2
SA-81-14.III
0119-81-3
A-81-19.III

A Computer Program (HEVSIM) for Heavy Duty Vehicle Fuel Economy and Performance Simulation

Volume 3: Appendices A Through F

Richard E. Buck

Transportation Systems Center
Cambridge MA 02142

September 1981
Final Report

U.S. Department of Transportation

National Highway Traffic Safety Administration

Office of Research and Development
Washington DC 20590

Urban Mass Transportation Administration

Office of Technology Development and Deployment
Washington DC 20590

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
DOT-HS-805-912 UMTA-MA-06-0119-81-3		

4. Title and Subtitle	5. Report Date
A COMPUTER PROGRAM (HEVSIM) FOR HEAVY DUTY VEHICLE FUEL ECONOMY AND PERFORMANCE SIMULATION Volume III: Appendices A through F	September 1981
	6. Performing Organization Code
7. Author(s) R. Buck	8. Performing Organization Report No. DOT-TSC-NHTSA-81-14.III DOT-TSC-UMTA-81-19.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) HS277/R2414 UM262/R2659
	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 Office of Heavy Duty Vehicle Research Washington DC 20590	13. Type of Report and Period Covered Final Report March 1980 - October 1980
	14. Sponsoring Agency Code

15. Supplementary Notes

*Joint Sponsor: U.S. Department of Transportation
Urban Mass Transportation Administration
Office of Technology Development and Deployment
Office of Bus and Paratransit Technology
Washington DC 20590

16. Abstract

This report presents a description of a vehicle simulation program, which can determine the fuel economy and performance of a specified motor vehicle over a defined route as it executes a given driving schedule. Vehicle input accommodated by HEVSIM include accessories, engine, rear axle, converter, transmission, tires, aerodynamic drag coefficient, and shift logic. The report consists of three volumes. Volume I presents a description of the numerical approach and equations, Volume II is a user's manual, and Volume III contains the program listings.

17. Key Words	18. Distribution Statement
Motor Vehicle, Truck, Bus, Simulation, Fuel Economy, Performance	DOCUMENT IS AVAILABLE TO THE PUBLIC THROUGH THE NATIONAL TECHNICAL INFORMATION SERVICE, SPRINGFIELD, VIRGINIA 22161

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	326	

PREFACE

Volume III is the third and last volume of a three volume document describing the computer program HEVSIM. This volume includes appendicies which list the HEVSIM program, sample part data, some typical outputs and updated nonmenclature.

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	1.8	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

*1 in = 2.54 (exactly). For other exact conversions and more detailed tables, see NBS Misc. Publ. 286, Units of Weights and Measures, Price $2.25, SD Catalog No. C13.10:286.

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

APPENDIX		Page
A	TYPICAL OUTPUTS	A-1
B	UPDATED DATA SHEETS	B-1
C	PROGRAM LISTINGS	C-1
D	UPDATED NOMENCLATURE	D-1
E	CONTROL FILES	E-1
F	SELECTED FLOW CHARTS	F-1

APPENDIX A

Typical Outputs

ENGINE DATA (871N60)

8V-71N-C60 RATED AT 280 BHP AT 2100 RPM ENGINE FUEL CONSUMPTION DATA

CYLINDERS = 8 FUEL DENSITY = 7.043 LB/GAL
BORE = 4.250 ROTATING INERTIA = 2.620 FT-LB-SEC**2
STROKE = 5.000
DISPLACEMENT = 567.5 MINIMUM MAXIMUM

8 SPEED POINTS THROTTLE ANGLE = 0.00 0.00 DEGREES
ENGINE SPEED = 550.0 2100.0 RPM

SPEED (RPM) = 800.00										
POWER(HP)	-17.00	0.00	30.00	50.00	60.00	70.00	80.00	90.00	100.00	109.50
FUEL RATE(LB/HR)	0.00	6.00	13.80	19.50	23.30	27.00	31.50	36.00	41.00	46.00
THROTTLE(DEGREES)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MANIFOLD VACUUM(IN-HG)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SPEED (RPM) = 1000.00										
POWER(HP)	-20.50	0.00	30.00	40.00	60.00	80.00	100.00	120.00	140.00	150.00
FUEL RATE(LB/HR)	0.00	6.50	15.20	18.00	24.00	30.60	37.70	46.00	56.70	62.50
THROTTLE(DEGREES)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MANIFOLD VACUUM(IN-HG)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SPEED (RPM) = 1200.00										
POWER(HP)	-24.00	0.00	20.00	40.00	60.00	80.00	100.00	125.00	150.00	175.00
FUEL RATE(LB/HR)	0.00	8.00	13.70	19.00	25.60	31.80	38.80	47.50	57.40	70.50
THROTTLE(DEGREES)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MANIFOLD VACUUM(IN-HG)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SPEED (RPM) = 1400.00										
POWER(HP)	-39.50	0.00	20.00	40.00	60.00	93.00	125.00	150.00	175.00	203.00
FUEL RATE(LB/HR)	0.00	9.50	15.20	21.00	27.00	39.90	48.10	56.50	65.60	80.20
THROTTLE(DEGREES)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MANIFOLD VACUUM(IN-HG)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SPEED (RPM) = 1600.00										
POWER(HP)	-40.00	0.00	30.00	40.00	70.00	95.00	120.00	150.00	200.00	230.00
FUEL RATE(LB/HR)	0.00	12.00	20.00	23.50	31.70	39.30	47.30	56.50	75.00	89.10
THROTTLE(DEGREES)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MANIFOLD VACUUM(IN-HG)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SPEED (RPM) = 1800.00										
POWER(HP)	-59.00	0.00	25.00	50.00	100.00	120.00	150.00	180.00	210.00	250.00
FUEL RATE(LB/HR)	0.00	15.50	22.50	28.50	43.40	49.20	58.20	68.00	78.50	96.30
THROTTLE(DEGREES)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MANIFOLD VACUUM(IN-HG)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SPEED (RPM) = 2000.00										
POWER(HP)	-71.00	0.00	20.00	40.00	80.00	120.00	160.00	200.00	240.00	270.00
FUEL RATE(LB/HR)	0.00	16.50	23.50	29.40	40.50	51.90	63.70	76.50	91.10	104.00
THROTTLE(DEGREES)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MANIFOLD VACUUM(IN-HG)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
SPEED (RPM) = 2100.00										
POWER(HP)	-77.00	0.00	20.00	40.00	80.00	120.00	160.00	200.00	240.00	280.00
FUEL RATE(LB/HR)	0.00	21.00	26.80	32.00	42.50	53.60	65.50	78.40	92.50	108.60
THROTTLE(DEGREES)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MANIFOLD VACUUM(IN-HG)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

DRIVING SCHEDULE (TRANSCEDA)

TRANS BUS CENTRAL BUSINESS DIST. 100% WOT ACCEL

INITIAL CONDITIONS

TIME	=	0.00	SEC
DISTANCE	=	0.00	MI
VEHICLE SPEED	=	0.00	MPH
ACCELERATION	=	0.00	MI/SEC**2
STARTING GEAR	=	1	
REAR AXLE	=	1	

SEGMENT DESIRED PERFORMANCE

SEGMENT NUMBER	********** SEGMENT TYPE ********** CONSTANT ACCELER	CONSTANT SPEED	CONSTANT PERCENT WOT	ACCEL IC AND POLD THROTTLE	HOLD GEAR	THROTTLE RATE OF CHANGE	********** SEGMENT ENDPOINT ********** RELATIVE TIME	RELATIVE DISTANCE	PASSING CLEARANCE	ABSOLUTE VELOCITY	ABSOLUTE VELOCITY	SEGMENT NUMBER
1			100.00								20.00	1
2		20.00								0.13		2
3	-6.79										0.00	3
4		0.00					7.00					4

ACCESSORY LOSS DATA (BUS 1)

TOTAL ACCESSORIES = LUMPED LOSS (AVERAGE VALUE)

INERTIA = 0.000 (LB-FT-SEC**2)

SPEED RATIO = 1.000

DUTY CYCLE = 1.000

SPEED (RPM)	TORQUE (LB-FT)
1200.0	100.700
1400.0	105.000
1600.0	110.000
1800.0	115.500
2000.0	122.600
2100.0	126.300

AXLE DATA (BUS-2)

4.625 AXLE

AXLE SPEED #	RAR	EFFICIENCIES
1	4.63	0.95 1.00

NO AXLE SPIN LOSS DATA SPECIFIED

SHIFT LOGIC (V730-ST)

D DED V730 STANDARD CALIBRATION

THIS TRANSMISSION HAS 4 GEARS

SHIFT LINE	1 - 2	AXLE SPEED	1 - 1	SHIFT TIME =	1.000 SEC
		THROTTLE (ECT WOT)	0.00	100.00	
		PROPSHAFT SPEED (RPM)	625.00	995.00	
SHIFT LINE	2 - 3	AXLE SPEED	1 - 1	SHIFT TIME =	1.000 SEC
		THROTTLE (ECT WOT)	0.00	100.00	
		PROPSHAFT SPEED (RPM)	700.00	1495.00	
SHIFT LINE	3 - 4	AXLE SPEED	1 - 1	SHIFT TIME =	1.000 SEC
		THROTTLE (ECT WOT)	0.00	100.00	
		PROPSHAFT SPEED (RPM)	1635.00	1640.00	
SHIFT LINE	4 - 3	AXLE SPEED	1 - 1	SHIFT TIME =	1.000 SEC
		THROTTLE (ECT WOT)	0.00	100.00	
		PROPSHAFT SPEED (RPM)	1495.00	1490.00	
SHIFT LINE	3 - 2	AXLE SPEED	1 - 1	SHIFT TIME =	1.000 SEC
		THROTTLE (ECT WOT)	0.00	100.00	
		PROPSHAFT SPEED (RPM)	430.00	1340.00	
SHIFT LINE	2 - 1	AXLE SPEED	1 - 1	SHIFT TIME =	1.000 SEC
		THROTTLE (ECT WOT)	0.00	100.00	
		PROPSHAFT SPEED (RPM)	400.00	800.00	

ROUTE SPECIFICATION (ZGRADE)

500 MILES OF LEVEL ROAD

POINT	DISTANCE (MILES)	PERCENT GRADE	ROAD COEFFICIENT	WIND SPEED (FPS)
1	500.000	0.000	1.000	1000.000

TIRE DATA (POS 1)

NON-RADIAL BUS TIRE

ROLLING RADIUS = 1.625 FT C1 = 0.010000

C2 = 0.000000

TIRE EFFICIENCY = 1.000

WHEEL INERTIA = 10.000 FT-LI-SEC**2 C4 = 0.000400

TRANSMISSION DATA (V-730E)

BUS-1 TRANSMISSION, 4 SPEED AUTOMATIC, GEAR 2=3

1 - V730E-1
2 - V730E-2
3 - V730E-2A
4 - V730E-3

GEAR DATA (V730E-1)

GEAR RATIO = 1.770 INPUT INERTIA = 0.015 FT-LB-SEC**2

EFFICIENCY = 0.980 OUTPUT INERTIA = 0.015 FT-LB-SEC**2

GEAR SPIN LOSS DATA

SPEED (RPM)	TORQUE (LB-FT)
800.0	2.200
1600.0	3.300
2400.0	4.800

GEAR DATA (V730E-2)

GEAR RATIO = 1.210 INPUT INERTIA = 0.015 FT-LB-SEC**2

EFFICIENCY = 0.990 OUTPUT INERTIA = 0.015 FT LB-SEC**2

GEAR SPIN LOSS DATA

SPEED (RPM)	TORQUE (LB-FT)
800.0	2.200
1600.0	3.300
2400.0	4.400

GEAR DATA (V730E-2A)

GEAR RATIO	=	1.209	INPUT INERTIA	=	0.020 FT-LB-SEC**2
EFFICIENCY	=	0.980	OUTPUT INERTIA	=	0.020 FT-LB-SEC**2

GEAR SPIN LOSS DATA

SPEED (RPM)	TORQUE (LB-FT)
800.0	2.200
1600.0	3.300
2400.0	4.600

GEAR DATA (V730E-3)

GEAR RATIO = 0.880 INPUT INERTIA = 0.015 FT-LB-SEC**2

EFFICIENCY = 0.980 OUTPUT INERTIA = 0.015 FT-LB-SEC**2

GEAR SPIN LOSS DATA

SPEED (RPM)	TORQUE (LB-FT)
800.0	2.200
1600.0	3.300
2400.0	4.600

VEHICLE DATA (DES-1)

STANDARD BUS

WEIGHT = 34600.0 LBS

FRONTAL AREA = 75.00 SQ FT NUMBER OF TIRES = 6

DRAG COEFFICIENT = 0.600000 CD SENSITIVITY COEFF = 1.000000

PROPSHAFT INERTIA = 0.000 FT-LB-SEC**2

BEVEL GEAR = 0.875; BEVEL GEAR EFF. = 0.980

DRIVE CONVERTER DATA (TC-490)

D PAD TC-490 BUS CONVERTER

DIAMETER = 24.0 POPE INERTIA = 2.000 FT-LB-SEC**2

TURBINE INERTIA = 2.000 FT-LB-SEC**2

CONSTANT INPUT TORQUE = 50.00 LB-FT

SPEED RATIO	0.000	0.100	0.200	0.300	0.370	0.475	0.600	0.700	0.750	0.800
TORQUE RATIO	2.510	2.360	2.200	2.030	1.890	1.680	1.460	1.290	1.210	1.120
INPUT SPEED	477.000	476.000	477.000	487.000	494.000	509.000	528.000	559.900	576.700	603.300
K-FACTOR	0.000	4.400	9.096	14.502	18.802	26.380	37.079	48.801	55.800	64.455
SPEED RATIO	0.830	0.865	0.900	0.925	0.950	0.975	0.990			
TORQUE RATIO	1.060	0.990	0.930	0.990	0.990	0.980	0.990			
INPUT SPEED	621.900	653.100	709.000	780.400	904.300	1229.800	1787.500			
K-FACTOR	70.907	80.296	90.696	102.602	122.105	171.294	252.804			

COAST CONVERTER DATA (TC-450-C)

D PAD TC-450 BUS CONVERTER

DIAMETER = 24.0 PUMP INERTIA = 2.000 FT-LB-SEC**2

TURBINE INERTIA 2.000 FT-LB-SEC**2

CONSTANT INPUT TORQUE = 20.00 LB-FT

SPEED RATIO	0.500	1.400	1.130	1.050	1.030	1.020	1.013	1.010	1.010
TORQUE RATIO	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
INPUT SPEED	600.000	1000.000	1500.000	2000.000	2500.000	3000.000	3500.000	4000.000	4500.000

```
 VEHSIM TRUCK  07(24) STATUS REPORT
 ----------------------------------

BATCH MODE     /PIY
SIMULATION MODE-BUS
RUN TITLE-TEST CASES OF VEHSIM
ENGINE(1)-871N60
DRIVE CONVERTER-TC-920
COAST CONVERTER-TC-920-C
VEHICLE-BUS-1
AXLE - BUS-2
TRANSMISSION - V-730B
GEAR # 1-V730B-1     ASSIGNED TO ENGINE-871N60
GEAR # 2-V730B-2     ASSIGNED TO ENGINE-871N60
GEAR # 3-V730B-3A    ASSIGNED TO ENGINE-871N60
GEAR # 4-V730B-3     ASSIGNED TO ENGINE-871N60
ACCESSORY #  1-BUS 1
DRIVING SCHEDULE-TRABUSCEDA
SHIFT LOGIC-V730-SP
ROUTE-ZGRADE
TIRE-BUS 1
GEARS LOCKED UP-  3  4
GEARS UNLOCKED -  1  2
LIMIT PRINT-SECON  0.5000000E-01
DEBUG-OFF     0.0000000E+00  0.0000000E+00
TTY OUTPUT-OFF
IPT OUTPUT-ON
REAR  MODIFIED FROM  4.625      TO  5.350
WEIGH MODIFIED FROM  .2460E+05 TO  .2000E+05
```

VEHICLE PERFORMANCE SIMULATION

RUN TITLE (TEST CASES OF VEHSIM)

DRIVING SCHEDULE (TRABUSCBDA) USING ROUTE (ZGRADE)

SEC.	MILES	MPH	ACC	INST MPG	BSFC	CUM MPG	GEAR		BOLD	HPW	HPE	RPM	TORQ	VAC	SS	ETA	%WOT	DRS SEG	BRAKES	GLOBLT
0.00	0.000	0.0	0.00	0.00	0.78	0.0	1	1	0.0	0.0	9.1	550.	87.	0.0	0.000	0.000	27.	1	507.0	0.00
0.08	0.000	0.0	0.00	0.00	0.72	0.0	1	1	0.0	0.0	12.5	750.	88.	0.0	0.004	0.010	27.	1	507.0	0.00
0.15	0.000	0.0	0.00	0.00	0.69	0.0	1	1	0.0	0.0	15.9	950.	89.	0.0	0.004	0.010	27.	1	507.0	0.00
0.22	0.000	0.1	3.00	0.02	0.39	0.0	1	1	0.1	0.7	68.3	948.	378.	0.0	0.005	0.014	27.	1	0.0	10.24
0.29	0.000	0.2	3.00	0.06	0.39	0.0	1	1	0.2	2.0	68.6	948.	380.	0.0	0.016	0.041	27.	1	0.0	10.24
0.37	0.000	0.4	3.00	0.10	0.39	0.0	1	1	0.3	3.3	69.0	948.	382.	0.0	0.027	0.068	27.	1	0.0	10.24
0.44	0.000	0.5	3.00	0.13	0.39	0.1	1	1	0.4	4.6	69.3	948.	384.	0.0	0.038	0.094	27.	1	0.0	10.24
0.51	0.000	0.7	3.00	0.17	0.39	0.1	1	1	0.5	5.9	69.7	948.	386.	0.0	0.049	0.120	27.	1	0.0	10.24
0.58	0.000	0.8	3.00	0.21	0.39	0.1	1	1	0.7	7.2	70.0	948.	388.	0.0	0.060	0.146	27.	1	0.0	10.24
0.65	0.000	0.9	3.00	0.24	0.39	0.1	1	1	0.8	8.5	70.4	948.	390.	0.0	0.071	0.171	27.	1	0.0	10.24
0.72	0.000	1.1	3.00	0.28	0.39	0.1	1	1	0.9	9.8	70.8	948.	392.	0.0	0.082	0.196	27.	1	0.0	10.24
0.79	0.000	1.2	3.00	0.32	0.39	0.1	1	1	1.0	11.1	71.1	948.	394.	0.0	0.093	0.221	27.	1	0.0	10.24
0.86	0.000	1.4	3.00	0.33	0.38	0.2	1	1	1.2	12.4	77.6	951.	428.	0.0	0.104	0.245	27.	1	0.0	10.24
0.94	0.000	1.5	3.00	0.36	0.38	0.2	1	1	1.3	13.7	79.0	956.	434.	0.0	0.114	0.267	27.	1	0.0	10.24
1.01	0.000	1.7	3.00	0.39	0.38	0.2	1	1	1.4	15.0	78.8	961.	431.	0.0	0.124	0.289	27.	1	0.0	10.24
1.08	0.000	1.8	3.00	0.43	0.38	0.2	1	1	1.5	16.4	78.7	965.	428.	0.0	0.135	0.310	27.	1	0.0	10.24
1.15	0.000	2.0	3.00	0.46	0.38	0.2	1	1	1.6	17.7	78.7	968.	427.	0.0	0.145	0.332	27.	1	0.0	10.24
1.22	0.000	2.1	3.00	0.49	0.38	0.3	1	1	1.8	19.0	78.9	971.	427.	0.0	0.155	0.353	27.	1	0.0	10.24
1.29	0.000	2.3	3.00	0.53	0.38	0.3	1	1	1.9	20.3	79.1	974.	427.	0.0	0.166	0.373	27.	1	0.0	10.24
1.36	0.000	2.4	3.00	0.56	0.38	0.3	1	1	2.0	21.6	79.4	976.	427.	0.0	0.176	0.394	27.	1	0.0	10.24
1.43	0.000	2.5	3.00	0.59	0.38	0.3	1	1	2.1	22.9	79.7	978.	428.	0.0	0.186	0.413	27.	1	0.0	10.24
1.51	0.000	2.7	3.00	0.62	0.38	0.3	1	1	2.2	24.2	80.0	980.	429.	0.0	0.196	0.433	27.	1	0.0	10.24
1.58	0.001	2.8	3.00	0.59	0.38	0.3	1	1	2.4	25.5	88.9	986.	474.	0.0	0.206	0.451	27.	1	0.0	10.24
1.65	0.001	3.0	3.00	0.62	0.38	0.3	1	1	2.5	26.8	89.2	993.	472.	0.0	0.215	0.467	27.	1	0.0	10.24
1.72	0.001	3.1	3.00	0.65	0.38	0.4	1	1	2.6	28.1	89.5	1000.	470.	0.0	0.224	0.483	27.	1	0.0	10.24
1.79	0.001	3.3	3.00	0.68	0.38	0.4	1	1	2.7	29.4	89.9	1006.	469.	0.0	0.232	0.499	27.	1	0.0	10.24
1.86	0.001	3.4	3.00	0.70	0.38	0.4	1	1	2.9	30.8	90.3	1012.	469.	0.0	0.241	0.514	27.	1	0.0	10.24
1.93	0.001	3.6	3.00	0.73	0.38	0.4	1	1	3.0	32.1	90.8	1018.	468.	0.0	0.250	0.529	27.	1	0.0	10.24
2.00	0.001	3.7	3.00	0.75	0.38	0.4	1	1	3.1	33.4	91.2	1023.	468.	0.0	0.259	0.544	27.	1	0.0	10.24
2.08	0.001	3.9	3.00	0.78	0.38	0.4	1	1	3.2	34.7	91.7	1028.	469.	0.0	0.268	0.559	27.	1	0.0	10.24
2.15	0.001	4.0	3.00	0.80	0.38	0.5	1	1	3.4	36.0	92.2	1032.	469.	0.0	0.277	0.573	27.	1	0.0	10.24
2.22	0.001	4.2	3.00	0.83	0.38	0.5	1	1	3.5	37.3	92.8	1036.	470.	0.0	0.286	0.587	27.	1	0.0	10.24
2.29	0.001	4.3	3.00	0.85	0.38	0.5	1	1	3.6	38.6	93.3	1040.	471.	0.0	0.295	0.601	27.	1	0.0	10.24
2.36	0.001	4.4	3.00	0.84	0.38	0.5	1	1	3.7	39.9	98.3	1046.	494.	0.0	0.303	0.614	27.	1	0.0	10.24
2.43	0.001	4.6	3.00	0.85	0.38	0.5	1	1	3.8	41.2	99.8	1053.	498.	0.0	0.311	0.625	27.	1	0.0	10.24
2.50	0.002	4.7	3.00	0.87	0.38	0.5	1	1	4.0	42.5	100.6	1059.	499.	0.0	0.319	0.635	27.	1	0.0	10.24
2.57	0.002	4.9	3.00	0.89	0.38	0.5	1	1	4.1	43.9	101.3	1066.	499.	0.0	0.327	0.646	27.	1	0.0	10.24
2.64	0.002	5.0	3.00	0.91	0.39	0.5	1	1	4.2	45.2	102.1	1072.	500.	0.0	0.335	0.656	27.	1	0.0	10.24
2.73	0.002	5.2	3.20	0.49	0.47	0.6	1	1	4.4	49.6	163.0	1105.	774.	0.0	0.336	0.658	100.	1	0.0	10.93
2.78	0.002	5.3	3.39	0.48	0.47	0.6	1	1	4.5	53.4	167.4	1133.	776.	0.0	0.335	0.657	100.	1	0.0	11.58
2.83	0.002	5.4	3.56	0.51	0.44	0.5	1	1	4.6	57.2	170.4	1159.	772.	0.0	0.335	0.657	100.	1	0.0	12.17
2.88	0.002	5.6	3.71	0.54	0.42	0.5	1	1	4.7	60.8	173.2	1183.	769.	0.0	0.336	0.658	100.	1	0.0	12.70
2.93	0.002	5.7	3.85	0.57	0.40	0.5	1	1	4.8	64.4	175.8	1204.	767.	0.0	0.338	0.660	100.	1	0.0	13.19
2.98	0.002	5.8	3.99	0.57	0.40	0.5	1	1	4.9	68.0	179.1	1225.	768.	0.0	0.340	0.663	100.	1	0.0	13.63
3.03	0.002	6.0	4.09	0.58	0.40	0.6	1	1	5.0	71.4	180.8	1243.	764.	0.0	0.343	0.667	100.	1	0.0	14.02

VEHICLE PERFORMANCE SIMULATION

RUN TITLE (TEST CASES OF VEHSIM)

DRIVING SCHEDULE (TRAPUSCBDA) USING ROUTE (ZGRADE)

SEC.	MILES	MPH	ACC	INST MPG	BSFC	CUM MPG	GEAR		RDLD	HPW	HPE	RPM	TORQ	VAC	SP	ETA	%WOT	ORS SEG	BRAKES	GRDELT
3.08	0.002	6.1	4.19	0.59	0.40	0.6	1	1	5.2	74.9	183.9	1260.	767.	0.0	0.347	0.671	100.	1	0.0	14.38
3.13	0.002	6.3	4.28	0.60	0.40	0.6	1	1	5.3	78.2	185.2	1276.	762.	0.0	0.351	0.676	100.	1	0.0	14.69
3.18	0.003	6.4	4.36	0.60	0.40	0.6	1	1	5.4	81.4	187.7	1290.	764.	0.0	0.355	0.681	100.	1	0.0	14.97
3.23	0.003	6.6	4.43	0.61	0.40	0.6	1	1	5.5	84.6	190.1	1304.	766.	0.0	0.359	0.687	100.	1	0.0	15.23
3.28	0.003	6.7	4.50	0.62	0.40	0.6	1	1	5.7	87.8	192.3	1317.	767.	0.0	0.364	0.693	100.	1	0.0	15.45
3.33	0.003	6.9	4.55	0.63	0.40	0.6	1	1	5.8	90.8	192.2	1328.	760.	0.0	0.370	0.699	100.	1	0.0	15.63
3.38	0.003	7.0	4.57	0.64	0.40	0.6	1	1	6.0	93.3	195.1	1339.	765.	0.0	0.375	0.705	100.	1	0.0	15.77
3.43	0.003	7.2	4.59	0.65	0.40	0.6	1	1	6.1	95.7	196.6	1349.	766.	0.0	0.380	0.711	100.	1	0.0	15.78
3.48	0.003	7.3	4.60	0.66	0.40	0.6	1	1	6.2	98.0	197.2	1359.	762.	0.0	0.386	0.717	100.	1	0.0	15.82
3.53	0.003	7.5	4.61	0.67	0.39	0.6	1	1	6.4	100.4	199.1	1368.	765.	0.0	0.391	0.723	100.	1	0.0	15.86
3.58	0.003	7.7	4.62	0.68	0.40	0.6	1	1	6.5	102.6	199.4	1376.	761.	0.0	0.397	0.729	100.	1	0.0	15.88
3.63	0.003	7.8	4.62	0.69	0.40	0.6	1	1	6.6	104.8	201.0	1384.	762.	0.0	0.403	0.735	100.	1	0.0	15.89
3.68	0.004	8.0	4.63	0.70	0.39	0.6	1	1	6.8	107.0	202.8	1392.	765.	0.0	0.409	0.741	100.	1	0.0	15.91
3.73	0.004	8.1	4.62	0.72	0.39	0.6	1	1	6.9	109.1	202.6	1399.	760.	0.0	0.415	0.747	100.	1	0.0	15.90
3.78	0.004	8.3	4.62	0.73	0.39	0.6	1	1	7.1	111.1	203.9	1406.	762.	0.0	0.421	0.753	100.	1	0.0	15.89
3.83	0.004	8.4	4.62	0.74	0.39	0.6	1	1	7.2	113.1	205.2	1413.	763.	0.0	0.427	0.758	100.	1	0.0	15.88
3.88	0.004	8.6	4.61	0.75	0.39	0.6	1	1	7.3	115.2	206.6	1419.	765.	0.0	0.433	0.764	100.	1	0.0	15.86
3.93	0.004	8.8	4.60	0.76	0.39	0.6	1	1	7.5	117.1	206.0	1424.	760.	0.0	0.439	0.769	100.	1	0.0	15.83
3.98	0.004	8.9	4.59	0.77	0.39	0.6	1	1	7.6	118.9	206.9	1430.	760.	0.0	0.445	0.775	100.	1	0.0	15.79
4.03	0.004	9.1	4.58	0.78	0.39	0.6	1	1	7.8	120.8	208.0	1435.	761.	0.0	0.451	0.780	100.	1	0.0	15.76
4.08	0.005	9.2	4.57	0.79	0.39	0.6	1	1	7.9	122.6	209.1	1439.	763.	0.0	0.458	0.785	100.	1	0.0	15.72
4.13	0.005	9.4	4.56	0.81	0.39	0.6	1	1	8.1	124.2	208.1	1443.	757.	0.0	0.464	0.790	100.	1	0.0	15.66
4.18	0.005	9.5	4.54	0.82	0.39	0.6	1	1	8.2	125.9	208.6	1447.	757.	0.0	0.471	0.795	100.	1	0.0	15.60
4.23	0.005	9.7	4.52	0.83	0.39	0.6	1	1	8.3	127.3	210.1	1451.	761.	0.0	0.477	0.800	100.	1	0.0	15.52
4.28	0.005	9.8	4.49	0.84	0.39	0.6	1	1	8.5	128.2	210.8	1455.	761.	0.0	0.483	0.805	100.	1	0.0	15.38
4.33	0.005	10.0	4.44	0.85	0.39	0.6	1	1	8.6	129.2	211.3	1459.	761.	0.0	0.489	0.810	100.	1	0.0	15.25
4.38	0.005	10.1	4.41	0.86	0.39	0.6	1	1	8.8	130.3	213.0	1463.	765.	0.0	0.495	0.814	101.	1	0.0	15.15
4.43	0.005	10.3	4.38	0.87	0.39	0.6	1	1	8.9	131.4	213.2	1467.	763.	0.0	0.501	0.819	100.	1	0.0	15.04
4.48	0.006	10.4	4.35	0.88	0.39	0.6	1	1	9.0	132.5	213.2	1471.	761.	0.0	0.507	0.823	100.	1	0.0	14.94
4.53	0.006	10.6	4.32	0.89	0.39	0.6	1	1	9.2	133.6	213.2	1475.	759.	0.0	0.513	0.827	100.	1	0.0	14.84
4.58	0.006	10.7	4.29	0.91	0.39	0.7	1	1	9.3	134.6	213.2	1479.	757.	0.0	0.518	0.831	100.	1	0.0	14.74
4.63	0.006	10.9	4.27	0.92	0.39	0.7	1	1	9.4	135.6	213.1	1482.	755.	0.0	0.524	0.835	100.	1	0.0	14.64
4.68	0.006	11.0	4.24	0.92	0.39	0.7	1	1	9.6	136.7	215.3	1486.	761.	0.0	0.530	0.839	100.	1	0.0	14.56
4.73	0.006	11.2	4.22	0.93	0.39	0.7	1	1	9.7	137.9	215.9	1489.	761.	0.0	0.536	0.843	100.	1	0.0	14.49
4.78	0.007	11.3	4.20	0.95	0.39	0.7	1	1	9.9	139.0	216.3	1493.	761.	0.0	0.541	0.846	100.	1	0.0	14.41
4.83	0.007	11.5	4.18	0.96	0.39	0.7	1	1	10.0	140.1	216.6	1496.	760.	0.0	0.547	0.849	100.	1	0.0	14.34
4.88	0.007	11.6	4.16	0.97	0.39	0.7	1	1	10.1	141.2	217.0	1499.	760.	0.0	0.552	0.853	100.	1	0.0	14.26
4.93	0.007	11.7	4.14	0.98	0.39	0.7	1	1	10.3	142.2	217.3	1503.	760.	0.0	0.558	0.856	100.	1	0.0	14.19
4.98	0.007	11.9	4.12	0.99	0.39	0.7	1	1	10.4	143.3	217.7	1505.	759.	0.0	0.563	0.859	100.	1	0.0	14.12
5.03	0.007	12.0	4.10	1.00	0.39	0.7	1	1	10.5	144.3	218.0	1508.	759.	0.0	0.569	0.862	100.	1	0.0	14.04
5.08	0.007	12.2	4.08	1.01	0.39	0.7	1	1	10.7	145.3	218.3	1511.	759.	0.0	0.575	0.865	100.	1	0.0	13.97
5.13	0.008	12.3	4.05	1.02	0.39	0.7	1	1	10.8	146.2	218.7	1513.	759.	0.0	0.580	0.867	100.	1	0.0	13.90
5.18	0.008	12.4	4.03	1.03	0.39	0.7	1	1	10.9	147.2	219.0	1516.	759.	0.0	0.586	0.870	100.	1	0.0	13.83
5.23	0.009	12.6	4.01	1.04	0.39	0.7	1	1	11.1	148.1	219.3	1518.	759.	0.0	0.591	0.872	100.	1	0.0	13.75
5.28	0.009	12.7	3.99	1.05	0.39	0.7	1	1	11.2	149.0	219.6	1520.	759.	0.0	0.597	0.875	100.	1	0.0	13.68

SHIFT FREQUENCY DATA BY GEAR

TOTAL SHIFTS = 4 SHIFTS PER MILE = 28.0 NUMB GEARS = 4

GEAR INTO	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
UPSHIFTS	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
DOWNSHIFTS	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

VEHICLE PERFORMANCE SIMULATION
****

RUN TITLE = TEST CASES OF VEHSIM

SCHEDULE AVERAGES
FUEL ECONOMY = 3.51 MPG
WORK PER MILE = 4.74 HP-HR/MI
AVG SP FUEL CONS = 0.42 LBS/HP-HR
AVG SPEED = 12.7 MPH

ADDITIONAL RUN DATA

DRIVING SCHEDULE NAME = TRABUSCBDA		ROUTE NAME	= ZGRADE
VEHICLE NAME = BUS-1		ENGINE NAME	= 871N60
CONVERTER NAME = TC-490-C		SHIFT LOGIC NAME	= V730-ST
WEIGHT (LBS) = 30000.		STROKE (INCHES)	= 5.00
DISPLACEMENT (CU INL = 567.5		REAR AXLE RATIO	= 5.35 0.00 0.00
WIND VELOCITY (MPH = 0.0		FUEL DENSITY (LB/GAL)	= 7.04
AERO DRAG = 75.00 , 0.60		TIRES = 10.00 , 0.00 , 0.40 , 1.00	

TOTALS

VARIABLE (UNITS)	TOTAL AMOUNT	PERCENT OF TOTAL (CRUISE	ACCEL	DECEL	IDLE)
TIME (SECS)	40.5	40.9	23.0	18.3	17.8
DISTANCE (MILES)	0.1	64.4	19.9	15.7	0.0
ENERGY (HP-HR)	0.68	26.4	66.7	4.2	2.7
FUEL (LBS)	0.29	29.0	61.7	5.3	5.0

ENERGY SUPPLY

	HP-HR
(1) ENGINE =	0.68
(2) KINETIC ENERGY =	0.00
(3) POTENTIAL ENERGY =	0.00
(4) ROTATING INERTIA =	0.00

BREAKDOWN

	PERCENT ENGINE HP-HR
(1) ACCESSORIES =	27.72
(2) TORQUE CONVERTER =	10.94
(3) CLUTCH =	0.00
(4) GEAR BOX =	1.71
(5) DIFFERENTIAL =	1.47
(6) TIRE SLIP =	0.00
3+4+5+6 =	3.18
2+3+4+5+6 =	14.12
(7) AERODYNAMIC DRAG =	2.07
(8) ROLLING RESIST =	16.73
SUBTOTAL 1- 8 =	61.86
(9) BRAKES =	35.65
(10) ENGINE MOTORING =	2.49
SUBTOTAL 1-10 =	100.00
(11) OTHER ENERGY =	0.00
TOTAL 1-11 =	100.00

VEHICLE PERFORMANCE SIMULATION
****

BREAKDOWN OF % TIME SPENT ON VARIOUS PARTS OF ENGINE MAP

	-200.0	-150.0	-100.0	-50.0	0.0	50.0	100.0	150.0	200.0	250.0	300.0
500.	0.	-118.	0.	0.	0.	87.	0.	0.	0.	0.	
	0.	552.	0.	0.	0.	550.	0.	0.	0.	0.	
	0.	7.	0.	0.	0.	22.	0.	0.	0.	0.	
600.	0.	-117.	0.	0.	0.	88.	0.	0.	0.	0.	
	0.	648.	0.	0.	0.	650.	0.	0.	0.	0.	
	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	
700.	0.	-115.	0.	0.	0.	89.	0.	0.	0.	0.	
	0.	750.	0.	0.	0.	750.	0.	0.	0.	0.	
	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	
800.	0.	-113.	0.	0.	0.	89.	0.	0.	0.	0.	
	0.	851.	0.	0.	0.	850.	0.	0.	0.	0.	
	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	
900.	0.	-112.	0.	0.	0.	88.	0.	0.	209.	0.	
	0.	941.	0.	0.	0.	950.	0.	0.	975.	0.	
	0.	1.	0.	0.	0.	0.	0.	0.	41.	0.	
1000.	0.	-110.	0.	0.	0.	88.	0.	0.	0.	0.	
	0.	1040.	0.	0.	0.	1050.	0.	0.	0.	0.	
	0.	1.	0.	0.	0.	0.	0.	0.	0.	0.	
1100.	-179.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	1148.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	2.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
1200.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
1300.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
1400.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
1500.											

	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
1600.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
1700.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0	0.	0.	0.	0.	0.	0
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
1800.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
1900.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2000.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2100.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2200.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2300.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2400.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0	0.	0	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2500.										

BREAKDOWN OF % TIME SPENT ON VARIOUS PARTS OF ENGINE MAP

	300.0	350.0	400.0	450.0	500.0	550.0	600.0	650.0	700.0	750.0	800.0
500.											
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
600.											
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
700.											
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
800.											
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	
900.											
	539.	586.	423.	472.	0.	0.	0.	0.	0.	776.	
	875.	548.	969.	991.	0.	0.	0.	0.	0.	974.	
	0.	2.	2.	0.	0.	0.	0.	0.	0.	0.	
1000.											
	0.	0.	0.	970.	500.	0.	0.	0.	0.	0.	
	0.	0.	0.	1044.	1074.	0.	0.	0.	0.	0.	
	0.	0.	0.	2.	0.	0.	0.	0.	0.	0.	
1100.											
	0.	0.	0.	0.	0.	0.	0.	0.	0.	771.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	1153.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	
1200.											
	0.	0.	0.	0.	0.	0.	0.	0.	0.	766.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	1250.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	1.	
1300.											
	0.	0.	0.	0.	0.	0.	0.	0.	0.	754.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	1355.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	2.	
1400.											
	0.	0.	0.	0.	0.	0.	0.	0.	0.	760.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	1456.	
	0.	0.	0.	0.	0.	0.	0.	0.	0.	2.	
1500.											

	0.	0.	0.	0.	0.	0.	0.	0.	0.	757.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	1549.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	4.
1600.										
	0.	0.	0.	0.	0.	0.	0.	0.	746.	754.
	0.	0.	0.	0.	0.	0.	0.	0.	1677.	1629.
	0.	0.	0.	0.	0.	0.	0.	0.	2.	2.
1700.										
	0.	0.	0.	0.	0.	0.	0.	0.	741.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	1707.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
1800.										
	0.	0.	0.	0.	0.	0.	0.	0.	C.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	C.	0.
1900.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2000.										
	0.	0.	0.	0.	0.	0.	0.	0.	C.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	C.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2100.										
	0.	0.	0.	0.	0.	0.	0.	0.	C.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2200.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2300.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2400.										
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
2500.										

END OF VEHSIM CONTROL FILE DSK :TEST.SEV [03224,6331

LPT FILE DSK :T2KBVG.DAT[00003,0031<100>/DISPOSE:SAVE

```
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     222222222     000000000
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     222222222     000000000
FFFFFFFFFFFFFFF     000000000     RRRRRRRRRRRR     222222222     000000000
FFF              000         000  RRR         RRR  222         222  000         000
FFF              000         000  RRR         RRR  222         222  000         000
FFF              000         000  RRR         RRR  222         222  000         000
FFF              000         000  RRR         RRR       222    222  000      000000
FFF              000         000  RRR         RRR       222    222  000      000000
FFF              000         000  RRR         RRR       222    222  000      000000
FFFFFFFFFFFF     000         000  RRRRRRRRRRRR            222    000  000      000
FFFFFFFFFFFF     000         000  RRRRRRRRRRRR            222    000  000      000
FFFFFFFFFFFF     000         000  RRRRRRRRRRRR            222    000  000      000
FFF              000         000  RRR     RRR          222       000000       000
FFF              000         000  RRR     RRR          222       000000       000
FFF              000         000  RRR     RRR          222       000000       000
FFF              000         000  RRR         RRR  222         000         000
FFF              000         000  RRR         RRR  222         000         000
FFF              000         000  RRR         RRR  222         000         000
FFF                 000000000     RRR         RRR  222222222222222     000000000
FFF                 000000000     RRR         RRR  222222222222222     000000000
FFF                 000000000     RRR         RRR  222222222222222     000000000
```



```
 *START* User SOMERS [3224,633]  Job  FOR20 Seq. 7412 Date 04-Sep-80 11:48:09 Monitor TSC DECsystem10/KL v4 *START*
File: DSKB:Q84S1.LPT<077>[3,3](FOR20) Created: 04-Sep-80 09130:43 Printed: 04-Sep-80 11:48:15
QUEUE Switches:  /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:1704 /FORMS:FORNOR

File: DSKB:Q84S1.LPT<077>[3,3](FOR20) will be DELETED after printing
```

Lengend - FOR20

NAME - name of routine (see below) where data originates

NUM - location in routine where data originates

I - segment type

M - engine map location

T - time (sec)

TOLD - time (sec) during previous time step

DT - time step size (sec)

D - incremental distance (miles)

V - vehicle velocity (miles/hour)

ACCEL - vehicle acceleration (ft/sec^2)

TORQE - engine torque (lb-ft)

RPME - engine speed (rev/min)

TORQA - accessory torque (lb-ft)

TORQ1 - torque at input to torque converter

RPM1 - Speed at input to torque converter

TORQP - propshaft torque (lb-ft)

RPMP - propshaft speed (rev/min)

BRAKES - Brake force (lb)

Routine "Name" abbreviations

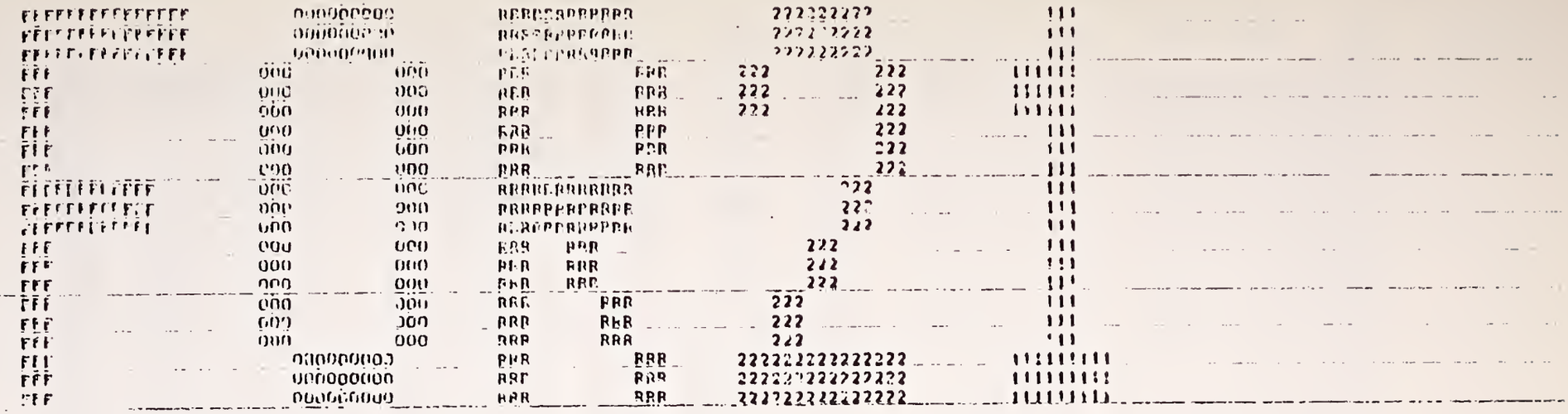
GOBAK - GOBACK

ITERA - ITERAT

SIMCT - SIMCTR

SHIFT - SHIFTS

NAME	NUM	I	M	T	TOLD	DT	D	V	VOLD	ACCEL	TORQE	RPME	TORQA	TORQ1	RPM1	TORQP	RPMP	PAGE 153 BRAKES
SIMCT	21	3	1	3.838	3.788	0.050	1.110	15.201	15.084	3.447	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
SIMCT	22	3	1	3.838	3.788	0.050	1.110	15.201	15.084	3.447	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
SHIFT	1	3	1	3.838	3.788	0.050	1.110	15.201	15.084	3.447	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
SIMCT	23	3	1	3.838	3.788	0.050	1.110	15.201	15.084	3.447	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
SIMCT	52	3	1	3.838	3.788	0.050	1.110	15.201	15.084	3.447	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
SIMCT	1	3	1	3.838	3.788	0.050	1.110	15.201	15.084	3.447	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
SIMCT	20	3	1	3.888	3.838	0.050	1.110	15.201	15.201	3.447	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
ITERA	1	3	1	3.888	3.838	0.050	1.110	15.201	15.201	3.447	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
GOBAK	1	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
GOBAK	2	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	752.68	1584.03	109.60	528.20	1810.32	1200.33	700.66	0.00
GOBAK	4	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	752.68	1584.03	109.60	528.20	1810.32	1200.33	705.48	0.00
GOBAK	5	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	752.68	1584.03	109.60	528.20	1810.32	1079.13	705.48	0.00
GOBAK	6	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	752.68	1584.03	109.60	528.20	1810.32	1079.13	705.48	0.00
GOBAK	7	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	752.68	1542.77	109.60	489.43	1763.17	1079.13	705.48	0.00
GOBAK	8	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	752.68	1542.77	109.60	489.43	1763.17	1079.13	705.48	0.00
GOBAK	9	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	752.68	1542.77	108.57	489.43	1763.17	1079.13	705.48	0.00
GOBAK	10	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	280.16	1542.77	108.57	489.43	1763.17	1079.13	705.48	0.00
ITERA	2	3	1	3.888	3.838	0.050	1.110	15.306	15.201	3.065	280.16	1542.77	108.57	489.43	1763.17	1079.13	705.48	0.00
GOBAK	1	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	280.16	1542.77	108.57	489.43	1763.17	1079.13	705.48	0.00
GOBAK	2	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	280.16	1542.77	108.57	489.43	1763.17	1079.13	705.48	0.00
GOBAK	4	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	280.16	1542.77	108.57	489.43	1763.17	1079.13	705.79	0.00
GOBAK	5	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	280.16	1542.77	108.57	489.43	1763.17	1142.68	705.79	0.00
GOBAK	6	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	280.16	1542.77	108.57	489.43	1763.17	1142.68	705.79	0.00
GOBAK	7	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	280.16	1565.99	108.57	511.51	1789.70	1142.68	705.79	0.00
GOBAK	8	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	280.16	1565.99	108.57	511.51	1789.70	1142.68	705.79	0.00
GOBAK	9	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	280.16	1565.99	109.15	511.51	1789.70	1142.68	705.79	0.00
GOBAK	10	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	531.10	1565.99	109.15	511.51	1789.70	1142.68	705.79	0.00
ITERA	3	3	1	3.888	3.838	0.050	1.110	15.313	15.201	3.265	531.10	1565.99	109.15	511.51	1789.70	1142.68	705.79	0.00
GOBAK	1	3	1	3.888	3.838	0.050	1.110	15.319	15.201	3.465	531.10	1565.99	109.15	511.51	1789.70	1142.68	705.79	0.00
GOBAK	2	3	1	3.888	3.838	0.050	1.110	15.319	15.201	3.465	531.10	1565.99	109.15	511.51	1789.70	1142.68	705.79	0.00
GOBAK	4	3	1	3.888	3.838	0.050	1.110	15.319	15.201	3.465	531.10	1565.99	109.15	511.51	1789.70	1142.68	706.11	0.00
GOBAK	5	3	1	3.888	3.838	0.050	1.110	15.319	15.201	3.465	531.10	1565.99	109.15	511.51	1789.70	1206.22	706.11	0.00
GOBAK	6	3	1	3.888	3.838	0.050	1.110	15.319	15.201	3.465	531.10	1565.99	109.15	511.51	1789.70	1206.22	706.11	0.00
GOBAK	7	3	1	3.888	3.838	0.050	1.110	15.319	15.201	3.465	531.10	1591.39	109.15	532.26	1818.73	1206.22	706.11	0.00
GOBAK	8	3	1	3.888	3.838	0.050	1.110	15.319	15.201	3.465	531.10	1591.39	109.15	532.26	1818.73	1206.22	706.11	0.00
GOBAK	9	3	1	3.888	3.838	0.050	1.110	15.319	15.201	3.465	531.10	1591.39	109.78	532.26	1818.73	1206.22	706.11	0.00
GOBAK	10	3	5	3.888	3.838	0.050	1.110	15.319	15.201	3.465	801.73	1591.39	109.78	532.26	1818.73	1206.22	706.11	0.00
ITERA	3	3	5	3.888	3.838	0.050	1.110	15.319	15.201	3.465	801.73	1591.39	109.78	532.26	1818.73	1206.22	706.11	0.00
GOBAK	1	3	5	3.888	3.838	0.050	1.110	15.316	15.201	3.365	801.73	1591.39	109.78	532.26	1818.73	1206.22	706.11	0.00
GOBAK	2	3	5	3.888	3.838	0.050	1.110	15.316	15.201	3.365	801.73	1591.39	109.78	532.26	1818.73	1206.22	706.11	0.00
GOBAK	4	3	5	3.888	3.838	0.050	1.110	15.316	15.201	3.365	801.73	1591.39	109.78	532.26	1818.73	1206.22	705.95	0.00
GOBAK	5	3	5	3.888	3.838	0.050	1.110	15.316	15.201	3.365	801.73	1591.39	109.78	532.26	1818.73	1174.45	705.95	0.00
GOBAK	6	3	5	3.888	3.838	0.050	1.110	15.316	15.201	3.365	801.73	1591.39	109.78	532.26	1818.73	1174.45	705.95	0.00
GOBAK	7	3	5	3.888	3.838	0.050	1.110	15.316	15.201	3.365	801.73	1578.84	109.78	521.88	1804.39	1174.45	705.95	0.00
GOBAK	8	3	5	3.888	3.838	0.050	1.110	15.316	15.201	3.365	801.73	1578.84	109.78	521.88	1804.39	1174.45	705.95	0.00
GOBAK	9	3	5	3.888	3.838	0.050	1.110	15.316	15.201	3.365	801.73	1578.84	109.47	521.88	1804.39	1174.45	705.95	0.00
GOBAK	10	3	1	3.888	3.838	0.050	1.110	15.316	15.201	3.365	667.88	1578.84	109.47	521.88	1804.39	1174.45	705.95	0.00
ITERA	3	3	1	3.888	3.838	0.050	1.110	15.316	15.201	3.365	667.88	1578.84	109.47	521.88	1804.39	1174.45	705.95	0.00
GOBAK	1	3	1	3.888	3.838	0.050	1.110	15.318	15.201	3.415	667.88	1578.84	109.47	521.88	1804.39	1174.45	705.95	0.00
GOBAK	2	3	1	3.888	3.838	0.050	1.110	15.318	15.201	3.415	667.88	1578.84	109.47	521.88	1804.39	1174.45	705.95	0.00
GOBAK	4	3	1	3.888	3.838	0.050	1.110	15.318	15.201	3.415	667.88	1578.84	109.47	521.88	1804.39	1174.45	706.03	0.00
GOBAK	5	3	1	3.888	3.838	0.050	1.110	15.318	15.201	3.415	667.88	1578.84	109.47	521.88	1804.39	1190.34	706.03	0.00



START User SOMERS [3,24,0,0] Job FOP21 Seq. 7413 Date 04-Sep-80 11:21:39 Monitor [SC DECsystem10/T1 V4 *START*
File: DSKB:[108551,GPT<077>]3;3](FOP21) Created: 04-Sep-80 09:10:43 Printed: 04-Sep-80 11:21:19
QUEUE S /ACCESS: /FILE:(ASC) /COPIES:1 /SPACING:1 /LIMIT:1519 /FORMS:FOT,NOP

File: DSKB:[108551,GPT<077>]3;3](FOP21) will be DELETED after printing

LEGEND-FOR21

RPMC - input speed to gears (rev/min)

SR - speed ratio-torque converter

TR - torque ratio

TORQW - torque at the wheels (lb-ft)

RPMW - speed at the wheels (rev/min)

TRR - torque at the rear end (lb-ft)

TORQF - torque at the front end (lb-ft)

TORQ - torque from the engine map (lb-ft)

FWHEEL - wheel force (lbs)

FAERO - aerodynamic force (lbs)

FACCEL - acceleration force (lbs)

FROLL - rolling resistance force (lbs)

RPM2 - torque converter output speed (rev/min)

```
   T       RPMC    SR    TR     TORQW  RPMW     TRR     TORQF     TORQ   FWHEEL  FAERO    FACCEL FRULL   RPM2

  3.838  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.838  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.838  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.838  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.838  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.838  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.888  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.888  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.888  1240.170 0.685 1.315  6257.09 130.96  484.715   27.10   752.68  3552.231 26.171  3214.060 312.001240.170
  3.888  1240.170 0.685 1.315  5625.33 131.86  431.013   27.10   752.68  3196.504 26.532  2857.973 312.001240.170
  3.888  1248.694 0.685 1.315  5625.33 131.86  431.013   27.10   752.68  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.685 1.315  5625.33 131.86  431.013   27.10   752.68  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.708 1.277  5625.33 131.86  431.013   27.10   752.68  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.708 1.277  5625.33 131.86  431.013   27.10   752.68  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.708 1.277  5625.33 131.86  431.013   27.10   752.68  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.708 1.277  5625.33 131.86  431.013 -399.18   752.68  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.708 1.277  5625.33 131.86  431.013 -399.18   280.16  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.708 1.277  5625.33 131.86  431.013 -399.18   280.16  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.708 1.277  5625.33 131.86  431.013 -399.18   280.16  3196.504 26.532  2857.973 312.001248.694
  3.888  1248.694 0.708 1.277  5956.58 131.92  459.141 -399.18   280.16  3383.037 26.556  3044.482 312.001248.694
  3.888  1249.250 0.708 1.277  5956.58 131.92  459.141 -399.18   280.16  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.708 1.277  5956.58 131.92  459.141 -399.18   280.16  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.698 1.293  5956.58 131.92  459.141 -399.18   280.16  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.698 1.293  5956.58 131.92  459.141 -399.18   280.16  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.698 1.293  5956.58 131.92  459.141 -399.18   280.16  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.698 1.293  5956.58 131.92  459.141 -174.57   280.16  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.698 1.293  5956.58 131.92  459.141 -174.57   531.10  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.698 1.293  5956.58 131.92  459.141 -174.57   531.10  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.698 1.293  5956.58 131.92  459.141 -174.57   531.10  3383.037 26.556  3044.482 312.001249.250
  3.888  1249.250 0.698 1.293  6287.82 131.98  487.268 -174.57   531.10  3569.570 26.579  3230.991 312.001249.250
  3.888  1249.806 0.698 1.293  6287.82 131.98  487.268 -174.57   531.10  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.698 1.293  6287.82 131.98  487.268 -174.57   531.10  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.687 1.312  6287.82 131.98  487.268 -174.57   531.10  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.687 1.312  6287.82 131.98  487.268 -174.57   531.10  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.687 1.312  6287.82 131.98  487.268 -174.57   531.10  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.687 1.312  6287.82 131.98  487.268   71.23   531.10  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.687 1.312  6287.82 131.98  487.268   71.23   755.26  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.687 1.312  6287.82 131.98  487.268   71.23   755.26  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.687 1.312  6287.82 131.98  487.268   71.23   755.26  3569.570 26.579  3230.991 312.001249.806
  3.888  1249.806 0.687 1.312  6122.20 131.95  473.205   71.23   755.26  3476.304 26.567  3137.736 312.001249.806
  3.888  1249.528 0.687 1.312  6122.20 131.95  473.205   71.23   755.26  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.687 1.312  6122.20 131.95  473.205   71.23   755.26  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.692 1.303  6122.20 131.95  473.205   71.23   755.26  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.692 1.303  6122.20 131.95  473.205   71.23   755.26  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.692 1.303  6122.20 131.95  473.205   71.23   755.26  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.692 1.303  6122.20 131.95  473.205  -50.20   755.26  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.692 1.303  6122.20 131.95  473.205  -50.20   667.88  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.692 1.303  6122.20 131.95  473.205  -50.20   667.88  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.692 1.303  6122.20 131.95  473.205  -50.20   667.88  3476.304 26.567  3137.736 312.001249.528
  3.888  1249.528 0.692 1.303  6205.01 131.97  480.237  -50.20   667.88  3522.937 26.573  3184.364 312.001249.528
  3.888  1249.667 0.692 1.303  6205.01 131.97  480.237  -50.20   667.88  3522.937 26.573  3184.364 312.001249.667
  3.888  1249.667 0.692 1.303  6205.01 131.97  480.237  -50.20   667.88  3522.937 26.573  3184.364 312.001249.667
```

VEHICLE PERFORMANCE SIMULATION

RUN TITLE (TEST CASES OF VEHSIM)

DRIVING SCHEDULE (TRANSCRD4) USING ROUTE (ZGRADE)

SEC.	MILES	MPH	ACC	INST MPG	RSEC	CUM MPG	GEAR	RDTE	HPW	HPE	FEM	TCEQ	VAC	SE	ETA	%ACT	EBS SPD	EBARIS	CFEFTI
0.00	0.000	0.0	0.00	0.00	0.78	0.0	1 1	0.0	0.0	9.1	550.	67.	0.0	0.000	0.000	27.	1	507.0	0.00

$SIMCTR-T,DT,V,RPM2,RPMC,HP2,HPCT,AICSSI,TCEQ2 ->
0.065 0.035 0.000 3.029 2.781 0.000 0.000 0.000 0.000

$SIMCTR - T,DT,V,CKE,CRCT,CFC,CGB,CCI,CDIF, CTIRE,CER,CEV,CAE,TEMTCT,TCTEN ,CTHEF->
.8507046E-01 .3507046E-01 .0000000E+00 .0000000E+00-.3039852E+04 .0000000E+00 .0000000E+00 .0000000E+00
.0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .1632252E-02 .1632252E-02 .1535125E-02

$SIMCTR-T,DT,V,RPM2,RPMC,HP2,HPCT,AICSSI,TCEQ2 ->
0.126 0.041 0.000 3.495 2.781 0.000 0.000 0.000 0.000

$SIMCTR - T,DT,V,CKE,CRCT,CFC,CGB,CCI,CDIF, CTIRE,CER,CEV,CAE,TEMTCT,TCTEN ,CTHEF->
.1261945E+00 .4113403E-01 .0000000E+00 .0000000E+00-.6586307E+04 .0000000E+00 .0000000E+00 .0000000E+00
.0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .2021682E-02 .2021682E-02 .1900980E-02

0.08	0.000	0.0	0.00	0.00	0.72	0.0	1 1	0.0	0.0	12.5	750.	88.	0.0	0.004	0.010	27.	1	507.0	0.00

$SIMCTR-T,DT,V,RPM2,RPMC,HP2,HPCT,AICSSI,TCEQ2 ->
0.165 0.039 0.000 3.961 2.781 0.000 0.000 0.000 0.000

$SIMCTR - T,DT,V,CKE,CRCT,CFC,CGB,CCI,CDIF, CTIRE,CER,CEV,CAE,TEMTCT,TCTEN ,CTHEF->
.1652984E+00 .3910390E-01 .0000000E+00 .0000000E+00-.1063946E+05 .0000000E+00 .0000000E+00 .0000000E+00
.0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .2422656E-02 .2422656E-02 .2046634E-02

$SIMCTR-T,DT,V,RPM2,RPMC,HP2,HPCT,AICSSI,TCEQ2 ->
0.203 0.037 0.000 4.427 2.781 0.000 0.000 0.000 0.000

$SIMCTR - T,DT,V,CKE,CRCT,CFC,CGB,CCI,CDIF, CTIRE,CER,CEV,CAE,TEMTCT,TCTEN ,CTHEF->
.2025713E+00 .3727203E-01 .0000000E+00 .0000000E+00-.1519526E+05 .0000000E+00 .0000000E+00 .0000000E+00
.0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .2838209E-02 .2838209E-02 .2302688E-02

0.13	0.000	0.0	0.00	0.00	0.69	0.0	1 1	0.0	0.0	15.9	950.	68.	0.0	0.004	0.010	27.	1	507.0	0.00

$SIMCTR-T,DT,V,RPM2,RPMC,HP2,HPCT,AICSSI,TCEQ2 ->
0.238 0.036 0.000 4.893 0.000 0.000 0.000 0.000 0.000

$SIMCTR - T,DT,V,CKE,CRCT,CFC,CGB,CCI,CDIF, CTIRE,CER,CEV,CAE,TEMTCT,TCTEN ,CTHEF->
.2381771E+00 .3560572E-01 .0000000E+00 .0000000E+00-.2026508E+05 .0000000E+00 .0000000E+00 .0000000E+00
.0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .0000000E+00 .3255281E-02 .3255281E-02 .2590503E-02

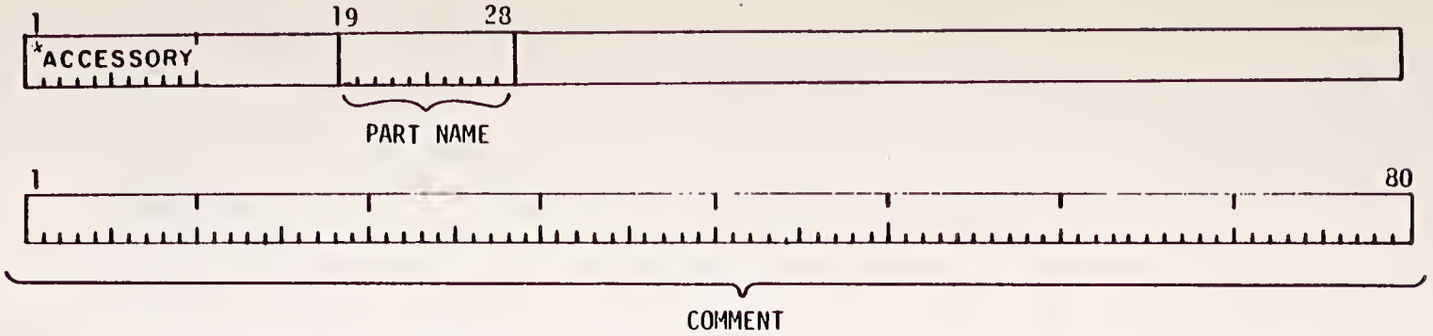
$SIMCTR-T,DT,V,RPM2,RPMC,HP2,HPCT,AICSSI,TCEQ2 ->
0.238 0.036 0.073 5.942 5.942 0.586 0.586 0.000 606.914

$SIMCTR - T,DT,V,CKE,CRCT,CFC,CGB,CCI,CDIF, CTIRE,CER,CEV,CAE,TEMTCT,TCTEN ,CTHEF->

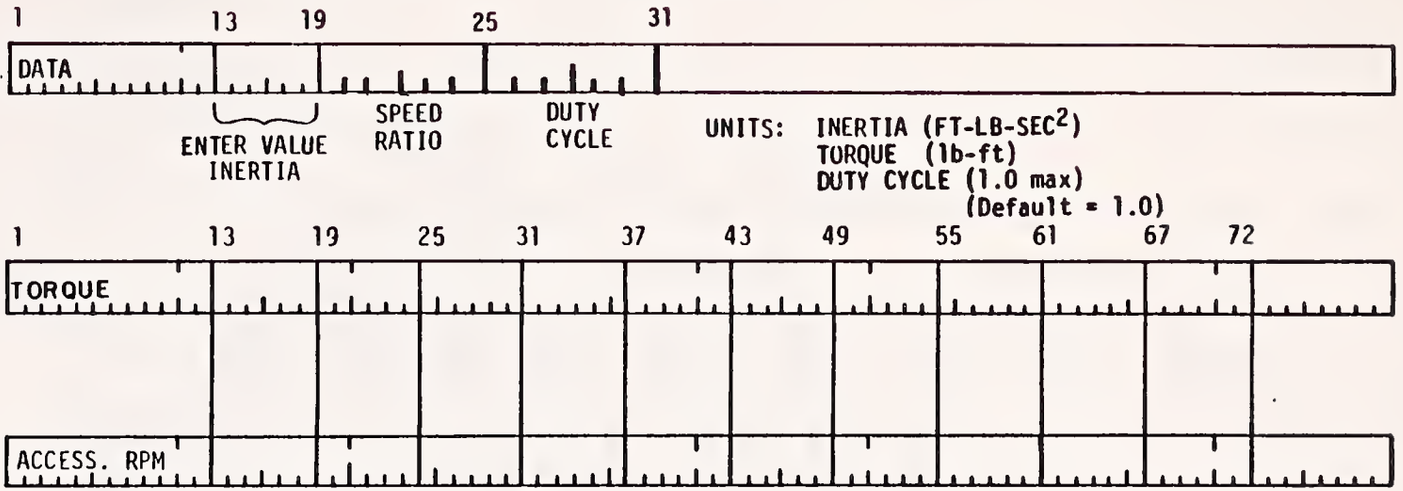
APPENDIX B

Updated Data Sheets

DATA SHEET 1. ACCESSORY

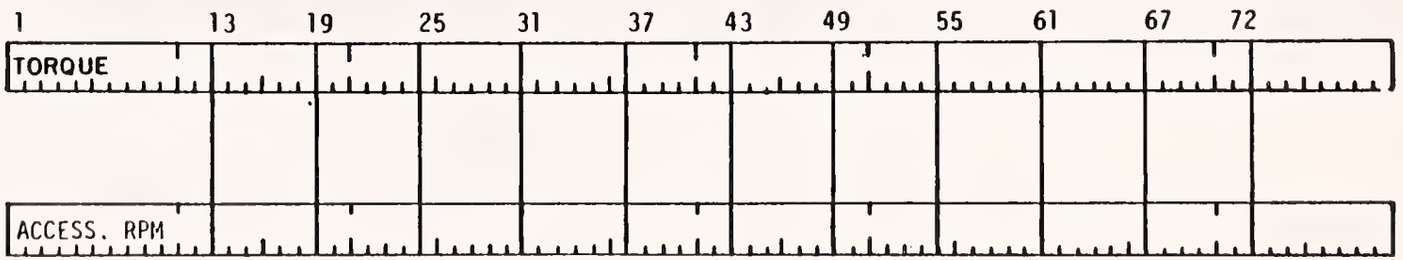


NOTE: ENTER ALL DATA WITH A DECIMAL POINT

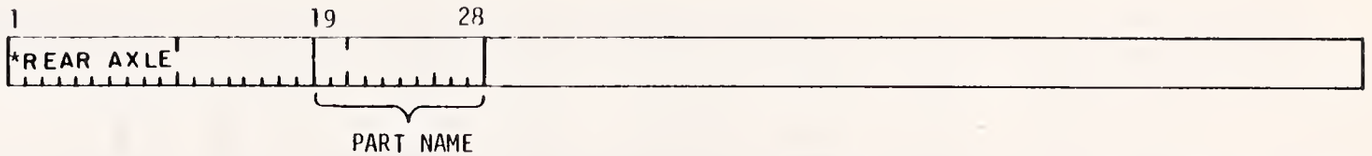


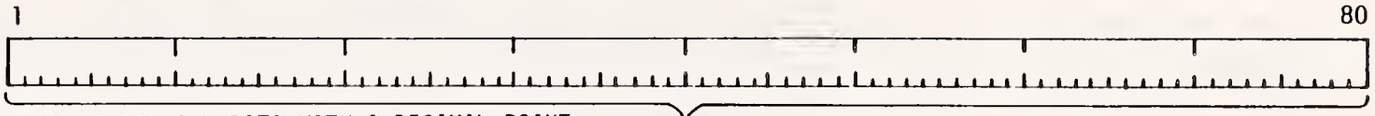
→ INCREASING RPM, FILL SPACES WITH DATA POINTS AS NEEDED UP TO 20 POINTS MAXIMUM

USE ONLY WHEN MORE THAN 10 DATA POINTS TO BE ENTERED

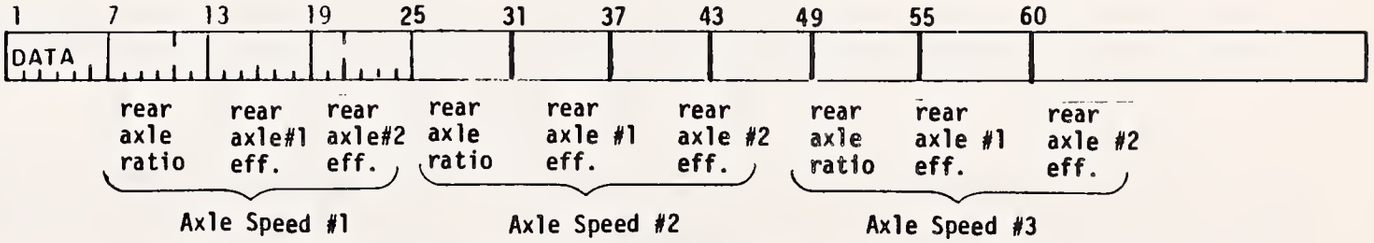


DATA SHEET 2. REAR AXLE (Sheet 1 of 2)

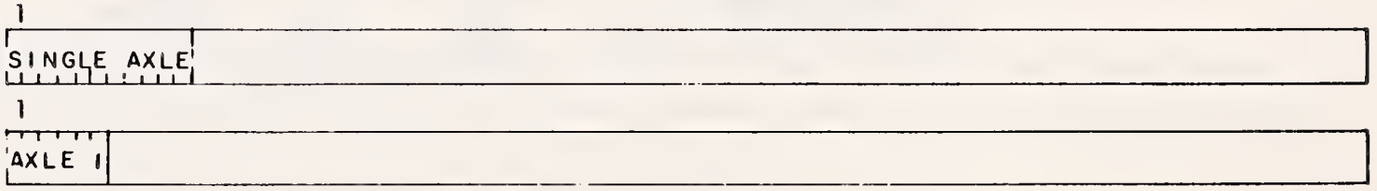




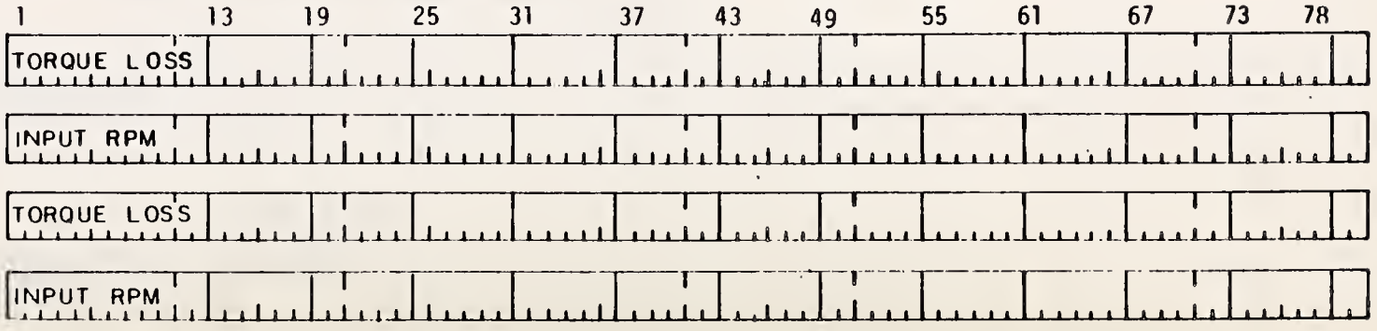
NOTE: ENTER ALL DATA WITH A DECIMAL POINT
AXLE #1 FORWARD-REAR AXLE (or single)
AXLE #2 REARWARD-REAR AXLE



NOTE: USE SINGLE AXLE IF ONLY ONE AXLE. DO NOT USE AXLE 1 OR AXLE 2 TITLES.
USE AXLE 1 AND AXLE 2 IF DOUBLE AXLE.

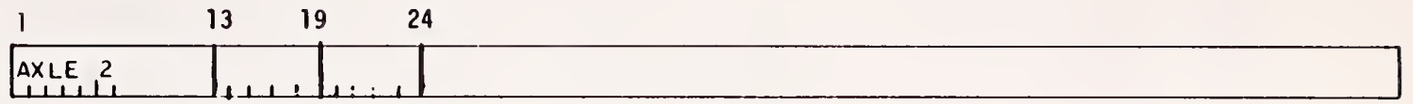


SPIN LOSSES:

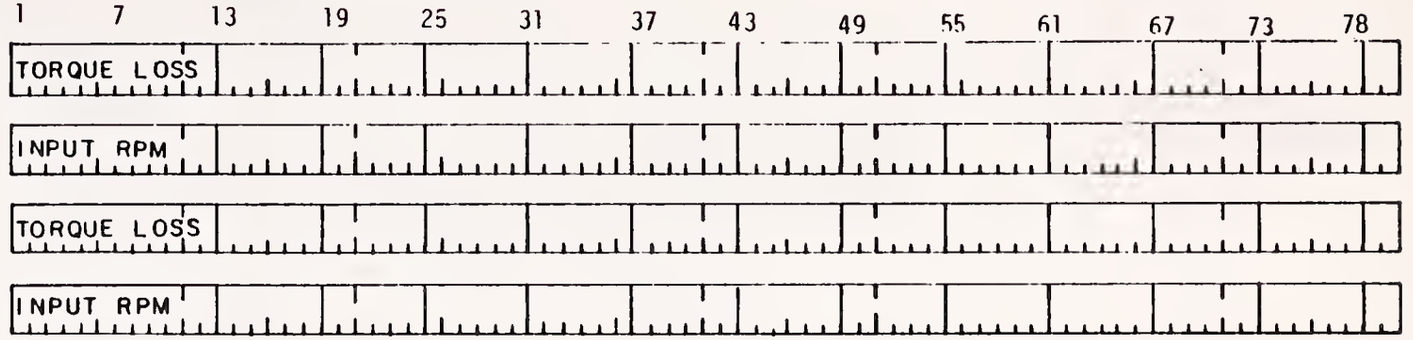


DATA SHEET 2. REAR AXLE (Sheet 2 of 2)

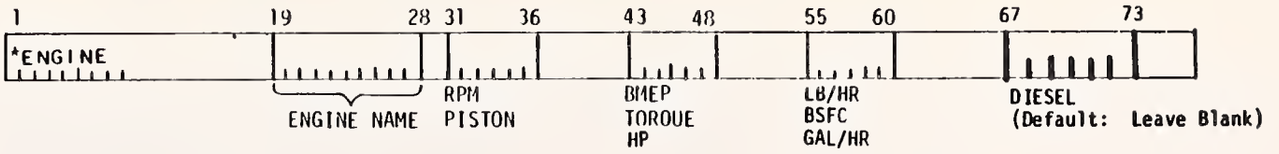
NOTE: ENTER ALL DATA WITH A DECIMAL POINT
TORQUE LOSS IN UNITS OF: LB-FT



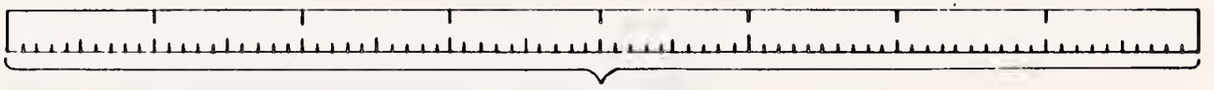
SPIN LOSSES



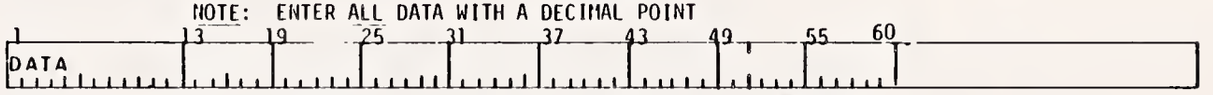
DATA SHEET 3. ENGINE (Sheet 1 of 2)



(ENTER ONE OF ABOVE FOR EACH SET OF UNITS)

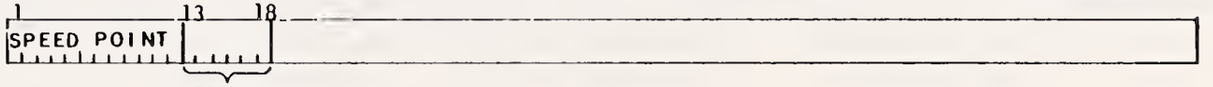


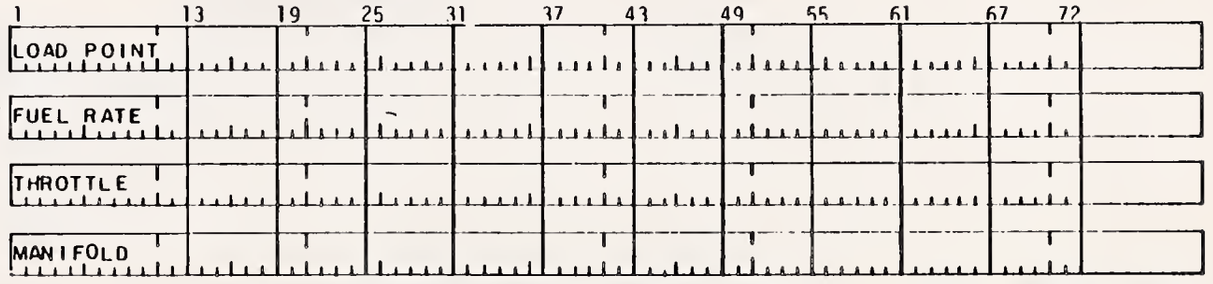
NOTE: ENTER ALL DATA WITH A DECIMAL POINT



*0.764 default if blank
*Use 0.1198 for emissions
**4. default if blank

REPEAT SPEED POINT SETS AS NEEDED UP TO 20 SPEED PTS.





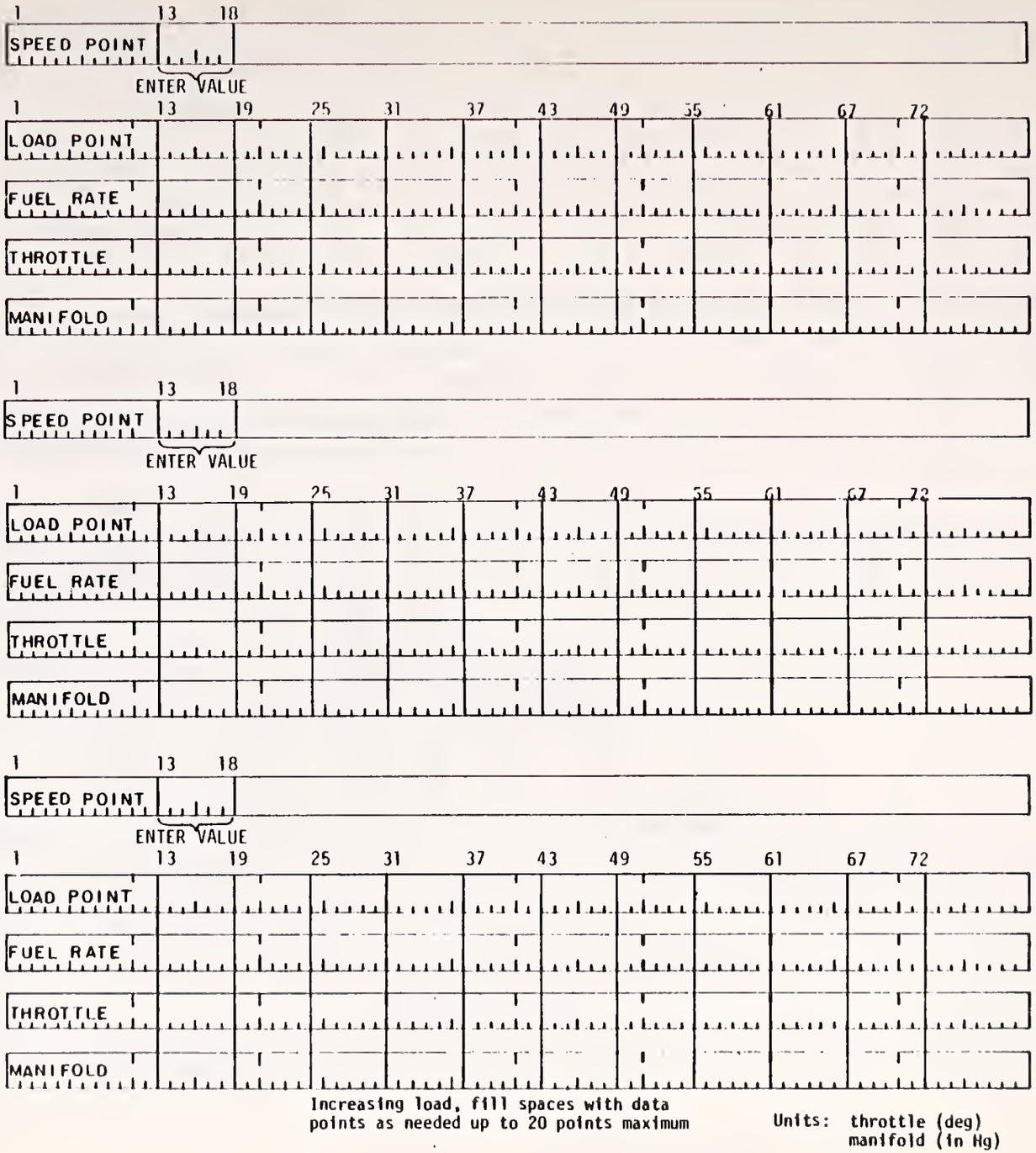
→ INCREASING LOAD, FILL SPACES WITH DATA POINTS AS NEEDED UP TO 20 POINTS MAXIMUM

UNITS: THROTTLE (DEGREE)
MANIFOLD (in. Hg.)

USE ONLY WHEN MORE THAN 10 DATA POINTS TO BE ENTERED

Emission rate (gm/hr) may be entered in FUEL RATE data field. See also *notes.

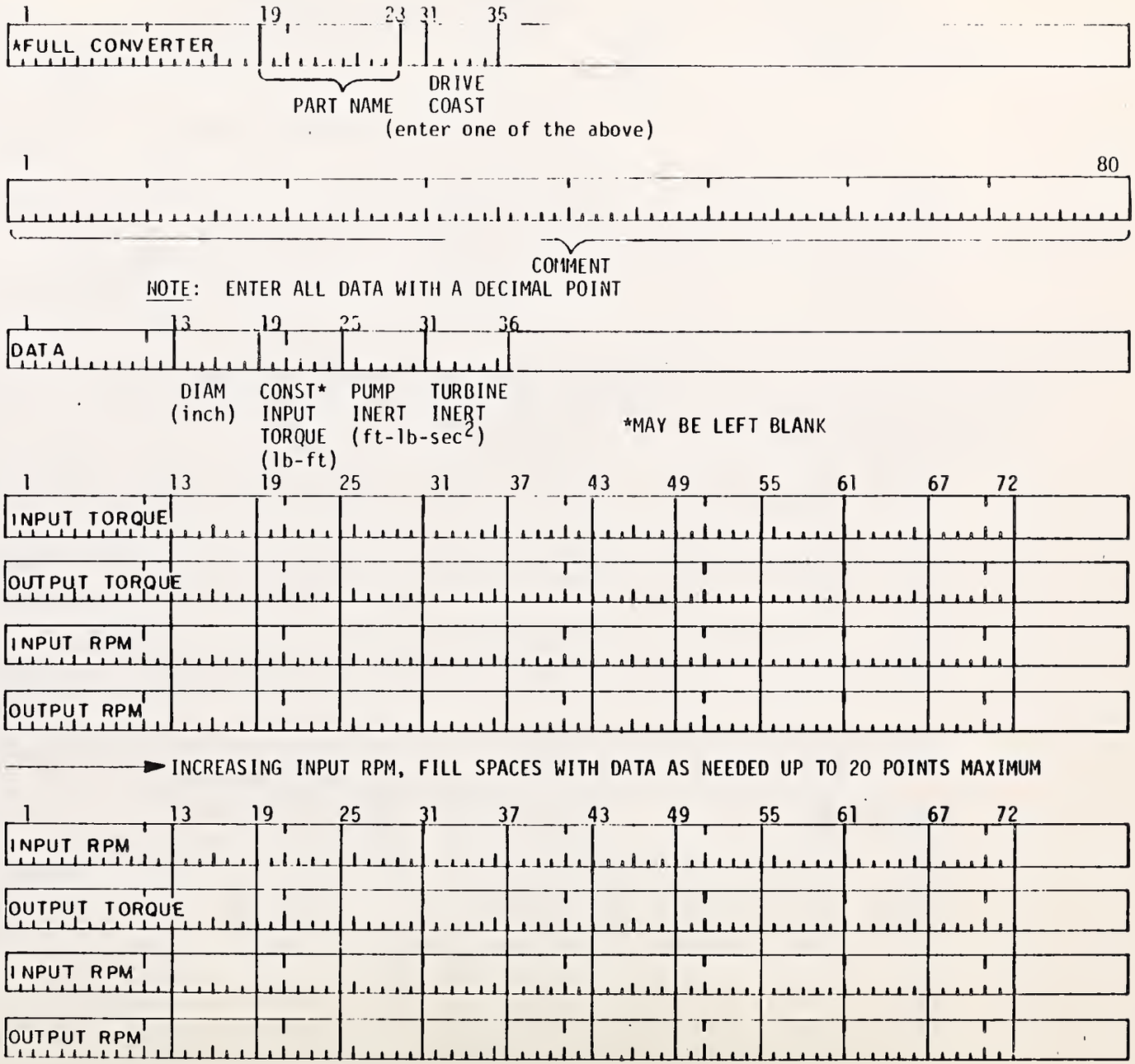
DATA SHEET 3. ENGINE (Sheet 2 of 2)



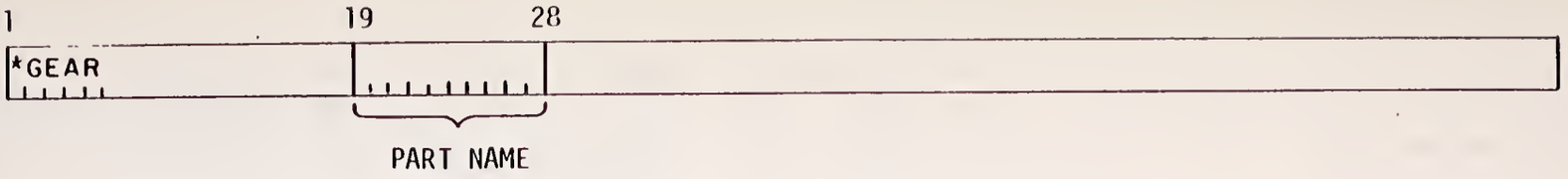
Increasing load, fill spaces with data points as needed up to 20 points maximum

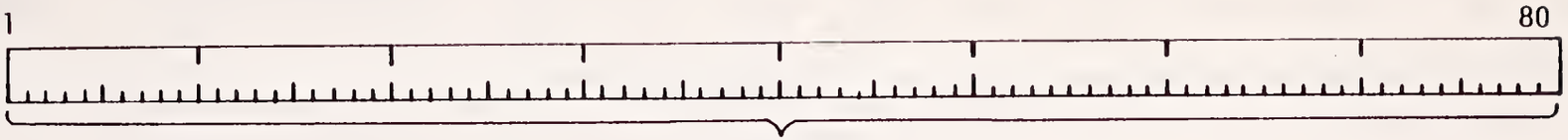
Units: throttle (deg)
manifold (in Hg)

DATA SHEET 4. FULL CONVERTER

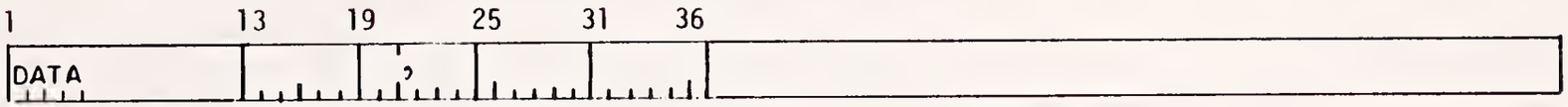


DATA SHEET 5. GEAR





NOTE: ENTER ALL DATA WITH A DECIMAL POINT
TORQUE LOSS IN UNITS OF: LB-FT



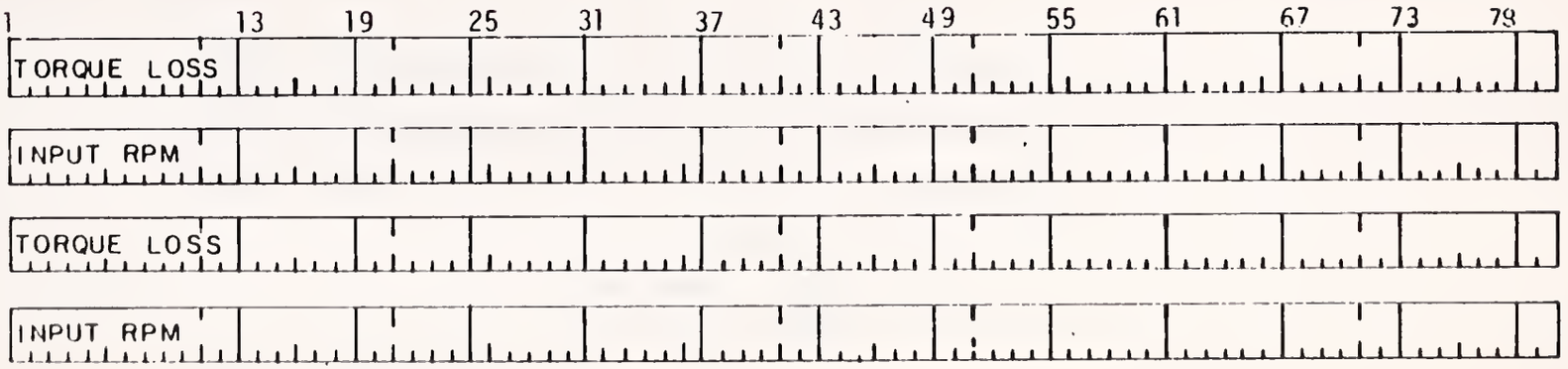
GEAR INPUT INERTIA (ft-lb-sec^2)

GEAR OUTPUT INERTIA

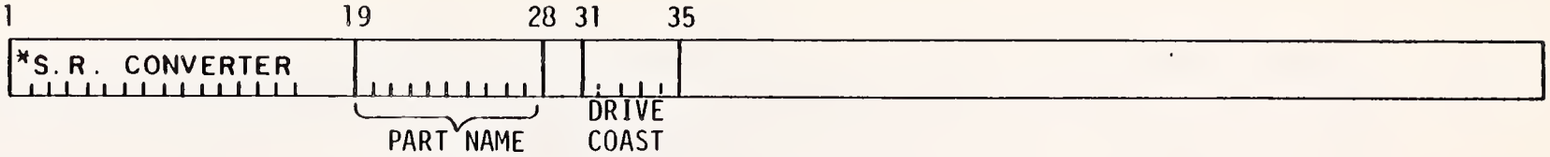
GEAR RATIO

GEAR LOAD EFFICIENCY (1. max)

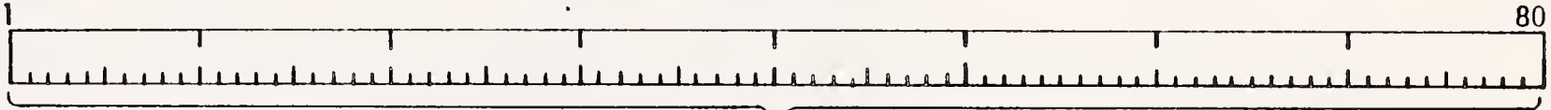
SPIN LOSS



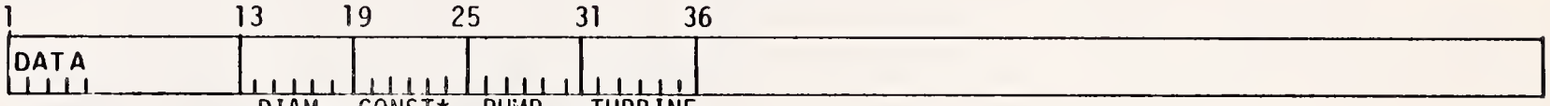
DATA SHEET 6. SPEED RATIO CONVERTER



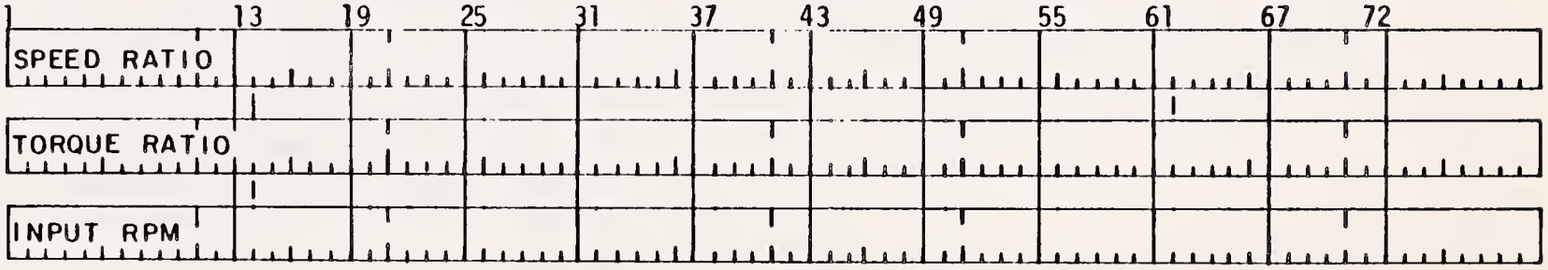
(enter one of the above)



NOTE: ENTER ALL DATA WITH A DECIMAL POINT

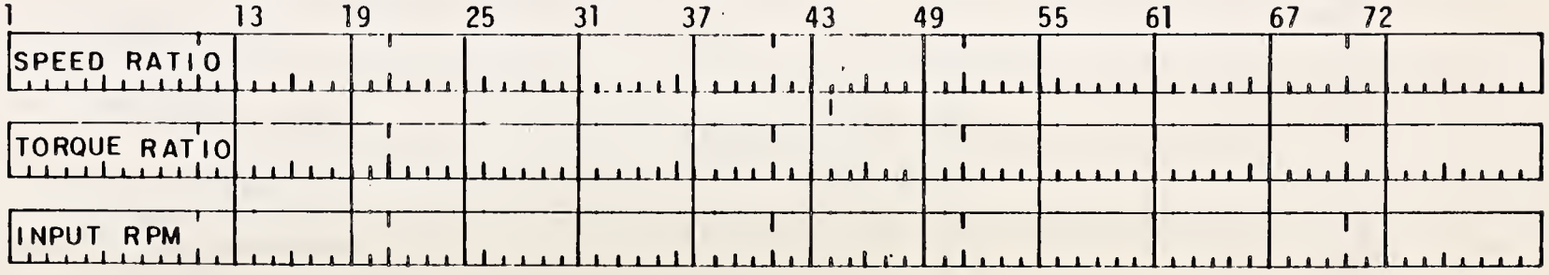


*CONST INPUT TORQUE MUST BE SPECIFIED

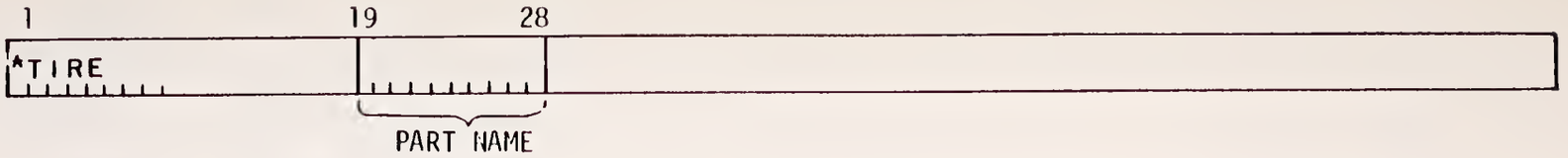


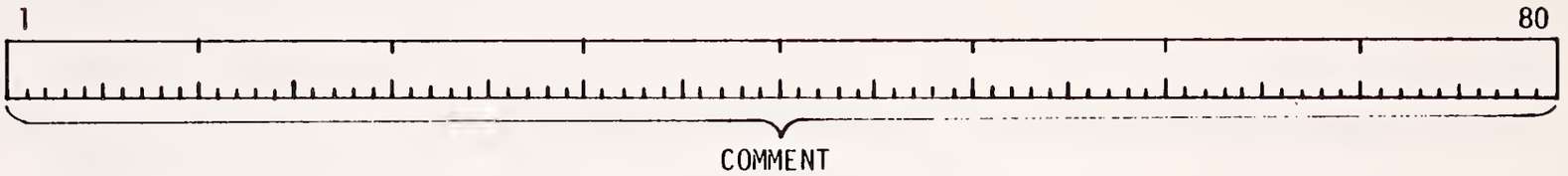
INCREASING RPM, FILL SPACES WITH DATA AS NEEDED UP TO 20 POINTS MAXIMUM

USE ONLY WHEN MORE THAN 10 DATA POINTS TO BE ENTERED

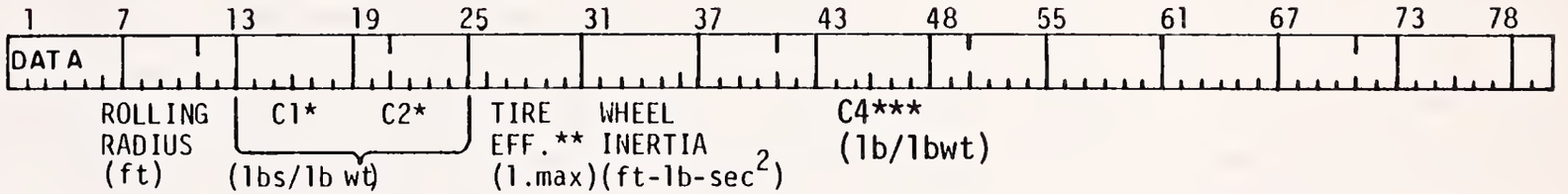


DATA SHEET 7. TIRE





NOTE: ENTER ALL DATA WITH A DECIMAL POINT

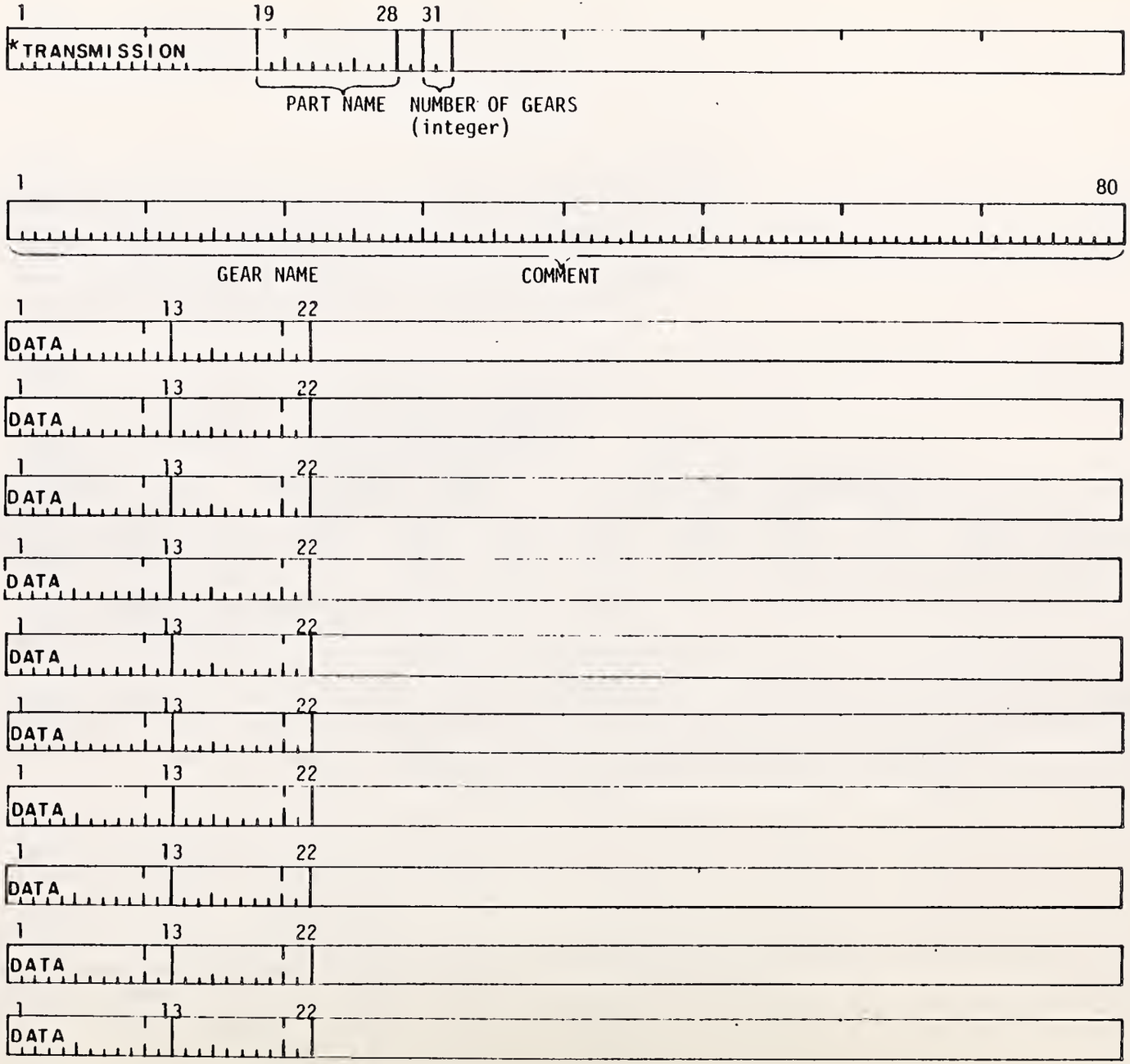


*Rolling resistance coefficients. Make C2 zero for linear approximation.

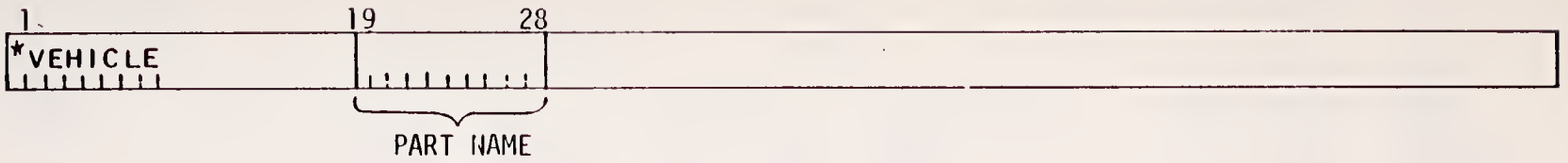
**Tire slip effect due primarily to road surface

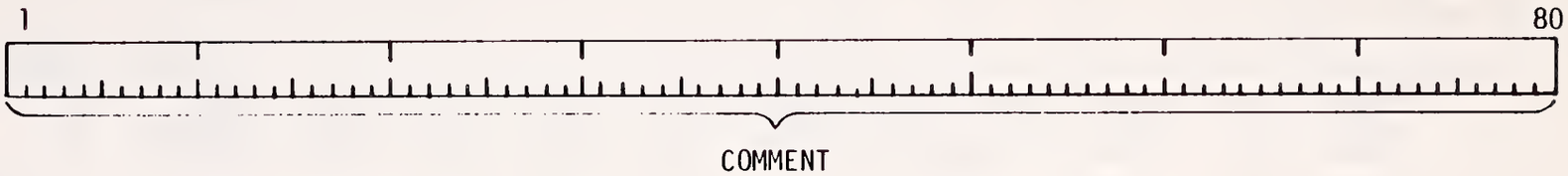
***Bearing Loss Coefficient, default value of 0.0004

DATA SHEET 8. TRANSMISSION

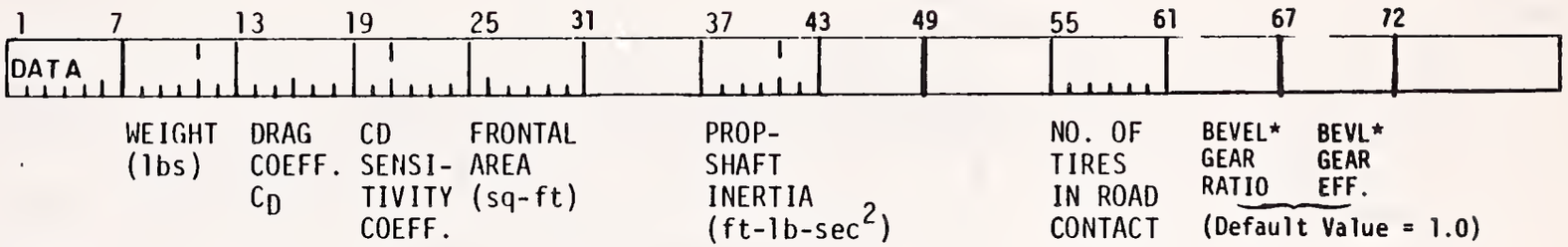


DATA SHEET 9. VEHICLE

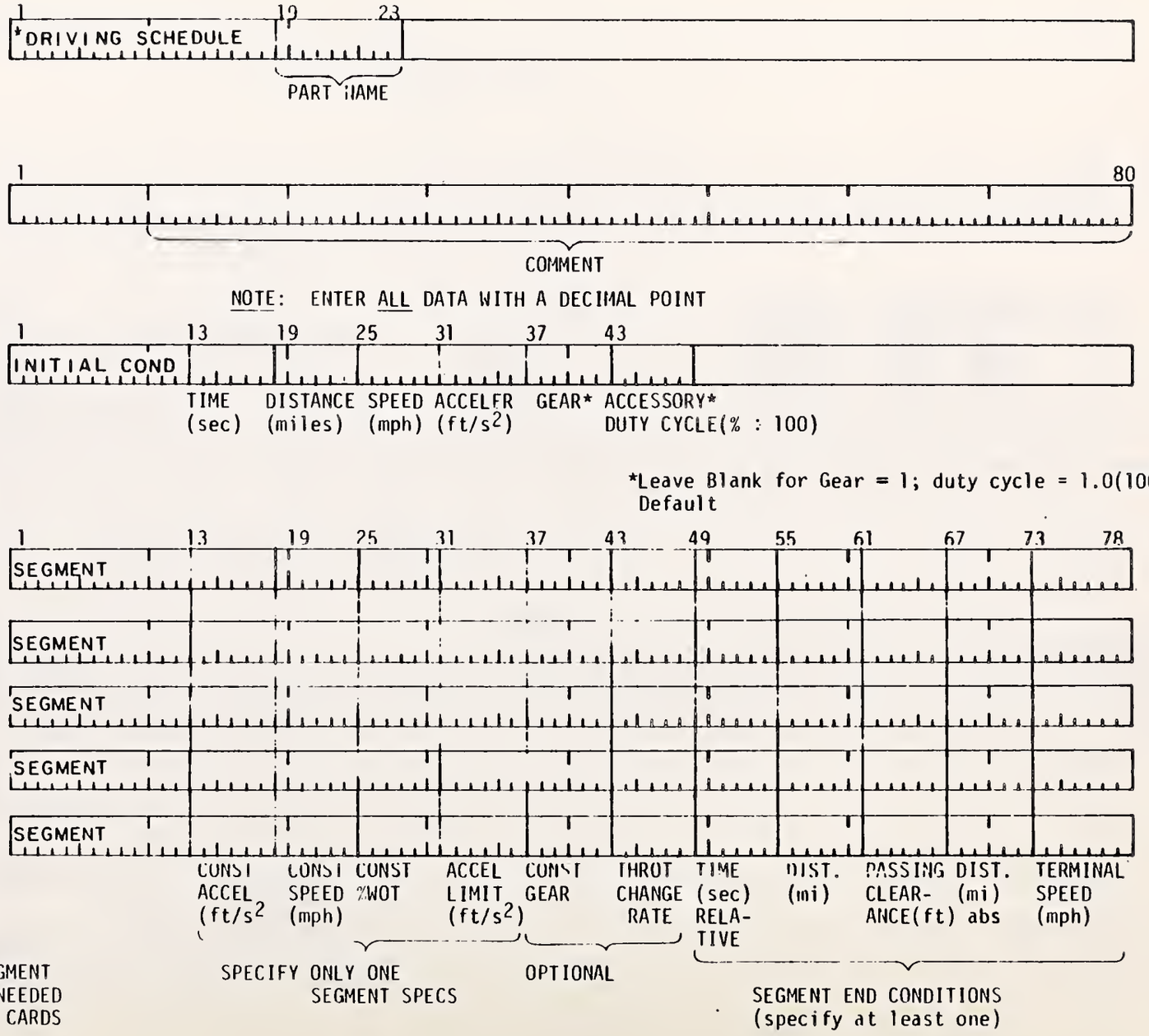


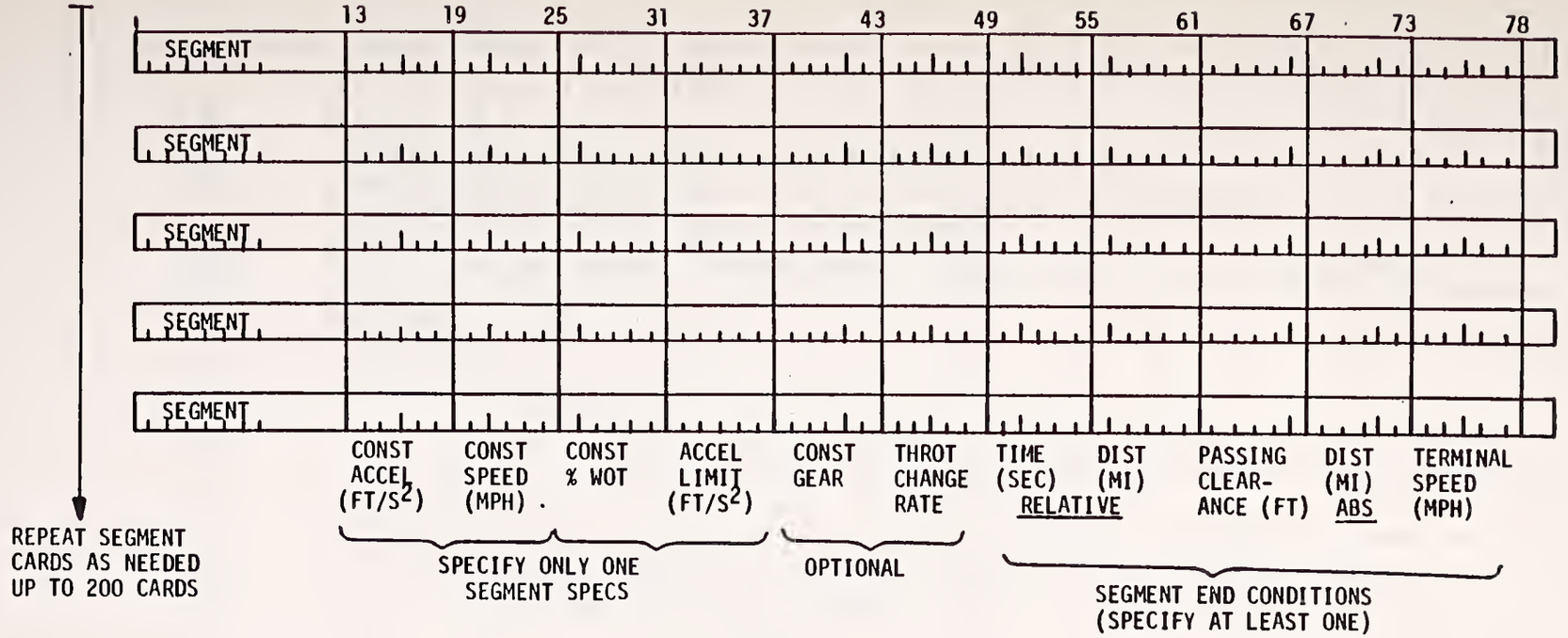


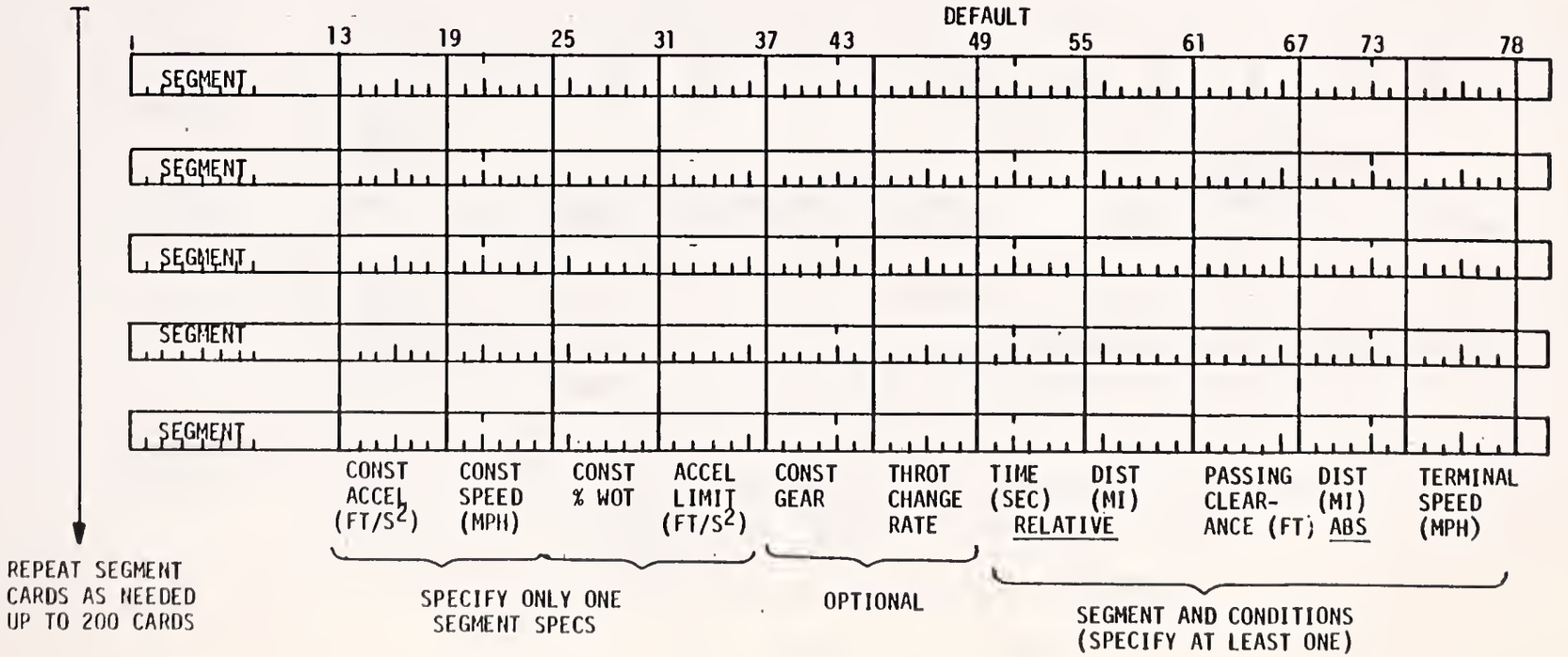
NOTE: ENTER ALL DATA WITH A DECIMAL POINT



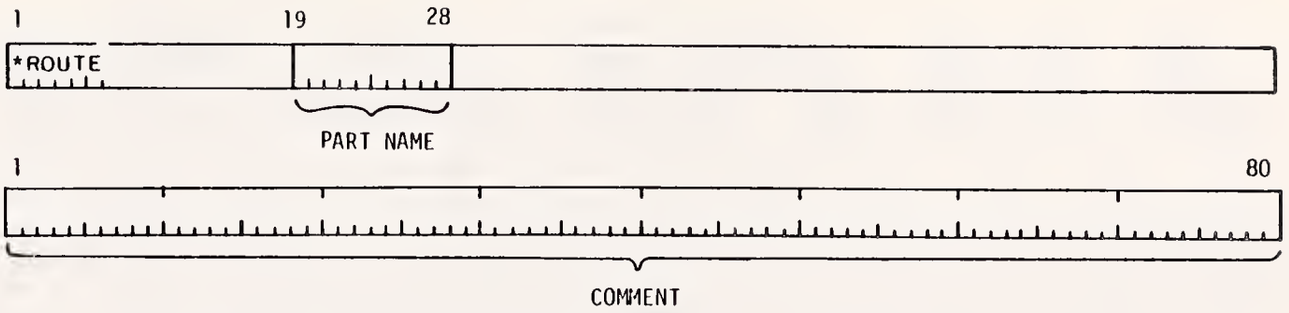
DATA SHEET 10. DRIVING SCHEDULE (Sheet 1 of 2)




13 19 25 31 37 43 49 55 61 67 73 78
SEGMENT
SEGMENT
SEGMENT
SEGMENT
SEGMENT
CONST ACCEL (FT/S2)
CONST SPEED (MPH)
CONST % WOT
ACCEL LIMIT (FT/S2)
CONST GEAR
THROT CHANGE RATE
TIME (SEC) RELATIVE
DIST (MI) RELATIVE
PASSING CLEAR-ANCE (FT)
DIST (MI) ABS
TERMINAL SPEED (MPH)
REPEAT SEGMENT CARDS AS NEEDED UP TO 200 CARDS
SPECIFY ONLY ONE SEGMENT SPECS
OPTIONAL
SEGMENT END CONDITIONS (SPECIFY AT LEAST ONE)

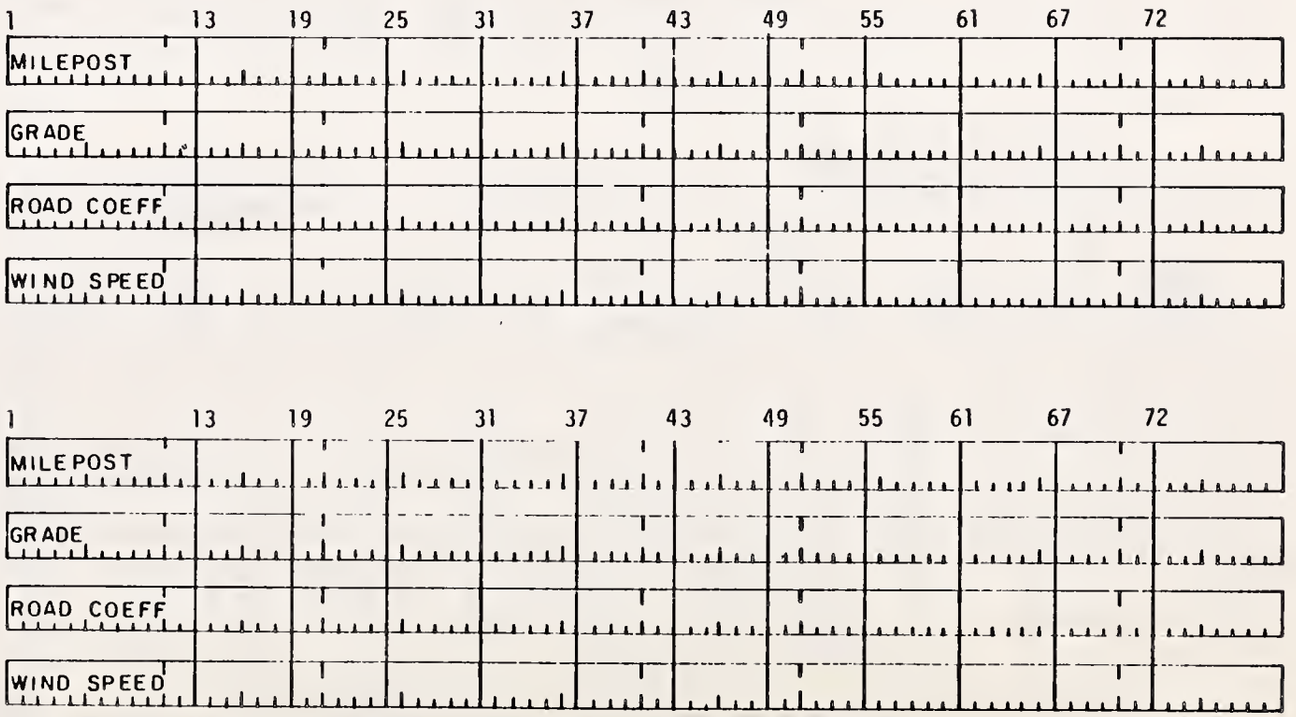

DEFAULT
13 19 25 31 37 43 49 55 61 67 73 78
SEGMENT
SEGMENT
SEGMENT
SEGMENT
SEGMENT
CONST ACCEL (FT/S2)
CONST SPEED (MPH)
CONST % WOT
ACCEL LIMIT (FT/S2)
CONST GEAR
THROT CHANGE RATE
TIME (SEC) RELATIVE
DIST (MI) RELATIVE
PASSING CLEAR-ANCE (FT)
DIST (MI) ABS
TERMINAL SPEED (MPH)
REPEAT SEGMENT CARDS AS NEEDED UP TO 200 CARDS
SPECIFY ONLY ONE SEGMENT SPECS
OPTIONAL
SEGMENT AND CONDITIONS (SPECIFY AT LEAST ONE)

DATA SHEET 11. ROUTE



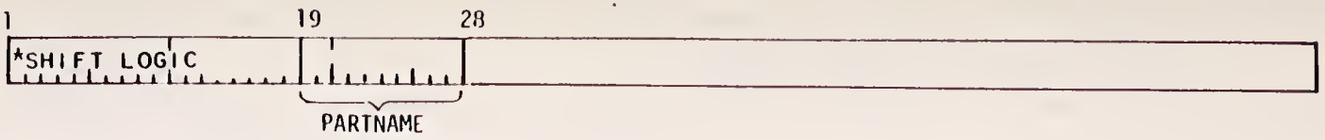
NOTE: ENTER ALL DATA WITH A DECIMAL POINT
SPECIFY GRADE AND ROAD COEFF FOR
UP TO THE DATA POINT MILEPOST

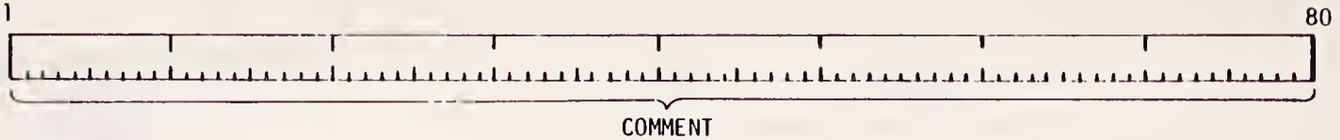
UNITS: MILEPOST (absolute miles)
GRADE (%)
ROAD COEFF (1.0 max)
WIND SPEED (mph)



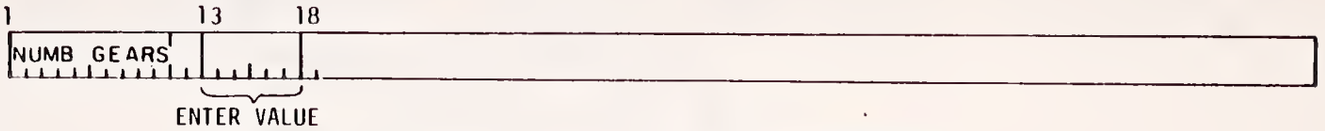
REPEAT 4-CARD SETS AS NEEDED UP TO 10 SETS (100 points maximum)

DATA SHEET 12. SHIFT LOGIC (Sheet 1 of 2)



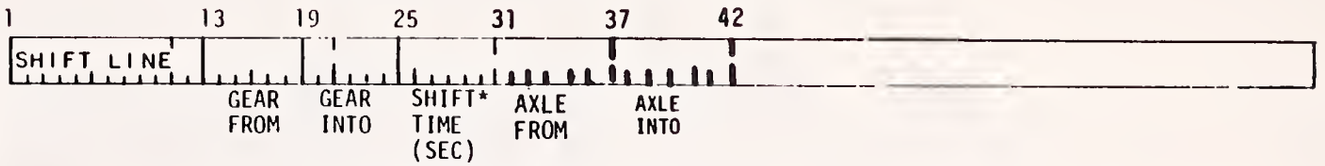


NOTE: ENTER ALL DATA WITH A DECIMAL POINT

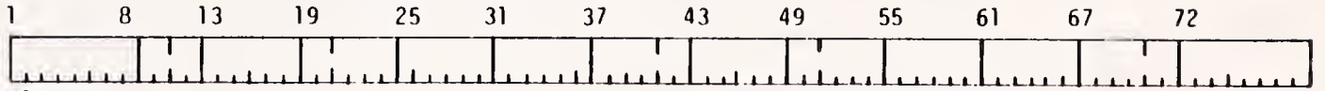


NOTE: 2*(NUMBER GEARS - 1) SHIFT LINES MUST BE SPECIFIED

REPEAT 3 OR 4 CARD SETS AS NEEDED UP TO 38 SETS FOR 20 GEARS

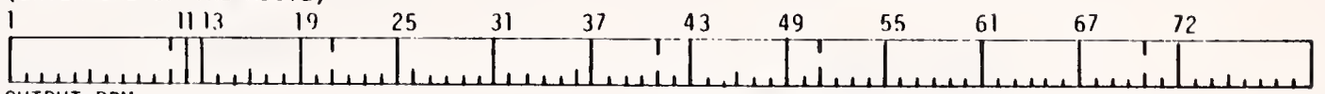


*LEAVE BLANK FOR SHIFT TIME = 1. SEC



UNITS: DTHROTTL (DEGREES)
VACUUM (IN Hg)
THROTTLE (% WOT)

(ENTER ONE OF THE ABOVE)

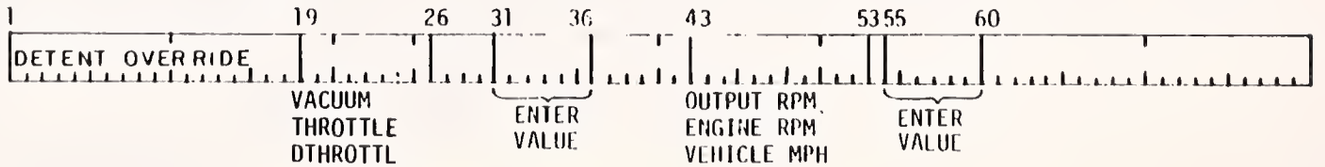


→ INCREASING SPEED FILL SPACES WITH DATA POINTS AS NEEDED UP TO 10 POINTS MAXIMUM

(ENTER ONE OF THE ABOVE)

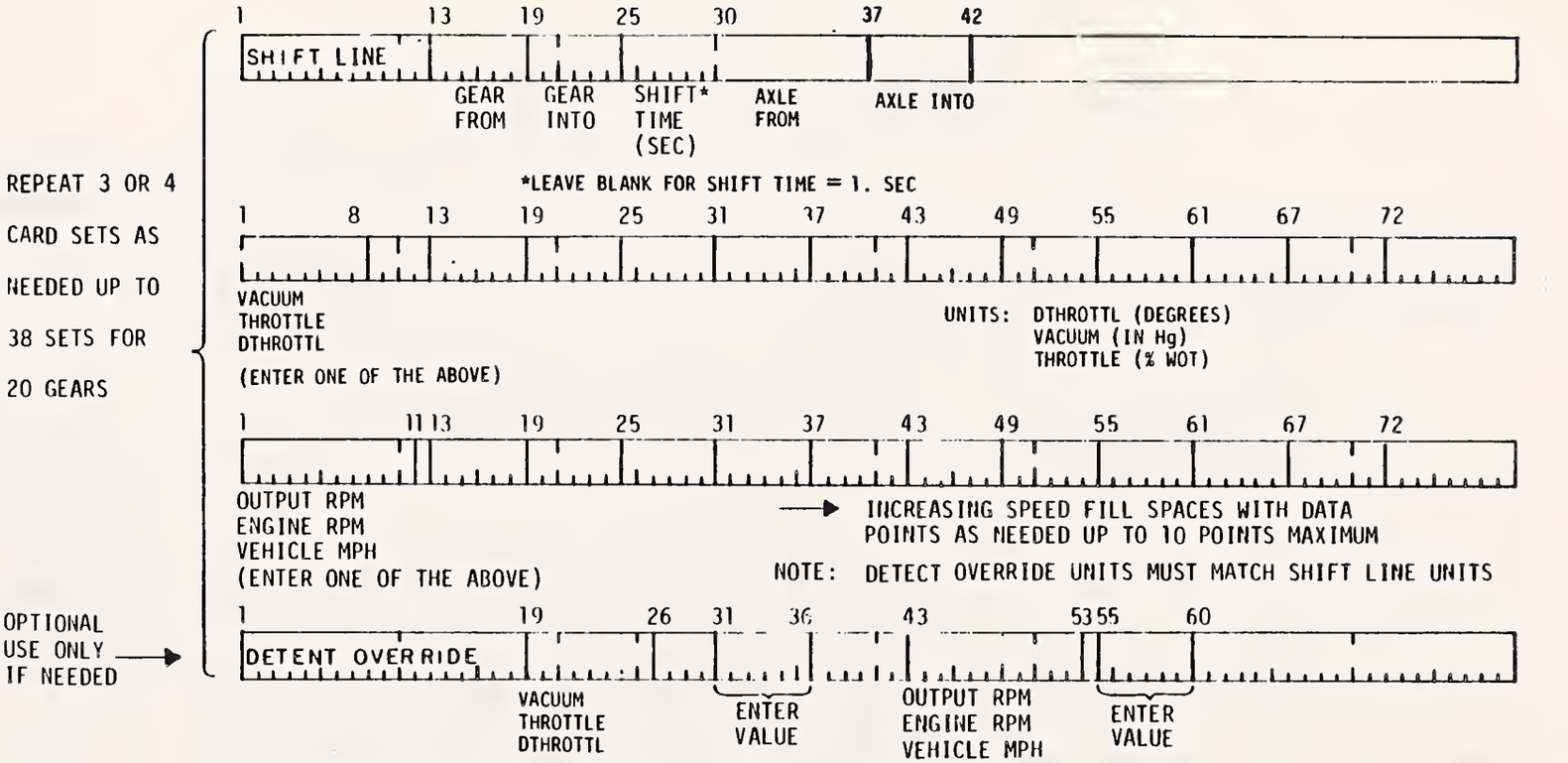
NOTE: DETENT OVERRIDE UNITS MUST MATCH SHIFT LINE UNITS

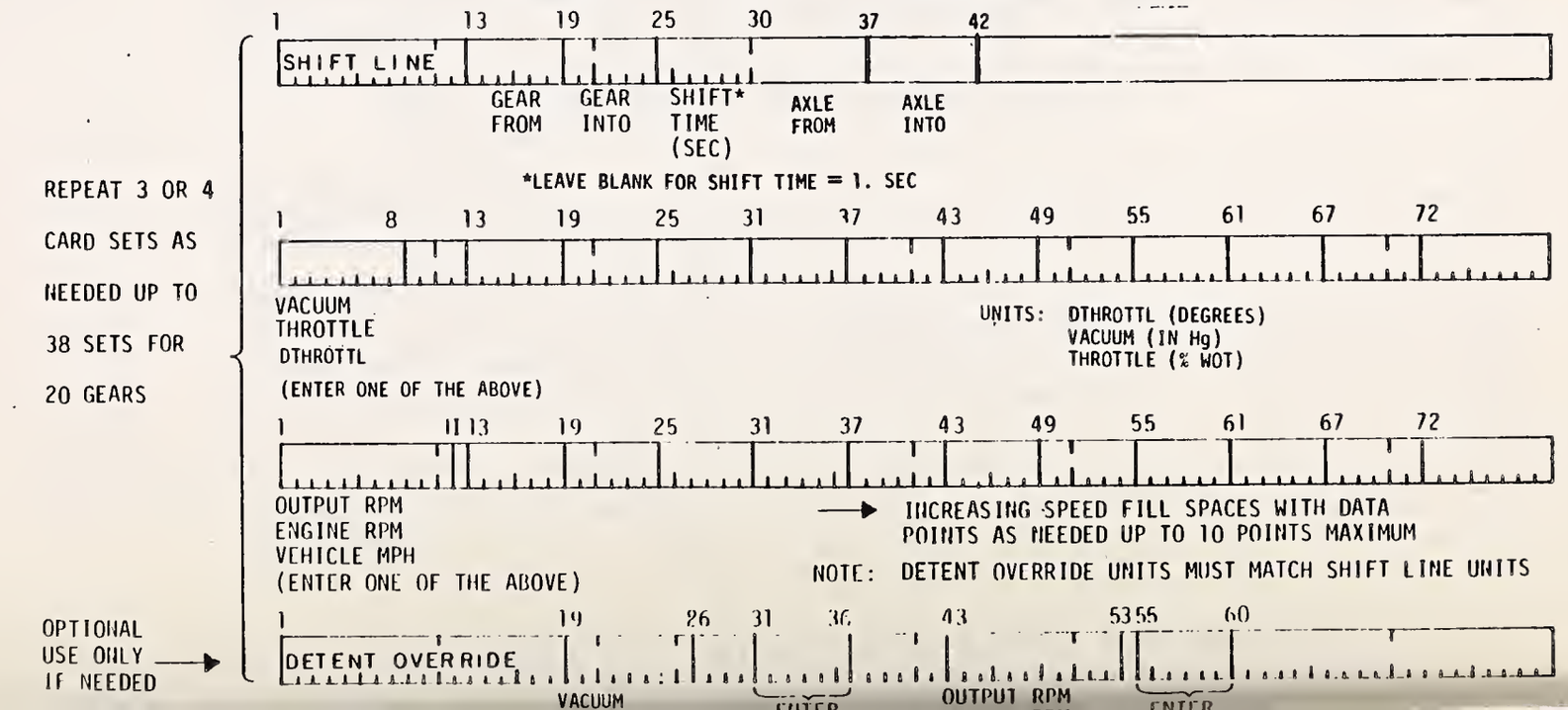
OPTIONAL USE ONLY IF NEEDED →



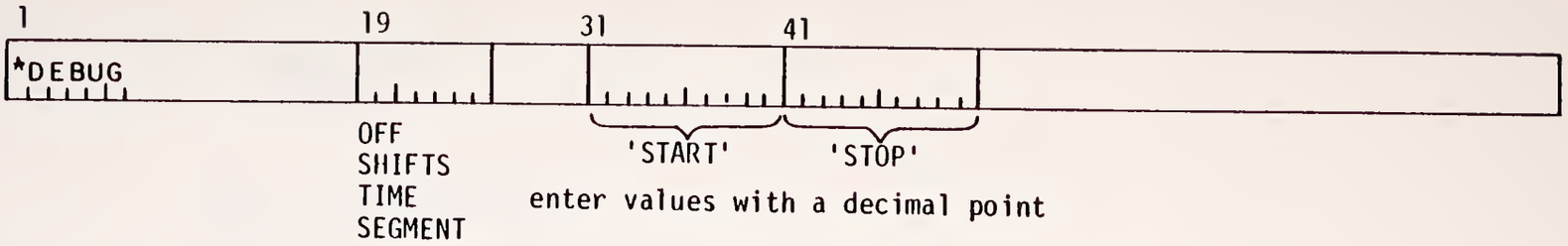
(ENTER ONE OF THE ABOVE FOR EACH SET OF UNITS)

DATA SHEET 12. SHIFT LOGIC (Sheet 2 of 2)





DATA SHEET 13. DEBUG



NOTES: FOR SHIFTS AND SEGMENT, ENTER SEGMENT NUMBERS
FOR TIME, ENTER VALUES IN SECONDS
IF NO 'START' SPECIFIED, DEBUG PRINTOUT STARTS AT BEGINNING OF RUN
IF NO 'STOP' SPECIFIED, DEBUG PRINTOUT CONTINUES TO END OF RUN

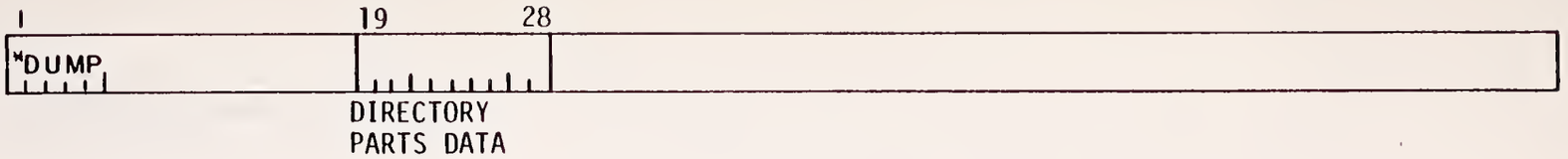
DATA SHEET 14. DROP

PART NAME

ACCESSORY
CONVERTER
DRIVING SCHEDULE
ENGINE
GEAR
ROUTE
SHIFT LOGIC
VEHICLE

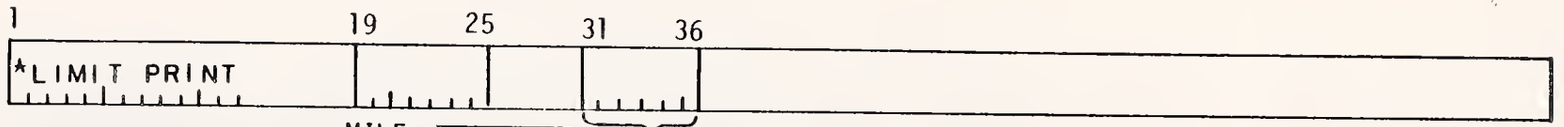
(enter one of the above)

DATA SHEET 15. DUMP



(enter one of the above)

DATA SHEET 16. LIMIT PRINT



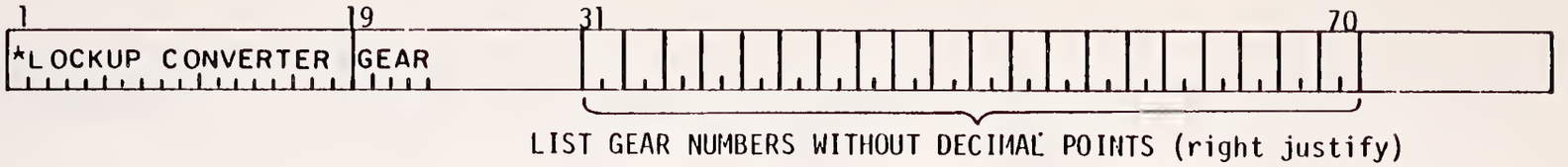
MILE
SEC
SEGMENT
OFF - FOR DETAILED PRINTOUT (including parts used data)
SUMMARY

ENTER VALUE FOR MILE, SEC WITH A DECIMAL POINT
(.1 mile, 10. sec defaults if no value entered)

(enter one of the above)

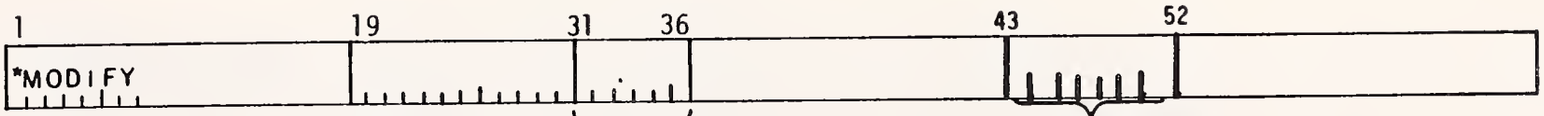
NOTE: IF NO LIMIT PRINT COMMANDED,
PRINTOUT IS SUMMARY ONLY

DATA SHEET 17. LOCKUP CONVERTER GEAR



NOTE: GEAR NUMBERS RANGE FROM 1 to 20

DATA SHEET 18. MODIFY



AREA
C1
C2
C4
CD
CYLINDERS
DIESEL
DISPLACEMENT
DUTY CYCLE
FUEL DENSITY
DYNAMOMETER
REAR AXLE R
STROKE
TIRE EFF
WEIGHT
WHEEL
WIND
IDLE
STEP
UPSHIFT
DOWNSHIFT
TRR
PHI

(enter one of the above)

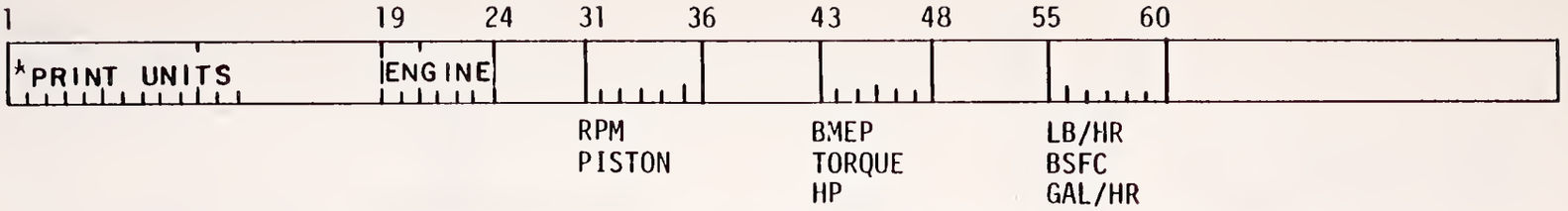
enter value with decimal point

PART NAME
OPTIMAL
(MUST USE WITH DUTY CYCLE)

Units:
AREA (sq. ft.)
DISPLACEMENT (cu in)
FUEL DENSITY (sp gr)
STROKE (in)
TIRE EFF (1. max)
WEIGHT (lbs)
WIND (mph)
IDLE (RPM)
STEP (sec) - default is .05 sec
UPSHIFT (± % change in vac/throt)
DOWNSHIFT (± % change in vac/throt)
PHI (Radians)
C_1, C_2, C_4 (lbs/lbwt)

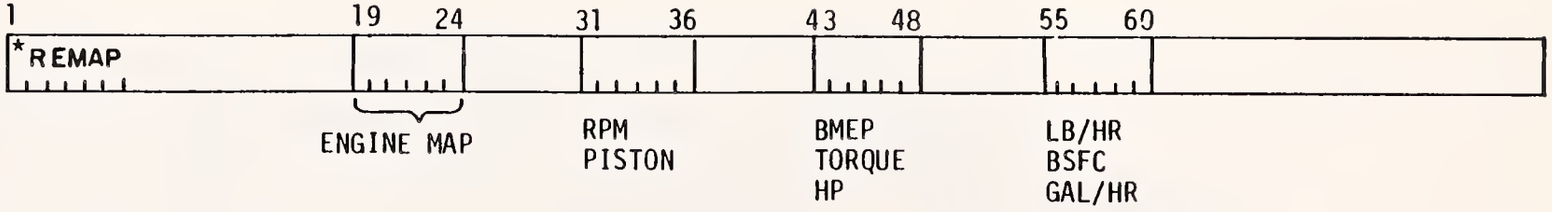
Enter Value with decimal point
(Note: Must specify accessory to be modified

DATA SHEET 19. PRINT UNITS



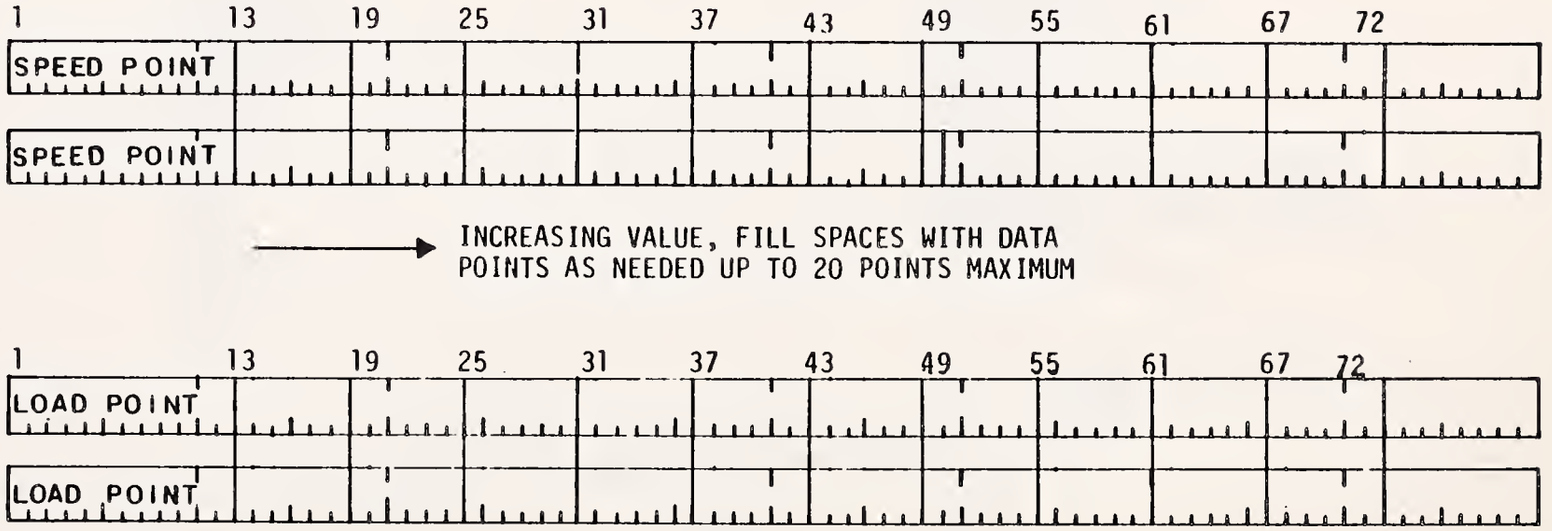
(enter one of the above for each set of units)

DATA SHEET 20. REMAP



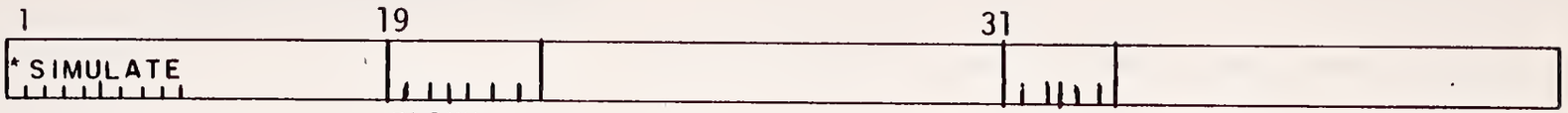
(enter one of the above for each set of units)

NOTE: ENTER ALL DATA WITH A DECIMAL POINT



USE ONLY
WHEN MORE
THAN 10
DATA POINTS
TO BE ENTERED
FOR EACH

DATA SHEET 21. SIMULATE

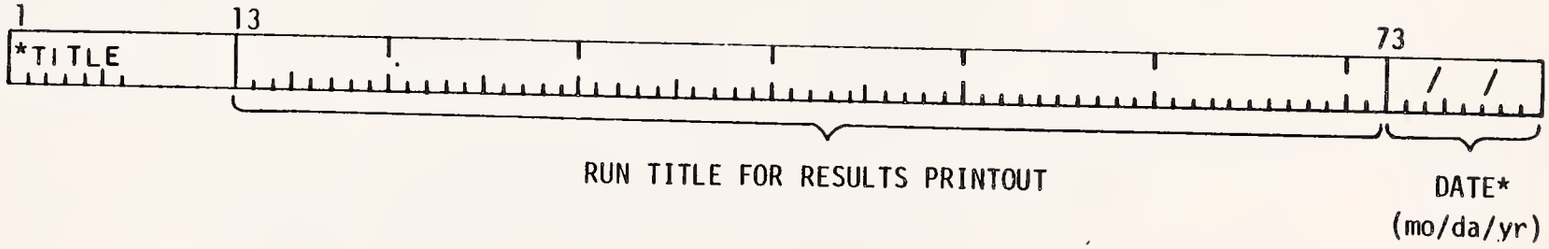


BLANK
TRUCK
BUS

(Default, Truck)

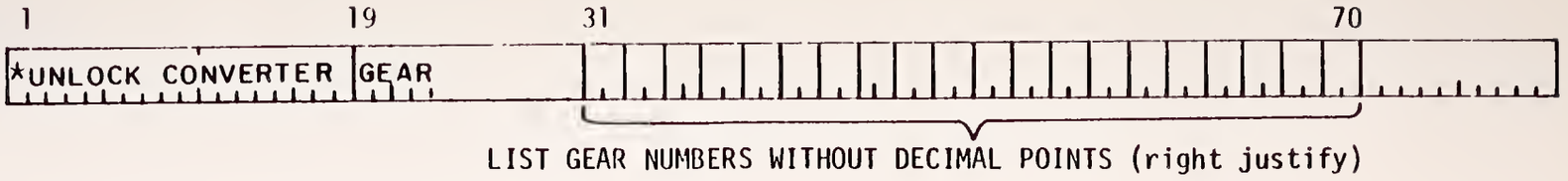
DIALOG - IF UNDER USER CONTROL,
BATCH GO TO COMMAND LEVEL
CONTI BEFORE SIMULATING

DATA SHEET 22. TITLE



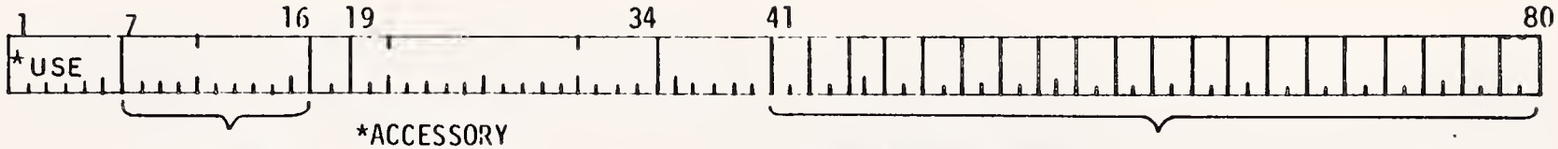
*Optional - insert to override actual date

DATA SHEET 23. UNLOCK CONVERTER GEAR



NOTE: GEAR NUMBERS RANGE FROM 1 TO 20

DATA SHEET 24. USE



PART NAME

*ACCESSORY
*CONVERTER

LIST GEAR NUMBERS WHEN REQUIRED WITHOUT DECIMAL POINTS
(right justified), gear numbers range from 1 to 20

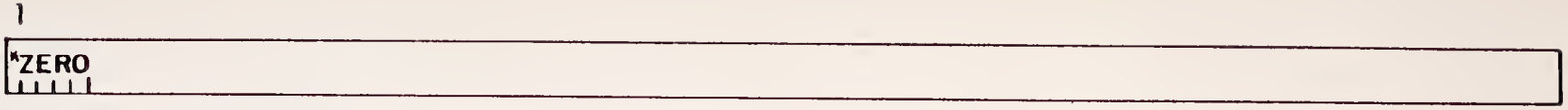
*DRIVING SCHEDULE
*ENGINE - LIST ALL GEARS TO BE USED WITH THE ENGINE
*GEAR - SPECIFY THE GEAR NUMBER THE GEAR IS TO BE USED FOR
*ROUTE
*SHIFT LOGIC
*VEHICLE

(enter one of the above)

NOTE: ONE OF EACH PART ABOVE MUST BE DEFINED BY
A USE COMMAND IN ORDER TO RUN A SIMULATION
WITH THE FOLLOWING EXPECTIONS:
1. ACCESSORIES ARE NOT REQUIRED.
2. TWO CONVERTERS (drive and coast) must be specified.

*More than one of these parts may be defined.

DATA SHEET 25. ZERO



NOTE: THIS COMMAND RESETS THE ENTIRE PROGRAM TO ITS' INITIAL STATE AND ERASES ALL PARTS LOADED WITH THE USE COMMAND

APPENDIX C

Program Listings

```
*  MODIFIED 1 AUGUST 1980: SOMERS
*
*****************************************************************************
C
      PARAMETER NMOD=24,NPART=11
C
      DOUBLE PRECISION  ANAME  ,APPEND ,APPN   ,APROT  ,AXNAME
     2,AXPPN  ,AXPROT ,BASDEV
     2,BASFIL ,BINARY ,BN     ,CCPPN  ,CCPROT
     3,CDATE  ,CDPPN  ,CDPROT ,CNAME  ,CTRDEV ,CTRFIL ,DATE
     4,DATPPN(15) ,DBLANK ,DELETE ,DNAME  ,DPPN   ,DPROT  ,DSTAR
     5,DWMASK ,ENAME  ,EPPN   ,EPROT  ,ESPECS
     6,GEANAM ,GNAME  ,GPPN   ,GPROT
     7,JOBDEV ,JOBPPN ,LPTDEV ,LPTDSP ,LPTFIL ,MASDEV ,MASFIL
     8,PNAME  ,RENAME ,RNAME  ,RPPN   ,RPROT  ,SCRDEV ,SCRFIL
     9,SEQIN  ,SEQIO  ,SEQOUT ,SNAME  ,SPPN   ,SPROT  ,TNAME
     1,TRNAM  ,TRPROT ,TRPPN  ,TPPN   ,TPROT
     2,UN     ,VNAME  ,VPPN   ,VPROT
C
C*****************************************************************************
C
      LOGICAL   BATCH  ,CLUTCH ,COAST  ,DIALOG ,DMPTTY
     2,ENDE   ,ENDLIN ,GDAT   ,LSTOP  ,LDIES
     2,LBMEP  ,LBRAKE ,LBSFC  ,LCLTCH ,LDETE  ,LDETNT ,LDYNOV
     2,LDETV  ,LDNSHF ,LDYNA  ,LEFIL  ,LENG   ,LGALHR ,LITER
     3,LIDLE  ,LHP    ,LIMPRN ,LIMTTY ,LIBHR  ,LLSH   ,LMPH   ,LOCKUP
     4,LPRNT  ,LPS    ,LRPM   ,LSCALE ,LSTRTE ,LSTSEC ,LTHR
     5,LTOR   ,LTRRZ  ,LVAC   ,LVEHAX ,LVNEW  ,LWOT   ,MILIM
     6,MIPSEC ,PBMEP  ,PBSFC  ,PGALHR ,PHP    ,PLBHR
     7,PPS    ,PRPM   ,PTOR   ,PTY    ,SECLIM ,SHFTNG ,SNGLBS
     8,TTY
C
C*****************************************************************************
C
      INTEGER GEANUM
C
C*****************************************************************************
C
C
      COMMON /ACCESS/  NACC,ACCT(20,20),ACCS(20,20),TORQA,ANAME(20)
     2, AIAS(20),ACCSR(20),DUTCYC(20),ASR
C
      COMMON/ACCTYP/APPEND,BINARY,DELETE,RENAME,SEQIN,SEQIO,SEQOUT
C
      COMMON /CDEBUG/  IDEBUG,DBEGIN,DSTOP,ISEG1,ISEG2,ISEG,CUMT,CUMD
     2       ,IDBUGO
C
      COMMON /CLUTCH/ TORQAO,TMINO,LCLTCH,SHFTNG,LDNSHF,ATORQF,ARPME
     2,CSTIM,CUMTLS,NGOCAL,CLUTCH,LIDLE,CCL,RPMC,HPCL,DTC,DRPMC
     3,HPCLO
C
      COMMON /CNTRL/   IC,TOLD,VOLD,T,V,ACCEL,D,DT,LITER,LSTRUP,LSTOP
C
      COMMON /CONSHF/ LLSH,ISHFT,STIME
C
      COMMON /CONST/   PRC1,PRC2,PAC,CD,AREA,VWIND,WGT,PGC,WRAD,RAR(3)
     2,  GRAT(20),NUMG,NGEAR,AIW,AIP,AI2,ERAT(20),EHAR(2,3)
     3,  AIE,AIA,AI1,BPEB,CDC,PHI,PSI,RIGIN(20),RIGOUT(20),WLSG,LTRRZ
     4,NGRLSS(20),GRPM(20,20),GRTORQ(20,20),GNAME(20),GCOM(16),PRC4,
```

```
     5 SAI1,SAI2,DVG,DVGEFF
C
      COMMON /CURVPT/  X(100),Y(100),NPTS,COEF(4)
C
      COMMON /DISK/  BN,BT,NREC,DUMCOM(16),NDISK,MLTSEC,CDATE,CHOUR
C
      COMMON /DSCHED/ DNAME,DCOM(16),TO,VO,DO,AO,NGO,NSEG,ASEG(50)
     2,VSEG(50),PWOT(50),ATHOLD(50),NGSEG(50),THRATE(50),TSEG(50)
     3,DSEG(50),PCSEG(50),POSTSE(50),VELSEG(50),ITYSEG(50)
     4,LSTSEC,NDSEG,NSEC,NAO
C
      COMMON /DSHIFT/ LMPH,AVEL
C
      COMMON /DYNAM/ LDYNA,DYM,LDYNOV
C
      COMMON /ENDOF/    ENDE,IPRNT,LEFII
C
      COMMON /ENGMAP/  RPMAX(2),RPMIN(2),NRPM(2),RPME,TORQE,FRATE,VAC,
     2           THR,MAPOK,IERRE,NTOR(2,20),EMAP(20,20,8),
     3           ERPM(2,20),ENMIN(2),SPIDLE(2),TORQEO,LTHR
C
      COMMON / FBRAKE / ABR,ABRO,LBRAKE
C
      COMMON /FILES/ JOBNUM,JOBDEV,JOBPPN,SNGLBS,MASDEV,MASFIL(15)
     2,MASPPN(2),BASDEV(15),BASFIL(15),BASPPN(2,15)
     2,CTRDEV,CTRFIL,CTRPPN(3)
     4,LPTDEV,LPTFIL,LPTPPN(3),LPTDSP
     5,SCRDEV,SCRFIL,SCRPPN(3),LPTPRO
C
      COMMON /GBI/      UT,UN,NUG(20),JENG(20),IENG
C
      COMMON/HCONS/HCOMND(40),HBLANK,HUP,HDN,HON,DSTAR
     2,HSTAR,DBLANK,HQ
C
      COMMON  /HISTOG/HIST(20,20,3),DELRPM,DELTOR,FIRRPM,FIRTOR
     2       ,TOPRPM,TOPTOR
C
      COMMON /INPCOM/ SIMODE,PNAME,WORD,DATA(80),IDATA(20),LOCKG(20)
     2,UNITS(4),FSPECS(8)
C
      COMMON /IO/ ECOM(16),ENAME(2),DISP,ICYI,IMIN,IMAX,THRMAX,
     2                 TBRMIN,EINEB,BORE,STROKE,FSPGR,NCYCLE,LDIES
C
      COMMON /LDIMEN/ LRPM,PRPM,LPS,PPS,ITOR,PTOR,LBMEP,PBMEP,IHP,PHP,
     2               LLDHR,PLBHR,LBSFC,PBSFC,LGALHR,PGALHR,LPRNT
C
      COMMON /MASKS/ WMASK(5),DWMASK(10)
C
      COMMON /MISC/     NNA(20),TIREFF
C
      COMMON /MOCOM/    BB1,BB2,RPMWO,RPMEO,CPHI,VWINDC,VWINDS,AA1,AA2,
     2                  AA3,AA4,AA5,AA6,AA7,AA8,AA9,AA10,RARSQ,AA11
C
      COMMON /MODE/ BATCH,DIALOG,PTY,TTY,DMPTTY
C
      COMMON /OLDIO/    ODISP,OBORE,OSTROK,IOCYL
C
      COMMON /OLDMAP/   EMAPO(20,20,4),ERPMO(20),NTORO(20),NRPMO
C
      COMMON /OUTP/FWHEEL,FAERO,FACCEL,TRR,TORQF,DRPMW,DRPME,FROLL,
     2   FGRADE
```

```
C
      COMMON /PRNLIM/ LIMPRN,MILIM,SECLIM,ENDLIM,ALIMN,LSCALE
C
      COMMON /PROT/ EPROT(2),EPPN(2),CDPROT,CDPPN,CCPROT,CCPPN
     2,VPROT,VPPN,GPROT(30),GPPN(30),APROT(30),APPN(30),DPROT,DPPN
     3,SPROT,SPPN,RPROT,RPPN,TRPROT,TRPPN
C
      COMMON /RTE/ RNAME,NRDIST,RDIST(10),RGRADE(10),RCOM(16)
     2,      RCOEF(10),RVWIND(10),ISTRTE,NDRTE,NRTE
C
      COMMON /RUNID/ TITLE(12),VERID(3)
C
      COMMON /SEGNO/ITS
C
      COMMON /SHFTL/   SNAME,SCOM(16),GOVPSI(4),OUTRPM(4),NGPT,IGP(60),
     2       IGI(60),SHFTIM(60),SHFTPT(10,60),SHFTRP(10,60),LVAC,LENG,
     3       GDAT,NSPTS(60),LDETNT,DETPT(60),DETRPM(60),PARAB,LDETE,
     4       LDETV,GOVLIN,IAP(60),IAT(60),NUMBSL,RERAT(47,60),
     5       IREB,IBERO,NREB
C
      COMMON /SIMCOM/ ILPT,DSFC,C1TO7,C2345,C345,CAC,CAE,CALLT,CBR,CDA
     2,CDCR,CDD,CDI,CDIF,CEA,CECR,CED,CEI,CENERG,CFA,CFB,CFCR,CFD,CBV
     3,CFI,CGB,CRO,CTA,CTCR,CTD,CTI,CTIRE,CTOTAL,CTR,CUMEN,CUMENM
     4,CUMFE,CUMFU,EFFC,FEINST,FPGAL,HPE,HPMI,HPW,NGCNT,NSFPSG
     5,NSPSEG(50),NSHIFT,PCKE,PCPE,PCROT,PCTHR,PPHPHR,RDLD,SMILE
     6,TFRC1,TFRC2,VAVG,TFRC4,PCOLD
C
      COMMON /SIMCM2/ RTHR,CEDN,CENG,CRE,CKE,CROT,HPBC,TTOT,DTOT
     2,NRTSEG,ENDRSG,ENDRTE,ROADC,MXNGCM,TTT1,CUMG,ICNT,ING
     3,NGOLD,HPAO,HPEO,HP1O,HP2O,HPPO,HPWO,FRATEO,FRATGO,FROLLO
     4,FGRADO,FWHEEO,FAERDO,DTIRE,DTIREO,ALOSGO,ALOSRO,ALOSCO
     5,RPMEI,RPMWI,CPE ,ALOSLO,ALOSBO
C
      COMMON/SSTIME/CPUS,CPUT
C
      COMMON /TMAP/    TORQ,LWOT,TWOT,TMIN
C
      COMMON /TRANSM/ TRNAM,TRCOM(16),NGTR,GEANAM(20),GEANUM(20)
C
      COMMON /TRQRPM/ TORQP,RPMP,TORQW,RPMW,TORQ1,RPM1
C
      COMMON /TTYOUT/ LIMTTY,ITTYWD,LLPT,JCT
C
      COMMON /TORCON/  TORBPK,TORQ2,RPM2,COAST,SR,TR,TRD(20),SRD(20),
     2                 AKD(20),NTD,SRC(20),AKC(20),NTC,NTBP,TIN(20),
     3                 TOUT(20),SPIN(20),SOUT(20),NTORP,CDIAM,CNAME,
     4                 CCOM(16),CONTOR,RPM2O
C
      COMMON /V2MISC/  LOCKUP(20),DATE,NPARTS(11),DEFDT,NORUN,MENG,
     2       IUNIT,IPART,IPMODE,IRMODE,ISMODE,NSGEAR(20,2),NXTPG
     3       ,NOMPAR,RZERO
C
      COMMON /VEHICL/ VNAME,VCOM(16),TNAME,TCOM(16),ACOM(16)
     2,NRAX,NPAX(2,3),AXRPM(20,2,3),AXTORQ(20,2,3),LVNEW,AXCOM(16)
     3,AXNAME,AXPPN,AXPROT,LVEHAX,TPPN,TPROT,NAXS,NAX,NAXO
C
C*****************************************************************************
C
      EQUIVALENCE (DATPPN(1),BASPPN(1,1))
C
```

```
*               **** ASCIZ ****
*
      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, RCRCNT
C
C       CALLED BY: HLPCMD, LOOKUP
C
C*****************************************************************
C
      INCLUDE 'COMMS/WOLIST'
C
      DIMENSION STRING(LWRDS)
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=RCRCNT(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
```

```
C
C       SUBROUTINE CONVTR
C
C       ENTRY POINTS: CONVTR
C
C       CALLED BY: GOBACK
C
C*****************************************************************
C
C
        INCLUDE 'COMMS/NOLIST'
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.TORBPK) 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
        GO TO 65
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
```

```
*               **** DEBUG ****
*
      SUBROUTINE DEBUG(HIDNAM)
C
C     ENTRY POINTS: CTRLD, DEBUG
C
C     CALLED BY: GOBACK, SHIFTS, SIMCTR, SIMIPT, SIMSTS
C
C*****************************************************************
C
C
      INCLUDE 'COMMS/NOLIST'
C
      ICALL=1                                          !FLG NORMAL SUBROUTINE ENTRY.

      GO TO(900,800,300,400),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                                 !CAI 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 % WPT.
      NRTEA=NRTSEG+NDRTE                               !CALC RTE SEG.
      IF(.NOT.SHFTNG) GO TO 820                        !(SHIFTING IN PROGRESS?)NO.

      HSMODE=HUP                                       !YES, ASSUME UPSHIFT.

      IF(LDWSHF) HSMODE=HDN                            !(DOWNSHIFT)YES
      HTMODE=HUN                                       !ASSUME GEAR UNLOCKED.

      IF(LCLTCH) HTMODE=HBLANK                         !(GEAR LOCKED?)YES.
      DRPM1E=RPM1-RPME                                 !CALC NEEDED CHG IN PRME 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=IOUNIT                                       !SEND OUTPUT TO LINE PRINTER.

  840 IF(IDBUGO.EQ.1)WRITE(LPT,1700)
      WRITE (LPT,1800)                                 !DEBUG OUTPUT.

      1  CUMT,CUMD,V,ACCEL, FWHEEL,FROLL,FAERO,FACCEL,FGRADE,
```

```
     2 DT,NGEAR,NAX,ISEGA,NRTEA,TORQW,TRR,TORQP,TORQ2,TORQ1,TORQF
     2,TORQE,
     3  PCTWOT,HIDNAM,ABR, RPMW,DRPMW,RPMP,RPM2,RPME,DRPME,
     4       NGOCAL,MAPOK
C
      IF(SHFTNG) WRITE(LPT,1900) HSMODE,HTMODE,CUMTLS,CSTIM,TORQA      !(SHIFTING?)YES, OUTPUT SHIFT DATA.
     1,TORQAO,AFORQF,ARPME,DRPM1E,RPMC,DRPMC,DTC,LCLTCH
      IF(LISTTY.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,1850)                                         !OUTPUT TO JCT.
     1  CONT,CUMD,V,ACCEL, FWHEEL,FROLL,FAERO,FACCEL,FGRADE,
     2 DT,NGEAR,NAX,ISEGA,NRTEA,TORQW,TRR,TORQP,TORQ2,TORQ1,TORQF
     2,TORQE,
     3  PCTWOT,HIDNAM,ABR, RPMW,DRPMW,RPMP,RPM2,RPME,DRPME,
     4       NGOCAL,MAPOK
C
      IF(SHFTNG) WRITE(JCT,1950) HSMODE,HTMODE,CUMTLS,CSTIM,TORQA      !(SHIFTING?)YES, OUTPUT SHIFT DATA.
     1,TORQAO,AFORQF,ARPME,DRPM1E,RPMC,DRPMC,DTC,LCLTCH
C
  900 IDBUGO=IDEBUG
      RETURN                                                       !BYE.

C
C*********************************************************************
C
C     FORMAT STATEMENTS
C
1700  FORMAT(/
     1' CONT,CUMD,V,ACCEL, FWHEEL,FROLL,FAERO,FACCEL,FGRADE',/,
     2' DT,NGEAR,NAX,ISEGA,NRTEA,TORQW,TRR,TORQP,TORQ2,TORQ1'
     2,',TORQF,TORQE',/,
     3'  PCTWOT,HIDNAM,ABR, RPMW,DRPMW,RPMP,RPM2,RPME,DRPME',/,
     4'       NGOCAL,MAPOK',/)
1800  FORMAT (/' 3'F9.2,F8.3,1X,F7.3,F10.3' (F)'5(1XG10.4)/
     1,'  DT'F6.3',GEAR'2I3',DRS'I4',RTE'I4'  (T)'7(1XG10.4)/
     2,'  PWOT'F8.3',ID:'A5',BRK='F7.1' (R)'6(1XG10.4),/
     3,'  GOBACK',I5,5X,'MAPOK:',I2)

1950  FORMAT(/' 3'F9.2,F8.3,2(1XG7.3)' (F)'4(1XG9.4)/
     1,37X3(1XG10.4)/
     1,'  DT'F6.3',GEAR'2I3',DRS'I4',RTE'I4'  (T)'4(1XG9.4)
     1,/35X3(1XG10.4)/
     2,'  PWOT'F8.3',ID:'A5',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)'8(1XG10.4)/
     1,3(1XG10.4))
C
      END
```

```
*                **** DSK ****
*
      SUBROUTINE DSK
*
*  MODIFIED 6 JUNE 1990: SOMERS
*
C     ENTRY POINTS: DSK
C
C     SUBROUTINES CALLED: DSKCTR, DSKRD, DSKWR, PRNOUT
C
C     CALLED BY: INPBAT, PRNTPD, SIMCTR
C
C*****************************************************************
C
C
      INCLUDE 'COMMS/NOLIST'
C
      LOGICAL NXTSEC
C
      DOUBLE PRECISION ADUM
C
      DIMENSION HPART(14)
C
      DATA ( HPART(I),I=1,11) /'ENGIN','CONVE','VEHIC','GEAR'
     2,'ACCES','DRIVI','SHIFT','ROUTE','TIRE','TRANS','AXLE'/
     3,LPT/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=NSEC+1                             ! CALC DRS SECT # TO BE GOTTEN
      WRITE(JCT,1011) 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,1011) 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),$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,SPOIN)                 ! NO, REOPEN FILE FOR INPUT ONLY
  235 READ  (NDISK ,END=240 )  BN,DT,NREC
      IF(BN.EQ.PNAME.AND.DT.EQ.HPART(IPRNT)) GO TO 250
      GO TO 235
C
```

```
C     ERROR SECTION IF /USE/ COMMAND , PART NOT ON PARTS DATA FILE
C
  240   IF ( JPRNT .EQ. 0 )  GO TO 300          ! (NOT/USE/?) YES, GO STORE NEW PART.
        IP=IPRNT
        IF(SNGLBS) IP=12
        IF(TTY.AND.BATCH) WRITE(JCT,1240) HPART(IPRNT),PNAME
        1,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        WRITE(LPT,1240) HPART(IPRNT),PNAME
        1,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
  243   IPRNT  = - 10
        GO TO 900
C
C  EMPTY OR NON-EXISTENT DATA BASE FILE
C
  245   IP=IPRNT
        IF(SNGLBS) IP=12
        IF(JPRNT.EQ.0) GO TO 247                ! ( PART TO STORE?) YES.
        WRITE(JCT,2450) BASDEV(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),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        1,HPART(IPRNT),PNAME                    ! % CREATED FILE MESS
        GO TO 300                               ! GO STORE PART NOW!!
C
C     IF /USE/ COMMAND , GO TO LOAD PART DATA FOR SIMULATION
C
  250   IF ( JPRNT.NE.0 )  GO TO 500            ! (/USE/?) YES.
C
C   CHECK FOR MULTIPLE SECTION DRIVING SCHEDULE (50 SEGS/SECTION)
C       OR ROUTE(10 SEGS/SECTION)
C
        IF((IPRNT.EQ.6.AND.NSEG.LT.0).OR.(IPRNT.EQ.8.AND.NRDIST.LT.0))
        1 GO TO 320                                                     ! NO,(MULTI SECT?) YES.

C
C     WRITE ERROR MESSAGE AND DO 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,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        WRITE (LPT,1250)  BT,BN
        1,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        GO TO 900
C
C*****************************************************************
C
C     STORE PART DATA AT END OF PARTS DATA FILE
C
  300   NREC=IPRNT
        IF(IPRNT.NE.6 .OR. 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 .OR. NRDIST.GT.0) GO TO 310
        NRDIST=IABS(NRDIST)
  305   NREC=-IPRNT
  310   BT=HPART(IPRNT)
        BN=PNAME
        CDATE=DATE
C
        CALL DSKCTR(IPRNT,APPEND)               ! OPEN DB FILE
```

```
C
      CALL TIME(CHOUR)                    !GET CREATION TIME OF PART ON DB
C
      CALL DSKWR                          ! 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,BT,NREC
      IF(BN.EQ.PNAME.AND.BT.EQ.HPART(IPRNT).AND.NREC.LT.0) GO TO 320
  325 WRITE(JCT,1409) JPRNT,IPRNT         ! IMPOSSIBLE ERR OCCURED IN DB FILE NOW OPN
      WRITE(LPT,1409) JPRNT,IPRNT
      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
```

```
C
      IF(ADUM.NE.PNAME) GO TO 325     !(ERR?) YES, GO ERR BYE
      IF(IPRNT.NE.6) GO TO 522        ! NO, (DRS?) NO.
      WRITE(JCT,1413) NSEG            !YES, "/NSEC:I3"
      NSEG=NDSEG+NSEG                 !SET DRS SEG LIM
      IF(.NOT.LSTSEC) GO TO 524       !(LAST DRS SECT?) NO.
      NSEC=)                          !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"
      NRDIST=NDRTE+NRDIST             !SET RTE SEG CNT LIM
      IF(.NOT.LSTRTE) GO TO 524       !(LAST RTE SECT?) NO.
      NRTE=)                          !YES, ZERO RTE REC PTR
  523 WRITE(JCT,1414)          !"/LAST SECTION"
  524 WRITE(JCT,1415)          !"]"
      GO TO 900                       !*DONE
C
  530 BACKSPACE NDISK                 ! POSI DB FILE
C
      CALL DSKRD                      ! LOAD PART DATA
      CALL PRNOUT                     ! PRINT IT
      GO TO 900                       ! *DONE
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']')
 1250 FORMAT (' ? DSK- PART 'A5,1XA10' NOT STORED'
     1,' ALREADY ON PARTS DATA FILE 'A6':'A10'['05','03']')
 1409 FORMAT(/' ? DSK- IMPOSSIBLE FILE POSITIONING ERROR.'
     1,' (JPRNT='I3'/IPRNT='I3')')
 1411 FORMAT(' [DSK-'A3'/NSEC:'I3$)
 1413 FORMAT('+/NSEG:'I4$)
 1414 FORMAT('+/LAST SECTION'$)
 1415 FORMAT('+]')
 2450 FORMAT(/' ? DSK- EMPTY PARTS DATA FILE 'A6':'A10'['05','03']')
 2470 FORMAT(/' % DSK- CREATED FILE 'A6':'A10'['05','03']'
     1,/5X'TO STORE PART 'A5,3XA10)
C
      END
```

```
*
*  MODIFIED 11 JULY 1980: SOMERS; REMOVED A PREVIOUS MOD.
*
      SUBROUTINE DSKCTR(IARG1,DARG2)
C
C     ENTRY POINTS: DSKCTR
C
C     CALLED BY: DSK, DSKDEL, DSKDIR, INPBAT, SIMCTR, VSMCTR
C
C*****************************************************************
C
C
      INCLUDE 'COMMS/NOLIST'
C
      DOUBLE PRECISION DARG21,ACCESS,OLDACC
C
      DIMENSION USEPPN(3)
      REAL ITMPPN(2),LPTPPN
C
      EQUIVALENCE (ITMPPN(1),JOBPPN)
C
      DATA ISCR/0/,IOPEN/0/,USEPPN(3)/0/
C
C*****************************************************************
C
C   DATA BASE DISK FILE CONTROL FOR SUBROUTINE DSK
C
      IPRNTI=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(SNGLBS) 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)
      IPRNTI=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(TTY)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   USERPPN(1)=BASPPN(1,IPRNTT)                                      !GET DIRECTORY PATH.
        USERPPN(2)=BASPPN(2,IPRNTT)
        OPEN(UNIT=3,DEVICE=BASDEV(IPRNTT),ACCESS=ACCESS,MODE=BINARY      !OPEN DB FILE.
        1,FILE=BASFIL(IPRNTT),DIRECTORY=USERPPN,PROTECTION="022)
        OLDACC=ACCESS                                                    !SAVE ACCESS OF HOW DB OPENED.

        IOPEN=IPRNTT                                                     !SAVE PTR TO OPEN DB FILE.
```

```
      RETURN
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 DD 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
```

```
*               **** DSKDEL ****
*
        SUBROUTINE DSKDEL
*
*  MODIFIED 23 SEPT 1980: SOMERS
*
C       ENTRY POINTS: DSKDEL
C
C       CALLED BY: INPRAT
C
C*****************************************************************
C
C
        INCLUDE 'COMMS/NOLIST'
C
        LOGICAL LTEST
C
        DIMENSION HPART(11)
C
C
C
        DATA  (HPART(I),I=1,11) /'ENGIN','CONVE','VEHIC','GEAR'
       2,'ACCES','DRIVI','SHIFT','ROUTE','TIRE','TRANS','AXLE'/
       2,LPT/6/,JCT/5/,ISCR/2/
C
C*****************************************************************
C
C     /DELETE/ COMMAND - DROP PART FROM PARTS DATA FILE
C
        NDISK=3                                          ! RESET IN CASE ^C , ,REENTER WAS DONE
        GO TO 620
C
  615   WRITE (JCT,1240)  UT,UN                        !ERROR PRINTOUT , PART NOT ON PARTS DATA FILE
       1,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
  617   IPRNT     = 10                                 ! SET ERROR FLAG
        GO TO 900                                      ! GO BYE
C
C     COPY PARTS DATA FROM UNIT 3 TO UNIT 2 SKIPPING OVER PART TO DROP
C
  620   IPRNTD=-IPRNT                                  ! SET PTR TO FILE TO OPEN & CALL FOR SCR FILE
        CALL DSKCTR(IPRNTD,SEQOUT)                     ! OPEN DB & SCR FILE
        IF(IPRNTD.EQ.-12) GO TO 617                    ! (FILE PROT FAILURE?) YES, GO ERR BYE
        NSECF=0                                        ! NO, INIT TO DROP
        IP=IPRNT                                       !SET FILE INFO PTR.
        IF(SNGLDS) IP=12                               !(SINGLE FILE DB?)YES, RESET PTR.
        IF(IPRNT.EQ.1) IENG=1                          !(PROCESSING ENGINE?)SET PTR TO LOC TO USE.
        IF(IPRNT.EQ.4) NGEAR=1                         !(PROCESSING GEAR?)SET PTR TO LOC TO USE.
        IF(IPRNT.EQ.5) NACC=1                          !(PROCESSING ACCESSORY?)SET PTR TO LOC TO USE.
        NDEL=0                                         !ZERO CNT OF PARTS DELETED.
```

```
      ASSIGN 625 TO READST                          !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 READ(NDISK,END=645) BN,BT,NREC,LTEST
      IPRNT=IABS(NREC)
      IF ( BN.NE.UN .OR. BT.NE.UT )  GO TO 630
      IF(NREC.LT.0) GO TO 627                       !(MULTI SECT?) YES, FIND LAST SECT.

  626 NDEL=NDEL+1                                   !NO, GOT ENTIRE PART

      WRITE(JCT,1620)
     1BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP),BT,BN
      WRITE (LPT,1620)
     1BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP),BT,BN
      ASSIGN 629 TO READST                          !FOUND PART SWITCH TO LOGIC TO GET REST OF FILE.
C
  627 NSECF=NSECF+1                                 ! INC # OF SECTS

      IF(NSECF.EQ.1) GO TO 625                      !(1ST SECT?) YES, GO LOOK NEXT REC.

      IF(IPRNT.NE.6.AND.IPRNT.NE.8) GO TO 617       !(DRS OR RTE?) NO, ERR.
      IF(.NOT.LTEST) GO 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(NDISK,END=645) BN,BT,NREC,LTEST          !LOOK NEXT PART.
      IPRNT=IABS(NREC)                              !SET NEXT PART TYPE PTR.
C
  630 BACKSPACE NDISK                               !POSI DB FILE TO LOAD PART.

      IF(NSECF.EQ.0) NREC=IPRNT                     !(1ST SECT?) YES, FLG PROPER READ ST.
C
      CALL DSKRD                                    !LOAD PART.
C
      IF(LEFIL) GO TO 645                           !(EOF?)YES.
      IF(NREC.GT.0) GO TO 640                       !(MULTI SECT?)YES.
      NSECF=NSECF+1                                 !YES, INC PTR.

      IF(NSECF.EQ.1) GO TO 640                      !(1ST SECT?)YES.
      IF(NREC.EQ.-6) NSEG=-NSEG                     !(DRS?)YES, FLG DSKWR.

      IF(NREC.EQ.-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.EQ.5.AND.LSTSEC).OR.(IPRNT.EQ.8.AND.LSTRTE)) NSECP=0  !(LAST SECT OF MULTI SECT PSRT?)YES,ZERO REC CNT,
      GO TO READST                                                    !NEXT PART.

C
C     RESET ALL PARTS ACCOUNTING ( ALL PARTS DATA OVERWRITTEN )
C
  645 IALL=0                                                          !ASSUME ONLY ONE PART TYPE OVERWRITTEN.

      IF(.NOT.SNGLBS) 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
       JELG(I)=0                                                      !ZERO GEAR ASSIGNMENTS.

  692 CONTINUE
      GO TO 695                                                       !GO SET ENG-NOT-LOADED FLG.

  693 IF(IPRNT.EQ.4) NUMG=0                                           !(GEAR?)YES,ZERO # OF GEARS.

      IF(IPRNT.EQ.5) NACC=0                                           !(ACCESSORY?)YES, ZERO # OF ACCESSORIES,

  695 NPARTS(IPRNT)=0                                                 !FLG PART TYPE IPRNT NOT LOADED.

      IF(IALL.EQ.0) GO TO 697                                         !(ZERO ALL PARTS?)NO,

  690 CONTINUE                                                        !NEXT,
  697 IF(NDEL.EQ.0) GO 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 HEVSIM PARTS DATA BASE:')
 1620 FORMAT(1XA6'!'A10'['05','03']'3XA5,3XA10)
C
      END
```

```
*               **** DSKDIR ****
*
        SUBROUTINE DSKDIR
*
*  MODIFIED 6 JUNE 1980: SOMERS
*
C       ENTRY POINTS: DSKDIR
C
C       SUBROUTINES CALLED: CHKFIL, DSKCTR, DSKRD, ICRCNT, IGET,
C                           SLOOKP, NWRCNT, PRNOUT, PUT
C
C       CALLED BY: INPBAT
C
C*****************************************************************
C
C
        INCLUDE 'COMMS/NOLIST'
C
        LOGICAL LPAGE
C
        DOUBLE PRECISION TUN,UNMASK,TTUN
C
        DIMENSION HPART(14),DIRDAT(27),BPRN2(2),INDIR(14)
C
C
        DATA ( HPART(I),I=1,11) /'ENGIN','CONVE','VEHIC','GEAR'
        2,'ACCES','DRIVI','SHIFT','ROUTE','TIRE','TRANS','AXLE'/
        3,ISCR/2/,JCT/5/,IQMARK/63/
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=DNMASK(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,IALL).EQ.IQMARK) CALL PUT(UNMASK,II,0)  !(CHAR #II IN NAME UN WILD?) YES, MASK IT.
  402   CONTINUE                                         !NXT CHAR.
  403   IF(UNMASK.NE.DNMASK(10)) IWILD=1                 !(ANY WILD CHAR?) YES, FLG IT.
        CALL DAND(UN,UNMASK,TUN)                         !SET NAME MASK.
        IF(UT.EQ.USTAR) GO TO 404                        !(PART TYPE WILD?) YES.
        CALL SLOOKP(UT,NUMPAR,HPART,IPRNT,$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
        IGTOT=0                                                     ! GRAND TOTAL OF ALL PARTS
        NXTPG=1                                                     !INTI DUMP PG#.
        IDIR=IUNIT                                                  ! DIR UNIT
        IF(JPRNT.NE.0)GO TO 410                                     !(NEED SCR FILE FOR DIR?)NO.
        IDIR=ISCR                                                   !YES,SET UP FOR DIR TO SCR FILE..
        CALL DSKCTR (-21 , SEQOUT )                                 ! OPEN SCR FILE TO WRITE DIR ON
C
  410   IF(JT.NE.HSTAR) GO TO 415                                   ! (ALL PARTS?) NO.
        DO 480 IPP=1,NUMPAR                                         ! LOOP THROUGH ALL PART TYPES
        IPRNT=IPP                                                   !SET PART TYP PTR.
  415   IPSAV=IPRNT
        CALL CHKFIL(BASDEV(IPRNT),BASFIL(IPRNT),BASPPN(1,IPRNT),$417)  !SEE IF FILE IS ACCESSDATE
        GO TO 470                                                   !ERROR RETURN
  417   IF(JPRNT.GT.0)GO TO 419
        IF(IPRNT.EQ.1)IENG=1
        IF(IPRNT.EQ.4)NGPART=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(SNGLBS) IP=12                                            ! (SINGLE FILE DB?) YES, RESET FILE PTR.
        WRITE(IDIR,1400) HPART(IPRNT)                               ! DIR FILE HEADER
        1,DATE,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        IF(JPRNT.GE.0.AND.TTY.AND.(.NOT.LIMTTY))
        2       WRITE(JCT,1400) HPART(IPRNT)
        3,DATE,BASDEV(IP),BASFIL(IP),BASPPN(1,IP),BASPPN(2,IP)
        ILIN=3                                                      ! SET LINE COUNT
        INDIR(IPRNT)=IDIRPG                                         !SAVE DIR PG #.
        LPAGE=.TRUE.                                                ! SET FLAG TO PAGE
C
  420   READ(NDISK,END=460) BN,ET,NREC,CDATE,CHOUR,BPROT,BPPN2,DUMCOM
        IF ( ET.NE.HPART(IPRNT) )  GO TO 420                        ! (PART TYPE NEEDED?) NO, LOOK NXT.
        CALL DAND(BN,UNMASK,TTUN)
        IF(TTUN.NE.TUN) GO TO 420                                   ! YES,(GOOD NAME?) NO, LOOK NXT.
        ICNTT=ICNTT+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,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/F 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.BT) 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, FIG 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.
      NDRTE=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 /*D
SE/.
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(SNGLBS) 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 IGTOT=IGTOT+ICNTT                                     ! SUB TOT GRAND TOTAL

      ICNTT=0                                               ! ZERO PRT CNT

      GO TO 475
C
  470 IP=IPRNT                                              !SET DB FIL PTR.

      IF(SNGLBS) IP=12                                      !(SINGLE DBFIL?)YES, RESET PTR.

      WRITE(JCT,1470) HPART(IPRNT)                          !%NON-EXISTENT DB FIL.
C
  475 IF(DT.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(DT.NE.HSTAR) GO TO 490                             !(WILD PART?) NO, INDEX TO DIR NOT NEEDED.

      WRITE(IUNIT,1486) DATE,IDIRPG                         !INDEX TO DIR HEADER.
      N=0                                                   !INIT 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(IGTOT.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 IPRNT=1,NUMPAR
  685   IF(IPRNT.NE.1) GO TO 693
        NENG=0
        DO 692 I=1,20
         JENG(I)=0
  692   CONTINUE
        GO TO 695
  693   IF(IPRNT.EQ.4) NUMGT=0
        IF(IPRNT.EQ.5) NACC=0
  695   MPARTS(IPRNT)=0                                          !SET FLG PART NOT LOADED.

        IF(UT.NE.HSTAR) GO 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'['05','03']'/2X,72('-')/)
 1401   FORMAT(1XA10,1XA9,1XA5' <'03'> ['05','03'] '16A5)
 1402   FORMAT(1XA10,1XA9,1XA5' <'03'> ['05','03'] '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'['05','03']')
 1406   FORMAT(' NAME'7X'CREATION'8X'PROT  PPN'9X'PAGE COMMENT'
        1,62X'PAGE A-'I4/
        1,1X10('-')1X15('-')' -----'1X11('-')' ----'1X80('-'))
 1407   FORMAT(' NAME'7X'CREATION'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 ON 'I2' FILES.'/'1')
 1412   FORMAT(1XA10)
 1470   FORMAT(/' % DSKDIR- SPECIFIED 'A5' DB FILE NOT FOUND OR'
        1,' EMPTY.'/)
 1466   FORMAT('1'A10,105X'PAGE A-'I4///10X'PART TYPE    '
        1,'DIRECTORY PAGE NUMBER'/10X9('-')3X21('-')//)
 1467   FORMAT(12XA5,7X'A-'I4)
 1468   FORMAT(12XA5,7X'% NO PARTS ON FILE.')
        END
```

```
*               **** DSKRD ****
*
        SUBROUTINE DSKRD
*
*  MODIFIED 22 SEPT 1980: SOMERS
*       ENTRY POINTS: DSKRD
C
C       CALLED BY: DSK, DSKDEL, DSKDIR
C
C       EDIT HISTORY:
C
C       (717)/SS-11-6-78        ACCESSORY SPEED RATIO, FIX RPMS HERE RATHER THAN IN ENTERP.
C       (720)/SS-11-24-78       TAKE OUT OF VEHICLE AL REF TO TIR AND AXLE
C       (724)/SS-2-1-79         NEW SHIFT LOGIC PART DESCRIPTION FOR MULTISPEED AXLES
*            /LS-9-22-80        REVISED LOAD ACCESSORY DATA
C*************************************************************
C
C
        INCLUDE 'COMMS/NOLIST'
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),LOIES
        2,DISP,ICYL,IMIN,IMAX,THRMAX
        2,THRMIN,EINER,BORE,STROKE,FSPGR,NCYCLE,RPMAX(IENG)
        3,RPMIN(IENG),NRPM(IENG),ENMIN(IENG),LHPM,LPS,LTOR
        4,LBMFP,LHP,LLHHR,LBSFC,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,CONTOR,CDIAM,NTORP
        2,AI1,AI2,TIN,TOUT,SPIN,SOUT,NTD,AKD,SRD,TRD,TORBPK,NTBP
        GO TO 900
C
 1020   READ(NDISK,END=920) CNAME,BT,NREC,CDATE,CHOUR,CCPROT,CCPPN            !LOAD COAST CONVERTER.
        1,CCOM,SPARE,COAST,CONTOR,CDIAM,NTORP
        2,AI1,AI2,TIN,TOUT,SPIN,SOUT,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,(SPARE(I),I=1,4),AIP,WGT,CD,CDC,AREA,WLSG,
        2 BVG,BVGEFF
        GO TO 900
C
C*****************************************************************
C     LOAD GEAR DATA
C
  104   READ(NDISK,END=920) GNAME(NGEAR),BT,NREC,CDATE,CHOUR
        1,GPROT(NGEAR),GPPN(NGEAR),TCOM,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,APPN(NACC),ACOM,(SPARE(I),I=1,3),ACCSR(NACC),OUTCYC(NACC)
        2,AIAS(NACC),NNA(NACC)
        3,(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,DPPN
        2,DCOM,(SPARE(J),J=1,4),NAO,TO,DO,VO,AO,NGC,NSEG
        3,(TSEG(I),ASEG(I),VSEG(I),PWOT(I),ATHOLD(I),NGSEG(I)
        4,THRATE(I),DSEG(I),PCSEG(I),POSTSE(I),VELSEG(I)
        5,ITYSEG(I),I=1,NSEG)
        LSTSEC=.TRUE.                                             !ASSUME LAST SECT.

        IF(NREC.LT.0) LSTSEC=.FALSE.                              !(LAST SECT?)NO, RESET FLG.

        IF(NAO.EQ.0)NAO=1
        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),DSEG(I),PCSEG(I),POSTSE(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,SPPN
        1,SCOM,SPARE,GOVPSI,OUTRPM,NGPT,LVAC
        2,LENG,GDAT,LDETHT,NUMG,PARAB,LDETE,LDETV,GOVLIN,NUMBSL
        3,(IGF(I),IGT(I),IAF(I),IAT(I),SHFTIM(I)
        4,NSPTS(I),DETPT(I),DETRPM(I)
        5,(SHFTPT(J,I),SHFTRP(J,I),J=1,10),I=1,NUMBSL),LTHR       !(720)
        GO TO 900
C
C*****************************************************************
```

```
C     LOAD ROUTE DATA
C
  108   IF(NREC.LT.0) GO TO 1080                                    !1ST SECT TYPE?)NO.
        READ(NDISK,END=920) RNAME,BT,NREC,CDATE,CHOUR,RPROT,RPPN
        1,RCOM,SPARE,NRDIST,(RDIST(I),RGRADE(I)
        2,RCOEF(I),RVWIND(I),I=1,NRDIST)
        LSTRTE=.TRUE.                                             !ASSUME LAST SECT.

        IF(NREC.LT.0) LSTRTE=.FALSE.                              !LAST SECT?)NO, RESET FLG.

        GO TO 900
C
 1080   READ(NDISK,END=920) RNAME,BT,NREC,LSTRTE,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
  109   READ(NDISK,END=920) TNAME,BT,NREC,CDATE,CHOUR,TPROT,TPPN
        1,TCOM,(SPARE(I),I=1,4),FRC4,WRAD,FRC1,FRC2,TIREFF,AIW
        IF(FRC4.EQ.0.)FRC4=0.0004
        GO TO 900
C
C*************************************************************************
C     LOAD TRANSMISSION DATA
C
  110   READ(NDISK,END=920)TRNAM,BT,NREC,CDATE,CHOUR,TRPROT,TRPPN
        2,TRCOM,SPARE,NGTR,(GEANAM(I),GEANUM(I),I=1,NGTR)
        GO TO 900
C
C*************************************************************************
C     LOAD AXLE DATA
C
  111   CALL ZEROP(AXRPM,120) ; CALL ZEROP(AXTORQ,120)
        CALL ZEROP(NPAX,6)
        READ(NDISK,END=920)AXNAME,BT,NREC,CDATE,CHOUR,AXPROT,AXPPN
        2,AXCOM,SPARE,NAR,NHAX,NAXS,
        2((ENAR(I,K),NPAX(I,K),(AXRPM(J,I,K)
        3,AXTORQ(J,I,K),J=1,NPAX(I,K)),I=1,NHAX),K=1,NAXS)
C
        GO TO 900
C
C*************************************************************************
C
  920   DEFIL=.TRUE.                                                !FLG EOF ON DB READ FOUNDISK.

  900   RETURN                                                      !*DONE, BYE.

C
        END
```

```
*                **** DSKWR ****
*
      SUBROUTINE DSKWR
*
*   MODIFIED 22 SEPT 1980 19801 SOMERS
C     ENTRY POINTS: DSKWR
C
C     CALLED BY: DSK, DSKDEL
C
C     EDIT    HISTORY
C
C     (720)/SS-11-24-78       TAKE OUT OF VEHICLE AL REF TO TIR AND AXLE
C     (724)/SS-2-1-79         NEW SHIFT LOGIC PART DESCRIPTION FOR MULTISPEED AXLES
*          /LS-9-22-80        REVISED STORE ACCESSORY DATA
C
C*****************************************************************
C
C
      INCLUDE 'COMMS/NOLIST'
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.
C
        IE=IB+3
C
        WRITE (NDISK )  BN,BT,NREC,CDATE,CHOUR,EPROT(IENG),EPPN(IENG)
        1,ECOM,(SPARE(I),I=1,4),LDIES
        2,DISP,ICYL,IMIN,IMAX,THRMAX
        1,THRMIN,EINEP,BORE,STROKE,FSPGR,NCYCLE,RPMAX(IENG)
        2,RPMIN(IENG),NRPM(IENG),ENMIN(IENG),LRPM,LPS,LTOR
        3,LBMEP,LHP,LLHHR,IBSFC,LGALHR
        4,(NTOR(IENG,K),ERPM(IENG,K),K=1,20)
        5,(((EMAP(I,J,K),J=1,20),K= IB,IE),I=1,NRPM(IENG))
        GO TO 900
C
C*****************************************************************************
C     STORE TORQUE CONVERTER DATA
C
  102   IF ( COAST )  GO TO 1020
        WRITE (NDISK )  BN,BT,NREC,CDATE,CHOUR,CDPROT,CDPPN
        1,CCOM,SPARE,COAST,CONTOR,CDIAM,NTORP,AI1
        2,AI2,TIN,TOUT,SPIN,SOUT,NTD,AKD,SRD,TRD,TORBPK,NTBP
        GO TO 900
C
 1020   WRITE (NDISK )  BN,BT,NREC,CDATE,CHOUR,CCPROT,CCPPN
        1,CCOM,SPARE,COAST,CONTOR,CDIAM,NTORP,AI1
        2,AI2,TIN,TOUT,SPIN,SOUT,RTC,AKC,SRC
        GO TO 900
C
C*****************************************************************************
C     STORE VEHICLE DATA
C
```

```
  103   WRITE(NDISK) BN,BT,NREC,CDATE,CHOUR,VPROT,VPPN,VCOM
        1,(SPARE(I),I=1,4),AIP,WGT,CD,CDC,AREA,WLSG,
        1 BVG,BVGEFF
        GO TO 900
C
C*************************************************************************
C     STORE GEAR DATA
C
  104   WRITE(NDISK) BN,BT,NREC,CDATE,CHOUR,GPROT(NGEAR),GPPN(NGEAR)
        1,GCOM,SPARE,AIGIN(NGEAR),AIGOUT(NGEAR)
        2,GRAT(NGEAR),ERAT(NGEAR),NGRLSS(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)  BN,BT,NREC,CDATE,CHOUR,APROT(NACC),APPN(NACC)
        1,ACOM,(SPARE(I),I=1,3),ACCSR(NACC),OUTCYC(NACC)
        2,AIAS(NACC),NNA(NACC)
        3,(ACCS(I,NACC),ACCT(I,NACC),I=1,NNA(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.
C
        WRITE(NDISK) BN,BT,NREC,CDATE,CHOUR,DPROT,DPPN            !YES.
        1,DCOM,(SPARE(J),J=1,4),NAO,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),POSTSE(I),VELSEG(I)
        4,ITYSEG(I),I=1,NSEG)
        GO TO 900
C
 1060   NSEG=IABS(NSEG)                                           !GET RID OF FLG.
        WRITE(NDISK) BN,BT,NREC,LSTSEC,SPARE,NSEG,(TSEG(I)
        2,ASEG(I),VSEG(I),PWOT(I),ATHOLD(I),NGSEG(I)
        3,THRATE(I),DSEG(I),PCSEG(I),POSTSE(I),VELSEG(I)
        4,ITYSEG(I),I=1,NSEG)
        GO TO 900
C
C*************************************************************************
C     STORE SHIFT LOGIC DATA
C
  107   WRITE(NDISK) BN,BT,NREC,CDATE,CHOUR,SPROT,SPPN
        1,SCOM,SPARE,GOVPSI,OUTRPM,NGPT,LVAC,LENG
        2,GDAT,LDETNT,NUMG,PARAB,LDETE,LDETV,GOVLIN,NUMBSL
        3,(IGF(I),IGT(I),IAF(I),IAT(I),SHFTIM(I)
        4,NSPTS(I),DETPT(I),DETRPM(I)
        5,(SHFTPT(J,I),SHFTRP(J,I),J=1,10),I=1,NUMBSL),LTHR
        GO TO 900
C
C*************************************************************************
C     STORE ROUTE DATA
C
  108   IF(NRDIST.LT.0) GO TO 1080                                !(1ST SECT TO BE WRITTEN?)NO.
C
        WRITE(NDISK) BN,BT,NREC,CDATE,CHOUR,RPROT,RPPN            !YES.
        1,RCOM,SPARE,NRDIST,(RDIST(I),RGRADE(I)
        2,RCOEF(I),RVWIND(I),I=1,NRDIST)
```

```
      GO TO 900
C
1060  NPDIST=IABS(NRDIST)                                          !GET RID OF FLG.
      WRITE(NDISK) BN,BT,PREC,LSTREC,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   IF(FRC4.EQ.0.)FRC4=0.0004
      WRITE(NDISK) BN,BT,NREC,CDATE,CHOUR,VPROT,VPPN,TCOM
     1,(SPARE(I),I=1,4),FRC4,WRAD,FRC1,FRC2,TIREFF,AIW
      GO TO 900
C
C*************************************************************************
C     STORE TRANSMISSION DATA
C
110   WRITE(NDISK)TRNAM,BT,NREC,CDATE,CHOUR,TRPROT,TRPPN
     2,TCOM,SPARE,NGTR,(GEANAM(I),GEANUM(I),I=1,NGTR)
      GO TO 900
C
C*************************************************************************
C     STORE AXLE DATA
C
111   WRITE(NDISK)AXNAME,BT,NREC,CDATE,CHOUR,AXPROT,AXPPN
     2,AXCOM,SPARE,RAR,NRAX,NAXS,
     2((ERAR(I,K),NPAX(I,K),(AXRPM(J,I,K)
     3,AXTORQ(J,I,K),J=1,NPAX(I,K)),I=1,NRAX),K=1,NAXS)
      CALL ZEROP(NPAX,246)
      CALL ZEROP(ERAR,6)
      CALL ZEROP(RAR,3)
C
      GO TO 900
C
C*************************************************************************
C
900   RETURN                                                       !DONE, BYE..

C
      END
```

```
*               **** ENGINE ****
*
      SUBROUTINE ENGINE
*
*  MODIFIED 15 JULY 1980: SOMERS
*
C
C       ENTRY POINTS: ENGINE
C
C       CALLED BY: GOBACK, REMAP
C
C*****************************************************************
C
C
      DOUBLE PRECISION   UN
C
      LOGICAL PRINT6,LWOT,LTHR
      COMMON /GET/ UT,UN,NUG(20),JENG(20),IENG
      COMMON /ENGMAP/ RPMAX(2),RPMIN(2),NRPM(2),RPME,TORQE,FRATE,
     1VAC,THR,MAPOK,IERRE,NTOR(2,20),EMAP(20,20,8),ERPM(2,20),
     2ENMIN(2),SPIDLE(2),TORQEO,LTHR
      COMMON /TMAP/ TORQ,LWOT,IWOT,TMIN
*
      PRINT6 = .FALSE.
*     PRINT6 = .TRUE.
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
      MAPOK=1
      IF(RPME.GE.RPMAX(IENG)) GO TO 10
      IF(RPME.GT.RPMIN(IENG) .OR.
     1  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
      MAPOK=2
      GO TO 40
C
C    ENGINE SPEED OFF MAP IN UPPER DIRECTION
C
   10 I=NRPM(IENG)
      MAPOK=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,
```

```
     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))
      LLL = 0
      IF(LWOT) GO TO 1999
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))
      IERRE=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)
      P6=P2-C1*(P2-P1)
      V6=V2-C1*(V2-V1)
      T6=T2-C1*(T2-T1)
      TO6=TO2-C1*(TO2-TO1)
      N=21-NTOR(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)
      C3=C0
C
C     INTERPOLATE ENGINE PARAMETERS BETWEEN SPEED SETTINGS IF ON MAP
C
      TORQ=TO5-C3*(TO5-TO6)
      IF(TORQ.GT.THOT) 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)
      LLI = 1
*
*  SPECIAL PRINT
```

```
*
 1999   CONTINUE
        IF(.NOT. PRINTE) RETURN
        WRITE(6,2060) FRATE,THR,VAC,TORQ,TWOT,TMIN,MAPOK,LLL
 2060   FORMAT(' $ ENGINE - FRATE,THR,VAC,TORQ,TWOT,TMIN,MAPOK,LLL ->',
        1 /6G,2I2)
        WRITE(6,2050) RPME,IENG,IM,K,J,ERPM(IENG,IM),ERPM(IENG,I),
        1 EMAP(IM,1,K1),EMAP(I,1,K1),EMAP(IM,20,K1),EMAP(I,20,K1)
 2050   FORMAT(' $ ENGINE - RPME,IENG,IM,K,J,ERPM(IENG,IM),ERPM(IENG,I),
        1 EMAP(IM,1,K1),EMAP(I,1,K1),EMAP(IM,20,K1),EMAP(I,20,K1) ->',
        2 /XF6.1,2X4I3,2G/4G)
*       WRITE(6,2010) T1,T2,T3,T4,T5,T6
 2010   FORMAT(' $ ENGINE - T1,T2,T3,T4,T5,T6 ->',6G)
*       WRITE(6,2020) V1,V2,V3,V4,V5,V6
 2020   FORMAT(' $ ENGINE - V1,V2,V3,V4,V5,V6 ->',6G)
*       WRITE(6,2000) F1,F2,F3,F4,F5,F6
*       WRITE(6,2030) TO1,TO2,TO3,TO4,TO5,TO6
 2000   FORMAT(' $ ENGINE -  F1, F2, F3, F4, F5, F6 ->',6G)
 2030   FORMAT(' $ ENGINE - TO1,TO2,TO3,TO4,TO5,TO6 ->',6G)
*       WRITE(6,2040) C0,C1,C2,C3
 2040   FORMAT(' $ ENGINE - C0,C1,C2,C3    ->',4G/)
*
        RETURN
C
C     FOLLOWING ARE INTERPOLATIONS OR 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 988
        MAPOK=4
        LLL = 2
        GO TO 1999
  988   MAPOK=MAPOK+4
        LLL = 3
        GO TO 1999
   99   FRATE=EMAP(I,20,K2)+C0*(EMAP(IM,20,K2)-EMAP(I,20,K2))
        THR  =EMAP(I,20,K3)+C0*(EMAP(IM,20,K3)-EMAP(I,20,K3))
        VAC  =EMAP(I,20,K4)+C0*(EMAP(IM,20,K4)-EMAP(I,20,K4))
        TORQ=TWOT
        IF(MAPOK.GT.1) GO TO 999
        MAPOK=5
        LLL = 4
        GO TO 1999
  999   MAPOK=MAPOK+6
        LLL = 5
        GO TO 1999
        END
```

```
*              **** ENTERP ****
*
      FUNCTION ENTERP(ACCT,ACCS,NPTS,NTBL,RPM,NLEN,DUTY)
C
*  MODIFIED 22 SEPT 1980 SONERS
*C     INTERPOLATE ALL DATA CURVES TO COMPUTE SUM OF SPECIFIED POINTS
C
C
C     ENTRY POINTS: ENTERP
C
C     CALLED BY: GOBACK,SIMCTR
C
C     EDIT HISTORY
C
C     (717)/BS-11-6-78        ADD DUTY CYCLE CODE.
*          /LS-9-22-80        REVISED TORQUE COMPUTATION
C
C*************************************************************
C
      DIMENSION ACCT(NLEN,NTBL),ACCS(NLEN,NTBL),NPTS(NTBL),DUTY(NTBL)
C
      ENTERP=0.                          | ZERO ACCUM
      IF(NTBL.LT.1) RETURN               | (ANY TABLES?) NO.
C
      DO 30 I=1,NTBL                     | YES, LOOP THROUGH TABLES.
C
      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.
C
      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
C
      IF(RPM.LE.ACCS(K,I)) GO TO 20                                  | (GOT IT?) YES.
10    CONTINUE                                                       | NO.
C
11    K=J                                                            | ABOVE MAX SPECIFIED
C
      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
      ACCSKI = ACCS(K,I)
      ACCTKI = ACCT(K,I)
      TOR = ACCTKI + (RPM - ACCSKI)*(ACCT(KM,I)
     1 - ACCTKI)/(ACCS(KM,I) - ACCSKI)
      IF(DUTY(I) .EQ. 0.0) DUTY(I) = 1.0
C
      IF(TOR.GT.0.) ENTERP=ENTERP+TOR*DUTY(I)  |(VALID?) YES,ADD PT FROM CURRENT TABLE TO TOTAL
30    CONTINUE                                                       | NO, ASSUME ZERO.
C
```

END

```
*               **** GETACL ****
*
      SUBROUTINE GETACL(TITER)
*
*  MODIFIED 10 JUNE 1980: SOMERS
*
C     ENTRY POINTS: GETACL
C
C     CALLED BY: ITERAT
C
C*****************************************************************
C
      INCLUDE 'COMMS/NOLIST'
      DOUBLE PRECISION TEMP,C1,C2,C3,C4,C5,C6,C7,C8,C9,C10
     1,C11,C12,C13,C14,C15,C16,C17,C18
*
*
*     GO TO 10
*
*  SPECIAL COMPUTATION FOR ACCEL.
*
      XYZ = 0.75*BVG*BVGEFF*TR
      ACCEL = (XYZ*(TITER - TORQA)*GRAT(NGEAR)*RAR(MAX)/WRAD
     1  -AA1 - AA2*VOLD - AA3*VOLD**2 - BPER*WGT)/AA4
      RETURN
*
   10 CONTINUE
      PRINT6 = .TRUE.
      IF(IDEBUG.GE.6.AND.CONT.GE.DBEGIN.AND.
     2       (CONT.LE.DSTOP.OR.ISEG2.EQ.0))PRINT6=.TRUE.
      C1=DT/1.4666666666666667D0
C
      C2=AA3*C1**2
C
      C3=(AA2*C1+2*AA3*C1*VOLD+AA4)
C
      C4=(AA1+AA2*VOLD+AA3*VOLD**2+BPER*WGT)
C
      AEFFG=ERAT(NGEAR)
      AGRAT=GRAT(NGEAR)
      C5=TR*AGRAT*AEFFG*AA8/WRAD
C
      IF(SR.LT.1.E-3)SR=1.E-3
      C8=AGRAT*RAR(MAX)*AA5/SR
C
      C9=C8*C1
C
      C10=C8*VOLD
C
      C11=BB1*(EINER+AIA)
C
      C12 = BB1*(WLSG*AIW + RARSQ*(AIP + (AIGIN(NGEAR)
     1 + AIGOUT(NGEAR))*AGRAT*AGRAT))
C
      C13=AA5*VOLD
C
      C14=AA5*C1
C
      C15=C11*(C10-RPMEO)/DT
C
      C16=C11*C9/DT
```

```
C17=C12*(C15-WRAD)/DT
C
      C18=C12*C14/DT
C
      C6=(C3+C16*C5/TR+C18/WRAD)/C2
C
      C7=((C4-((TITER-TORQA-C15)/TR*C5))+(C17/WRAD))/C2
C
      TEMP=DSQRT(C6**2-4.*C7)
C
      ACCELM=(-C6-TEMP)/2.
      ACCELP=(-C6+TEMP)/2.
C
      IF(PRINT6)WRITE(6,9001)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(PRINT6)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
```

```
*               **** GOBACK ****
*
      SUBROUTINE GOBACK
*
*  MODIFIED 10 JUNE 1980: SOMERS
*
C     ENTRY POINTS: GOBACK
C
C     SUBROUTINES CALLED: CONVTR, CTRLD, DEBUG, ENGINE,
C                         ENTERP, EXIT, SIMSTS, TTYINP
C
C     CALLED BY: ITERAT, SIMCTR, SIMINT
C
C     EDIT HISTORY
C
C     [607]/SS-8-10-78        CLUTCH
C     [611]/SS-6-22-78        NO BRAKES IF COMING FROM ITERAT
C     [615]/SS-7-17-78        IF IDLEING ON A GRADE, PUT ON THE BRAKES
C
C*********************************************************************
C
C
C
      INCLUDE 'COMMS/NOLIST'
      EXTERNAL SIMSTS
C
C
      LOGICAL PRINT6
      DATA HNORM/'NORML'/,HCOAST/'COAST'/,M20/20/,MXCALL/500/
     1,HGOBER/'GOBER'/
      DATA NAME/'GOBAK'/
C
C*********************************************************************
C
*
      PRINT6 = .FALSE.
*     PRINT6 = .TRUE.
*
      NGOCAL=NGOCAL+1                                   !INC CNT OF CALLS THIS DT.

      IF(NGOCAL.GT.MXCALL) GO TO 999                    !(#OF CALLS THIS DT EXCEEDED MAX?) YES.

C
      LSTOP=.FALSE.                                     ![615]
      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 IF(ABS(ACCEL).LT.1.E-5 .AND. V.LT.1.E-5)LSTOP=.TRUE.    ![615]
*
      CALL TAPEWR(NAME,1)
*
      AGRAT=GRAT(NGEAR)
      AEFFG=ERAT(NGEAR)
      FROLL = (AA1+AA2*V)*ROADC + AA11
      IF(FROLL.LT.1.E-30)FROLL=0.
      VTOT=V+VWINDC
```

```
      VTOTSQ=VTOT*VTOT
      PSI = 0
      IF(VTOTSQ.GT.1.E-30) PSI=ATAN(VWINDS/VTOT)
      FACCEL=ACCEL*AA4
      FGRADE=BPER*WGT
      RPMW=AA5*V                                          !CALC RPM OF WHEELS.

      IF(LDYNA) GO TO 4                                   !(DYNO SIM?) YES.

      FAERO=AA3*VTOTSQ*(1.+CDC*PSI)                       !NO, CALC AERO DYANMIC DRAG.

      GO TO 5
    4 CONTINUE
      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-RPMWO                                    !CALC DIF BETWEEN RPM OF WHEELS AND OLD VAL.

      TRR=0.0
      IF(LTRRZ) GO TO 15                                  !(TRR ALWAYS .0?) YES.

      AI1 = SAI1; AI2 = SAI2
      IF(.NOT.LOCKUP(NGEAR)) GO TO 10                     !(GEAR LOCKEDUP ?) YES.

      AI1 = 0.0; AI2 = 0.0
   10 CONTINUE
      TRR=BB1*(WLSG*AIW+RARSQ*ERAR(1,NAX)*(AIP+AIGOUT(NGEAR)   !CALC TRR FOR UNLOCKED GEAR.
     1  +AGRAT*AGRAT*AEPFG*(AI2+AIGIN(NGEAR))))*DRPMW/DT
*     GO TO 15
C
*     TRR = BB1*(WLSG*AIW + RARSQ*(AIP + (AIGIN(NGEAR)    !CALC TRR FOR LOCKED GEAR
*     1 + AIGOUT(NGEAR))*AGRAT*AGRAT))*DRPMW/DT
C
   15 TORQW=WRAD*FWHEEL+TRR                               !CALC TORQUE AT WHEELS.

*
      CALL TAPEWR(NAME,2)
*
      IF(PRINT6)WRITE(6,2000) V,ACCEL,FWHEEL,FROLL,FAERO,FACCEL,
     1    FGRADE,TRR,TORQW
 2000 FORMAT(' $GOBACK - V,ACCEL,FWHEEL,FROLL,FAERO,
     1FACCEL,FGRADE,TRR,TORQW ->',/9G)
C
C     USING WHEEL TORQUE AND RPM , COMPUTE BACK THROUGH DIFFERENTIAL
C     AND REAR BOX TO THE TORQUE CONVERTER
C
      RPMP=RAR(NAX)*RPMW
      RPMC=AGRAT*RPMP                                     ![607]
      IF(.NOT.LSTRUP.OR..NOT.LCLTCH)GO TO 14
      IF(.NOT. LOCKUP(NGEAR)) GO TO 11
      IF(RPMC.LT.RPM2)GO TO 16
   11 CONTINUE
      RPM2 = RPMC
      IF(VOLD < 5.0) GO TO 16
      LSTRUP=.FALSE.
      LCLTCH=.FALSE.
*
      CALL TAPEWR(NAME,3)
```

```
*
      GO TO 16
C
14    RPM2=RPMC                                                    ![607]
*
      CALL TAPEWR(NAME,4)
*
      IF(.NOT.LCLTCH)GO TO 16                                      ![607]
      RPM2=RPM20+DRPMC*DT                                          ![607]
C
      IF(LDNSHF)GO TO 19                                           ![607](DOWNSHIFT?)YES.

      IF(RPM2.GT.RPMC)GO TO 16                                     !(SHIFT OVER?)NO.
18    RPM2=RPMC                                                    !YES.
      LLSH=.FALSE.
      LCLTCH=.FALSE.
      GO TO 16
19    IF(RPM2.LT.RPMC)GO TO 16
      GO TO 18
C
16    TORQP=TORQW/AA8+ENTERP(AXTORQ(1,1,MAX),AXRPM(1,1,MAX)
      1,MPAX(1,MAX),MAX,RPMP,M20,RZERO)
*
      TORQ2=TORQP/(AGRAT*AEFFG)+ENTERP(GRTORQ(1,NGEAR),GRPM(1,NGEAR)
      1,NGRLSS(NGEAR),1,RPM2,M20,RZERO)
*
      CALL TAPEWR(NAME,5)
*
      IF(SHFTNG.OR.LSTRUP) GO TO 17                                ![613](SHIFTING.OR STARTING ?) YES, DON'T CHG COAST/DRIVE ST
ATE.
      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
  996 CALL CONVTR                                                  !GET SR+TR RATIO.

*
      CALL TAPEWR(NAME,6)
*
      IF(SR.GT.0.) GO TO 20                                        !(SPEED RATIO 70.?) YES.

C
C     FOR NEGATIVE SPEED RATIO USE MIN ENGINE SPEED AND MIN SPEED
C
      RPME=ENMIN(IENG)
      SR=SRD(1)
      TR=TRD(1)
      TORQ1=0.
      GO TO 25
   20 RPME=RPM2/SR*DVG
      TORQ1=TORQ2/TR
   25 CONTINUE
```

```
      RPM1=RPME/BVG
*
      CALL TAPEWR(NAME,7)
*
      IF(RPME.GE.ENMIN(IENG).OR.LCLTCH) GO TO 30                    !(RPM GE MAP OR CLUTCH IN ?) YES.
      RPME=ENMIN(IENG)                                              !SET ENG RPM TO MIN ENG RPM.

      RPM1 = RPME/BVG
   29 SR   = RPM2 / RPM1
   30 IF ( V.LT.1.E-5 )  RPME = ENMIN(IENG)                         !VEL 0.?) YES, SET ENG RPM TO MIN ENG RPM.
*
      CALL TAPEWR(NAME,8)
*
      IF(.NOT.LSTRUP)GO TO 32                                       ![ 613 ]
      IF(RPME.LT.ENMIN(IENG))RPME=ENMIN(IENG)                       ![ 613 ]
      RPM1=RPME/BVG
   32 DRPME = RPME - RPMEO
C     COMPUTE TORQUE USED BY ACCESSORIES
C
      TORQA=0.
      IF(NACC.GT.0)
     2TORQA=ENTERP(ACCT(1,1),ACCS(1,1),NNA(1),NACC,RPME,N20,DUTCYC)
C
C     COMPUTE FRONT END ROTATING INERTIAS
C
      TORQF=0.0                                                     !SET.
      IF(V < 0.1) GO TO 35
      IF(LTERZ) GO TO 35                                            !(TORQF ALWAYS 0.) YES.

      IF(LOCKUP(NGEAR)) GO TO 33                                    !(GEAR LOCKED UP?) YES.

      TORQF=AA9*DRPME/DT                                            !CALC FOR UNLOCKED GEAR.

      GO TO 35
   33 TORQF=BB1*(EINER+AIA)*DRPME/DT                                !CALC FOR LOCKED UP GEAR.
   35 CONTINUE
*
      CALL TAPEWR(NAME,9)
*
      IF(LCLTCH) GO TO 40                                           !(CLUTCH IN?) YES.
      TORQE=TORQA+TORQF+TORQ1/BVG/BVGEFF                             CALC TORQUE OF ENG.

      GO TO 70
   40 IF(LDNSHF) TORQF=ATORQF                                       !(DOWN SHIFTING?) YES, ADD TORQUE FOR ENG SPIN UP.

      TORQE=TORQA+TORQF                                             !CALC TORQUE OF ENG.

      IF(LSTRUP)TORQE=TORQE+TORQ1/BVG/BVGEFF
C
C     DETERMINE STATE OF THE ENGINE
C
   70 CALL ENGINE
C
*
      CALL TAPEWR(NAME,10)
*
      IF(.NOT.LCLTCH.OR.LDNSHF.OR.LSTRUP) GO TO 75                  !(CLUTCH OUT OR DOWN SHIFTING?) YES.
      TORQE=TMIN                                                    !SET ENG TORQUE.

      TORQF=TMIN                                                    !SET FRONT END TORQUE.
```

```
   75   IF(IDEBUG.EQ.1)GO TO 78
        CALLER=HNORM
        IF(LSTROP)CALLER='START'
        IF((IDEBUG.GT.2.AND.IDEBUG.LT.5).OR.IDEBUG.EQ.7.OR.
        2       (IDEBUG.EQ.2.AND.SHFTNG)) CALL DEBUG (CALLER)          !(DEBUG PRINT OUT?) YES, DO IT.
   78   IF((.NOT.COAST).OR.(TORQE.GE.TMIN).OR.LCLTCH
        2       .OR.LITER) GO TO 99                                    ![611]

        ABR=TORQW
        LBRAKE=.TRUE.
        TORQW=TORQW-(TORQE-TMIN)*AGRAT*AEFFG*AA8*TR*BVG*BVGEFF
        ABR=TORQW-ABR
        TORQP=TORQW/AA8
        TORQ2=TORQP/(AGRAT*AEFFG)
        TORQ1=TORQ2/TR
        TORQE=TORQA+TORQF+TORQ1/BVG/BVGEFF
        MAPOK=MAPOK-4
*
        CALL TAPEWR(NAME,11)
*
        IF((IDEBUG.GT.2.AND.IDEBUG.LT.5).OR.
        2       IDEBUG.EQ.7) CALL DEBUG (HCOAST)                       !(DEBUG ?) YES, DO IT.
C
   99   IF(.NOT.LSTOP)GO TO 110                                         ![615]
        ABR=TORQW                                                      ![615]
        LBRAKE=.TRUE.                                                  ![615]
        TORQW=0.
        TORQP=0.
        TORQ2=0.
        TORQ1=0.
        TORQE=TORQF+TORQA
        CALL ENGINE
*
        CALL TAPEWR(NAME,12)
*
        IF((IDEBUG.GT.2.AND.IDEBUG.LT.5).OR.
        2       IDEBUG.EQ.7) CALL DEBUG ('IDLE')                       !(DEBUG ?) YES, DO IT.
  110   SHFTNG=LLSH                                                    ![607]TURN OFF WHEN LEAVING

        IF(TTY) CALL TTYINP(SIMSTS)                                    !(JCT IS TTY? )) YES, IF INTERRUPT FOUND GO HANDLE IN SIMST.

        RETURN                                                         !*DONE, BYE.

C
  999   WRITE(JCT,1999) MXCALL                                         !"?FAILURE TO CONVERGE."

        CALL CTRLD(HGOERR)                                             !DEBUG TO JCT.
        IF(PTY) CALL EXIT                                              !(JCT IS PTY?) YES, OH WELL BETTER LUCK NEXT TIME.

        CALL RESETM                                                    !NO,"*RESET" TO JCT, REINT FOROTS, + REENTER VSMCTR.

C
 1999   FORMAT(/' ? GOBACK - FAILURE TO CONVERGE IN 'I4' ATTEMPTS'/)
        END
```

```
      SUBROUTINE HLPCMD
C
C     ENTRY POINTS: HLPCMD
C
C     SUBROUTINES CALLED: ASCIZ, CHKFIL, CRLF, ICRCNT, SLOOKP, SKPREC
C
C     CALLED BY: INPDIA
C
C*****************************************************************
C
      INCLUDE 'COMMS/NOLIST'
      DOUBLE PRECISION DHLPFL,INF,HELPF
      DIMENSION LINE(16),HLPFIL(2)
      INTEGER HLPPPN(3)
      LOGICAL FOUND
C
      EQUIVALENCE (HLPFIL,DHLPFL)
C
      DATA FOUND/.FALSE./,HELPF/'VEHSIM.HLP'/, HLPFIL(2)/'D.HLP'/
C
      DATA 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=1,40)
80    FORMAT(' THESE ARE THE COMMANDS',4(/10(1X,A5)))
      GO TO 20
C
100   CALL SLOOKP(WORD,40,HCOMND,ICMD,$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. "416 32
      CALL CHKFIL(MASDEV,HELPF,MASPPN,$170)
      GO TO 120
170   OPEN(UNIT=25,DEVICE=MASDEV,ACCESS='SEQIN',FILE='VEHSIM.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='VEHSIM.HLP'
        1,DIRECTORY=HLPPPN)
560     READ(25,200,END=120)INF,LEND
        DO 580 J=1,LEND
        READ(25,220)LINE
        CALL ASCIZ(LINE,16,NDUM)
        CALL CBIF
580     CONTINUE
        CALL CRLF
        CALL CRLF
        GO TO 560
        END
```

```
      FUNCTION ICRCNT(STRING,LWRDS)
C
C     ENTRY POINTS:    ICRCNT
C
C     SUBROUTINES CALLED:      RCRCNT
C
C     CALLED BY:       DSKDIR, HLPCMD, INPBAT, INPDIA, VALID2
C
C*****************************************************************
C
      DIMENSION STRING(LWRDS)
C
      ICRCNT=RCRCNT(STRING,LWRDS)                    !INTEGER CALL OF RCRCNT.
      RETURN                                         !BYE.
      END
```

```
*               **** INPBAT ****
*
      SUBROUTINE INPBAT  ( IECOND , ENDD , NLSTCF )
*
*  MODIFIED 23 SEPT 1980 SOMERS
C     EDIT HISTORY
C
C     (612)/SS-6-23-78         MODIFY DYNAMOMETER HORSE POWER
C     (716)/SS-11-3-78         DIESEL STUFF
C     (717)/SS-11-6-78         MODIFY ACCESSORY DUTYCYCLE.
C     (720)/SS-11-24-78        TAKE OUT OF VEHICLE AL REF TO TIR AND AXLE
C     (723)/SS-1-8-79 SUBSTITUTE CALLS TO SLOOKP FOR LOOKUP.
C
C
      INCLUDE 'COMMS/NOLIST'
C
      LOGICAL LSAVE,NLSTCF,ENDD,ENG1,ENG2,LSIMHL
     2,LSIMDM,LFULL,LSKIP
C
      DIMENSION CARD(16),HPART(14),CNVTYP(2),MFLG(NMOD)
     1,OLDVAL(NMOD),VALNEW(NMOD),HMOD(NMOD)
     2,HLIMT(5),OLDDTY(20)
C
      INTEGER DEBTAB(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),HTHR),(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(24),HDIES)
C
      DOUBLE PRECISION CNVNAM(2),HNLOAD,DATE1,HPVEHI
     2,HINPBA,RNAMEO,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,HPVEHI/'$VEHICLE'/,HPDYNO/'$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 HDTHRO/5HDTHRO/
C
      DATA   HENGIN/5HENGIN / , HFUEL /5HFUEL  / , HON/'ON'/
C
      DATA   HGALHP/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  /
C
      DATA   HOFF  /5HOFF    / , HOUTPU/5HOUTPU / , HPISTO/5HPISTO /
C
      DATA HDIES/'DIESE'/
C
      DATA   (HPART(I),I=1,11) /5HENGIN ,5HCONVE ,5HVEHIC ,5HGEAR
     2,5HACCES , 5HDRIVI ,5HSHIFT ,5HROUTE, 5HTIRE  ,5HTRANS
     3 ,'AXLE'   /
     4,DSTAR/'*     '/   ,HCOM/'!'/
C
      DATA HPARTS/'PARTS'/,HWHEEL/'WHEEL'/
     1,HTIRE/'TIRE'/,HMOD(20)/'PHI'/
     2,HMOD(21)/'DYNAM'/,HMOD(22)/'DUTYC'/  ,HMOD(23)/'C4'/
C
      DATA   HREAR /5HREAR  / , HROUTE/5HROUTE / , HRPM  /5HRPM   /
C
      DATA   HSEC  /5HSEC   / , HSEGME/5HSEGME / , HSHIFT/5HSHIFT /,
     1        HSLASH/5H  /   / , HSPEED/5HSPEED / , HSTAR /5H*     / ,
     2        HSTEP /5HSTEP  / , HSTROK/5HSTROK / , HSUMMA/5HSUMMA /
C
      DATA   HTHROT/5HTHROT / , HTIRE /5HTIRE  / , HTORQU/5HTORQU /
      DATA   HTIME /5HTIME  /, HCONTI/'CONTI'/
C
      DATA   HUPSHI/5HUPSHI / , HVACHU/5HVACUU / , HVEHIC/5HVEHIC /
C
      DATA   HWEIGH/5HWEIGH / , HWIND /5HWIND  /,HMILEP/'MILEP'/
C
      DATA HBATCH/'BATCH'/,HTRUCK/'TRUCK'/,HCAR/'CAR'/,HBUS/'BUS'/
C
      DATA HLIMT/'SEGME','MILE','SECON','OFF','SUMMA'/
C
C**********************************************************************
C
C     INITIALIZE LOGICAL FLAGS FOR PRINT LIMITATION AND END OF INPUT
C     DATA ON FIRST PASS THROUGH
C
20    ISTART=IECOND                                    !GET CTRFIL STATUS.

      LSKIP=.FALSE.
      ISKIP=0
      IECOND=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 ONTO LPT FILE
C
40    IF(NLSTCF) GO TO 100                             !(LIST CTRFIL ON LPT?) NO.

      WRITE(6,4980)                                    !HEADER TO LPT.

60    READ(1,5000,END=80) CARD                         !READ CTR REC(EOF?) YES.

      WRITE(6,5040) (CARD(I),I=1,NWRCNT(CARD,16))      !NO, WRITE REC.
      GO TO 60                                         !NEXT.
```

```
C
 80     REWIND 4                                  |POSI FOR EXE.
100     LSIMDM=.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.
        LSIMDM=.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(DIALOG) RETURN                         |(DIALO MODE?) YES.
        IF(ENDE) 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)GO TO 200                   |ARE WE SKIPPING?
        IF(COL1.NE.HSTAR)GO TO 180                |GOT A CONTROL CARD?
        ISKIP=ISKIP+1                             |YES, INCREASE COUNTER
        IF(ISKIP.LE.NSKIP)GO TO 180               |HAVE WE SKIPPED ENOUGH?
        LSKIP=.FALSE.                             |YESS
200     WRITE(5,5020) COL1,COMND,(CARD(K),K=1,NWRCNT(CARD,15))   |WRITE CTR REC ON TTY.
C
C     DETERMINE COMMAND ON COMMAND CARD (* IN COLUMN 1)
C
220     BACKSPACE 4                               |POSI CTRFIL.
        IF(COL1.NE.HSTAR) GO TO 260               |(IS REC COMD?) NO.
        LCMD=ICMD                                 |YES, SAVE COMD CODE OF LAST COMD.
        CALL SLOOKP(COMND,40,HCOMND,ICMD,6340)    |(KNOWN COMD?) YES.
C
C     PRINT COMMAND INVALID MESSAGE AND GO TO NEXT COMMAND
C
240     IF(DIALOG) RETURN                         |NO, (DIALO MODE?) YES.
260     IF(COL1.NE.HCOM)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) GO TO 320                         |(JCT IS TTY?) YES.
```

```
      NOKUN=NORUN+1                                          !NO, INC FATAL ERR CNT.
      GO TO 100                                              !NEXT REC.
C
C     BACKSPACE FOR USE BY INPUT ROUTINE WHEN OTHER ERRORS DETECTED
C
300   IF(DIALOG) RETURN                                      !(DIALOG MODE?) YES.
      BACKSPACE 4                                            !POSI CTRFIL TO BAD REC.
      GO TO 260                                              !GO PRI IT.
C
320   IF(DIALOG) RETURN                                      !(DIALO MODE?) YES.
      WRITE(5,5040) (CARD(K),K=1,1)                          !PRI ON TTY BAD REC.
      WRITE(5,5100) HCOMND(ICMD)                             !"%CTR ERR SWITCH TO DIALO MODE".
      GO TO 380                                              !GO TO USER FOR HELP.
C
C     GO TO PROCESS COMMAND
C
C        COMMAND          LABEL          COMMAND          LABEL
C        -------          -----          -------          -----
C
C        BATCH             380           OUTPUT            440
C        ACCESSORY        2760           REMAP            3760
C        AXLE             2400           RESET            4900
C        DEBUG             400           ROUTE            2960
C        DELETE           1200
C        DRIVING SCHEDULE 2860           SHIFT LOGIC      2780
C                                        SIMULATE          500
C        DUMP             1300           S.R. CONVERTER   3140
C        ENGINE           3320           STATUS           4620
C        FULL CONVERTER   3120           TIME             3980
C        GEAR             2700           TITLE             480
C                                        TRANSMISSION     2060
C        LIMIT PRINT      1340           UNLOCK CONVERTER 2100
C        LOCKUP CONVERTER 2100           USE              4000
C        MODIFY           1400           VEHICLE          2220
C        PRINT UNITS      1260           ZERO              960
C
340   GO TO (4000,500,1400,1260,1340,1300,3760,2100,2100,480,
     1       240,4620,960, 240, 240, 380,3320,3120,3140,2220,
     2      2700,2760,2780,2860,2960,3980, 400,4900, 240, 240,
     3       240,1200, 240, 240, 240, 240, 240, 440,2060, 2400) , 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(IECOND)                                    !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     IECOND=2                                                !ASSUME ILLEGAL DEBUG COMMAND.

        IDBUGO=IDEBUG
        CALL SLOOKP(WORD,NDEB,DEBTAB,IDEBUG,$430)               !LOOKUP DEBUG COMMAND.
          GO TO 300                                             !(VALID DEBUG COMMAND?)NO.

430     IECOND=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.
        IECOND=1                                                !SET NO ERR FLG.

        SDEBUG=DEBTAB(IDEBUG).AND.-1                            !SAVE DEBUG COMMAND FOR T*STATUS/.

        GO TO 160                                               !DONE.
C
C*****************************************************************
C
C     /OUTPUT/ COMMAND - GET NEW OUTPUT FILE SPECS
C
        ENTRY OUTCMD
440     ITASK=-1
        CALL CLSLPT(ITASK)
        IF(BATCH)GO TO 442
        WRITE(5,10400)
        READ(5,5000)FSPECS
        GO TO 444
C
442     READ(4,5160)FSPECS
C
444     CALL FILSPC(FSPECS,LPTDEV,LPTFIL,LPTPPN)
        ITASK=-1
        CALL OPNLPT(ITASK)
        GO TO 160
C
C*****************************************************************
C
C     /TITLE/ COMMAND - READ RUN TITLE AND DATE
C
        ENTRY TTLCMD(IECOND)                                    !DIALOG ENTRY.
        READ(5,5180,END=440) TITLE,DATE1                        !GET TITLE FROM JCT.
        IECOND=1                                                !SET NO ERR FLG.

        GO TO 160                                               !DONE.
C
470     IECOND=2
        GO TO 300
C
480     READ  (4,5160)  TITLE,DATE1                             !READ TITLE + OVERRIDE DATE.
```

```
490     IF ( DATE1.NE.BBLANK) DATE  = DATE1                    |(OVERRIDE DATE?)YES.
        GO TO 100
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.

        IECOND=3                                               |SET ERR FLG.

        GO TO 540

C
500     READ  (4,5280)  DATA(1),DATA(2)                        |READ SIMULATE CARD.
        IF(DATA(1).EQ.HBLANK .OR. DATA(1) .EQ. HTRUCK) GO TO 520   |SIMODE FIELD BLANK?)YES.
        IF(DATA(1) .EQ. HBUS)GO TO 510                 |(BUS MODE?)YES
        WRITE(JCT,5290)DATA(1)                                 |NO, REPORT, BUT GO ON ANYWAY

510     SIMODE=DATA(1)                                         |YES, SET MODE.

520     IF(PTY.OR.(DATA(2).EQ.HBLANK).OR.(DATA(2).EQ.HCONTI)
        1 .OR.(DATA(2).EQ.BATCH))  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 ( NORUN.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 INPDIA.

        IF ( NPARTS(I).GT.0 )  GO TO 580                       |(PARTS LOADED?)YES.
        GO TO(560,660,560,560,880,560,820,560,560,880,560),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)  HPART(I)                               |REPORT PART MISSING.

        IDATA(I)=1                                             |FLGPART TYPE FOR INPDIA.

        GO TO 880                                              |NEXT.
C
580     GO TO(600,640,880,880,880,780,880,800,880,880,880),I   |ON PART LOADED SPECIAL HANDLING BRANCH, IF ANY.
C
C     CHECK THAT NUMB OF ENGINES MATCHES GEAR REQUIREMENTS
C
600     NUME   = 1                                             |ASSUME ONE ENG.

        NUMBG  = NPARTS(4)                                     |GET # OF GEARS.

        IF ( NUMBG.LT.1 )  GO TO 880
        DO 620  J = 1,NUMBG
```

```
        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,NUMBG                                       |NO LOOP THRU 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='COCY2 '                                         |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 PRNOUT                                            |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='CODY2 '                                         |SET DEFAULT DRIVE CONV NAME.

        CALL DSK                                               |GET IT.

        IF(IPRNT.EQ.-10) GO TO 740                             |(ERR?)YES.
        CALL PRNOUT                                            |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 880                                              |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(NOSEG.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(NDRTE.EQ.0.OR.LDYNA) GO TO 880          !(1ST SECT OF RTE LOADED?) YES, CONT
      IPRM1=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 880                   !(GOT IT?) YES, CONT
      WRITE(5,5240) HPART(8),RNAME               !NO, GO REPORT ERR
      GO TO 920                                  !GO ERR BYE
  820 IF(LDYNA) GO TO 880
      GO TO 560
C
  880 CONTINUE
C
      IF ( IERROR.GT.0 )  GO TO 160              !(ALL PARTS LOADED?) NO.
C
      CALL DSKCTR(0,'INPBAT')                    ! YES, RELEASE ALL DB FILES
C
      LSIMUL=.TRUE.
      GO TO 4640                                 ! DO A VEHSIM STATUS REPORT TO LPT FILE
  900 LSIMUL=.FALSE.
C
      WRITE(5,5300)                              ! "(SIMULATING)"
C
      CALL SIMCTR ( IECOND , SIMODE )            ! GO SIMULATE
C
      IF(IECOND.EQ.0) IECOND=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.
```

```
          IECOND=3                                                    | SET INPDIA ERR FLG
          GO TO 100                                                   | BYE.
C
C*************************************************************
C
C      /ZERO/ COMMAND - RESET ALL PROGRAM VARIABLES TO INITIAL STATE
C
          ENTRY ZERCMD(IECOND)                                        |DIALOG ENTRY.
          IECOND=1                                                    |FLG DIALOG ENTRY.
          GO TO 980
C
  900     READ  (4,5540)  UN,UT                                       |READ BATCH COMMAND CARD.
          IECOND=0                                                    |FLG BATCH ENTRY.
  980     IF(UT.EQ.HPART(5)) GO TO 1040                               |(ZERO ACCESSORY?)YES.
          CALL ZERO                                                   |NO, ZERO ALL SUB 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.
          LDIES=.FALSE.
          ENG1=.FALSE.                                                |ENG 1 NOT LOADED.
          ENG2=.FALSE.                                                |ENG 2 NOT LOADED.
          CNVNAM(1)=HNLOAD                                            |DRIVE CONV NAME.
          CNVNAM(2)=HNLOAD                                            |COAST CONV NAME.
          RNAMEO=HNLOAD                                               |SAVE LOC FOR "$DYNO" OF ROUTE NAME.
          NPAR18=0                                                    |SAVE LOC FOR NPARTS(8) WHEN "$DYANO".
          SDEBUG=HOFF                                                 |DEBUG DEFAULT COMMAND.
          SLIMIT=HSUMMA                                               |LIMIT PRINT DEFAULT COMMAND.
          DO 1000 I=1,NMOD                                                    |LOOP THRU ALL MODIFY FLGS.
 1000     MFLG(I)=0                                                   |FLG NO MODS.
 1020     IF(IECOND.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.DSTAR) 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.

        OLDUTY(I)=OLDUTY(NACC)                  !(717)
        DUTCYC(I)=DUTCYC(NACC)                  !(717)
        APROT(I)=APROT(NACC)                    !MOVE PROTECTION.

        APPN(I)=APPN(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(IB)=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(IECOND)                    !DIALOG ENTRY.
        IECOND=2                                !ASSUME ERROR.

        GO TO 1220
C
1200    READ  (4,5560)  UN,UT,UP                !READ COMMAND CARD FROM CTRFIL.

1220    CALL SLOOKP(UT,NUMPAR,MPART,IPRNT,$1240)        !LOOK UP PART TYPE.
        IECOND=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 DSKDEL                                        !GO DELETE IT.

        IF ( IPRNT .EQ. - 10 )  GO TO  280                 !(ERR?)YES, FLG IT.
        IECOND=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(IECOND)                               !DIALOG ENTRY.
        IECOND=2                                           !SET ERROR UNKNOWN UNITS.

        GO TO 1280

C
 1260   READ  (4,5280)  WORD,UNITS                         !READ COMMAND CARD FROM CTR FIL.

        IF ( WORD.NE.HENGIN   )  GO TO 300                 !(UNITS FOR ENG?)NO,ERR.
 1280   IF (( UNITS(1).NE.HRPM .AND.UNITS(1).NE.HPISTO )  .OR.   !(ENG SPD UNITS VALID?)NO.
        1   ( UNITS(2).NE.HBMEP.AND.UNITS(2).NE.HTORQU.AND.
        2     UNITS(2).NE.HHP    )     .OR.                !(ENG LOAD UNITS VALID?)NO.
        3   ( UNITS(3).NE.HLBHR.AND.UNITS(3).NE.HBSFC.AND.
        4     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.
        IECOND=1                                           !SET NO ERROR FLG.

        GO TO 160
C
C*****************************************************************
C
C     /DUMP/ COMMAND - PRINT ALL PARTS DATA STORED ON PARTS DATA FILE
C                      AND/OR A DIRECTORY OF ALL THESE PARTS
C
        ENTRY DMPCMD(IFCOND)                               !DIALOG ENTRY.
        IPRNT=IECOND                                       !GET PART TYPE PTR.
```

```
        IECOND=4                                                    !SET ERR FLG.

        GO TO 1320

C
1300    READ(4,5540) DN,DT
        IF(DT.NE.HDIREC .AND. DT.NE.HPARTS) GO TO 1320              ! (ALL PART TYPES?) NO.
        IPRNT=-1                                                    ! ASSUME DIR ONLY.

        IF(DT.EQ.HPARTS) IPRNT=0                                    ! (DIR ONLY?) YES, SET FLG.

        DN=DSTAR                                                    !SET PART NAME WILD.

        DT=HSTAR                                                    !SET PART TYPE WILD.

C
1320    LSAVE=LIMPRN                                                !SAVE.
        LIMPRN=.FALSE.                                              ! PRINT ON.
C
        CALL DSKDIR                                                 ! EXECUTE DUMP/DIR REQUEST.

C
        LIMPRN=LSAVE                                                ! RESTORE.

        IECOND=1                                                    ! ASSUME NO ERR.

        IF(IPRNT.EQ.-10) IECOND=3                                   ! (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(IECOND)                                        !DIALOG ENTRY.
        ALIMN=DATA(1)                                               !GET PRINT OUT INC VAL.

        IECOND=2                                                    !ASSUME ERROR.

        GO TO 1360

C
1340    READ  (4,5340)  WORD,ALIMN                                  !READ COMMAND CARD FROM CTR FILE.

1360    CALL SLOOKP(WORD,5,HLIMT,NARG,&1370)                        !(VALID LIMIT PRINT?)YES, .
        GO TO 300
1370            LIMPRN = .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 )  LIMPRN = .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    IECOND=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(IECOND)                                      !DIALOG ENTRY.
      IECOND=1
      VALUE=DATA(1)
      PNAME=HINPBA                                              !SET ROUTINE NAME FOR ERR MESS FROM LOOKUP.
      GO TO 1420
C
1400  READ  (4,5350)  WORD,VALUE,UN
      PNAME=DBLANK                                              !FLG LOOKUP NO ERR MESS.
1420  CALL SLOOKP(WORD,NMOD,HMOD,IMOD,&1440)
      IECOND=2
      GO TO 300
1440  GO TO(1780,1500,1520,1540,1600,1620,1480,1800,1460,1560
     2,1640,1820,1580,1840,1860,1860,1860,1660,1680,1720
     3,1740,2080,2082,2090),IMOD                                !GO TO SAVE OLD VALUE, SET NEW VALUE, AND CHECK IF LEGAL.
C
1460  OLDVAL(9)=ISMODE                                          ! MODIFY SHIFT NOT IN DOCUMENTATION- INTENDED FOR SUPPORT DE
BUGGING
      ISMODE = VALUE + .001
      IF(ISMODE.LT.1 .OR. ISMODE.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)=DEFDT
      DEFDT  = VALUE
      GO TO 1940
1580  OLDVAL(13)=FSPGR
      FSPGR  = VALUE
      GO TO 2000
1600  OLDVAL(5)=RAR(1)
      RAR(1) = VALUE
      GO TO 1980
1620  OLDVAL(6)=WLSG
      WLSG=VALUE
      GO TO 1940
1640  OLDVAL(11)=LTRFZ
      LTRFZ=.FALSE.
      IF(VALUE.GT..0) LTRFZ=.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.

        WRITE(5,1702)
1702    FORMAT(' $SHIFT LOGIC temporarily unmodifiable due to mods'
        2' in multi speed axle SHIFT LOGICS')
        GO TO 1940
        CALL MODSL ( MODE,VALUE )                            |YES, MODIFY IT.
        GO TO 1940
1720    OLDVAL(20)=PHI
        PHI=VALUE
        PHIO=VALUE
        GO TO 1940
1740    OLDVAL(21)=LDYNA
        IF(VALUE.EQ.0.) GO TO 1760
        LDYNA=.TRUE.
        RNAMEO=RNAME
        NPART8=NPARTS(8)
        PHIO=PHI
        RNAME=MPDYNO
        NPARTS(8)=1
        LDYNOV=.FALSE.|[612]
        IF(VALUE.GT.0)GO TO 1940|[612]ONLY OVERIDE IF NEG VALUE.
        DYN=-VALUE|[612]ASSIGN VALUE
        LDYNOV=.TRUE.|[612]SET FLAG
        GO TO 1940
1760    LDYNA=.FALSE.
        LDYNOV=.FALSE.                                       |[612]
        RNAME=RNAMEO
        NPARTS(8)=NPART8
        PHI=PHIO
        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.

      DISP   = 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/IOCYL)*ICYL
C
 1920 CALL SCALEN                                    !SCALE ENGINE.

C
 1940 MFLG(IMOD)=1                                   ! SET MODIFIED VALUE FLG.

      VALNEW(IMOD)=VALUE                             ! SAVE ,NEW VALUE.

      GO TO 160                                      ! *DONE.

C
 1960 IF(NPARTS(9).LT.1.AND.LVNEW) GO TO 2020        ! (TIRE LOADED?) NO, ERR.
      GO TO 1940                                     ! YES, ALLOW MODIFICATION.

C
 1980 IF ( NPARTS(3).LT.1 )  GO TO 2020              ! (VEHICLE LOADED?) NO, ERR.
      GO TO 1940                                     ! YES, ALLOW MODIFICATION.

C
 2000 IF ( NPARTS(1).LT.1 )  GO TO 2020              ! (ENGINE LOADED?) NO, ERR.
      GO TO 1940                                     ! YES, ALLOW MODIFIFCATION.

C
 2020 IECOND=3                                       ! ? PART TO MODIFY NOT LOADED.

      GO TO 2060

C
 2040 IECOND=4                                       ! ? ILLEGAL MODIFY VALUE.

 2060 WORD=HMOD(IMOD)                                !SET WRD TO UNABREVIATED FORM.

      GO TO 300
C
 2080 IF(NACC.LT.1)GO TO 2020                        !(717) IF NO ACCES, ERROR

      IF(DIALOG)WRITE(5,7400)                        !(717)(DIALOG?)SOLICIT ACC NAME.

      IF(DIALOG)READ(5,6340)UN                       !(717)(DIALOG?)GET NAME.

      DO 20800 I=1,NACC                              !(717)LOOP THRU ALL ACCES LOADED.

      IF(UN.EQ.ANAME(I))GO TO 20820
20800 CONTINUE                                       !(717)
      WRITE(5,5480)UN                                !(717)!REPORT NOT FOUND.

      WRITE(6,5480)UN                                !(717)
      GO TO 2020                                     !(717)GO TO ERROR HANDLER.
```

```
C
20820   OLDDTY(I)=DUTCYC(I)                                    !(717)SAVE OLD VALUE.

        DUTCYC(I)=VALUE                                        !(717)GET NEW VALUE.

        GO TO 1940
C
2082    OLDVAL(23)=FRC4
        FRC4=VALUE
        GO TO 1960
C
2090    OLDVAL(24)=LDIES
        LDIES=.FALSE.
        IF(VALUE.GT.0.0)LDIES=.TRUE.
        GOTO 1940
C
C*****************************************************************
C
C     /LOCKUP/ , /UNLOCK/ COMMANDS - LOCK OR UNLOCK THE TRANSMISSION
C                                    GEARS
C
        ENTRY DALCMD(IECOND)                                   !DIALOG ENTRY.
        IECOND=-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.
        IECOND=0                                               !FLG NO ERR.

C
        GO TO 160                                              !DONE.
C
2200    IECOND=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,8140)                                !VALID NAME?
        READ  (4,5000)  VCOM                                   !GET VEHICLE COMMENT.
```

```
      READ(1,5430) 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.

2340  READ(4,5410) WORD,WGT,CD,CDC,AREA,DUM,AIP,DUM,DUM,WLSG,    !READ DATA.
     1 BVG,BVGEFF
      IF(BVG < 0.2)     BVG    = 1.0
      IF(BVG .EQ. 1.0) BVGEFF = 1.0
2360  IPRNT=3                                                    !FLG PART TYPE.

      GO TO 3720                                                 !GO STORE.

C
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,8140)
C
      READ(4,5000)AXCOM
      NAX=1
      READ(4,5400)WORD,(RAR(IT),(ERAR(JT,IT),JT=1,2),IT=1,3)
      NAXS=NRPTS(RAR,3)
      IF(ERAR(2,1).GT.1.E-7)NRAX=2
C
      CALL NXTCRD(HAXLE,IFLAG,WORD)
      GO TO (2420,2440,2480,2480),IFLAG
2420  IF(WORD.NE.HSINGL)GO TO 2500
2440  READ(4,5420)DUM,DNA,DNAX
      NA=DNA+.001
      NAX=DNAX+.001
      IF(NA.LT.0 .OR. NA.GT.2)GUTO 2500
      IF(NAX.LT.0 .OR. NAX.GT.3)GOTO 2500
      CALL READPD(2,20,10,HAXLE,NPAX(NA,NAX),IFLAG,AXTORQ(1,NA,NAX)
     2,AXRPM(1,NA,NAX),DUMMY,DUMMY)
C
      GO TO (2500,2460,2480,2480),IFLAG
C
2460  GO TO 2440
C
2480  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,62640)
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),82580)
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,8140)
C
      READ(4,5000)TRCOM
C
      DO 2680 I=1,NGTR
      GEANUM(I)=I
      READ(4,6460)GEANAM(I)
      CALL VALID2(GEANAM(I),8140)
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
C
2700  NGEAR=1                                                    !SET GEAR STORAGE PTR.

      READ  (4,5380)  PNAME                                      !GEAR NAME.
      CALL VALID2(PNAME,8140)                                    !VALID NAME?
      GNAME(NGEAR)=PNAME
      READ  (4,5000)  GCOM                                       !COMMENT.
      READ(4,5420) WORD,AIGIN(NGEAR),AIGOUT(NGEAR),GRAT(NGEAR)   !READ GEAR DATA.
      1,ERAT(NGEAR)
      IF ( WORD.NE.HDATA  )  GO TO 300                                  !(ERR?) YES.

      NGRLSS(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 READPD(2,20,10,HBLANK,NGRLSS(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=I                                                        !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,AIAS(NACC),ACCSR(NACC),DUTCYC(NACC)      !READ INTERTIA DATA.
      IF ( WORD.NE.HDATA )  GO TO 300                               !(ERR?) YES.
      CALL READPD(2,20,10,HBLANK,NNA(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.HNUMB  )  GO TO 300
      NUMG  = DNUMG + .001
      IF ( NUMG.LT.1 .OR. NUMG.GT.20 )  GO TO 300
      NUMBSL =0
C
2790  READ(4,5060,END=2840)COL1
      IF(COL1.NE.HSTAR)GO TO 2792
      BACKSPACE 4
      GO TO 2840
2792  IF(COL1.EQ.HCOM)GO TO 2790
      NUMBSL=NUMBSL+1
      I=NUMBSL
      REREAD  5420,  WORD,DGF,DGT,SHFTIM(I),DAF,DAT
      IF ( WORD.NE.HSHIFT )  GO TO 300
      IGF(I) = DGF + .001
      IGT(I) = DGT + .001
      IF(DAF.LT..5)DAF=1.
      IF(DAT.LT..5)DAT=1.
      IAF(I) = DAF + .001
      IAT(I) = DAT + .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.HVACOU .AND. WORD.NE.HTHROT .AND.
     1     WORD.NE.HOTHER ) GO TO 300
      SAVE1  = WORD
      LVAC = .FALSE.
```

```
      LTHR = .FALSE.
      IF ( WORD.EQ.HVACUU )  LVAC  = .TRUE.
      IF ( WORD.EQ.HDTHRO )  LTHR  = .TRUE.
      READ  (4,5000)  WORD
      IF ( WORD.NE.HOUTPU .AND. WORD.NE.HENGIN .AND. WORD.NE.HVEHIC )
     1   GO TO 300
      SAVE2  = WORD
      IF ( WORD.EQ.HOUTPU )  LENG  = .FALSE.
      IF ( WORD.EQ.HENGIN )  LENG  = .TRUE.
      IF ( WORD.EQ.HVEHIC )  PARAB = .TRUE.
      IF ( WORD.NE.HVEHIC )  PARAB = .FALSE.
      BACKSPACE 4
      BACKSPACE 4
C.....................................
2800  CALL READPD ( 2,10,10,NSHIFT,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=2840)  WORD
      BACKSPACE 4
      IF ( WORD.NE.HDETEN )  GO TO 2790
      LDETNT = .TRUE.
      READ  (4,5340)  WORD1,DETPT(I),WORD2,DETRPM(I)
      IF ( WORD1.NE.HVACUU .AND. WORD1.NE.HTHROT )  GO TO 300
      IF ( WORD2.NE.HOUTPU .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.HOUTPU )  LDETE = .FALSE.
      IF ( WORD2.EQ.HENGIN )  LDETE = .TRUE.
2820  GO TO 2790
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,8140)                                   !VALID NAME?
      READ  (4,5000)  DCOM                                      !GET DRS COMMENT.

      READ  (4,5420)  WORD,T0,D0,V0,A0,G0,XA0                   !READ INITIAL CONDITIONS.
      IF ( WORD.NE.HLIMITI )  GO TO 300                                !(ERROR?)YES, BYE.

      NG0    = G0 + .001                                        !FIX INITIAL GEAR.

      IF ( NG0.EQ.0 )  NG0 = 1                                  !(ZERO?)SET TO 1ST GEAR.
```

```
      NAO    = XAO + .001                                  !FIX INITIAL REAR AXLE.

      IF ( NAO.EQ.0 )  NAO = 1                             !(ZERO?)SET TO 1ST AXLE.

      NDSEG=0                                              !INITALIZE DRS SEG OFFET.

      MLTSEC=.FALSE.                                       !ASSUME NOT MULTI SECT.

C
2880  DO 2900  NSEG = 1,50                                 !READ UP TO 50 DRS SEGS.

      READ  (4,5420)  WORD,ASEG(NSEG),VSEG(NSEG),PWOT(NSEG),   !READ A SEGMENT.
     1       ATHOLD(NSEG),GSEG,THRATE(NSEG),TSEG(NSEG),
     2       DSEG(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)=NWRCNT(CARD,4)                          !DETERMINE TYPE OF SEGMENT
C
C
      IF ( CBL.EQ.HBLANK )  PCSEG (NSEG) = -1.             !DETERMINE SEGMENT END CONDITIONS
      IF ( DBL.EQ.HBLANK )  DSEG  (NSEG) = -1.
      IF ( EBL.EQ.HBLANK )  VELSEG(NSEG) = -1.
      IF ( OBL.EQ.HBLANK )  POSTSE(NSEG) = -1.
      IF ( TBL.EQ.HBLANK )  TSEG  (NSEG) = -1.
C
      CALL NXTCRD(HSEGME,IFLAG,WORD)                       !LOOK NEXT CARD.
      GO TO(300,2900,2940,2940),IFLAG                      !(ERR/MORE DATA/COMMAND/EOF)?
C
C
2900  CONTINUE                                             !NEXT CARD.
C
      NSEG=NSEG-1                                          !CALC # OF SEG'S WHEN NORMAL LOOP TERMINATION, MUST BE MULTI
 SECT.
      IF(MLTSEC) GO TO 2920                                !(1ST TIME?)NO.
      NPARTS(6)=0                                          !YES, FLG DRS NOT LOADED.

      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.
      CALL DSK                                             !GO STORE ON DB.

      NDSEG=NDSEG+NSEG                                     !CALC DRS SEG OFFSET.

      IF(.NOT.LSAVE) GO TO 2880                            !(LAST SECT?)NO.
      IPRNT=106                                            !YES, FLG DSK /USE/ DRS.

      GO TO 4120                                           !RELOAD 1ST SECT.

C
```

```
 2940    LSTSEC=.TRUE.                                          !FLG LAST SECT OF DRS.
         IF(MLTSEC) GO TO 2920                                  !(MULTI SECT?)YES.
         IPRNT=6                                                !NO, FLG DSK TO STORE DRS.
         GO TO 3720                                             !GO STORE.
C
C*****************************************************************
C
C      /ROUTE/ COMMAND - LOAD ROUTE DATA AND STORE ON PARTS DATA
C                        FILE AND PRINT LISTING OF DATA
C
 2960    READ  (4,5380)  RNAME                                  ! ROUTE NAME
         PNAME=RNAME
         CALL VALID2(PNAME,8140)                                !VALID NAME?
         READ  (4,5000)  RCOM                                   ! COMMENT
C
         MLTSEC=.FALSE.                                         !ASSUME NOT MULTI SECT RTE.
         DRVWND=1000.0                                          !SET DRS WIND SPEED TO FLG RUN TIME DEFAULT.
         READ(4,5420) WORD,VALUE                                !READ NXT CARD.
         IF(WORD.NE.HDATA) GO TO 2980                           !(DRS WIND SPEED DATA CARD?)NO.
         DRVWND=VALUE                                           !YES, RESET DRS WIND SPEED DEFAULT VALUE.
         GO TO 3000
 2980    BACKSPACE 4                                            !POSI CTRFIL.
         IF(WORD.NE.HMILEP) GO TO 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 N
XT/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=8                                                !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,8140)                         |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,COIAN,CONTOR,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 3201
3180    CALL READPD ( 4,20,10,HBLANK,NTORP,IFLAG,TIN,TOUT,SPIN,SOUT )
3200    NTD    = NTORP
        NTC    = NTORP
        NTEP   = 0
C
        DO 3280  I = 1,NTORP
```

```
        IF ( CONTOR.GT..001 )  TIN(I) = CONTOR
        IF(LFULL) GO TO 3220                                    !(FULL CONV?)YES.

        TOUT(I)  = TOUT(I) * TIN(I)
        SOUT(I)  = SOUT(I) * SPIN(I)
 3220   IF ( COAST )  GO TO 3260
C
C     CALCULATE CAPACITY FACTOR,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 BREAKPOINT FOR DRIVE CONVERTER
C
        IF ( NTRP.GT.0  )  GO TO 3280
        IF ( 1.EQ.NTORP )  GO TO 3240
        IF ( ABS ( TRD(I) - 1. ) .GT. .0001 )  GO TO 3280
 3240   NTRP   = I
        TORDPK = AKD(I)
        GO TO 3280
C
C     CALCULATE CAPACITY FACTOR,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,8140)                                 !VALID NAME?
        IF( .NOT.((UNITS(1).EQ.HRPM    .OR.  UNITS(1).EQ.HPISTO )
        1     .AND. (UNITS(2).EQ.HBMEP  .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)  ECON
        READ  (4,5460)  WORD,BORE,STROKE,CYL,ENMIN(1),RPMAX(1),
        1                EINER,FSPGR,CYCLE
        IF ( WORD.NE.HDATA  )  GO TO 300
        ICYL   = CYL + .001
        DISP   = 3.1415927 * ((BORE/2.)**2.) * STROKE * ICYL
        IF ( FSPGR.LT..0001 )  FSPGR = .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
CC    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
CC    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
CC    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
CC    FILL LEFT HALF OF MAP WITH FIRST DATA POINT
C
      N      = 20 - NPOINT
      DO 3480  J = 1,N
      DO 3480  K = 1,4
```

```
3480    EMAP(NRPMP,J,K) = DATA(20*(K-1)+1)
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(1,1)
        ENMIN(1)  = CONST * ENMIN(1)
        RPMAX(1)  = CONST * RPMAX(1)
        SPIDLE(1) = CONST * SPIDLE(1)
C
C     CONVERT BMEP,HP TO LB-FT
C
3560    IF ( LTOR  )  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      = ENMIN(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.
        IPRNT     = 1                                          |SET DSK FLG TO STORE ENG.
```

```
        NPARTS(1) = 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(IECOND)
        IECOND=2
C
3760    LRPM   = .TRUE.
        LTOR   = .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,8140)                                !VALID NAME?
        ENAME(1)=PNAME
        IF ( UNITS(1).NE.HRPM  .AND.UNITS(1).NE.HPISTO )  GO TO 300
        IECOND=3
        IF ( UNITS(2).NE.HBMEP .AND.UNITS(2).NE.HTORQU.AND.
       1 UNITS(2).NE.HHP     )     GO TO 300
        IECOND=4
        IF ( UNITS(3).NE.HLBHR .AND.UNITS(3).NE.HBSFC .AND.
       1 UNITS(3).NE.HGALHR )     GO TO 300
        IECOND=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.HPISTO )  GO TO 3780
      LRPM   = .TRUE.
      LPS    = .FALSE.
3780  IF ( UNITS(2).NE.HBMEP .AND. UNITS(2).NE.HHP    )  GO TO 3820
      LTOR   = .FALSE.
      IF ( UNITS(2).EQ.HBMEP  )  GO TO 3800
      LHP    = .TRUE.
      GO TO 3820
3800  LBMEP  = .TRUE.
3820  IF ( UNITS(3).NE.HBSFC .AND. UNITS(3).NE.HGALHR )  GO TO 3860
      LLBHR  = .FALSE.
      IF ( UNITS(3).EQ.HBSFC  )  GO TO 3840
      LGALHR = .TRUE.
      GO TO 3860
3840  LBSFC  = .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              ERPMO,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  NTORQ(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 VAL102(PNAME,8140)                               !VALID NAME?
      READ(4,5000) TCOM                                     !GET TIRE COMMENT.

      READ(4,5400) WORD,WRAD,FRC1,FRC2,TIREFF,AIW,DUM,FRC4  !READ TIRE DATA.
      IF(WORD.NE.HDATA) GO TO 330                           !(ERROR?)YES, BYE.
      IPRNT=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(IECOND)                                            |DIALOG ENTRY.
        LPDP=.FALSE.                                                    | ASSUME NOT PDP DUMP

        IF(IECOND.GT.0)GO TO 40000                                      | IF NEG, PDP DUMP.
        LPDP=.TRUE.
        IECOND=ABS(IECOND)                                              |RESET FLAG
40000   IP=IECOND                                                       |GET PART TYPE PTR.

        IECOND=3                                                        |ASSUME ERROR.

        GO TO 4020
C
4000    READ  (4,5580)  PNAME,UT,I,IDATA                                        |GET USE COMMAND.

        CALL VALID2(PNAME,8140)                                         |VALID NAME?
        CALL SLOOKP(UT,NUMPAR,HPART,IP,84020)                           |LOOK UP PART TYPE AND SET PTR,(GOT IT?)YES.

        GO TO 4600                                                      |?TYPE?
4020    IPRNT=100+IP                                                    |SET D8K FLG TO /*USE/

        GO TO (4140,4080,4460,4320,4040,4100,4420,4380,4500,4280,4440),IP       |GO SET UP & DO ANY SPECIAL HANDLING.
C
C
C*****************************************************************
C     LOAD ACCESSORY DATA FROM PARTS DATA FILE
C
4040    NACC      = NACC + 1                                            |INC # OF ACC LOADED.

        ANAME(NACC) = PNAME                                             |SET ACC NAME.
        CALL D8K                                                        |LOAD ACC.

        IF ( IPRNT.EQ.-10 )  GO TO 4060                                 |(ERROR?)YES.
        IF(LPDP)GO TO 4070
        NPARTS(5) = NACC                                                |FLG ALL LOADED.

        OLDUTY(NACC)=0.                                                 |[717] ZERO MODIF FOR THIS ACCESS.

        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)(ACOM(IT),IT=1,NWRCNT(ACOM,16))                    |[617]
        WRITE(6,7020)AIAS(NACC)                                         |[617]
        IS=1
        IP=NNA(NACC)                                                    |[617]
        IF(IP.LE.10)GO TO 4072
        IP=10
4074    WRITE(6,7040)(ACCT(IT,NACC),IT=IS,IP)                           |[617]
        WRITE(6,7060)(ACCS(IT,NACC),IT=IS,IP)                           |[617]
        IF(NNA(NACC).LE.10 .OR. IP.GT.10)GO TO 4074
```

```
      IS=11
      IP=ANA(NACC)
      GO TO 4072
 4074 NACC=NACC-1                                                    |(617)
      GO TO 4560                                                     |(617)
C
C=====================================================================
C     LOAD CONVERTER DATA FROM PARTS DATA FILE
C
 4080 CNAME     = PNAME                                              |GET CONV NAME.
C.....................................
      CALL DSK                                                       |LOAD CONV.

      IF ( IPRNT.EQ.-10 )  GO TO 4580                                |(ERROR?)YES, BYE.
      IF(LPDP)GO TO 4560
      NPARTS(2) = NPARTS(2) + 1                                      |FLG CONV AND INC CNT.

      IF(.NOT. COAST) CNVNAM(1)=CNAME                                |)DRIVE CONV?)YES, SAVE NAME.

      IF(COAST) CNVNAM(2)=CNAME                                      |(COAST CONV?)YES, SAVE NAME.

      GO TO 4560                                                     |DONE.
C
C=====================================================================
C     LOAD DRIVING SCHEDULE DATA FROM PARTS DATA FILE
C
 4100 DNAME     = PNAME                                              |GET DRS NAME.
 4120 CALL DSK                                                       |LOAD DRS.

      IF ( IPRNT.EQ.-10 )  GO TO 4580                                |(EROR?)YES, BYE.
      IF(LPDP)GO TO 4560
      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
      IENG=1
      GO TO 4180
 4142 IF(DIALOG) GO TO 4160                                          |(DIALOG MODE?)YES.
      IENG=1
      IF ( IENG.GT.2 )  GO TO 4600
      IF ( IENG.EQ.0 )  IENG = 1
      GO TO 4180
 4160 IENG=LOCKG(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
      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 17)YES, DONE.
      NPARTS(1)=2                                                |FLG 2 ENG'S LOADED.

      GO TO 4560                                                 |DONE.
C
 4270 IF(LRPM)UNITS(1)=HRPM                                      |[617]
      IF(LPS)UNITS(1)=HPS                                        |[617]
      IF(LTOR)UNITS(2)=HTORQU                                    |[617]
      IF(LBMEP)UNITS(2)=HBMEP                                    |[617]
      IF(LHP)UNITS(2)=HHP                                        |[617]
      IF(LLBHR)UNITS(3)=HLBHR                                    |[617]
      IF(LBSFC)UNITS(3)=HBSFC                                    |[617]
      IF(LGALHR)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,ENMIN(IENG),RPMAX(IENG),EINER
     2       ,FSPGR,XCYC                                         |[617]
      IA=(IENG-1)*4+1                                            |[617]
      DO 4274 IR=1,NRPM(IENG)                                    |[617]
      WRITE(6,7120)ERPM(IENG,IR)
      NPS=NTOR(IENG,IR)
      NP=20
      IS=21-NPS
      IF(NPS.LE.10)GO TO 4272
      NP=IS+9
 4272 WRITE(6,7140)(EMAP(IR,IP,IA),IP=IS,NP)                     |[617]
      WRITE(6,7160)(EMAP(IR,IP,IA+1),IP=IS,NP)                   |[617]
      WRITE(6,7180)(EMAP(IR,IP,IA+2),IP=IS,NP)                   |[617]
      WRITE(6,7200)(EMAP(IR,IP,IA+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 4360 I=1,NGTR
      NGEAR=GEANUM(I)
      PNAME=GEANAM(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 4580                     !(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                     !(EROR?)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
        NUMBSL=(NUMG-1)*2
        DO 44260 IG=1,NUMBSL
        NP=NSPTS(IG)
        WRITE(6,7260)IGF(IG),IGT(IG),SHFTIM(IG),IAF(I),IAT(I)
        IF(LVAC)GO TO 44230
```

```
          WRITE(6,7280)(SHFTPT(IP,IG),IP=1,NP)
          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(PARAM)WRITE(6,7340)(SHFTRP(IP,IG),IP=1,NP)
          IF(.NOT.LENG .AND. .NOT.PARAM)
          2       WRITE(6,7360)(SHFTRP(IP,IG),IP=1,NP)
          IF(.NOT.LDETNT)GO TO 44260
          WORD1=HTHROT
          IF(LDETV)WORD1=HVACUU
          WORD2=HOUTPU
          IF(LENG)WORD2=HENGIN
          IF(PARAM)WORD2=HVEHIC
          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     LVEHAX=.FALSE.
          CALL D&K
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.

          CALL D&K                                             !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.

          GO TO 4540                                           !DONE.
C
C=======================================================================
C     LOAD TIRE DATA FROM PARTS DATA FILE
C
 4500     TNAME=PNAME                                          !GET TIRE NAME.

          CALL D&K                                             !LOAD TIRE.

          IF(IPRNT.EQ.-10) GO TO 4580                          !(ERROR?)YES, DONE.
          IF(LPDP)GO TO 4560
 4520     MFLG(1)=0                                            !FLG TIRE NOT MODIFIED.

          MFLG(2)=0
          MFLG(3)=0                                            !DONE.
          MFLG(23)=0                                           !DONE.
C
C=======================================================================
 4540     NPARTS(IPRNT)=1                                      !FLG PART LOADED.

C
```

```
 4560    IECOND=1                                                !FLG NO ERROR.
         GO TO 160                                               !DONE.
C
C     ERROR RETURN WHEN PART REQUESTED NOT ON PARTS DATA FILE
C
 4580    IECOND=2                                                !? PART NOT FOUND.
         GO TO 280
C
 4600    IECOND=3                                                !? UNKNOWN PART TYPE.
         GO TO 300
C
C*************************************************************************
C
C     /STATUS/ COMMAND - REPORT CURRENT VEHSIM STATUS
C
         ENTRY STSCMD(IECOND)                                    !DIALOG ENTRY.
         JCT=IECOND                                              !GET UNIT NUMBER TO SEND STATUS REPORT TO.
         IECOND=1                                                !ASSUME ERROR.
         GO TO 4660
C
 4620    SKIP RECORD 4                                           !LOOK NEXT COMMAND.
         READ(4,5060) DUMMY,WORD
         BACKSPACE 4
         IF(WORD.EQ.HCOMND(2)) GO TO 160                         !IS IT /*SIMULATE/?)YES, SKIP STATUS COMMAND SIMULATE WILL D
O IT.
 4640    JCT=6                                                   !SEND STATUS REPORT TO LPT.
C
 4660    WRITE(JCT,5600) VERID                                   !VERSION ID.
         IF(BATCH) WRITE(JCT,5620)                               !MODE.
         IF(DIALOG) 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 TO 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)
         IF(ENG2) WRITE(JCT,5800) ENAME(2)                       !(ENG # 2 LOADED?)YES.
         IF(ENG2.AND.LDIES)WRITE(JCT,5750)
         IF( .NOT. (ENG1 .OR. ENG2) ) WRITE(JCT,5820)            !(NO ENGS LOADED?)YES, REPORT IT.
         WRITE(JCT,5840) ((CNVTYP(I),CNVNAM(I)),I=1,2)           !REPORT CONVERTERS.
         WRITE(JCT,5860) VNAME                                   !REPORT VEHICLE.
         WRITE(JCT,5960)AXNAME                                   !REPORT AXLE NAME
C
         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 ASSIGNME
NTS.
        DO 4800 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
 4780   WRITE(JCT,5940) 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.
        WRITE(JCT,6040) SNAME                             |SHIFT LOGIC.
        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=NOFF                               |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,NMOD                        |LOOP THRU ALL MODS EXCEPT DYNA.
      IF(I.EQ.21 .OR. I.EQ.22 .OR. I.EQ.24)GO TO 4880   |SKIP DYNA OR DUTYCYCLE OR DIESEL
      IF(MFLG(I).EQ.0) GO TO 4880             |(MODIFIED VALUE?)NO.
      IMCNT=IMCNT+1                           |YES, INC CNT OF MODIFIED VALUES.
      WRITE(JCT,6260) HMOD(I),OLDVAL(I),VALNEW(I)   |REPORT MODIFICATION.
4880  CONTINUE                                |NEXT.
      IF(NACC.LE.0)GO TO 4896                 |(717)
      DO 4890 I=1,NACC
      IF(OLDUTY(I).EQ.0.)GO TO 4890
      WRITE(JCT,7420)ANAME(I),OLDUTY(I),DUTCYC(I)   |(717)
      IMCNT=IMCNT+1
4890  CONTINUE
4896  IF(IMCNT.EQ.0) WRITE(JCT,6280)          |(ANY MODIFICATIONS?)NO, REPORT IT.
C
      IF(LSIMUL) GO TO 900                    |/*SIMULATE/ CALL TO /*STATUS/?)YES, GO SIMULATE.
      IECOND=0                                |NO, FLG NO ERR.
      GO TO 160                               |DONE.
C
C*****************************************************************
C
C     SET PROPER *ENDL* 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  ENDD=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(1X,A1,16A5)
 5040    FORMAT(1X,16A5)
 5060    FORMAT (A1,16A5)
 5080    FORMAT(/' ? INPBAT-COMMAND OR DATA CARD IN ERROR ---'16A5/)
 5100    FORMAT(/' ? INPBAT-DUE TO HEVSIM CONTROL FILE ERRORS DETECTED'
         1/10X'DURING EXECUTION OF "'A5'" COMMAND SWITCHING TO'
         1,' DIALOGUE MODE.'
         2/10X'GIVE "CONTINUE" COMMAND TO RESUME PROCESSING OF VEHSIM'
         3,' CONTROL FILE.'/)
 5120    FORMAT(/' /*STOP/ COMMAND ENCOUNTERED IN HEVSIM CONTROL FILE.')
 5140    FORMAT (18XA5,7X2F10.1)
 5160    FORMAT (12X12A5,A8)
 5180    FORMAT (12A5,A8)
 5200    FORMAT ('1? INPBAT-/*SIMULATE/ COMMAND IGNORED  - '
         1,' THE FOLLOWING PARTS ARE UNDEFINED'//)
 5220    FORMAT (/52XA5)
 5240    FORMAT(/' ? INPBAT- FIRST SECTION OF 'A5,3XA10' WAS'
         1,' NOT LOADED'/10X'AND RELOAD ATTEMPT FAILED.')
 5260    FORMAT (' ?INPBAT-/*SIMULATE COMMAND BYPASSED DUE TO PREV'
         1,'IOUS CONTROL CARD ERRORS'/'     ERROR COUNT='I//)
 5280    FORMAT (18XA5,8(7XA5))
 5290    FORMAT(' %Simulating a ',A5,' with the truck version of HEVSIM')
 5300    FORMAT(/' [SIMULATING]')
 5320    FORMAT(/' % INPBAT - NO TORQUE CONVERTERS LOADED BUT'
         1,/13X'ARE ALL GEARS LOCKED UP.'///' INPBAT-ATTEMPING TO LOAD'
         2,' DEFAULT TORQUE CONVRTERS CODY2 AND COCY2')
 5340    FORMAT (18XA5,7XF6.1,6XA5,7XF6.1)
 5350    FORMAT (18XA5,7XF6.1,6XA10)
 5360    FORMAT (18XA5,7X20I2)
 5380    FORMAT (18XA10,2XA5,3(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)
 5480    FORMAT(/' % INPBAT-ACCESSORY 'A10' NOT LOADED.')
 5500    FORMAT(/' % INPBAT-NO ACCESSORIES LOADED.')
 5520    FORMAT(/' PARTS ZEROED;')
 5540    FORMAT (6XA10,2XA5,17X20I2)
 5560    FORMAT(6XA10,2XA5,1XA5)
 5580    FORMAT (6XA10,2XA5,11X,I1,5X,20I2)
 5600    FORMAT('1 HEVSIM 'A5,2X02'('O2') STATUS REPORT'/2X,34('-'))
 5620    FORMAT(/' BATCH MODE'$)
 5640    FORMAT(/' DIALOGUE MODE')
 5660    FORMAT('+'14X'/PTY')
 5680    FORMAT('+'14X'/TTY')
 5700    FORMAT(' SIMULATION MODE-'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-NOT LOADED')
 5840    FORMAT(1XA5' CONVERTER-'A10)
 5860    FORMAT(' VEHICLE-'A10)
 5880    FORMAT(' TRANSMISSION - 'A10)
 5900    FORMAT(' GEARS-NOT LOADED')
 5920    FORMAT(' GEAR #'I2'-'A10' UNASSIGNED')
 5940    FORMAT(' GEAR #'I2'-'A10' ASSIGNED TO ENGINE-'A10)
 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 HEVSIM CONTROL FILE 'A6'['A10'['05','03']')
 6320    FORMAT(' ENTER TRANSMISSION NAME: '6)
 6340    FORMAT(A10)
 6360    FORMAT(' ? ',A10,' NOT A VALID NAME.')
 6380    FORMAT(' HOW MANY GEARS? (MAX. 20) '8)
 6400    FORMAT(' ENTER NAME FOR GEAR #',I2': '8)
 6420    FORMAT(' ENTER COMMENT: (1 LINE) '8)
 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',T19,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,T31,2F6.0)
 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)
 7400    FORMAT(' Enter accessory name:' 8)
 7420    FORMAT(' Dutycycle for accessory ',A10
        2,' modified from'G10.4,' TO',G10.4)
10400    FORMAT(' Enter new output file: '6)
C
         END
```

```
*               **** INPDIA ****
*
      SUBROUTINE INPDIA ( IECOND , ENDET )
C
      INCLUDE 'COMMS/NOLIST'
C
      DOUBLE PRECISION HINPDI
C
      LOGICAL ENDET,LPAUSE,LTRANS
C
      DIMENSION HPART(14),CNVTYP(3),IPCRS(14)
C
      DATA (HPART(I),I=1,NPART)/'ACCES','CONVE','DRIVI','ENGIN','GEAR '
     1,'ROUTE','SHIFT','VEHIC','TIRE','TRANS','AXLE'/
C
      DATA CNVTYP/'DRIVECOASTTORQU'/,YES/'Y'/,RNO/'N'/,DIA/'D'/
     1,HON/'ON'/,HOFF/'OFF'/
C
      DATA   HCOMND/ 5HUSE    ,5HSIMUL ,5HMODIF ,5HPRINT ,5HLIMIT ,
     1               5HDUMP   ,5HREMAP ,5HLOCKU ,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 ,5HCITY  ,5HDDT   ,
     6               5HPDUMP  ,5HDELET ,5HEXIT  ,5HDIAIO ,5HCONTI ,
     7               5HLPT    ,5HHELP  ,5HOUTPU ,5HTRANS  ,5HAXLE /
     1,(IPCRS(I),I=1,11)/5,2,6,1,4,8,7,3,9,10,11/
     2,HINPDI/'INPDIA'/
C
C*******************************************************************************
C
      LPAUSE=.FALSE.                                    !INITIALIZE NOT IN PAUSE MODE.

      GO TO 9
C
      ENTRY INPPSE(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 IECOND=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=950) COMND,FSPECS         !GET COMMAND.
      IF(COMND.EQ.HBLANK) GO TO 11                      !(EXTRANEOUS "<CR>"?) YES,TRY AGAIN.

      IF((COMND.AND.WMASK(1)).NE.('@'.AND.WMASK(1)))GO TO 13
      REREAD 1021,FSPECS
      ICMD=16
      GO TO 206
C
13    CALL SLOOKP(COMND,40,HCOMND,ICMD,$20)             !YES.
      GO TO 10                                          !NO, GO PROMPT.

C
   17 WRITE(5,1043) HCOMND(ICMD)                        !%BATCH COMMAND NOT IMPLEMENTED FOR DIALOGUE MODE.
```

```
      GO TO 10                                                          !GO PROMPT.
C
   20 PNAME=PSPECS(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),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 POROTS.

      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 NOT BATCH

      GO TO 10                                                          !CONTINUE

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 SLOOKP(UT,NUMPAR,HPART,IPRNTI,$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.
```

```
      IPRNTI=IP                                                    !SET PART TYPE PTR.
C
C  NORMAL BRANCH
C
  102 GO TO (110,120,130,140,1510,160,170,180,190,1500,1520),IPRNTI !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),IPRNTI           !ON "ERR=" GET BACK TO WHERE WE WERE.
C
C=================================================================
C
C  *USE ACCESSORY
C
  110 IPRNTI=1                                                         !FLG ACCES.
C
  111 IR=1                                                         !PTR TO NEXT STATEMENT IN CASE OF ERR=.
C
  115 WRITE(5,1001)                                                !ASK # OF ACCES TO BE USED.
C
      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 198                                       !(ANNY ACCES TO LOAD?)NO, DONE.
C
      IR=2                                                         !SET PTR TO ST. # 112.
C
      DO 117 I=1,NUM                                               !ASK FOR NEEDED # OF ACCES.
C
  112 WRITE(5,1003) I                                              !ASK ACCES NAME.
      READ(5,1004,END=920,ERR=990) PNAME                           !GET NAME.
      GO TO 182                                                    !GO GET IT.
C
  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 IPRNTI=2                                                         !SET COMMAND PTR.
C
      DO 126 M=1,2                                                 !NEED 2 CONVERTERS COAST AND DRIVE.
C
  122 WRITE(5,1006) CNVTYP(M)                                      !ASK CONV NAME.
      READ(5,1004,END=920,ERR=990) PNAME                           !GET NAME.
      GO TO 182                                                    !GO GET IT.
C
  126 CONTINUE                                                     !(DONE?)NO.
C
      GO TO 198                                                    !YES.
C
C=================================================================
C
C  *USE DRIVING SCHEDULE
C
  130 IPRNTI=3                                                         !SET COMMAND PTR.
```

```
  131   WRITE(5,1003)                                                    !ASK FOR ENG NAME.
        READ(5,1004,END=920,ERR=990)  PNAME                              !GET NAME.
        GO TO 182                                                        !GO GET IT.

C
C=================================================================
C
C  *USE ENGINE
C
  140   IPRNTI=4                                                                 !SET COMMAND PTR.

  141   WRITE(5,1010)                                                    !ASK FOR ENG #,NAME,AND GEAR ASSIGNMENTS.

        READ(5,1011,END=920,ERR=990)  IENG,PNAME,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)  PNAME                                              !GET NAME.
        IF(PNAME.EQ.DBLANK) GO TO 141                                    !("<CR>"?)YES, GO ASK FOR ALL INFO ON ENG.[TO PROVIDE EXIT F
ROM 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   IPRNTI=10                                                        !SET POINTER.

C
        WRITE(5,1080)
        READ(5,1004,END=920,ERR=990) PNAME                               !GET NAME.
C
        IF(PNAME.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=990) 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 IPRNTI=5                                          !FIG GEAR USE.
  151 IR=1                                              !FIG ERR= BRANCH TO NXT ST.
  152 WRITE(5,1013)                                     !ASK # OF GEARS TO BE USED.
      READ(5,1002,END=920,ERR=990) NUM                  !GET #.
      IF(NUM.LT.1 .OR. NUM.GT.20) GO TO 152             !(LEGAL #?)NO, TRY AGAIN TURKEY.
C
      IMOD=0                                            !FIG DO LOOP IN CONTROL.
      IR=2                                              !FIG ERR= ER 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 182                                         !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                                            !FIG 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.DBLANK) 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 198
      IPRNTI=11
```

```
      WRITE(5,1004)
      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.
      IECOND=IPCRS(IPRNTT)                                    !SET PART TYPE PTR AS USED BY OTHER ROUTINE.
      CALL USECMD(IECOND)                                     !GO LOAD PART.
      GO TO (185,184,900),IECOND                              !GOT IT /DIDN'T/OOPS.
  184 CALL NOPART(5,PNAME,IPCRS(IPRNTT),CNVTYP(M))            !?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=950) 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.RNO) 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=LIMITY                              !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.
        IPRNTI=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(IPAUSE) 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) COMND                           !?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 (ECF OR ERR?)YES.
        WORD=HCOMND(8)                                !PASS COMMAND NAME.

        CALL DALCMD(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=HCOMND(9)                                !PASS COMMAND NAME.

        CALL DALCMD(IECOND)                           !TRY IT.
        IF(IECOND) 900,229,224                        !(IMPOSS ERR/NO ERR/ILL GEAR #?).

  224   WRITE(5,1025) IDATA(IECOND)                   !?ILL GEAR #.
```

```
        GO TO 220                                                       !TRY AGAIN.
C
  229   IF(.NOT. LASK) 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.HBLANK) GO TO 239                                    !(ANY MODIFY COMND?)NO, GET OUT.

        CALL MODCMD(IECOND)                                             !YES, LET'S TRY IT.

        GO TO (230,230,234,236),IECOND                                  !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. LASK) 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.HBLANK) GO TO 249                                    !(ANY ANS?)NO,LEAVE CURRENT OPTION IN EFFECT.

        CALL LIMCMD(IECOND)                                             !YES, GO SET IT.

        IF(IECOND.EQ.1) GO TO 249                                       !(ERR?) NO.
        WRITE(5,1033) WORD                                              !YES, REPORT IT.

        GO TO 241                                                       !TRY AGAIN.
C
  249   IF(.NOT. LASK) GO TO 10                                         !(TELL MODE?)YES, GO PROMPT.

        GO TO 280                                                       !NO, NEXT,
C
C*************************************************************************
C
C  *STATUS
C
```

```
  250   ICMD=12                                                  !FLG STATUS COMMAND.
  251   IECOND=5                                                 !PASS UNIT # TO SEND JCT.
        CALL STSCMD(IECOND)                                      !GO DO ?*STATUS/.
        IF(IPAUSE) WRITE(5,1047) ISEGT                           !(¬P MODE?)YES, REPORT IT.
        IF(IECOND.EQ.1) GO TO 900                                !(ERR?)YES, IMPOSSIBL.
        IF(.NOT. LASK) 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) WORD,DATA(1),DATA(2)                !GET OPT AND LIMITS.
        CALL DEBCMD(IECOND)                                      !GO SET OPTION.
        GO TO (269,262,264),IECOND                               !(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. LASK) 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(PNAME.EQ.DBLANK) GO TO 275                            !)ZERO ACCES?)NO.
        UT=HPART(1)                                              !YES, SET PART TYPE TO ACCES.
        UN=PNAME                                                 !GET ACCES NAME.
        GO TO 277                                                !GO ZERO ACCES FROM CORE.
  275   UT=HBLANK                                                !FLG ZERO ALL.
        NENGI=0                                                  !ENG LOADED CNT.
  277   CALL ZERCMD(IECOND)                                      !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(5,1023,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.

        LTYOLD=.TRUE.                                            !SAVE FOR RESET.

        GO TO 289
  283   LIMTTY=.FALSE.                                           !SET TTY PRINT ON.

        LTYOLD=.FALSE.                                           !SAVE FOR RESET.

        GO TO 289
  289   IF(.NOT. LASK) GO TO 10                                  !(TELL MODE?)YES, GO PROMPT.

        GO TO 260                                                !NXT Q IN ASK SEQ.

C
C*****************************************************************************
C
C  *DELETE
C
  290   ICMD=31                                                  !SET COMMAND PTR.


  291   WRITE(5,1044)
        READ(5,1023,END=920,ERR=990) UT
        IF(UT.EQ.HBLANK) GO TO 10
C
  295   WRITE(5,1048)
        READ(5,1004,END=920,ERR=990) UN
        IF(UN.EQ.DBLANK) GO TO 295
        CALL DRPCMD(IECOND)
        GO TO (291,297),IECOND
C
  297   CALL NOPART(5,UN,IPCBS(IPRNTT),CNVTYP(3))
        GO TO 291
C*****************************************************************************
C
C  *DUMP
C
  300   ICMD=6                                                   !FLG DUMP COMMAND.

        IECOND=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.RNO) GO TO 307                    !(ANS NO?)NO, BAD ANS, GO ASK AGAIN.

      IECOND=-1                                   !ANS=NO, FLG DUMP ONLY.

      GO TO 315
C
C*****************************************************************************
C
C  *DIRECT
C
  310 ICMD=11                                     !FLG COMMAND.

      IECOND=1                                    !FLG DIR ONLY.

  315 UN=DSTAR                                    !ASSUME WILD NAME.

      UT=HSTAR                                    !ASSUME WILD PART TYPE.

      WRITE(5,1044)                               !ASK PART TYPE.

      READ(5,1023,END=920,ERR=900) 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=900) PNAME          !GET NAME (EOF OR ERR?)YES.
      IF(PNAME.NE.DBLANK) UN=PNAME                !(ANY ANS?)YES, OVERRIDE ASSUMPTION.

  318 CALL DMPCMD(IECOND)                         !GO DUMP/DIRECT.
      DMPTTY=.FALSE.
      GO TO 10
C
C*****************************************************************************
C
C  *PDUMP
C
320   ICMD=31
      IECOND=31
      WRITE(5,1044)
      READ(5,1023)UT
      IF(UT.EQ.HBLANK)GO TO 10
      CALL SLOOKP(UT,NPART,HPART,IPRNTT,$340)
      WRITE(5,1042)UT
      GO TO 320
C
340   WRITE(5,1050)
      READ(5,1004)PNAME
      CALL USECMD(-IPCRS(IPRNTT))
      GO TO 320
C
C
C*****************************************************************************
C
C  *TITLE
C
 400  IECOND=1                                    !DUMMY TO FULFIL ARGUMENT REQUIREMENTS.

      WRITE(5,1060)
      CALL TTLCMD(IECOND)
```

```
C
      GO TO 10
C
C
C
C*****************************************************************************
C
C  *IPT
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  LLPT=.FALSE.
      GO TO 508
 506  LLPT=.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   ERROR CONTROL SECTION
C
C       FATAL ERRORS DETECTED ON RETURN FROM SUBROUTINE ,
C               BUT NO RECOVERY METHOD YET IMPLEMENTED OR IMPOSSIBLE
C               ERROR.
C
  900 WRITE(5,901) HCOMND(ICMD),ICMD,IPRNTT,IECOND                    !SOME HELP TO JCT.
      WRITE(6,901) HCOMND(ICMD),ICMD,IPRNTT,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**'5X45,5X,3I3' /HC,IC,IP,IE/'/'1')
C
      IF(TTY) 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                                                       !FIG ZERO ALL.

        CALL ZERCMD(IECOND)                                             !GO ZERO.
        CALL TTYCLR                                                     !CLR TTY INP BUF.

        RETURN                                                          !BYE.

C
C  HERE ONLY IN PTY SUBMODE- END OF BATCH CONTROL FILE
C
  920   WRITE(5,921)                                                    !TO JCT.
        WRITE(6,921)                                                    !TO IPT.
  921   FORMAT(//' ? INPDIA-UNEXPECTED END OF BATCH CONTROL FILE.')
C
        ENDET=.TRUE.                                                    !FIG EOF.
        RETURN                                                          !BYE.

C
C  HERE ON TTY FORMAT I/O ERROR
C
  950   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, GC PROMP
T.
  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,     900,900,900,900,900,900,261,900,291,900
        3,     291,291,900,900,900,900,900,900,900,900),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: '$)
 1021   FORMAT(1X,10A10)
 1022   FORMAT(/' SIMULATE ? (ANS Y/N/D): '$)
 1023   FORMAT(A5,1X10A10)
 1024   FORMAT(/' ENTER GEAR NUMBERS TO BE LOCKED UP: '$)
 1025   FORMAT(/' ? INPDIA-'I3' IS ILLEGAL GEAR NUMBER.')
 1026   FORMAT(/' ENTER GEAR NUMBERS TO BE UNLOCKED: '$)
```

```
 1027   FORMAT(/' *MODIFY(NEW VALUE,ITEM): '$)
 1028   FORMAT(A5,10F)
 1029   FORMAT(/' ? INPDIA- 'A5' UNKNOWN MODIFY COMMAND.')
 1030   FORMAT(/' ? INPDIA- NO PART LOADED FOR 'A5' MODIFY.')
 1031   FORMAT(/' ? INPDIA- INPBAT ERROR! PLEASE RELOAD PARTS DATA.')
 1032   FORMAT(/' *LIMIT PRINT: '$)
 1033   FORMAT(/' ? INPDIA- 'A5' UNKNOWN LIMIT PRINT COMMAND.')
 1034   FORMAT(/' ? INPDIA- FORMAT I/O ERROR.')
 1035   FORMAT(/' ? INPDIA- PARTS MISSING.')
 1036   FORMAT(/' DEBUG: '$)
 1037   FORMAT(/' ? INPDIA- "'A5'" UNKNOWN DEBUG COMMAND.')
 1038   FORMAT(/' ? INPDIA- DEBUG START VALUE LARGER THAN STOP VALUE.')
 1039   FORMAT(/' ENTER GEAR # , NAME: '$)
 1040   FORMAT(F,A5)
 1041   FORMAT(/' *'$)
 1042   FORMAT(/' ?'A5'?')
 1043   FORMAT(/' ?INPDIA- "'A5'" UNIMPLEMENTED VENSIM COMMAND FOR'
        1,' DIALOGUE MODE.')
 1044   FORMAT(/' PART TYPE: '$)
 1045   FORMAT(/' TTY OUTPUT(ON/OFF): '$)
 1046   FORMAT(/' ?INPDIA-"'A5'" ILLEGAL COMMAND WHEN IN ¬P(PAUSE)'
        1,' MODE.')
 1047   FORMAT(/' %INPDIA- IN ¬P(PAUSE) MODE ISEG:'I4' **BE CAREFUL**')
 1048   FORMAT(/' ENTER NAME OF PART TO BE DROPPED: '$)
 1049   FORMAT(/' ENTER PART NAME OF TIRE TO BE USED: '$)
 1050   FORMAT(/' ENTER PART NAME: '$)
 1051   FORMAT(/' ? INPDIA- 'F' IS ILLEGAL VALUE FOR 'A5' MODIFY.')
11052   FORMAT(/' DUMP TO TTY?(ANS Y/N) '$)
 1052   FORMAT(/' DO YOU WANT A DIRECTORY OF PARTS DUMPED?(ANS Y/N): '$)
 1060   FORMAT(/' ENTER RUN TITLE: '$)
 1062   FORMAT(/' LPT OUTPUT(ON/OFF): '$)
 1070   FORMAT(/' ENTER SIMULATION MODE: '$)
 1080   FORMAT(/' ENTER PART NAME OF TRANSMISSION (OR <CR> FOR GEARS): '
        2,      $)
 1082   FORMAT(/' ENTER GEAR NUMBER: '$)
 1084   FORMAT(/' ENTER AXLE NAME: '$)
        END
```

```
*                **** ITERAT - NEW VERSION ****
*
      SUBROUTINE ITERAT(TITER)
*
*  MODIFIED 28 AUGUST 1980: SOMERS
*
      INCLUDE 'COMMS/NOLIST'
*
      LOGICAL PRINT6
      DATA NAME/'ITERA'/
*
      PRINT6 = .FALSE.
*     PRINT6 = .TRUE.
      LITER  = .TRUE.
      APWOO  = 0.01*PWOT(ISEG)
      TOL    = 0.005*(TWOT - TMIN); DACC = 0.2; NITER = 0
*
      IF(PRINT6)WRITE(6,9001) TITER,ACCEL,TORQE,TOL,TWOT,TMIN
 9001 FORMAT(' $ITERAT - TITER,ACCEL,TORQE,TOL,TWOT,TMIN ->',/6(2XG)/)
      CALL TAPEWR(NAME,1)
*
*     CALL GETACL(TITER)---REPLACED BY FOLLOWING EQUATION
*
      XYZ   = 0.75*BVG*BVGEFF*TR
      ACCEL = (XYZ*(TITER - TORQA)*GRAT(NGEAR)*RAR(NAX)/WRAD
     1 -AA1 - AA2*VOLD - AA3*VOLD**2 - BPER*WGT)/AA4
*
      CALL GOBACK
*
      CALL TAPEWR(NAME,2)
*
      IF(TORQE > TITER) DACC = -DACC
      TORD  = TITER - TORQE
*
   10 CONTINUE
      IF(NITER < 1) GO TO 15
      GO TO (300,300,500,15) ITS
*
  300 CONTINUE
      IF(TITER > 1) TITER = TWOT
      IF(TITER < 1) TITER = TMIN
      GO TO 15
*
  400 CONTINUE
      PITER = RTHR*DT*PCOLD
      TITER = TMIN + 0.01*PITER*(TWOT - TMIN)
      GO TO 15
*
  500 CONTINUE
      TITER = TMIN + APWOO*(TWOT - TMIN)
*
   15 CONTINUE
      NITER = NITER + 1
      ACCEL = ACCEL + DACC
      CALL GOBACK
      TORDO = TORD
      TORD  = TITER - TORQE
*
      IF(PRINT6)WRITE(6,9000) TITER,TORD,ACCEL,DACC,TORQE,NITER,ITS
 9000 FORMAT(' $ITERAT - TITER,TORD,ACCEL,DACC,TORQE,NITER,ITS ->',
     1 /X5G,2X2I3,/)
```

```
      CALL TAPEWR(NAME,3)
*
      IF(ABS(TORD) < TOL) GO TO 20
      IF(TORD*TORDO > 0.0) GO TO 10
      DACC = -0.5*DACC
      GO TO 10
*
   20 CONTINUE
      LITER = .FALSE.
*
      CALL TAPEWR(NAME,4)
*
      RETURN
      END
```

```
*               **** KPTIME ****
*
      SUBROUTINE KPTIME(ITASK)
C
C     ENTRY POINTS:   KPTIME
C
C     CALLED BY:      SIMCTR, SIMINT, SIMSTS
C
C     EDIT HISTORY:
C
C     [721]/SS-12-11-78       IF ANY DRIVING SEGMENT TAKES OVER 10 MIN OF CPU,
C                             RESET, CUZ PROBABLE ERROR.

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.
      IF(CPUS.LT.600.) RETURN                                    ![722]
C
      WRITE(5,100)
      WRITE(6,100)
  100 FORMAT(' ?KPTIME - Driving segment taking over 600 sec CPU.',/,
     2       ' Probable error.')
      CALL RESETM
      END
```

```
*               **** MODSL ****
*
        SUBROUTINE MODSL    ( MODE,PCT )
C
C       ENTRY POINTS:    MODSL
C
C       SUBROUTINES CALLED:      PRNTPD
C
C       CALLED BY:       INPDAT
C
C*****************************************************************
C
C
        LOGICAL      ISCALE,LIMPRN,LTRRZ
C
        DOUBLE PRECISION    SNAME,GNAME,DATE
C
        COMMON /PRNLIM/  IIMPRN,MILIM,SECLIM,ENDLIM,ALIMN,ISCALE
        COMMON /CONST/   FRC1,FRC2,FAC,CD,AREA,VWIND,WGT,FGC,WRAD,BAR(3)
        2,  GRAT(20),NUMG,NGEAR,AIW,AIP,AI2,ERAT(20),EBAR(2,3)
        3,  AIE,AIA,AI1,BPER,CDC,PHI,PSI,AIGIN(20),AIGOUT(20),WLSG,LTRRZ
        4,NGRLSS(20),GRPM(20,20),GRTORQ(20,20),GNAME(20),GCOM(16),FRC4
        COMMON /SHFTI/   SNAME,SCOM(16),GOVPSI(4),OUTRPM(4),NGPT,IGF(60),
        2       IGT(60),SHFTIM(60),SHFTPT(10,60),SHFTRP(10,60),LVAC,LENG,
        3       GDAT,NSPTS(60),LDETNT,DETPT(60),DETRPM(60),PARAB,LDETE,
        4       LDETV,GOVLIN,IAP(60),IAT(60),NUMBSL,RERAT(47,60),
        5       IRER,IRERO,NRER
        COMMON /V2MISC/  LOCKUP(20),DATE,NPARTS(11),DEFDT,NORUN,NENG,
        1                 IUNIT,IPART,IPMODE,IRMODE,ISMODE,NSGEAR(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
        NUMBSL = ( NUMG - 1 ) * 2
        DO 200  I = 1,NUMBSL
C
C     SELECT UP/DOWN SHIFT LINE TO MODIFY
C
        GO TO ( 110,120 ) , MODE
  110   IF ( IGF(I).GE.IGT(I) )  GO TO 200
        GO TO 140
  120   IF ( IGF(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 ( LIMPRN )  GO TO 900
      GO TO ( 310,320 ) , MODE
  310 WRITE (IUNIT,1310)  PCT
      GO TO 340
  320 WRITE (IUNIT,1320)  PCT
  340 IPART  = 7
      CALL PRNTPD
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
```

```
*               **** NOPART ****
*
      SUBROUTINE NOPART(IUNIT,PNAME,IPRNT,IFLG)
C
C     ENTRY POINTS:    NOPART
C
C     CALLED BY:       INPBAT, INPDIA
C
C*****************************************************************
C
C
      DOUBLE PRECISION PNAME
C
C*****************************************************************************
C
      GO TO (10,20,30,40,50,60,70,80,90,100,110),IPRNT                    !WHAT PART TYPE NOT FOUND.

      WRITE(IUNIT,1030)PNAME,IPRNT
      GO TO 900
C
   10 WRITE(IUNIT,1012) PNAME                                             !ENG NOT FOUND.
      GO TO 900
C
   20 WRITE(IUNIT,1007) IFLG,PNAME                                        !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)PNAME                                              !TRANSMISSION NOT FOUND.

      GO TO 900
C
  110 WRITE(IUNIT,1024)PNAME                                              !AXLE NOT FOUND.
C
  900 RETURN
C
C*****************************************************************************
C
C   FORMAT STATEMENTS
C
 1005 FORMAT(/' ? ACCESSORY '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,' NOT FOUND IN PARTS DATA BASE')
 1024   FORMAT(/' ? AXLE ',A10,' NOT FOUND IN PARTS DATA BASE')
 1030   FORMAT(/' ? Part ',A10,' not found. (Part code - ',I,')')
C
        END
```

```
*               **** NRPTS ****
*
      FUNCTION NRPTS(X,MAX)
C
C     GIVEN A REAL ARRAY, RETURN # OF LAST LOCATION
C     CONTAINING A NON-ZERO.
C
      DIMENSION X(MAX)
      DO 20 I=MAX,1,-1
      IF(X(I).NE.0.)GO TO 40
20    CONTINUE
      I=1
40    NRPTS=I
      RETURN
      END
```

```
C**** NWRCNT ****
      FUNCTION NWRCNT(STRING,LWRDS)
C
C     ENTRY POINTS:   NWRCNT
C
C     CALLED BY:      DSKDIR, INPBAT, 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
```

```
*               **** PRNOUT ****
*
        SUBROUTINE PRNOUT
C
C       ENTRY POINTS:   PRNOUT
C
C       SUBROUTINES CALLED:     PRNTPD
C
C       CALLED BY:      DSK,    DSKDIR, INPBAT, REMAP,  SCALEN
C
C*****************************************************************
C
C
        INCLUDE 'COMMS/NOLIST'
C
        DIMENSION  AKK(20),SOUZ(20),TOUZ(20)
C
C
C
C*****************************************************************
C
C
        IF (LIMPRN) 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 FIG.
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=NRPM(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=NRPM(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./ERPM(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(PTOR) GO TO 1002
        IF(PBMEP) GO TO 9
C
C     CONVERT LB-FT TO HP
C
        DO 8 I=1,NRDUM
        DUM=ERPM(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(PRPM) GO TO 1004
C
C     CONVERT RPM TO PISTON SPEED
C
        DUM=STROKE/6.
        DO 1003 I=1,NRDUM
 1003   ERPM(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 PRNTPD
C
C     CONVERT UNITS BACK
C
        IF(PRPM) GO TO 15
C
C     PISTON SPEED TO RPM
C
        DUM=6./STROKE
        DO 12 I=1,NRDUM
   12   ERPM(IENG,I)=DUM*ERPM(IENG,I)
        ENMIN(IENG)=DUM*ENMIN(IENG)
        RPMAX(IENG)=DUM*RPMAX(IENG)
        SPIDLE(IENG)=DUM*SPIDLE(IENG)
```

```
   15   IF(PIOB) GO TO 20
        IF(PBMEP) GO TO 17
C
C     HP TO LB-FT
C
        DO 16 I=1,NRDUM
        DUM=5252./ERPM(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(PLBHR) GO TO 25
        IF(PBSFC) GO TO 23
C
C     GAL/HR TO LB/HR
C
        DUM=7.480520/(FSPGR*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=ERPM(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 PRNTPD                                          !GO DO ACTUAL PRINTING.
C
C*********************************************************************
C
 9000   RETURN                                               !DONE, BYE.
C
        END
```

```
*                **** PRNTPD ****
*
      SUBROUTINE PRNTPD
*
*  MODIFIED 2 JULY 1980: SOMERS
*
C
C     ENTRY POINTS:    PRNTPD
C
C     SUBROUTINES CALLED:      DSK
C
C     CALLED BY:       PRNOUT
C
C*****************************************************************
C
C
      INCLUDE 'COMMS/NOLIST'
C
      LOGICAL FIRST,LSAVE,LTRANS
C
      DIMENSION AKK(20),SOUZ(20),TOUZ(20),HPART(9)
C
      DOUBLE PRECISION HNA,EOUT(10)
C
      DATA HPART/3*' ','GEAR ACCESDRIVISHIFTROUTETIRE'/
     1,HNE/' **N/A**'/,HAXLE/'AXLE'/
C
C*****************************************************************
C
C
      FIRST=.TRUE.
      ISUNIT=IUNIT
      IF(NXTPG.LE.0) NXTPG=1                                    !(PG CNT SET?)NO, SET IT.
C
      WRITE (IUNIT,1000)  DATE,NXTPG                            !TOP OF PAGE.
      NXTPG=NXTPG+1                                             !INC PAGE COUNT.
C
      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  = PSPGR * 8.3452
      IF(.NOT.LDIES) WRITE(IUNIT,1100) ENAME(IENG),ECOM,ICYL,FPGAL,BORE
     2,EINER,STROKE
     1,DISP,NRPM(IENG),THRMIN,THRMAX
      IF(IDIES) WRITE(IUNIT,1101) ENAME(IENG),ECOM,ICYL,FPGAL,BORE
     2,EINER,STROKE
     1,DISP,NRPM(IENG),THRMIN,THRMAX
      IF ( PRPM   )  WRITE (IUNIT,1102)  ENMIN(IENG),RPMAX(IENG)
      IF ( PPS    )  WRITE (IUNIT,1104)  ENMIN(IENG),RPMAX(IENG)
      NRDUM  = NRPM(IENG)                                       !GET # OF SPEED PTS ON ENG MAP.
C
      IPAGE  = 2                                                !SET LINE COUNT.
C
      DO 160  I = 1,NRDUM                                       !LOOP THRU ALL SPEED PTS.
```

```
      IF ( PEPM   )  WRITE (IUNIT,1110)  EBPM(IENG,I)
      IF ( PPS    )  WRITE (IUNIT,1112)  EBPM(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 ( PTOR   )  WRITE(IUNIT,1120)  (EMAP(I,J,1+KENG),J=IB,IE)
      IF ( PBMEP  )  WRITE(IUNIT,1121)  (EMAP(I,J,1+KENG),J=IB,IE)
      IF ( PHP    )  WRITE(IUNIT,1122)  (EMAP(I,J,1+KENG),J=IB,IE)
      IF ( PLBHR  )  WRITE(IUNIT,1123)  (EMAP(I,J,2+KENG),J=IB,IE)
      IF( .NOT. PBSFC )  GO TO 130
C
C     SPECIAL HANDLING FOR PRINTING OF BSFC 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 ENCODE(6,11124,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( PBALHR )  WRITE(IUNIT,1125)  (EMAP(I,J,2+KENG),J=IB,IE)
      WRITE (IUNIT,1130)  (EMAP(I,J,3+KENG),J=IB,IE)
      WRITE (IUNIT,1132)  (EMAP(I,J,4+KENG),J=IB,IE)
      IPAGE  = IPAGE + 1                                      !INC LINE CNT.
      IF ( IPAGE.NE.10 .OR. I.EQ.NRDOM )  GO TO 140           !(PAGE?)NO.
      WRITE (IUNIT,1000) DATE,NXTPG                           !(YES.
      NXTPG=NXTPG+1                                           !INC PAGE CNT.

      IF(NTOR(IENG,I).LE.10) WRITE(IUNIT,1002)                      !(MORE THAN 10 LOAD PTS?) NO,OUTPUT BBINK LINE.

      IPAGE  = 0                                              !ZERO LINE CNT.

  140 IF ( .NOT.LPRNT         )  GO TO 150
      IF ( NTOR(IENG,I).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.EQ.20           )  GO TO 160
      IE     = 20
      GO TO 120
  160 CONTINUE
      GO TO 999                                               !BYE.
C
C*********************************************************************
C
C     PRINT TORQUE CONVERTER DATA
```

```
C
  200   IF ( COAST      )  WRITE(IUNIT,1200)  CNAME                    !(COAST CONVERTER?)YES.
        IF ( .NOT.COAST )  WRITE(IUNIT,1202)  CNAME                    !(DRIVE CONVERTER?)YES.
        WRITE (IUNIT,1210) CCOM,CDIAM,AI1,AI2
        IF ( ABS(CONTOR).GT..001 )  WRITE (IUNIT,1212)  CONTOR
        DO  220  I = 1,NTORP
        SOUZ(I)    = SOUT(I) / SPIN(I)
  220   TOUZ(I)    = TOUT(I) / TIN(I)
        IB     = 1                                                     !SET PTR TO 1ST DATA FLD.

        IE     = NTORP                                                 !SET PTR TO LAST DATA FLD.

        IF ( NTORP.GT.10 )  IE = 10                                    !(MORE 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)
        WRITE (IUNIT,1234)  (SPIN(I),I=IB,IE)
        IF ( .NOT.COAST  )  WRITE (IUNIT,1236)  (AKD(I),I=IB,IE)              !(DRIVE CONVERTER?)YES.
        IF ( IE.EQ.NTORP )  GO TO 999                                  !(MORE THAN 10 SPD PTS?)NO.

        WRITE (IUNIT,1002)                                             !YES, OUTPUT BLANK LINE.

        IB     = 11                                                    !SET PTR TO NXT DATA FLD TO BE PRINTED.

        IE     = NTORP                                                 !SET PTR TO LAST DATA FLD.

        GO TO 230
C
C*****************************************************************
C
C     PRINT VEHICLE DATA
C
  300   IWLSG = WLSG
        WRITE(IUNIT,1300) VNAME,VCOM,WGT,AREA,IWLSG,CD,CDC,AIP,BVG,
        1 BVGEFF
        GO TO 999
C
C*****************************************************************
C
C     PRINT GEAR DATA
C
  400   LPRAMS=.FALSE.
C
  410   WRITE(IUNIT,1400) GNAME(NGEAR),TCOM,GRAT(NGEAR),AIGIN(NGEAR)
        1,EFAC(NGEAR),AIGOUT(NGEAR)                                    !GEAR DATA.
        IF(NGRLSS(NGEAR).GT.1)  GO TO 420                              !(ANY SPINN LOSS DATA?)YES.

        WRITE(IUNIT,1410) HPART(4)                                     !NO, REPORT IT.
        GO TO 999                                                      !BYE.
  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                                                      !BYE.
C
C*****************************************************************
C
C     PRINT ACCESSORY LOSS DATA
C
  500   WRITE (IUNIT,1500)  ANAME(NACC),ACOM,AIAS(NACC)
        2,ACCSR(NACC),DUTCYC(NACC)                                     !ACCES HEADER.
        DO 520 IA=1,NNA(NACC)
```

```
      TEMP=ACCS(IA,NACC)/ACCSR(NACC)
      WRITE(IUNIT,1520) TEMP,ACCT(IA,NACC)                    !ACCES TORQUE LOSS DATA.
520   CONTINUE
      GO TO 999                                               !BYE.
C
C*****************************************************************
C
C     PRINT DRIVING SCHEDULE
C
600   WRITE (IUNIT,1600)  DNAME,DCOM,TO,DO,VO,AO,NGO,NAO      !DRIVE SCHD HEADE AND INITIAL CONDITION.
      WRITE (IUNIT,1602)                                      !SEGMENT HEADER.

      IPAGE=25                                                !SET LINE USED CNT.

      NSEGA=NSEG                                              !GET SEG CNT.

620   DO 690  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,ATHOID(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 OF 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 TO 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 .OR. 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=ISAVE                                                    !RESTORE.
      IF(IPRNT.EQ.6) GO TO 999                                        !(DSK ERR?) NO, BYE.

      WRITE(JCT,1640) HPART(6),DNAME                                  !YES, *IMPOSS, BUT REPORT.

      IPRNT=-10                                                       ! SET ERR FIG

      CALL TRACE                                                      ! SOME HELP?

      GO TO 999                                                       ! GOOD LUCK! BYE.

C
C*****************************************************************
C
C     PRINT SHIFT LOGIC
C
  700 WRITE (IUNIT,1700)  SNAME,SCOM,NUMG                             !HEADER.
      NSL    = NUMBSL                                                 !CALC # OF SHIFT LINES.

      IPAGE=2                                                         !SET LINE USED CNT.

      DO 770  I = 1,NSL                                               !LOOP THRU ALL SHIFT LINES.

      WRITE(IUNIT,1708)IGF(I),IGT(I),IAF(I),IAT(I)
      WRITE (IUNIT,1714)  SHFTIM(I)
      N      = NSPTS(I)
      IF ( LVAC       )  WRITE (IUNIT,1720)  (SHFTPT(J,I),J=1,N)
      IF ( LTHR       )  WRITE (IUNIT,1721)  (SHFTPT(J,I),J=1,N)
      IF(.NOT.(LVAC.OR.LTHR))  WRITE(IUNIT,1722)  (SHFTPT(J,I),J=1,N)
      IF ( .NOT.PARAB )  GO TO 730
      WRITE (IUNIT,1724)  (SHFTBP(J,I),J=1,N)
      GO TO 740
  730 IF ( LENG       )  WRITE (IUNIT,1730)  (SHFTRP(J,I),J=1,N)
      IF ( .NOT.LENG  )  WRITE (IUNIT,1732)  (SHFTBP(J,I),J=1,N)
  740 IF ( .NOT.LDETNT)  GO TO 760
      WRITE (IUNIT,1740)
      IPAGE=IPAGE+1                                                   !INC LINE CNT.

      IF ( LDETV    )  WRITE (IUNIT,1720)  DETPT(I)
      IF (.NOT.LDETV)  WRITE (IUNIT,1722)  DETPT(I)
      IF ( .NOT.PARAB )  GO TO 750
      WRITE (IUNIT,1724)  DETBPM(I)
      GO TO 760
  750 IF ( LDETE    )  WRITE (IUNIT,1730)  DETRPM(I)
      IF (.NOT.LDETE)  WRITE (IUNIT,1732)  DETRPM(I)
C
  760 IPAGE=IPAGE+1                                                   !INC LINE CNT.
      IF(IPAGE.LE.10 .OR. I.EQ.NSL) GO TO 770                         !NO PAGE OR END OF LOOP?) YES.
      WRITE(IUNIT,1000) DATE,NXTPG                                    !PAGE.
      NXTPG=NXTPG+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,RCOM                                     !ROUTE HEADER.
      IPAGE=45                                                           !SET LINE LEFT CNT.

  803 IK=NDRTE+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,1802) (I,RDIST(I-NDRTE),RGRADE(I-NDRTE)                !PRINT ROUTE SECTION.
     1,RCOEF(I-NDRTE),RVWIND(I-NDRTE),I=IB,IE)
      IF(IE.EQ.NRDIST) GO TO 810                                         !(END OF RTE SECT?) YES.

      IK=IE+1                                                            !POINT TO NEXT RTE SEG.

      IPAGE=IPAGE-(IE-IB+1)                                              !DEC LINE LEFT CNT.

      IF(IPAGE.GT.0) GO TO 805                                           !(ANY LEFT?) YES.
      WRITE(IUNIT,1000) DATE,NXTPG                                       !NO, PAGE.
      NXTPG=NXTPG+1                                                      !INC PAGE #.

      IPAGE=55                                                           !SET LINE LEFT CNT.

      GO TO 805                                                          !GO PRINT.
C
  810 IF(LSTRTE) GO TO 820                                               !(LST RTE SECT?) YES,.

      IPRNT=208                                                          !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.8) GO TO 999                                           !(ERR?) NO, BYE
      WRITE(JCT,1640) HPART(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     PRINT TIRE DATA
C
  900   WRITE(IUNIT,1900) TNAME,TCOM,WRAD,FRC1,FRC2,TIREFF,AIW,FRC4
C
        GO TO 999
C
C*********************************************************************
C
C     PRINT TRANSMISSION DATA
C
 2000   WRITE(IUNIT,2500)TRNAM,TRCOM,(GEANUM(I),GEANAM(I)
     2,       I=1,NGTR)
C
        GO TO 999
C
C*********************************************************************
C
C     PRINT AXLE DATA
C
2100    WRITE(IUNIT,1330)AXNAME,AXCOM
        DO 2120 I=1,NAXS
        WRITE(IUNIT,1332)I,RAR(I),(ERAR(J,I),J=1,NRAX)
2120    CONTINUE
        IF(NRAX.EQ.2) GO TO 2340                          !(2 AXLES?)YES.

 2330   IF(NPAX(1,1).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(1,1).LE.1.AND.NPAX(2,1).LE.1) GO TO 2333  !(FOR 2 AXLES ANY SPIN LOSS DATA?)NO.
        WRITE(IUNIT,1340)                                 !YES, HEADER.

        IF(NPAX(1,1).LE.1) WRITE(IUNIT,1341)              !(DATA AXLE 1?)NO, REPORT.
        IF(NPAX(2,1).LE.1) WRITE(IUNIT,1342)              !(DATA AXLE 2?)NO, REPORT.
2350    DO 2380 IAX=1,NAXS
        IE=NPAX(1,IAX)                                    !ASSUME AXLE 1 DATA LONGER,SET END DATA PTR.

        IF(NPAX(2,IAX).GT.IE) IE=NPAX(2,IAX)              !(AXLE 2 DATA LONGER?)YES,RESET PTR.

C
        WRITE(IUNIT,2520)IAX
        DO 2360 I=1,IE                                    !LOOP THRU ALL AXLE DATA POINTS.

        L1=0                                              !ASSUME NO DATA AXLE 1 TO PRINT.

        IF(NPAX(1,IAX).LE.1.OR.NPAX(1,IAX).GT.IE) GO TO 2355  !(DATA TO PRINT?)NO.
        L1=1                                              !YES,FLG IT.

        WRITE(IUNIT,1420) AXRPM(I,1,IAX),AXTORQ(I,1,IAX)                  !PRINT AXLE 1.

2355    IF(NRAX.EQ.1.OR.NPAX(2,IAX).LE.1.OR.NPAX(2,IAX).GT.IE)GOTO 2360
C
        IF(L1.EQ.0) WRITE(IUNIT,1002)                     !(WAS AXLE 1 DATA PRINTED?)NO,"<CR><LF>"
```

```
      WRITE(IUNIT,1355) AXRPM(I,2,IAX),AXTORQ(I,2,IAX)                  !PRINT AXLE 2 DATA.
 2360 CONTINUE                                                          !NEXT.
2380  CONTINUE
C
C
C*********************************************************************
C
  999 IF(.NOT.DMPTTY)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 (1H1,1XA9,110X'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=,F9.3,9X,
     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(LP/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)
```

```
1330    FORMAT(/20X,'AXLE DATA  ( ',A10,' )'/20X,25('-')///2X,16A5//
       2        20X,'AXLE SPEED #',10X,'RAR',10X,'EFFICIENCIES'/)
1332    FORMAT(20X,I5,F20.2,8X,2F7.2)
 1300   FORMAT (/20X'VEHICLE DATA  ( 'A10' )'/20X,28(1H-)///2X,16A5//
       1        /2X,6HWEIGHT,9X,1H=,F8.1,3X,3HLBS,
       2        //2X,12HFRONTAL AREA,3X,1H=,F9.2,2X,5HSQ FT,
       3        10X,'NUMBER OF TIRES =',I3,//2X,
       4        16HDRAG COEFFICIENT,2X,1H=,F11.6,2X,
       5        '  CD SENSITIVITY COEFF  ='F11.6//
       6        2X,20HPROPSHAFT INERTIA  =,F6.3,
       7        2X,12HFT-LB-SEC**2//,2X,'BEVEL GEAR = ',F6.3,
       8        '; BEVEL GEAR EFF. = ',F6.3,//)
 1340   FORMAT(48X'AXLE SPIN LOSS DATA'/48X19('-')//35X'AXLE 1'
       1,34X'AXLE 2'/35X6('-')34X6('-')///7X2(20X'SPEED'10X'TORQUE')
       2,/7X2(20X'(RPM)'10X'(LB-FT)')/7X2(20X6('-')10X7('-'))//)
 1341   FORMAT(25X'NO SPIN LOSS DATA SPECIFIED')
 1342   FORMAT(65X'NO SPIN LOSS DATA SPECIFIED')
 1355   FORMAT('+'59XF10.1F16.3)
 1400   FORMAT (/20X,13HGEAR DATA  ( ,A10,2H )/20X,25(1H-)///2X,16A5//
       1        /2X,13HGEAR RATIO  =,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   FORMAT(25XA4' SPIN LOSS DATA'/25X19('-')///
       1,27X'SPEED'10X'TORQUE'/27X'(RPM)'10X'(LB-FT)'/27X6('-')
       1,10X7('-')//)
 1420   FORMAT(22XF10.1F16.3)
 1410   FORMAT(/19X'NO 'A4' SPIN LOSS 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       10X,'SPEED RATIO  =',F6.3///
       2       10X,'DUTY CYCLE   =',F7.3///
       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   FORMAT (/20X,20HDRIVING SCHEDULE  ( ,A10,2H )/20X,32(1H-)///2X,
       1        16A5///2X,18HINITIAL CONDITIONS,10X,4HTIME,11X,1H=,F8.2,
       2        2X,3HSEC/30X,8HDISTANCE,7X,1H=,F8.2,2X,2HMI/30X,
       3        16HVEHICLE SPEED  =,F8.2,2X,3HMPH/30X,12HACCELERATION,
       4        3X,1H=,F8.2,2X,9HFT/SEC**2/30X,16HSTARTING GEAR  =,I5,
       5        /30X,'REAR AXLE      =',I5,
       6        ///2X,7HSEGMENT,11X,
       7        19HDESIRED PERFORMANCE/2X,7(1H-),11X,19(1H-))
 1602   FORMAT (/13X,12(1H*),14H SEGMENT TYPE ,12(1H*),18X,15(1H*),
       1        13H SEGMENT ENDPOINT ,15(1H*)/33X,16HCONSTANT  ACCEL TO,
       2        8X,8HTHROTTLE/2X,7HSEGMENT,4X,2(8HRCONSTANT,2X),7HPERCENT,
       3        3X,23HAND HOLD  HOLD  RATE OF,3X,2(8HRELATIVE,2X),
       4'PASSING   ABSOLUTE  ABSOLUTE  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   FORMAT (1XI6,14XF10.2)
 1623   FORMAT (1XI6,24XF10.2)
 1624   FORMAT (1XI6,34XF10.2)
 1630   FORMAT ('+'51XI5)
 1631   FORMAT ('+'56XF10.2)
 1632   FORMAT ('+'66XF10.2,41XI6)
 1633   FORMAT ('+'76XF10.2,31XI6)
 1634   FORMAT ('+'86XF10.2,21XI6)
 1635   FORMAT ('+'96XF10.2,11XI6)
 1636   FORMAT ('+'106XF10.2,1XI6)
 1640   FORMAT(/' ? PRNTPD- RELOAD ERR ON 'A5,3XA10/)
```

```
 1700   FORMAT (20X,15HSHIFT LOGIC  ( ,A10,2H )/20X,27(1H-)///2X,16A5//
        1        /2X,21HTHIS TRANSMISSION HAS,I4,7H  GEARS)
 1708   FORMAT (//2X,13HSHIFT LINE   ,I3,2H -,I3,5X
        2        ,'AXLE SPEED',5X,I3,' -',I3)
 1710   FORMAT (//2X,10HUP SHIFT  ,I3,2H -,I3,5X
        2        ,'AXLE SPEED',5X,I3,' -',I3)
 1712   FORMAT (//2X,10HDOWN SHIFT,I3,2H -,I3,5X
        2        ,'AXLE SPEED',5X,I3,' -',I3)
 1714   FORMAT (1H+,55X,12HSHIFT TIME =,F6.3,4H SEC)
 1720   FORMAT (22X,21HVACUUM (IN-HG)        ,10F8.2)
 1721   FORMAT (22X,21HTHROTTLE (DEGREES)    ,10F8.2)
 1722   FORMAT (22X,21HTHROTTLE (PCT WOT)    ,10F8.2)
 1724   FORMAT (22X,21HVEHICLE SPEED (MPH)   ,10F8.2)
 1730   FORMAT (22X,21HENGINE SPEED (RPM)    ,10F8.2)
 1732   FORMAT (22X,21HPROPSHAFT SPEED (RPM),10F8.2)
 1740   FORMAT (/22X,26HDETENT OVERIDE DESCRIPTION/)
 1800   FORMAT (/20X,23HROUTE SPECIFICATION  ( ,A10,2H )/20X,35(1H-)/
        1,//10X,16A5/
        1/23X'DISTANCE'8X'PERCENT'9X'ROAD'8X'WIND SPEED'/
        1,10X'POINT'8X'(MILES)'10X'GRADE'7X'COEFFICIENT'7X'(MPH)'/
        2,10X5(1H-)8X7(1H-)10X5(1H-)7X11(1H-)4X10('-')/)
 1802   FORMAT(10X14,1X4F15.3)
 1900   FORMAT(/20X'TIRE DATA  ( 'A10' )'/20X25('-')//2X16A5//
        1,'  ROLLING RADIUS  ='F9.3'   FT'12X'C1 ='F11.6//
        2,'  C2 ='F11.6//
        3,'  TIRE EFFICIENCY ='4XF6.3///
        4,'  WHEEL INERTIA   ='4XF6.3'  FT-LB-SEC**2',12X,'C4  =',F11.6)
 2500   FORMAT(/20X,'TRANSMISSION DATA  ( ',A10,' )',/
        2,20X,34('-')///2X,16A5//
        3,20(30X,I2,' - ',A10/))
 2520   FORMAT(20X,'[AXLE SPEED',I3,']')
 C
        END
```

```
*               **** RCRCNT ****
*
        FUNCTION RCRCNT(STRING,LWRDS)
C
C       ENTRY POINTS:   RCRCNT
C
C       SUBROUTINES CALLED:     NWRCNT
C
C       CALLED BY:      ASCIZ,  ICRCNT, INPBAT
C
C*****************************************************************
C
C
        DIMENSION STRING(LWRDS),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,LWRDS)                 !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
```

```
*               **** READPD ****
*
      SUBROUTINE READPD
     1       ( NARRAY,LENGTH,NPCARD,HWORD,NPOINT,IFLAG,A1,A2,A3,A4 )
C
C     ENTRY POINTS:   NXTCRD, READPD
C
C     SUBROUTINES CALLED:     NWRCNT
C
C     CALLED BY:      INPBAT
C
C*****************************************************************
C
C
      LOGICAL  ENDE,LEFIL
C
      DIMENSION  A1(1),A2(1),A3(1),A4(1),ALPHA(11)
C
      COMMON /ENDOF/   ENDE,IPRNT,LEFIL
C
      DATA HBLANK/' '/,HSTAR /'*'/
C
C    NARRAY = NUMBER OF ARRAYS TO BE FILLED
C    LENGTH = LENGTH OF ARRAYS TO BE FILLED
C    NPCARD = NUMBER OF DATA POINTS TO BE READ OFF EACH CARD
C    HWORD  = ALPHA WORD WHEN FOUND ON DATA CARD STOPS INPUT
C    NPOINT = NUMBER OF DATA POINTS ACTUALLY READ INTO EACH ARRAY
C    IFLAG  = RETURN STATUS FLAG WHERE 1-NO 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                !(OLD CALL JUST TO LOOK AT NEXT CARD?)YES, SWITCH.
      GO TO 90                                    !NO,GO READ IN DATA.
C
      ENTRY NXTCRD(HWORD,IFLAG,WORD)              !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,NPCARD
      IE     = IB + NPCARD - 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                                                    !POSI 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) GO TO 97                                           !(CARD DATA FIELD BLANK?) YES, GO FLG.

      NPOINT = IB + J - 1                                           !CALC # OF DATA PTS./REC
C                                                  !SAME # OF PTS./REC ASSUMED
      GO TO (98,99),NPFIG                                           ! (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 )  GO TO 120                                 !(READ RECORD?) NO.
      READ  (4,1000)   (A2(I),I=IB,IE)                             ! YES.
      IF(NPFLG.EQ.2) GO TO 95                                       !(FIRST DATA REC BLANK?) YES, GO SCAN ON 2ND REC.

 99   IF ( NARRAY.LT.3 )  GO TO 120                                 !(READ RECORD?) NO.
      READ  (4,1000)   (A3(I),I=IB,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
```

```
*                **** REMAP ****
*
      SUBROUTINE REMAP
C
C     ENTRY POINTS:    REMAP
C
C     SUBROUTINES CALLED:      ENGINE, PRNOUT
C
C     CALLED BY:       INPBAT
C
C*****************************************************************
C
C
      DOUBLE PRECISION   ENAME,UN
C
      LOGICAL SRPM,SPS,STOR,SBMEP,SHP,SLBHR,SBSFC,SGALHR,LWOT
     1,LRPM,PRPM,LPS,PPS,LTOR,PTOR,LBMEP,PBMEP,LHP,PHP,LLBHR
     2,PLBHR,LBSFC,PBSFC,LGALHR,PGALHR,LPRNT
      COMMON /TMAP/ TORQ,LWOT,TWOT,THIN
      COMMON /GET/ UT,UN,NUG(20),JENG(20),IENG
      COMMON /ENDOF/ ENDE,IPRNT
      COMMON /ENGMAP/ RPMAX(2),RPMIN(2),NRPM(2),RPME,TORQE,FRATE,VAC,
     1THR,MAPOK,
     1IERRE,NTOR(2,20),EMEP(20,20,8),ERPM(2,20),ENMIN(2),SPIDLE(2)
      COMMON /LDIMEN/ LRPM,PRPM,LPS,PPS,LTOR,PTOR,LBMEP,PBMEP,LHP,PHP,
     1LLBHR,PLBHR,LBSFC,PBSFC,LGALHR,PGALHR,LPRNT
      COMMON /IO/ECOM(16),ENAME(2),DISP,ICYL,IMIN,IMAX,THRMAX,THRMIN
     1,EINER,BORE,STROKE,FSPGR,NCYCLE
      COMMON /OLDMAP/ 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 PENOUT
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=NRPMO
      NRPMO=NRPM(IENG)
      NRPM(IENG)=NDUM
      DO 40 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 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
```

```
*               **** SCALEN ****
*
        SUBROUTINE SCALEN
C
C       ENTRY POINTS:   SCALEN
C
C       SUBROUTINES CALLED:     PRNOUT
C
C       CALLED BY:      INPBAT
C
C*****************************************************************
C
C
        LOGICAL           LSCALE,LIMPRN,LTRRZ
C
        DOUBLE PRECISION   ENAME,UN,GNAME
C
        COMMON /PRNLIM/  LIMPRN,MILIM,SECLIM,ENDLIM,ALIMN,ISCALE
        COMMON /GEI/     UT,UN,NUG(20),JENG(20),IENG
        COMMON /ENGMAP/  RPMAX(2),RPMIN(2),NRPM(2),RPME,TORQE,FRATE,VAC,
        1                 THR,MAPOK,IERRE,NTOR(2,20),EMAP(20,20,8),
        2                 ERPM(2,20),ENMIN(2),SPIDLE(2)
        COMMON /IO/ ECOM(16),ENAME(2),DISP,ICYL,IMIN,IMAX,THRMAX,
        1                 THRMIN,BINER,BORE,STROKE,FSPGR,NCYCLE
        COMMON /CONST/   PRC1,PRC2,FAC,CD,AREA,VWIND,WGT,PGC,WRAD,RAR(3)
        2,  GRAT(20),NUMG,NGEAR,AIW,AIP,AI2,ERAT(20),ERAR(2,3)
        3,  AIE,AIA,AI1,BPER,CDC,PHI,PSI,AIGIN(20),AIGOUT(20),WLSG,LTRRZ
        4,NGRLSS(20),GRPM(20,20),GRTORQ(20,20),GNAME(20),GCOM(16),PRC4
        COMMON /OLDIO/   ODISP,OBORE,OSTROK,IOCYL
        COMMON /ENDOF/   ENDE,IPRNT
C
C
C*****************************************************************
C
C     DEFINE SCALING RATIOS
C
        DRATIO = DISP/ODISP
        SRATIO = OSTROK/STROKE
C
C     DETERMINE NUMBER OF ENGINE MAPS TO RESCALE
C
        NUME   = 1
        DO 100  I = 1,NUMG
        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
C
  200   EMAP(I,J,2*KENG) = DRATIO * SRATIO * EMAP(I,J,2*KENG)
C
C     REDEFINE ENGINE MAP BOUNDARIES
C
        RPMAX(NENG)  = ERPM(NENG,NRPMP)
  400   CONTINUE
C
C     PRINT RESCALED ENGINE MAP
C
        IF ( LIMPRN )  GO TO 900
        WRITE (6,1000)  NUME,ODISP,OBORE,OSTROK,IOCYL,
        1                       DISP, BORE,STROKE,ICYL
        IPRNT  = 1
        DO 600  IENG = 1,NUME
  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
```

```
*               **** SHIFTS ****
*
      SUBROUTINE SHIFTS
*
*  MODIFIED 10 JUNE 1980: SOMERS
*
C     ENTRY POINTS:   BKSHFT, SHIFTS, STSHFT
C
C     SUBROUTINES CALLED:     DEBUG,RESETM,BLOCKT
C
C     CALLED BY:      SIMCTR, SIMINT
C
C     EDIT HISTORY
C
C     [724]/SS-2-1-79         REWRITTEN ENTIRELY TO TAKE INTO ACCOUNT
C                             THE MULTI-SPEED AXLE SHIFT LOGIC.
C
C*****************************************************************
C
      DIMENSION IGRS(2),TEMP(47)
C
      DOUBLE PRECISION   CNAME,SNAME,UN,GNAME
C
      LOGICAL LLSH,PARAB,LVAC,LENG,GDAT,LDETV,LDETE
     1,LDETNT,LTBRZ,LOCKUP,LMPH,COAST,LWOT,PRINT6,IUP,LIMTTY,LTHR
C
      COMMON /DSHIFT/ LMPH,AVEL
      COMMON /V2MISC/  LOCKUP(20)
      COMMON /CDEBUG/ IDEBUG
      COMMON /GET/ UT,UN,NOG(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,IGP(60),
     2      IGT(60),SHFTIM(60),SHFTPT(10,60),SHFTRP(10,60),LVAC,LENG,
     3       GDAT,NSPTS(60),LDETNT,DETPT(60),DETRPM(60),PARAB,LDETE,
     4       LDETV,GOVIIN,IAP(60),IAT(60),NUMBSI,RERAT(47,60),
     5       IRER,IRERO,NRER
      COMMON /TORCON/ TORBPK,TORQ2,RPM2,COAST,SR,TR,TRD(20),SRD(20),
     1AKD(20),NID,SRC(20),ARC(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,
     1       IHR,MAPOK,IERRE,NTOR(2,20),EMAP(20,20,8),ERPM(2,20),
     2       ENMIN(2),SPIDLE(2),TORQEO,LTHR
      COMMON /TMAP/ TORQ,LWOT,TWOT,TMIN
      COMMON /CONST/   FRC1,FRC2,FAC,CD,AREA,VWIND,WGT,FGC,WRAD,BAR(3)
     2,  GRAT(20),NUMG,NGEAR,AIW,AIP,AI2,ERAT(20),ERAR(2,3)
     3,  AIE,AIA,AI1,BPER,CDC,PHI,PSI,AIGIN(20),AIGOUT(20),WLSG,LTRRZ
     4,NGRLSS(20),GRPM(20,20),GRTORQ(20,20),GNAME(20),GCOM(16),FRC4
      COMMON /CNTRL/ IC,TOLD,VOLD,T,V,ACCEL,D,DT
      COMMON /SEGNO/ITS
      COMMON /MOCON/   BB1,BB2,RPMWO,RPMEO,CPHI,VWINDC,VWINDS,AA1,AA2,
     2                 AA3,AA4,AA5,AA6,AA7,AA8,AA9,AA10,RARSQ,AA11
      COMMON /TTYOUT/ LIMTTY,ITTYWD,LIPT,JCT
      COMMON /VEHICL/ DUM(52),NRAX,DUM1(274),NAXS,NAX,NAXO
C
      DATA HDEFOR/'BEFOR'/
      DATA NAME/'SHIFT'/
C
C     TURN SHIFT FLAG OFF AND DETERMINE WHAT GEARS TO UP/DOWN SHIFT TO
C     IF REQUIRED
```

```
      LLSH   = .FALSE.                                          !ASSUME NO SHIFT.

      PRINT6 = .FALSE.
C
C     SET CURRENT VALUES OF SHIFT PARAMETERS TO BE MONITORED
C
      X = 100.0*(TORQE - TMIN)/(TWOT - TMIN)
      IF(LVAC) X = VAC
      IF(LTHR) X = THR
      Y=RPMP
      IF(LENG) Y=RPME
      IF(PARAB) Y=V
      LDETNT=.FALSE.  !TEMP NO DETENT CODE UNTIL FIX FOR MULTI SPEED AXLES.
      IF(.NOT.LDETNT.OR.ITS.NE.3) GO TO 60
C
C     IF IN CONSTANT PERCENT WOT SEGMENT AND THROTTLE IS WIDE OPEN USE
C     DETENT  OVERRIDE SHIFT CRITERIA
C
      PCTHR  = X
      IF(LVAC) PCTHR=(TORQE-TMIN)/(TWOT-TMIN)*100.
      IF(PCTHR.LT.99.) GO TO 60
      IF(NGEAR.EQ.NUMG) GO TO 40
      IGR=IGRS(1)
      IF(Y.GE.DETRPM(IGR)) GO TO 140
      IF(NGEAR.EQ.1) RETURN
40    IGR=IGRS(2)
      IF(Y.LE.DETRPM(IGR)) GO TO 120
      RETURN
C
C
C...  TRY UPSHIFT
C
60    LUP.= .TRUE.
      IF(IRER.EQ.NRER .OR.                                      !AT TOP GEAR
     1 Y.LT.RERAT(16,IRER)) GO TO 240                           !BELOW LOWEST SPEED CURVE VALUE.

      NUP=RERAT(5,IRER)                                         !GET # OF PTS IN CURVE.

      IF(Y.GE.RERAT(16-1+NUP,IRER)) GO TO 140                   !ABOVE HIEST SPEED CRV VAL
C
C...  WITHIN SPEED CURVE.  GET SURROUNDING PTS.
C
      DO 100 I=2,NUP
      IF(Y.GT.RERAT(16-1+I,IRER)) GO TO 100                     !HAVN'T GOT IT YET
      ID     =  6-1+I                                           !GOT IT. GET UPPER ID IND.

      LDM1   = LD-1                                             !GET LOWER ID IND.

      ISP    = 16-1+I                                           !GET UPPER SPD IND.

      ISPM1  = ISP-1                                            !GET LOWER SPD IND.

      TLD    = RERAT(ID,IRER)                                   !GET UPPER ID VAL.

      TLDM1  = RERAT(IDM1,IRER)                                 !GET LOWER ID VAL.
      TSP    = RERAT(ISP,IRER)                                  !GET UPPER SPD VAL.

      TSPM1  = RERAT(ISPM1,IRER)                                !GET LOWER SPD VAL.
      GO     = (TLDM1-TLD)*(Y-TSP)/(TSPM1-TSP)+TID

      CALL TRPEWR(NAME,1)
```

```
*
      IF(PRINT6)WRITE(6,1000) T,V,X,GO
 1000 FORMAT(' SHIFTS - 1',F8.3,3(2X,F8.3))
      IF(LVAC)GO TO 80
      IF(X.LE.GO)GO TO 140
      GO TO 240
80    IF(X.GE.GO)GO TO 140
      GO TO 240
  100 CONTINUE
      GO TO 140
*
120   IF(ITS.EQ.2 .AND. BPER.LT.0. .AND. LOCKUP(NGEAR))RETURN
140   IF(ITS.NE.1 .OR.                                          !(CONST ACCEL?)NO.
      1 .NOT.LMPH .OR.                                          !(END DRS SEG SPEC BY VEL?)NO.

      2 ABS(ACCEL).LT.1.E-3 .OR.                                !(ZERO ACCEL?)YES.
      3 (AVEL-VOLD)*1.466667/ACCEL.GT.RERAT(4,IRER).OR.
      4 LOCKUP(NGEAR))GO TO 160                                 !(WILL WE GO PAST END DRS SEG DURING SHIFT?)

      RETURN                                                    !YES, IF WE GET TO HERE DON'T SHIFT. BYE.

C
C...  DO SHIFT
C
160   IF(IDEBUG.EQ.2) CALL DEBUG(HBEFOR)                        !DEBUG SHIFTS)YES.
      IRERO=IRER
      IRER=IRER+1                                               !ASSUME UPSHIFT

      IF(.NOT. LUP)IRER=IRER-2                                  !DOWN SHIFT.
      NGEAR=RERAT(2,IRER)                                       !GET NEW GEAR #.

      IENG=JENG(NGEAR)                                          !GET NEW ENG MAP #.

      STIME=RERAT(4,IRERO)                                      !GET SHIFT TIME.
      LLSH=.TRUE.                                               !FLAG THAT WE SHIFTED.

      NAXO=NAX
      NAX=RERAT(3,IRER)
      IF(NAX.GT.NAXS)GO TO 200
      IF(NAXS.EQ.1 .OR. NAX.EQ.NAXO)RETURN
180   RARSQ=RAR(NAX)**2
      AA8=RAR(NAX)*ERAR(1,NAX)
      IF(NRAX .EQ. 2) AA8 = 1./(0.5/AA8 +0.5/(RAR(NAX)*ERAR(2,NAX)))
      RETURN
C
200   WRITE(5,220)
      CALL RESETH
C
220   FORMAT(/' ?SHIFTS - Incompatible rear axle and shift logic',/,
      2 ' Trying to shift into non-existent multi-speed axle')
      RETURN                                                    !SO LONG.

C
C...  TRY DOWN SHIFT
C
240   LUP=.FALSE.
      IF(IRER.EQ.1)RETURN                                       !AT LOWEST GEAR.

      IF(Y.LT.RERAT(38,IRER))GO TO 120                          !BELOW LOWEST SP CRV VAL.
      NDN=RERAT(27,IRER)                                        !GET # OF DN SHIFT PTS.
```

```
      DO 280 I=2,NDN
      IF(Y.GT.RERAT(38-1+I,IRER))GO TO 280                    !HAVEN'T GOT IT YET
      LD    = 28-1+I                                          !GOT IT. GET UPPER LD IND.

      LDM1  = LD-1                                            !GET LOWER LD IND.

      ISP   = 38-1+I                                          !GET UPPER SPD IND.

      ISPM1 = ISP-1                                           !GET LOWER SPD IND.

      TLD   = RERAT(LD,IRER)                                  !GET UPPER LD VAL.

      TLDM1 = RERAT(LDM1,IRER)                                !GET LOWER LD VAL.
      TSP   = RERAT(ISP,IRER)                                 !GET UPPER SPD VAL.

      TSPM1 = RERAT(ISPM1,IRER)                               !GET LOWER SPD VAL.
      GO    = (TLDM1-TLD)*(Y-TSP)/(TSPM1-TSP)+TLD
*
      CALL TAPEWR(NAME,2)
*
      IF(PRINT6)WRITE(6,1010) T,V,X,GO
 1010 FORMAT(' SHIFTS - 2',F8.3,3(2X,F8.3))
      IF(.NOT.LVAC)GO TO 260
      IF(X.LE.GO)GO TO 120
      RETURN
  260 IF(X.GE.GO)GO TO 120
      RETURN
  280 CONTINUE
      RETURN
C
C...  ENTRY FOR BACK SHIFT ... REVERSE THE LAST SHIFT
C
      ENTRY BKSHFT(NGOLD)
C
      IF(IDEBUG.NE.2)GO TO 300
      IF(LIMTTY)WRITE(5,320)
      IF(LLPT)WRITE(6,320)
  300 IRER=IRERO
      NGEAR=RERAT(2,IRER)
      LLSH=.FALSE.
      NAX=NAXO
      GO TO 180
C
  320 FORMAT(' [Back shifting]')
C
      ENTRY   STSHFT
C
C...  SET UP RERAT ARRAY
C
C     RERAT(47,60)
C     LOC             WHAT'S IN IT
C     ---             ------------
C     1       -       COMPUTED REAR END RATIO = GEAR RATIO*AXLE RATIO
C     2       -       GEAR #
C     3       -       AXLE #
C     4       -       UPSHIFT TIME.
C     5       -       # OF UPSHIFT CURVE PTS.
C     6-15    -       UPSHIFT LOAD CURVE
C     16-25   -       UPSHIFT SPEED CURVE
C     26      -       DOWNSHIFT TIME.
C     27      -       # OF DOWNSHIFT CURVE PTS.
```

```
C       28-37   -       DOWNSHIFT LOAD CURVE
C       38-47   -       DOWNSHIFT SPEED CURVE
C
        NDONE = 0
C
        DO 460 I=1,NUMBSL
        IIGF=IGF(I)
        IIAF=IAF(I)
        IIGT=IGT(I)
        IIAT=IAT(I)
        RF=GRAT(IIGF)*RAR(IIAF)
        RT=GRAT(IIGT)*RAR(IIAT)
        IF(NDONE.EQ.0)GO TO 360
*
        DO 340 IR=1,NDONE
        IF(RF.EQ.RERAT(1,IR))GO TO 380
340     CONTINUE
*
360     NDONE=NDONE+1
        RERAT(1,NDONE)=RF
        RERAT(2,NDONE)=IIGF
        RERAT(3,NDONE)=IIAF
        IR=NDONE
*
380     IF(RF.LT.RT)GO TO 400                                   !DOWNSHIFT
C
C...    UPSHIFT INDICES
C
        ITIME  =  4
        IPTS   =  5
        ILOAD  =  6
        ISPEED = 16
        GO TO 420
C
C...    DOWNSHIFT INDICES
C
400     ITIME  = 26
        IPTS   = 27
        ILOAD  = 28
        ISPEED = 38
C
C..     FILL IN THE VALUES
C
420     RERAT(ITIME,IR)=SHFTIM(I)
        RERAT(IPTS,IR)=NSPTS(I)
*
        DO 440  IT=1,NSPTS(I)
        RERAT(IT-1+ILOAD,IR)=SHFTPT(IT,I)
        RERAT(IT-1+ISPEED,IR)=SHFTRP(IT,I)
440     CONTINUE
460     CONTINUE
        NRER=NDONE
C
C...    SORT
C
        NP=NDONE-1
        DO 500 I=1,NP
        NDONE=NDONE-1
        DO 480 J=1,NDONE
        IF(RERAT(1,J+1).LT.RERAT(1,J))GO TO 480
C
```

```
C...    SWITCH
C
        CALL BLOCKT(RERAT(1,J),TEMP,47)
        CALL BLOCKT(RERAT(1,J+1),RERAT(1,J),47)
        CALL BLOCKT(TEMP,RERAT(1,J+1),47)
480     CONTINUE
500     CONTINUE
        IRER=1
        IRERO=IRER
        RETURN
        END
```

```
*       **** SIMCTR ****
*
        SUBROUTINE SIMCTR ( IECOND , SIMODT )
*
*  MODIFIED 28 AUGUST 1980: SOMERS
*
C       ENTRY POINTS:   SIMCTR
C
C       SUBROUTINES CALLED:     CTRLD,  DEBUG,  DSK,    DSKCTR, EXIT,
C                       GOBACK, ITERAT, KPTIME, RESETM, SHIFTS, SIMINT,
C                       SIMLPT, SIMSTS
C
C       CALLED BY:      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 CONS %WOT SECTION WITH CALL TO ITERAT
C       [606]/SS-3-28-78        IF CONST ACCEL SEG TO AB VEL, AND VEL HIGHER THAN
C                               DESIRED V, SWITCH TO CONST VEL INORDER TO
C                               DECCEL TO REQ VEL THEN ENDSEG.
C       [607]/SS-4-10-78        CLUTCH
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-7-17-78        IF IDLEING ON A GRADE, PUT ON THE BRAKES
C       [725]/SS-7-23-79        TAKE OUT SOME LEFT OVER CODE THAT CHECKS
C                               SHIFT TIME AGAINST ACCUMULATED SHIFT TIME.
C
C*****************************************************************
C
        INCLUDE 'COMMS/NOLIST'
        COMMON/JDSFIX/QUMD
C
        LOGICAL LSEC,LMILE,LPASS,LMP,ITHRR,ENDSEG
        1,ARRIVE,PRINT6
C
        DATA HAFTER/'AFTER'/
        DATA NAME/'SIMCT'/
C
C*************************************************************************
C
        PCTWOT(X)=100.*(TORQE-TMIN)/(TWOT-TMIN)
        PCTHR = PCTWOT(X)
*
*       PRINT5 = .TRUE.
        PRINT6 = .FALSE.
*
        SIMODE=SIMODT
        CALL SIMINT                                     ! BRING VEHICLE UPTO INITIAL CONDITIONS
C
        SDELAY=.8
        IF(BPER.GT.0.)SDELAY=SDELAY*BPER
        BPERO=BPER                                      !SAVE ROAD GRADE.
*
        IECOND=0                                        !ASSUME NO ERR.
```

```
      KEND=0                                                    !SET PRINT LINE FLG.

      GO TO 6
C
    5 IF(ISEG+NDSEG.IE.NSEG) GO TO 6                            !(END DRS SECT?) NO.
      IF(LSTSEC) 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 ALI 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
      NSESEG(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)
      APWO=PWOT(ISEG)
      APWOO=APWO*.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
      ISTRUP=.FALSE.                                            ![613]
      IF(V.LT.1.E-5)LSTRUP=.TRUE.                               ![613]
      ASTART=ACCEL
      RT=0.
      RD=0.
      LSEC=.FALSE.
      LMILE=.FALSE.
      IPASS=.FALSE.
      LMP=.FALSE.
      IMPH=.FALSE.
      LTHRB = .FALSE.
      LSHIFT=.FALSE.
      ACCO=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(AVEI.GT.-.5) LMPH=.TRUE.
        IF(ABS(ATHR).GT.1.E-20) LTHRR = .TRUE.
        IF(LMILE) ADSE=ADSE*5280.
        IF ( LMP )  APOS = APOS * 5280.
        TSAVE=0.
        DSAVE=0.
        IF(NNGS.EQ.0) GO 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.

        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]
*
        CALL TAPEWR(NAME,1)
*
        GO TO (40,20,60,80),ITS                        !(ACCEL/VEL/%WOT/ACCEL TO CON ACCEL & HOLD %WOT) (SEG TYPE L
OGIC?).
C
C     CONSTANT VELOCITY SEGMENT
C
   20   DT=DEPDT
        IF(.NOT.LOCKUP(NGEAR)) DT=.25
        TOLD=T
        T=T+DT
        VOLD=V
        LWOT=.FALSE.
        ACCEL=0.
        IF(ISTRUP.AND.AVSE.NE.0.)CALL SIMLPT(12)       ![613]ERROR
        IF(LSTRUP.AND.AVSE.EQ.0.)LSTRUP=.FALSE.        ![613]IF IDLE STEP, TURN OFF START FLAG
*
        CALL TAPEWR(NAME,2)
*
        CALL GOBACK
*
        CALL TAPEWR(NAME,3)
*
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(DPER.GT.0.) GO TO 206                       !(UPHILL?)YES.
```

```
      IF(NGCNT.LE.2) GO TO 204                                  !(HERE FOR 3RD GEAR OR GREATER TIME THRU DRS SEG?)NO.
  206 IF(NGEAR.EQ.NGOLD) GO TO 201                              !YES,(VEH SETTLE INTO COAST VEL?)YES.
C
C     ALLOW CAR TO REACH CONSTANT VELOCITY REQUIRED BY SEGMENT IF NOT
C     ALREADY AT THAT VELOCITY AFTER LAST SEGMENT
C
  204 CONTINUE
*
      CALL TAPEWR(NAME,4)
*
      IF(ABS(RPME-RPMEO).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
      LWOT=.FALSE.
      ENDSEG=.FALSE.                                            !FIG NOT END DRS SEG.
C
C     ACCELERATE TO DESIRED VELOCITY
C
      ACCEL=(AVSE-V)*1.46667/DT                                 !COMPUTE DESIRED ACCELERATION.
*
      CALL TAPEWR(NAME,5)
*
      CALL GOBACK
*
      CALL TAPEWR(NAME,6)
*
      IF(MAPOK.LT.4)GO TO 15                                    !(ON THE MAP)YES, DONE, RETURN.

      LOWMAP=(MAPOK.EQ.4.OR.MAPOK.EQ.6.OR.MAPOK.EQ.7)
      IF(.NOT.LOWMAP)GO TO 15
      AGRAT=GRAT(NGEAR)
      AEFFG=ERAT(NGEAR)
      ABR=TORQW
      LBRAKE=.TRUE.
      XYZ = TR*BVG*BVGEFF
      TORQW=TORQW-(TORQE-TMIN)*AGRAT*AEFFG*AA8*XYZ
      ABR=TORQW-ABR
      TORQP=TORQW/AA8
      TORQ2=TORQP/(AGRAT*AEFFG)
      TORQ1=TORQ2/TR
      TORQE=TORQA+TORQF+TORQ1/BVG/BVGEFF
*
      CALL TAPEWR(NAME,7)
*
      GO TO 15
   16 CONTINUE
      RPME=RPMEO
      LWOT=.TRUE.
*
      CALL TAPEWR(NAME,8)
*
      CALL ENGINE
*
```

```
      CALL TAPEWR(NAME,9)
*
      IWOT=.FALSE.
      TRR=0.                                                    !ZERO FRONT END ROTATING INERTIA.

      TORQF=0.
      TITER=TWOT                                                !ASSUME TARGET TORQE TO BE MAX.

      IF(MAPOK.GT.3 .AND. LOWMAP)TITER=TMIN                                  !(POINT TO MIN)YES, SET TO MIN.

*
      CALL TAPEWR(NAME,10)
*
      CALL ITERAT(TITER)
C
   15 TSAVE=TSAVE+DT
   17 PCTHR=PCTWOT(X)
      DSAVE=DSAVE+DT*V*1.46667
      GO TO 45                  !IF NOT SETTLED, BETTER MAKE ENDSEG CHECKS
  201 IF(ABS(RPME-RPMEO).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
      IF(ABS(BPER) > 0.01) GO TO 30
   21 TEND=1.E20                                                !YES.
      DD=COMD*5280.
      IF(.NOT.LSEC) GO TO 22
C
C     IF RELATIVE TIME END POINT GIVEN
C
      TEND=ATSE-TSAVE
   22 IF(.NOT.LMILE) 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-V0).LT..00001) GO TO 26
      TEST=(APCS-COMD*5280.+DSTART+T*V0*1.466667)/((V-V0)*1.466667)
      IF(TEST.LT.TEND) TEND=TEST
   26 IF(.NOT.LMP) 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                                   !I 60 I
      ENDSEG=.TRUE.
      GO TO 99
  280 TOLD=T
      IF(TEND.GE.1.E-10) GO TO 281                              !OVERSHOOT?)NO.
C
      CALL SIMLPR(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 TAPEWR(NAME,11)
*
      CALL GOBACK                                               !ZAP - WERE THERE.
*
      CALL TAPEWR(NAME,12)
*
      PCTHR=PCTWOT(X)                                 ! % THROTTLE.
      GO TO 99                                                  !GO UPDATE ACCUMULATORS.
C
   30 ITS=1                                                     !WE'RE ON A HILL AND REACHED CONST VEL SW TO CONST ACCEL 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(LMPH.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(BPER).LE. 1.-3)ITS=2
      TOLD=T
      DT=DEFDT
      T=T+DT
      ACCEL=AASE
      VOLD=V
      PCOLD=PCTHR
*
      CALL TAPEWR(NAME,13)
*
      IF(LSTRUP)GO TO 360                                       ![613]
C
```

```
C     TRY REQUIRED ACCELERATION
C
*
        CALL TAPEWR(NAME,14)
*
        CALL GOBACK
        PCTHR=PCTWOT(X)
*
        CALL TAPEWR(NAME,15)
*
       .IF(MAPOK.GT.3) 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   CONTINUE
*
        CALL TAPEWR(NAME,16)
*
        PITER=RTHR*DT+PCOLD
        IF(100..GT.PITER-.01) GO TO 434
        CALL ITERAT(TWOT)
        GO TO 44
  434   CONTINUE
*
        CALL TAPEWR(NAME,17)
*
        TITER=TMIN+PITER*(TWOT-TMIN)*0.01
        IF(LOCKUP(NGEAR)) GO TO 435
        ACCEL=ACCO+0.01
        CALL GOBACK
  435   CONTINUE
*
        CALL TAPEWR(NAME,18)
*
        CALL ITERAT(TITER)
C
C     BEGIN CHECKS TO SEE IF AT THE END OF A SEGMENT
C
   44   PCTHR=PCTWOT(X)                          ! % THROTTLE.
   45   DD=CUMD*5280.
*
        CALL TAPEWR(NAME,19)
*
        IF(ACCEL.GT.0.) GO TO 451                                  !(+ ACCEL?) YES.
        IF(BASE.LE.0.) GO TO 451
        IF(ABS(BPER).GT..01) GO TO 451                             !(EVEN GROUND?) NO.
        ACCOO=ACCEL
        ACCEL=0.
  452   DT=DT+DEPDT
        CALL GOBACK
        PCTHR=PCTWOT(X)
```

```
        IF ( PCTHR.GE.100.1 )  GO TO 452
        IF(.NOT.LIMPRN) CALL SIMLPT(5)
  451   IF(V.GE.0.) GO TO 455
C
C     CORRECT SMALL NEGATIVE VELOCITY IF FIT OCCURS
C
        DT=-VOLD*1.46667/ACCEL
        CALL GOBACK
        ENDSEG=.TRUE.
        IF(TORQE.LT.TMIN) TORQE=TMIN
        PCTHR=PCTWOT(X)
        GO TO 99                                          !GO UPDATE ACCUMULATORS.
C
C     DETERMINE IF CAR HAS ARRIVED AT REQUIRED ACCELERATION
C
  455   IF(ABS(ACCEL-BASE).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.LMILE) 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
        DTOT=DTOT+1.466667*VOLD*DT+ACCEL*DT*DT/2.
        TTOT=TTOT+DT
        IF(APOS-(DTOT-V0*TTOT*1.466667).GT..001) GO TO 51
        ENDSEG=.TRUE.
        GO TO 99
   51   IF(.NOT.LMP) GO TO 53
C
C     IF ABSOLUTE MILE POST END POINT GIVEN
C
        RMPD=1.466667*VOLD*DT+ACCEL*DT*DT/2.
        IF ( (APOS-(DD+RMPD)).GT..001 )  GO TO 53
        ENDSEG=.TRUE.
        GO TO 99
   53   IF(.NOT.LMPH) GO TO 52
        IF ((T-TSTART).LT.100.) GO TO 52
        IF (ABS(ACCEL).GT.1.E-3) GO TO 52
        IF(LOCKUP(NGEAR).AND.CUMTLS.EQ.CUMT+DT) GO TO 52
C
        CALL SIMLPT(4)                                    ! ? SIMCTR- FAILURE TO REACH TERMINAL VELOCITY

        RETURN
C
C     IF TERMINAL VELOCITY END POINT GIVEN
C
```

```
   52   IF((VSTART-AVEL)*(AVEL-V).LT.0.) GO TO 99
        ENDSEG=.TRUE.
        IF(ACCEL.EQ.0.)ACCEL=ACCOO
   55   IF(.NOT.LLSH) DT=(AVEL-VOLD)*1.466667/ACCEL
        CALL GOBACK
        IF(TORQE.LT.TMIN) TORQE=TMIN
        PCTHR=PCTWOT(X)
        IF (PCTHR.LT.100.1) GO TO 99
        CALL ITERAT(TWOT)
        GO TO 99
C
C     CONSTANT PERCENT WOT SEGMENT
C
   60   DT=DEFDT
        VOLD=V
        PCOLD=PCTHR
        TOLD=T
        T=T+DEFDT
        LWOT=.FALSE.
        TITER=TMIN+APWOO*(TWOT-TMIN)
        IF(LSTROP)GO TO 860                                    ![603]
*                                                              ![613]
        CALL TAPEWR(NAME,20)
*
        CALL ITERAT(TITER)
        PCTHR=PCTWOT(X)                                        ![603]
        GO TO 45                          ![603]FOR PRINT-OUT
                                                               ![603]
C
C     ACCELERATE TO REQUIRED ACCELERATION AND HOLD CONSTANT
C     THROTTLE FROM THEN ON SEGMENT
C
*
*  ITS = 4 STARTS HERE.
*
   80   DT=DEFDT
        VOLD=V
        TOLD=T
        T=T+DT
        ACCEL=SATH
        PCOLD=PCTHR
        LWOT=.FALSE.
        IF(LSTROP)GO TO 860
C                                                              ![613]
C     SEE IF POSSIBLE TO DO IT ON THE MAPS
C
        CALL GOBACK
        PCTHR=PCTWOT(X)
        IF(MAPOK.GT.3) GO TO 800
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 93
        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
 8499   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 8499
  952   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 851
        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(BPER).GT.0.01) GO TO 855
        ACCEL=0.
  856   DT=DT+DEPDT
        CALL GOBACK
        PCTHR=PCTWOT(X)
        IF(PCTHR.GT.100.1) GO TO 856
  855   IF(SATH-ACCEL.GT.0.) GO TO 45
C
C     CONTINUE THE SEGMENT AS IF IT WERE A CONST PERCENT WOT SEGMENT
C
        ARRIVE=.TRUE.
        ITS=3
        APWO=PCTHR
        GO TO 45
C
C     COMPUTATIONS PERFORMED AT END OF EACH TIME STEP
```

```
C
   99   D=1.466667*VOLD*DT+ACCEL*DT*DT/2.
*
        CALL TAPEWR(NAME,21)
*
        ACCO=ACCEL
C
C     UPDATE ACCUMULATORS
C
        IF(V.LT.0.)  V=0.
        DHR=DT/3600.
        DDIS=D/5280.
        DHRD=DHR*.5
        IF(LDIES .AND. PCTWOT(X).LT.1.E-4 .AND.
       2       RPME.GT.ENMIN(IENG)) FRATE=0.0
        DFU=(FRATE+FRATEO)*DHRD
        CUMD=CUMD+DDIS
        CUMT=CUMT+DT
        IF(CUMD.GE.ENDRTE.AND.LSTRTE)  ENDSEG=.TRUE.
        NGOCAL=0                                          !ZERO CNT OF CALLS TO GOBACK.

        VAVG=3600.*CUMD/CUMT
C
C     FUEL ECONOMY COMPUTATIONS
C
        CUMFU=CUMFU+DFU
        HPE=TQFQE*RPME*BB2
        BSFC = 1.0E10
        IF(HPE.GT.0.)  BSFC=FRATE/HPE
        FRATG=FRATE*AA10
        FEINST = 1.0E3
        IF(FRATG.NE.0) FEINST=V/FRATG
        CUMG=CUMG+(FRATG+FRATGO)*DHRD
        RDLD=(FWHEEL-FACCEL)*V/375.
        GALHR=0.
        IF(CUMT.GT.0.)  GALHR=3600.*CUMG/CUMT
        CUMFE=(CUMD-QUMD)/CUMG
C
C     HORSEPOWER AND EFFICIENCY COMPUTATIONS
C
        HPA =TQFQA*RPME*BB2
        HP1 =TORQ1*RPM1*BB2
        HP2 =TORQ2*RPM2*BB2
        HPCL=TORQ2*RPMC*BB2                               !{607}
        HPP =TORQP*RPMP*BB2
        HPW =TORQW*RPMW*BB2
        ALOSSR=ABS(HPP-HPW)
        ALOSS3=ABS(HP2-HPP)
        ALOSSL=ABS(HP2-HPCL)
        ALOSSB = HP1*(1.0 - AVGEFF)
*
        IF(PRINT6)WRITE(6,9001)T,DT,V,RPM2,RPMC,HP2,HPCL,ALOSSL,TORQ2
 9001   FORMAT(' $SIMCTP-T,DT,V,RPM2,RPMC,HP2,HPCL,ALOSSL,TORQ2 ->'/,
     *  9(2X,F9.3)/)
*
        DEN=(HPE+HPEO)*DHRD
        IF(DEN.LT.0.)  CUMENM=CUMENM+DEN
        IF(DEN.GE.0.)  CUMEN=CUMEN+DEN
        EFFC=TR*SR
        IF(COAST.AND.EFFC.GT.1.)  EFFC=1./EFFC
        ALOSS2=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
C
C     IDLE STEP
C
        CFI=CFI+DFU
        CTI=CTI+DT
        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.) CEDN=CEDN+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   PDUM=WRAD*BB2*DHRD
        FF1=PDUM*RPMW
        FF10=PDUM*RPMWO
        CAC=CAC+(HPA+HPAO)*DHRD
        CDV = CDV + (ALOSSB + ALOSBO)*DHRD
        CTR=CTR+(ALOSSC+ALOSCO)*DHRD
        CRO=CRO+FROLL*FF1+FROLLO*FF10
        CPE=CPE+FGRADE*FF1+FGRADO*FF10
        CGB=CGB+(ALOSSG+ALOSGO)*DHRD
        CCL=CCL+(ALOSSL+ALOSLO)*DHRD                              1[60 ]
        CDIF=CDIF+(ALOSSR+ALOSRO)*DHRD
*       DTIRE=ABS(HPW*TTT1)
*       CTIRE=CTIRE+(DTIRE+BTIREO)*DHRD
        IF(.NOT.LSTOP)CBR=CBR+(ABR*RPMW+ABRO*RPMWO)*DB2*DHRD      1[615 ]
        CAE=CAE+PAERO*FF1+PAEROO*FF10
        TEMIOT=CAC+CTR+CRO+CGB+CDIF+CUMENM+CBR+CAE+CCL+CDV
        DCKE = 2.525E-7*AA6*AA4*(V*V - VOLD*VOLB)
*
        RPMSQ = RPMWO*RPMWO - RPMW*RPMW
```

```
      DCROT = 0.5*AA6*((WLSG*AIW + RARSQ*AIP)*RPMSQ +
     1  RARSQ*((AIGOUT(NGEARO) + GRAT(NGEARO)*GRAT(NGEARO)*(AI2 +
     2  AIGIN(NGEARO)))*RPMWO*RPMWO - (AIGOUT(NGEAR) + GRAT(NGEAR)*
     3  GRAT(NGEAR)*(AI2 + AIGIN(NGEAR)))*RPMW*RPMW) +
     4  (EINER + AIA + AI1)*(RPMEO*RPMEO - RPME*RPME))
      CROT=CROT+DCROT
      CKE=CKE+DCKE
      IF(.NOT.PRINT6)GO TO 700
      OTHER=CPE+DCKE-DCROT*5.05E-7
      TOTEN = CUMEN + ABS(OTHER)
      IF(OTHER.GT.0.)TEMTOT=TEMTOT+OTHER
*
      IF(PRINT6)WRITE(6,9000)T,DT,V,CKE,CROT,CRO,CGB,CCL,
     2 CDIF,CTIRE,CBR,CBV,CAE,TEMTOT,TOTEN,OTHER
 9000 FORMAT(' $SIMCTR - T,DT,V,CKE,CROT,CRO,CGB,CCL,CDIF,
     1 CTIRE,CBR,CBV,CAE,TEMTOT,TOTEN ,OTHER->'/2(X8E13.7,/)/)
C
C     DO HISTOGRAM ACCUMULATIONS  [614]
C
700   XLOW=FIRRPM
      XHIGH=XLOW+DELRPM
      DO 7020 IHR=1,20
      IF(RPME.GE.XLOW .AND. RPME.LT.XHIGH)GO TO 7040
      XLOW=XHIGH
      XHIGH=XLOW+DELRPM
7020  CONTINUE
      IHR=20
C
7040  XLOW=FIRTOR
      XHIGH=XLOW+DELTOR
      DO 7060 IHT=1,20
      IF(TORQE.GE.XLOW .AND. TORQE.LT.XHIGH)GO TO 7080
      XLOW=XHIGH
      XHIGH=XLOW+DELTOR
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(LBRAKE) CFB=CFB+DFU
      IF(DEN.LT.0) CENG=CENG-DEN*DHR
      IF(PCTHR.LT.0.) PCTHR=0.
      HPAO=HPA
      HPEO=HPE
      HP1O=HP1
      HP2O=HP2
      HPPO=HPP
      HPWO=HPW
      FRATEO=FRATE
      FRATGO=FRAIG
      DTIREO=DTIRE
      FROLLO=FROLL
      FGRADO=FGRADE
      FWHEEO=FWHEEL
      FAEROO=FAERO
      ALOSBO=ALOSSB
      ALOSCO=ALOSSC
      ALOSGO=ALOSSG
      ALOSRO=ALOSSR
```

```
      ALOSLO=ALOSSL
      RPM2O=RPM2                                          ![607]
      RPMWO=RPMW
      RPMEO=RPME
      TORQAO=TORQA                                        !SAVE ACC TORQ.

      TORQEO=TORQE
      TMINO=TMIN                                          !SAVE MIN ENG TORQ.

      IF(LSTRUP.AND.IDEBUG.EQ.3)CALL DEBUG('START')       !TEMP PRINT
      IF(SHFTNG.AND.LOCKUP(NGEAR)) GO TO 917              !(MANNUAL SHIFTING IN PROGRESS?) YES.[625]

C
C     DETERMINE IF LIMIT PRINT SPECIFIED
C
      IF(.NOT.LIMPRN) 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..0003) 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 ABR=ABRO
C
      CALL SIMLPT(5)                                      !PRINT LINE FOR THIS TIME STEP IF REQUIRED

      IF(LSTRUP)GO TO 867                                 ![613]
C
      IF(.NOT.ENDSEG) GO TO 999                           !(END DRS SEG?) NO.
      KEND=1                                              !FLG DATA LINE FOR LST TIME STEP TO IPT.

      GO TO 110
C
  998 IF(LSTRUP)GO TO 867                                 ![613]
      IF(ENDSEG) GO TO 110                                !(END DRS SEG?) YES.

C
C     CHECK FOR SHIFT POSSIBILITY
C
  999 NGOLD=NGEAR                                         !SAVE GEARS #.

      IF(.NOT.LLSH) GO TO 988
      DT=DTO
      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 TAPEWR(NAME,22)
*
        CALL SHIFTS                                               !YES, TRY IT.
*
        CALL TAPEWR(NAME,23)
*
        IF(.NOT.LLSH) GO TO 102                                   !(DID WE?)NO.
C
C     COUNT UPSHIFTS AND DOWNSHIFTS INTO GEAR
C
        ACCEL=-(FAERO+FROLL+FGRADE)/AA4
        IF ( NGOLD.LT.NGEAR )  GO TO 740                          !(WAS IT DOWNSHIFT?)NO.
        IF(BPER.LE.0.)GO TO 730
        VNEW=VOLD+ACCEL*DT/1.46667
        VELTAR=SHFTRP(2,NGEAR)/(GRAT(NGEAR)*RAR(NAX)*AA5)
        IF(.NOT.(VNEW.GT.VELTAR))GO TO 730
  725   CALL BKSHFT(NGOLD)
        DT=.05
        ACCEL=ACCO
*
        CALL TAPEWR(NAME,24)
*
        CALL GOBACK
*
        CALL TAPEWR(NAME,25)
*
        GO TO 102
  730   NSGEAR(NGEAR,2) = NSGEAR(NGEAR,2) + 1
        LDNSHF=.TRUE.
        GO TO 750
  740   IF(BPER.LE. 0.)GO TO 745
        NGN=(NUMG-NGEAR)+NUMG
        RPMES=RPME
        RPME=SHFTRP(1,NGN)
        TORQA=0.
        IF(NACC.GT.0)TORQA=ENTERP(ACCT(1,1),ACCS(1,1),NVA(1)
        2,NACC,RPME,N20,DUTCYC)
*
        CALL TAPEWR(NAME,26)
*
        CALL ENGINE
*
        CALL TAPEWR(NAME,27)
*
        RPME=RPMES
        DT=(EINER+AIA)*(RPME-SHFTRP(1,NGN))*BB1/(TORQA-TMIN)
        VNEW=VOLD+ACCEL*DT/1.46667
        VELTAR=SHFTRP(1,NGN)/(GRAT(NGEAR)*RAR(NAX)*AA5)
        IF(.NOT.(VNEW.LT.VELTAR))GO TO 745
        GO TO 725
  745   NSGEAR(NGEAR,1) = NSGEAR(NGEAR,1) + 1
        LDNSHF=.FALSE.
  750   NSFSEG(ISEG)=NSFSEG(ISEG)+1
        DRPME=0.                                                  ![607]

        LCLTCH=.TRUE.
```

```
      SHFTNG=.TRUE.                                          !FLAG SHIFT IN PROGRESS.
*
      CALL TAPEWR(NAME,28)
*
      CALL GOBACK                                            ![607]
*
      CALL TAPEWR(NAME,29)
*
      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
*
      CALL TAPEWR(NAME,30)
*
      IF(TORQE.GE.(TMIN-.001)) GO TO 982
      CALL ITERAT(TMIN)
      GO TO 983
C
  982 CALL GOBACK
  983 CONTINUE
*
      CALL TAPEWR(NAME,31)
*
      IF(.NOT.SHFTNG)GO TO 975                               ![SID]
      IF(ISMODE.EQ.2) GO TO 997
C
C     PERFORM CONSTANT THROTTLE SHIFT
C
*
      CALL TAPEWR(NAME,32)
*
      IF(.NOT.(ITS.EQ.1 .AND. AASE.GE.0. .AND. LDNSHF .AND.
     2       ACCEL.LT.0.)) GO TO 984                         ![601]
      CALL ITERAT(TWOT)
      GO TO 985
  984 CONTINUE
      CALL GOBACK                                            ![601]
  985  CONTINUE
*
      CALL TAPEWR(NAME,33)
*
```

```
      PCTHR=PCTWOT(X)
      DIF=HPCTHR-PCTHR
      DACC=1.
      IF(DIF.LT.0.) DACC=-DACC
 9950 ACCEL=ACCEL+DACC
*
      CALL TAPEWR(NAME,34)
*
      CALL GOBACK
*
      CALL TAPEWR(NAME,35)
*
      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 = PCTWOT(X)
      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 TAPEWR(NAME,36)
*
      CALL GOBACK
*
      CALL TAPEWR(NAME,37)
*
      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 = PCTWOT(X)
      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
      VELTAR=SHFTRP(2,NGEAR)/(GRAT(NGEAR)*RAR(NAX)*AA5)
      IF(VNEW.GT.VELTAR)DT=1.46667*(VELTAR+(.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.

C
  917 TSAVE=TSIMP+DT
```

```
      DSAVE=DSAVE+DT*V*1.46667
      TOLD=T
      T=T+DT
      VOLD=V
      ACCEL=-(FAERO+FROLL+FGRADE)/AAA                    !CALC ACCEL OF COASTING VEHICLE DURING CURRENT DT.

      IF(LDNSHF)GO TO 320
      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
      VELTAR=SHFTRP(1,(NUMG-NGEAR)+NUMG) /(GRAT(NGEAR)*RAR(NAX)*AA5)
      IF(VNEW.LT.VELTAR)DT=1.46667*(VELTAR-(.1E-4)-VOLD)/ACCEL   !(IS IT BELOW DNSHFT SPEED) YES.
      IF(DT.LT..01)DT=.05
*
      CALL TAPEWR(NAME,38)
*
920   CALL GOBACK
*
      CALL TAPEWR(NAME,39)
*
      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 LOOKS ONLY.

      ARPME=0.                                           !FOR LOOKS ONLY.

C
  992 CUMTLS=CUMT+DT                                     !SAVE SIM TIME AT END OF SHIFT.

      SHFTNG=.FALSE.                                     !FLG NO LONGER SHIFTING.

      LCLTCH=.FALSE.                                     !POSSIBLLY NOT BELONG, PUT IN CUZ IN CAR MODEL.

      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 PROD
TE.
      CALL DEBUG  ( HAPTPR )                             !DEBUG OUTPUT AFTER SHIFT.

      CUMT   = CUMTO                                     !RESTORE CUMT.

      GO TO 45                                           !GO END OF SEGMENT CHECKS.

C
C     STARTUP CODE [613]
C
860   VOLD=V
      DT=DEFDT
      ACCEL=1.
*
      CALL TAPEWR(NAME,40)
*
      CALL GOBACK
*
      CALL TAPEWR(NAME,41)
```

```
*
      LWOT=.TRUE.
      HPTRYO=0.
      RPME=ENMIN(IENG)
861   CONTINUE
*
      CALL TAPEWR(NAME,42)
*
      CALL ENGINE
*
      CALL TAPEWR(NAME,43)
*
C     HPTRY=RPME*TWOT*BB2
      HPTRY = TWOT
      IF(HPTRY.LT.HPTRYO)GO TO 863                    !ARE WE AT MAX TORQE YET?)YES

      HPTRYO=HPTRY
      RPME=RPME+100.                                  !NO, INC RPM AND TRY AGAIN

      IF(RPME.LE.RPMAX(IENG))GO TO 861
      CALL SIMLPT(15)                                 !CAN'T MOVE)C

      RETURN
C
863   RPME=RPME-100.
865   NDRP=IFIX((RPME-ENMIN(IENG))/100.)
      IDRP=0
*
      CALL TAPEWR(NAME,44)
*
      CALL ENGINE
*
      CALL TAPEWR(NAME,45)
*
      LWOT=.FALSE.
      ACCEL=0.
      TORQW=0.
      TRR=0.
      TORQP=0.
      TORQ2=0.
      TORQ1=0.
      RPMW=0.
      DRPMW=0.
      RPMP=0.
      TORQE=TORQA+TORQF
      V=0.
867   VOLD=V
      IF(LCLTCH)GO TO 868
      IDRP=IDRP+1
      DT=BB1*(EINER+AIR)*100./TWOT
      TORQF=TWOT
      T=T+DT
      RPME=ENMIN(IENG)+100*IDRP
      RPM2=RPME/BVG*SR
      DRPME=RPME-RPMEO
*
      CALL TAPEWR(NAME,46)
*
      CALL ENGINE
*
```

```
*
      IF(IDRP.NE.NDRP)GO TO 99
      RPM1=RPME/BVG
      RPM2=SR*RPM1
      LCLTCH=.TRUE.
      ABR=0.
      LBRAKE=.FALSE.
      RPMC=0.
      DTC=1.
      DRPMC=(RPM2-RPMC)/DTC
      GO TO 99
858   CONTINUE
      IF(.NOT. LOCKUP(NGEAR)) GO TO 869
      RPMC=RPMC+DRPMC*DT
      ACCEL=1.466667*(RPMC/(RAR(NAX)*GRAT(NGEAR)*AA5)-VOLD)/DT
      GO TO 870
  869 CONTINUE
      RPMC  = RPM2
      ACCEL = 3.0
  870 CONTINUE
*
      CALL TAPEWR(NAME,49)
*
      CALL GOBACK
*
      CALL TAPEWR(NAME,49)
*
      IF(MAPOK.LE.3)GO TO 99
*
      CALL TAPEWR(NAME,50)
*
      CALL ITERAT(TWOI)
*
      CALL TAPEWR(NAME,51)
*
      IF(ACCEL.GT.0)GO TO 99
      CALL SIMLPT(13)
      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 TO 105                  !(END RTE SEG?) NO.
      NRTSEG=NRTSEG+1                               !YES, INC RTE SEG PTR

      IF(NRTSEG+NDRTE.LE.NRDIST) GO TO 104          !(END OF RTE SECT?) NO.
      IF(LSTRTE) GO TO 1101                         !YES,(LAST RTE SECT?) YES.

  103 IPRNT=209                                     !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
        BPER=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-AVSE).GE.0.1))) GO TO 105
        ITS=2
        SDELAY=.8
        IF(BPER.GT.0.)SDELAY=SDELAY*BPER
        NGCNT=0
  105   IF(ENDSEG) GO TO 112                                     !(END OF DRSSEG?) YES.

C
C     RESET BRAKE FLAG FOR NEXT TIME STEP
C
        ABROO=ABRO                                               !SAVE OLD BRAKE.

        ABRO=ABR                                                 !SAVE BRAKE.

        ABR=0.                                                   !ZERO BRAKE.

        LBRAKE=.FALSE.                                           !BRAKES OFF.

*
        CALL TAPEWR(NAME,52)
*
        GO TO 10                                                 !NEXT TIME STEP PLEASE.

C
  110   IF(COMD.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.SECTM.AND..NOT.MILTM).OR.(KEND.GT.0)) GO TO 120
        LIDLE=.FALSE.                                            !(607)
        IF((ITS.GT.1).OR.(ACCEL.GE.0.).OR.(V.GT.1.E-4)) GO TO 122
        ABR=ABRO
        IF(ABR.LT.ABROO-10.) ABR=ABROO
C
  122   CALL SIMLPT(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
        CDA=100.*CDA/CUMD
        CEA=100.*CEA/CUMEN
        CFA=100.*CFA/CUMFU
C
C     DECELERATION CONDITION
C
        CTD=100.*CTD/CUMT
        CDD=100.*CDD/CUMD
        CED=100.*CED/CUMEN
        CFD=100.*CFD/CUMFU
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 = 2.525E-7*AA6*AA4*(V*V - VO*VO)
*
        RPMSQ = RPMWI*RPMWI - RPMW*RPMW
        CROT = 0.5*AA6*((WLSG*AIW + RARSQ*AIP)*RPMSQ +
        1  RARSQ*((AIGOUT(NGO)  + GRAT(NGO)*GRAT(NGO)*(AI2 +
        2  AIGIN(NGO)))*RPMWI*RPMWI - (AIGOUT(NGEAR) + GRAT(NGEAR)*
        3  GRAT(NGEAR)*(AI2 + AIGIN(NGEAR)))*RPMW*RPMW) +
        4  (EINER + AIA + AI1)*(RPMEI*RPMEI - RPME*RPME))
        CROT=-CROT*5.05E-7
        OTHER=CPE+CKE+CROT
        CUMENM = -CUMENM
        TOTEN = CAC+CBR+CBV+CIR+CRO+CAE+CGB+CCI+CDIF+CTIRE+CUMENM
        1  +ABS(OTHER)
        CAC=100.*CAC/TOTEN
        CBR=100.*CBR/TOTEN
        CBV=100.*CBV/TOTEN
        CTR=100.*CTR/TOTEN
        CRO=100.*CRO/TOTEN
        CAE=100.*CAE/TOTEN
        CGB=100.*CGB/TOTEN
        CCI=100.*CCI/TOTEN                                          [ 60
        CDIF=100.*CDIF/TOTEN
        CTIRE=100.*CTIRE/TOTEN
        CUMENM=100.*CUMENM/TOTEN
C
C     COMPUTE SUBTOTALS OF ENERGY LOSSES
C
        C345   = CGB + CDIF + CTIRE
        C2345  = CTR + C345 +CCL                                    [ 60
```

```
      C1TO7  = CAE + CRO + CAC + C2345 + CRV
C
      CENERG=100.*OTHER/TOTEN
      IF(CENERG.LT.0.)  CENERG=0.
C
C     TOTAL INCLUDING BRAKE AND RETURN TO ENGINE
C
      CTOTAL = C1TO7 + CBR + CUMEPM
      PCKE=-CKE
      PCPE=-CPE
      PCROT=-CROT
C
C     ENERGY LOSSES
C
      CALLT=CTOTAL+CENERG
      PPHPHR=CUMPU/CUMEN
      HPMI=CUMEN/CUMD
C
C     PRINT UPSHIFT/DOWNSHIFT GEAR DATA
C
      IF(.NOT.(MILIM.OR.SECLIM.OR.ENDLIM.OR..NOT.LIMPRN) )  GO TO 770
      NSHIFT    = 0
      DO 760  I = 1,20
      DO 760  J = 1,2
  760 NSHIFT    = NSHIFT + NSGEAR(I,J)
      SMILE     = NSHIFT / CUMD
      ASFPSG= NSHIFT / NSEG
C
      CALL SIMLPT(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.
      TFRC4 = FRC4 * 1000.
C
      CALL SIMLPT(9)                                    !FINAL SUMMARY PAGE.

      IECOND=1                                          !FIG NO ERROR.

      RETURN                                            !BYE, WE FANED SAFELY.

C
C     PRINT OUT A LINE ONLY IF LAST TIME STEP IN SEGMENT HAS BEEN SPECIF
C
  112 IF(.NOT.ENDLIM) 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 SIMLPT(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 #.
```

```
      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
      IF(NGEAR.NE.1)NSGEAR(NGEAR,2)=NSGEAR(NGEAR,2)+1
                                                !(IDLE?)NO, GO TO NEXT SEGMENT.
                                                !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 SIMLPT(10)
      GO TO 5040
 5020 CALL SIMLPT(11)
 5040 IF(PTY)CALL EXIT                          !(PSEUDO-TTY)YES, EXIT.

      CALL RESETM                               !NO, RESET.

C
      END
```

```
*                **** SIMINT ****
*
      SUBROUTINE SIMINT
*
*  MODIFIED 22 SEPT 1980: SOMERS
C
C     ENTRY POINTS:   SIMINT
C
C     SUBROUTINES CALLED:     GOBACK, KPTIME, RESETM, SHIFTS,
C                             SIMLPT, TTYCLR
C
C     CALLED BY:      SIMCTR
C
C     EDIT HISTORY
C
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     (724)/SS-2-1-79         CALL TO STSHFT TO SET UP RERAT ARRAY FOR SHIFTING
*          /LS-9-22-80        ACCESSORY SPEED RATIO ADDED
C
C*****************************************************************
C
C
      INCLUDE 'COMMS/NOLIST'
      COMMON/JDSFIX/QUMD
C
      DIMENSION 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./
C
C*****************************************************************************
      JCT=5
C
C...  HISTOGRAM INIT
C
      CALL ZEROP(HIST,1200)
      FIRTOP=0.
      FIRRPM=ENMIN(1)
      TOPRPM=RPMAX(1)
      IF(NENG.EQ.1)GO TO 10
      IF(ENMIN(2).NE.0..AND.ENMIN(2).LT.FIRRPM)FIRRPM=ENMIN(2)
      IF(RPMAX(2).GT.TOPRPM)TOPRPM=RPMAX(2)
C
10    IF(FIRRPM.LT.100. .OR. TOPRPM.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.EQ.2)IM=5
      DO 20 IR=1,NRPM(IE)
      DO 20 IT=1,20
      TEMP=EMAP(IR,IT,IM)
      IF(TEMP.LT.FIRTOR)FIRTOR=TEMP
      IF(TEMP.GT.TOPTOR)TOPTOR=TEMP
20    CONTINUE
C
28    DO 30 I=1,10
```

```
      IF(FIRRPM+I*200..GT.TOPRPM)GO TO 32
30    CONTINUE
      I=10
32    DELRPM=I*100
      FIRRPM=IFIX(FIRRPM/DELRPM)*DELRPM
      IF(FIRRPM+DELRPM*20. .LT.TOPRPM)GO TO 28
C
38    DO 10 I=1,10
      IF(FIRTOR+I*200.GT.TOPTOR)GO TO 42
40    CONTINUE
      I=10
42    DELTOR=I*10
      FIRTOR=(IFIX(FIRTOR/DELTOR)-1.)*DELTOR
      IF(FIRTOR+DELTOR*20..LT.TOPTOR)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 TTYCLR                             ! (PHYSICAL TTY IS JCT?) YES, CLEAR JCT INPUT BUFF

*  ACCESSORY SPEED RATIO IS DEFINED AS RATIO OF ACCESSORY SPEED
*  TO ENGINE SPEED.
      AIA=0.0                                         !ZERO ACCESSORY INERTIA.

      DO 90 J = 1,NACC
      ASR = ACCSR(J)
      IF(ASR .EQ. 0.0) ASR = 1.0
      ASRSQ = ASR*ASR
      AIA   = AIA + AIAS(J)/ASRSQ
      DO 90 I = 1,NNA(J)
      ACCS(I,J) = ACCS(I,J)/ASR
      ACCT(I,J) = ACCT(I,J)*ASR
   90 CONTINUE
*
*  SAVE INERTIA VALUES
*
      SAI1 = AI1
      SAI2 = AI2
*
*
      DO 100  I = 1,20
      NSGEAR(I,1)=0
  100 NSGEAR(1,2)=0                                   ! ZERO UP,DOWN SHIFT ACCUMS

      FAC=.0025168
      AA1=FRC1*WGT
      AA2=FRC2*WGT
      AA11=FRC4*WGT
      AA3=FAC*CD*AREA
      AA4=WGT/32.17
      AA5=14./WRAD
      NAX=NAG
      RARSQ=RAR(NAX)**2
      AA6=BB1**2
      AA7=BB1*RARSQ
      AA8=DZR(NAX)*ERAR(1,NAX)                        ! ASSUME 1 AXLE
      IF(NAX.EQ.2) AA8=1./(.5/AB+.5/(RAR(NAX)*ERAR(2,NAX)))   ! (2 AXLES?) YES.
      AA9=BB1*(EIDEL+AIA+AI1)
```

```
      AV10=7.180529/(FSPGR*62.426134)
      IF(.NOT.LDYNA) GO TO 200                    |(DYNA SIM?)NO.
      IF(LDYNOV)GO TO 200                         |[612](OVERRIDE?)NO

      IF(WGT.LT.WGTLIM(0)) GO TO 125              |YES,(WGT ABOVE MIN?)NO.
      DO 120 I=1,11                               |SCAN WGT TABLE.

      IF(WGT.LT.WGTLIM(I)) GO TO 130              |(GOT WGT CLASS?)YES.
120   CONTINUE                                    |NO, 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   RTHI=4000.                                  |RATE OF CHG FOR THROTTLE,(400 FOR 20%,4000FOR 100%)

      LLSH=.FALSE.
      CUMIN=0.
      CUMFU=0.
      CUMINM=0.
      CFDN=0.
      LBRAKE=.FALSE.
      CLUTCH=.FALSE.                              |[607]
      CCL=0.                                      |[607]

      CBV=0.0
      CTI=0.
      CDI=0.
      CFI=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.
      CIR=0.
      CRO=0.
      CAE=0.
      CBR=0.
      CGR=0.
      CDIF=0.
      CTIRE=0.
      CENG=0.
      CPE=0.
      CKE=0.
      CROT=0.
      ABR=0.
```

```
      ARRO=0.
      NGOCAL=0
C
      T=TO                                  |SET INITIAL CONDITIONS AS SPECIFIED BY DRIVING SCHEDULE

      MPRC=0.
      D=DO
      TTOT=0.
      DTOT=0.
      IC=0
C
      V=VO
      NDRTE=0                               | # OF RTE SEGS EX. IN PAST RTE SECTS

      NRTSEG=1                              |PTR TO 1ST RTE SEG.

      IF(LDYNA) 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.
      ENDRSG=RDIST(1)                       |END OF RTE SEG.

      ENDRTE=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 ENDRSG=1.E20                          |INFINITE ROAD. MOST ROADS THAT GO IN CIRCLES ARE.

      ENDRTE=1.E20                          |INFINTE ROAD.

      ROADC=1.0                             |ROAD COEFF. PERFECT.

      BPER=0.                               |GRADE.

      VSWIND=0.                             |NO WIND.

      PHI=0.                                |WIND ANGLE.

      FAERO=0.
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.

      MXNGCN=NUMG*2+5                       |CALC MAX ATTEMPTS TO FIND GEAR OR WE HAVE STUTTERS.

      IENG=JENG(NGEAR)                      |GET INITIAL ENG # TO USE 1 OR 2.

      RPMLO=ENMIN(IENG)                     |SET OLD RPME TO MIN ENG RPM.
```

```
      DRPMC=0.
      RPMC=0.
      DTC=0.
      RPM20=0.
      RPMWO=0.                                          !OLD VAL OF WHEEL ROM.

      ACCEL=AG                                          !GET INITIAL ACCEL.

      ITT1=0.
C START CUM. DISTANCE AND TIME AT INITIAL DISTANCE AND TIME (JDS, 25-OCT-79)
      CUMD=D
      CUMT=T
      CUMG=0.
      CUMTLS=0.                                         !CUMT AT END OF LAST SHIFT.

      CSTIM=0.                                          !SHIFT TIME ACCUM.

      CALL STSHFT                                       !SET UP SHIFTING ARRAYS AND PARAMETERS.

C
C     GO BACK FROM WHEELS TO ALLOW CAR 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.

CTRY  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.MXNGCN )  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 SIMLPT(2)                                    !% CAN'T FIND INITIAL GEAR, BUT WE'LL TRY SEE IF SIM FLIES A
NYWAY.
C
  403 SHFTNG=.FALSE.                                    !FLG NOT SHIFTING.

      RPMLO=RPME
      RPMEO=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
      HPAO=TORQA*RPME*BB2                       !HORSE POWER.
      HPEO=TORQE*RPME*BB2
      HP1O=TORQ1*RPM1*BB2
      HP2O=TORQ2*RPM2*BB2
      HPCLO=TORQ2*RPMC*BB2
      HPPO=TORQP*RPMP*BB2
      HPWO=TORQW*RPMW*BB2
      FRATEO=FRATE                              !FORCES.
      FRATGO=FRATE*AA10
      FROLLO=FROLL
      FGRADO=FGRADE
      FWHEEO=FWHEEL
      FAEROO=FAERO
      DTIRE=0.                                  !LOSSES.

      DTIREO=ABS(HPWO*TTT1)
      ALOSBO=ABS(HP1O)*(1.0 - BVGEFF)
      ALOSGO=ABS(HP2O-HPPO)
      ALOSPO=ABS(HPPO-HPWO)
      ALOSCO=ABS(HP2O-HP1O)
      ALOSLO=ABS(HP2O-HPCLO)
      PCTHR=100.*(TORQE-TMIN)/(TWOT-TMIN)       !% THROTTLE.
      RPMEI  = RPME
      RPMWI  = RPMW
      NGO    = NGEAR
      TORQEO=TORQE
      TORQAO=TORQA
      TMINO=TMIN
      ATORQF=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=TORQE*RPME*BB2
      BSFC = 1.0E10                             !FOR LOOKS.

      IF(HPE.GT.0.) BSFC=FRATE/HPE              !(ENG GOT POWER?)YES, CALC A REAL VALUE.

      COMFE=0.                                  !GREAT MPG.

      ROLL=(FWHEEL-FACCEL)*V/375.
      HPW=TORQW*RPMW*BB2
      HPE=TORQE*RPME*BB2
      EFFC=SR*TR
      IF(COAST.AND.EFFC.GT.1.) EFFC=1./EFFC
      IF(PCTHR.LT.0.) PCTHR=0.
C
```

```
      CALL SIMLPT(1)                                        !INIT SIM PRINT ROUTINE.
      CALL SIMLPT(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
```

```
*               **** SIMLPT ****
*
      SUBROUTINE SIMLPT(ITASK)
*
*  MODIFIED 4 SEPT 1980: SOMERS
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
C*****************************************************************
C
      INCLUDE 'COMMS/NOTISE'
      DIMENSION TEMPR(21),TEMPT(21)
C
C*************************************************************************
C
      JCT=5
      LPT=IUNIT                                                 !GET LPT UNIT #.

      ISEG2=ISEG+NDSEG                                          !CALC CURRENT DRS SEG.

      GO TO (10,20,20,20,30,20,20,800,900,20
     2 ,20,1200,1300,1400),ITASK                                !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.LLPT))GO TO 9000
      GO TO (9999,200,300,400,500,600,700,800,900,1100
     2 ,1120,1200,1300,1400,1500),ITASK                         !GO OUTPUT.
C
C
C*************************************************************************
C
C   ERROR MESSAGE - FAILURE TO REACH INITIAL CONDITIONS
C
  200 CALL TTYSET                                               !RESET SO 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
      IF(LOCKUP(NGEAR)) TSR=1.
      IF(LOCKUP(NGEAR)) TEFFC=1.
      GRDBLT = 10E20
      HRDIF  = HPW - RDLD
      IF(HRDIF < 0.0) GO TO 510
      GRDBLT = ASIN(HRDIF*375.0/V/WGT)
      GRDBLT = 100.0*SIN(GRDBLT)/COS(GRDBLT)
  510 CONTINUE
      IF(LPT.EQ.IUNIT)                                               !(LPT ON?)YES.SEND DATA TO LPT.

      1WRITE(LPT,1002) CUMT,CUMD,V,ACCEL,FEINST,BSFC,CUMFE,NGEAR,NAX,
      1RDLD,HPW,HPE,RPME,TORQE,VAC,TSR,TEFFC,PCTHR,ISEGA,ABR,GRDDLT
      IF(TTY .AND. (.NOT. LIMTTY) )                                  !(JCT IS TTY & TTY ON?)
      1 WRITE(JCT,1012) CUMT,CUMD,V,ACCEL,NGEAR,NAX,HPW,HPE          !YES. SEND DATA TO TTY.
      1,RPME,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 CMPLD('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 TO AND CLR JCT INP BUFFER.

        WRITE(IPT,1980)  TORQE,TMIN,ISEGA,CHNT                         !ERR MESS.
        GO TO 9000
C
C*******************************************************************************
C
C   PRINT SHIFTING DATA
C
  900   IF(LPT.EQ.5.AND.LIMTTY.AND.IDEBUG.NE.2) GO TO 700              !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) ASFSG                                          !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)  (NSFSEG(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=0                                                        !SET LINE CNT.

        WRITE(IPT,1774) NPAGE                                          !PAGE AND WRITE HEADER.

        GO TO 772                                                      !CONTINUE.

  780   GO TO 9000
C
C*******************************************************************************
C
```

```
C     WRITE TOTALS FOR THE ENTIRE SIMULATION
C
  900   NPAGE=NPAGE+1                                                    !INC PAGE CNT.
        WRITE(LPT,1005)  DATE,NPAGE,TITLE,CUMPE,HPMI,PPWPHR,VAVG
C
C     CAPSULE SUMMARY OF DATA USED FOR THIS SIMULATION
C
        DIECOM=D
        IF(LDIES)DIECOM='[D]'
        IF(.NOT.LDYNOV)GO TO 920                                       ![612]
        WRITE(LPT,1108)  DNAME,RNAME,DYN,VNAME,ENAME(1),DIECOM
        2,CNAME,SNAME,WGT
        1,STROKE,DISP,RAR
        WRITE(LPT,11080)VWIND,FRGAL,AREA,CD,TERC1,TERC2,TERC4,TIREFF
        GO TO 940
C
920     WRITE(LPT,1008)  DNAME,RNAME,VNAME,ENAME(1),DIECOM
        2,CNAME,SNAME,WGT
        1,STROKE,DISP,RAR
        WRITE(LPT,10080)VWIND,FRGAL,AREA,CD,TERC1,TERC2,TERC4,TIREFF
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,PCKE,PCPE,PCROT
C
C     LOSSES AS PERCENT AVAILABLE TOTAL ENERGY
C
        WRITE(LPT,1007) CAC,CTR,CCL,CGB,CDIF,CTIRE,C345,C2345,CAR
        2,CRO,C1T07,CBR,CUMENM,CTOTAL,CENERG,CALLT                     ![607]
C
C...    HISTOGRAM OUTPUT
C
        IF(LPT.EQ.JCT)GO TO 9000
        NPAGE=NPAGE+1
        TEMPR(1)=FIRRPM
        TEMPT(1)=FIRTOR
        DO 9420 I=2,21
        TEMPR(I)=TEMPR(I-1)+DELRPM
        TEMPT(I)=TEMPT(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 BOUDARY 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
9480     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=1,3),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=1,3),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=1,3),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,CUMT,RPM1,ENMIN(1)
      GO TO 1140
 1120 WRITE(LPT,2120)ISEGA,CUMT,RPM1,RPMAX(IENG)
 1140 IF(LPT.EQ.JCT)CALL CTRLD('STALL')
      GO TO 9000
C
C*****************************************************************************
C
C   ERROR MESSAGE - BAD DRIVING SCHEDULE
C
1200  CALL TTYSET
      WRITE(LPT,2200)ISEGA,CUMT
      CALL RESETM
      GO TO 9000
C
C*****************************************************************************
C
C   ERROR MESSAGE - NO TORQUE
C
1300  CALL TTYSET
      WRITE(LPT,2300)ISEGA,CUMT
      CALL CTRLD('NOTOR')
      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
C   ERROR MESSAGE - BAD ENGINE NO GOVERNOR DROOP - STARTUP
C
1500  CALL TTYSET
      WRITE(LPT,2500)
      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 (1H1/6XA8,31X'VEHICLE PERFORMANCE SIMULATION'32X,
        1         5HPAGE ,I3/6X,9(1H-),31X,30(1H-),33X,8(1H-)//
        2,26X,12HRUN TITLE ( ,12A8,2H ),/,26X,74(1H-),//
        3         29X'DRIVING SCHEDULE ( '
        4A10' )'10X'USING ROUTE ( 'A10' )'/29X31(1H-)10X36(1H-)//
        5,32X'INST'8X'CUM'6JX'DRS'/
        6 X,4HSEC.,3X,5HMILES,3X,3HMPH,3X,3HACC,4X,3HMPG,2X,4HBSFC,2X,
        73HHP3,1X,4HGEAR,2X,4HRPLD,4X,3HHPW,4X,3HHPE,3X,3HRPM,2X,4HTORQ,
        83X,3HVAC,3X,2HSR,3X,3HETA,3X,4H%HOT,2X,3HSEG,1X,6HBRAKES,
        9 X6HGRADE//2X,125(1H-)/)
C
C=================================================================
  204   FORMAT (//2X,5X,60(1H*)//2X,30H***** FAILURE TO REACH INITIAL,
        1         23H CONDITION GEAR SETTING/2X,16H***** EXECUTION ,
        2         25HCONTINUES STARTING IN GEAR,I4//2X,5X,60(1H*)//)
C
C=================================================================
 1990   FORMAT(//,7X,65(1H*),//,3X
        1,'***** SHIFT STUDIER DETECTED DURING CONSTANT VELOCITY '
        2'MENT' ,/,43H  ***** UNABLE TO ATTAIN CONSTANT VELOCITY   ,/
        3,48H  *****       POSSIBLY DUE TO FAULTY SHIFT LOGIC  ,/
        4,13H  ***** SEG = ,I5,10H   AT TIME ,F8.2,17H 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         21H 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,F6.2,2XF5.2,XF5.2,F5.1,I3,I2,F6.1,
        1         2F7.1,2F6.0,F6.1,2F6.3,F6.0,I4,F7.1,XF6.2)
C
C=================================================================
 1960   FORMAT (//2X,5X,60(1H*)//2X,31H***** TORQUE REQUIRED BY ENGINE,
        1   32H BELOW MINIMUM PRIOR TO SHIFTING/2X,15H***** TORQUE = ,
        2   E10.3,5X,10HMINIMUM = ,E10.3/2X,17H***** FOR SEGMENT,I5,
        3   2X, 8H AT TIME,F8.3/2X,16H***** SIMULATION,
        4         11H TERMINATED//2X,5X,60(1H*)//)
C
C=================================================================
 1010   FORMAT('1SHIFT FREQUENCY DATA BY GEAR'33XA8
        1,20X,5HPAGE ,I3,/1X,28(1H-),60X,8(1H-)
        1,///,3X,15HTOTAL SHIFTS  =,I6,5X,18HSHIFTS PER MILE  =,
        2         F6.1,5X,13HNUMB 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,11HDOWNSHIFTS ,20I3//)
C
C=================================================================
 1770   FORMAT(32H SHIFT FREQUENCY DATA BY SEGMENT,/,1X,31(1H-),//
        1,23H   SHIFTS PER SEGMENT =,E10.3)
```

```
 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=======================================================================
 1000   FORMAT (// 1X,12HADDITIONAL 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)       =,F7.3/
       9         3X,23HDISPLACEMENT (CU IN)  =,F7.1,
       1         7X,23HREAR AXLE RATIO       =,3F7.2)
10000  FORMAT( 3X,23HWIND VELOCITY (MPH)   =,F7.1,
       2         7X,23HFUEL DENSITY (LB/GAL) =,F7.3/
       3         3X,12HAERO DRAG  =,F6.2,2H ,F5.2,5X,
       4         7X, 8HTIRES  =,F7.2,3(2H ,F5.2))
C
 1100   FORMAT (// 1X,12HADDITIONAL 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)       =,F7.3/
       9         3X,23HDISPLACEMENT (CU IN)  =,F7.1,
       1         7X,23HREAR AXLE RATIO       =,3F7.2)
11000  FORMAT(3X,23HWIND VELOCITY (MPH)   =,F7.1,
       2         7X,23HFUEL DENSITY (LB/GAL) =,F7.3/
       3         3X,12HAERO DRAG  =,F6.2,2H ,F5.2,5X,
       4         7X, 8HTIRES  =,F7.2,3(2H ,F5.2))
C
C=======================================================================
 1000   FORMAT (//1X, 6HTOTALS,30X,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         18X,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,I3,I2,F6.1,3(1XF7.1)1XI3)
C
C=====================================================================
 2010    FORMAT('1  'A9,9X'VEHICLE PERFORMANCE SIMULATION'9X'PAGE'I0//
     1,3X'RUN TITLE ( '12A5' )'//
     2,8X'DRIVING SCHEDULE ( 'A10' )'6X'USING ROUTE ( 'A10' )'//
     3,16'SEC.'T14'MILES'T23'MPH'T31'ACC'T35'GEAR'T43'HPW'
     4,T51'HPE'T59'RPM'T66'TORQ'T71'SEG'/1X72('-')/)
C
C=====================================================================
   502   FORMAT(' ? SIMCTR -  OVER SHOT END OF SEGMENT'IS
     1,/10X'THIS A CONSTANT VELOCITY SEGMENT AND OVER SHOOT'
     2,/10X'CAUSED BY SHIFT SPLITTER OR SEGMENT TIME TO 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=====================================================================
2500  FORMAT(' ? SIMCTR - BAD ENGINE. NO GOVERNOR DROOP',/,
     2 ' CAN''T FIND MAXIMUM ENGINE HP')
C
C=====================================================================
3000  FORMAT(1H1/10X'VEHICLE PERFORMANCE SIMULATION'3X8
     1,40X,5HPAGE ,I3,/,10X,42('*')///
     210X,'BREAKDOWN OF % TIME SPENT ON VARIOUS PARTS OF ENGINE MAP',///
     3,3X,11F8.1,/,1X,11(7(' '),'!'))
3020  FORMAT(20('*'F6.0,'-+',10('--------+'),/,'*'
     2       11(7(' '),'!'),/,'*'
     3       7(' '),'!',10(F6.0,' !'),/,'*'
     4       7(' '),'!',10(F6.0,' !'),/,'*'
     5       7(' '),'!',10(F6.0,' !')/))
3040  FORMAT('+',F6.0,82('-'))
C
3060  FORMAT(1H1/10X'VEHICLE PERFORMANCE SIMULATION'3X8
     1,40X,5HPAGE ,I3,/,10X,42('*')///
     210X,'BREAKDOWN OF % TIME SPENT ON VARIOUS PARTS OF ENGINE MAP',///
     3,3X,15F8.1,/,1X,15(7(' '),'!'))
3080  FORMAT(20('*'F6.0,'-+',14('--------+'),/,'*'
     2       15(7(' '),'!'),/,'*'
     3       7(' '),'!',14(F6.0,' !'),/,'*'
     4       7(' '),'!',14(F6.0,' !'),/,'*'
     5       7(' '),'!',14(F6.0,' !')/))
3100  FORMAT('+',F6.0,114('-'))
      END
```

```
      SUBROUTINE SIMSTS(CHAR)
*
*  MODIFIED 23 SEPT 1980 SOMERS
C
C     ENTRY POINTS:   SIMSTS
C
C     SUBROUTINES CALLED:     CTRLD,  EXIT,   KPTIME,
C                             INPPSE, TTYCLR, TTYSET
C
C     CALLED BY:      GOBACK, SIMCTR
C
C*****************************************************************
C
C
      INCLUDE 'COMMS/NOLIST'
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-D(DEBUG TO JCT.
  200 CALL CTRLD('CTRLD')
      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,CUMO,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,CUMFE,CPUT   !TO JCT.
        GO TO 900                                     !BYE.
  550   WORDT=HERTE                                   !SET TO END RTE.

        PNAME=RNAME                                   !GET RTE NAME.

        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   LIMTTY=.NOT. LIMTTY                           !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('.% SIMSTS-ILLEGAL INTERUPT CHARACTER.'
        1,' TYPE ^H FOR HELP'/)
```

```
 1100   FORMAT(1XA4'I'I3' CUMT:'F8.2' CUMD:'F8.3' MPH:'F5.1
        1,' CPUS:'F6.2' CPUT:'F8.2/)
 1300   FORMAT(/' (SIMULATION CONTINUING)')
 1500   FORMAT(1XA4'I'A10'  CUMT:'F8.2'  CUMD:'F8.3'  MPG:'F5.1
        1,'  CPUT:'F6.2/)
 1600   FORMAT(/' HEVSIM INTERRUPT 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 - SUPPRESS END OF DRIVING SEGMENT MESSAGES'
        3,/'    ^P - PAUSE TO MODIFY AND THEN CONTINUE SIMULATION'
        4,/'    ^S - TYPE OUT CURRENT STATUS OF SIMULATION'
        5,/'    ^U - 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
```

```
*
*               **** SKPREC ****
*
      SUBROUTINE SKPREC(IUN,I END)
      DO 40 I=1,I END
      SKIP RECORD IUN
40    CONTINUE
      RETURN
      END
```

```
*                ****  SLOOKP  ****
*
      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=ICBCNT(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
```

```
*       **** LINE PRINTER VERSION ****
*
        SUBROUTINE TAPEWR(NAME,NUMBER)
*
*   MODIFIED 4 SEPT 1980: SOMERS
*
        INCLUDE 'COMMS/NOLIST'
        DATA ILINE,IPAGE/2*0/
*
*
        RETURN
*
*
        IF(ILINE .EQ. 0) GO TO 20
   10   CONTINUE
        WRITE(20,100) NAME,NUMBER,ITS,MAPOK,T,TOLD,DT,D,V,
     1  VOLD,ACCEL,TORQE,RPME,TORQA,TORQ1,RPM1,TORQP,RPMP,ABR
*
        WRITE(21,110) T,RPMC,SR,TR,TORQW,RPMW,TRR,TORQF,TORQ,FWHEEL,
     1  FAERO,FACCEL,FROLL,RPM2
*
        ILINE = ILINE + 1
        IF(ILINE .EQ. 52) ILINE = 0
*
        RETURN
   20   CONTINUE
        IPAGE = IPAGE + 1
        WRITE(20,120) IPAGE
        WRITE(21,130) IPAGE
        GO TO 10
*
  100   FORMAT(X,A5,I3,2I2,2F7.3,F9.3,F8.3,3F7.3,F9.2,
     1  F8.2,F7.2,2F8.2,F9.2,F7.2,4XF7.2)
  110   FORMAT(X,F5.3,F10.3,2F6.3,F10.2,F7.2,F10.3,F9.2,F10.2,F11.3,
     1  F7.3,F11.3,F7.2,F8.3)
*
  120   FORMAT('1NAME NUM I M   T      TOLD       DT',
     1  '       D        V     VOLD    ACCEL     TORQE     RPME  ',
     2  'TORQA    TORQ1     RPM1     TORQP    RPMP    PAGE',I4,/126X'BRAKES')
  130   FORMAT('1  T        RPMC     SR     TR        TORQW    RPMW     ',
     1  'TRR.     TORQF       TORQ      FWHEEL  FAERO      FACCEL   ',
     2  'FROLL     RPM2',6X,'PAGE',I4,/)
*
        END
```

```
*               **** VALID2 ****
*
        SUBROUTINE VALID2(PNAME,*)
C
C       ENTRY POINTS:   VALID2
C
C       SUBROUTINES CALLED:     ICRCNT, IGET,   PUT
C
C       CALLED BY:      INPBAT
C
C*****************************************************************
C
        DOUBLE PRECISION PNAME,POINT,DBLANK,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(PNAME.EQ.DBLANK)GO TO 100
        IF(PNAME.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 ****
*
      BLOCK DATA VSMBLK
C
C
      INCLUDE 'COMMS'
C
C
      DATA HDBLANK/' '/,DSTAR/'*'/,HSTAR/'*'/,DBLANK/' '/
      DATA HUP/'UP'/,HDN/'DN'/,HUN/'UN'/,HQ/'?'/, RZERO/0.0/
      DATA SEQIN/'SEQIN'/,SEQOUT/'SEQOUT'/,SEQIO/'SEQINOUT'/
      DATA APPEND/'APPEND'/,DELETE/'DELETE'/,BINARY/'BINARY'/
      DATA RENAME/'RENAME'/
      DATA SNGLDS/.FALSE./,MASDEV/'DSKB   '/,MASPPN/"3161,"444/
      1,(MASFIL(I),I=1,12)/'VSMENG.BIN','VSMTCV.BIN','VSMVEH.BIN'
      3,'VSMGRS.BIN','VSMACC.BIN','VSMDRS.BIN','VSMSLG.BIN'
      4,'VSMROU.BIN','VSMTIR.BIN','VSMTRA.BIN','VSMAXI.BIN'
      5,'VSMBAS.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 (DASFIL(I),I=1,12)/'VSMENG.BIN','VSMTCV.BIN','VSMVEH.BIN'
      2,'VSMGRS.BIN','VSMACC.BIN','VSMDRS.BIN','VSMSLG.BIN'
      3,'VSMROU.BIN','VSMTIR.BIN'
      4,'VSMTRA.BIN','VSMAXI.BIN'
      6,'VSMBAS.BIN'/
C
      DATA DASDEV/15*'DSKB       '/
C
      DATA DATPPN/15*"000000003213000000000672/
      DATA LPTDSP/'APPEND'/,BATCH/.TRUE./,DIALOG/.FALSE./
      1,NDSEG/0/
C
C*****************************************************************************
C
C
      DATA DB1/0.104720/,DB2/1.20404E-4/
      1,WMASK/"774000000000,"777760000000,"777777700000,"777777777400
      2,"777777777776/,DWMASK(10)/"777777777776777777777776/
      DATA NPMPAR/NPART/
C
      END
```

```
*              **** VSMCTR ****
*
      SUBROUTINE VSMCTR
*
*  MODIFIED 23 SEPT 1980: SOMERS
C
C       ENTRY POINTS:   CLOSE,  RENTER, VSMCTR
C
C       SUBROUTINES CALLED:     CHKFIL, DSKCTR, EXIT,   FILSPC,
C                       INPBAT, INPDIA, JOBPPN, 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,IPRGNA
C
      DIMENSION IDEV(2)
C
      LOGICAL SNGLBS,BATCH,ENDE,NLSTCF,DIALOG,PTY,TTY,LPTOPN,CTROPN
     1,LOCKUP,LAPEND,LIMTTY,LLPT
C
      REAL LPTPPN
C
      EQUIVALENCE (IDEV(1),JOBDEV)
C
      COMMON /TTYOUT/ LIMTTY,JCTWD,LLPT
      COMMON /GET/ UT
      COMMON /INPCOM/ SIMODE,PNAME,WORD,DUMMY(124),FSPECS(8)
      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(IPRGNA)
C
C     VSMCTR(HEVSIM CONTROL) DOES DISK PROGRAM MODE & FILE CONTROL ONLY
C
C    INTIALIZE
C
      UT=BLANK                                                  !FLG ZERO ALL.
```

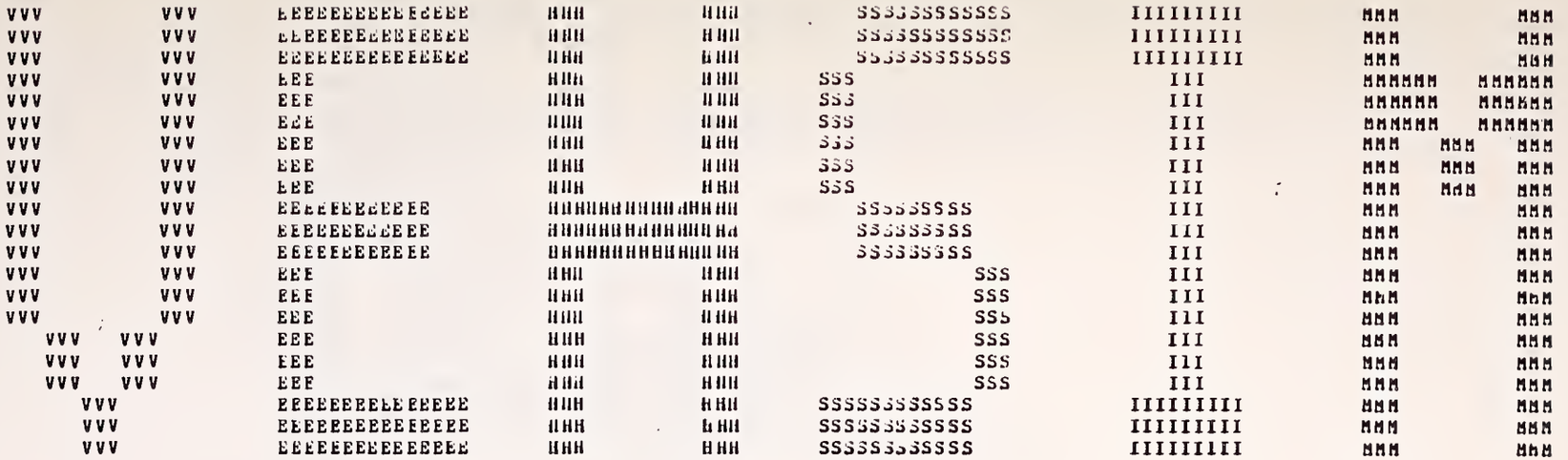
```
      CALL ZERCMD(IECOND)                                  !INITIALIZE HEVSIM.

      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 JOBPPN(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) HZERO                           !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.
      LPTORG=LPTFIL
      LPTPPN(1)=PPNJOB(1)
      LPTPPN(2)=PPNJOB(2)
      LPTOPN=.FALSE.
      LPTDSP=DBLANK
C
      CTRDEV=JOBDEV                                        !SET DEFAULT BATCH CONTROL FILE STRUCTURE.
      CTRPPN(1)=PPNJOB(1)
      CTRPPN(2)=PPNJOB(2)
      CTROPN=.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".
```

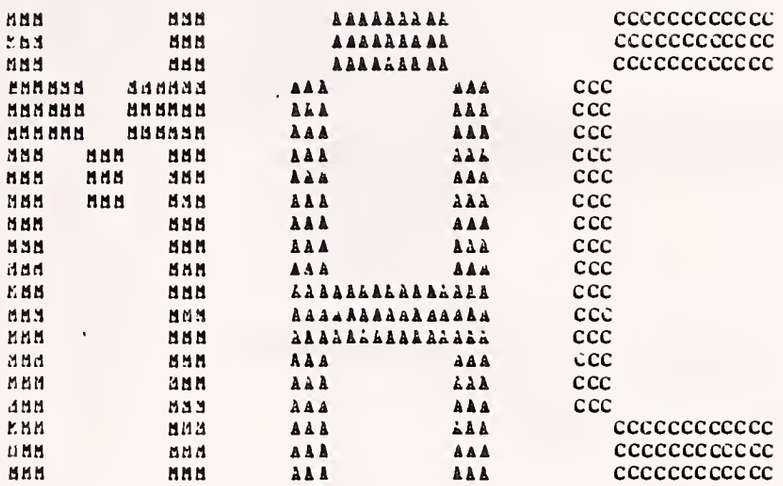
```
      WRITE(5,1000)IPRGNA, VERID,DATE,HOUR,JOBNUM,TTYNUM,JOBDEV,PPNJOB      !REPORT JOB INFO TO TTY.
C
      CALL ERRSET(0)
C PROMPT FOR COMMAND INPUT
  100 IF(DIALOG)GO TO 110
      IF(TTY) WRITE(5,1007) BELL                                            !(TTY?) YES.
  101 WRITE(5,1001)                                                         !PROMPT "*"
      READ (5,1002) COMND,FSPECS                                            !READ COMMAND.
      IF((COMND.AND."774000000000) .NE. ('@'.AND."774000000000))
      2GO TO 103
      REREAD 1030,FSPECS
      GO TO 117
C
103   FILNAM=FSPECS(1)
      DLSTCF=FSPECS(2)
C
      IF(COMND.EQ.BLANK) GO TO 101                                          !(EXTRANEOUS "CR"?) YES, GO PROMPT.

      IF(COMND.EQ.'BATCH') GO TO 115                                        !(BATCH?) YES.
      IF(COMND.EQ.'DIALO') GO TO 110                                        !(DIALOGUE?) YES.
      IF(COMND.EQ.'STOP' .OR. COMND.EQ.'EXIT') CALL EXIT                    !(STOP OR EXIT HEVSIM RUN COMPLETE?) YES.

      IF(COMND.EQ.'CONTI') GO TO 105                                        !(CONTINUE EXECUTION OF CONTROL FILE?) YES.

      WRITE(5,1006) COMND                                                   !?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.IECOND.EQ.3) GO TO 210                                  !(DIALOG &       ?).
      GO TO 100                                                             !GO PROMPT.

  107 MCTR=3                                                                !FLG INPBAT FOR *CONTINUE.

  109 BATCH=.TRUE.                                                          !SET BATCH MODE.

      DIALOG=.FALSE.
      IF(MCTR.EQ.3)GO TO 200                                                !*CONTINUE COMMAND
      CALL FILSPC(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                                     !(HEVSIM 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 HEVSIM CONTROL FILE IS OPEN.
C
  215   IF(CTROPN) GO TO 218                                    !(HEVSIM CONTROL FILE OPEN?) YES, SKIP.

        IF(MCTR.NE.2) CTRFIL=FILNAM                             !(*REWIND?) NO, GET HEVSIM CTR FIL NAME TO OPEN.

        OPEN(UNIT=4,DEVICE=CTRDEV,ACCESS=SEQIN,FILE=CTRFIL      !OPN HEVSIM CNTRL FIL.
        1,DIRECTORY=CTRPPN)                                     !EFFECTIVE AS REWIND IF SAME FIL NAME.

        CTROPN=.TRUE.                                           !FLG HEVSIM CONTROL FILE OPEN.
```

```
      NLSTCF=.FALSE.                                    !ASSUME LIST CONTROL FILE ON LPT.

      IF(NLSTCF.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 HEVSIM CONTROL FILE AS REQUESTED.

      IF(.NOT. ENDE) GO TO 225                          !(EOF ON CTR FIL?) NO.

      CLOSE(UNIT=4)                                     !RELEASE CONTROL FILE.

      CTRCFN=.FALSE.                                    !SET FLG CONTROL FILE NOT OPEN.

C
  225 IF(DIALOG) GO TO 210                              !(DIALOGUE COMMAND FROM CTR FIL?) YES, GO INPDIA.

      IF(.NOT.CTROPN)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.CTROPN) GO TO 320                         !(CTR FIL OPEN?) NO.
      CALL RELEAS(4)                                    !YES, RELEASE IT.

      CTROPN=.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(PTY) 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..ENAME) GO TO 325                    !(RENAME LPT FIL?) NO. GO CLOSE.

  327 WRITE(5,1012)                                     !LPT FIL RENAME=?.

      READ(5,1011) FSPECS                               !GET ANS.
      IF(FSPECS(1).EQ.DBLANK) GO TO 327                 !(ANY ANS?) NO, GO ASK AGAIN.
```

```
      CALL FILSPC(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=LPTPRO,ERR=3250)
      GO TO 3255
3250  CALL ERRSNS(ITEMP,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.
      LPTOPN=.FALSE.                                                  !SET FILG LPT CLOSED.

C     IF(.NOT.ITASK)LPTFIL=LPTCPG                                     !RESET LPT FIL NAME.
      LPTDSP=DBLANK                                                   !BE INITIALIZE

      LPTPPN(1)=PPNJOB(1)
      LPTPPN(2)=PPNJOB(2)
  330 IF(.NOT.ITASK)GO TO 332
      ITASK=0
      RETURN
C
332   CALL DSKCTR(0,'VSMCTR')                                         !FLG DSKCTR TO CLOSE ALL IT'S FILES.

      RETURN                                                          !*DONE ,BYE.

C
C*****************************************************************************
C
C   FORMAT STATEMENTS
C
 1000 FORMAT(1X,A6,1XA5,1XO2'('O2') 'A10,2XA5' JOB#'I3' TTY'O3
      1,2XA6':['O5,','O3']')
 1001 FORMAT(' *'$)
 1002 FORMAT(A5,1X10A10)
 1003 FORMAT(I3)
 1004 FORMAT('1 HEVSIM CONTROL FILE 'A6':'A10'['O5','O3']'/)
 1005 FORMAT(1A3)
 1006 FORMAT(' ?'A5'?')
 1007 FORMAT(1XA1$)
 1008 FORMAT(/' ? VSMCTR-NO HEVSIM 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(' ? HEVSIM CONTROL FILE 'A6':'A10'['O5','O3'] NOT FOUND'/
      2,      ' SWITCHING TO DIALOGUE MODE')
1030  FORMAT(1X,10A10)
1031  FORMAT(10A10)
1035  FORMAT(' ?VSMCTR - CANNOT RENAME OUTPUT FILE TO '
      2,A5,':',A10,'['O4,',',O3,']')
      END
```





START User SOMERS [3224,633] Job VENSIM Seq. 7149 Date 30-Dec-80 11:19:56 Monitor DECsystem10/KL v7 *START*
File: DSKB:VENSIM.MAC<055>[3224,633] Created: 17-Nov-80 14:43:31 Printed: 30-Dec-80 11:21:09
QUEUE Switches: /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:71 /FORMS:UNLNED

```
        TITLE VEHSIM
;
;*****************************************************************
;
;       VEHSIM PROGRAM  ===  MAY , 1976  ( BATCH/INTERACTIVE VERSION )
;                            ORIGINATOR - M.GOULD/E.WITHJACK,DOT-FSC
;
;                            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:      GOBACK, INPBAT, INPDIA, LOOKUP, SIMCTR,
;                       SIMINT, SIMSTS, VSMCTR
;
;       RUNS UNDER MACRO-10/FORTRAN-10/LINK-10
;
;*****************************************************************
;
;
        VERMAJ=700
        VERMIN=24
        .JBSA==120
        .JBREN==124
        .JBOPC==130
        ENTRY VEHSIM,EXIT,RESET,RESETM,TAMPER
        EXTERNAL RESET.,VSMCTR,CLOSE,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
?
? VSMdaC - BAD PUSHDOWN STACK

*EXIT VIA VEHSIM/FOROTS

 ]      ;BAD PUSHDOWN STACK PROBABLY DUE TO VEHSIM MISHANDLING NO. FOROTS
TAMPER: SETZM 0,.JBSA   ;PREVENT RESTART OF VEHSIM
```

```
        SETZM 0,.JBREN
        SETZM 0,.JBOPC
        JRST TEXIT.         ;BYE
;
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         ;FORCES TERMINATION OF PROGRAM VEHSIM
        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 FORDTS
        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   /TRUCK/
;
;
        PRGEND VEHSIM
```

```
        TITLE   BLOCKT
SUBTTL   BLOCK TRANSFER SUBROUTINE


;
;       USAGE           CALL BLOCKT(ARRAY1,ARRAY2,NWORDS)
;               WHERE   ARRAY1: IS ARRAY (VECTOR) TO BE TRANSFERED
;
;                       ARRAY2: IS ARRAY (VECTOR) TO TRANSFER ARRAY1 TO
;                       NWORDS: IS THE NUMBER OF WORDS TO TRANSFER
;


        ENTRY   BLOCKT
BLOCKT: MOVSI   0,00(16)        ;PICK UP STARTING ADDRESS
        HRRI    0,01(16)        ;PICK UP DESTINATION ADDRESS
        HRRZ    1,0             ;COPY
        ADD     1,02(16)        ;ADD LENGTH
        BLT     0,-1(1)         ;TRANSFER. LIMIT =C(1)-1
        POPJ    17,
        PRGEND
```

```
        TITLE   CRLF
        ENTRY   CRLF
CRLF:   CAIA
        PUSH    17,CEXIT.##
        OUTCHR  XWD[0,,15]
        OUTCHR  XWD[0,,12]
        POPJ    17,
        PRGEND
```

```
        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,$OKLABEL), OR
;       CALL CHKFIL(DEV,FILE,PPN,$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.
;               $OKLABEL=       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##
        JUMPG   .+4
        OUTSTR  [ASCIZ/
?CHANEL - no free channels)
/]
        EXIT    1,
        EXIT
        LSH     5
        HRLZM   CHNO.
        ARRAY   CHNO.[1]>
;;
;;--------------------------------------------------------
;;
        DEFINE  FERMES(STRING)<
        JRST    [       OUTSTR  [ASCIZ/
?STRING
/]
        EXIT    1,
        EXIT    ]
```

```
>
;;
;;-------------------------------------------------------------
CHKFIL: CHANEL                          ;GET A CHANNEL
        MOVE    1,@(16)                 ;GET DEVICE
        JUMEN   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
          FERMES        (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 #
        JUMEN   4,GOTPPN                ;GOT ONE?
        MOVE    11,[-1,,.PTFLD]         ;NO GET PATH
        MOVE    10,[3,,11]
        PATH.   10,
          SETZM 11+.PIPPN               ;ERROR - ZERO TO GET DEF DIR.
        MOVE    4,11+.PIPPN             ;GET IT
GOTPPN: SETZM   3
        MOVE    CHNO.
        IOR     [LOOKUP 0,1]
        XCT
        JRST    TRYLIB                   ;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    CHNO.
        IOR     [LOOKUP 0,1]
        XCT
          JRST  ERROR
        MOVEI   1,@2(16)
        HLRZ    13
        MOVEM   (1)
        HRR     13
        MOVEM   1(1)
        SETOM   @5(16)
        JRST    OKRET

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 OUR LINE
        GETTCH  (17)            ;CHARACTERISTICS
        SETTCH  [2]             ;SET TTY NORMAL!!
```

```
^PUSH:  JRST    STOP            ;BRANCH TO SUBROUTINE
        SETICH  (17)            ;RESTORE LINE CHARACTER AS BEFORE
        POP     17,(17)         ;FIX STACK
        POPJ    17,             ;RETURN IF CONTINUE
;
SETC:   MOVE    (16)            ;GET ADDR OF SUBROUTINE
        HRRM    ^PUSH           ;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 FILSPC(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 FILE
;                         DIRECTORY.  MUST BE DIMENSIONED TO 3
;                         IF NO SFD'S PRESENT.  IF SFD'S ARE PRESENT,
;                         MUST BE DIMENSIONED TO 5.  IF NO DIRECTORY
;                         IS GIVEN, DEFAULT OF PPN FOR JOB IS RETURNED.
;
;       ENTRY POINTS: FILSPC
;
;       CALLED BY: VSMCTR
;
;       ***************************************************************
;
        SUBTTL      WRITTEN BY SID SHAPIRO/KHL  12/20/76
        ENTRY   FILSPC
FILSPC: MOVE    [XWD ZBUFF,NCHAR]
        BLT     DEVFLG
        MOVE    [XWD SAVPTR,LOOPTR]
        BLT     LOOPTR+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     9,              ;INC CHAR COUNTER
        CAIN    72              ;IS IT A COLON?
        SETOM   DEVFLG          ;YES, SET DEVICE FLAG
        CAIN    133             ;IS IT A LEFT BRACKET?
        SETOM   DIRFLG          ;YES, SET DIRECTORY FLAG
        CAIE    135             ;IS IT A RIGHT BRACKET?
        JRST    GETBYT          ;NO, GET ANOTHER BYTE
```

```
DONE:   MOVEM   9,NCHAR
        MOVE    10,NCHAR        ;GET NUMBER OF CHARS
        AOJ     10,
        MOVE    SBFPTR

        MOVEM   BUFPTR
        SKIPL   DEVFLG          ;DEVICE?

        JRST    DONDEV          ;NO. DO FILE
DODEV:  SOJE    10,DONDEV
        ILDB    BUFPTR          ;GET BYTE
        CAIN    72              ;IS IT COLON
        JRST    DONDEV          ;YES, DONE
        IDPB    DEVPTR          ;PUT IT AWAY
        JRST    DODEV
DONDEV: MOVE    DEV             ;GET DEVICE
        MOVEM   01(16)          ;RETURN IT
        SETCM   DEFEXT          ;ASSUME USE DEFAULT EXTENSION
DOFIL:  SOJE    10,DONFIL
        ILDB    BUFPTR          ;GET BYTE
        CAIN    133             ;IS IT LEFT BRACKET?
          JRST  DONFIL          ;YES, DONE
        CAIN    "."             ;GOT A PERIOD?
          SETZM DEFEXT          ;YUP, CAN'T USE DEFAULT EXT.
        IDPB    FILPTR          ;PUT IT AWAY
        JRST    DOFIL
DONFIL: SKIPL   DEFEXT          ;USE DEFAULT?
          JRST  DNFL3           ;NO
        MOVE    14,[ASCIZ/.VSM/]        ;GET DEFAULT
        MOVE    15,[POINT 7,14] ;GET POINTER
DNFL2:  ILDB    15
        JUMPE   DNFL3           ;GOT NULL, DONE WITH DEFAULT
        IDPB    FILPTR          ;PUT IT AWAY
        JRST    DNFL2
DNFL3:  DMOVE   FILE            ;GET FILE NAME
        DMOVEM  02(16)          ;PUT IT AWAY
        SKIPL   DIRFLG          ;IS THERE A DIRECTORY?
        JRST    DEFPPN          ;NO DO DEFAULT PPN
        JRST    SETPPN          ; YES, DO PPN STUFF
NXTWRD: MOVEM   7,05            ;PUT PPN AWAY
        AOJ     5,              ;INC PPN WORD ADDR
        SETZ    7,              ;ZERO OUT 7
        JRST    DOPPN
SETPPN: MOVE    5,3(16)         ;GET ADDR OF PPN RETURN
        SETZ    7,
DOPPN:  SOJE    10,DONPPN
        ILDB    BUFPTR          ;GET BYTE
        CAIN    54              ;IS IT A COMMA
        JRST    NXTWRD          ;YES, GO GET ANOTHER WORD
        CAIN    135             ;IS IT RIGHT BRAKET
        JRST    DONPPN          ;YES, DONE
        SUBI    60              ;CONVERT TO OCTAL
        IMULI   7,10            ;SHIFTRECEIVING WORD
        ADD     7,0             ;ADD IN NEW DIGIT
        JRST    DOPPN
DONPPN: MOVEM   7,05            ;PUT AWAY LAST WORD
        POPJ    17,
;
DEFPPN: GETPPN
        JFCL    0,0             ;NO-OP
        MOVE    5,3(16)         ;GET ADDR OF RETURN PPN ARG
```

```
        HLRM    @5              ;PUT PROJ NUMBER AWAY
        ADJ     5,
        HRRM    @5              ;PUT PROG NUMBER AWAY
        POPJ    17,

        HALT
        PAGE
SAVPTR:SBFPTR:  POINT   7,BUFF
SDVPTR: POINT   7,DEV
SFLPTR: POINT   7,FILE
SDEV:   422471,,300000
        Z
;
LODPTR:BUFPTR:  Z
DEVPTR: Z
FILPTR: 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)        ;BLANK WORD
        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" PACKED 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  0(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, JOBPPN, 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,PCINI(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,PSAVE          ;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,PSAVE            ;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/KMI
;
;       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:   DSKDIR, JOBPPN
;
;       SUBROUTINES CALLED:     SEVBIT
;
;       CALLED BY:      VSMCTR
;
;*****************************************************************
;
        ENTRY JOBPPN,DSKSTR
JOBPPN: PJOB 1,         ;UUO TO RETURN 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 /LST/]
        JRST RETURN
```

```
;
AC16:   2
        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:      KPTIME
;
;*****************************************************************
;
;
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
ARDSIX=1        ;STARTING ADDRESS OF SIXBIT WORD(S)
ARDSEV=2        ;STARTING ADDRESS OF ASCII(7 BIT) WORD(S)
COUNT=3         ;NUMBER OF INPUT BYTES
INPB=4          ;BYTE POINTER FOR INPUT
OUTPB=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
```

```
;
ACIo:   Z
;
        -5,,0
PTRSIX: @0(15)
        @1(15)
        0,,IEMP
        0,,IEMP+1
        [0,,6]
;
        -5,,0
PTRSEV: @2(15)
        @3(15)
        0,,IEMP
        0,,IEMP+1
        [0,,5]
;
IEMP:   BLOCK 2
        PAGEND
```

```
        TITLE TTYINP
;
;       FORTRAN-10 CALLABLE ROUTINE TO ALLOW A FORTRAN-10
;           PROGRAM AN INTERRUPT STRUCTURE WITH LIMITATIONS
;
;
;       CALLED BY:      GOBACK, INPDIA, SIMINT, SIMLPT, SIMSIS
;
        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 INTERRUPT
        MOVEI 16,ARGBLK ; GET POINTER TO INTERRUPT CHARACTER
        PUSHJ 17,@0(15) ;CALL INTERRUPT HANDLER
        MOVE 16,AC16    ;PROGRAM INTERRUPT COMPLETED! RESTORE AC 16
;
TTYCLR: CLRBFI          ;CLEAR INPUT BUFFER
        POPJ 17,0       ;RETURN
;
TTYSET: INCHRS  .CHAR   ;BUFFER EMPTY? ALSO NULLIFIES -0 !
        POPJ 17,0       ; YES ! RETURN.
        CLRBFI          ; NO! CLEAR BUFFER
        POPJ 17,0       ; RETURN
;
;       ARGUMENT BLOCKS
;
AC16:   Z
        -1,,0
ARGBLK: .+1
.CHAR:  Z
        END
```

```
        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
;
```

```
ADRWID: .IOWID
        Z
:

        PRGEND
```

```
        TITLE   VALID3
        ENTRY   VALID3
VALID3: MOVE    SIPTR
        MOVEM   IPTR
        MOVE    SOPTR
        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
        SETO    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
        SETO    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
SOPTR:  POINT   6,0,35
;
IPTR:   Z
OPTR:   Z
;
        PRGEND
```

```
        TITLE   ZEROP
        SUBTTL PROGRAM TO ZERO ARRAY.
        SEARCH  FORPRM

COMMENT %

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

%

        A=1
        N=2

        HELLO   (ZEROP, )       ;ZEROP 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
```

APPENDIX D

Updated Nomenclature

Updated Nomenclature

AASE - required acceleration for a constant acceleration segment

ACCEL - the current acceleration at the wheels

ACCS - accessory speed table array

ACCSR - accessory array of speed ratios

ACOM - accessory comment array

ADSE - terminal distance specified as a relative segment end point

AIAS - accessory input inertia array

AIGIN - gear input inertia

AIGOUT - gear output inertia

AIP - propshaft inertia

AIW - wheel inertia

AI1 - torque converter pump inertia

AI2 - torque converter turbine inertia

AK - instantaneous torque converter capacity factor used in converter routine

AKC - coast converter input speed array

AKD - drive converter capacity factor array

ANAME - accessory name

APCS - passing clearance defining the current segment end

APOS - absolute milepost defining the current segment end

APWO - constant percent wide open throttle required for entire segment

AREA - vehicle frontal area

ARRIVE - logical flag indicating arrival at a request

ASEG - array of constant accelerations required by driving segments

ATHOLD - array of arrival accelerations for accelerate to and hold throttle driving segments

ATHR - maximum 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 the current driving segment

AXCOM - array of axle comment

BORE - engine bore size

BPER - current slope in percent of grade being used

BVG - bevel gear ratio;

BVGEFF - bevel gear efficiency

CAC - accessory loss accumulator

CAE - aerodynamic drag loss accumulator

CBR - brake loss accumulator

CCOM - torque converter comment array

CD - vehicle drag coefficient

CDA - distance driven during acceleration

CDC - drag coefficient wind angle sensitivity factor

CDCR - distance driven during cruise

CDD - distance driven during deceleration

CDI - distance driven during idle

CDIAM - torque converter diameter

CEA - engine energy used during acceleration

CECR - engine energy used during cruise

CED - engine energy used during deceleration

CEI - engine energy used during idle

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

CNAME - converter name

COAST - logical flag indicating drive or coast condition to select converter

CONTOR - constant input torque - torque converter

CTA - time spent in acceleration

CTCR - time spent in cruise

CTD - time spent in deceleration

CTI - time spent in idle

CTR - torque converter loss accumulator

CUMD - cumulative distance travelled

CUMEN - cumulative engine energy expended

CUMENM - cumulative energy lost in return to engine during coast

CUMFE - cumulative fuel economy (MPG)

COMFU - cumulative weight of fuel consumed (LBS)

CUMG - cumulative gallons of fuel consumed

CUMT - cumulative time elapsed

CYL - number of cylinders in the engine (also, ICYL)

D - distance travelled in feet during current time step

DACC - variable used to alter acceleration of vehicle during an iteration

DCOM - driving schedule comment array

DDIS - distance travelled in miles during current time step

DEFDT - the default time step to be used

DEN - incremental engine energy produced during time step

DETPT - detent override shift point array

DETRPM - detent override shift speed point

DFU - incremental fuel consumed during time step

DHR - length of time step in hours

DISP - engine displacement (CID)

DNAME - driving schedule name

DNUMG - number of gears in shift logic

DRPMW - delta rpm wheels

DRPME - delta rpm engine

DSAVE - distance travelled during a gear shift step

DSEG - array of relative distance end prints for the driving segment

DSTART - start absolute distance at the beginning of a driving segment

DT - length of time step in seconds

ECOM - engine comment array

EINER - engine inertia

EMAP - engine map containing torques, fuel rates, manifold vacuums, and throttle settings for both engines

ENAME - engine name

ENDLIM - logical flag indicating print limited to last time step in each driving segment

ENDRSG - logical flag indicating the end of a route segment

ENDRTE - logical flag indicating the end of an entire route

ENDSEG - logical flag indicating the end of a driving cycle segment

ENMIN - minimum engine speed as input

ERAR - array of efficiencies of the rear axles.

ERAT - array of gear efficiencies

ERPM - engine map speed point array

FACCEL - force at the wheels dur to acceleration

FAERO - force at the wheels due to aerodynamic drag

FGRADE - force at the wheels due road grade

FRATE - fuel flow rate for current time step

FRC1
FRC2 - tire rolling resistance

FRC4 - wheel bearing friction

FROLL - force at the wheels due to rolling resistance

FSPGR - fuel specific gravity

FWHEEL - total force at wheels after tire slip accounted for

GCOM - gear comment array

GEANAM - array of gear names

GEANUM - array of gear numbers

GRAT - array of gear ratios

GRTORG - array of gear torque values for spin losses

HIPNAM - name tag of debug routine

HPA - horsepower lost to accessories

HPBR - horsepower lost to brakes

HPE - horsepower produced by engine

HPMI - total horsepower hours/mile for entire driving cycle

HPP - horsepower between differential and gear box

HPW - horsepower delivered at the wheel

HP1 - horsepower on bevel gear side of torque converter

HP2 - horsepower on gearbox side of torque converter

IAF - array of axle numbers to shift "from"

IAT - array of axle numbers to shift "to"

IENG - index indicating which engine being used

IERRE - error flag in engine routine

IGF - array of gear numbers to shift "from"

IGT - array of gear numbers to shift "to"

IPRNT - index indicating what component to print or use in disk routine

IMAX - maximum engine speeds on engine map

IMIN - minimum engine speeds on engine map

ISEG - the current segment number being executed

ITS - the current segment type being executed

ITYSEG - array of segment types for driving cycles

JENG - array of engine numbers, one assigned for each gear

LBRAKE - logical flag indicating whether or not brake is applied

LDIES - logical flag indicating diesel engine

LIMPRN - logical flag indicating whether or not limited print is desired

LLSH - logical flag indicating whether to hold a constant gear or examine shift logic during current driving segment

LRPM

LSCALE - logical flag indicating whether engine needs to be scaled

LSHFT - logical flag indicating whether or not a gear shift is required

LSTRTE - logical flag indicating last route segment

LWOT - logical flag indicating whether or not to return only maximum and minimum throttle settings from engine routine

MAPOX - integer flag indicating whether or not a gear shift is required

MILIM - logical flag indicating a limited printout every so many miles

NACC - number of accessories being used

NAXS - number of rear axle speeds

NCYCLE - number of cycles of the engine type being used

NGEAR - number of gear being used during current time step

NGOCAL - number of iterations through GOBACK routine

NGRLSS - array containing the number of points for gear spin losses for each gear.

NGSEG - array of constant gears used during driving cycle (if any)

NGT -

NGTR - number of gears

NNA - array containing the number of data prints for all accessories

NNGS - constant gear setting for a driving segment

NRAX - number of rear axles

NRDIST - total number of route segments in the route being used

NRPM - array containing number of speed points in each engine map

NRTSEG - current number of the route segment being executed

NSEG - total number of segments in the driving cycle

NSPTS - array containing number of data points in each shift line

NTBP - index of the data values where the torque converter break point exists

NTOR - array containing the number of torque points for each speed point on the engine map

NTORP -

NUMBSL - number of gear shift lines

NUMG - total number of gears being used by the transmission

PCSEG - array containing passing clearances (if any) for driving cycle segments

PCTHR - percent engine wide-open throttle

PCTWOT - percent wide-open throttle

POSTSEG - array of absolute distance mileposts (if any) for ending driving cycle segments

PPHPHR - total fuel lbs./HP-hr. used during the entire driving cycle

PWOT - array of constant parent WOT - driving schedule

RAR - array containing rear axle ratios

RCOEF - array of road coefficients for the route segments

RCOM - route comment array

RDIST - array of lengths for the route segments

RDLD - road load

RERAT - array of shift curve values

RGRADE - array of grades for the route segments

RNAME - route name

ROADC - current road coefficient used during time step

RPMAX - maximum engine speed given by map

RPME - engine speed during current time step

RPMED - engine speed during previous time step

RPMIN - minimum engine speed given by map

RPMP - propshaft speed

RPMW - wheel speed during current time step

RPMWO - wheel speed during previous time step

RPM1 - speed on engine side of torque converter

RPMW - wheel speed during current time step

RPMWO - wheel speed during previous time step

RPM1 - speed on engine side of torque converter

RPM2 - speed on gear box side of torque converter

RVWIND - array of wind values for route segment

SATH - arrival acceleration for an accelerate and hold throttle driving segment

SCOM - shift logic comment array

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

SHFTIME - array of shift times for shift lines

SHFTPT - array of shift points for shift lines

SHFTRPM - array of shift speeds for shift lines

SNAME - shift logic name

SOUT - array of output speeds for torque converter

SPIDLE - engine idle speed

SPIN - array of input speeds for torque converter

SR - speed ratio for current time step

SRC - array of speed ratios for the coast converter

SRD - array of speed ratios for the drive converter

STIME - shift time during the current shift being performed

STROKE - engine stroke size

T - end elapsed time in seconds of the current driving segment

TCOM - tire comment array

TEND - relative end time of current time step if a constant velocity driving segment is being executed

THR - throttle setting in degrees for this time step

THRATE - array of maximum rates of change of throttle settings for driving segments

THRMAX - maximum throttle setting on engine map

THRMIN - minimum throttle setting on engine map

TIN - array of input torques for torque converter

TIREFF - tire efficiency

TITLE - run title input by user

TMIN - minimum throttle setting for current time step engine speed

TNAME - transmission gear name

TOLD - elapsed time at beginning of current time step

TORBPK - the capacity factor value at the drive converter torque break point

TORQA - total torque needed by the accessories during current time step

TORQE - required engine torque during current time step

TORQF - torque of front end

TORQP - propshaft torque during current time step

TORQ1 - torque on engine side of torque converter during current time step

TORQ2 - torque on gear box side of torque converter during current time step

TOUT - array of output torques for torque converter

TR - torque ratio used during current time step

TRCOM - transmission data comment array

TRD - array of torque ratios for the drive converter

TRR - torque at rear end

TSAVE - time spent initializing current driving segment

TSEG - array of relative end point times for driving segments

TSTART - relative start time of current time step

TWOT - wide open throttle torque for current engine speed setting

TYPE - name of command read in input routine

UN - component name on input data card

UT - component type on input data card

V - vehicle velocity at end of current time step

VAC - engine manifold vacuum during current time step

VARNAME - name of single variable value to be modified without reading a new component from the disk

VAVG - average vehicle velocity over the entire driving cycle

VCOM - vehicle comment array

VELSEG - array of constant velocities to be used for driving segments

VNAME - vehicle name

VOLD - absolute velocity at the beginning of the current time step

VSEG - array of velocity end points for the driving segment

VSTART - relative velocity since beginning of current time step

VWIND - absolute wind velocity

WGT - vehicle weight (lbs)

WLSG - number of wheels

WRAD - wheel radius

APPENDIX E

Control Files

```
.RUN BVG2
*BATCH TEST.SEV
@DRIVE.ONE
@SIMULA.ONE
@DRIVE.TWO
@SIMULA.ONE
@DRIVE.THR
@SIMULA.ONE
@DRIVE.FOU
@SIMULA.ONE
@DRIVE.FIV
@SIMULA.ONE
@DRIVE.SIX
@SIMULA.ONE
@RESET.ONE
.IF(ERROR) .GOTO A
.GOTO B
A::.CLOSE
B::
.PRINT/DIS:DEL BUSBVG.DAT[3,3]
.PRINT/DIS:DEL NEWFIL.ICG
.KJOB
```

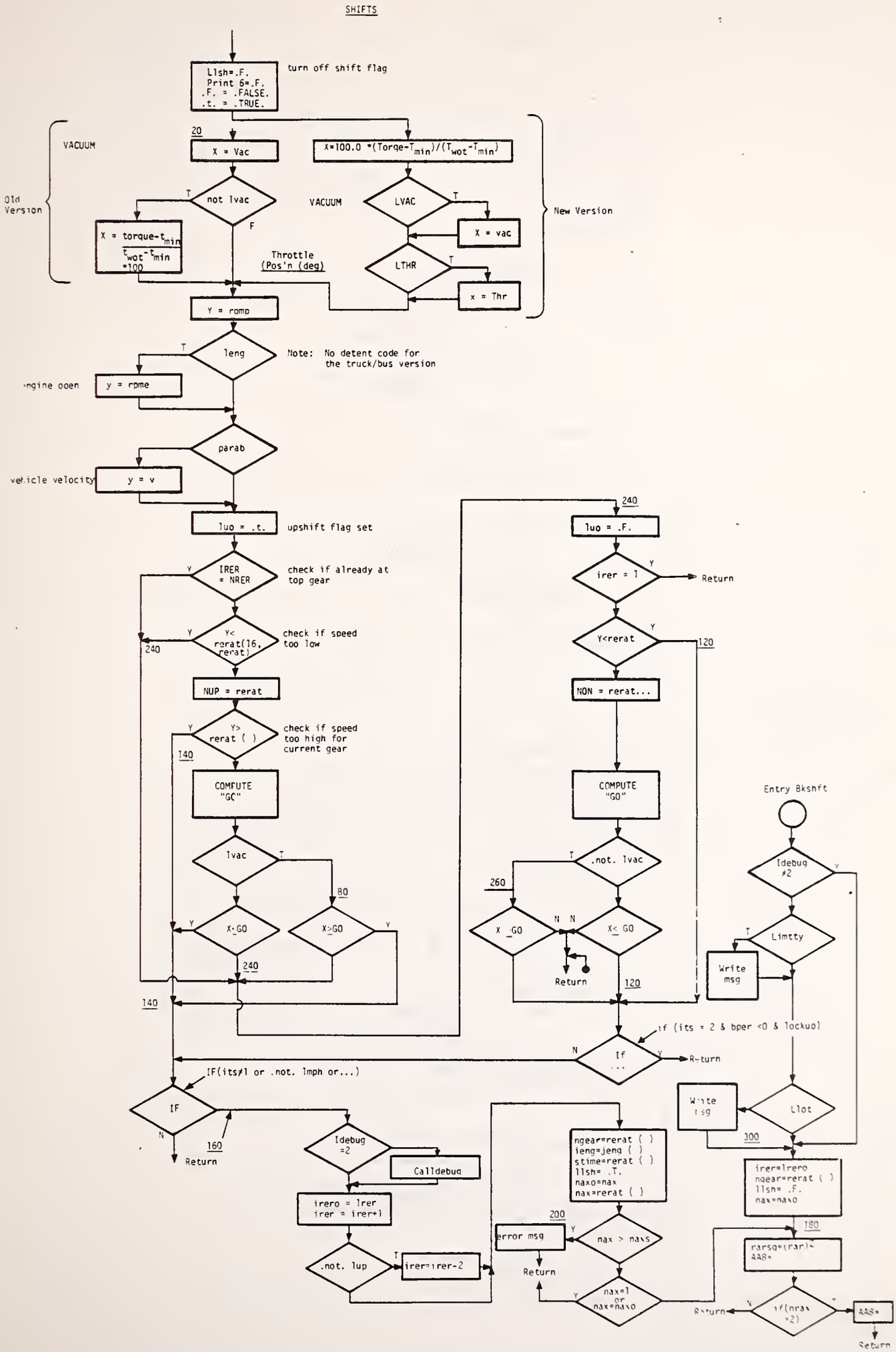
TYPICAL CONTROL FILE FOR PRODUCTION RUNS

```
#OUTPUT      HEVSIM.DAT[3,3]
#LIMIT PRINT       OFF
#TITLE       TEST CASES OF TRUCKS AND BUSES
#USE   871N60      ENGINE          1        1 2 3 4
#USE   TRABUSCBDA  DRIVING SCH
#USE   BUS 1       ACCESSORY
#USE   BUS-2       AXLE
#USE   V730-ST     SHIFT LOGIC
#USE   ZGRADE      ROUTE
#USE   BUS 1       TIRE
#USE   V-730B      TRANSMISSION
#USE   BUS-1       VEHICLE
#USE   TC-490      CONVERTER
#USE   TC-490-C    CONVERTER
#UNLOCK CONVERTER GEAR          1 2
#LOCKUP CONVERTER GEAR          3 4
#LIMIT PRINT      SEC          0.05
#MODIFY           DUTYCL       0.50        BUS 1
#MODIFY           WEIGHT       35000.
#MODIFY           REAR         5.35
#SIMULATE         BUS
```

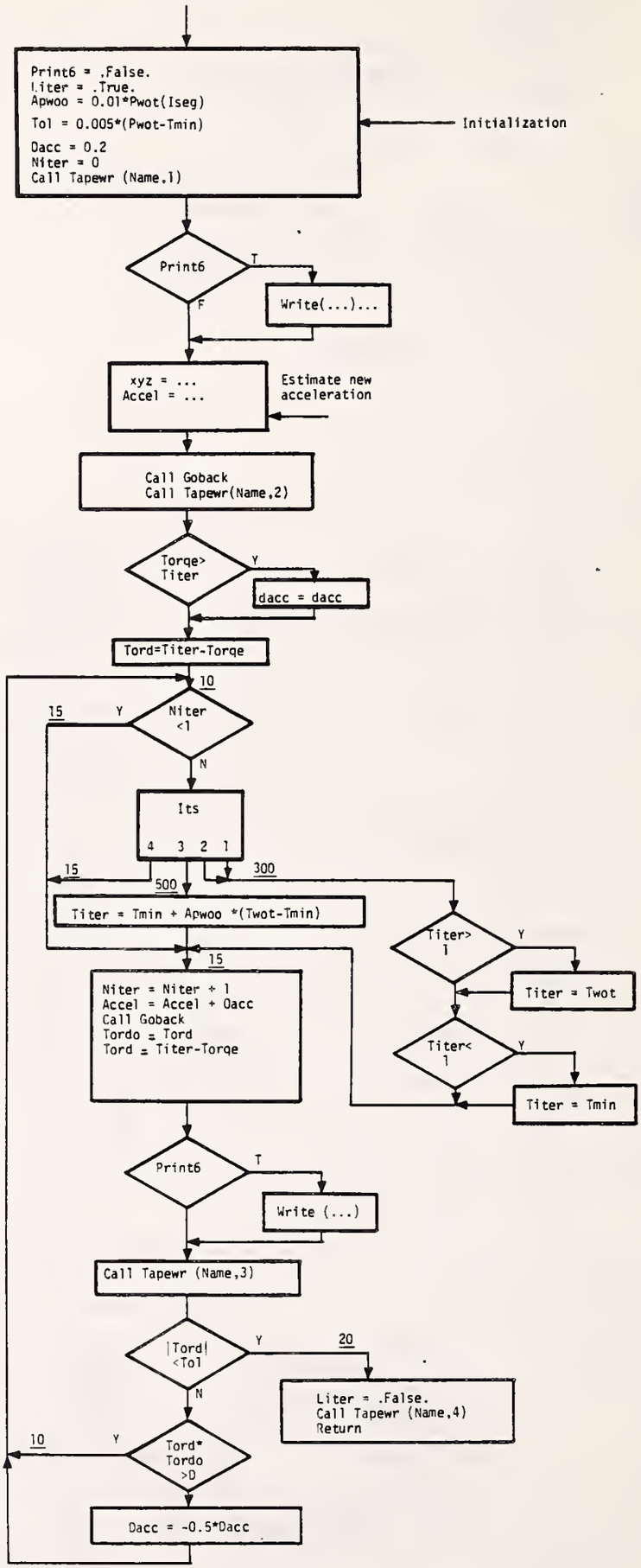
TYPICAL CONTROL PARTS FILE FOR A BUS

APPENDIX F

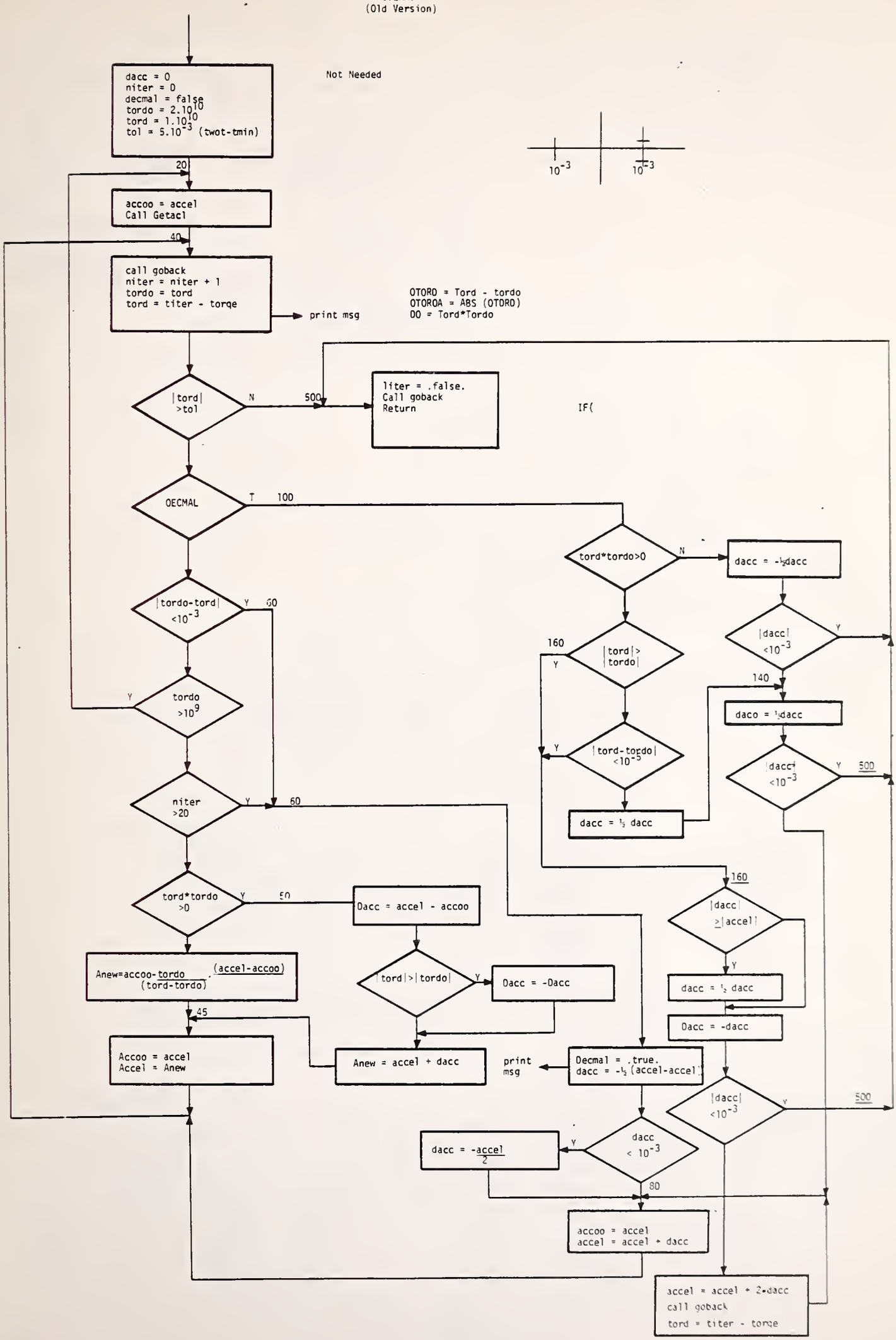
Selected Flow Charts

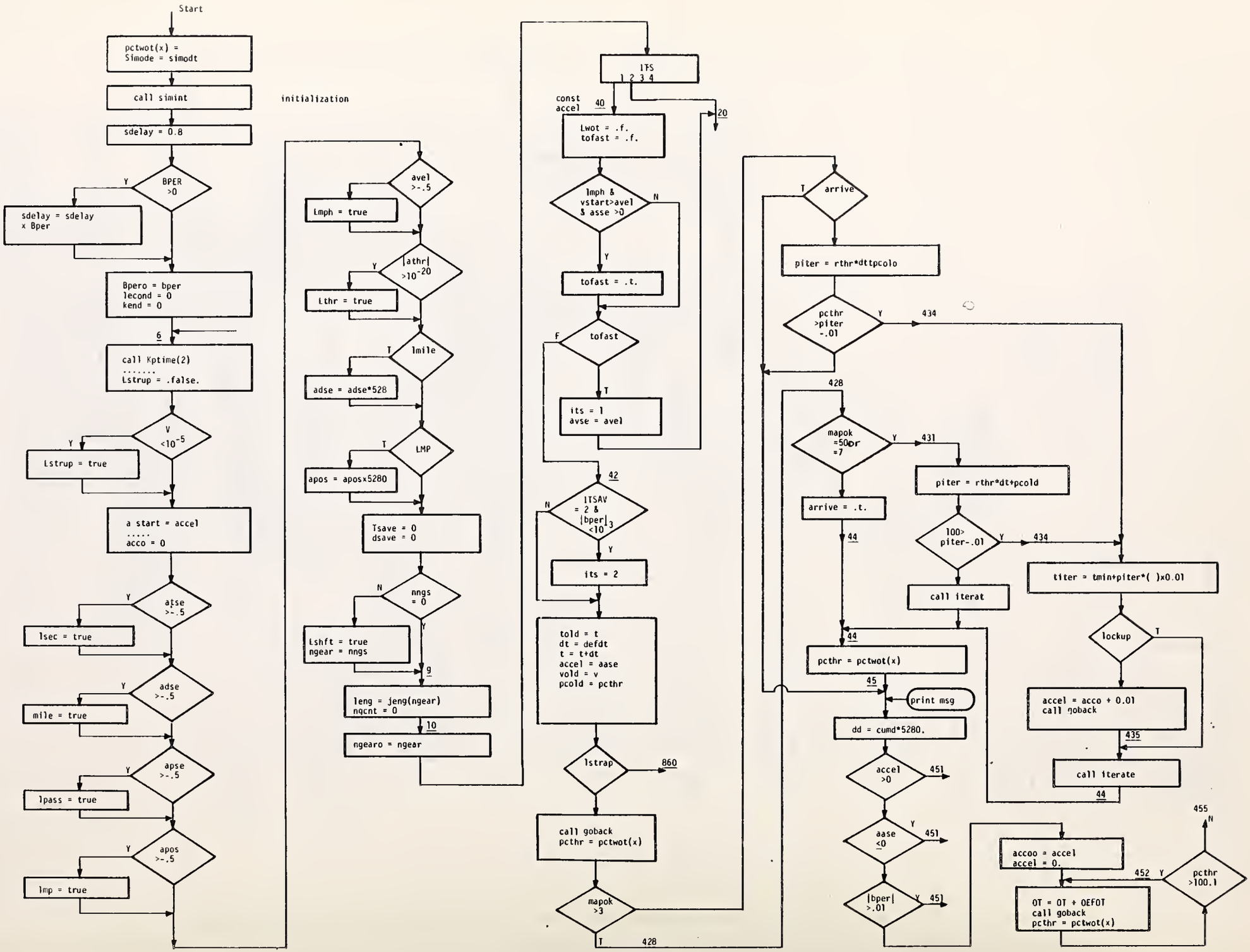

SHIFTS
llsh=.F.
Print 6=.F.
.F. = .FALSE.
.t. = .TRUE.
turn off shift flag
VACUUM
Old Version
20
X = Vac
not lvac
X = torque-t_min / t_wot-t_min *100
x=100.0 *(Torqe-T_min)/(T_wot-T_min)
VACUUM
LVAC
X = vac
LTHR
x = Thr
New Version
Throttle (Pos'n (deg)
Y = romp
leng
Note: No detent code for the truck/bus version
ngine ooen
y = rpme
parab
vehicle velocity
y = v
luo = .t.
upshift flag set
IRER = NRER
check if already at top gear
Y< rerat(16, rerat)
check if speed too low
240
NUP = rerat
Y> rerat ( )
check if speed too high for current gear
140
COMPUTE "GC"
lvac
80
X-GO
X>GO
240
140
IF(its≠1 or .not. lmph or...)
IF
N
160
Return
Idebug =2
Calldebug
irero = lrer
irer = irer+1
.not. lup
irer=irer-2
240
luo = .F.
irer = 1
Return
Y<rerat
120
NON = rerat...
COMPUTE "GO"
.not. lvac
260
X _GO
N
Return
X≤ GO
N
120
if (its = 2 & bper <0 & lockuo)
If ...
N
Y
Return
Entry Bkshft
Idebug ≠2
Limtty
Write msg
Write msg
Llot
300
ngear=rerat ( )
ieng=jeng ( )
stime=rerat ( )
llsh= .T.
naxo=nax
nax=rerat ( )
irer=lrero
ngear=rerat ( )
llsh= .F.
nax=naxo
200
error msg
Return
nax > naxs
180
nax=1 or nax=naxo
Return
if(nax =2)
Return

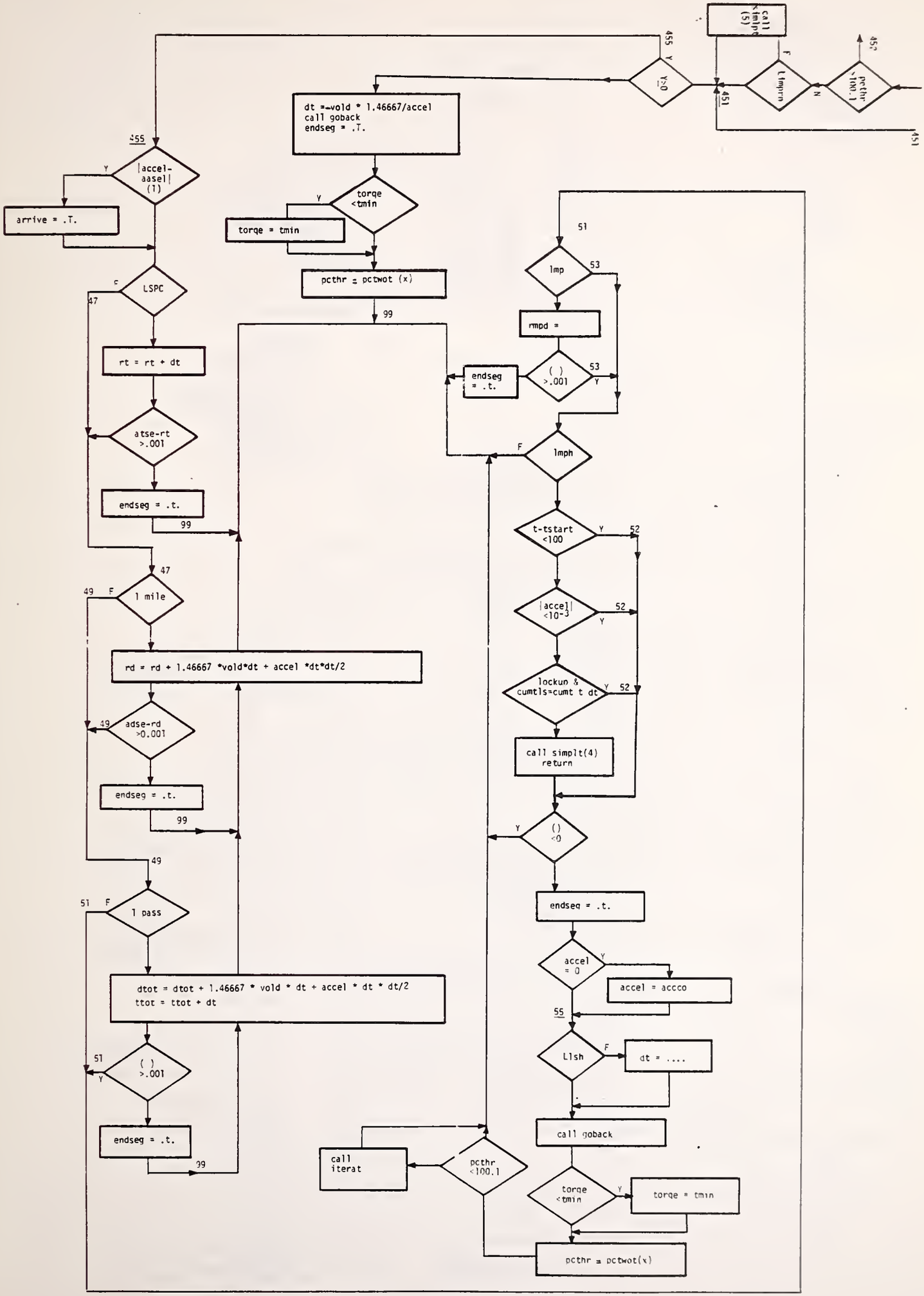
ITERAT
(Revised Version)

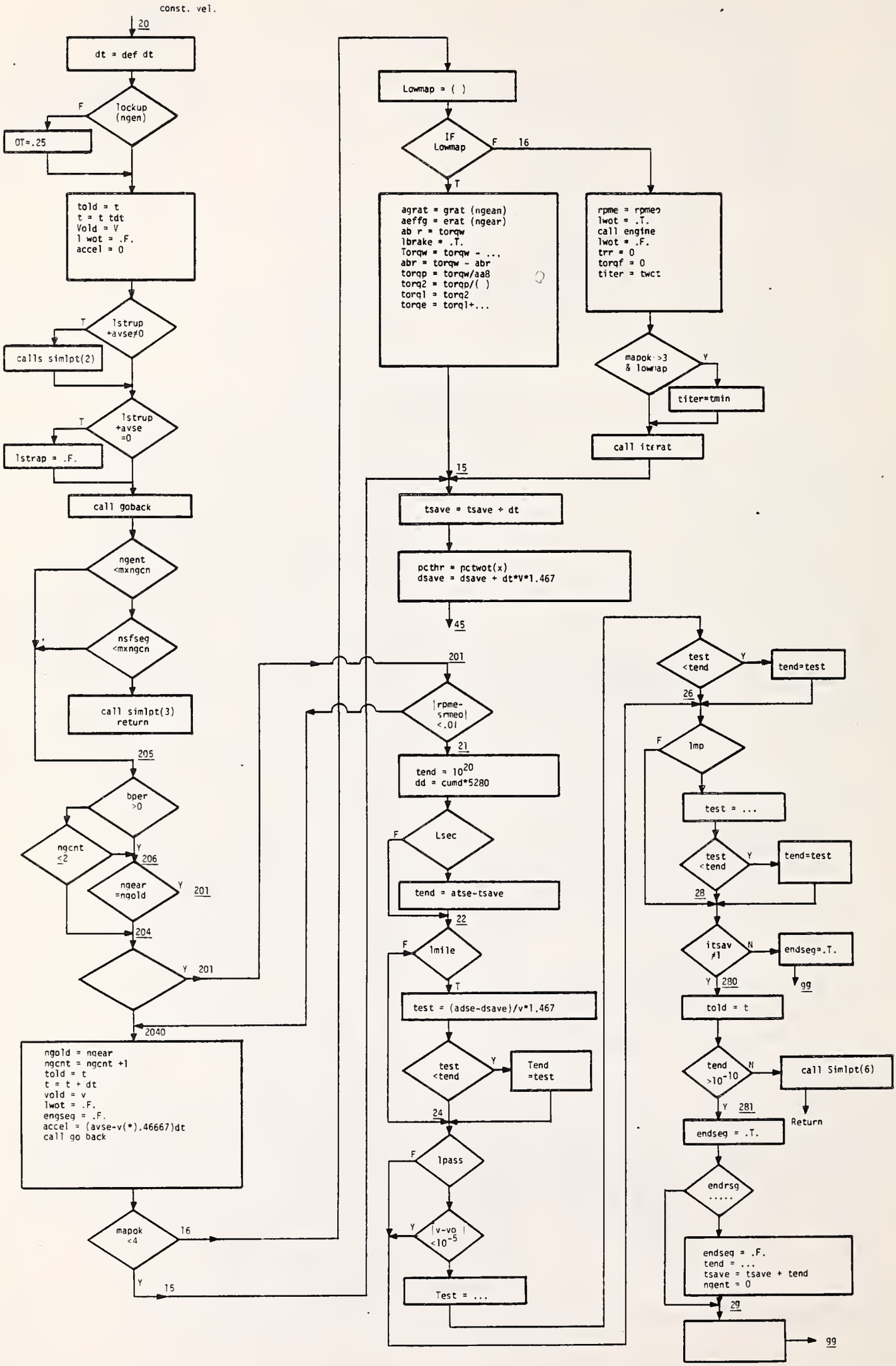


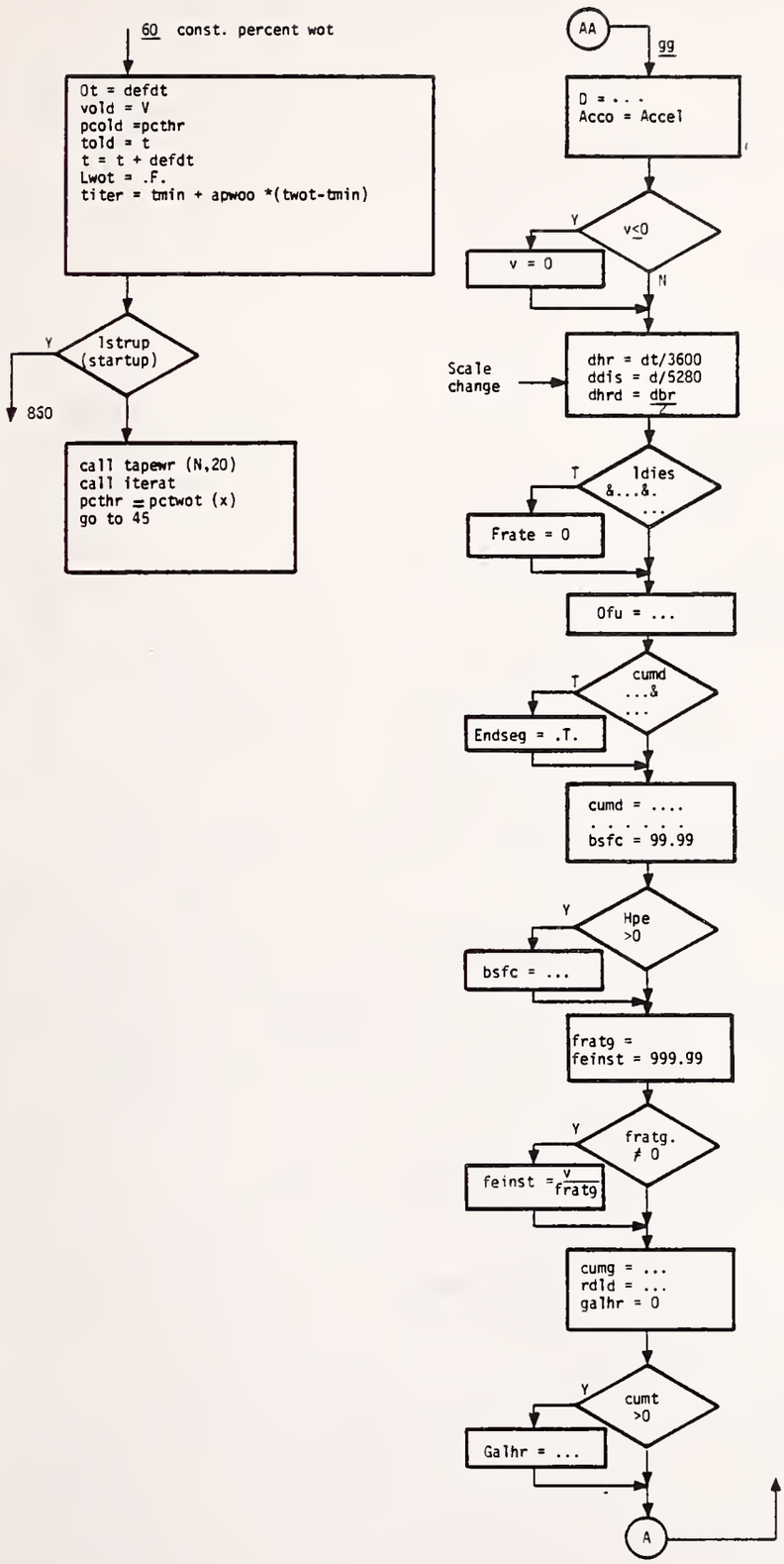
ITERAT
(Old Version)

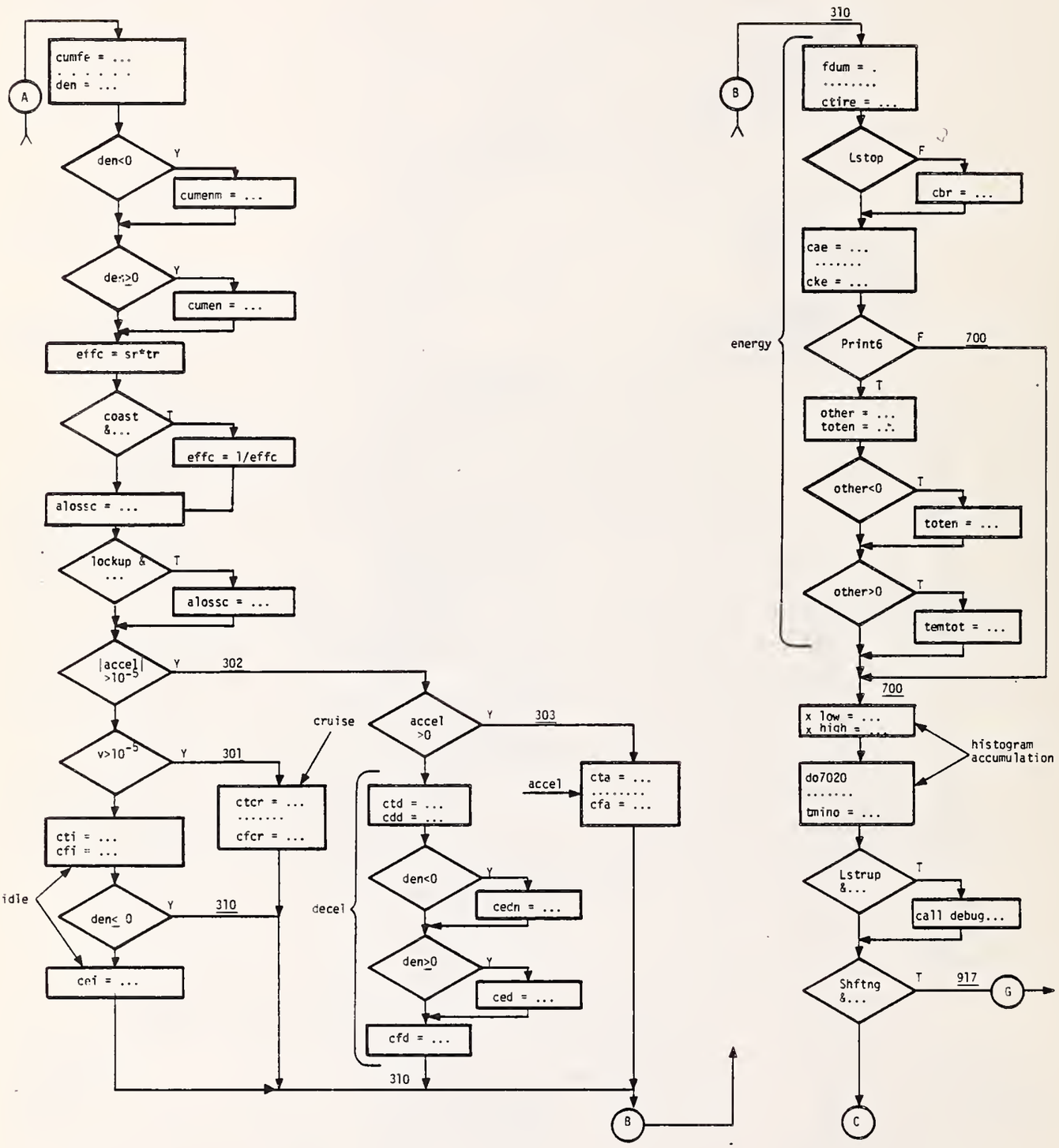


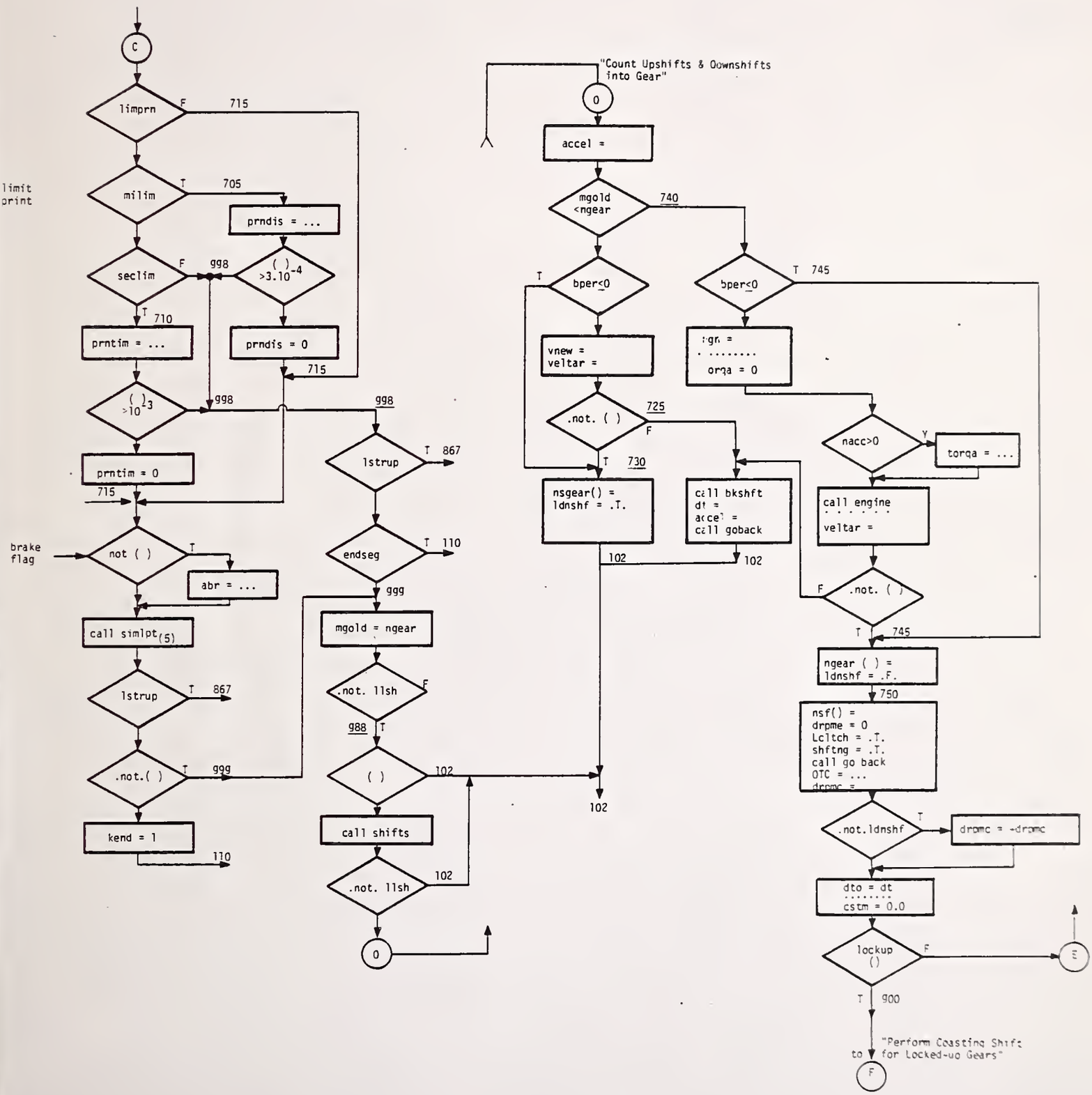

Start
pctwot(x) =
Simode = simodt
call simint
sdelay = 0.8
BPER >0
sdelay = sdelay x Bper
Bpero = bper
lecond = 0
kend = 0
6
call Kptime(2)
Lstrup = .false.
V <10^-5
Lstrup = true
a start = accel
acco = 0
apse >-.5
lsec = true
adse >-.5
mile = true
apse >-.5
lpass = true
apos >-.5
lmp = true
initialization
avel >-.5
lmph = true
|athr| >10^-20
lthr = true
lmile
adse = adse*528
LMP
apos = apos*5280
Tsave = 0
dsave = 0
nngs = 0
lshft = true
ngear = nngs
9
leng = jeng(ngear)
ngcnt = 0
10
ngearo = ngear
ITS
1 2 3 4
const accel 40
Lwot = .f.
tofast = .f.
lmph & vstart>avel & asse >0
tofast = .t.
tofast
its = 1
avse = avel
20
42
ITSAV = 2 & |bper|_3 <10^-3
its = 2
told = t
dt = defdt
t = t+dt
accel = aase
vold = v
pcold = pcthr
lstrap
860
call goback
pcthr = pctwot(x)
mapok >3
428
arrive
piter = rthr*dttpcolo
pcthr >piter -.01
434
mapok =50r =7
431
piter = rthr*dt+pcold
arrive = .t.
100> piter-.01
434
44
call iterat
titer = tmin+piter*( )x0.01
lockup
pcthr = pctwot(x)
45
print msg
dd = cumd*5280.
accel >0
451
accel = acco + 0.01
call goback
435
call iterate
aase <0
451
|bper| >.01
451
455
accoo = accel
accel = 0.
452
pcthr >100.1
OT = OT + OEFOT
call goback
pcthr = pctwot(x)

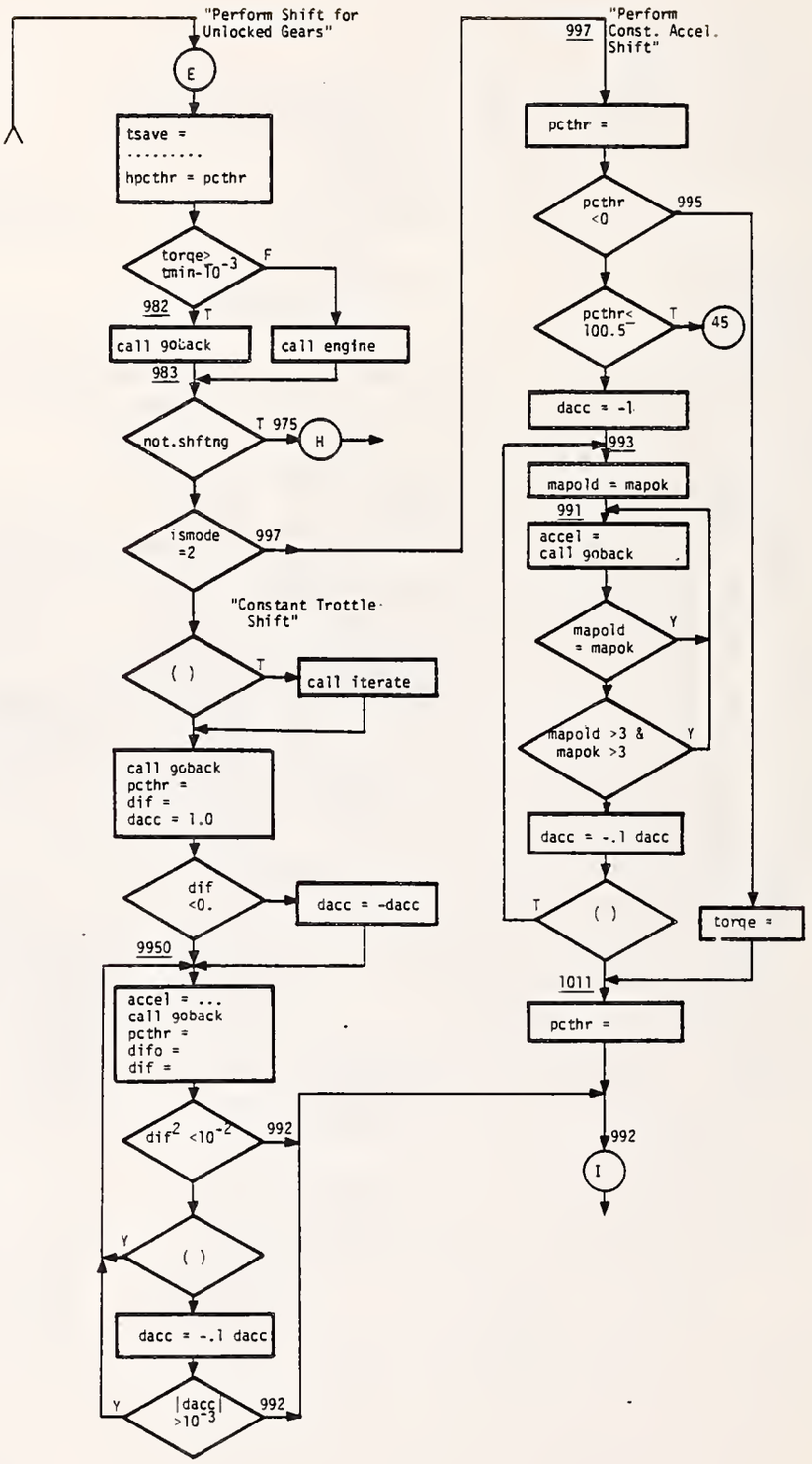

call limpn (5)
455
Y
V≥0
F
Limpn
N
pcthr <100.1
452
451
dt =-vold * 1.46667/accel
call goback
endseg = .T.
455
|accel-aasel| (1)
Y
arrive = .T.
torqe <tmin
torqe = tmin
pcthr = pctwot (x)
99
51
lmp
53
rmpd =
( ) >.001
endseg = .t.
lmph
t-tstart <100
52
|accel| <10**-3
lockup & cumtls=cumt t dt
call simplt(4) return
( ) <0
endseg = .t.
accel = 0
accel = accco
55
Llsh
dt = ...
call goback
torqe <tmin
torqe = tmin
pcthr = pctwot(x)
pcthr <100.1
call iterat
LSPC
47
rt = rt + dt
atse-rt >.001
endseg = .t.
99
47
49
F
1 mile
rd = rd + 1.46667 *vold*dt + accel *dt*dt/2
adse-rd >0.001
endseg = .t.
99
49
51
F
1 pass
dtot = dtot + 1.46667 * vold * dt + accel * dt * dt/2
ttot = ttot + dt
51
( ) >.001
endseg = .t.
99

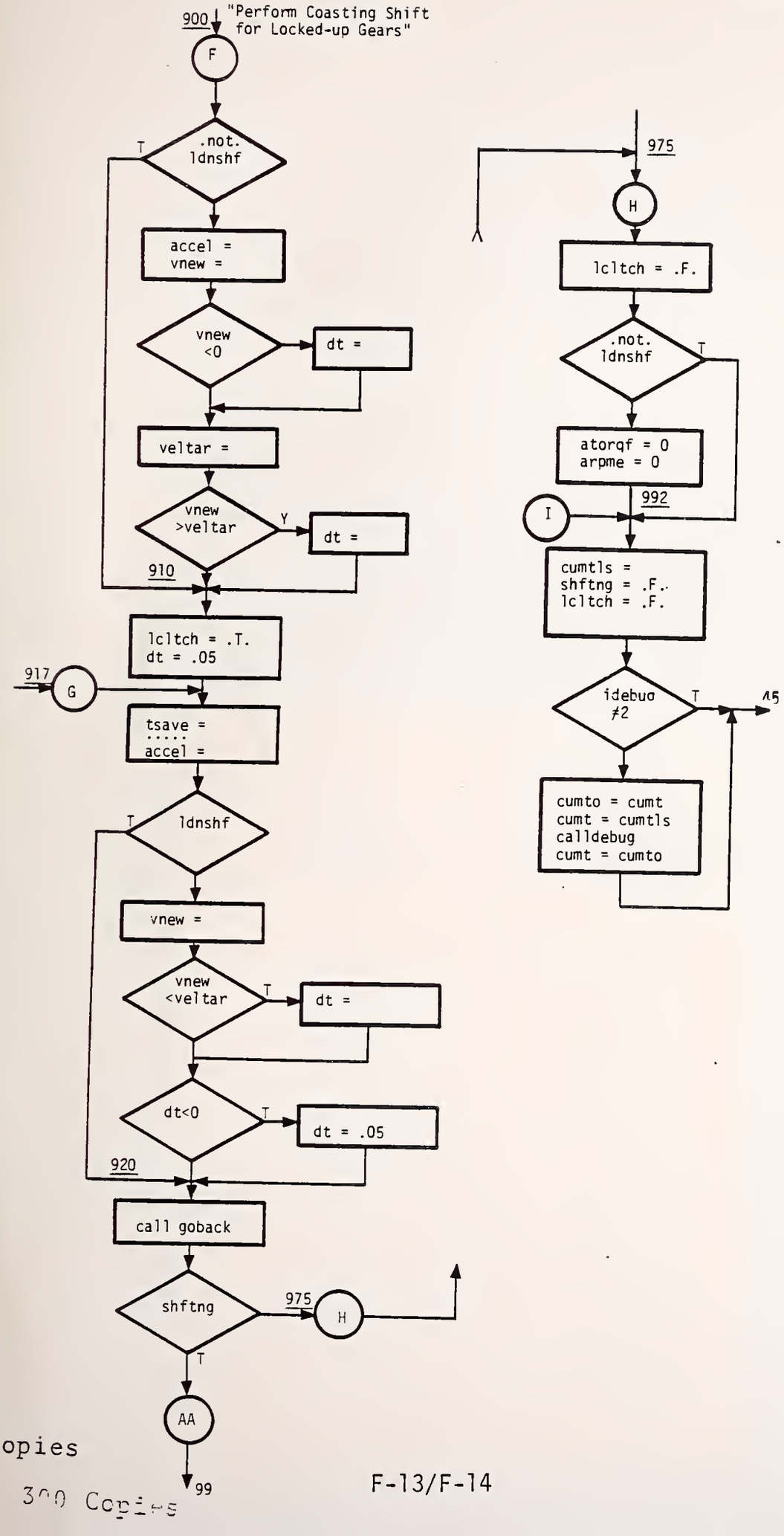

const. vel.
20
dt = def dt
lockup (ngen)
F
DT=.25
told = t
t = t tdt
Vold = V
l wot = .F.
accel = 0
lstrup +avse≠0
T
calls simlpt(2)
lstrup +avse =0
lstrap = .F.
call goback
ngent <mxngcn
nsfseg <mxngcn
call simlpt(3) return
205
bper >0
ngcnt ≤2
206
ngear =ngold
201
204
2040
ngold = ngear
ngcnt = ngcnt +1
told = t
t = t + dt
vold = v
lwot = .F.
engseg = .F.
accel = (avse-v(*).46667)dt
call go back
mapok <4
16
15
Lowmap = ( )
IF Lowmap
agrat = grat (ngean)
aeffg = erat (ngear)
ab r = torqw
lbrake = .T.
Torqw = torqw - ...
abr = torqw - abr
torqp = torqw/aa8
torq2 = torqp/( )
torql = torq2
torqe = torql+...
rpme = rpmeo
lwot = .T.
call engine
lwot = .F.
trr = 0
torqf = 0
titer = twct
mapok >3 & lowmap
titer=tmin
call iterat
tsave = tsave + dt
pcthr = pctwot(x)
dsave = dsave + dt*V*1.467
45
201
|rpme-rmeo| <.01
21
tend = 10^20
dd = cumd*5280
Lsec
tend = atse-tsave
22
lmile
test = (adse-dsave)/v*1.467
test <tend
Tend =test
24
lpass
|v-vo| <10^-5
Test = ...
test <tend
tend=test
26
lmp
test = ...
test <tend
tend=test
28
itsav ≠1
endseg=.T.
gg
280
told = t
tend >10^-10
call Simlpt(6)
Return
281
endseg = .T.
endrsg .....
endseg = .F.
tend = ...
tsave = tsave + tend
ngent = 0
29
99


60 const. percent wot
0t = defdt
vold = V
pcold =pcthr
told = t
t = t + defdt
Lwot = .F.
titer = tmin + apwoo *(twot-tmin)
Y
lstrup
(startup)
830
call tapewr (N,20)
call iterat
pcthr = pctwot (x)
go to 45
AA
gg
D = . . .
Acco = Accel
Y
v≤0
v = 0
N
Scale
change
dhr = dt/3600
ddis = d/5280
dhrd = dbr
T
ldies
&...&.
...
Frate = 0
0fu = ...
T
cumd
...&
...
Endseg = .T.
cumd = ...
. . . . . .
bsfc = 99.99
Y
Hpe
>0
bsfc = ...
fratg =
feinst = 999.99
Y
fratg.
≠ 0
feinst = v/fratg
cumg = ...
rdld = ...
galhr = 0
Y
cumt
>0
Galhr = ...
A


A
cumfe = ...
den = ...
den<0
Y
cumenm = ...
den≥0
Y
cumen = ...
effc = sr*tr
coast &...
T
effc = 1/effc
alossc = ...
lockup & ...
T
alossc = ...
|accel| >10^-5
Y
302
v>10^-5
Y
301
cruise
ctcr = ...
cfcr = ...
cti = ...
cfi = ...
idle
den≤ 0
Y
310
cei = ...
accel >0
Y
303
accel
cta = ...
cfa = ...
decel
ctd = ...
cdd = ...
den<0
Y
cecn = ...
den≥0
Y
ced = ...
cfd = ...
310
B
B
310
fdum = .
ctire = ...
Lstop
F
cbr = ...
cae = ...
cke = ...
energy
Print6
F
700
T
other = ...
toten = ...
other<0
T
toten = ...
other>0
T
temtot = ...
700
x low = ...
x high = ...
histogram accumulation
do7020
tmino = ...
Lstrup &...
T
call debug...
Shftng &...
T
917
G
C


C
limprn
F
715
limit print
milim
T
705
prndis = ...
seclim
F
998
( )-4 >3.10
T
710
prntim = ...
prndis = 0
715
( )3 >10
998
998
prntim = 0
715
brake flag
not ( )
T
abr = ...
call simlpt(5)
lstrup
T
867
.not.( )
T
999
kend = 1
110
lstrup
T
867
endseg
T
110
999
mgold = ngear
.not. llsh
F
988
T
( )
102
call shifts
.not. llsh
102
0
102
"Count Upshifts & Downshifts into Gear"
0
accel =
mgold <ngear
740
bper≤0
T
vnew = veltar =
.not. ( )
725
F
T
730
nsgear() = ldnshf = .T.
call bkshft
dt =
accel =
call goback
102
102
bper≤0
T
745
orqa = 0
nacc>0
Y
torqa = ...
call engine
veltar =
F
.not. ( )
T
745
ngear ( ) = ldnshf = .F.
750
nsf() =
drpmc = 0
Lcltch = .T.
shftng = .T.
call go back
OTC = ...
.not.ldnshf
T
drpmc = -drpmc
dto = dt
cstm = 0.0
lockup ( )
F
E
T
900
to
"Perform Coasting Shift for Locked-up Gears"
F


"Perform Shift for Unlocked Gears"
E
tsave =
.....
hpcthr = pcthr
torqe> tmin-T0^-3
F
982
T
call goback
call engine
983
not.shftng
T 975
H
ismode =2
997
"Constant Trottle Shift"
( )
T
call iterate
call goback
pcthr =
dif =
dacc = 1.0
dif <0.
dacc = -dacc
9950
accel = ...
call goback
pcthr =
difo =
dif =
dif^2 <10^-2
992
( )
Y
dacc = -.1 dacc
|dacc| >10^-3
Y
992
997
"Perform Const. Accel. Shift"
pcthr =
pcthr <0
995
pcthr< 100.5
T
45
dacc = -1
993
mapold = mapok
991
accel = call goback
mapold = mapok
Y
mapold >3 & mapok >3
Y
dacc = -.1 dacc
( )
T
torqe =
1011
pcthr =
992
I



300 Copies

300 Copies

Research and Special Programs Administration

Kendall Square
Cambridge, Massachusetts 02142

Official Business
Penalty for Private Use $300

U.S. MAIL