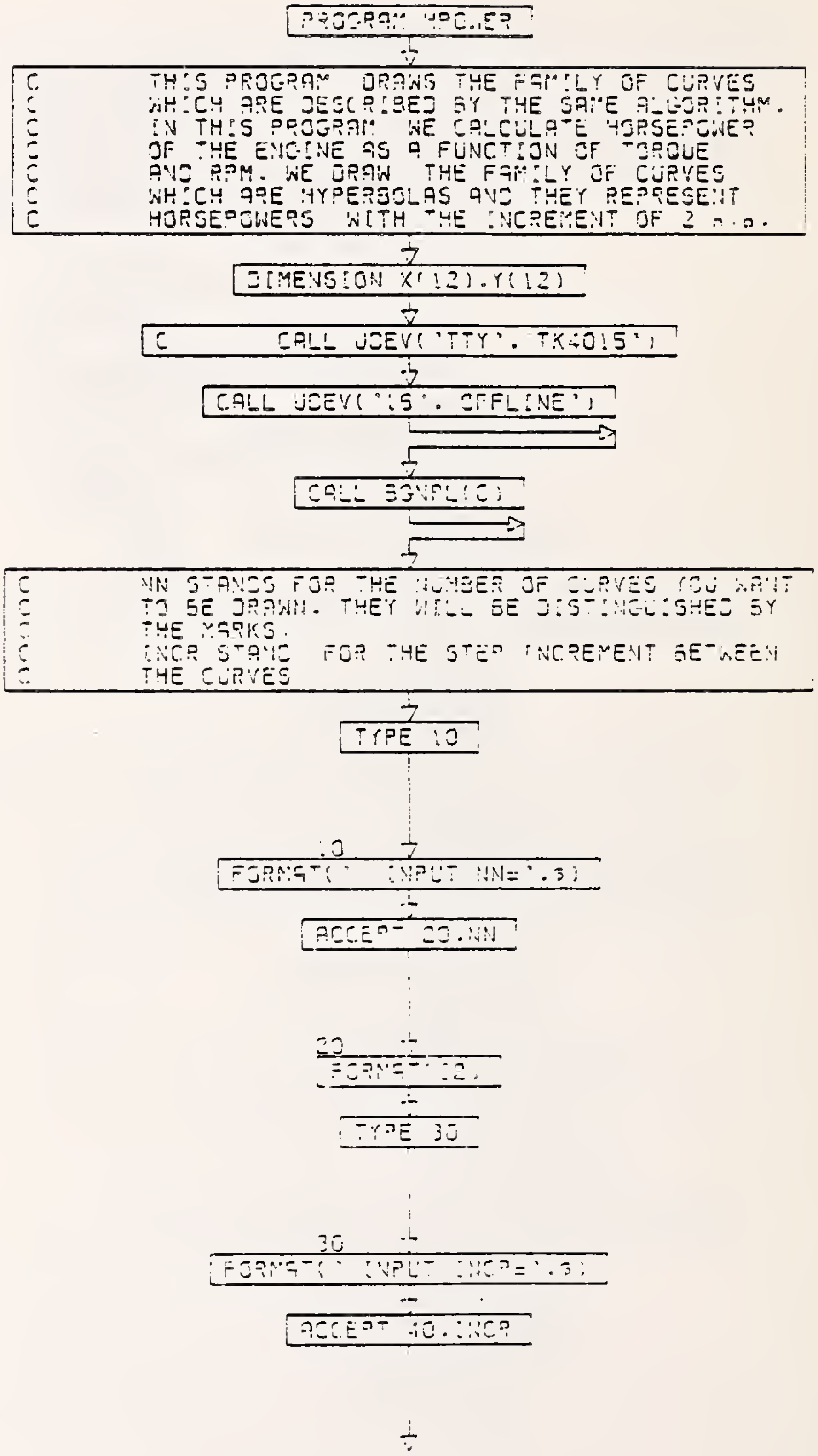


```
CALL RLVEC3(3000..500..ZDAT(NY.1).3000..500..0..0)
CALL RLVEC3(3000..2000..160.0.3000..2000..0..0)
XDEL=XDEL+0.01
3 - - - > 10
CONTINUE
CALL RESET('DOT')
CALL RESET('BSHIFT')
CALL GRFITI(0..0..0.0.1..0..0.0.0..1..0.)
CALL INTAXS
CALL TITLE(14 .1.14 .0.14 .0.10..10.)
CALL GRAF(100..200..300..900..400..4000.)
CALL BGCMON(4000)
C          SET CONTURE PARAMETERS
CALL CONMAK(ZDAT.NX.NY.1.)
CALL CONLIN(4.'SOLID'.'LABELS'.2.10)
CALL CONANG(90.)
CALL FRAME
```

CONT. ON PG 5

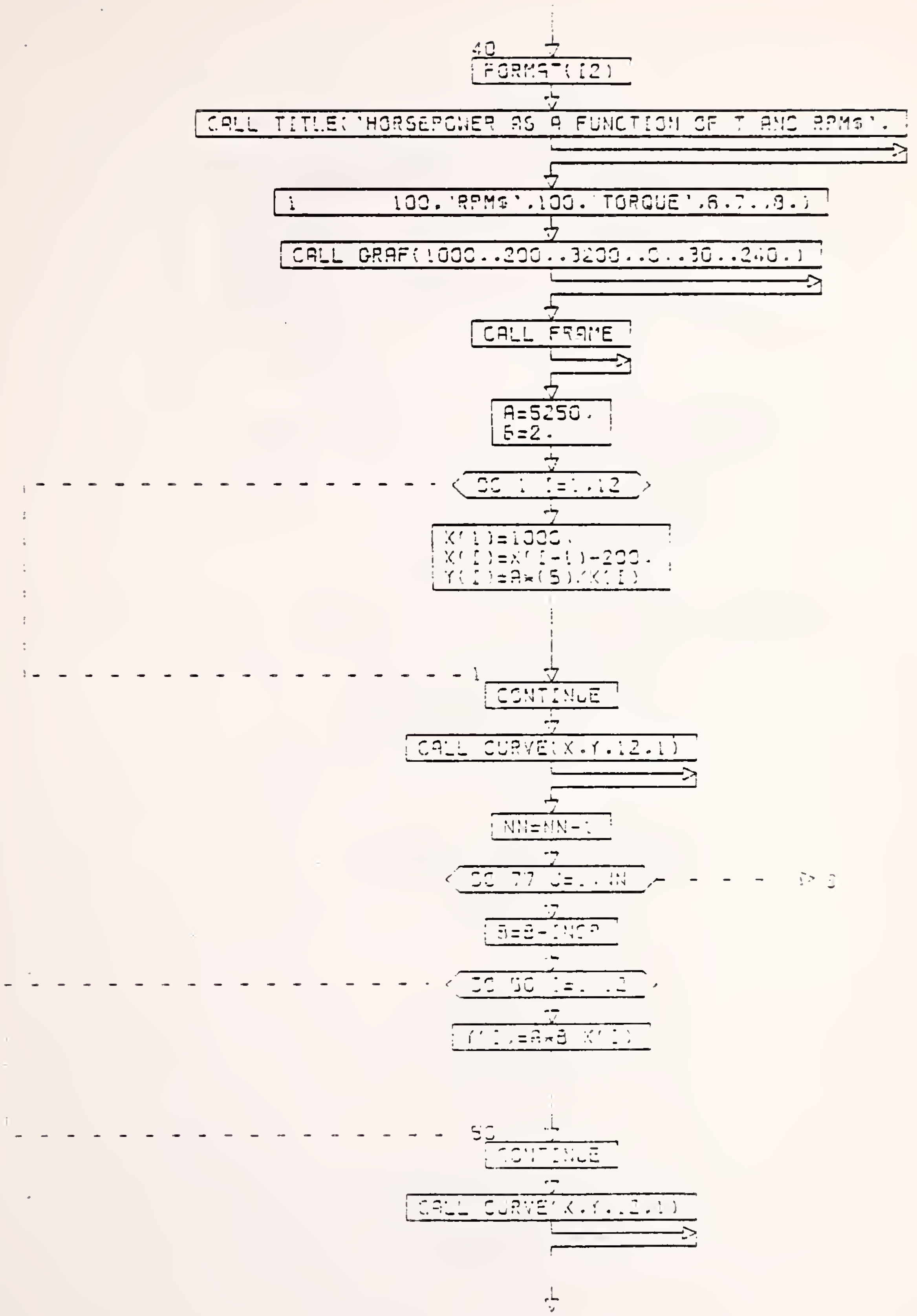

CALL HEIGHT(.3)
CALL CONTUR(5,'LABELS','DRAW')
CALL END3GR(0)
CALL ENDPL(0)
CALL DONEPL
END

PG 5 [illegible]

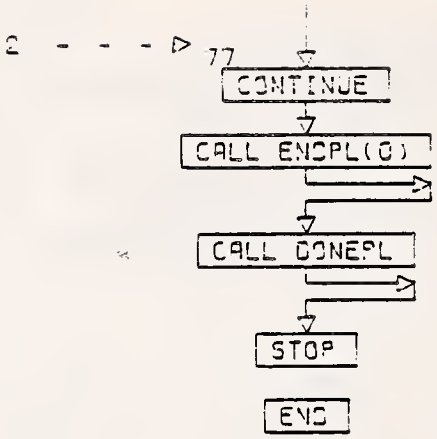


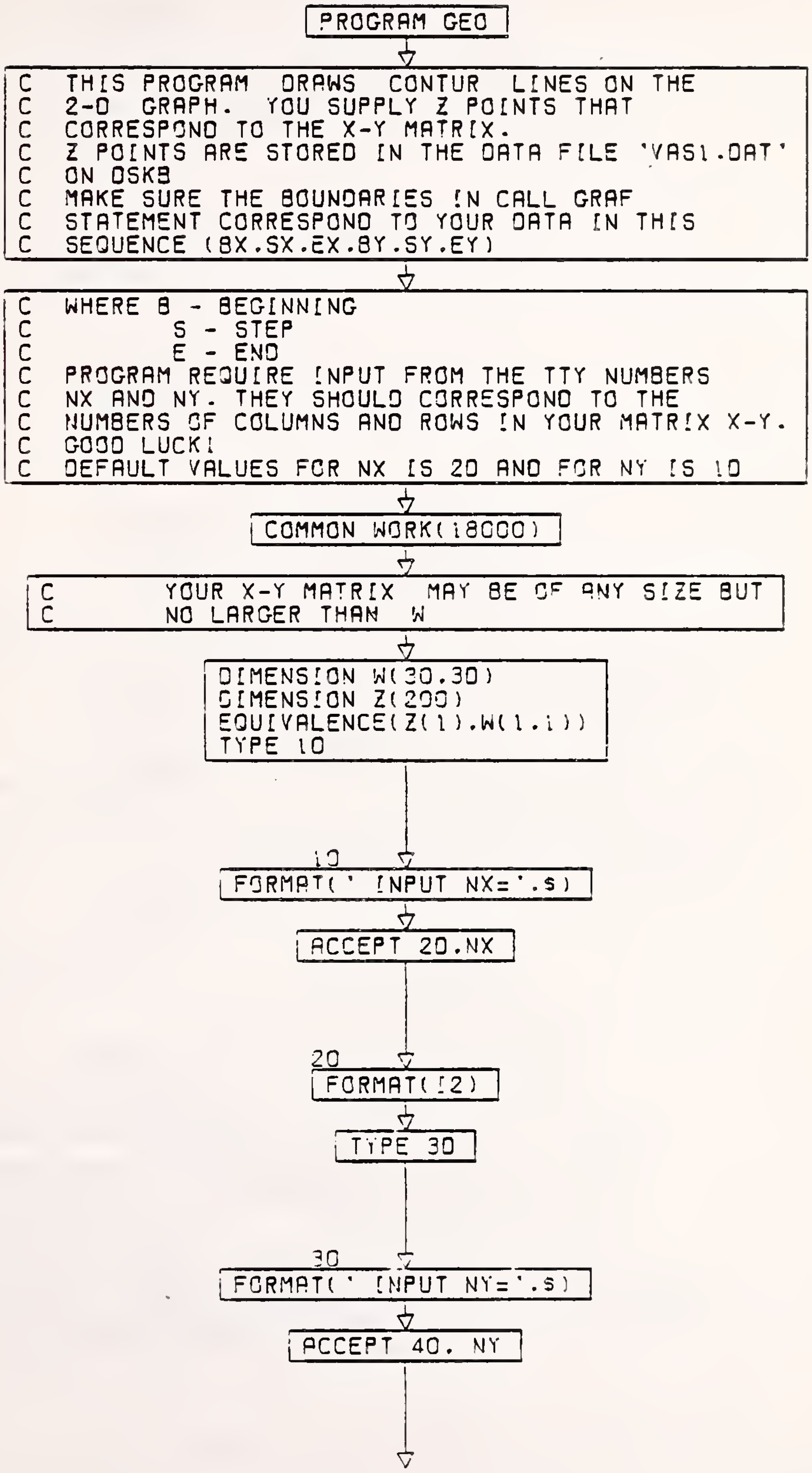
```
PROGRAM HPOWER
C         THIS PROGRAM  DRAWS THE FAMILY OF CURVES
C         WHICH ARE DESCRIBED BY THE SAME ALGORITHM.
C         IN THIS PROGRAM  WE CALCULATE HORSEPOWER
C         OF THE ENGINE AS A FUNCTION OF TORQUE
C         AND RPM. WE DRAW  THE FAMILY OF CURVES
C         WHICH ARE HYPERBOLAS AND THEY REPRESENT
C         HORSEPOWERS  WITH THE INCREMENT OF 2 h.p.
DIMENSION X(12).Y(12)
C         CALL UDEV('TTY'. 'TK4015')
CALL UDEV('LS'. 'OFFLINE')
CALL BGNPL(0)
C         NN STANDS FOR THE NUMBER OF CURVES YOU WANT
C         TO BE DRAWN. THEY WILL BE DISTINGUISHED BY
C         THE MARKS.
C         INCR STAND  FOR THE STEP INCREMENT BETWEEN
C         THE CURVES
TYPE 10
10 FORMAT('  INPUT NN='.$)
ACCEPT 20.NN
20 FORMAT(I2)
TYPE 30
30 FORMAT(' INPUT INCR='.$)
ACCEPT 40.INCR
```

CONT. ON PG 2


40
FORMAT(I2)
CALL TITLE('HORSEPOWER AS A FUNCTION OF T AND RPM$',
1 100.'RPM$'.100.'TORQUE'.6.7..8.)
CALL GRAF(1000..200..3200..0..30..240.)
CALL FRAME
A=5250.
B=2.
DO 1 I=1.12
X(1)=1000.
X(I)=X(I-1)+200.
Y(I)=A*(B)/X(I)
1
CONTINUE
CALL CURVE(X.Y.12.1)
NN=NN-1
DO 77 J=1.NN
P 3
B=B-INCR
DO 50 I=1.12
Y(I)=A*B/X(I)
50
CONTINUE
CALL CURVE(X.Y.12.1)

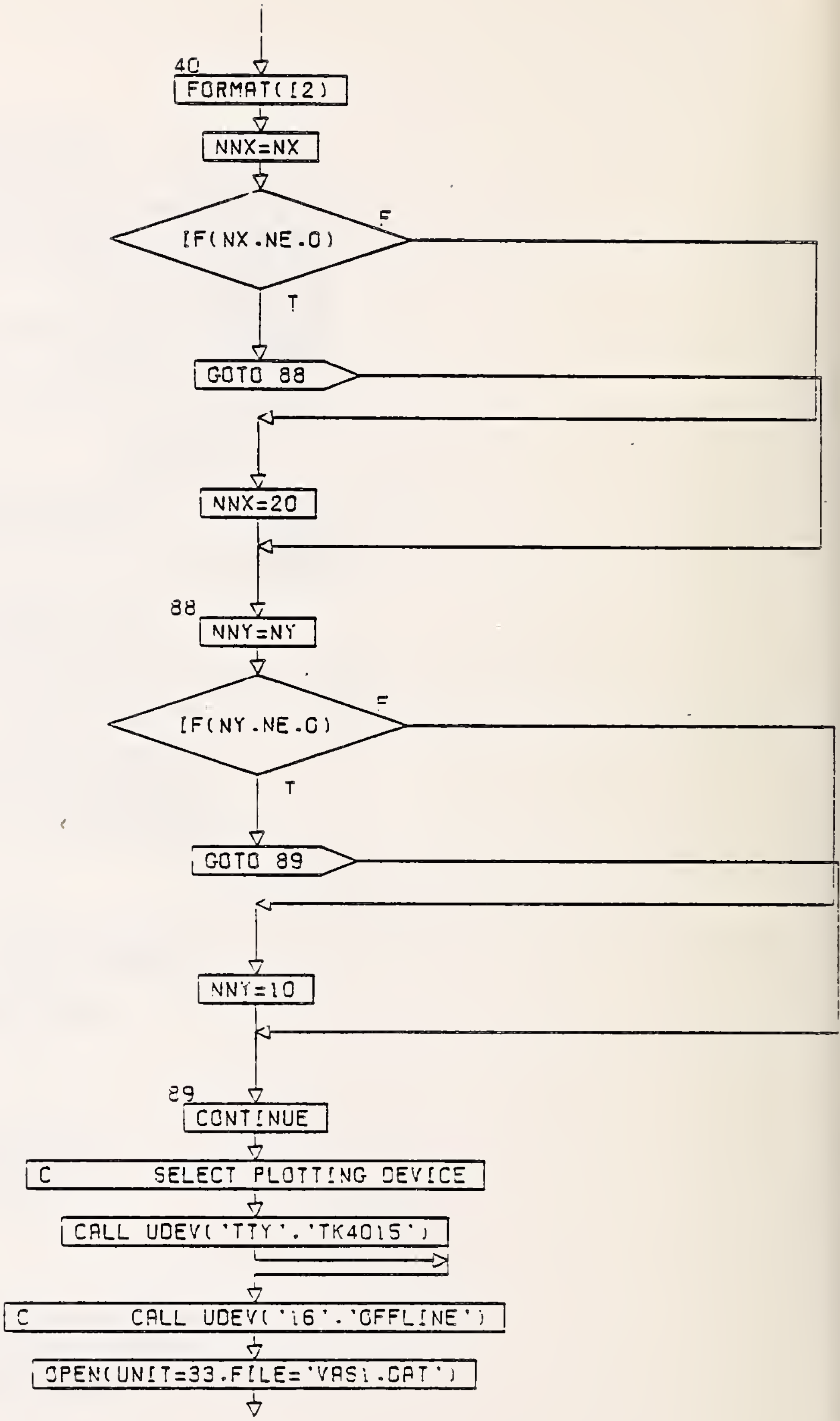
CONT. ON PG 3


2
77
CONTINUE
CALL ENDPL(0)
CALL DONEPL
STOP
END

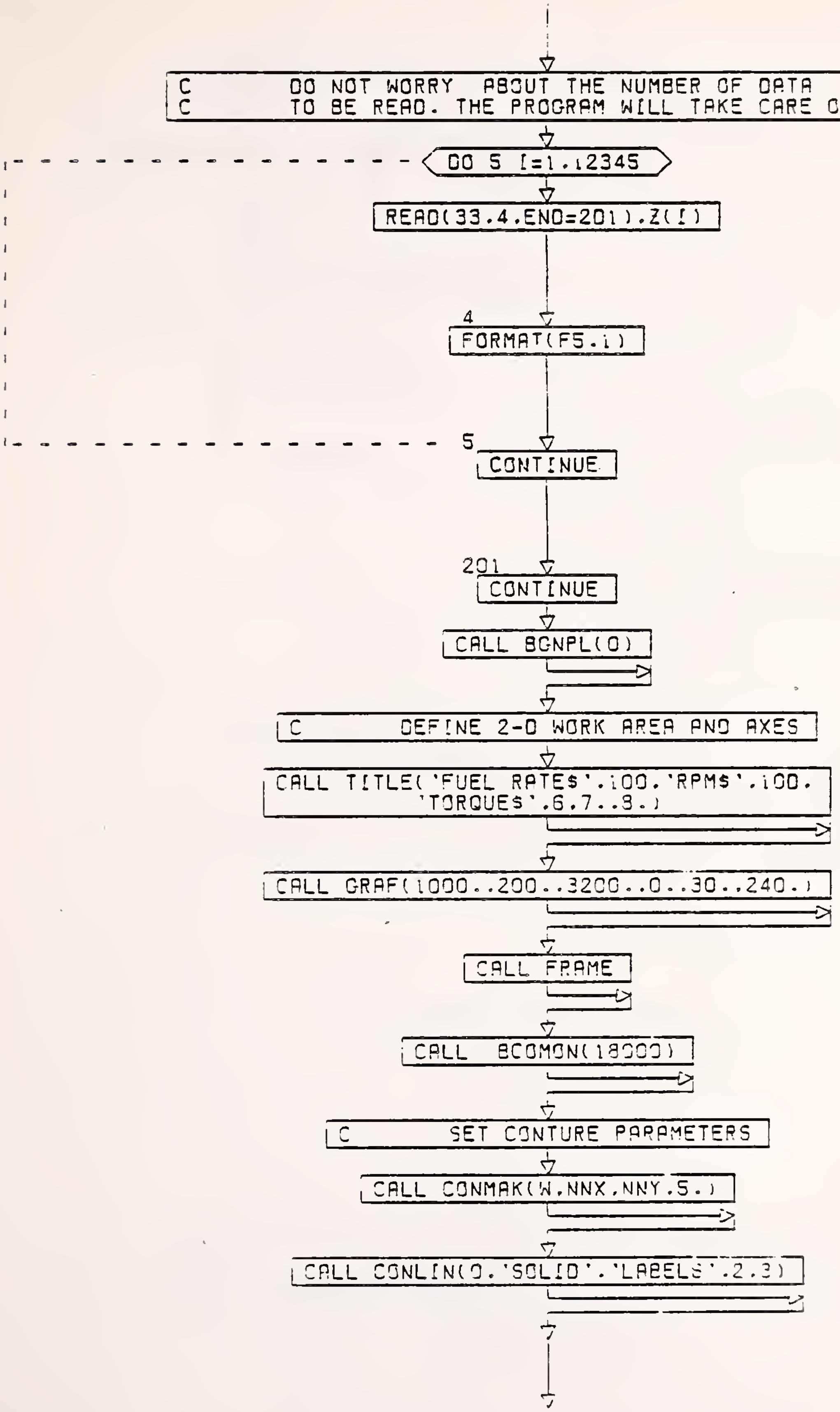


```
PROGRAM GEO

C   THIS PROGRAM  DRAWS  CONTUR  LINES ON THE
C   2-D  GRAPH.  YOU SUPPLY Z POINTS THAT
C   CORRESPOND TO THE X-Y MATRIX.
C   Z POINTS ARE STORED IN THE DATA FILE 'VAS1.DAT'
C   ON DSKB
C   MAKE SURE THE BOUNDARIES IN CALL GRAF
C   STATEMENT CORRESPOND TO YOUR DATA IN THIS
C   SEQUENCE (BX.SX.EX.BY.SY.EY)

C   WHERE B - BEGINNING
C             S - STEP
C             E - END
C   PROGRAM REQUIRE INPUT FROM THE TTY NUMBERS
C   NX AND NY. THEY SHOULD CORRESPOND TO THE
C   NUMBERS OF COLUMNS AND ROWS IN YOUR MATRIX X-Y.
C   GOOD LUCK!
C   DEFAULT VALUES FOR NX IS 20 AND FOR NY IS 10

COMMON WORK(18000)

C            YOUR X-Y MATRIX  MAY BE OF ANY SIZE BUT
C            NO LARGER THAN  W

DIMENSION W(30.30)
DIMENSION Z(200)
EQUIVALENCE(Z(1).W(1.1))
TYPE 10

10  FORMAT(' INPUT NX='.$)

ACCEPT 20.NX

20  FORMAT(I2)

TYPE 30

30  FORMAT(' INPUT NY='.$)

ACCEPT 40. NY
```

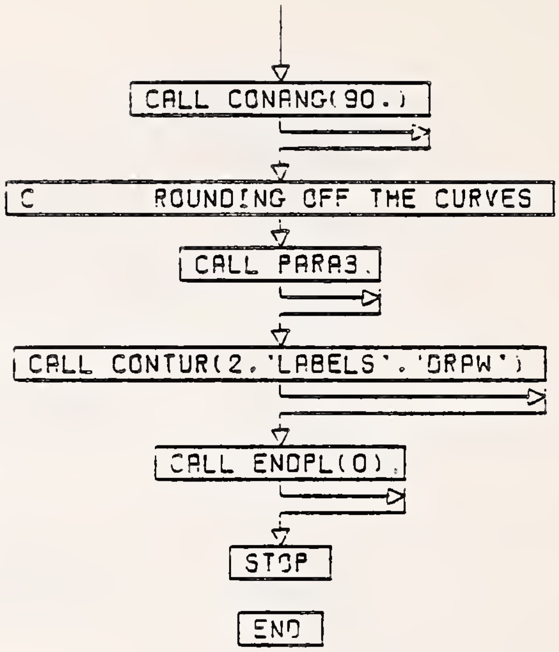
CONT. ON PG 2

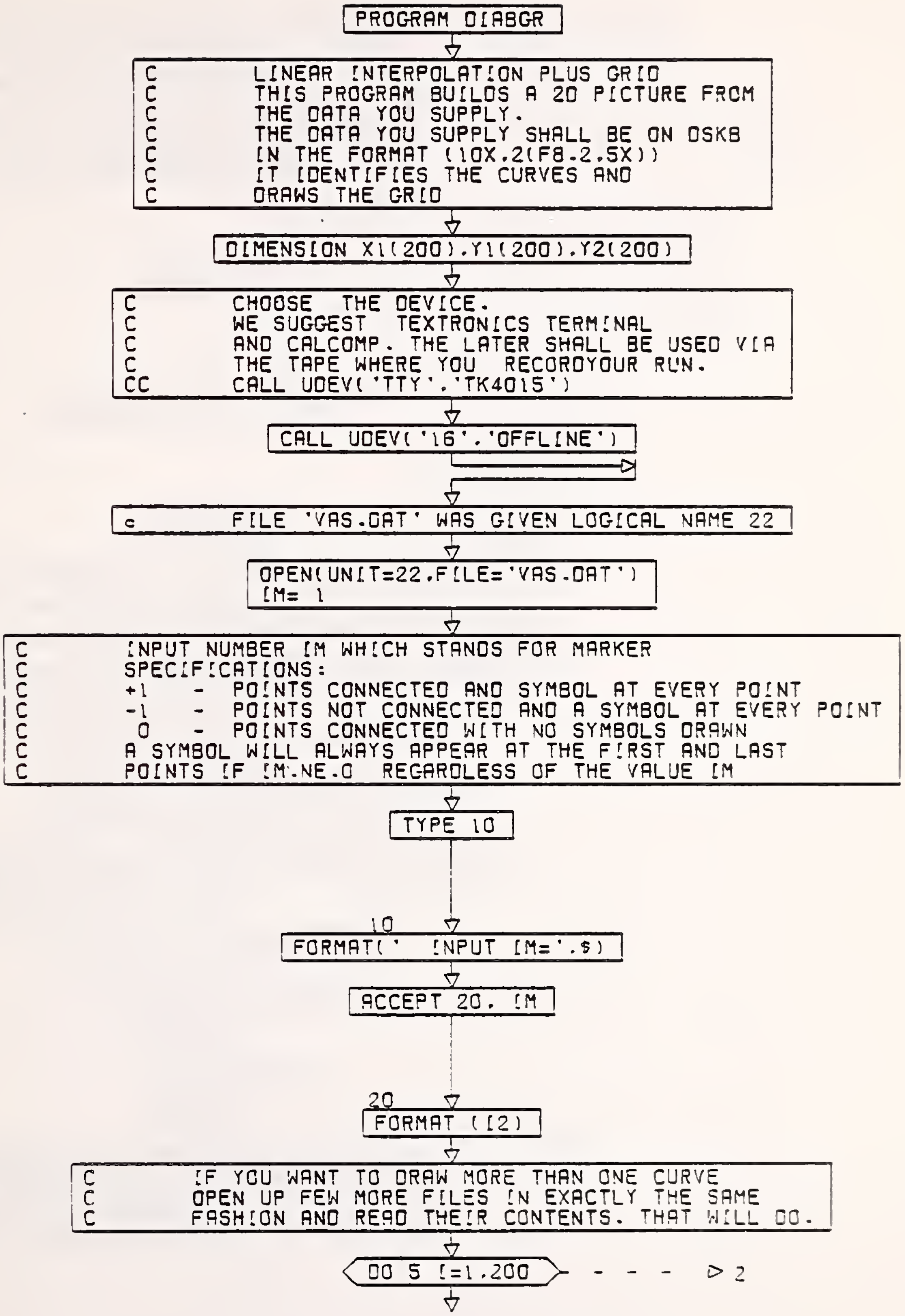

40
FORMAT(I2)
NNX=NX
IF(NX.NE.0)
F
T
GOTO 88
NNX=20
88
NNY=NY
IF(NY.NE.0)
F
T
GOTO 89
NNY=10
89
CONTINUE
C SELECT PLOTTING DEVICE
CALL UDEV('TTY','TK4015')
C CALL UDEV('16','OFFLINE')
OPEN(UNIT=33,FILE='VAS1.DAT')

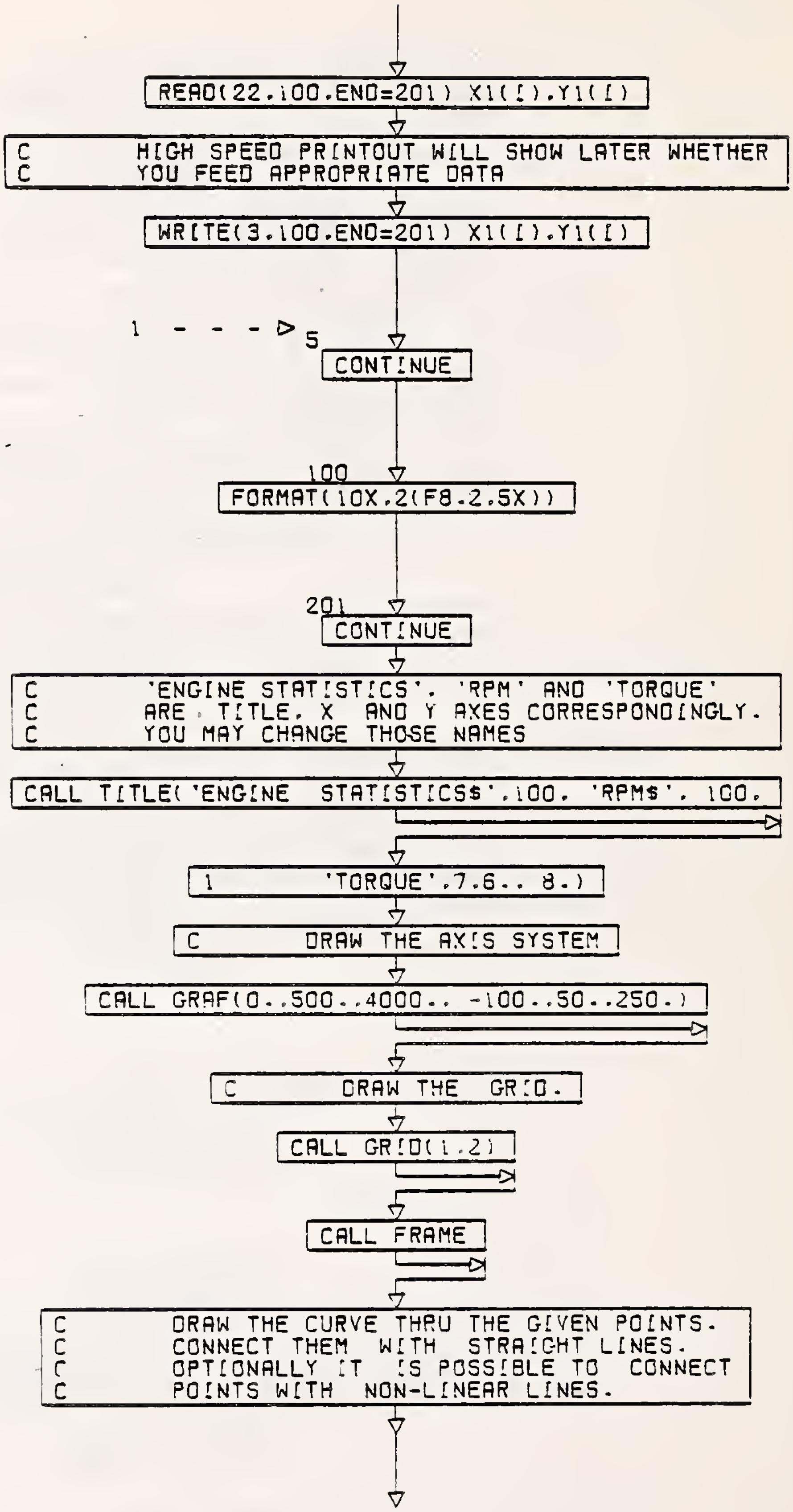
CONT. ON PG 3


C DO NOT WORRY ABOUT THE NUMBER OF DATA
C TO BE READ. THE PROGRAM WILL TAKE CARE OF IT.
DO 5 I=1.12345
READ(33.4.END=201).Z(I)
4
FORMAT(F5.1)
5
CONTINUE
201
CONTINUE
CALL BGNPL(0)
C DEFINE 2-D WORK AREA AND AXES
CALL TITLE('FUEL RATES'.100.'RPMS'.100.
'TORQUES'.6.7..8.)
CALL GRAF(1000..200..3200..0..30..240.)
CALL FRAME
CALL BCOMON(18000)
C SET CONTURE PARAMETERS
CALL CONMAK(W.NNX.NNY.5.)
CALL CONLIN(0.'SOLID'.'LABELS'.2.3)

CONT. ON PG 4

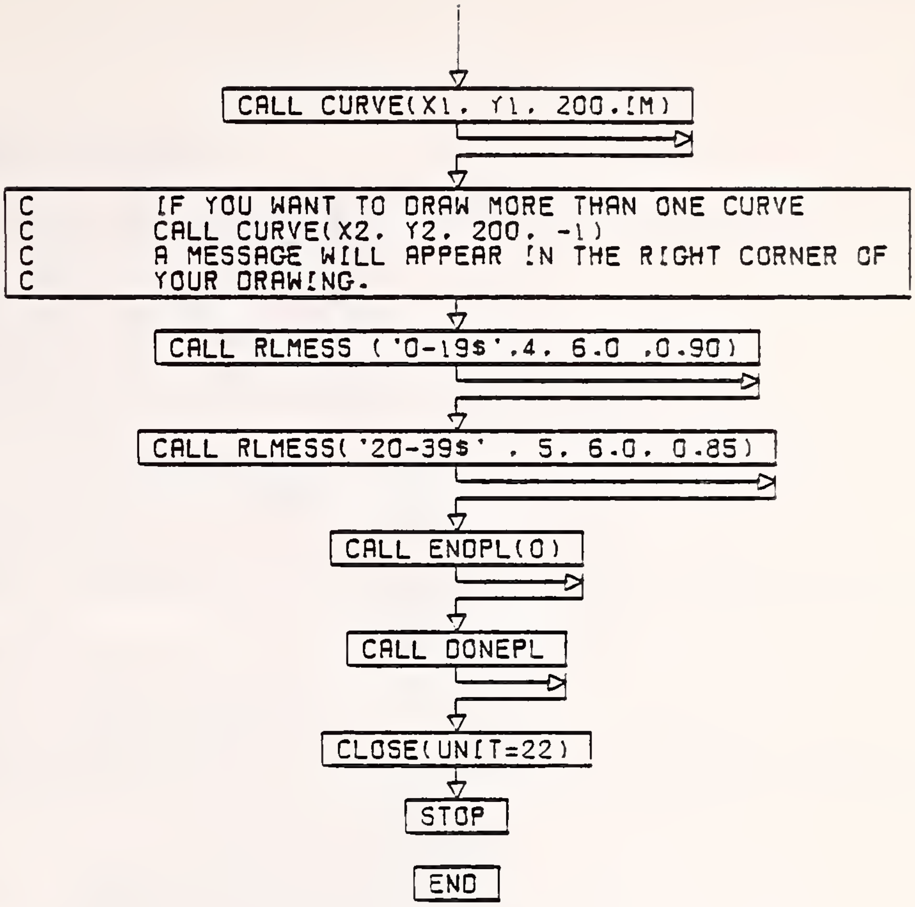

CALL CONANG(90.)
C ROUNDING OFF THE CURVES
CALL PARA3.
CALL CONTUR(2.'LABELS'.'DRAW')
CALL ENDPL(0).
STOP
END



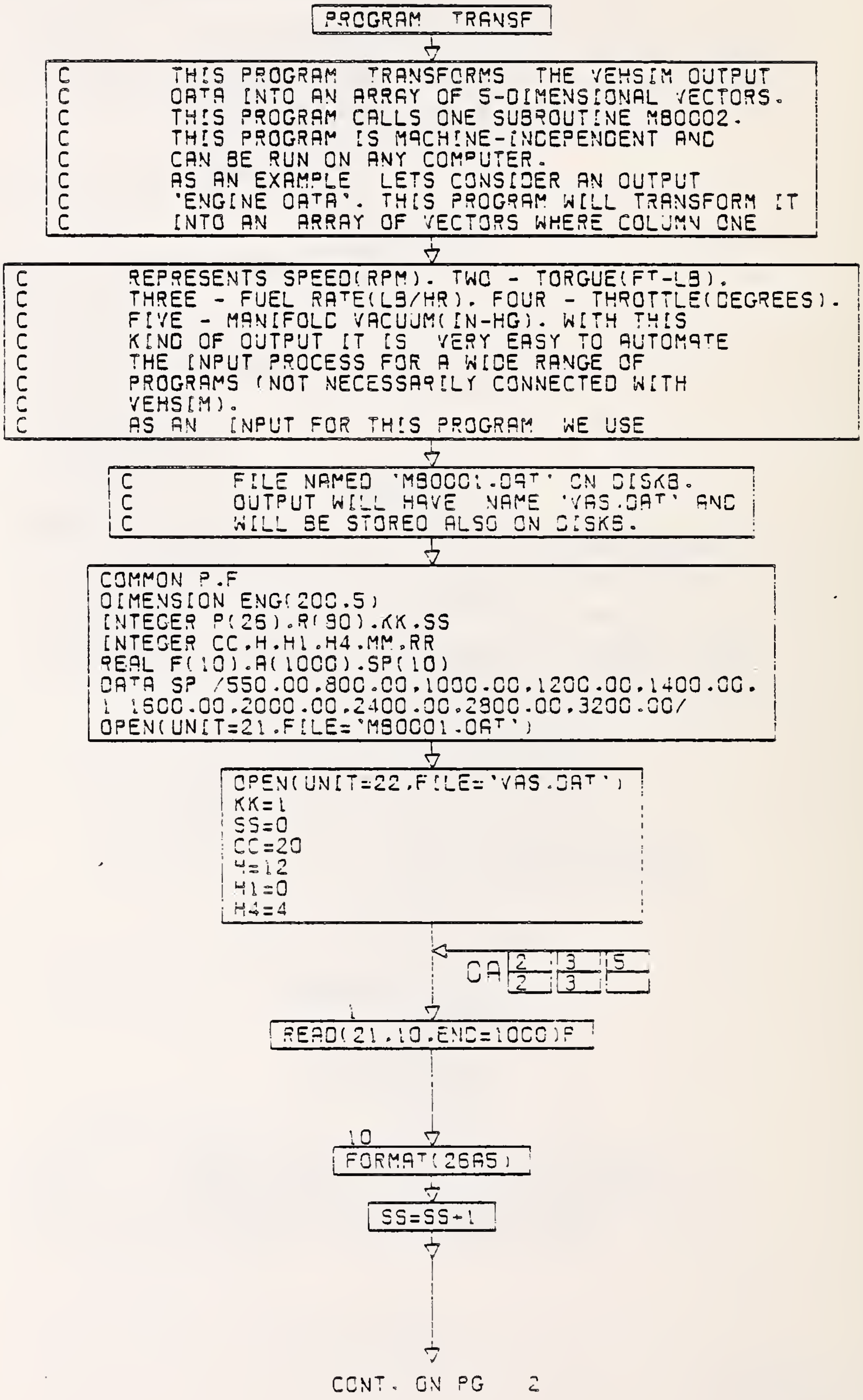


```
      READ(22,100,END=201) X1(I),Y1(I)
C         HIGH SPEED PRINTOUT WILL SHOW LATER WHETHER
C         YOU FEED APPROPRIATE DATA
      WRITE(3,100,END=201) X1(I),Y1(I)
1 - - - ▷ 5
5     CONTINUE
100   FORMAT(10X,2(F8.2,5X))
201   CONTINUE
C         'ENGINE STATISTICS'. 'RPM' AND 'TORQUE'
C         ARE TITLE, X  AND Y AXES CORRESPONDINGLY.
C         YOU MAY CHANGE THOSE NAMES
      CALL TITLE('ENGINE  STATISTICS$',100, 'RPM$', 100,
     1          'TORQUE',7.6., 8.)
C         DRAW THE AXIS SYSTEM
      CALL GRAF(0.,500.,4000., -100.,50.,250.)
C         DRAW THE GRID.
      CALL GRID(1,2)
      CALL FRAME
C         DRAW THE CURVE THRU THE GIVEN POINTS.
C         CONNECT THEM WITH  STRAIGHT LINES.
C         OPTIONALLY IT  IS POSSIBLE TO  CONNECT
C         POINTS WITH  NON-LINEAR LINES.
```

CONT. ON PG 3

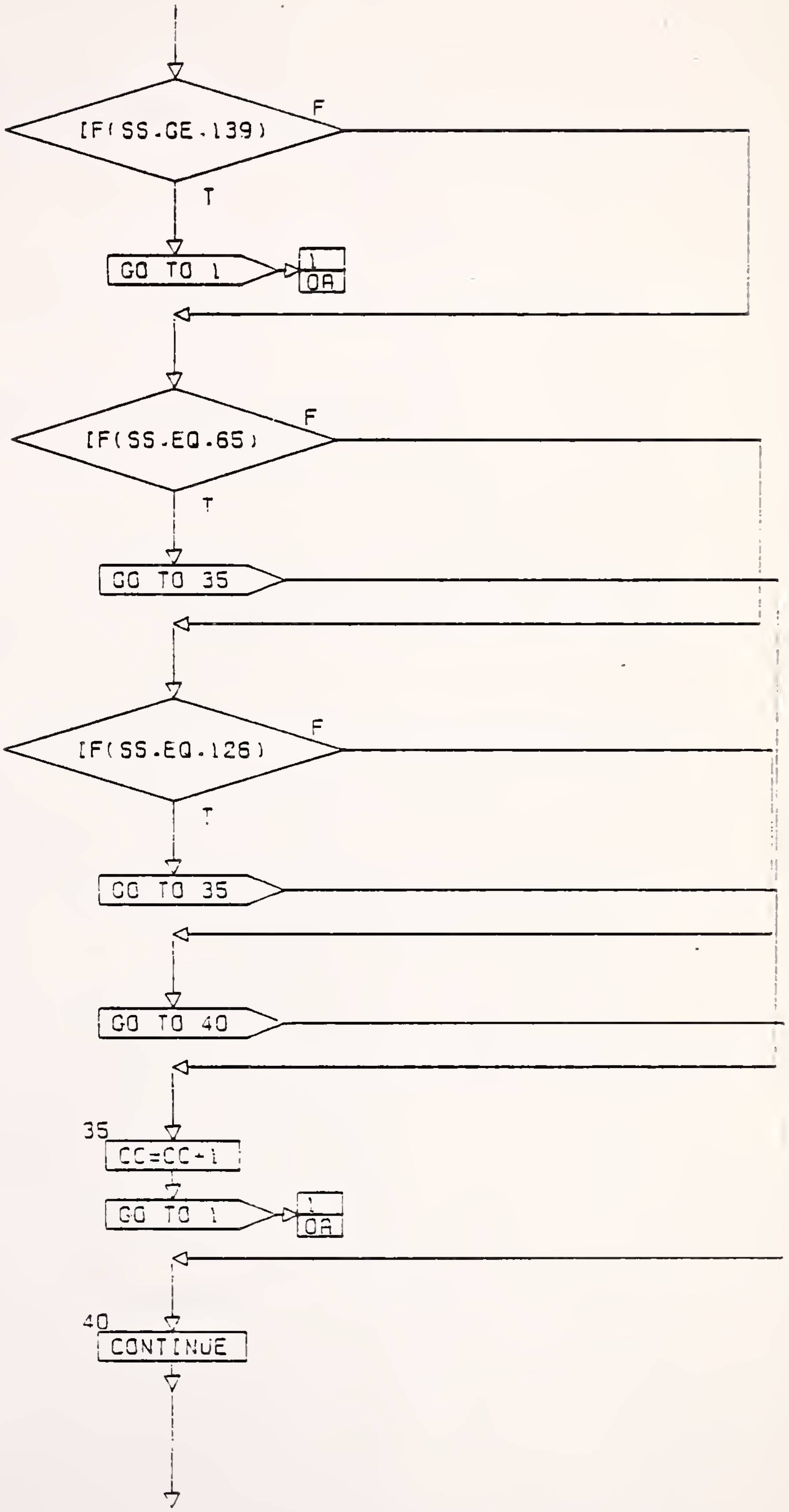


```
CALL CURVE(X1, Y1, 200,IM)

C         IF YOU WANT TO DRAW MORE THAN ONE CURVE
C         CALL CURVE(X2, Y2, 200, -1)
C         A MESSAGE WILL APPEAR IN THE RIGHT CORNER OF
C         YOUR DRAWING.

CALL RLMESS ('0-19$',4, 6.0 ,0.90)

CALL RLMESS('20-39$' , 5, 6.0, 0.85)

CALL ENDPL(0)

CALL DONEPL

CLOSE(UNIT=22)

STOP

END
```

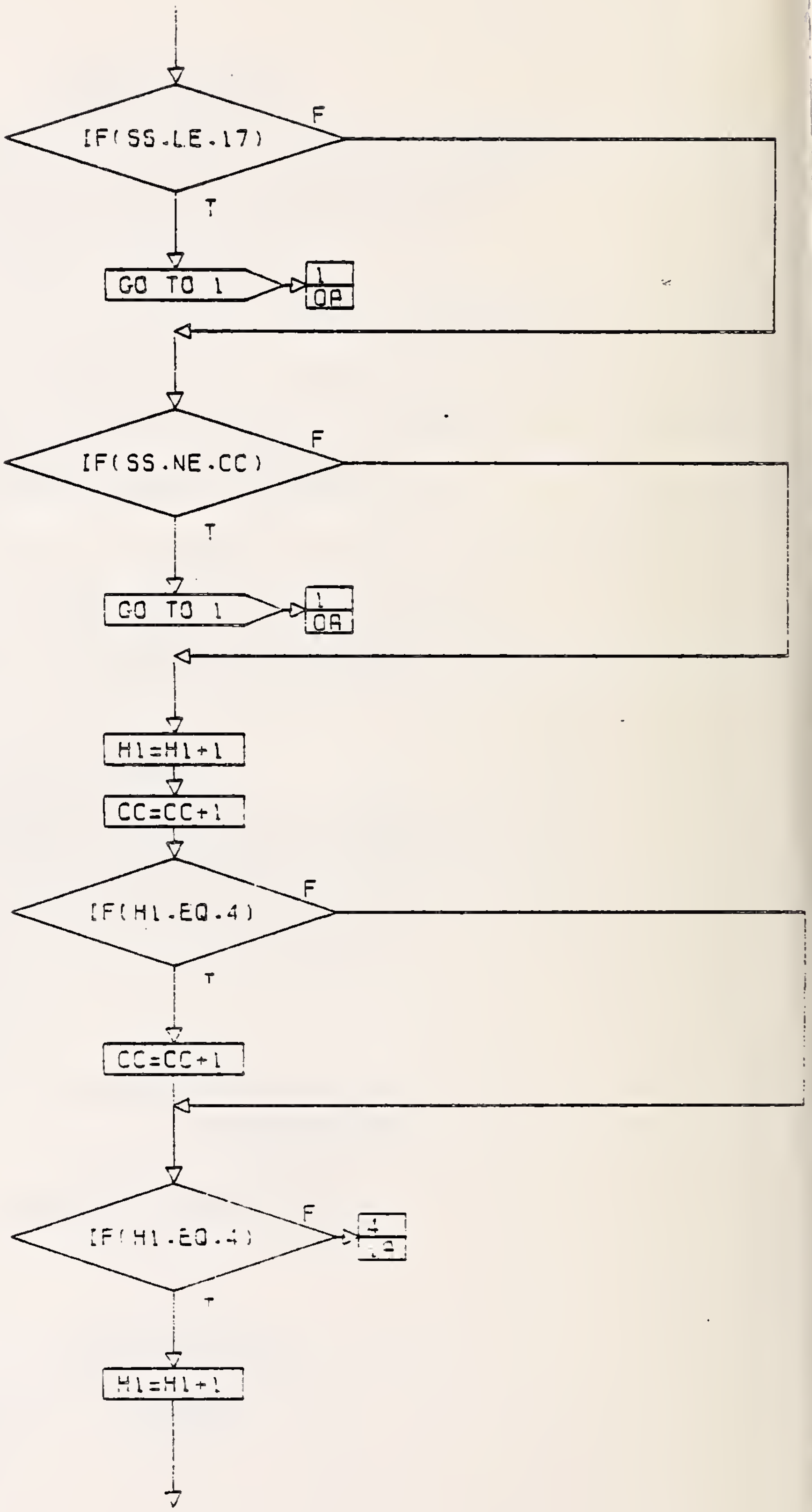


```
PROGRAM TRANSF

C        THIS PROGRAM  TRANSFORMS  THE VEHSIM OUTPUT
C        DATA INTO AN ARRAY OF 5-DIMENSIONAL VECTORS.
C        THIS PROGRAM CALLS ONE SUBROUTINE M80002.
C        THIS PROGRAM IS MACHINE-INDEPENDENT AND
C        CAN BE RUN ON ANY COMPUTER.
C        AS AN EXAMPLE  LETS CONSIDER AN OUTPUT
C        'ENGINE DATA'. THIS PROGRAM WILL TRANSFORM IT
C        INTO AN  ARRAY OF VECTORS WHERE COLUMN ONE

C        REPRESENTS SPEED(RPM). TWO - TORQUE(FT-LB).
C        THREE - FUEL RATE(LB/HR). FOUR - THROTTLE(DEGREES).
C        FIVE - MANIFOLD VACUUM(IN-HG). WITH THIS
C        KIND OF OUTPUT IT IS  VERY EASY TO AUTOMATE
C        THE INPUT PROCESS FOR A WIDE RANGE OF
C        PROGRAMS (NOT NECESSARILY CONNECTED WITH
C        VEHSIM).
C        AS AN  INPUT FOR THIS PROGRAM  WE USE

C        FILE NAMED 'M80001.DAT' ON DISK8.
C        OUTPUT WILL HAVE  NAME 'VAS.DAT' AND
C        WILL BE STORED ALSO ON DISK8.

COMMON P.F
DIMENSION ENG(200.5)
INTEGER P(26).R(90).KK.SS
INTEGER CC.H.H1.H4.MM.RR
REAL F(10).A(1000).SP(10)
DATA SP /550.00,800.00,1000.00,1200.00,1400.00,
1 1600.00,2000.00,2400.00,2800.00,3200.00/
OPEN(UNIT=21.FILE='M80001.DAT')

OPEN(UNIT=22.FILE='VAS.DAT')
KK=1
SS=0
CC=20
H=12
H1=0
H4=4

1  READ(21.10.END=1000)F

10 FORMAT(26A5)

SS=SS+1
```

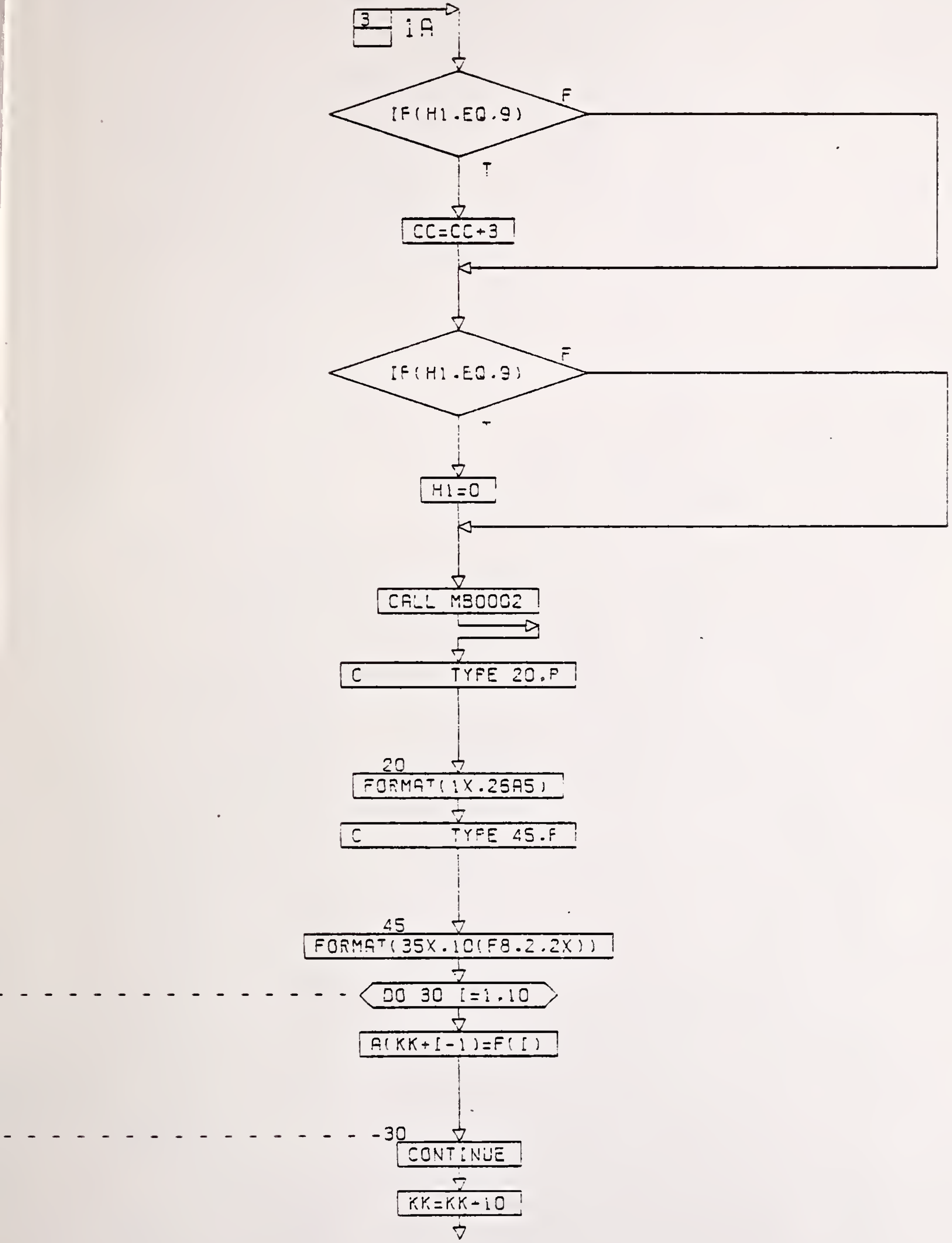
CONT. ON PG 2


IF(SS.GE.139)
F
T
GO TO 1
1
OA
IF(SS.EQ.65)
F
T
GO TO 35
IF(SS.EQ.126)
F
T
GO TO 35
GO TO 40
35
CC=CC+1
GO TO 1
1
OA
40
CONTINUE

CONT. ON PG 3

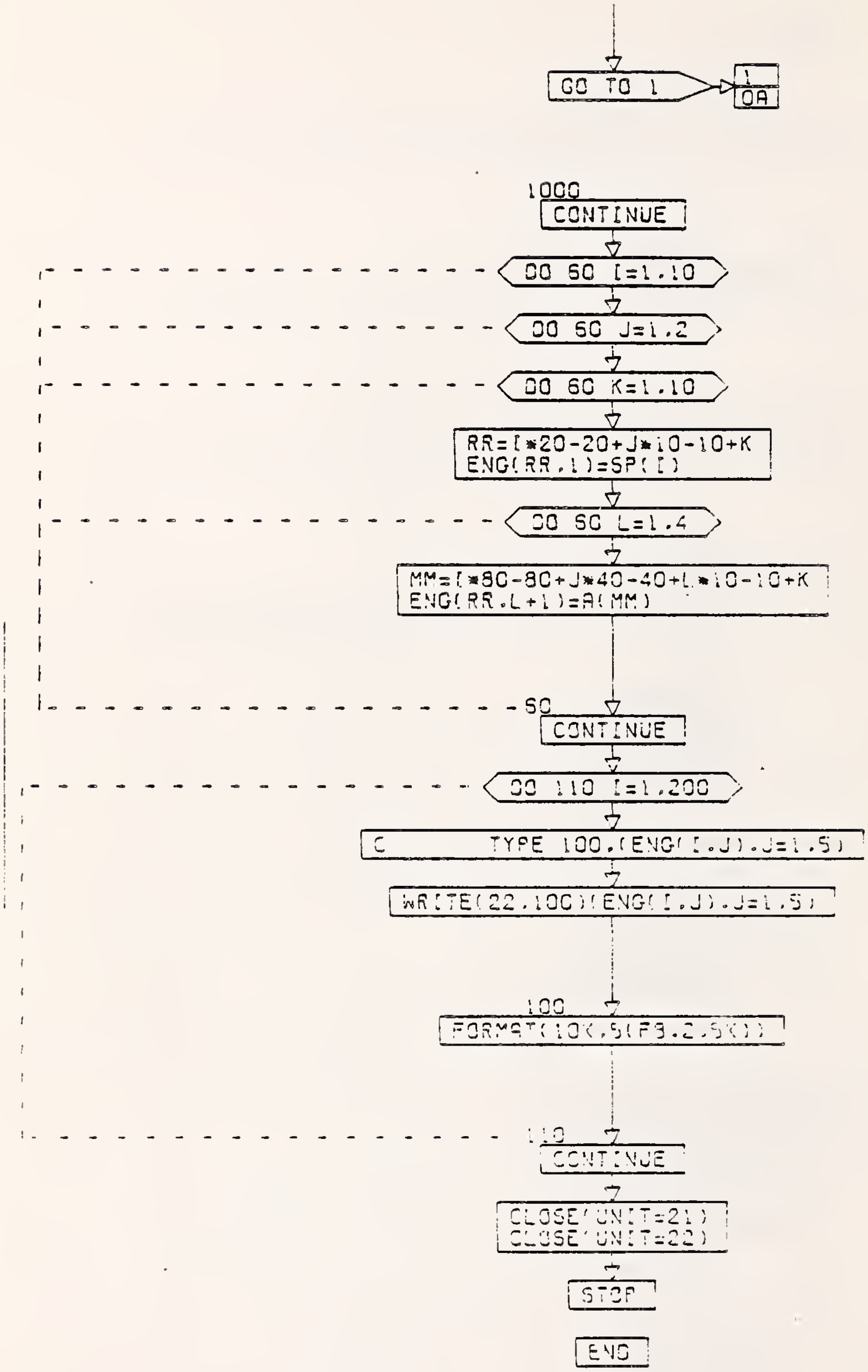

IF(SS.LE.17)
F
T
GO TO 1
1
OP
IF(SS.NE.CC)
F
T
GO TO 1
1
OA
H1=H1+1
CC=CC+1
IF(H1.EQ.4)
F
T
CC=CC+1
IF(H1.EQ.4)
F
1
13
T
H1=H1+1

CONT. ON PG 4



CONT. ON PG 5

PG 4 OF 5



```
      GO TO 1
1000  CONTINUE
      DO 60 I=1,10
      DO 60 J=1,2
      DO 60 K=1,10
      RR=I*20-20+J*10-10+K
      ENG(RR,1)=SP(I)
      DO 60 L=1,4
      MM=I*80-80+J*40-40+L*10-10+K
      ENG(RR,L+1)=A(MM)
60    CONTINUE
      DO 110 I=1,200
C           TYPE 100,(ENG(I,J),J=1,5)
      WRITE(22,100)(ENG(I,J),J=1,5)
100   FORMAT(10X,5(F8.2,5X))
110   CONTINUE
      CLOSE(UNIT=21)
      CLOSE(UNIT=22)
      STOP
      END
```

PG 5 FINAL

165 Copies

Zub, Russell

A computer p
for vehicle

Form DOT F 172(
FORMERLY FORM DO

U.S. Department
[illegible]ransportation

search and
ecial Programs
ministration

ali Square
bridge, Massachusetts 02142

Official Business
Penalty for Private Use $300