[illegible]
[illegible]

System Operations Studies for Automated Guideway Transit Systems
Detailed Station Model
Programmer's Manual

John F. Duke
Roger B. Blanchard

GM Transportation Systems Division
General Motors Corporation
GM Technical Center
Warren MI 48090

January 1982
Final Report

US Department of Transportation

**Urban Mass Transportation
Administration**

Office of Technology Development and Deployment
Office of New Systems and Automation
Washington DC 20590

E5
A37
no. DOT-TSC-UMTA-
-44.1

OCLC # 8350

Technical Report Documentation Page

1. Report No. UMTA-MA-06-0048-81-8	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle SYSTEM OPERATIONS STUDIES FOR AUTOMATED GUIDEWAY TRANSIT SYSTEMS - Detailed Station Model Programmer's Manual		5. Report Date January 1982
		6. Performing Organization Code DTS-723
7. Author's) John F. Duke, GM TSD, and Roger Blanchard, IBM Federal Systems Division		8. Performing Organization Report No. DOT-TSC-UMTA-81-44, I
9. Performing Organization Name and Address GM Transportation Systems Division* General Motors Corporation GM Technical Center Warren, MI 48090		10. Work Unit No. (TRAIS) UM268/R2670
		11. Contract or Grant No. DOT-TSC-1220-4
12. Sponsoring Agency Name and Address U.S. Department of Transportation Urban Mass Transportation Administration Office of Technology Development and Deployment Office of New Systems and Automation Washington DC 20590		13. Type of Report and Period Covered Final Report June 1977 - January 1979
		14. Sponsoring Agency Code UTD-40

15. Supplementary Notes
*Under contract to: U.S. Department of Transportation
Research and Special Programs Administration
Transportation Systems Center
Cambridge, MA 02142

16. Abstract

The Detailed Station Model (DSM) provides operational and performance measures of alternative station configurations and management policies with respect to vehicle and passenger capabilities. It provides an analytic tool to support tradeoff studies between alternative operational strategies and station traffic flow patterns to assist the initial station design selection by planners. This report describes global variables, subprogram logic, and subprogram descriptions for the maintenance and modification of this model.

17. Key Words -Scheduled Service, Queue Size, Queue Time, Process Time, Demand-Responsive Single Party, Demand-Responsive Multiparty	18. Distribution Statement DOCUMENT IS AVAILABLE TO THE PUBLIC THROUGH THE NATIONAL TECHNICAL INFORMATION SERVICE, SPRINGFIELD, VIRGINIA 22161

19. Security Classif. (of this report) Unclassified	20. Security Classif. (of this page) Unclassified	21. No. of Pages 361	22. Price

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

PREFACE

In order to examine specific Automated Guideway Transit (AGT) developments and concepts--and to build a better knowledge base for future decision-making--the Urban Mass Transportation Administration (UMTA) undertook a program of studies and technololgy investigations called the UMTA Automated Guideway Transit Technology (AGTT) program. The objectives of one segment of the AGTT program, the Systems Operation Studies (SOS), were to develop models for the analysis of system operations, to evaluate performance and cost, and to establish guidelines for the design and operation of AGT systems. A team headed by GM Transportation Systems Division (GMTSD) was awarded a contract by the Transportation Systems Center to pursue these objectives. The Technical Monitor for the project at TSC was Arthur Priver, who was assisted by Li Shin Yuan and Thomas Dooley.

The Detailed Station Model (DSM) is a discrete event model representing the inter-related queueing processes associated with vehicle and passenger activities in an AGT station. The DSM will be used to analyze alternative station configurations and management policies. This Programmer's Manual describes global variables, subprogram logic, and subprogram descriptions for the maintenance and modification of this model.

This document was prepared under the direction of the SOS Program Manager at GMTSD, James F. Thompson. The first draft of this report was prepared by the IBM Federal Systems Division (FSD) under the direction of Roger Blanchard, and its final preparation was the responsibility of John F. Duke of GMTSD.

METRIC CONVERSION FACTORS

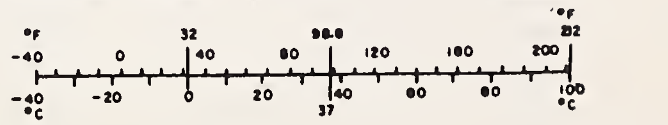
Approximate Conversions to Metric Measures

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

inches
1
2
3
4
5
6
7
8
9
cm
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23

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



CONTENTS

ILLUSTRATIONS

SECTION 1. INTRODUCTION

This is the Programmer's Manual of the Detailed Station Model (DSM). This model consists of three processors: Input Processor (IP), Model Processor (MP), and Output Processor (OP). These three processors are executed independently to prepare input data, perform the simulation, and report the results, respectively.

The code itself consists of:

1. Routines that are compiled using PARAFOR or the assembler;

2. Included Members that are placed in line in routines when they are compiled;

3. Macros that are expanded into code in line when the routine is compiled (PL/I and ASM);

4. Common Area Data Definitions that are included in line when the routine is compiled (just a subset of included members); and

5. Entry Points within routines called when a particular subprocess of the routine is to execute.

The names of the 'code segments' (i.e., routines, included members, macros, and common area definitions) and entry points are listed for the IP in Table 1-1, for the MP in Table 1-2 and for the OP in Table 1-3.

Each table lists for each code segment and entry point, its name, description, language, type, and sources of detailed information. There are three main sources of detailed information on these code segments:

- PDL -- Program Design Language -- given in Appendix A;

- Program Descriptions -- given in Section 6; and

- Preambles -- included at the beginning of every major routine and included member -- see source listings for members with names of the form xxx0.

These tables serve as a guide to where detailed information can be found on each code segment.

Table 1-1. DSM Input Processor (Page 1 of 3)

DSM - INPUT PROCESSOR --- CODE SEGMENTS & ENTRY POINTS

NAME	LIB	TYPE	PDL	C.D.	PREAM	DESCRIPTION
ATYPE	ASM	M	YM	YM	YM	STANDARDIZED ASM LANG RTN LINKAGE
CALLS	ASM	M	YM	YM	YM	STANDARDIZED ASM LANG RTN LINKAGE
COMN	ASM	M	YM	YM	YM	GDIP COMMON AREA CSECT GENERATION MACRO
DAYTIM	FORT	R	X	X	X	GET CURRENT DATE AND TIME YY/MM/DD/HH/MM/SS
DBUG	PLI	M	M	X	M	WRITE INTERMEDIATE OUTPUT
DO	PLI	M	YM	YM	YM	STANDARDIZED REGISTER SAVE MACRO
DTIMEL	ASM	R	Y	X	X	SOURCE MEMBER FOR SUBROUTINE TIMES
ENDEFS	ASM	R	YM	YM	YM	GDIP COMMON AREA PROCESSOR MACRO
ENTER	ASM	M	YM	YM	YM	STANDARD ASM LANG ENTRY MACRO
ERROR	FORT	R	X	X	X	ERROR MESSAGE WRITE (SOURCE=SIERROR)
GDIPSECT	ASM	R	X	X	X	READ GDIP DATA (SOURCE=SIGDIP4)
GDIPF4	FORT	R	Y	Y	X	READ FULLWORD GDIP DATA (SOURCE=XGDIPF4)
GDIPH4	FORT	R	Y	Y	X	READ HALFWORD GDIP DATA (SOURCE=XGDIPH4)
GDIPX4	FORT	R	Y	Y	X	READ BYTE SIZE GDIP DATA (SOURCE=XGDIPX4)
GDIP4	ASM	E	E	E	E	ENTRY POINT OF GDIPSECT
LBL	ASM	M	YM	YM	YM	STANDARD ASM LANG TEST FOR 0 MACRO
LEAVE	ASM	M	YM	YM	YM	STANDARD ASM LANG EXIT MACRO
LODCOM	ASM	E	E	E	E	SYSCHAR ADDR+LGTH LOAD (ENTRY IN SIPSAV)
NDBOR	ASM	R	Y	Y	X	READ GDIP FMT SPEC'D BY USER(SOURCE=XNDBOR)
NODIMENS	ASM	M	YM	YM	YM	ESTAB. NO. DIMENS FOR BLDG GDIP TABLE
SACOMN	FORT	R	X	X	X	INPUT COMMON AREA ORDERING
SAFLAG	FORT	R	X	X	X	INTERMEDIATE OUTPUT FLAG PROCESSING
SCAMSG	FORT	C	X	C	C	MESSAGE DATA MAINTAINED BY IP
SCICFG	FORT	C	X	C	C	STATION CONFIGURATION INPUT
SCIFEL	FORT	C	X	C	C	FEL TIMING INPUT DATA
SCIMAX	FORT	C	X	C	C	RUN TIME MAXIMA
SCISL	FORT	C	X	C	C	STATION LINK INPUT DATA
SCISYS	FORT	C	X	C	C	SYSTEM INPUT DATA
SCITL	FORT	C	X	C	C	TRIP LINK INPUT DATA
SCNMAX	FORT	C	X	C	C	RUNTIME LIMITS
SCNSYS	FORT	C	X	C	C	SIMULATION SYSTEM DATA
SCNTDM	FORT	C	X	C	C	TRIP DEMAND DATA
SCNVDM	FORT	C	X	C	C	VEHICLE DEMAND DATA
SIADDR	ASM	R	X	X	X	SYSTEM CHARAC ADDR AND LENGTH SAVE
SIBWRT	FORT	R	X	X	X	STRUCTURED DATA FILE WRITE
SICHCK	FORT	R	X	X	X	DATA INITIALIZATION AND CHECKING
SICHCK1	FORT	I	X	I	I	STATION LINK - EVENT COMPATIBILITY CHECK
SICUMP	FORT	R	X	X	X	CUM PROB. DIST. CONVERSION
SIERROR	FORT	R	X	X	X	SOURCE MEMBER NAME OF SUBRTN ERROR
SIGDIP4	ASM	R	X	X	X	SOURCE MEMBER NAME OF SUBRTN GDIPSECT
SIGIAT	FORT	R	X	X	X	VEH INTERARRIVAL TIME GENERATION
SIINIT	FORT	R	X	X	X	DATA INITIALIZATION
SILIMITS	FORT	C	C	C	C	DATA CHECK LIMITS
SIMNAM	FORT	R	X	X	X	PARSE PARM FIELD AND WRITE LOAD MODULE NAME
SINCOMNS	FORT	I	I	I	I	TRIP/VEH GENERATION COMMONS
SINCOMNT	FORT	I	I	I	I	TRIP GENERATION COMMONS INCLUDE
SINCOMNV	FORT	I	I	I	I	VEH GENERATION COMMONS INCLUDE
SINERR	FORT	R	X	X	X	ERROR MESSAGE GENERATION
SINPUT	FORT	R	X	X	X	INPUT PROCESSOR CONTROL
SINPUT1	FORT	I	X	I	I	SYSTEM CHARACTERISTICS READ
SINPUT2	FORT	I	X	I	I	RUNTIME DATA INPUT PROCESSING
SINPUT3	FORT	I	X	I	I	TRIP DEMAND DATA INPUT
SINPUT4	FORT	I	X	I	I	VEHICLE DEMAND DATA INPUT
SINPUT4A	FORT	I	X	I	I	NEXT STOP SELECTION DATA INPUT
SINPUT4B	FORT	I	X	I	I	INTERARRIVAL TIME DATA INPUT
SINPUT4C	FORT	I	X	I	I	TRIP SIZE DATA INPUT
SIPARM	ASM	R	X	X	X	PARAMETER LIST SAVE
SIPLST	ASM	E	E	E	E	ENTRY POINT OF SIPARM
SIPSAV	ASM	R	–	X	X	IP AND SYSCHAR COMMONS ADDR AND LGTH SAVE
SIREPT	FORT	R	X	X	X	INITIAL CONDITIONS REPORT
SIREPT10	FORT	I	I	I	I	REPORT FORMATS

Table 1-1. DSM Input Processor (Page 2 of 3)

NAME	LIB	T	P	C	P	DESCRIPTION
SIREPT2	FORT	I	I	I	I	REPORT LOCAL DECLARES AND INCLUDES
SIREPT3	FORT	I	X	I	I	TRIP AND VEH CHARACTERISTICS WRITE
SIREPT4	FORT	I	X	I	I	SERVICE CHARACTERISTICS SUMMARY
SIREPT4A	FORT	I	X	I	I	VEH SPACING AND HEADWAY SUMMARY
SIREPT4B	FORT	I	X	I	I	EMPTY POLICY SUMMARY
SIREPT4C	FORT	I	I	I	I	SIMULATION CONTROL DATA SUMMARY
SIREPT4D	FORT	I	X	I	I	ROUTE ASSIGNMENT SUMMARY
SIREPT5	FORT	I	X	I	I	STATION LINK SUMMARY
SIREPT5A	FORT	I	I	I	I	STATION LINK DATA
SIREPT5B	FORT	I	I	I	I	DNSTRM LINK AND DVRG FN VALIDATION
SIREPT5C	FORT	I	I	I	I	LINK CHARACTERISTICS CHECK
SIREPT5D	FORT	I	I	I	I	MULTIPLE UPSTRM/DNSTRM LINKS + EVENTS WRITE
SIREPT6	FORT	I	X	I	I	TRIP TRIP LINK SUMMARY
SISADD	FORT	E	E	E	E	SYS CHAR ADDR + LGTH SAVE (EP IN SIBWRT)
SISCFG	FORT	R	X	X	X	STATION CONFIGURATION
SISCFG1	FORT	I	I	I	I	LINK TYPE SCAN
SISCFG2	FORT	I	I	I	I	ERROR CHECKING
SISCFG3	FORT	I	I	I	I	STRUCTURED TABLE BUILD
SISCFG4	FORT	I	I	I	I	UPSTREAM POINTER TABLE BUILD
SISCFG4A	FORT	I	I	I	I	INPUT Q US LINKS BUILD
SISCFG4B	FORT	I	I	I	I	DOCK US LINKS BUILD
SISCFG4C	FORT	I	I	I	I	OUTPUT Q US LINKS BUILD
SISCFG4D	FORT	I	I	I	I	OUTPUT RAMP US LINKS BUILD
SISCFG4E	FORT	I	I	I	I	STORAGE US LINKS BUILD
SISCFG4F	FORT	I	I	I	I	DOCK TO STORAGE US LINKS BUILD
SISCFG4G	FORT	I	I	I	I	DL US LINKS BUILD
SISCFG4H	FORT	I	I	I	I	MOA US LINKS BUILD
SISCFG5	FORT	I	I	I	I	DOWNSTREAM LINKS PROCESSING
SISCFG5A	FORT	I	I	I	I	INPUT RAMP DS LINKS BUILD
SISCFG5B	FORT	I	I	I	I	INPUT Q DS LINKS BUILD
SISCFG5C	FORT	I	I	I	I	DOCK DS LINKS BUILD
SISCFG5D	FORT	I	I	I	I	OUTPUT QDS LINKS BUILD
SISCFG5E	FORT	I	I	I	I	STORAGE TO INPUT DS LINKS BUILD
SISCFG5F	FORT	I	I	I	I	UL DS LINKS BUILD
SISCFG5G	FORT	I	I	I	I	MIB DS LINKS BUILD
SISWRT	FORT	E	E	E	E	ENTRY POINT OF SIBWRT
SITDGN	FORT	R	X	X	X	TRIP GENERATION
SITDGN1	FORT	I	I	I	I	LOCAL DATA DEFNS
SITDGN2	FORT	I	X	I	I	INPUT VERIFICATION
SITDGN3	FORT	I	X	I	I	ERROR MESSAGE GENERATION
SITDGN4	FORT	I	X	I	I	TRIP GENERATION SUMMARY
SIVDGN	FORT	R	X	X	X	VEHICLE GENERATION
SIVDGN1	FORT	I	I	I	I	LOCAL DATA DEFNS
SIVDGN2	FORT	I	X	I	I	INPUT VERIFICATION
SIVDGN2A	FORT	I	I	I	I	PROB. DIST. CONVERSION TO CUM. PROB. DIST.
SIVDGN2B	FORT	I	X	I	I	ERROR MESSAGE GENERATION
SIVDGN4	FORT	I	X	I	I	SCHEDULED VEHICLE GENERATION
SIVDGN5	FORT	I	X	I	I	DEMAND RESPONSIVE VEHICLE GENERATION
SIVDGN6	FORT	I	X	I	I	ONBOARD TRIP GENERATION
SIVDGN7	FORT	I	X	I	I	VEHICLE GENERATION SUMMARY
SIWNAM	FORT	E	E	E	E	LIST USED MEMBERS IN INDEX
SMAXSIZE	PLI	I	X	I	I	COMPILE TIME SIZES
SMRNG	FORT	R	X	X	X	RANDOM NUMBER GENERATOR
SMRSEL	FORT	R	X	X	X	CUM. PROB. DIST. SAMPLING
SPIEL	ASM	E	E	E	E	INIT. FOR PROGRAM INTERRUPT (ENTRY IN TRACBK)
TIMES	ASM	R	X	X	X	GET CURR SYS DATE + TIME (SOURCE=DTIMEL)
TRACBK	ASM	R	Y	Y	X	PERFORM TRACEBACK (SOURCE=XTRACBK)
TRCBKP	FORT	R	Y	Y	X	PRINT CALLING RTN INFO (SOURCE=XTRCBKP)
TRCKI	FORT	E	E	E	E	PRINT HEADING LINK (ENTRY IN TRCBKP)
TRCBKV	FORT	E	E	E	E	PRINT 2 LINES FOR ARGUMENT (ENTRY IN TRCBKP)
TRCBKR	FORT	E	E	E	E	PRINT 3 LINES FOR GEN. REG. (ENTRY IN TRCBKP)
UNDO	ASM	M	YM	YM	YM	STANDARD REGISTER RESTORE MACRO
XGDIPP4	FORT	R	Y	Y	X	SOURCE MEMBER NAME OF SUBRTN GDIPP4
XGDIPH4	FORT	R	Y	Y	X	SOURCE MEMBER NAME OF SUBRTN GDIPH4
XGDIPX4	FORT	R	Y	Y	X	SOURCE MEMBER NAME OF SUBRTN GDIPX4
XNDBOR	FORT	R	Y	Y	X	SOURCE MEMBER NAME OF SUBROUTINE NDBOR
XTRACBK	ASM	R	Y	Y	X	SOURCE MEMBER NAME OF SUBROUTINE TRACBK
XTRCBKP	FORT	R	Y	Y	X	SOURCE MEMBER NAME OF SUBROUTINE TRCBKP

Table 1-1. DSM Input Processor (Page 3 of 3)

```
                    Y   D   .   R
                    P   L   D   E
                    E       .   A
                                M
------------------------------------------------------------------------------
NOTATIONS:

  * = DOES NOT INCLUDE PREAMBLE CODE SEGMENTS(XXXXXX0)

  TYPE:
     R = ROUTINE
     I = INCLUDED MEMBER
     M = MACRO (PLI OR ASM)
     C = COMMON AREA DEFINITION (WHICH IS INCLUDED)
     E = ENTRY POINT (IN ROUTINE WHOSE NAME IS GIVEN IN DESCRIPTION)

  PDL:
     X = PDL GIVEN
     I = PDL NOT GIVEN, SINCE IT INCLUDED MEMBER AND TREATED IN THE PD
           OF WHAT IT IS INCLUDED IN
     E = PDL NOT GIVEN, SINCE IT IS AN ENTRY POINT
     M = PDL NOT GIVEN, SINCE IT IS A MACRO
     Y = PDL NOT GIVEN, SINCE IT IS EXISTING CODE AND DISCUSSED
           UNDER GENERAL PURPOSE ROUTINES

  COMPONENT DESCRIPTION:
     X = COMPONENT DESCRIPTION GIVEN
     I = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT INCLUDED MEMBER
           AND TREATED IN THE COMPONENT DESCRIPTION OF WHAT IT IS
           INCLUDED IN
     C = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS A COMMON AREA
           DEFINITION
     E = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS AN ENTRY POINT
     M = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS A MACRO
     Y = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS EXISTING CODE
           AND IS DISCUSSED UNDER GENERAL PURPOSE ROUTINES

  PREAMBLE:
     X = PREAMBLE GIVEN
     I = PREAMBLE NOT GIVEN, SINCE IT INCLUDED MEMBER
           AND TREATED IN THE PREAMBLE OF WHAT IT IS
           INCLUDED IN
     C = PREAMBLE NOT GIVEN, SINCE IT IS A COMMON AREA
           DEFINITION
     E = PREAMBLE NOT GIVEN, SINCE IT IS AN ENTRY POINT
     M = PREAMBLE NOT GIVEN, SINCE IT IS A MACRO
     Y = PREAMBLE NOT GIVEN, SINCE IT IS EXISTING CODE
           AND IS DISCUSSED UNDER GENERAL PURPOSE ROUTINES
```

Table 1-2. DSM Model Processor (Page 1 of 3)

DSM - MODEL PROCESSOR --- CODE SEGMENTS & ENTRY POINTS(*)

NAME	LIB	TYPE	PDL	C.D.	PREAM	DESCRIPTION
ATYPE	ASM	M	YM	YM	YM	STD MACRO GENERATION LIMIT MACRO
CALLS	ASM	M	YM	YM	YM	STD ASSEMBLY LANG ROUTINE LINKAGEMACRO
COMN	ASM	M	YM	YM	YM	GDIP COMMON AREA CSECT GENERATION MACRO
DAYTIM	FORT	R	X	X	X	CONVERT DATE & TIME TO YY/MM/DD/HH/MM/SS
DBUG	PLI	M	M	X	M	WRITE INTERMEDIATE OUTPUT
DO	ASM	M	YM	YM	YM	STANDARDIZED REGISTER SAVE MACRO
DQUE	PLI	M	M	X	M	DEQUEUE XTN FIFO
DQUEM	PLI	M	M	X	M	DEQUEUE XTN FROM ANYWHERE IN CHAIN
DQUEMID	PLI	M	M	X	M	DEQUEUE XTN FROM ANYWHERE IN CHAIN (W.QLOOP)
DTIMEL	ASM	R	Y	X	X	GET DATE & TIME FROM SYSTEM
ENDEFS	ASM	M	YM	YM	YM	GDIP COMMON AREA PROCESSOR MACRO
ENTER	ASM	M	YM	YM	YM	STANDARD ASSEMBLY LANGUAGE ENTRY MACRO
ENDQLOOP	PLI	M	M	X	M	END QLOOP CODE SEGMENT
FREE	PLI	M	M	X	M	RETURN XTN TO AVAILABLE CHAIN
GET	PLI	M	M	X	M	GET XTN FROM AVAILABLE CHAIN
LBL	ASM	M	YM	YM	YM	STANDARD ASSEMBLY LANGUAGE TEST FOR 0 MACRO
LEAVE	ASM	M	YM	YM	YM	STANDARD ASSEMBLY LANGUAGE EXIT MACRO
LODCOM	ASM	E	E	E	E	SYSCHAR ADDR+LGTH LOAD (EP-SSASAV)
MULTICK	PLI	M	M	X	M	CHECK IF XTN IS ALREADY IN A CHAIN
NODIMENS	ASM	M	YM	YM	YM	ESTAB.NO.DIMENS FOR BUILDING GDIP TABLE
NQUE	PLI	M	M	X	M	ENQUEUE XTN FIFO
PSEUDO	ASM	E	E	E	E	XPSEUDO-MAIN ENTRY
QLOOP	PLI	M	M	X	M	LOOP THROUGH CHAIN & DO CODE SEGMENT ON EACH
SAASYN	FORT	R	X	X	X	ASYNCHRONOUS DATA READ
SACKPT	FORT	E	E	E	E	WRITE CHECKPOINT RECORD(EP-SACKR)
SACKR	FORT	R	X	X	X	CHECKPOINT & RESTART PROCESSING
SACOMN	FORT	R	X	X	X	DEFINE ORDER OF INPUT COMMONS FOR IP & MP
SADADD	FORT	R	X	X	X	INITIALIZE INPUT AREA ADDRESSES & MESSAGES
SAFAIL	FORT	R	X	X	X	FAILURE ACTIVITY PROCESSING
SAFINM	FORT	R	X	X	X	WRITE MODEL REPORT
SAFINS	FORT	R	X	X	X	FEL USAGE REPORT
SAFLAG	FORT	R	X	X	X	INTERMEDIATE OUTPUT FLAG SETTING
SAINIT	FORT	R	X	X	X	INITIALIZE SIMULATION
SAINIT1	FORT	I	I	I	I	SCHEDULE INITIAL SYSTEM SERVICE TRANSACTIONS
SAMAIN	FORT	R	X	X	X	MAIN CONTROL LOOP
SANDTA	FORT	E	E	E	E	READ INPUT DATA INTO INPUT COMMONS(EP-SADADD)
SANFEL	FORT	R	X	X	X	INITIALIZE FUTURE EVENT LIST
SANMDL	FORT	R	X	X	X	MODEL VARIABLE INITIALIZATION
SANSAV	ASM	R	X	X	X	INIT CKPT & SYSTEM DATA READ PROCESSES
SANTIX	ASM	R	X	X	X	INITIALIZE MEMBER NAME STRING FOR INDEX FILE
SANTSA	FORT	R	X	X	X	INITIALIZE SYSTEM STATUS AREA ADDRESSES
SANXTN	FORT	R	X	X	X	INITIALIZE XTN HEADER DATA & AVAILABLE LISTS
SAPFEL	FORT	R	X	X	X	PUT TRANSACTION ON FUTURE EVENT LIST
SAPFEL1	FORT	I	I	I	I	FIND TIME ORDER POSITION WITHIN CHAIN
SAPFEL2	FORT	I	X	I	I	WRITE TRIP/VEHICLE FILE AND UPDATE STATS
SAPFEL3	FORT	I	X	I	I	UPDATE VTIME FOR FOLLOWING VEHICLES IN TRAIN
SARFEL	FORT	I	X	X	X	REMOVE MOST IMMENENT XTN FROM FEL
SARFEL1	FORT	I	I	I	I	RELOAD CLOCK TABLE FROM MULTIPLE THREAD CHAIN
SAREST	FORT	E	E	E	E	READ CHECKPOINT RECORDS & RESET FILES(EP-SACKR
SASAMP	FORT	R	X	X	X	SAMPLE EVENT PROCESSING
SASCTL	FORT	I	X	X	X	CONTROL FOR VEHICLE EVENT PROCESSING
SASCTL1	FORT	I	X	I	I	FREE VEHICLES & TRIPS TO AVAILABILITY LISTS
SASPRM	FORT	I	X	X	X	STATION LINK PROMPT EVENT PROCESSING
SATORG	FORT	R	X	X	X	MOVE ARRIVING TRIP
SATRD	FORT	R	X	X	X	READ TRIP FROM TRIP FILE
SAUCTL	FORT	I	X	X	X	CONTROL FOR TRIP EVENT PROCESSING
SAUPRM	FORT	I	X	X	X	TRIP LINK PROMPT EVENT PROCESSING
SAUPTX	ASM	E	E	E	E	PASS MEMBER NAME STRING TO SAWTIX(EP-SANTIX)
SAVORG	FORT	R	X	X	X	MOVE ARRIVING VEHICLE
SAVRD	FORT	R	X	X	X	READ VEHICLE FROM VEHICLE FILE
SAWTIX	FORT	R	X	X	X	PARSE PARM LIST & WRITE LOAD MODULE NAME

Table 1-2. DSM Model Processor (Page 2 of 3)

SAWTIX	FORT	E	E	E	E	LIST USED MEMBERS IN INDEX(EP-SAWTIX)
SAZNIT	FORT	R	X	X	X	INITIALIZE STATISTICAL VARIABLES
SCAMSG	FORT	C	C	C	C	MESSAGE DATA MAINTAINED BY MP
SCHED	PLI	M	M	X	M	SCHEDULE TRIP OR VEHICLE ON FEL
SCIFEL	FORT	C	X	C	C	FEL TIMING INPUT DATA
SCIMAX	FORT	C	X	C	C	RUN TIME MAXIMA
SCISL	FORT	C	X	C	C	STATION LINK INPUT DATA
SCISYS	FORT	C	X	C	C	SYSTEM INPUT DATA
SCITL	FORT	C	X	C	C	TRIP LINK INPUT DATA
SCMFEL	FORT	C	X	C	C	FEL TIMING DATA MAINTAINED BY MP
SCMFS	FORT	C	X	C	C	FEL STATISTICS MAINTAINED BY MP
SCMSL	FORT	C	X	C	C	STATION LINK DATA MAINTAINED BY MODEL PROC.
SCMSYS	FORT	C	X	C	C	SYSTEM DATA MAINTAINED BY MODEL PROC.
SCMTL	FORT	C	X	C	C	TRIP LINK DATA MAINTAINED BY MODEL PROC.
SCMT	FORT	C	X	C	C	TRIP DATA MAINTAINED BY MODEL PROC.
SCMV	FORT	C	X	C	C	VEHICLE DATA MAINTAINED BY MODEL PROC.
SCMXTN	FORT	C	X	C	C	TRANSACTION HEADER DATA MAINTAINED BY MP
SCZ	FORT	C	X	C	C	STATISTICAL VARIABLES MAINTAINED BY MP
SERROR	FORT	R	X	X	X	WRITE ERROR MSG AND CONTINUE/TERMINATE
SHEAD	PLI	I	X	I	I	IMPLICIT(A-Z),PARAFOR,SMAXSIZE,SMACRO
SMACRO	PLI	I	X	I	I	PLI MACROS
SMAXSIZE	PLI	I	X	I	I	COMPILE TIME SIZES
SMBRD	FORT	R	X	X	X	PLANNING TRIP BOARDING
SMDBRD	FORT	I	X	X	X	PLANNING TRIP DEBOARDING
SMDLTR	FORT	R	X	X	X	DETRAIN VEHICLES FROM THE LEAD VEH OF TRAIN
SMDIVF	FORT	R	X	X	X	DIVERGE FUNCTIONS
SMDIVO	FORT	R	X	X	X	ORDER STATION LINKS BY OCC/PSEUDO-OCC
SMDIVS	FORT	R	X	X	X	SEARCH FOR STATION LINK TYPE
SMENTR	FORT	R	X	X	X	ENTRAIN FOLLOWING VEHICLES TO LEAD VEHICLE
SMEVM	FORT	I	X	X	X	EMPTY VEHICLE MANAGEMENT
SMGDIP4	ASM	R	X	X	X	DEFINE LAYOUT OF INPUT COMMON AREAS
SMLTIM	FORT	I	X	X	X	LAUNCH TIME DELAY DUE TO SCHEDULE
SMNXST	FORT	I	X	X	X	VEHICLE NEXT STOP DETERMINATION
SMRNG	FORT	R	X	X	X	GENERATE UNIFORMLY DISTRIBUTED RANDOM NUMBER
SMRSEL	FORT	R	X	X	X	RANDOMLY SELECT POINT ON CUMULATIVE DIST.
SMTABQ	FORT	R	X	X	X	PREPARE A TRIP FOR BOARDING
SMTABQ1	FORT	I	I	I	I	ORIGINATE A VEHICLE
SMTABQ2	FORT	I	X	I	I	BOARD WAITING TRIP
SPIEL	ASM	E	E	E	E	INTERFACE TO INTERRUPT HANDLER
SSASAV	ASM	R	X	X	X	INITIALIZE ARRAY SYSTEM STATUS AREA WORDS
SSLEAV	FORT	R	X	X	X	PROCESSING A VEHICLE/TRAIN LEAVING A SL
SSMOD	FORT	R	X	X	X	MODEL THE VEHICLE ON ITS CURRENT STATION LINK
SSMODA	FORT	R	X	X	X	VEHICLE PROCESSING AFTER A STATION LINK EVENT
SSMODA1	FORT	I	X	I	I	AFTER DEBOARD EVENT
SSMODA2	FORT	I	X	I	I	AFTER BOARD EVENT
SSMODA3	FORT	I	X	I	I	AFTER LAUNCH EVENT
SSMODB	FORT	R	X	X	X	VEHICLE PROCESSING BEFORE A STATION LINK EVENT
SSMODB1	FORT	I	X	I	I	BEFORE TRAVEL EVENT
SSMODB2	FORT	I	X	I	I	BEFORE DEBOARD EVENT
SSMODB3	FORT	I	X	I	I	BEFORE BOARD EVENT
SSMODB4	FORT	I	X	I	I	BEFORE JOINT EVENT
SSMODB5	FORT	I	X	I	I	BEFORE LAUNCH EVENT
SSMODN	FORT	R	X	X	X	VEHICLE'S NEXT SL EVENT DETERMINATION
SSPMAC	PLI	M	M	X	M	STATION LINK PROMPT TEST
SSTEST	FORT	R	X	X	X	STATION LINK ENTRY TESTING & NEXT LINK DETERM
SUDOGO	ASM	E	E	E	E	INITIALIZE PSEUDO-I/O (EP-XPSEUDO)
SULEAV	FORT	I	X	X	X	PROCESSING A TRIP LEAVING A TRIP LINK
SUMOD	FORT	R	X	X	X	MODEL THE TRIP ON ITS CURRENT TRIP LINK
SUPMAC	PLI	M	M	X	M	TRIP LINK PROMPT EVENT TEST
SUTEST	FORT	I	X	X	X	TRIP LINK ENTRY TESTING
SZHDR	FORT	R	X	X	X	WRITE SAMPLING HEADER RECORD
SZINT	FORT	R	X	X	X	CALCULATE INTEGRALS, AVERAGES & MISC. STATS.
SZSTAT	FORT	R	X	X	X	COLLECT STATISTICS
SZSTATE	FORT	I	X	I	I	COLLECT STATISTICS ON THOSE ENTERING A STATE
SZSTATEN	FORT	I	X	I	I	COLLECT STATISTICS ON THOSE ENTERING STN STATE
SZSTATES	FORT	I	X	I	I	COLLECT STATISTICS ON THOSE ENTERING SL STATE
SZSTATET	FORT	I	X	I	I	COLLECT STATISTICS ON THOSE ENTERING TL STATE
SZSTATL	FORT	I	X	I	I	COLLECT STATISTICS ON THOSE LEAVING A STATE
SZSTATLN	FORT	I	X	I	I	COLLECT STATISTICS ON THOSE LEAVING STN STATE
SZSTATLS	FORT	I	X	I	I	COLLECT STATISTICS ON THOSE LEAVING SL STATE

Table 1-2. DSM Model Processor (Page 3 of 3)

NAME	LIB	TYPE	PDL	C.D.	PREAM	DESCRIPTION
SZSTATLT	FORT	I	X	I	I	COLLECT STATISTICS ON THOSE LEAVING TL STATE
SZZERO	FORT	R	X	X	X	RESET STATISTICS
TIMES	ASM	E	E	E	E	GETS DATE & TIME FROM SYSTEM CLOCK(EP-DTIMEL)
TRACBK	ASM	E	E	E	E	XTRACBK-MAIN ENTRY
TRCBKP	FORT	E	E	E	E	XTRCBKP-MAIN ENTRY
TRCKI	FORT	E	E	E	E	PRINTS HEADING LINE(EP-TRCBKP)
TRCBKV	FORT	E	E	E	E	PRINTS TWO LINES FOR AN ARGUMENT(EP-TRCBKP)
TRCBKR	FORT	E	E	E	E	PRINTS THREE LINES FOR GEN.REG.(EP-TRCBKP)
UNDO	ASM	M	YM	YM	YM	STANDARD REGISTER RESTORE MACRO
VRAND	PLI	M	M	X	M	GENERATE A UNIFORMLY DISTRIBUTED RANDOM NUMBER
VRANDN	PLI	M	M	X	M	GENERATE A NORMALLY DISTRIBUTED RANDOM NUMBER
XGDIPF4	FORT	R	Y	Y	X	READ VARIABLES THAT ARE 4 BYTES LONG
XGDIPH4	FORT	R	Y	Y	X	READ VARIABLES THAT ARE 2 BYTES LONG
XGDIPX4	FORT	R	Y	Y	X	READ VARIABLES THAT ARE 1 BYTE LONG
XNDBOF	FORT	R	Y	Y	X	READ GDIP-FORMATTED DATA FROM FT05
XPSEUDO	ASM	R	Y	Y	X	PROVIDE PSEUDO-I/O
XTRACBK	ASM	R	Y	Y	X	PERFORM TRACEBACK
XTRCBKP	FORT	R	Y	Y	X	PRINTS LINE DESCRIBING CALLING ROUTINE

NOTATIONS:

* = DOES NOT INCLUDE PREAMBLE CODE SEGMENTS(XXXXXX0)

TYPE:
- R = ROUTINE
- I = INCLUDED MEMBER
- M = MACRO (PLI OR ASM)
- C = COMMON AREA DEFINITION (WHICH IS INCLUDED)
- E = ENTRY POINT (IN ROUTINE WHOSE NAME IS GIVEN IN DESCRIPTION)

PDL:
- X = PDL GIVEN
- I = PDL NOT GIVEN, SINCE IT INCLUDED MEMBER AND TREATED IN THE PDL OF WHAT IT IS INCLUDED IN
- E = PDL NOT GIVEN, SINCE IT IS AN ENTRY POINT
- M = PDL NOT GIVEN, SINCE IT IS A MACRO
- Y = PDL NOT GIVEN, SINCE IT IS EXISTING CODE

COMPONENT DESCRIPTION:
- X = COMPONENT DESCRIPTION GIVEN
- I = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT INCLUDED MEMBER AND TREATED IN THE COMPONENT DESCRIPTION OF WHAT IT IS INCLUDED IN
- C = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS A COMMON AREA DEFINITION
- E = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS AN ENTRY POINT
- M = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS A MACRO
- Y = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS EXISTING CODE

PREAMBLE:
- X = PREAMBLE GIVEN
- I = PREAMBLE NOT GIVEN, SINCE IT INCLUDED MEMBER AND TREATED IN THE PREAMBLE OF WHAT IT IS INCLUDED IN
- C = PREAMBLE NOT GIVEN, SINCE IT IS A COMMON AREA DEFINITION
- E = PREAMBLE NOT GIVEN, SINCE IT IS AN ENTRY POINT
- M = PREAMBLE NOT GIVEN, SINCE IT IS A MACRO
- Y = PREAMBLE NOT GIVEN, SINCE IT IS EXISTING CODE

Table 1-3. DSM Output Processor (Page 1 of 2)

DSM - OUTPUT PROCESSOR --- CODE SEGMENTS & ENTRY POINTS (*)

NAME	LIB	TYPE	PDL	C.D.	PREAM	DESCRIPTION
ABIN	FORT	E	E	E	E	ZABIN-MAIN ENTRY
ATYPE	ASM	M	YM	YM	YM	STD MACRO GENERATION LIMIT MACRO
BNCHK	FORT	E	E	E	E	ZBNCHK-MAIN ENTRY
CALLS	ASM	M	YM	YM	YM	STD ASSEMBLY LANG ROUTINE LINKAGE MACRO
CKFOLLOW	PLI	M	M	X	M	CHECK FOLLOWER RECORD
DAYTIM	FORT	R	X	X	X	CONVERT DATE & TIME TO YY/MM/DD/HH/MM/SS
DBIN	FORT	E	E	E	E	ZDBIN-MAIN ENTRY
DBUG	PLI	M	M	X	M	WRITE INTERMEDIATE OUTPUT
DO	ASM	M	YM	YM	YM	STANDARDIZED REGISTER SAVE MACRO
DTIMEL	ASM	R	Y	X	X	GET DATE & TIME FROM SYSTEM
DUMBIN	FORT	E	E	E	E	ZDUMBIN-MAIN ENTRY
ENTER	ASM	M	YM	YM	YM	STANDARD ASSEMBLY LANGUAGE ENTRY MACRO
ERROR	FORT	E	E	E	E	ZERROR-MAIN ENTRY
GRAPH	FORT	E	E	E	E	ZGRAPH-MAIN ENTRY
HEADER	FORT	E	E	E	E	SHEADER-MAIN ENTRY
HIST	FORT	E	E	E	E	SHIST-MAIN ENTRY
LBL	ASM	M	YM	YM	YM	STANDARD ASSEMBLY LANGUAGE TEST FOR 0 MACRO
LEAVE	ASM	M	YM	YM	YM	STANDARD ASSEMBLY LANGUAGE EXIT MACRO
LIST	FORT	E	E	E	E	SLIST-MAIN ENTRY
MNMX	FORT	E	E	E	E	SMNMX MAIN ENTRY
PSEUDO	ASM	E	E	E	E	XPSEUDO-MAIN ENTRY
READ02	FORT	E	E	E	E	SREAD02-MAIN ENTRY
READ03	FORT	E	E	E	E	SREAD03-MAIN ENTRY
READ04	FORT	E	E	E	E	SREAD04-MAIN ENTRY
REQTLU	FORT	E	E	E	E	SREQTLU-MAIN ENTRY
SETUP	FORT	E	E	E	E	SSETUP-MAIN ENTRY
SHIFT	FORT	E	E	E	E	ZSHIFT-MAIN ENTRY
SHIST	FORT	R	X	X	X	OUTPUT HISTOGRAM OF DATA
SKIPFO	FORT	E	E	E	E	ZSKIPFO-MAIN ENTRY
SLIST	FORT	R	X	X	X	LIST ITEMS OR OUTPUT SUMMARY
STOFLO	FORT	E	E	E	E	EP-ZSTORE
STORE	FORT	E	E	E	E	ZSTORE-MAIN ENTRY
SUDOGO	ASM	E	E	E	E	INITIALIZE PSEUDO-I/O (EP-XPSEUDO)
SODATA	FORT	B	X	X	B	INITIALIZE MAJOR COMMON AREAS
SODCLS	FORT	C	X	C	C	DECLARE MAJOR COMMON AREAS & PARAFOR
SODEFS	FORT	C	X	C	C	DECLARE MAJOR COMMON AREAS
SONTIX	ASM	R	X	X	X	ESTAB.PARM FIELD ADDRESSIBILITY
SOUPTX	ASM	E	E	E	E	PASS MEMBER NAME TO SOWTIX(EP-SONTIX)
SOPSUM	FORT	R	X	X	X	PERFORMANCE SUMMARY PROCESSING
SOUTPT	FORT	R	X	X	X	DSM OUTPUT PROCESSOR CONTROL
SOUTPT1	FORT	I	I	I	I	PROCESS A DATA REQUEST
SOUTPT2	FORT	I	I	I	I	PERFORMANCE SUMMARY FILE PROCESSING
SOWTIX	FORT	R	X	X	X	PARSE PARM LIST & WRITE LOAD MODULE NAME
SOWTIW	FORT	E	E	E	E	LIST USED MEMBERS IN INDEX
SOWTIY	FORT	E	E	E	E	WRITE PERSUM MEMBER NAME IN PERSUM
SOZNIT	FORT	R	X	X	X	INITIALIZATION OF OUTPUT PROCESSOR
SPIEL	ASM	E	E	E	E	INTERFACE TO INTERRUPT HANDLER(EP-TRACBK)
SREAD02	FORT	R	X	X	X	READ SYSTEM STATISTICS
SREAD03	FORT	R	X	X	X	READ STATION LINK STATISTICS
SREAD04	FORT	R	X	X	X	READ TRIP LINK STATISTICS
SREQTLU	FORT	R	X	X	X	RECORD/REQUEST CORRELATION
SSETUP	FORT	R	X	X	X	INITIALIZE OP DATA TABLES
SZPLOT	FORT	R	X	X	X	PLOT OUTPUT CONTROL
SZREAD	FORT	R	X	X	X	DATA ACQUISITION OF SYSTEM CONSTANTS
TIMES	ASM	E	E	E	E	GETS DATE & TIME FROM SYSTEM CLOCK(EP-DTIMEL)
TRACBK	ASM	E	E	E	E	XTRACBK-MAIN ENTRY
TRCBKP	FORT	E	E	E	E	XTRCBKP-MAIN ENTRY
TRCKI	FORT	E	E	E	E	PRINTS HEADING LINE(EP-TRCBKP)
TRCBKV	FORT	E	E	E	E	PRINTS TWO LINES FOR AN ARGUMENT(EP-TRCBKP)
TRCBKR	FORT	E	E	E	E	PRINTS THREE LINES FOR GEN.REG.(EP-TRCBKP)
UNDO	ASM	M	YM	YM	YM	STANDARD REGISTER RESTORE MACRO

Table 1-3. DSM Output Processor (Page 2 of 2)

XPSEUDO	ASM	R	Y	Y	X	PROVIDE PSEUDO-I/O
XTRACBK	ASM	R	Y	Y	X	PERFORM TRACEBACK
XTRCBKP	FORT	R	Y	Y	X	PRINTS LINE DESCRIBING CALLING ROUTINE
ZABIN	FORT	R	X	X	X	BIN REALLOCATION
ZBINL	FORT	F	X	X	X	GET LENGTH OF DATA IN BIN
ZBNCHK	FORT	R	X	X	X	BIN EXPANSION
ZCAMSG	FORT	C	X	C	C	ERROR MESSAGE COMMON
ZDBIN	FORT	R	X	X	X	ALLOCATE BIN STORAGE
ZDUMBIN	FORT	R	X	X	X	PRINT CONTENTS OF BIN AREA FOR DEBUG
ZERROR	FORT	R	X	X	X	WRITE ERROR MSG AND CONT/TERM
ZFLAG	FORT	R	X	X	X	INTERMED.OUTPUT FLAG SETTING
ZGRAPH	FORT	R	X	X	X	PRODUCE TIME SERIES PLOT
ZHEADER	FORT	R	X	X	X	READ NEXT HEADER RECORD
ZHIST	FORT	R	X	X	X	HISTOGRAM OUTPUT CONTROL
ZLIST	FORT	R	X	X	X	LIST OUTPUT CONTROL
ZMNMX	FORT	R	X	X	X	COMPUTE MINIMUM AND MAXIMUM VALUES
ZODCLS	FORT	C	X	C	C	DECLARE VARIABLES GLOGAL TO OP'S
ZPLOT	FORT	E	E	E	E	SZPLOT-MAIN ENTRY
ZRCLEAN	FORT	R	X	X	X	RESET BIN ADDRESSES
ZREAD	FORT	E	E	E	E	SZREAD-MAIN ENTRY
ZREQU	FORT	R	X	X	X	REQUEST HANDLING
ZSHIFT	FORT	R	X	X	X	REALLOCATE BIN STORAGE ASSIGNMENTS
ZSKIPFO	FORT	R	X	X	X	SKIP A FOLLOWER RECORD
ZSTORE	FORT	R	X	X	X	STORE DATA IN BIN
ZSYSMAX	FORT	C	X	C	C	COMPILE TIME MAXIMA

NOTATIONS:

* = DOES NOT INCLUDE PREAMBLE CODE SEGMENTS(XXXXXX0)

TYPE:
- R = ROUTINE
- I = INCLUDED MEMBER
- M = MACRO (PLI OR ASM)
- C = COMMON AREA DEFINITION (WHICH IS INCLUDED)
- B = BLOCK DATA SUBPROGRAM
- E = ENTRY POINT (IN ROUTINE WHOSE NAME IS GIVEN IN DESCRIPTION)
- F = FUNCTION SUBPROGRAM

PDL:
- X = PDL GIVEN
- I = PDL NOT GIVEN, SINCE IT INCLUDED MEMBER AND TREATED IN THE PDL OF WHAT IT IS INCLUDED IN
- E = PDL NOT GIVEN, SINCE IT IS AN ENTRY POINT
- M = PDL NOT GIVEN, SINCE IT IS A MACRO
- Y = PDL NOT GIVEN, SINCE IT IS EXISTING CODE

COMPONENT DESCRIPTION:
- X = COMPONENT DESCRIPTION GIVEN
- I = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT INCLUDED MEMBER AND TREATED IN THE COMPONENT DESCRIPTION OF WHAT IT IS INCLUDED IN
- C = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS A COMMON AREA DEFINITION
- E = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS AN ENTRY POINT
- M = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS A MACRO
- Y = COMPONENT DESCRIPTION NOT GIVEN, SINCE IT IS EXISTING CODE

PREAMBLE:
- X = PREAMBLE GIVEN
- I = PREAMBLE NOT GIVEN, SINCE IT INCLUDED MEMBER AND TREATED IN THE PREAMBLE OF WHAT IT IS INCLUDED IN
- C = PREAMBLE NOT GIVEN, SINCE IT IS A COMMON AREA DEFINITION
- E = PREAMBLE NOT GIVEN, SINCE IT IS AN ENTRY POINT
- M = PREAMBLE NOT GIVEN, SINCE IT IS A MACRO
- Y = PREAMBLE NOT GIVEN, SINCE IT IS EXISTING CODE
- B = PREAMBLE NOT GIVEN, SINCE IT IS SIMPLY A BLOCK DATA SUBPROGRAM

Additionally a programmer reading this manual is expected to be familiar with the User's Manual; card formats, error messages, and the like which are given in the User's Manual are not repeated here.

Overlay segments exist in the DSM-IP and MP, but only for the purpose of obtaining common area starting and ending addresses for checkpointing. No code segments in any processor are overlaid. These data overlays will be discussed below.

This section identifies the programming languages and system support software used in developing the DSM.

<u>System Control Program</u>

1. IBM OS/VS2 (SVS or MVS options)

    o Time Sharing Option (TSO)

<u>Compilers/Editor</u>

1. FORTRAN IV (H Extended)
2. Assembler (H)
3. PL/1 Optimizer
4. Linkage Editor

<u>Others</u>

1. OS/VS2 Utilities
2. PARAFOR (Structured FORTRAN)
3. Structuring Programming Facility (TSO-3270)

The following computing system hardware is required:

<u>Central Processing Unit</u>

IBM System 370 Model 145 or 148 processing capability at a minimum.

<u>High-Speed Core Storage</u>

Approximately 330k bytes of problem core is required for the Model Processor. Note that these figures do not include System Control Program core requirements, which can vary between 300,000 bytes and 2 million bytes.

Direct-Access Storage

Storage requirements for various functional areas of DSM are given below, in units of IBM 2314 cylinders (approximately 144,000 bytes).

1. Program Development Libraries -- 20 cylinders

2. Input from Data Base (per configuration) -- 2 cylinders

3. Trip File (1 hour of 1,000 trips assumed) -- .2 cylinders

4. Vehicle Files -- 2 cylinders

5. Checkpoint Files -- 10 cylinders per file

6. Raw statistical Output (assuming one hour run, one minute sampling interval) -- 10 cylinders.

Magnetic Tape

DSM has no explicit requirements for magnetic tape storage, but it may be a preferrable medium over direct-access storage for the following files:

1. Input from data base (2 cylinders)

2. Checkpoint Files (10 cylinders per file)

3. Raw Statistical Output (10 cylinders for 3,600 samples)

4. Trip/Vehicle Sequence File.

The choice of tape over disk will be based primarily on the amount of disk space available, the frequency of access required, and the operational procedures at the computing center being used. For planning purposes, a 2,400 foot reel of tape recorded at 1,600 bytes/inch has a capacity equivalent to 329 cylinders of 2314 disk space.

Unit Record Equipment

DSM will require a card reader for input data and a high-speed printer for output.

Display Terminal

The IBM 3270 Display Terminal or equivalent or a standard printer type terminal is required for online file edit and online job submission.

Storage and processing allocation for the three processors is as follows:

DSM Input Processor

- Programs 118k
- Data 50k

DSM Model Processor

- Programs 247k
- Data 67k

DSM Output Processor

- Programs 110k
- Data 189k

SECTION 2. PROGRAM DESCRIPTION

Figures 2-1, 2-2, and 2-3 contain control tree overviews of the code segments in the IP, MP, and OP, respectively.

The diagrams in Appendix B illustrate the DSM high-level design through the use of Hierarchy plus Input-Process-Output (HIPO) diagrams. The Visual Table of Contents illustrates program organization and contains the names and identification numbers of the detail Input-Process-Output diagrams that define the processing to be performed. These diagrams should be used in conjunction with the Process Design Language (PDL) descriptions contained in Appendix A, which provide descriptions of the program design in greater detail. Where the Visual Table of Contents and Input-Process-Output diagrams reference a segment name and identification number, that segment is further expanded in both an Input-Process-Output diagram (having that identification number) and a PDL segment (having that segment name). If an Input-Process-Output diagram references a function by segment name only, then the design of that segment will be found in the PDL segment having that segment name. These HIPO diagrams are intended to supply a high level introductory description of the processing; PDL and component descriptions provide the detail.

The following three sections give an overview of the three processors.

2.1 INPUT PROCESSOR

The Input Processor (IP) reads all user input data and builds structured data files for Model Processor (MP) use. The user input data is one or more of four types:

- o System characteristics
- o Runtime data
- o Trip demand
- o Vehicle demand

```
INPUT PROCESSOR

 SACOMN                    ORDER INPUT COMMONS

 SIPARM                    SAVE PARAMETER LIST
   |
 SINPUT                    INPUT PROCESSOR CONTROL
   |
   |---SPIEL               INITIALIZE PGM INTERRUPT HANDLING
   |
   |---NDBOR               READ GDIP DATA FORMATS
   |     |
   |     |---GDIP4         READ GDIP DATA
   |           |
   |           |---GDIPF4  READ FULLWORD DATA
   |           |---GDIPH4  READ HALFWORD DATA
   |           |---GDIPX4  READ BYTE SIZE DATA
   |           |---ERROR   WRITE ERROR MESSAGE
   |
   |---SINPUT1             READ SYSTEM CHARACTERISTICS
   |---SINPUT2             READ RUNTIME DATA
   |     |
   |     |---SAFLAG        SET INTERMEDIATE OUTPUT FLAGS
   |           |
   |           |---ERROR   WRITE ERROR MESSAGE
   |
   |---SIINIT              INPUT INITIALIZATION
   |     |
   |     |---LODCOM        LOAD ADDRESS AND LENGTH OF SYSTEM CHARS
   |     |---SIADDR        PASS SYS CHAR ADDR AND LGTH TO SISADD
   |     |     |
   |     |     |---SISADD  SAVE ADDRESS AND LENGTH OF SYS CHAR
   |     |
   |     |---SIPLST        PROCESS PARAMETER LIST
   |     |     |
   |     |     |--SIMNAM   PARSE PARAMETER LIST
   |     |
   |     |---SIPSAV        COMMON FOR ADDRS OF SYS CHAR AND IP COMMONS
   |
   |---SINPUT3             READ TRIP DEMAND DATA
   |
   |---SITDGN              TRIP DEMAND GENERATION
   |     |
   |     |---SITDGN2       VERIFY AND INITIALIZE INPUT
   |     |---SITDGN3       WRITE ERROR MESSAGES
   |     |---SITDGN4       WRITE TRIP GEN. SUMMARY
   |     |
   |     |---SICUMP        BUILD CUMULATIVE PROBABILITY DISTRIBUTION
   |     |---SMRNG         GEN RANDOM NUMBER BETWEEN 0-1
   |     |---SMRSEL        SELECT RANDOM POINT ON DISTRIBUTION
   |     |     |
   |     |     |---SMRNG   GENERATE RANDOM NUMBER
   |     |
   |     |---ERROR         WRITE ERROR MESSAGES
   |
   |---SINPUT4             READ VEHICLE DEMAND DATA
   |     |---SINPUT4A      READ NEXT STOP SELECTION DATA
   |     |---SINPUT4B      READ INTERARRIVAL TIME DATA
   |     |---SINPUT4C      READ TRIP SIZE DATA
   |
   |---SIVDGN              VEHICLE DEMAND GENERATION
   |     |
   |     |---SIVDGN2       VERIFY AND INITIALIZE INPUT
   |     |     |--SIVDGN2A CONVERT PROBABILITY DISTRIBUTIONS
   |     |     |--SIVDGN2B WRITE ERROR MESSAGES
   |     |
   |     |---SIVDGN4       GENERATE SCHEDULED VEHICLES
   |     |---SIVDGN5       GENERATE DEMAND RESPONSIVE VEHICLES
   |     |---SIVDGN6       GENERATE ONBOARD TRIPS
   |     |---SIVDGN7       WRITE VEHICLE GENERATION SUMMARY
```

Figure 2-1. Input Processor (Page 1 of 3)

```
|         |
|         |---SICUMP        BUILD CUMULATIVE PROBABILITY DISTRIBUTION
|         |
|         |---SIGIAT        GET NEXT VEHICLE INTERARRIVAL TIME
|         |
|         |         |
|         |         |---SMRNG   GEN RANDOM NUMBER BETWEEN 0-1
|         |         |---SMRSEL  SELECT RANDOM ENTRY IN CUM. PROB. DIST.
|         |               |-SMRNG GEN RANDOM NUMBER BETWEEN 0-1
|         |
|         |---SMRNG         GEN RANDOM NUMBER BETWEEN 0-1
|         |
|         |---SMRSEL        SELECT RANDOM ENTRY IN CUM. PROB. DIST.
|         |         |
|         |         |---SMRNG   GENERATE RANDOM NUMBER
|         |
|         |---ERROR         WRITE ERROR MESSAGE
|
|---SISCFG                  STATION CONFIGURATOR
|         |
|         |---SISCFG1       ESTABLISH NUMBERS FOR LINK TYPES
|         |---SISCFG2       MISCELLANEOUS ERROR CHECKS
|         |---SISCFG3       BUILD STRUCTURED TABLES
|         |---SISCFG4       BUILD UPSTREAM POINTERS
|         |
|         |         |
|         |         |--SISCFG4A BUILD INPUT QUEUE  US LINKS
|         |         |--SISCFG4B BUILD DOCK US LINKS
|         |         |--SISCFG4C BUILD OUTPUT QUEUE US LINKS
|         |         |--SISCFG4D BUILD OUTPUT RAMP US LINKS
|         |         |--SISCFG4E BUILD STORAGE US LINKS
|         |         |--SISCFG4F BUILD DOCK TO STORAGE US LINKS
|         |         |--SISCFG4G COMPUTE DL US LINKS
|         |         |--SISCFG4H BUILD MOA US LINKS
|         |
|         |---SISCFG5       BUILD DOWNSTREAM STATION LINKS
|         |
|         |         |
|         |         |--SISCFG5A BUILD INPUT RAMP DS LINKS
|         |         |--SISCFG5B BUILD INPUT QUEUE DS LINKS
|         |         |--SISCFG5C BUILD DOCK DS LINKS
|         |         |--SISCFG5D BUILD OUTPUT QUEUE DS LINKS
|         |         |--SISCFG5E BUILD STORAGE TO INPUT DS LINKS
|         |         |--SISCFG5F BUILD UL DS LINKS
|         |         |--SISCFG5G BUILD MIB DS LINKS
|         |
|         |---ERROR         WRITE ERROR MESSAGE
|
|---SICHCK                  PARAMETER CHECKING AND INITIALIZATION
|         |
|         |---SICHCK1       VERIFY CERTAIN LINK/EVENT COMBINATIONS
|         |---SINERR        CALL ERROR MESSAGE ROUTINE
|         |         |
|         |         |---ERROR   WRITE ERROR MESSAGE
|         |
|         |---ERROR         WRITE ERROR MESSAGE
|
|  ---SIREPT                 INITIAL CONDITIONS REPORT
|         |
|         |---SIREPT10      DATA FORMATS
|         |---SIREPT2       LOCAL DATA DEFINITIONS
|         |---SIREPT3       WRITE TRIP AND VEHICLE CHARACTERISTICS
|         |---SIREPT4       WRITE SERVICE CHARACTERISTICS SUMMARY
|         |         |
|         |         |--SIREPT4A WRITE VEHICLE SPACING AND HEADWAY DATA
|         |         |--SIREPT4B WRITE EMPTY VEHICLE POLICY DATA
|         |         |--SIREPT4C WRITE SIMULATION CONTROL DATA
|         |         |--SIREPT4D WRITE ROUTE ASSIGNMENT DATA
|         |         |
|         |---SIREPT5       WRITE STATION LINK SUMMARY
|         |         |
|         |         |--SIREPT5A STATION LINK DATA
|         |         |--SIREPT5B VALIDATE DOWNSTREAM LINKS AND DIVERGE FNS
```

Figure 2-1. Input Processor (Page 2 of 3)

```
|        |          |--SIREPT5C LINK CHARACTERISTICS CHECK
|        |          |--SIREPT5D MULTIPLE UPSTRM/DNSTRM LINKS + EVENTS WRITE
|        |          |
|        |---SIREPT6         WRITE TRIP LINK SUMMARY
|        |
|        |---SICUMP          CONVERT TO CUMULATIVE PROBABILITY DIST.
|        |
|        |---SINERR          CALL ERROR MESSAGE ROUTINE
|        |          |
|        |          |---ERROR    WRITE ERROR MESSAGE
|        |
|        |---ERROR           WRITE ERROR MESSAGE
|
|---SIBWRT                   WRITE STRUCTURED DATA FILES
|
|---SINERR                   CALL ERROR MESSAGE ROUTINE
|        |
|        |---ERROR           WRITE ERROR MESSAGE
|
|---SIWNAM(EP-SIMNAM)        LIST FILES IN INDEX
|---ERROR                    WRITE ERROR MESSAGE
|
----------------------------------------------------------------------------
|---TRACBK                   PROCESS PROGRAM INTERRUPT
|        |
|        |---TRCBKI          PRINT INTERRUPT HEADING
|        |---TRCBKV          PRINT 2 LINES FOR ARGUMENT
|        |---TRCBKR          PRINT 3 LINES FOR GENERAL REGISTERS
|        |
----------------------------------------------------------------------------
NOTES:
  ALL SUBROUTINES ARE IDENTIFIED BY THE ENTRY POINT USED BY THE
  CALLING SUBROUTINE.
  NO PREAMBLES, COMMONS, OR INCLUDED SEGMENTS WITH DATA DEFINITIONS
  ONLY ARE LISTED. (SIPSAV IS AN EXCEPTION TO THIS SINCE IT IS
  BOTH A COMMON DEFINED IN SIINIT AND AN ENTRY POINT IN SIPSAV.)
  ALL INCLUDED CODE SEGMENTS ARE IDENTIFIED BY THE NAME OF THE
  INCLUDING SUBROUTINE PLUS A SUFFIX.
```

Figure 2-1. Input Processor (Page 3 of 3)

```
MODEL PROCESSOR CONTROL GRAPH OF MAJOR COMPONENTS

SANTIX                                  <INIT MEMBER STRING FOR INDEX>
   |
   |---SAMAIN                           <MAIN CONTROL LOOP>
         |
         |---SAINIT                     <SYSTEM INITIALIZATION>
         |     |
         |     |---SANTSA               <INIT SYST.STATUS AREA ADDRESSES>
         |     |     |
         |     |     |---SANSAV         <INIT CKPT & SYST.DATA READ>
         |     |           |
         |     |           |---SADADD   <INIT INPUT AREA ADDRS & MSGS>
         |     |           |---SACKR    <INIT CKPT/RESTART PROCESS>
         |     |
         |     |---SAREST(E.P. IN SACKR)  <PERFORM RESTART>
         |     |     |
         |     |     |---SAPFEL         <SCHEDULE XTN ON FEL>
         |     |
         |     |---SAUPTX(E.P. IN SANTIX) <PASS MEMBER NAME STRING>
         |     |     |
         |     |     |---SAWTIX         <PARSE PARM LIST>
         |     |           |
         |     |           |---DAYTIM   <CONVERT DATE & TIME>
         |     |                 |
         |     |                 |---DTIMEL   <GET DATE & TIME FM SYSTEM>
         |     |
         |     |---SANDTA(E.P.IN SADADD) <READ INPUT INTO INPUT COMMONS>
         |     |---SANFEL               <INIT FEL>
         |     |---SANXTN               <INIT TRANSACTIONS>
         |     |---SANMDL               <INIT INTERNAL MODEL COMMONS>
         |     |---SAZNIT               <INIT STATISTICS COMMON>
         |           |
         |           |---SZZERO         <RESET STATISTICS>
         |
         |---SASCTL                     <CONTROL SL EVENTS>
         |     |
         |     |---SSMOD                <SL EVENT MODELING>
         |     |     |
         |     |     |---SSMODB         <VEH PROCESSING BEFORE SL EVENT>
         |     |     |     |
         |     |     |     |---SMDETR   <DETRAIN PROCESSING>
         |     |     |     |---SMBRD    <PLAN BOARDING>
         |     |     |     |     |
         |     |     |     |     |---SMRSEL  <SELECT PT FROM CUMM.DIST.>
         |     |     |     |           |
         |     |     |     |           |---SMRNG   <GEN UNIFORM RAND.NO.>
         |     |     |     |
         |     |     |     |---SMDBRD   <PLAN DEBOARDING>
         |     |     |     |     |
         |     |     |     |     |---SMRSEL  <SELECT PT FROM CUMM.DIST.>
         |     |     |     |
         |     |     |     |---SMLTIM   <CALC.LAUNCH DELAY>
         |     |     |     |
         |     |     |     |---SZSTAT   <COLLECT STATISTICS>
         |     |     |
         |     |     |
         |     |     |---SSMODN         <FIND NEXT SL EVENT>
         |     |     |     |
         |     |     |     |---SZSTAT   <COLLECT STATISTICS>
         |     |     |
         |     |     |---SSMODA         <VEH PROCESSING AFTER SL EVENT>
         |     |           |
         |     |           |---SSPMAC   <SL PROMPT TEST>
         |     |           |---SUPMAC   <TL PROMPT TEST>
         |     |           |---SMENTR   <ENTRAIN PROCESSING>
         |     |           |---SMNXST   <FIND NEXT STATION>
         |     |           |     |
         |     |           |     |---SMEVM    <EMPTY VEH MANAGEMENT>
         |     |           |                |
```

Figure 2-2. Model Processor (Page 1 of 3)

```
|     |           |           |---SMRSEL <SELECT PT RANDOMLY>
|     |           |
|     |           |---SMTABQ   <TRIP AT BOARD QUEUE PROCESS>
|     |           |     |
|     |           |     |---SSPMAC   <SL PROMPT TEST>
|     |           |     |---SZSTAT   <COLLECT STATISTICS>
|     |           |
|     |           |---SZSTAT   <COLLECT STATISTICS>
|     |
|     |---SSTEST               <SL ENTRY TESTING>
|     |     |
|     |     |---SMDIVF         <DIVERGE FUNCTIONS>
|     |     |     |
|     |     |     |---SMDIVS   <SEARCH FOR SL OF GIVEN TYPE>
|     |     |     |---SMDIVO   <ORDER SLS>
|     |
|     |---SSLEAV               <SL LEAVE PROCESSING>
|     |     |
|     |     |---SSMOD          <SL EVENT MPODELING>
|     |     |---SSPMAC         <SL PROMPT TESTING>
|     |
|     |---SZSTAT               <COLLECT STATISTICS>
|
|---SAUCTL                     <TL EVENT CONTROL>
|     |
|     |---SUMOD                <TL EVENT MODELING>
|     |     |
|     |     |---SZSTAT         <COLLECT STATISTICS>
|     |
|     |---SUTEST               <TL ENTRY TESTING>
|     |---SULEAV               <TL LEAVE PROCESSING>
|     |     |
|     |     |---SUMOD          <TL EVENT MODELING>
|     |     |---SUPMAC         <TL PROMPT TESTING>
|     |     |---SZSTAT         <COLLECT STATISTICS>
|     |
|     |---SMTABQ               <PROCESS TRIP AT BOARD QUEUE>
|
|---SAASYN                     <PERFORM ASYNCHRONOUS PROCESS>
|     |
|     |---NDBOR                <READ GDIP DATA>
|     |---SAFAIL               <FAILURE ACTIVITY PROCESSING>
|     |     |
|     |     |---SSPMAC         <SL PROMPT TESTING>
|     |     |---SUPMAC         <TL PROMPT TESTING>
|     |
|     |---SAFLAG               <FLAG CARD PROCESSING>
|     |---SACKPT(E.P. IN SACKR) <WRITE CKPT RECORD>
|     |---SAPFEL               <SCHEDULE XTN ON FEL>
|     |---SATORG               <MOVE ARRIVING TRIP>
|     |---SAVORG               <MOVE ARRIVING VEHICLE>
|
|---SASAMP                     <WRITE RAW STATISTICS FILE>
|     |
|     |---SAFINM               <WRITE SNAPSHOT REPORT>
|     |---SZHDR                <WRITE SAMPLE HEADER RECORD>
|     |---SZINT                <END POINT INTEGRALS>
|     |---SZZERO               <RESET STATISTICS>
|
|---SACKPT(E.P. IN SACKR)      <WRITE CHECKPOINT RECORD>
|
|---SASPRM                     <SL PROMPT PROCESSING>
|     |
|     |---SSTEST               <SL ENTRY TESTING>
|     |---SSLEAV               <SL LEAVE PROCESSING>
|     |---SSMOD                <SL EVENT MODELING>
|
|---SAUPRM                     <TL PROMPT PROCESSING>
|     |
|     |---SUTEST               <TL ENTRY TESTING>
```

Figure 2-2. Model Processor (Page 2 of 3)

```
|         |---SULEAV                    <TL LEAVE PROCESSING>
|         |---SUMOD                     <TL EVENT MODELING>
|
|---SATRD                               <READ TRIP FILE>
|---SATORG                              <MOVE ARRIVING TRIP>
|         |
|         |---SUMOD                     <TL EVENT MODELING>
|         |---SZSTAT                    <COLLECT STATISTICS>
|
|---SAVRD                               <READ VEHICLE FILE>
|---SAVORG                              <MOVE ARRIVING VEHICLE>
|         |
|         |---SSMOD                     <SL EVENT MODELING>
|         |---SZSTAT                    <COLLECT STATISTICS>
|
|---SAPFEL                              <SCHEDULE XTN ON FEL>
|         |
|         |---SZSTAT                    <COLLECT STATISTICS>
|
|---SARFEL                              <REMOVE NEXT XTN FROM FEL>
|---SAFINM                              <WRITE FINAL MODEL REPORT>
|---SAFINS                              <WRITE FINAL SYSTEM REPORT>
          |
          |---SAWTIW(EP-SAWTIX)         <LIST FILES IN INDEX>

NOTES:
   1. XXX
        |         IMPLIES THAT YYY IS CALLED BY OR IS A MOJOR INCLUDED
        |---YYY     SEGMENT OF XXX
   2. THE SUBSTRUCTURE OF A COMPONENT APPEARS ONLY ONCE (I.E., IS NOT
        REPEATED EVERYWHERE THE COMPONENT APPEARS).
   3. ONLY MAJOR COMPONENTS ARE INCLUDED.
```

Figure 2-2. Model Processor (Page 3 of 3)

```
DSM OUTPUT PROCESSOR:
SONTIX                                 <ESTABLISH PARM FIELD ADDRESSABILITY>
  |---SOUTPT                           <DSM OUTPUT PROCESSOR CONTROL>
        |---SOZNIT                     <INITIALIZATION>
        |      |---ZDBIN               <ALLOCATE BIN STORAGE>
        |      |---SZREAD              <ACQUIRE SYSTEM CONSTANTS>
        |      |---ZREQU               <INITIALIZE REQUEST TABLE>
        |
        |---ZREQU                      <REQUEST HANDLING>
        |      |---ZBNCHK              <BIN EXPANSION>
        |            |---ZSHIFT        <REALLOCATE BIN STORAGE ASSIGNMENTS>
        |
        |---SZREAD                     <DATA ACQUISITION>
        |      |---SSETUP              <INITIALIZE OF DATA TABLES>
        |      |---SZREQTLU            <RECORD/REQUEST CORRELATION>
        |      |---ZHEADER             <READ HEADER RECORD>
        |      |---ZSKIPFO             <SKIP A FOLLOWER RECORD>
        |      |---SREAD02             <READ SYSTEM STATISTICS>
        |      |     |---STOFLO        <STORE DATA IN BIN>
        |      |           |---ZABIN      <BIN REALLOCATION>
        |      |                 |---ZSHIFT <REALLOCATE BIN ASSIGNMENTS>
        |      |---SREAD03             <READ STATION LINK STATISTICS>
        |      |     |---STOFLO        <STORE DATA IN BIN>
        |      |---SREAD04             <READ TRIP LINK STATISTICS>
        |      |     |---STOFLO        <STORE DATA IN BIN>
        |      |---ZRCLEAN             <RESET BIN ADDRESSES>
        |
        |---ZLIST                      <LIST OUTPUT CONTROL>
        |      |---SLIST               <LIST ITEMS OR OUTPUT SUMMARY>
        |---ZHIST                      <HISTOGRAM OUTPUT CONTROL>
        |      |---ZMNMX               <COMPUTE MINIMUM AND MAXIMUM VALUES>
        |      |---ZBNCHK              <BIN EXPANSION>
        |      |---SHIST               <OUTPUT HISTOGRAM OF DATA>
        |---SZPLOT                     <PLOT OUTPUT CONTROL>
        |      |---ZGRAPH              <PRODUCE TIME SERIES PLOT>
        |
        |---ZFLAG                      <SET INTERMEDIATE OUTPUT FLAGS>
        |---ZDUMBIN                    <DUMP BIN HEADERS>
        |---SOPSUM                     <PERFORMANCE SUMMARY PROCESSING>
        |---ZBINL                      <FIND BIN LENGTH>
        |---SOUPTX(EP-SONTIX)          <PASS MEMBER NAME INFO>
        |      |---SOWTIX              <PARSE PARM LIST>
        |            |---DAYTIM        <CONVERT DATE & TIME>
        |                  |---TIMES(EP-DTIMEL)  <FIND DATE & TIME>
        |---SOWTIW(EP-SOWTIX)          <LIST FILES IN INDEX>
        |---SOWTIY(EP-SOWTIX)          <LIST PERSUM MEMBER NAME IN MEMBER>

NOTES:
  1. XXX
      |        IMPLIES THAT YYY IS CALLED BY OR IS A MOJOR INCLUDED
      |---YYY    SEGMENT OF XXX
  2. THE SUBSTRUCTURE OF A COMPONENT APPEARS ONLY ONCE (I.E., IS NOT
       REPEATED EVERYWHERE THE COMPONENT APPEARS).
  3. ONLY MAJOR COMPONENTS ARE INCLUDED (EG., ERROR IS EXCLUDED)
  4. SEE SUBLOGIC TABLE & COMPONENTS/ENTRY POINT LIST ALSO.
```

Figure 2-3. Output Processor

The resulting structured data files are

- System characteristics
- Runtime
- Trip arrival
- Vehicle arrival
- Run index

The IP receives control via a Job Control Language cataloged procedure which provides the information necessary to access both input and output data files. Before reading any user data, the IP initializes several data items in preparation for user input or sets default values for some items in the event that no user data is supplied.

The first user data file read is the System Characteristics file which supplies the initial values (zero time data) for all system variables required by the MP and possibly processor options required by the IP. The only other valid System Characteristics data types are comments accompanying the data.

The second user data file read is the Runtime file which may contain both zero time data and time tagged data. In addition to data items, IP options and comments, the Runtime file may contain other input data types:

- Checkpoint request
- Simulation stop time
- Run index data
- Failure/repair request
- Synchronous trip
- Asynchronous vehicle
- Flag request

The IP writes run index data directly to the run index file and processes zero time items and flag requests immediately. All time tagged data and the rest of the above list, the IP writes to the structured Runtime file for later use by the MP.

The rest of the IP proceeds based upon the IP options entered in either the System Characteristics or Runtime data. If the user has entered the trip generation option, the IP reads the user's Trip Demand file and generates structured Trip Arrival file whose characteristics conform to those specified in the input file. If the user has also entered the vehicle generation option, the IP reads the user's Vehicle Demand file and generates the appropriate structured Vehicle Arrival file. If the user has specified the last available option, model setup, the IP does several things. First the IP builds the structured data tables which define the station link configuration. Next the IP converts user time input values into internal clock units and validity checks much of the System Characteristics data as it writes the Initial Conditions Report.

If no serious errors are found while processing user input, the IP writes the structured System Characteristics file. For each file that the IP writes, it also adds an entry to the Run Index file.

2.1.1 Architecture

One routine, SINPUT, controls all IP functions calling subroutines as indicated by user input. Figure 2-4 shows the hierarchy of the IP. All of the Process Design Language (PDL) describing IP functions indicated by Figure 2-4 is located in Appendix A.

All communication to and from the IP is done by data files. The user directs the IP by providing input data files containing system data and processing options. The IP in turn directs the MP by passing user data and IP generated data to the MP in structured data files. Figure 2-5 shows the relationship of the IP to its input and output data files.

```
SINPUT                          INPUT PROCESSOR CONTROL ROUTINE
  |
  |---SIINIT                    DATA INITIALIZATION
  |
  |---SITDGN                    TRIP DEMAND GENERATION
  |
  |---SIVDGN                    VEHICLE DEMAND GENERATION
  |
  |---SISCFG                    STATION CONFIGURATION
  |
  |---SICHCK                    DATA CHECKING AND INITIALIZATION
  |
  |---SIREPT                    INITIAL CONDITIONS REPORT
  |
  |---SIBWRT                    SYSTEM CHARACTERISTICS WRITE
```

Figure 2-4. Input Processor Control Hierarchy

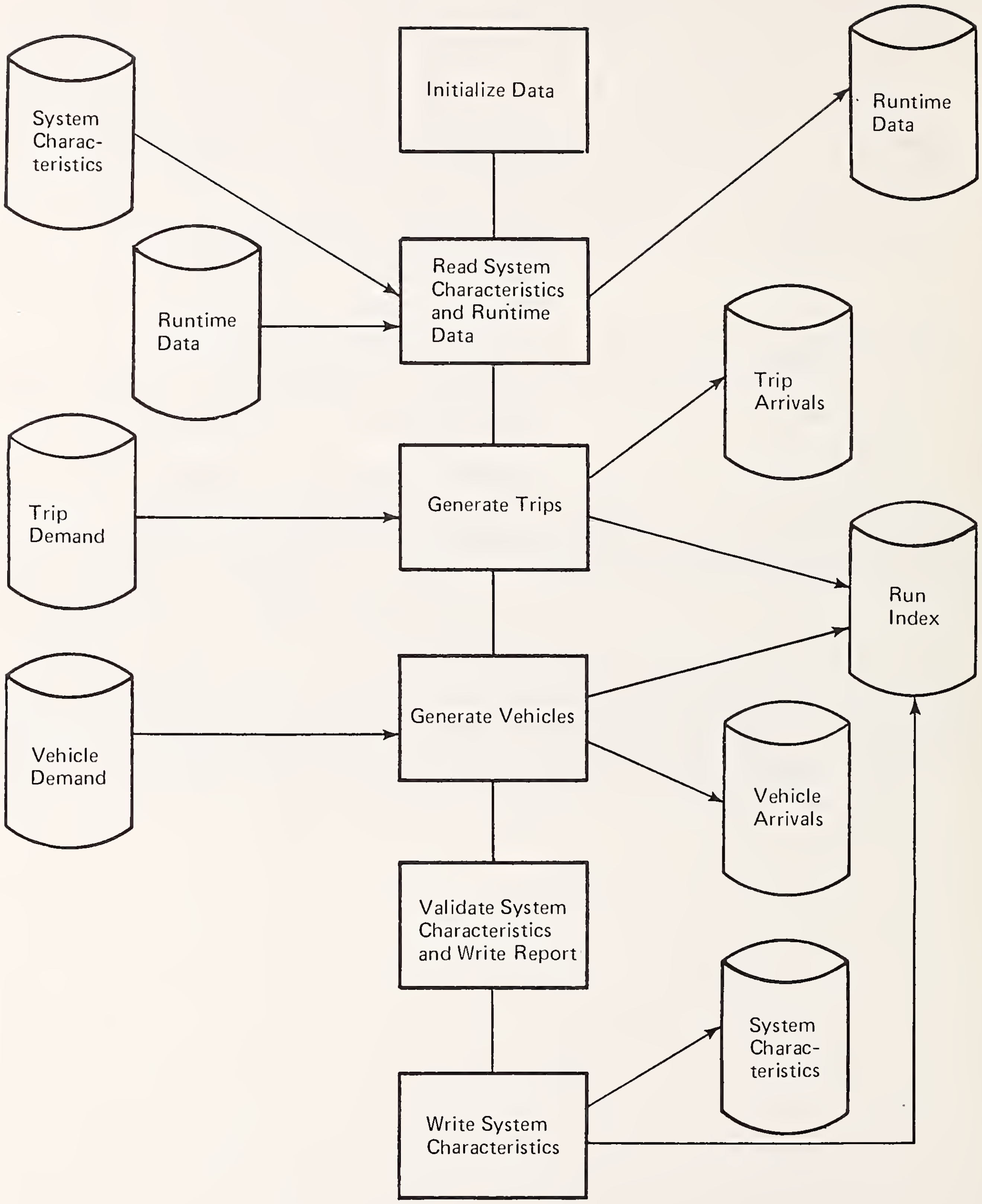


Figure 2-5. Input Processor

2.2 MODEL PROCESSOR

The DSM Model Processor provides an event processing structure for modeling the detailed operation of an automated transit system station. Events are scheduled within the simulation for occurrence or completion at some future time, in response to transaction processing requirements. Transactions are defined within the simulation environment as transit vehicles, trips, and system service requests. Transactions are appropriately processed when the event time for which they were scheduled is completed and the next event for the transaction becomes the next most imminent task to be performed in the simulation system. Transactions are rescheduled when processing for the current event of the transaction is completed and the next required event and its completion time has been determined. In the case of vehicle transactions, if the next event cannot be performed, the transaction is queued as waiting to begin its next required event. Dequeueing and rescheduling of the transaction in this case occurs in response to a system service request (scheduled as the result of another transaction event completion) becoming the next most imminent task to be performed. This concept of transactions and discrete event scheduling is more fully described in Section 2.2.1, Architecture.

Execution of the Model Processor (MP) is initiated by the invocation of a cataloged job control procedure contained in the procedure library. Upon entry, the MP performs initialization of the simulation experiment. This initialization begins with the reading of structured data files created as the result of input processing. The Model Processor Control module, SAMAIN, controls the order of processing as shown in Figure 2-6. SAMAIN invokes the lower-level segments as driven by the Future Event List.

Once the required input is read, the MP reads and updates the index file to reflect the current execution of the MP. Initialization then proceeds with establishing the event timing and control mechanism, defining required transactions and scheduling of initial system service transaction for accommodating the first trip arrival, first vehicle arrivals, and sampling asynchronous data input. Initialization for both the trip link and station link models is then performed.

Upon the completion of initialization, the basic control loop for accomplishing the recognition, scheduling, and processing of transaction events is started. This control process provides for obtaining the next transaction to be processed, updating the simulation clock, and invoking required architectural components to perform the processing as required in response to a transaction's event occurrence.

The processing components invoked by the architecture perform the processing tasks as indicated by the active transaction. This processing may cause the reading of asynchronous data input and data summarization and recording or processing within the simulation models (station links or trip links) of the simulation system. As the result of processing, the transaction which invoked the processing may be rescheduled to occur depending upon the processing task performed. Service request transactions for sampling and system checkpointing are always rescheduled to occur at a fixed interval in the simulation. Service request

```
MODEL PROCESSOR:

SAMAIN
   |
   |---SAINIT       <INITIALIZE SIMULATION>
   |
   |---SASCTL       <VEHICLE (STATION LINK) EVENT>
   |
   |---SAUCTL       <TRIP (TRIP LINK) EVENT>
   |
   |---SAASYN       <ASYNCHRONOUS EVENT>
   |
   |---SASAMP       <SAMPLING EVENT>
   |
   |---SACKPT       <PERIODIC CHECKPOINT EVENT>
   |
   |---SASPRM       <STATION LINK PROMPT EVENT>
   |
   |---SAUPRM       <TRIP LINK PROMPT EVENT>
   |
   |---SATORG       <TRIP ORIGINATION EVENT>
   |---SATRD
   |
   |---SAVORG       <VEHICLE ORIGINATION EVENT>
   |---SAVRD
   |
   |---SAFINM       <SIMULATION TERMINATION EVENT>
   |---SAFINS
   |
```

Figure 2-6. Model Processor Control Hierarchy

transactions, which are used to cause recognition of trip and vehicle arrivals or model data updates, are rescheduled to occur at the time indicated by the next asynchronous trip or data record to be processed. Transactions which are used for restarting queued transactions within the modeling subsystem are not rescheduled, but reclaimed and returned to the available pool of transactions. Reuse of the transaction depends upon operational conditions within the modeling subsystems. Vehicle transactions as described previously are rescheduled or queued depending upon whether their next event can be performed within the modeling subsystem. After transaction processing is completed, control is returned to the architecture for execution of the system control mechanism.

The end of simulation occurs in response to recognition of a termination transaction. This results in performing simulation termination activities and ending the simulation experiment.

2.2.1 Architecture

The MP architecture is designed to provide a separation between system and model dependent functions. Those functions which are system dependent represent the basic control mechanism of the simulation and serve as the fixed structural elements of the system. The relationship of these components is shown in Figure 2-7. The interface between the simulation control mechanism and model dependent function is provided via architectural components which perform system-level processing functions. Transaction flows link the system architecture and the modeling subsystem. All scheduling and manipulation of transactions is handled by the system architecture through requests made by the modeling subsystem via standard system macros. The transaction parameters and data are controlled and manipulated by the modeling subsystem. The common transaction attributes recognized and communicated between the architecture and modeling subsystem are the transaction ID, next event function (or branch ID), and the delta time increment for occurrence of the next transaction event. Thus, processing flow within the simulator is maintained with three fixed pieces of information which represent a standard control-modeling interface. The control relationship is shown in Figure 2-8.

The definition of entities within the MP is oriented toward increasing execution efficiency by limiting the amount of event scheduling which must be performed.

Station links, trip links, vehicles, and trips are designated as simulator elements. Elements are further defined by type as transactions or system entities. Defined as transactions, elements are subject to event scheduling each time processing is required. As system entities, elements are given attribute status and can only be assigned to other simulator elements; and, therefore, do not require any event scheduling. The conceptual view allows greater efficiency in simulator execution since usage demands on the event control mechanism are reduced.

Transactions within the MP are defined as either vehicles, trips, or system service requests. System service requests are used in scheduling events in the future that are not directly related to model processing events. This includes such functions as data input reading, trip arrival recognition, and sampling. Vehicle transactions are used in the architectural sense to represent requests for simulator control or model processing services. These requests may take the form of a vehicle completing a specific event such as station link travel, passenger embarkation, etc. Regardless of transaction type, control processing and flow through the simulation system is handled in the same manner by the control architecture. The distinctions made between the three types of transactions are totally model dependent.

Simulator elements, such as station links and trip links, are assignable to system transactions. Any processing performed while an entity is assigned to a transaction is totally dependent upon the organization of the station link and trip link models. These models can contain as many internal processing paths and event points as desired, provided that transaction flow back to the control program is handled according to the requirements specified above.

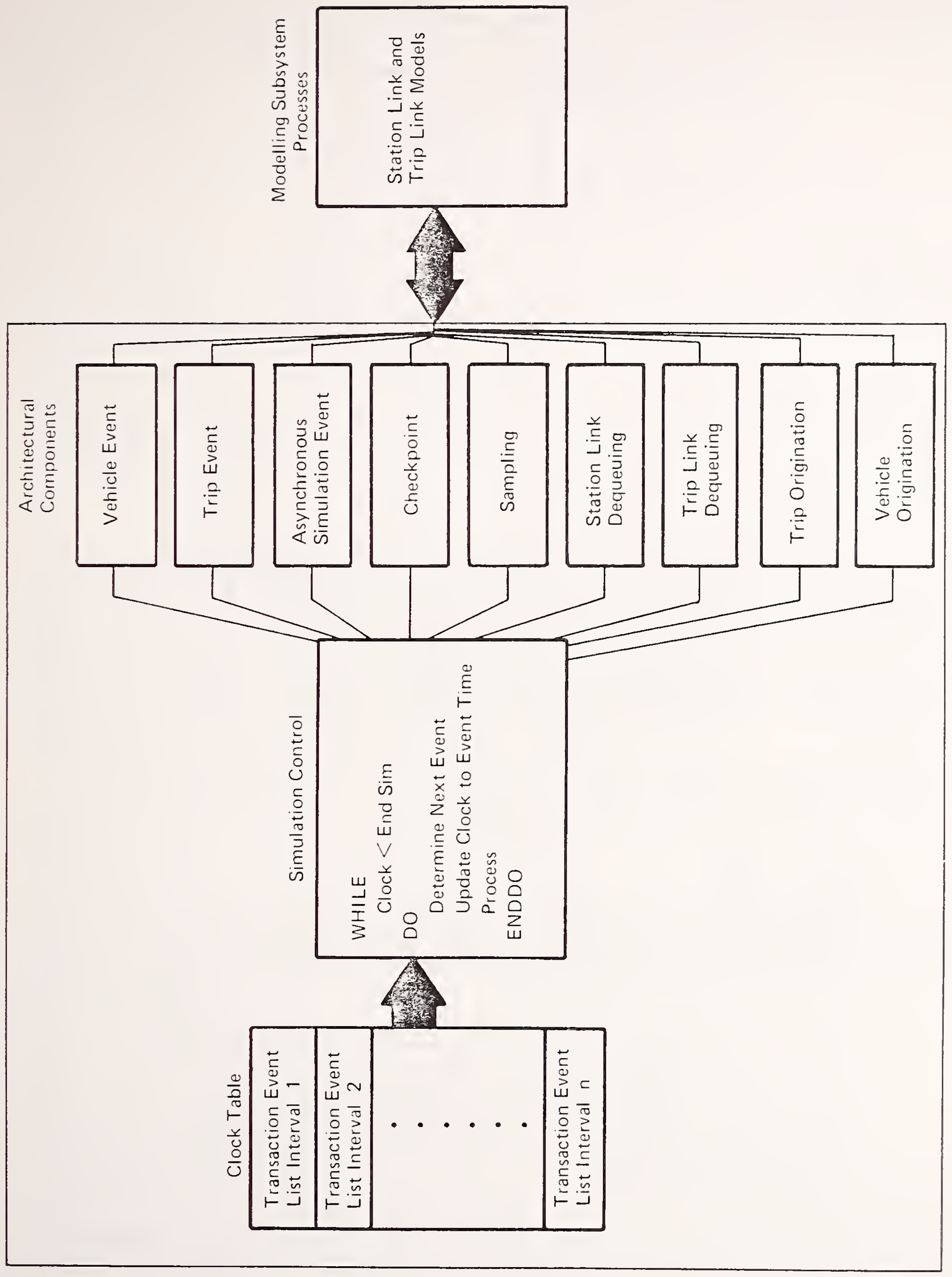


Figure 2-7. Model Processor Architecture

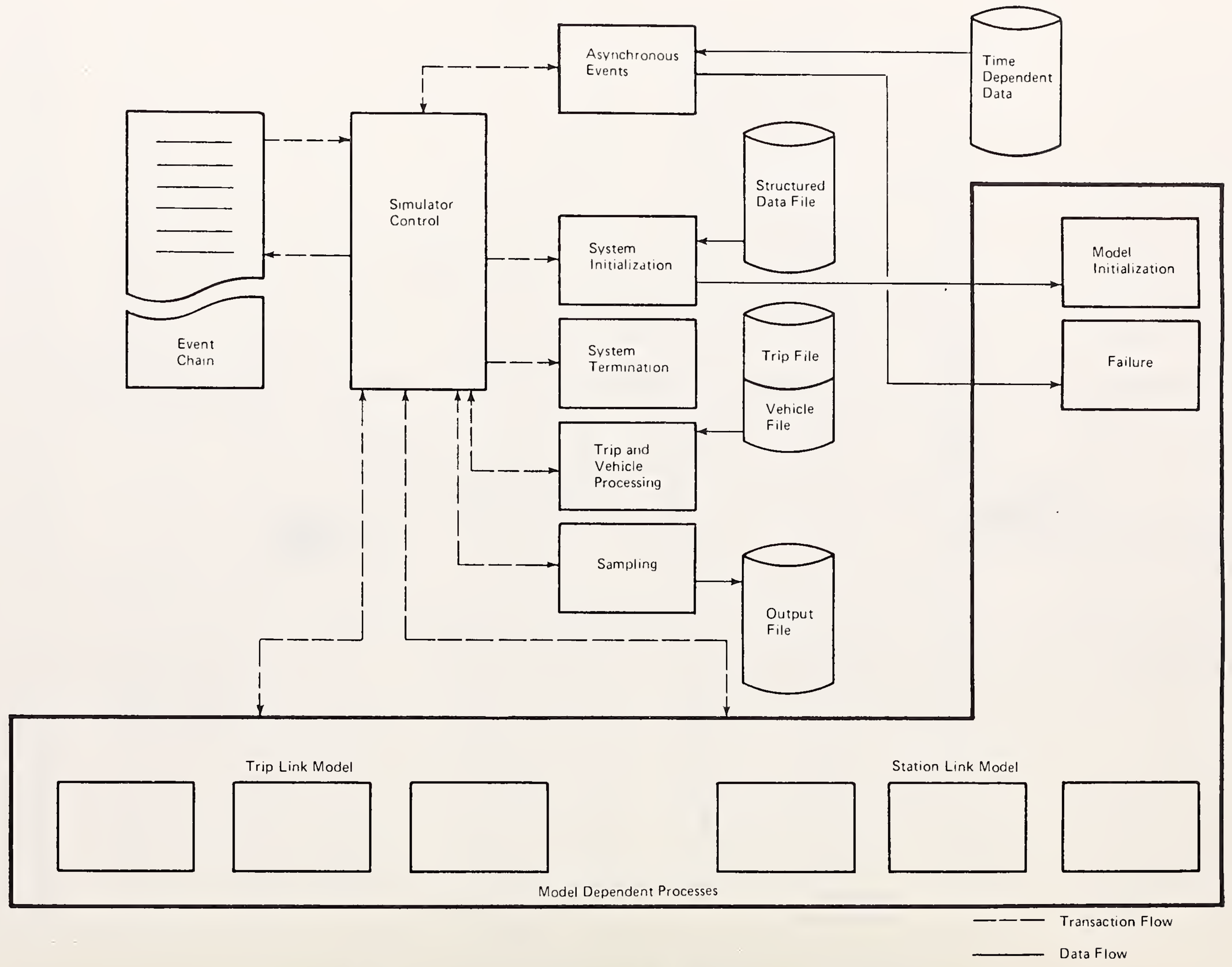


Figure 2-8. Architecture/Modeling Control Relationship

Since vehicle and trip transactions are associated with different station link and trip link entities as a simulation progresses, the system architecture must maintain records to account for all transactions at all times. Accordingly, transactions must always be a member of one of three possible lists:

1. An Available List (AL)
2. The Future Event List (FEL)
3. A Queue List (QL) from which transaction restart is required

At the start of simulation, all of the transactions are allocated to an Available List. As vehicles arrive, vehicle transactions are initialized to be located at the source of the vehicle, they are removed from the Available List, and remain while the vehicle is in the simulated area. As trips arrive trip transactions are removed from the Available List and are initialized to be located at the ticketing trip link. As a trip leaves the system (e.g., reaches its destination) its transaction is returned to this available list for future reuse. Similarly, system service request transactions can be reused during the simulation.

2.2.1.1 Modeling Entity Control—From an architectural or control program view, station links and trip links are considered as entities requiring a basic set of fixed processes. Both require entry and exit testing and a processing component to provide for transaction movement within the entity being modeled. As such, within the MP, the station link and trip link models contain parallel processing components as shown in Figure 2-9. The actual decision logic and event processing within these components differs for station links and trip links.

The relationship between the simulation architecture and the station link and trip link model is shown in the PDL segment hierarchies given in Figures 2-10 and 2-11.

2.2.1.1.1 Station Link Event Process - Stations are configured from canonical station links. A canonical station link (shown in Figure 2-12) contains all seven possible types of events that can happen to a vehicle in a station in a fixed order:

1. Tragel the headway zone
2. Travel the main body of the link
3. Undergo the deboarding of passengers
4. Undergo the boarding of passengers
5. Undergo joint deboarding and boarding of passengers

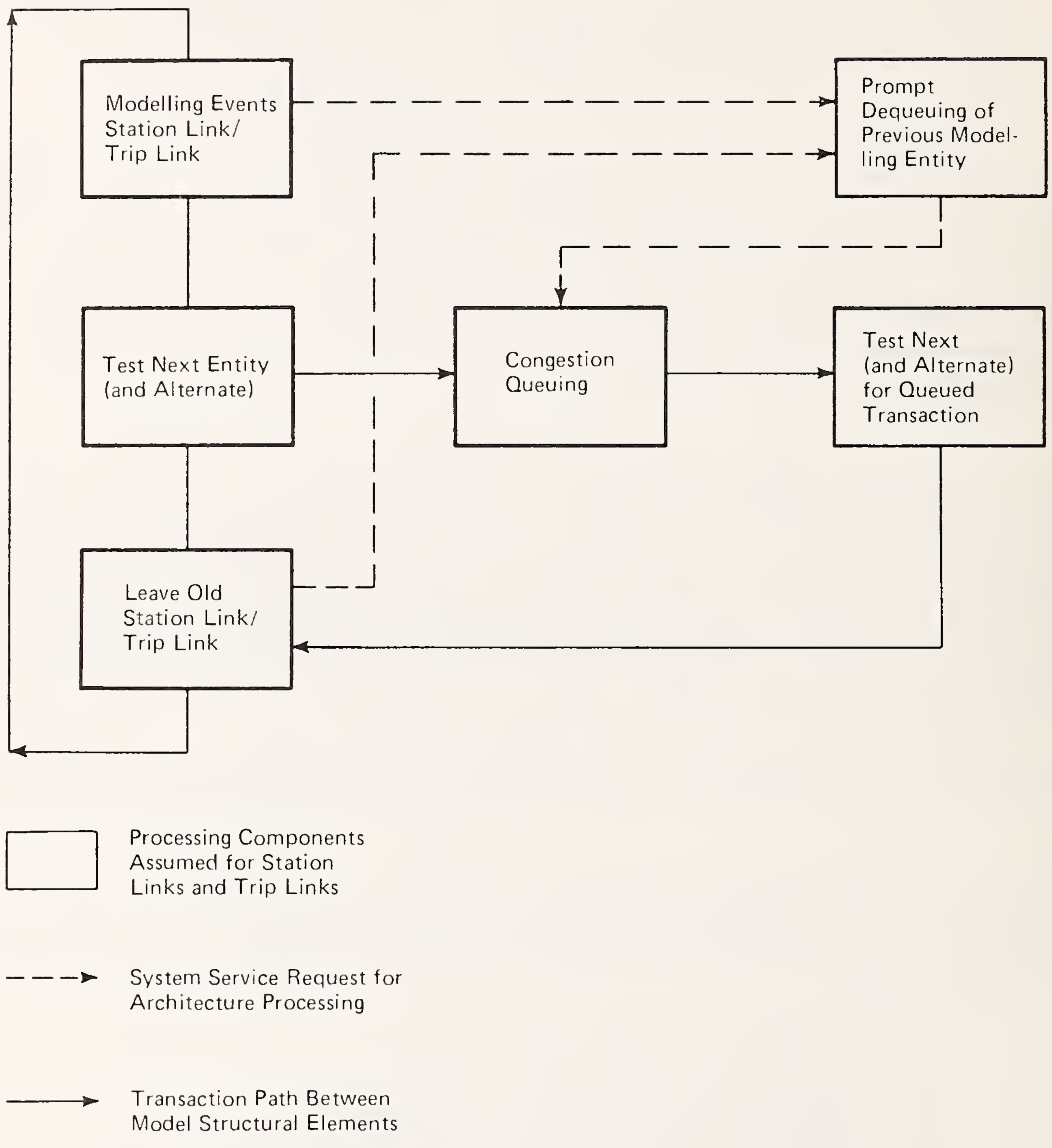

Modelling Events Station Link/ Trip Link
Prompt Dequeuing of Previous Model-ling Entity
Test Next Entity (and Alternate)
Congestion Queuing
Test Next (and Alternate) for Queued Transaction
Leave Old Station Link/ Trip Link
Processing Components Assumed for Station Links and Trip Links
System Service Request for Architecture Processing
Transaction Path Between Model Structural Elements

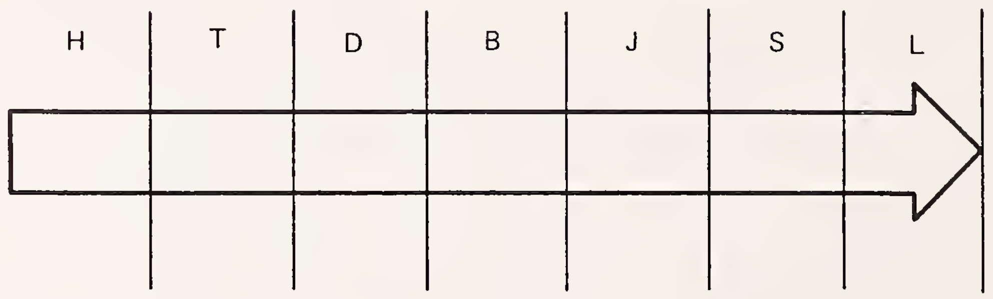
Figure 2-9. DSM Entity Modeling Architecture

```
                    STATION LINK MODEL

SASCTL     <STATION LINK MODEL CONTROL>
   |
   |---SSMOD      <STATION LINK EVENT PROCESSING>
   |     |
   |     |---SSMODB     <BEFORE TIME SEGMENT PROCESSING>
   |     |     |
   |     |     |---SMBRD      <GENERATE BOARD LIST>
   |     |     |     |
   |     |     |     |---SMRSEL     <SELECT RANDOM POINT ON DIST.>
   |     |     |           |
   |     |     |           |---SMRNG     <GENERATE UNIFORM RAND.NO.>
   |     |     |
   |     |     |---SMDBRD     <GENERATE DEBOARD & XFER LISTS>
   |     |     |     |
   |     |     |     |---SMRSEL     <SELECT RANDOM POINT ON DIST.>
   |     |     |           |
   |     |     |           |---SMRNG     <GENERATE UNIFORM RAND.NO.>
   |     |     |
   |     |     |---SMLTIM     <FIND LAUNCH DELAY>
   |     |     |---SMDETR     <DETRAIN VEHICLES>
   |     |
   |     |---SSMODN     <NEXT EVENT SELECTION>
   |     |---SSMODA     <AFTER TIME SEGMENT PROCESSING>
   |           |
   |           |---SMENTR     <ENTRAIN VEHICLE>
   |           |---SSPMAC     <SCHEDULE STATION LINK PROMPT>
   |           |---SUPMAC     <SCHEDULE TRIP LINK PROMPT>
   |           |---SMNXST     <DETERMINE NEXT STOP>
   |           |     |
   |           |     |---SMEVM     <EMPTY VEHICLE MANAGEMENT>
   |           |           |
   |           |           |---SMRSEL     <SELECT RANDOM PT. ON DIST.>
   |           |                 |
   |           |                 |---SMRNG     <GENERATE UNIF.RAND.NO.>
   |           |
   |           |---SMTAEQ     <TRIP ARRIVES AT BOARD QUEUE PROCESSING>
   |                 |
   |                 |---SSPMAC     <SCHEDULE STATION LINK PROMPT>
   |
   |---SSTEST     <FIND NEXT STATION LINK & DO EXIT & ENTRY TESTS>
   |     |
   |     |---SMDIVF    <BUILD LIST OF POSSIBLE NEXT SLS ORDERED BY PREF>
   |           |
   |           |---SMDIVS    <SEARCH FOR LINK OF A GIVEN TYPE>
   |           |---SMDIVU    <ORDER LIST OF LINKS BY OCC/PSEUDO OCC>
   |
   |---SSLEAV     <STATION LINK LEAVE PROCESSING>
         |
         |---SSMOD      <STATION LINK EVENT PROCESSING>
         |---SSPMAC     <SCHEDULE STATION LINK PROMPT>
```

Figure 2-10. Station Link Model Processing Hierarchy

```
                    TRIP LINK MODEL

SAUCTL      <TRIP LINK MODEL CONTROL>
  |
  |---SUMOD      <TRIP LINK EVENT PROCESSING>
  |---SUTEST     <FIND NEXT TRIP LINK & DO EXIT & ENTRY TESTS>
  |---SULEAV     <TRIP LINK LEAVE PROCESSING>
  |      |
  |      |---SUMOD      <TRIP LINK EVENT PROCESSING>
  |      |---SUPMAC     <SCHEDULE TRIP LINK PROMPT>
  |
  |---SMTABQ     <TRIP ARRIVES AT BOARDING QUEUE PROCESSING>
         |
         |---SSPMAC     <SCHEDULE STATION LINK PROMPT>
```

Figure 2-11. Trip Link Model Processing Hierarchy



H — travel the headway zone;
T — travel the main body of the link;
D — undergo the deboarding of passengers;
B — undergo the boarding of passengers;
J — joint deboarding and boarding of passengers
S — store the vehicle on this link;
L — undergo the delay waiting for launch.

Figure 2-12. Station Link Canonical Definition

6. Being stored
7. Undergo the delay waiting for launch

On any one particular link the user will specify only a subset of these events to occur on that link. Some examples are:

- Model the input ramp just by (1) and (2)
- Model a docking lane by (1), (2), (3), and (4)
- Model an output queueing lane by (1), (2), and (7)
- Model a storage lane by (6)

The vehicle is assumed to move from one event to the next without any intermediate queueing with the exception of the launch event for which the vehicle must be at the head of the link. Thus, the ordered sequence of time periods on a link is as follows:

a. Time on FEL for all events except launch event

b. Time queued waiting to get to head of link for launch event (if launch event specified on link)

c. Time on FEL for launch event (if specified on link)

d. Time queued waiting to get off link due to congestion/failure/other vehicle ahead.

Most links in a station would not have the launch event; it usually occurs on the end of an output queue or, if there is none, the end of the docking lane. Thus in the simplest case vehicles just pass from one event to the next, do not queue and move onto the next link. In the next most complex case a vehicle moves from one event to the next, but when it gets through with all of them it discovers that there is another vehicle ahead of it still undergoing events. There the subject vehicle then queues waiting for the other to finish (done with events and not at the head). To add the next increment of complexity, the vehicle when it gets to the end could discover that it cannot leave due to congestion, failure of the exit of the link it is on, or failure of the entrance of the next link it must go on. There the vehicle also queues (done with events on at the head of the link). The next increment of complexity comes when the launch event is at the end of the link. In this case the vehicle may have to queue before beginning this event to wait to get to the end of the link (not done with events and not at the head of the link). After queueing for this reason, it then would go on the FEL for a period of time associated with the launch event and may then queue again due to downstream congestion or failure. It is for these reasons associated with queueing that the selected subset of events from the canonical link must be in the same order as on the canonical link. The store event simply queues the vehicle on the link for yet another (a fourth) queueing reason. The only activity that will take a vehicle out of this state is a request for an entry from local storage. Thus this event must appear only on the storage link as defined by SLSTOR and also must be the last event on the link on which it appears.

In addition to specifying the events that are to occur on each station link, the user specifies the connectivity of station links that form the station to be modeled. This connectivity is defined by giving the following four data items for each station link that is to appear in the modeled station:

1. List of station links downstream of the link being defined

2. Number of the diverge function (case within code segment SMDIVF) to be used to determine which link should be used next if there is a diverge at the end of the link being defined

3. List of the station links upstream of the link being defined

4. Indicator as to whether vehicles are to be dequeued from the upstream links in FIFO order or in a priority order as defined by the order they are given in (3) above.

By specifying these four data items and the list of events that are to occur on each link, the user is given great latitude in the amount of detail that can be put in the station. A diagram of one possible configuration is given in Figure 2-13

It should be noted that the six diverge functions given in the code segment SMDIVF are intended to support a baseline configuration of the form shown in Figure 2-11 and that other configurations may require that additional diverge functions be added to SMDIVF. See User's Manual for a complete description of station link processes.

2.2.1.1.2 Trip Link Event Processes - When a trip enters the station in DSM, it does so by entering a ticketing link. This is the first of three trip links it enters in a fixed order: ticketing link, turnstile link, and boarding link. They are shown in Figure 2-14. Each of the three links contains an event in which the trip spends a period of time on the FEL that corresponds to walking to the process or queue at the end of the link. The walk time on each link is a user input. Each link also contains a queue. For ticketing and turnstile links, a trip remains in the queue until it arrives at the head of the link to begin processing. For the boarding link a trip remains in the queue until it boards a vehicle. The ticketing and turnstile links also have processing.

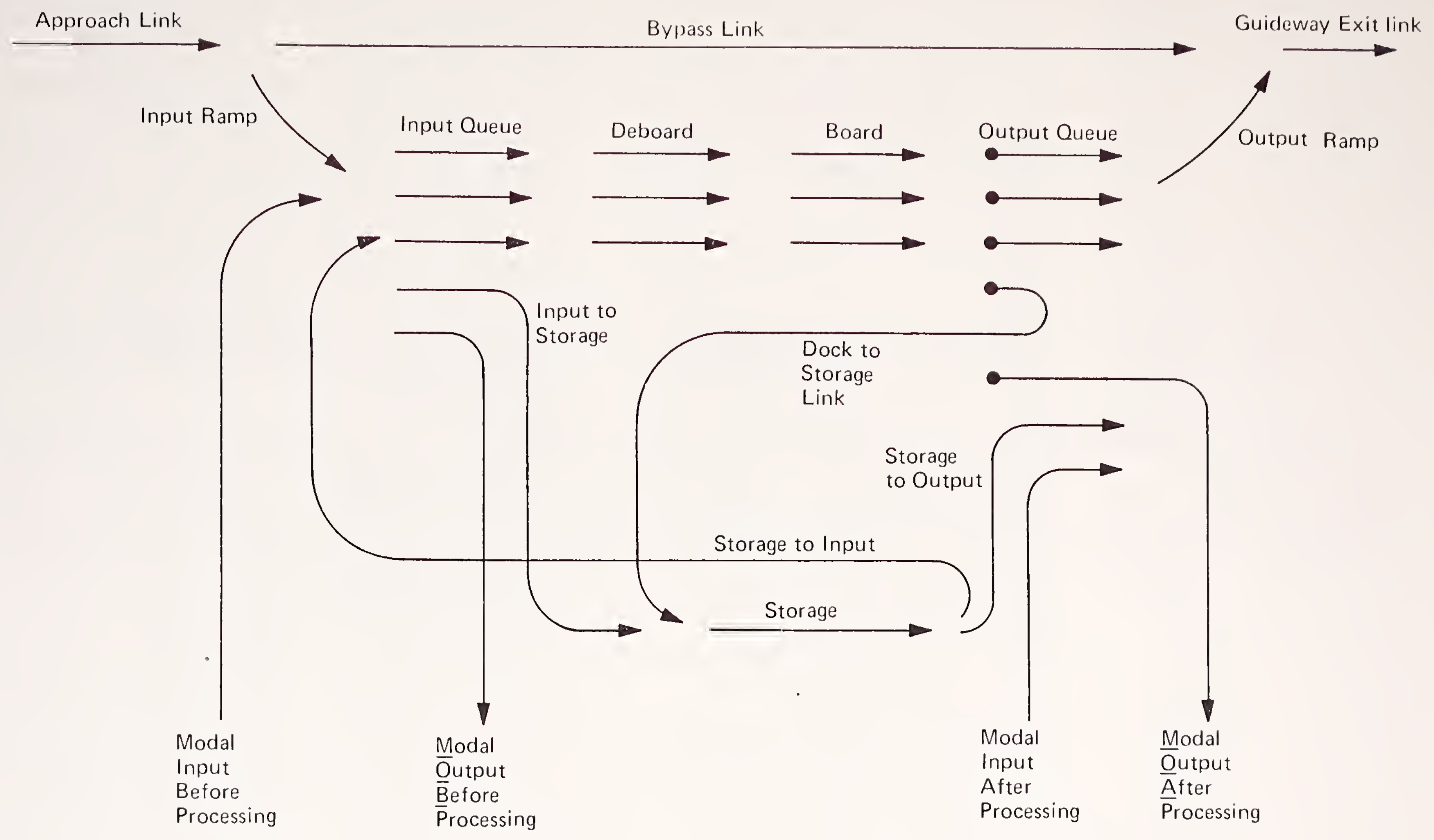


Figure 2-13. Sample Configuration of Station Links

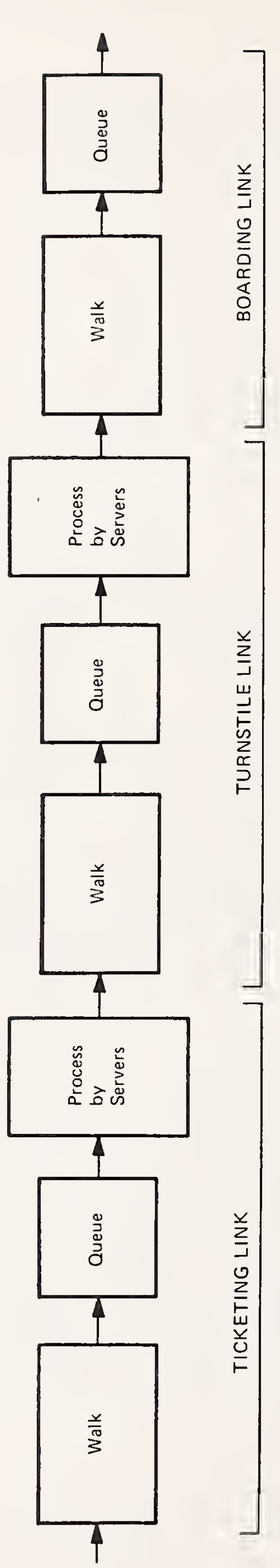


Figure 2-14. Trip Link Sequence

mechanisms, which represent a set of parallel ticketing/turnstile machines, through which the trips must pass. For these two links the lead trip on the link spends an amount of time in the processing mechanism as computed from the form $ax/y + b$, where x is the number of passengers in the trip, y is the number of active servers (ticketing machines/turnstiles) and a and b are user-specified times.

When all servers are busy or failed or the next area is at capacity, the trip waits in the current area (except in case of trips arriving at a capacitated ticketing link, which are rejected). See User's Manual for a complete description of trip link processes.

2.2.1.1.3 Transaction Dequeueing—As previously mentioned, vehicle and trip transactions in the simulation are subject to queueing within the modeling subsystem depending upon whether the next entity or processing event for which they are to be scheduled is available or can be performed. The MP provides the means by which queued transactions can be restarted (scheduled for their next event) when conditions are such that the event upon which they are waiting can now be performed. This is accomplished by the scheduling of a system transaction which causes the prompting (interrogation) of upstream station links or trip links. The scheduling of prompt transactions results from completion of specific event processes within the simulation system. Specifically, the scheduling of a prompt transaction occurs each time an entity exit is processed. Additionally, prompts are scheduled in response to asynchronous events such as failure recoveries.

2.2.1.2 Future Events List—The Future Events List (FEL) is a time ordered list of pointers used to chain transaction ID's for scheduling of events for occurrence in future simulated time. Time is quantized into discrete, finite units called 'clock units,' with each unit representing some period of simulated time, e.g., one millisecond. Each pointer in the clock table begins the list of transaction ID's which require processing during a simulation interval. The point in real time at which the simulator is currently operating is given by clock time which provides the number of clock units which have passed since the start of the simulation experiment.

Every transaction that represents an action to be performed at some future time is placed into the FEL, at the proper time point. To record when events are to occur, each transaction has a time word that defines the time at which it is to be processed. Scheduling of transactions on the FEL is performed by both the architecture control and modeling subsystems. Each transaction has as a part of its definition a chain word which is used for inserting it into the FEL. Transactions are inserted into the FEL by determining the time interval (pointer) within which the event for which is being scheduled is to occur. The transaction is then chained in time order into the list of transactions which are to become active in the specified simulation interval. The organization of the FEL is shown in Figure 2-15.

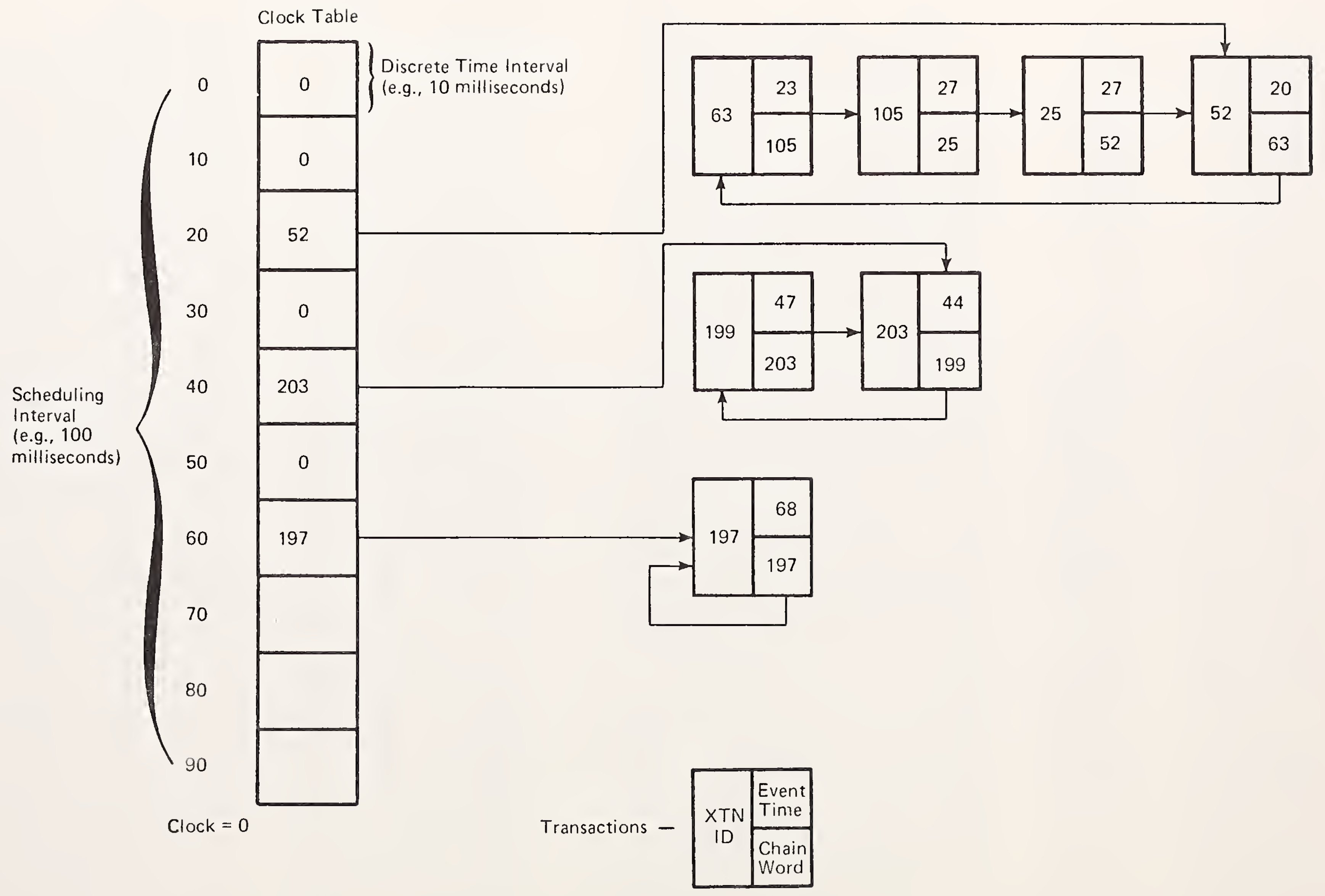


Figure 2-15. Clock Table Organization

Since the clock table portion of the FEL is of finite length, only a finite number of time intervals can be represented. Transactions which must be scheduled for a time interval greater than the time period represented by the clock table are scheduled on the FEL extension or multiple thread future events list. Entries or quantized intervals in the multiple thread list represent an interval of time corresponding to an entire clock table interval. Multiple thread list pointers differ from clock table pointers in that they are created dynamically as required during the simulation experiment by chaining available transactions which serve as the FEL pointer for chaining transactions which are scheduled during that simulation time interval. Transactions placed on the multiple thread list are chained from the multiple thread transaction without regard to discrete simulation intervals as maintained in the clock table. The organization of the multiple thread list is shown in Figure 2-16.

Once the simulation interval encompassed by the clock table has passed, (all transactions processed and clock updated to last transaction time), the clock table is updated from the next available multiple thread list pointer.

2.2.1.3 Event Recognition and Control - The basic control loop in the MP is to determine the next event to be performed, update the simulation clock, and perform the event. Since every event is represented by a transaction, the transaction is the basis for determining the next process to be performed. The control loop in the simulator consists of the following as shown by PDL segment SAMAIN:

1. Obtain the next most imminent transaction. The next event to be performed is indicated within the transaction which is first on the FEL.

2. Remove the transaction from the FEL.

3. Update the simulation clock to the time of the transaction. Whenever the simulation clock is updated, it is updated to the time of the next most imminent event.

4. Perform the indicated event. The type of event to be performed is indicated by another item of information associated with the transaction. This item is used to determine which architectural processing component is required and a control transfer is performed.

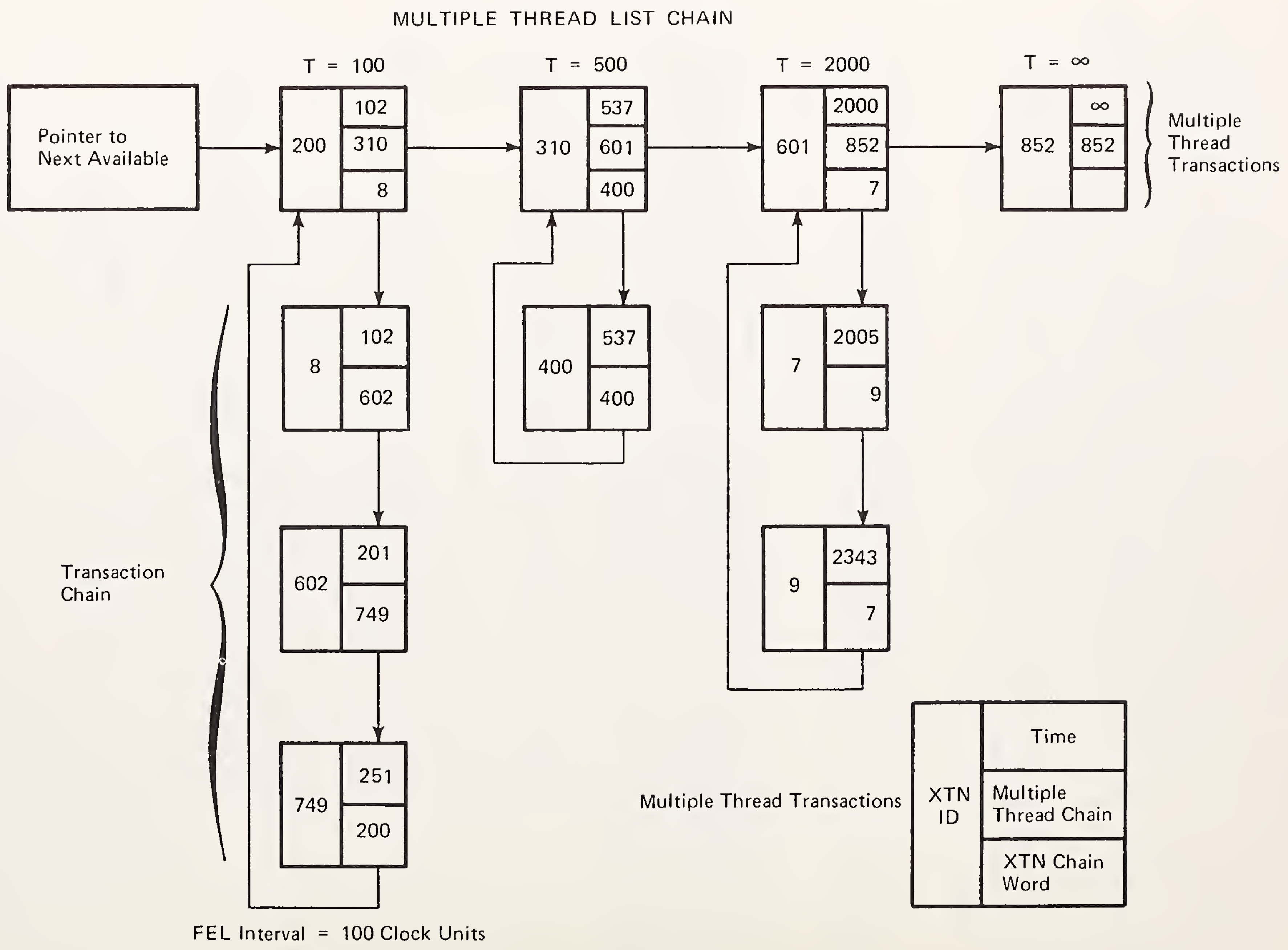


Figure 2-16. Multiple Thread List Organization

2.3 OUTPUT PROCESSOR

The DSM Output Processor provides the means by which sampling data, written to the Raw Statistics File during a simulation experiment, can be retrieved and formatted for station analysis. The Output Processor permits access to and manipulation of the raw statistics in a convenient and unrestrictive manner. This is achieved by providing a user interface which does not require knowledge of how data is formatted, acquired from the input source or arranged internal to the processor itself.

The processing performed by the Output Processor is directed by service request commands input by the user. These commands invoke the four basic processes provided by the OP as follows:

1. Data storage
2. Data acquisition
3. Data manipulation
4. Data display

Data request commands provide the means by which desired statistics are specified for retrieval and the presentation format is chosen. These requests are accumulated until a read command,which causes actual accumulation and formatting of data,is encountered.,

2.3.1 Architecture

Execution of the OP is initiated by invoking a cataloged job control procedure contained in the procedure library. Upon entry, the OP saves parm field information required for index file updating and control is passed to the main OP control routine. The OP then performs initialization processing. This involves initial reading of the Raw Statistics File to retrieve required control information and the allocation of internal storage areas used for data accumulation.

Once initialization is complete, the basic control loop of the Output Processor is started by reading the first data request command and creating the first entry in the data request table. If output is to be generated for the Performance and Summary File, the Index File is updated as required. Consecutive reading and storing of data requests is performed until a read command is encountered. This causes the data acquisition and display process to begin. This involves the following procedures:

1. Positioning of the Raw Statistics File to the first sampling data records contained within the time interval specified in the read command.
2. Determining the type of data records required to satisfy stored requests.

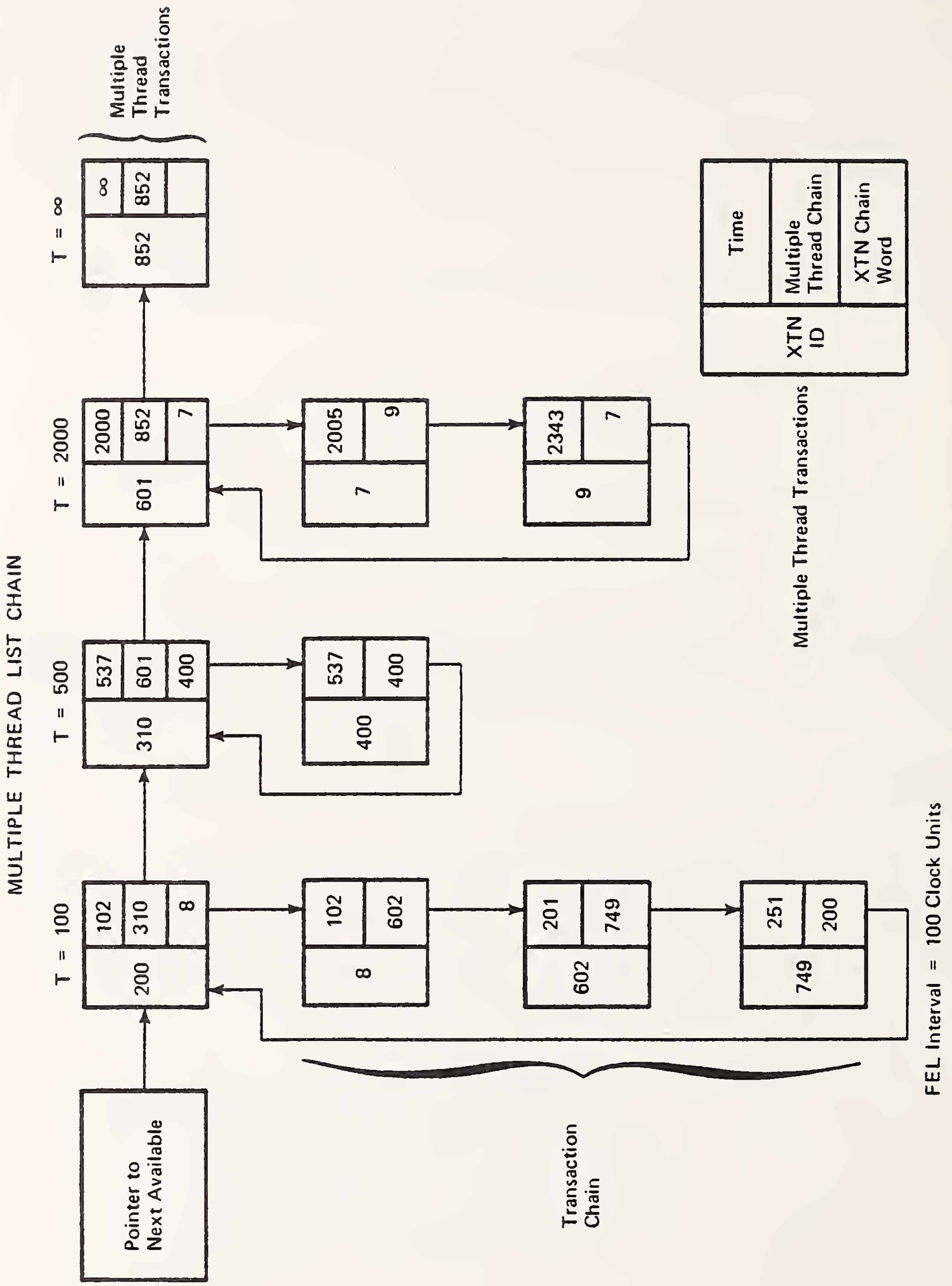


Figure 2-16. Multiple Thread List Organization

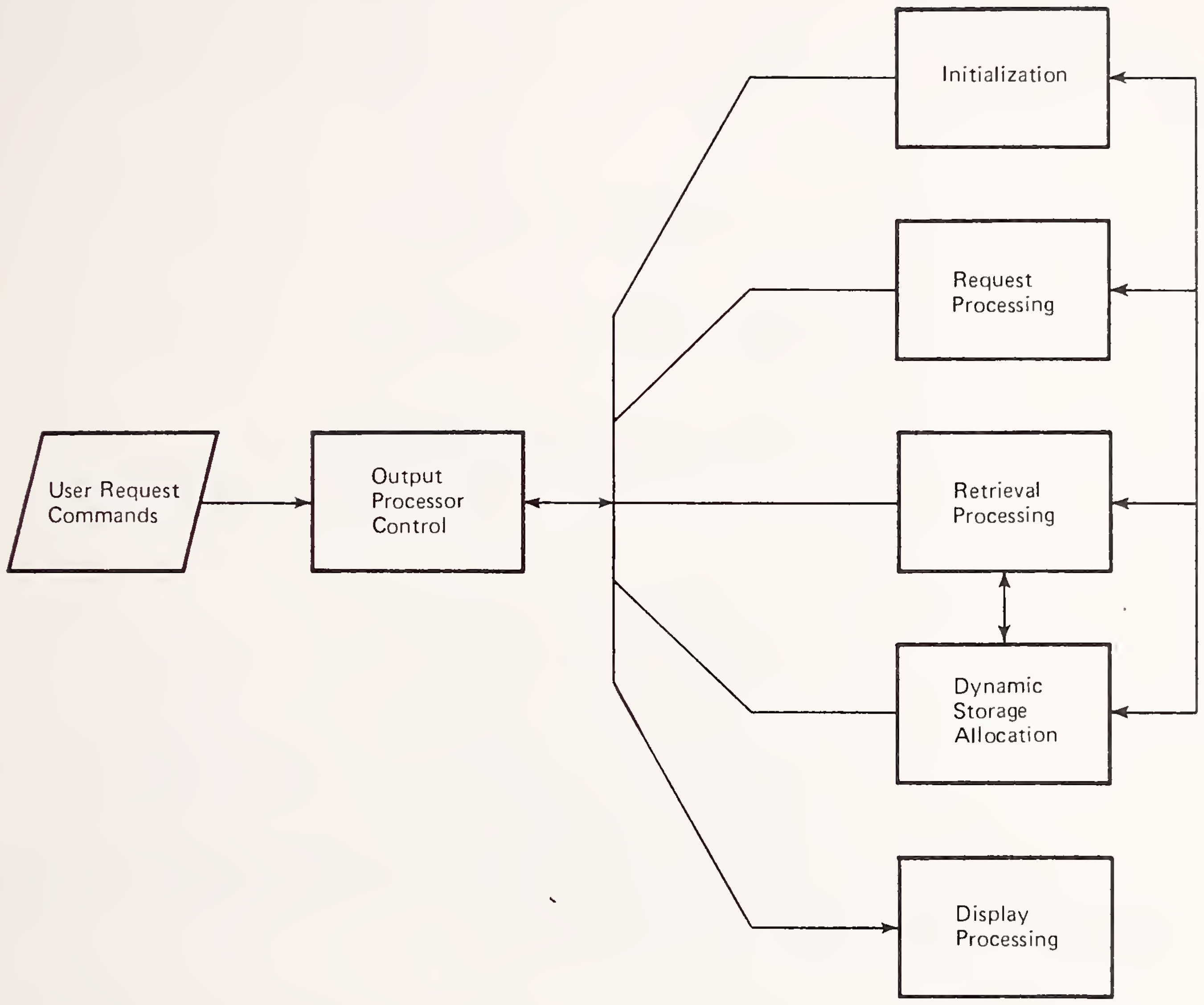


Figure 2-17. Output Processor Architecture

Figure 2-18 illustrates the manner in which bins are referenced within the OP. The bin number, i, is used to index the bin location pointer. The number in the location pointer (i) provides the position in the bin storage area at which the bin is located. By convention, j=location pointer (i) always indicates the third word that has been allocated to the bin is that data retrieval by bin can be accomplished. Each line is intially allocated as five words consisting of four header words plus one unused data word. The initial number of bins allocated is given by the number of entities (station links and trip links) used in the simulation experiment as reflected in the Raw Statistics File.

In general, a bin consists of several distinct areas as shown in Figure 2-18:

1. The system header--entries j-2 and j-1.

   This area is used exclusively by the bin storage allocation and maintenance services. It specifies:

   a. The total number of words in this bin area, including those in the system header

   b. The identity number of this bin itself.

2. The data header--entries j-j+2 used during the data retrieval process:

   a. The starting index of the data in the bin

   b. The ending index of the data in the bin

   c. A data identity area used to identify the data in the bin according to the mnemonic used in requesting the specific data item.

The unused portion of bin allocation area is set up as a pseudo-bin, with bin number set to zero to indicate its being unused.

2.3.1.2 Command Request Storage - Each data request entered by the user causes the contents of the request to be filed in a request table used in data retrieval processing. As part of this filing process, a bin assignment and reallocation is made for internal storage of the sampling data to be retrieved during request servicing. The amount of space reallocated to a particular bin depends upon the display mode specified in the request. The amount of space allocated at this point serves only as initial estimate of storage required. If further space is required during the data retrieval process, it is obtained dynamically by repositioning bin assignments within the bin storage area.

In addition to the above, request filing results in the category definition for the data item selected. This definition is stored with the request for identifying the required sampling records which must be processed to service the data request. In the

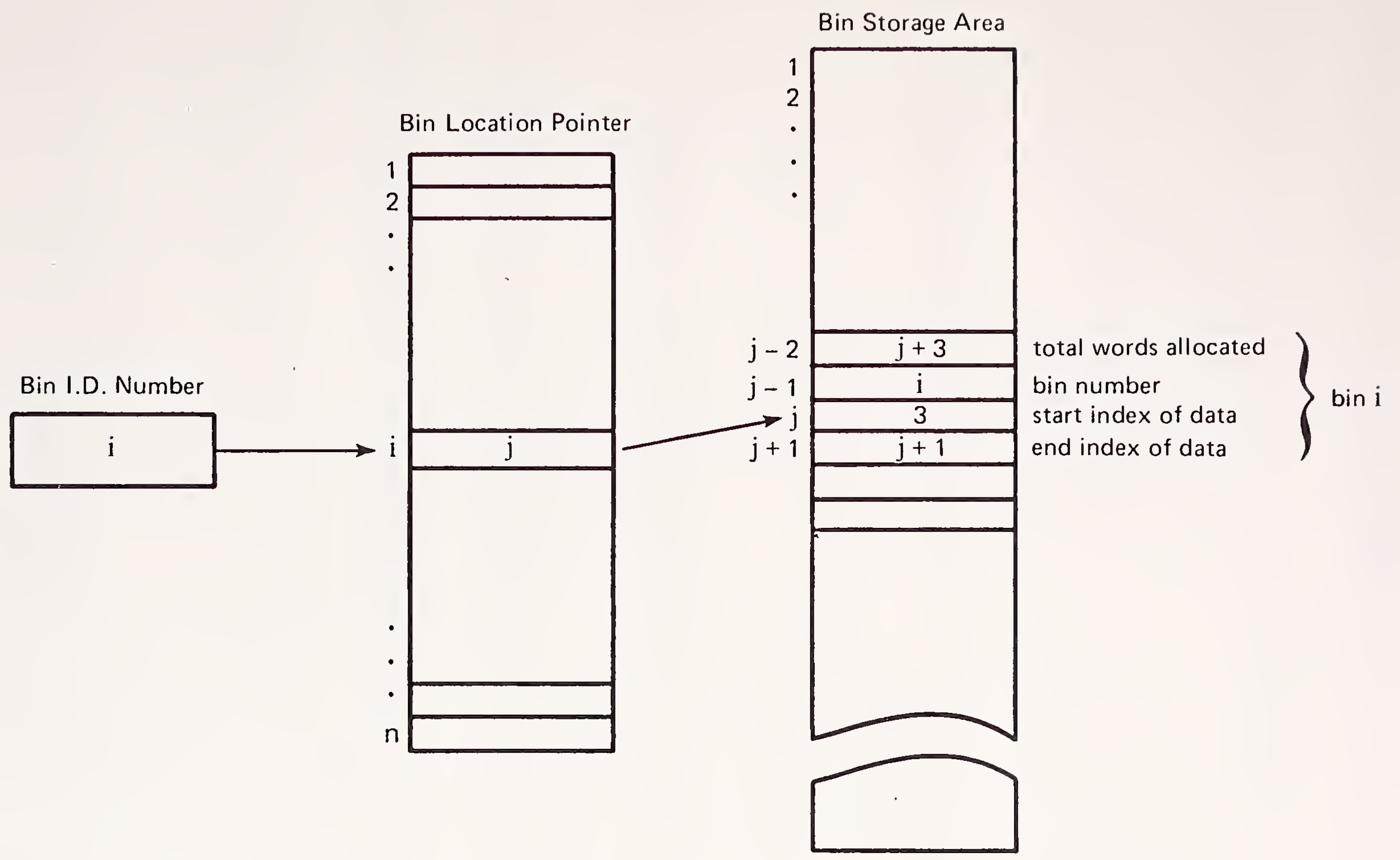


Figure 2-18. Output Processor Bin Referencing

Raw Statistics File, data is stored on a time sample basis according to a category hierarchy. Data is classified as to which portion of the model it pertains (major category): system, station link, or trip link.

System-level data requires no subscript or index--each data element is a single number. Station link or trip link data elements require an entity index since each element consists of multiple values--one per station link, or trip link.

Further, data are classified as to whether they are status data or historical data (subcategories). Status data reflect the data of a modeled area at the instant at which sampling took place (e.g., the number of vehicles on link five). On the other hand, historical data reflect what events transpired over the interval preceding the sampling event (beginning after the previous sample and ending at the time of the current sample). Examples of historical data are the number of vehicles leaving Link 5 exit queue, and the average number of vehicles on Link 5.

The organization of data in the Raw Statistics File consists of groups of unformatted logical records. The first record of a group is a header record. The header contains the following information:

1. A code number that indicates the type of group each one is (major category).

2. A count of the number of logical records (sampling data followers) in the group, excluding the header. If the group consists of the header only, this count is zero.

3. The value of the simulation clock at the time the record group was written. (This value is non-decreasing along the file.)

The remaining logical records of the group, if any, have a format unique to that type of group, which is indicated by the header. This header-follower organization has several distinct advantages:

1. Widely varying kinds of data may be interspersed on the tape.

2. Within a given simulator clock value, the order of information is unimportant.

3. Groups of records may be placed upon the tape or omitted from it, at the sole discretion of the program which is writing the tape (i.e., the Model Processor).

4. Information not needed on a given pass of the tape can be quickly skipped, simply by skipping the number of followers specified by the header.

The relationship between the request table and data storage bins resulting from the request filing process is shown in Figure 2-19.

2.3.1.3 Establishing Request/Record Correlation - Once the data acquisition process is started in response to a read command, a matching process which identifies which data records can be used to satisfy filed requests is performed. The matching process is started during data acquisition for the first records groups contained within the desired accumulation interval. Subsequent occurrences of record groups need not be matched once the matching process has been performed.

Prior to reading the header for the first record group in the requested interval, a match table is built which provides for each major category (record group), M, the following indicator:

> 0--Record groups of type M are to be ignored (if found on the tape, they will be skipped).
>
> 1--Groups of type M contain data necessary for proper operation of the OP and are always to be read (examples are conversion tables, system dimensions, etc.).
>
> -1--Groups of type M might be required, depending upon their existence on the tape and upon a request for data contained within them.

As reading begins, if the major category indicator for the record group is -1, the match process is performed (as shown in the example given in Figures 2-20 and 2-21).

As shown in Figure 2-20, record type M has the main category ' α ' and a list of subcategories given by column j of the subcategory mnemonic table. This list includes subcategories ' β ' and ' γ '. In looking for matches, the request table entries containing the major and subcategory mnemonic designators are IMAIN and ISUB compared with ' α ' and the subcategory list for all i. Lines s and t of the request table matched both ' β ' and either ' γ ', or ' ϕ '. Consequently:

1. The subcategory indicator of the request table was set to indicate the index of the matching subcategory. That is, since line s matched the 1st subcategory (' γ '), the indicator was set to 1. Likewise, the indicator for request (t) was set to k.

2. A list of requests that can be satisfied by record type M (namely, requests s and t) was created by forming a chain:

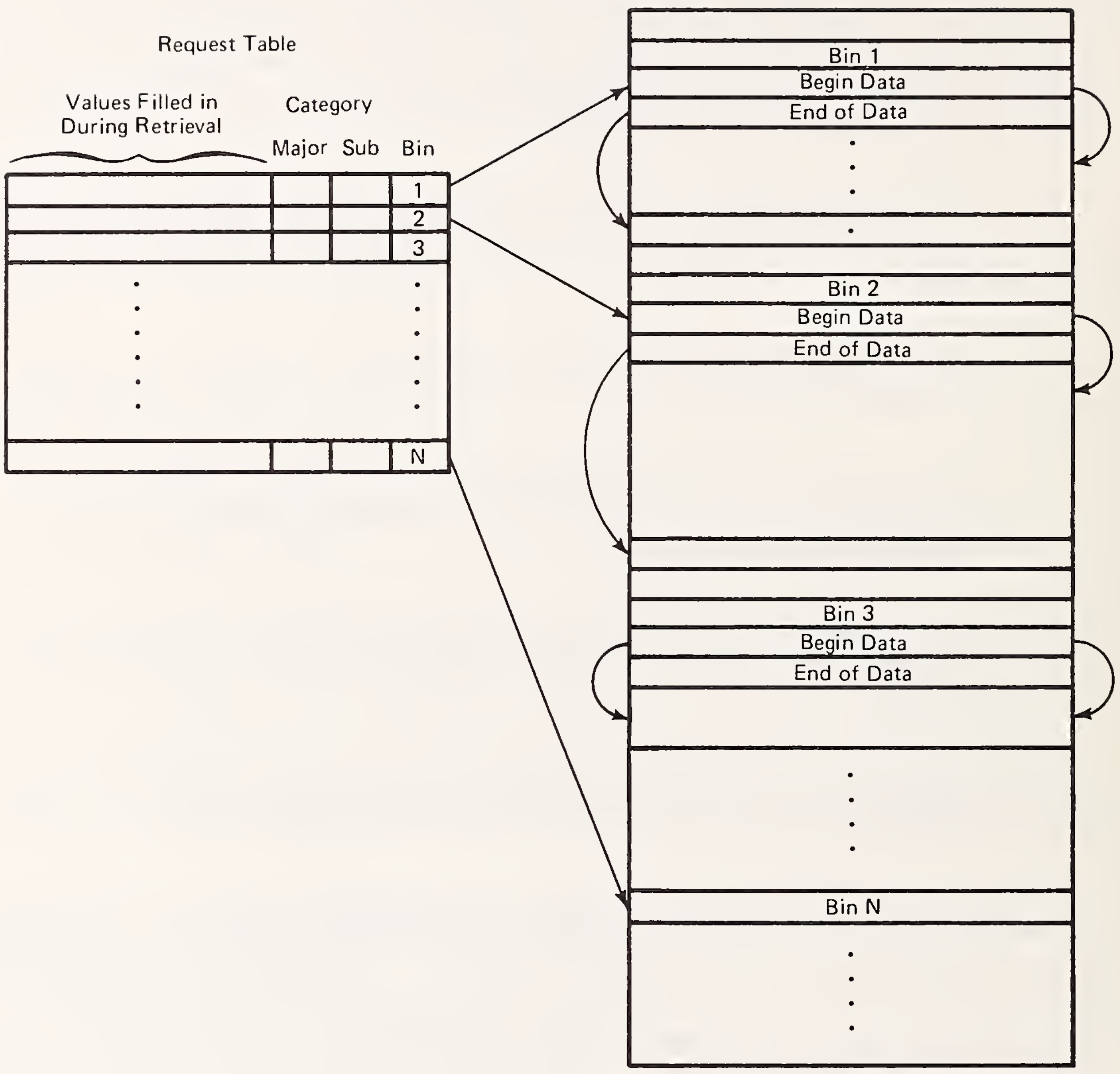


Figure 2-19. Request Filing/Bin Storage Relationship

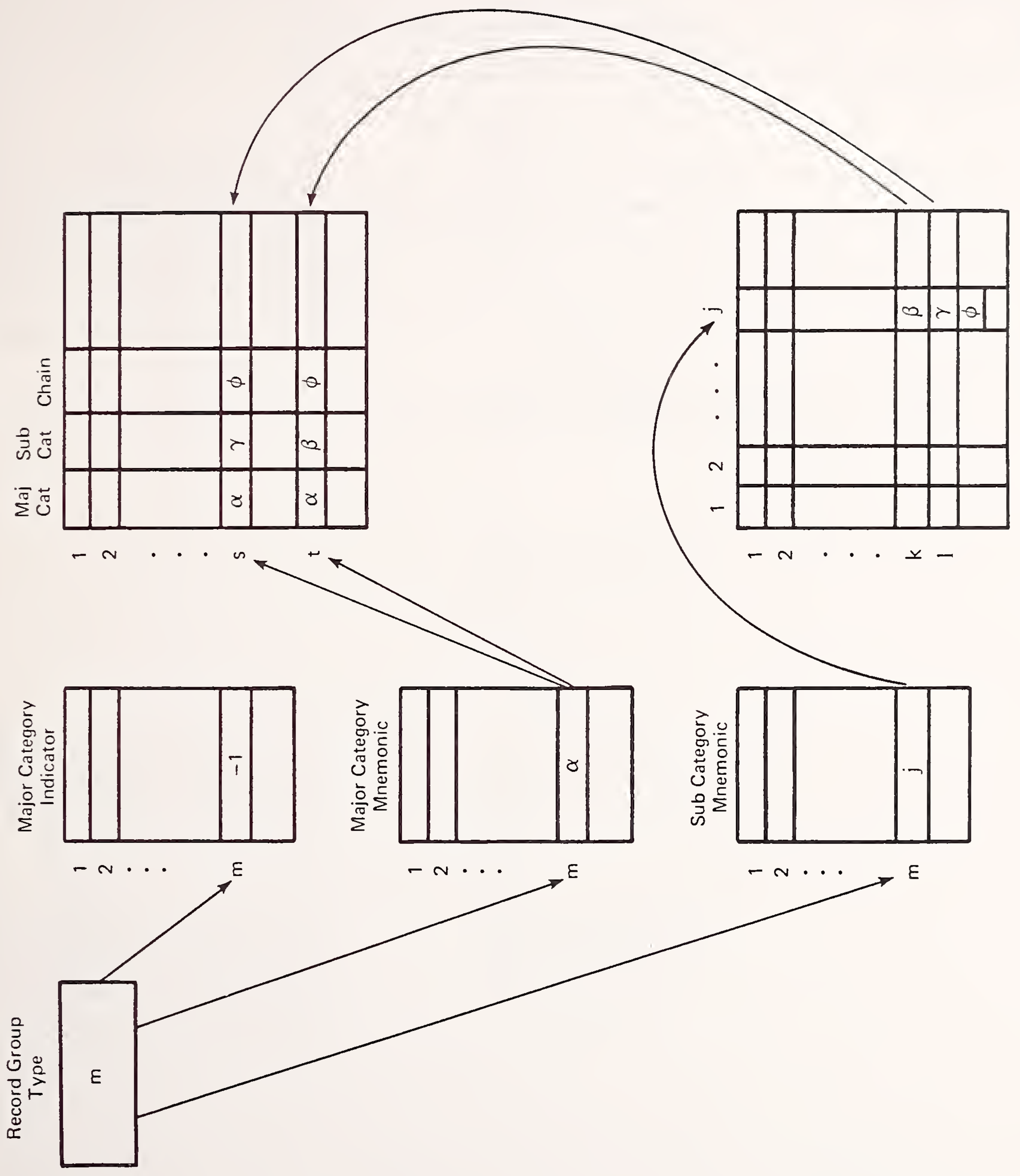


Figure 2-20. Data Matching Process

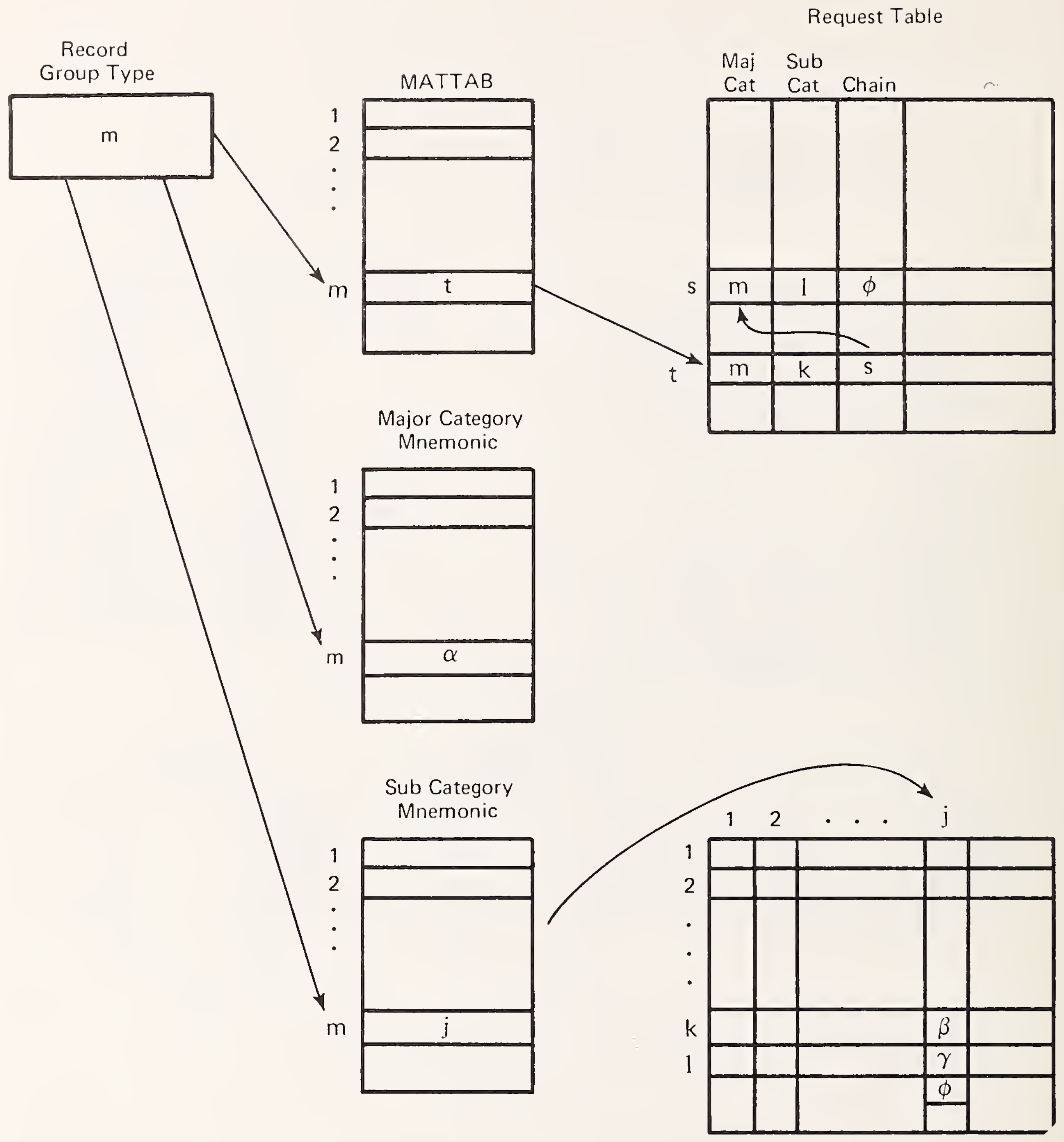


Figure 2-21. Data Matching Results

a. The entry in the major category indicator table which was -1 prior to the matching process, was set to "t", one of the request table lines satisfied by record type M.

b. Item next request for entry (t) in the request table gives a second line (s) satisfied by record type M.

c. Item next request for entry (s)=0 indicates that no other lines in the table are satisfied by record type M.

As a result of these actions, note the major category indicator that indicates those lines of the request table that are satisfied by records of type M, via the chain. Also, the subcategory entry of the request table indicates the exact subcategory of data that was requested, by number. Consequently, the main and subcategory mnemonic tables need not be referenced hereafter for records of this type.

The subcategory indicator now contained in the request table entries serves as the position designator of the required item within the sampled data record.

SECTION 3. GLOBAL VARIABLE DICTIONARY

The following table (Table 3-1) defines variables that are global within the processors. Those that begin with SCN are internal to the input processor. Those that begin with SCI are input by the user through the input processor to the model processor. Those that begin with SCM are internal to the model processor. The remainder are used in the output processor.

- SCAMSG -- MESSAGE COMMONS
- SCICFG -- STATION CONFIGURATION INPUT
- SCIFEL -- FUTURE EVENT TIMING INPUT DATA
- SCIMAX -- RUN-TIME DATA
- SCISL -- STATION LINK INPUT DATA
- SCISYS -- SYSTEM INPUT DATA
- SCITL -- TRIP LINK INPUT DATA
- SCMFEL -- FUTURE EVENT TIMING MAINTAINED BY MODEL PROCESSOR
- SCMFS -- FEL STATISTICS
- SCMSL -- STATION LINK DATA MAINTAINED BY MODEL PROC.
- SCMSYS -- SYSTEM DATA MAINTAINED BY MODEL PROCESSOR
- SCMT -- TRIP DATA MAINTAINED BY MODEL PROCESSOR
- SCMTL -- TRIP LINK DATA MAINTAINED BY MODEL PROC.
- SCMV -- VEHICLE DATA MAINTAINED BY MODEL PROCESSOR
- SCMXTN -- TRANSACTION HEADER DATA MAINTAINED BY MODEL PROCESSOR
- SCNMAX -- IP RUN-TIME MAXIMA
- SCNSYS -- SIMULATION SYSTEM DATA

- SCNTDM -- TRIP DEMAND GENERATION DATA
- SCNVDM -- VEHICLE DEMAND GENERATION DATA
- SCZ -- MODEL STATISTICS
- SMAXSIZE -- COMPILE TIME MAXIMA
- SODCLS -- COMMON AREAS UNIQUE TO SOP
- SODEFS -- COMMON AREAS IN SODCLS AND ZODCLS WITH FULL DIMENSIONS
- ZCAMSG -- OP ERROR MESSAGE COMMON
- ZODCLS -- COMMON AREAS COMMON TO ALL OP
- ZSYSMAX -- OP COMPILE TIME MAXIMA

The format of the definitions is as follows:

Var Name	Dim	Description
A	B/C	D (E, F, G) (H)

where

A is the official name under which the data is used

B is the dimension of the variable; dash (-) implies it is a scalar

C is the type of variable:

L1 -- Logical, 1 byte

I2 -- Integer, 2 bytes

I4 -- Integer, 4 bytes

R4 -- Real, 4 bytes

D is the definition of the variable and the values it can assume

E is the value it is initialized to by IP

F is the lowest legal value (checked by IP)

G is the highest legal value (checked by IP)

H is other checks, initializations, time conversions, and miscellaneous notes.

Table 3-1. Global Variables -- SCAMSG (Page 1 of 58)

SCAMSG: ERROR MESSAGE DATA

VAR NAME	DIM	DESCRIPTION
KNMSG	3/I2	NUMBER OF MESSAGES ISSUED DURING A RUN, BY CLASS: 1 = INFORMATION 2 = WARNING 3 = SEVERE
NMSGS	-/I2	TOTAL NUMBER OF MESSAGES OF ANY CLASS ISSUED DURING A RUN
MSGC	<MMSGS /I2	MESSAGE NUMBERS ISSUED DURING RUN
MSGCN	<MMSGS /I2	NUMBER OF REMAINING MESSAGES OF THIS TYPE ALLOWED PRIOR TO TERMINATION
TERM	-/L1	INDICATOR TO SIGNAL TERMINATION DUE TO EXCESSIVE MESSAGES
MFLAG	-/L1	ERROR PROCESSING IN PROGRESS INDICATOR TO HALT RECURSIVE ERROR PROCESSING
MSGID	-/L1	ID OF PROCESSOR BEING EXECUTED (1 = INPUT PROCESSOR 2 = MODEL PROCESSOR)

Table 3-1. Global Variables -- SCICFG (Page 2 of 58)

```
-----------------------------------------------------------------------------
NAME: SCICFG                   CATEGORY: INPUT PROCESSOR STATION CONFIGURA-
                                         TION DATA
-----------------------------------------------------------------------------

VARIABLE    DIM     TYPE     DESCRIPTION

----------  ------  -------  --------------------------------------------------
 ***NOTE:   STATION LINKS ARE IMPLICITLY NUMBERED BY THE ORDER IN WHICH
            THEY ARE DESCRIBED IN INPUT. EXAMPLE: IF THE FIRST LINK
            DESCRIBED IN SLCFIG IS THE UPSTREAM LINK, THEN THE
            UPSTREAM LINK IS ALSO KNOWN AS STATION LINK 1.

 SLCFIG     13,      I*2     DESCRIPTORS FOR EACH LINK IN STATION
             KMSL
                          ***NOTE:  FOR EACH LINK IN THE STATION, THE TABLE
                             SLCFIG CONTAINS 13 DESCRIPTORS FOR ENTERING
                             THE STATION LINK'S ATTRIBUTES.  COLUMUN 1
                             IS SLCFIG(1) AND COLUMN 13 IS SLCFIG(13).
                             DEFINITIONS FOR EACH COLUMN ARE PROVIDED:

                             COL 1 = STATION LINK TYPE USED TO GROUP
                               LINKS FOR REPORTING PURPOSES, FOR
                               DIVERGE FUNCTIONS, AND TO IMPLICITLY
                               CONFIGURE THE STATION (DETERMINE THE
                               UPSTREAM AND DOWNSTREAM LINKS FOR EACH
                               LINK):

                                1 = IR  = INPUT RAMP
                                2 = IQ  = INPUT QUEUE
                                3 = D   = DOCK(DEBOARD AND/OR BOARD)
                                          (DEBOARD, BOARD, AND JOINT
                                          EVENTS--3,4,5--CAN APPEAR ONLY
                                          ON THIS LINK TYPE)
                                4 = OQ  = OUTPUT QUEUE
                                5 = OR  = OUTPUT RAMP
                                6 = S   = STORAGE
                                7 = IS  = INPUT-TO-STORAGE
                                8 = SI  = STORAGE-TO-INPUT
                                9 = DS  = DOCK-TO-STORAGE
                               10 = SO  = STORAGE-TO-OUTPUT
                               11 = UL  = UPSTREAM-LINK (APPROACH LINK)
                               12 = BL  = BYPASS LINK
                               13 = DL  = DOWNSTREAM LINK
                               14 = MIB= MODAL INPUT BEFORE PROCESSING
                               15 = MIA= MODAL INPUT AFTER PROCESSING
                               16 = MOB= MODAL OUTPUT BEFORE PROCESSING
                               17 = MOA= MODAL OUTPUT AFTER PROCESSING
                               (1,1,17)
```

Table 3-1. Global Variables -- SCICFG (Page 3 of 58)

```
        COL 2 = TOTAL NON-DEGRADED TRAVEL TIME ON
          THE STATION LINK = HEADWAY EVENT TRAVEL
          TIME + TRAVEL EVENT TRAVEL TIME.  INPUT
          IN SECONDS. (OPTIONAL; REQUIRED IF
          COL 3 AND SLVEL ARE NOT USED)

        COL 3 = STATION LINK LENGTH IN FEET (OPTIONAL
           REQUIRED WITH SLVEL IF COL 2 IS
           NOT USED)

        COL 4 = STATION LINK CAPACITY (NUMBER OF
           VEHICLES) MUST BE GREATER THAN OR EQUAL TO
           THE MAXIMUM TRAIN LENGTH. (REQUIRED)
           (0,PMXTRL,1)

***NOTE COL 5-9.  THE EVENTS ON A LINK MUST
BE IN THE ORDER HEADWAY/TRAVEL/DEBOARD/BOARD/
JOINT/STORE/LAUNCH. THE STORE AND LAUNCH EVENTS
MUST BE THE LAST EVENTS ON THE LINK WHEN USED.
THE JOINT EVENT CANNOT BE USED WITH THE DEBOARD
OR BOARD EVENTS IN THE SAME STATION.  THE STORE
EVENT MUST BE PRECEEDED BY THE HEADWAY OR TRAVEL
EVENTS ON THE LINK ON WHICH IT APPEARS. THE
FOLLOWING NUMBERS SIGNIFY EVENTS:
          1 = HEADWAY
          2 = TRAVEL
          3 = DEBOARD
          4 = BOARD
          5 = JOINT (DEBOARD AND BOARD)
          6 = STORE
          7 = LAUNCH

        COL 5 = 1ST EVENT ON LINK

        COL 6 = 2ND EVENT ON LINK (OPTIONAL; DEFAULT
          ZERO)

        COL 7 = 3RD EVENT ON LINK (OPTIONAL; DEFAULT
          ZERO)

        COL 8 = 4TH EVENT ON LINK (OPTIONAL; DEFAULT
          ZERO)

        COL 9 = 5TH EVENT ON LINK (OPTIONAL; DEFAULT
          ZERO)

        COL 10 = DIVERGE FUNCTION NUMBER ASSIGNED
          TO THE LINK WHEN IT HAS TWO OR MORE
          DOWNSTREAM LINKS MUST RANGE FROM 1 THROUGH
          SIX WHERE USE OF THESE FUNCTIONS IS
```

```
                                      TYPIFIED AS FOLLOWS:
                                        1 = END OF UL
                                        2 = END OF IR, MIB, SI
                                        3 = END OF D
                                        4 = END OF S
                                        5 = ORDER BY OCCUPANCY
                                        6 = ORDER BY PSEUDO-OCCUPANCY
                                       DEFAULT IS ZERO WHEN THERE IS ONLY ONE
                                       DOWNSTREAM LINK.
                                       (0,1,6)

                                  WHEN SLPF = 1:
                                    COL 11 = UPSTREAM LINK ORDERING OPTION
                                     USED WHEN UPSTREAM VEHICLES ARE DEQUEUED IN
                                     PRIORITY ORDER.
                                                ORDER OF UPSTREAM LINKS
                                           -------------------------------
                                     OPT    FIRST          SECOND        THIRD
                                     ---   --------       --------      --------
                                      1 = GUIDEWAY       STORAGE       MODAL
                                      2 = STORAGE        GUIDEWAY      MODAL
                                      3 = MODAL          GUIDEWAY      STORAGE
                                      4 = GUIDEWAY       MODAL         STORAGE
                                      5 = STORAGE        MODAL         GUIDEWAY
                                      6 = MODAL          STORAGE       GUIDEWAY
                                      (OPTIONAL; DEFAULT 1)
                                      (1,1,6)

                                    COL 12 = HEADWAY TIME PER TRAIN IN SECONDS
                                      USED TO  COMPUTE TIME TO TRAVEL THE HEADWAY
                                      ZONE. TOTAL HEADWAY ZONE TRAVEL TIME =
                                      (COL 12) * (TRAIN LENGTH) + (COL 13).
                                      (0,0,)

                                    COL 13 = HEADWAY TIME PER VEHICLE IN SECONDS
                                      USED TO COMPUTE TIME TO TRAVEL THE HEADWAY
                                      ZONE. TOTAL HEADWAY ZONE TRAVEL TIME =
                                      (COL 12) * (TRAIN LENGTH) + (COL 13).
                                      (0,0,)

SLVEL          -          I*2       AVERAGE LINK VELOCITY (FT/SEC)
                                      (OPTIONAL; HOWEVER REQUIRED
                                      WITH COL 2 IF COL 3 IS NOT USED)
----------------------------------------------------------------------------------
```

Table 3-1. Global Variables -- SCIFEL (Page 5 of 58)

CIFEL: FUTURE EVENT LIST DATA

AR NAME	DIM	DESCRIPTION
SIZE	-/I4	NUMBER OF CLOCK UNITS PER MINUTE (60,,)
LOOP	-/I4	MAXIMUM NUMBER OF ENTRIES ALLOWED IN ONE CLOCK TABLE ENTRY (1000,,)
LSMAL	-/I4	SPACING BETWEEN CLOCK TABLE ENTRIES EXPRESSED IN CLOCK UNITS * 10 UNITS (100,,)

Table 3-1. Global Variables -- SCIMAX (Page 6 of 58)

RUN-TIME MAXIMA:
THE FOLLOWING VARIABLE NAMES DEFINE THE ACTUAL NUMBER OF ENTITIES USED IN A GIVEN RUN. THESE ARE READ IN AT RUN-TIME AND MUST BE LESS THAN OR EQUAL TO THEIR COMPILE-TIME MAXIMA COUNTERPARTS.

VAR NAME	DIM/TYPE	DESCRIPTION
KNSL	-/I2	ACTUAL NUMBER OF STATION LINKS (1,1,KMSL)
KNV	-/I2	RUN TIME LIMIT ON THE ACTUAL NUMBER OF SIMULTANEOUS VEHICLES IN THE SIMULATION (KMV,1,KMV)
KNT	-/I2	RUN TIME LIMIT ON THE ACTUAL NUMBER OF SIMULTANEOUS TRIPS IN THE SIMULATION (KMT,1,KMT)
KNR	-/I2	ACTUAL NUMBER OF ROUTES (1,1,KMR)
KNRT	-/I2	ACTUAL NUMBER OF ENTRIES IN SCHEDULED ROUTE LIST (PVRLST) (1,1,KMRT)
KNEVP	-/I2	ACTUAL NUMBER OF ENTRIES IN USER'S PRIORITY ORDERED LIST OF WHERE TO PUT EMPTY VEHICLES (PVEPR) (1,1,KMEVP)
KNSVP	-/I2	ACTUAL NUMBER OF ENTRIES IN USER'S ORDERED LIST OF WHERE TO SEARCH FOR EMPTIES (PVSPR) (3,1,KMSVP)
KNNMD	-/I2	ACTUAL NUMBER OF ENTRIES IN NETWORK MERGE DELAY DISTRIBUTION (PNMDDT) (1,1,KMNMD)
KNEVD	-/I2	ACTUAL NUMBER OF ENTRIES IN EMPTY VEHICLE DELAY DISTRIBUTION (PEVDDT) (1,1,KMEVD)
KNSLE	-/I2	ACTUAL NUMBER OF ENTRIES IN EVENT LIST (SLEVL) (2,2,KMSLE)
KNSLD	-/I2	ACTUAL NUMBER OF ENTRIES IN DOWNSTREAM STATION LINK LIST (SLDSL) (2,2,KMSLD)

Table 3-1. Global Variables -- SCIMAX (Page 7 of 58)

KNSLU	-/12	ACTUAL NUMBER OF ENTRIES IN UPSTREAM STATION LINK LIST (SLUSL) (2,2,KMSLU)

Table 3-1. Global Variables -- SCISL (Page 8 of 58)

```
SCISL:    STATION LINK DATA - INPUT
------------------------------------------------------------------------------
VAR NAME    DIM       DESCRIPTION
----------  --------  ---------------------------------------------------------
 ***NOTE:   STATION LINKS ARE IMPLICITLY ORDERED BY THE ORDER IN WHICH
            THEIR ATTRIBUTES ARE SPECIFIED IN INPUT.  EXAMPLE:  IF
            THE FIRST LINK DESCRIBED IS KNOWN AS THE UPSTREAM LINK
            THEN THAT LINK IS STATION LINK, SL, 1 AND THAT LINK'S
            ATTRIBUTES ARE SLTYPE(1), SLEVP(1), SLUSP(1), ETC.

SLCAP       KMSL/I2  STATION LINK CAPACITY (NUMBER OF VEHICLES)
                       (1,PMXTRL,)

SLTYPE      KMSL/I2  STATION LINK TYPE USED TO GROUP LINKS FOR
                       REPORTING PURPOSES AND FOR DIVERGE FUNCTIONS
                        -SLTYPE-    -MEANING-
                            1        IR    INPUT RAMP
                            2        IQ    INPUT QUEUE
                            3        D     DOCK (DEBOARD AND/OR BOARD)
                                           (DEBOARD, BOARD, AND JOINT EVENTS
                                           --3,4,5-- CAN APPEAR ONLY ON THIS
                                           LINK TYPE)
                            4        OQ    OUTPUT QUEUE
                            5        OR    OUTPUT RAMP
                            6        S     STORAGE
                            7        IS    INPUT-TO-STORAGE
                            8        SI    STORAGE-TO-INPUT
                            9        DS    DOCK-TO-STORE
                           10        SO    STORAGE-TO-OUTPUT
                           11        UL    UPSTREAM LINK (APPROACH LINK)
                           12        BL    BYPASS LINK
                           13        DL    DOWNSTREAM LINK
                           14        MIB   MODAL INPUT BEFORE PROCESSING
                           15        MIA   MODAL INPUT AFTER PROCESSING
                           16        MOB   MODAL OUTPUT BEFORE PROCESSING
                           17        MOA   MODAL OUTPUT AFTER PROCESSING
                       (1,1,17)

SLEVL       KMSLE/I2 STATION LINK EVENT LIST--A CONCATENATED LIST
                       OF SUBLISTS.  EACH SUBLIST LISTS EVENTS TO OCCUR
                       ON THE LINK AND ENDS IN ZERO.  SLEVP(SL) POINTS
                       TO THE START OF THE SUBLIST FOT THE STATION LINK.
                       THE EVENTS IN A SUBLIST MUST BE IN
                       THE ORDER 1/2/3/4/5/6/7/0, WHERE:
                         1 = HEADWAY
                         2 = TRAVEL
                         3 = DEBOARD
                         4 = BOARD
                         5 = JOINT (DEBOARD AND BOARD)
                         6 = STORE
```

Table 3-1. Global Variables -- SCISL (Page 9 of 58)

		7 = LAUNCH 0 = END OF LIST DELIMITER THE STORE AND LAUNCH EVENTS MUST BE THE LAST NON-ZERO EVENTS ON THEIR SUBLISTS. THE JOINT EVENT CANNOT BE USED WITH THE DEBOARD OR BOARD EVENTS IN THE SAME STATION. THE STORE EVENT MUST BE PRECEEDED BY THE HEADWAY OR TRAVEL EVENTS ON THE LINK ON WHICH IT APPEARS. (0,,) (SLEVL(SLEVP(I)-1)=0 ,I=2,KNSL) (SLEVL(SLEVP(I))¬=0 ,I=1,KNSL) (SLEVL(KMSLE)=0)
SLEVP	KMSL/I2	POINTER TO STARTING ENTRY IN STATION LINK EVENT LIST (SLEVL) FOR EACH STATION LINK (1,,)
SLUSL	KMSLU/I2	UPSTREAM STATION LINK LIST--A CONCATENATED LIST OF SUBLISTS. EACH SUBLIST LISTS THE UPSTREAM STATION LINKS THAT FEED INTO THE SL AND ENDS IN ZERO. SLUSP(SL) POINTS TO THE START OF THE SUBLIST FOR THE STATION LINK. WHEN AN UPSTREAM LINK IS A VEHICLE SOURCE, THE FOLLOWING VALUES ARE INPUT: -CODE- -MEANING- -1 SOURCE 1 IS THE GUIDEWAY -2 SOURCE 2 IS THE MODAL ENTRANCE BEFORE PROCESSING -3 SOURCE 3 IS THE MODAL ENTRANCE AFTER PROCESSING AS A MINIMUM, THE GUIDEWAY SOURCE MUST BE USED. (0,-3,) (SLUSL(SLUSP(I)-1)=0 ,I=2,KNSL) (SLUSL(SLUSP(I))¬=0 ,I=1,KNSL) (SLUSL(KMSLU)=0)
SLUSP	KMSL/I2	POINTER TO STARTING ENTRY IN THE UPSTREAM STATION LINK LIST (SLUSL) FOR EACH STATION LINK (1,,)
SLDSL	KMSLD/I2	DOWNSTREAM STATION LINK LIST--A CONCATENATED LIST OF SUBLISTS. EACH SUBLIST LISTS THE DOWNSTREAM SL'S THAT LEAVE THE STATION LINKS BEING DESCRIBED AND ENDS IN ZERO. SLDSP(SL) POINTS TO THE SUBLIST FOR THE STATION LINK. WHEN THE DOWNSTREAM LINK IS A VEHICLE SINK, THE FOLLOWING VALUES ARE USED:

```
                          -CODE-   -MEANING-
                            -1     SINK 1 IS THE GUIDEWAY
                            -2     SINK 2 IS THE MODAL OUTPUT BEFORE
                                   PROCESSING
                            -3     SINK 3 IS THE MODAL OUTPUT AFTER
                                   PROCESSING
                          AS A MINIMUM THE GUIDEWAY SINK MUST BE USED.
                          (0,-3,)
                          (SLDSL(SLDSP(I)-1)=0 ,I=2,KNSL)
                          (SLDSL(SLDSP(I))¬=0 ,I=1,KNSL)
                          (SLDSL(KMSLD)=0)

SLDSP       KMSL/I2    POINTER TO STARTING ENTRY IN THE DOWNSTREAM
                          STATION LINK LIST (SLDSL) FOR EACH STATION
                          LINK.
                          (1,,)

SLDIVC      KMSL/I2    DIVERGE FUNCTION NUMBER ASSIGNED TO A LINK
                          WITH TWO OR MORE DOWNSTREAM LINKS.  VALUES
                          RANGE FROM 1 THROUGH 6 WHERE USE OF THESE
                          FUNCTIONS IS TYPIFIED AS FOLLOWS:
                            1 = END OF UL
                            2 = END OF IR, MIB, SI
                            3 = END OF D
                            4 = END OF S
                            5 = ORDER BY OCCUPANCY
                            6 = ORDER BY PSEUDO-OCCUPANCY
                          ZERO IS USED WHEN THERE IS ONLY ONE DOWNSTREAM
                          LINK.
                          (1,1,6)

SLPF        KMSL/I2    PRIORITY/FIFO INDICATOR USED TO SPECIFY THE
                          ORDER IN WHICH QUEUED UPSTREAM VEHICLES
                          ARE DEQUEUED.
                          1===>PRIORITY (BASED ON THE ORDER IN WHICH
                                 UPSTREAM STATION LINKS ARE LISTED IN
                                 SLUSL)
                          2===>FIFO (BASED ON THE ORDER IN WHICH UPSTREAM
                                 VEHICLES BECOME QUEUED)
                          (1,1,2)

SLAVAL      KMSL/L1    INDICATES WHETHER SL IS AVAILABLE (ENABLED)
                          IN THE MODEL
                          T===>AVAILABLE
                          F===>NOT AVAILABLE
                          (.TRUE.,,)

SLPENT      KMSL/R4    PENALTY FACTOR TO BE MULTIPLIED BY TRAVEL EVENT
                          TIME ON THE SL TO DEGRADE THE SL
                          (1,0,)
```

Table 3-1. Global Variables -- SCISL (Page 11 of 58)

SLTTIM	KMSL/I4	TOTAL NON-DEGRADED TRAVEL TIME ON THE SL = HEADWAY EVENT TRAVEL TIME + TRAVEL EVENT TRAVEL TIME. IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0,0,)
SLHTA	KMSL/I4	HEADWAY TIME PER VEHICLE USED TO COMPUTE TIME TO TRAVEL THE HEADWAY ZONE. TOTAL HEADWAY ZONE TRAVEL TIME = SLHTA*(TRAIN LENGTH) + SLHTB. IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0,0,)
SLHTB	KMSL/I4	CONSTANT TERM USED TO COMPUTE TIME TO TRAVEL THE HEADWAY ZONE (SEE SLHTA). IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0,0,)

Table 3-1. Global Variables -- SCISYS (Page 12 of 58)

```
SCISYS:       SYSTEM DATA INPUT
-------------------------------------------------------------------------------
VAR NAME      DIM       DESCRIPTION
-----------   -------   -------------------------------------------------------
POLSER        -/I2      THE SERVICE POLICY IN EFFECT:
                          1===>DEMAND RESPONSIVE SINGLE PARTY
                          2===>DEMAND RESPONSIVE MULTIPARTY
                          3===>SCHEDULED
                          (1,1,3)

PVSPAC        -/I2      WHEN POLSER = 3:
                          VEHICLE DISPATCH SPACING ALGORITHM TO BE USED
                          FOR SPACING VEHICLES THAT WILL BE READY TO LEAVE
                          THE DOCKING AREA:
                          1===>MIDWAY BETWEEN THE TIME THE PREVIOUS
                               VEHICLE ON THE SAME ROUTE DID LEAVE AND THE
                               TIME AT WHICH THE FOLLOWING VEHICLE ON
                               THE ROUTE SHOULD LEAVE IF IT LEAVES ON
                               SCHEDULE.
                          2===>FIXED ROUTE DEPARTURE TIME

                          (1,1,2)

PNXSLV        KMR/I4    WHEN POLSER=3 AND
                        WHEN PVSPAC=1:
                          THE TIME AT WHICH THE VEHICLE
                          WHICH IS NOW BEING SCHEDULED (ON THIS
                          ROUTE) SHOULD LEAVE THE DOCK.
                        WHEN POLSER=3 AND
                        WHEN PVSPAC=2:
                          THE TIME THE LAST VEHICLE ON
                          THE ROUTE WAS SCHEDULED TO LEAVE THE
                          DOCK. (INTERNAL PROGRAM ALIAS IS PLSCHT.)
                        WHEN THE SIMULATION BEGINS THIS IS THE TIME THAT
                        THE FIRST VEHICLE LEAVING THE DOCK SHOULD LEAVE.
                        IF IT HAS A VALUE OF ZERO, THE FIRST VEHICLE WILL,
                        BY DEFINITION LEAVE WHEN IT CAN AND WILL BE ON TIME.
                        IF IT HAS A NON-ZERO VALUE, THE
                        USER WILL HAVE DETERMINED WHEN THE FIRST VEHICLE
                        SHOULD LEAVE THE DOCK.  THE FIRST VEHICLE COULD
                        SUBSEQUENTLY BE BEHIND OR ON TIME BASED ON THIS
                        USER INPUT.
                        IT IS INPUT BY THE USER IN SECONDS AND CONVERTED
                        TO CLOCK UNITS BY THE INPUT PROCESSOR.
                        (0,,)

PLSTLV        KMR/I4    WHEN POLSER=3 AND
                        WHEN PVSPAC=1:
                          THE TIME AT WHICH THE PRECEEDING VEHICLE
                          ON THIS ROUTE WAS SCHEDULED TO LEAVE THE DOCK.
```

```
                              INITIALIZED BY THE INPUT PROCESSOR TO:
                                PLSTLV(I)=PNXSLV(I)-PRTEHW(I),  I=1,KMR
                              IN CLOCK UNITS.

PRTEHW       KMR/I4     WHEN POLSER = 3:
                          DESIRED HEADWAY BETWEEN VEHICLES ON THE SAME ROUTE
                          IT IS INPUT BY THE USER IN SECONDS AND CONVERTED
                          TO CLOCK UNITS BY THE INPUT PROCESSOR.
                          (0,0,)

PVRLST       KMRT/I2    WHEN POLSER = 3:
                          ROUTES' STATION LIST--CONCATENATED LIST
                          OF SUBLISTS.  EACH SUBLIST LISTS
                          THE STATIONS ON THE ROUTE AND ENDS IN ZERO.
                          PVRPTR(ROUTE) POINTS TO THE START OF THE SUBLIST
                          FOR THE ROUTE. USED TO MATCH TRIPS TO VEHICLE
                          ROUTES WHEN THE STATION ROUTE ASSIGNMENT TABLE
                          (PRASGN) HAS NOT BEEN INPUT BY THE USER.
                          (0,,)
                          (PVRLST(PVRPTR(I)-1)=0,   I=2,KMR)
                          (PVRLST(PVRPTR(I)) ¬=0,   I=1,KMR)
                          (PVRLST(KMRT)=0)

PVRPTR       KMR/I2     WHEN POLSER = 3:
                          POINTER TO STARTING ENTRY (HOME STATION)
                          FOR EACH ROUTE IN THE ROUTE'S STATION LIST
                          (PVRLST).
                          (0,,)

PRASGN       KMS/I2     WHEN POLSER = 3:
                         STATION ROUTE ASSIGNMENT TABLE USED TO DETERMINE
                         DESTINATION COMPATIBILITY OF TRIPS AND VEHICLES.
                         PRASGN(I)=ROUTE NUMBER SATISFYING TRIPS
                                   GOING FROM STSIM TO STATION I
                          (0,0,KNR)

PNMDDP       KMNMD      NETWORK MERGE DELAY DISTRIBUTIION:
              /R4         CUMULATIVE PROBABILITY DISTRIBUTION SUCH
                          THAT:
                           PNMDDP(I)=PROBABILITY(DELAY DUE TO
                                         HAVING TO ARRANGE MERGES
                                         IN THE REST OF THE NETWORK)
                                         <=PNMDDT(I)
                          (0,0,)
                          (0 =< PNMDDP(I) =< PNMDDP(I+1) =< 1.0,I=1,KNNMD-1)
                          (INPUT AS FREQ.DIST.& CONVERTED BY IP TO CUM.DIST)

PNMDDT       KMNMD      NETWORK MERGE DELAY DISTRIBUTION:
              /I4         PNMDDT(I)=DELAY TIME
                        IT IS INPUT BY THE USER IN SECONDS AND CONVERTED
                        TO CLOCK UNITS BY THE INPUT PROCESSOR.
                          (0,0,)

PEVDDP      KMEVD       EMPTY VEHICLE DELAY DISTRIBUTION:
             /R4          CUMULATIVE PROBABILITY FUNCTION SUCH THAT:
                           PEVDDP(I)=PROBABILITY(DELAY IN HAVING
                                          AN EMPTY VEHICLE COME TO
                                          SERVICE THE TRIP) <=
                                          PEVDDT(I)
                          (0,0,)
                          (0 =< PEVDDP(I) =< PEVDDP(I+1) =< 1.0,I=1,KMEVD-1)
                          (INPUT AS FREQ.DIST.& CONVERTED BY IP TO CUM.DIST)

PEVDDT      KMEVD       EMPTY VEHICLE DELAY DISTRIBUTION:
             /I4          PEVDDT(I)=DELAY TIME
                        IT IS INPUT BY THE USER IN SECONDS AND CONVERTED
                        TO CLOCK UNITS BY THE INPUT PROCESSOR.
                          (0,0,)

PCMETH       -/I2       WHEN POLSER = 3
                        OR WHEN POLSER = 2:
                          METHOD TO DETERMINE COMPATIBILITY BETWEEN TRIPS
                          AND VEHICLES:
                            0===>PROBABILITY OF COMPATIBILITY
                            1===>TABLE OF STATIONS ON EACH ROUTE (USING
                                 PRASGN IF SPECIFIED OTHERWISE USING PVRLST)
                          (0,0,1)

PCOMPD       2/R4       WHEN POLSER = 3 AND  PCMETH=0
                        OR WHEN POLSER=2:
                          CUMULATIVE PROBABILITY FUNCTION SUCH THAT
                          PCOMPD(1)=PROBABILITY OF A COMPATIBLE VEHICLE
                          PCOMPD(2)=1
                          (0,0,1)
                          (0 =< PCOMPD(1) =< PCOMPD(2) =< 1.0)
                          (INPUT AS FREQ.DIST.& CONVERTED BY IP TO CUM.DIST)

PXFERI       -/L1       WHEN POLSER=3 OR
                        WHEN POLSER = 2:
                          INDICATES WHETHER OR NOT TRANSFERS ARE TO BE
                          ALLOWED:
                            F===>NO TRANSFERS ARE TO BE CONSIDERED
                            T===>TRANSFERS MAY OCCUR AS DEFINED BY
                                 THE TRANSFER DISTRIBUTION (PXFERD).
                          (F,,)*

PXFERD       2/R4       WHEN POLSER=3 OR WHEN POLSER = 2
                        AND WHEN PXFERI = T:
                          CUMULATIVE PROBABILITY FUNCTION WHERE:
                          PXFERD(1)=PROBABILITY OF A TRIP HAVING TO
```

Table 3-1. Global Variables -- SCISYS (Page 15 of 58)

		DEBOARD AT STSIM TO TRANSFER TO ANOTHER VEHICLE PXFERD(2)=1 (0,0,1) (0 =< PXFERD(1) =< PXFERD(2) =< 1.0) (INPUT AS FREQ.DIST.& CONVERTED BY IP TO CUM.DIST)
PNEEDD	2/R4	CUMULATIVE PROBABILITY FUNCTION SUCH THAT PNEEDD(1)=PROBABILITY THAT THE EMPTY VEHICLE BEING CONSIDERED WILL BE NEEDED AT ANOTHER STATION PNEEDD(2)=1 (0,0,1) (0 =< PNEEDD(1) =< PNEEDD(2) =< 1.0) (INPUT AS FREQ.DIST.& CONVERTED BY IP TO CUM.DIST)
PVEPR	KMEVP /I2	INDICATION AS TO THE EMPTY VEHICLE MANAGEMENT METHOD TO BE USED 0===>ATTEMPT TO SEND EMPTY VEHICLES TO LOCAL STORAGE 1===>SEND ALL EMPTY VEHICLES OUT OF THE STATION (0,0,1)
PVSPR	KMSVP /I2	WHEN POLSER = 1 OR 2: ORDERED LIST OF WHERE TO SEARCH FOR AN EMPTY VEHICLE: PVSPR(1)=FIRST PLACE TO LOOK PVSPR(2)=SECOND PLACE TO LOOK ... PVSPR(KNSVP)=LAST PLACE TO LOOK. 1===>LOCAL STORAGE 2===>FETCH EMPTY VEHICLE FROM ELSEWHERE IN NETWORK (ALSO THE DEFAULT METHOD IF NONE WERE SPECIFIED) 3===>LOOK AT SL'S ON SEARCH LIST FOR EMPTIES (PSLIST) ((1,2,3),1,3)
PSLIST	KMSL/I2	WHEN POLSER = 1 OR 2: SEARCH LIST--THE LIST OF STATION LINKS TO BE SEARCHED WHEN LOOKING FOR AN EMPTY VEHICLE TO SERVICE A WAITING TRIP. END OF LIST DELIMITED BY ZERO. (0,0,KNSL)
STSIM	-/I2	THE STATION NUMBER OF THE STATION BEING SIMULATED. THE VALUE ENTERED FOR THE STATION SHOULD ALSO BE THE ORIGIN STATION FOR TRIPS WALKING INTO THE

```
                        STATION, THE DESTINATION OF TRIPS RIDING INTO THE
                        STATION AND DEBOARDING TO LEAVE THE STATION, THE
                        NEXT STOP FOR VEHICLES ENTERING THE STATION TO
                        DEBOARD AND BOARD TRIPS, AND A STOP ON THE ROUTES
                        WHICH PASS THROUGH THE STATION.
                        (1,1,)

STYPE         -/L1    TYPE OF STATION:
                        F===>OFFLINE
                        T===>ONLINE
                        OFFLINE STATIONS CONTAIN A BYPASS LINK AND ONLINE
                        STATIONS DO NOT.  VEHICLES NOT STOPPING AT ONLINE
                        STATIONS AVOID THE DEBOARDING AND BOARDING OF
                        TRIPS EVEN THOUGH THEY PASS THROUGH THE STATION.
                        VEHICLES NOT STOPPING AT AN OFFLINE STATION CAN
                        TAKE THE BYPASS LINK AROUND THE STATION.
                        (F,,)

***NOTE:  DEBOARDING AND BOARDING DELAY CALCULATIONS DESCRIBED BELOW
          ARE USED IN PART TO DETERMINE THE BOARD, DEBOARD, AND
          JOINT EVENT TIMES FOR VEHICLES AND TRAINS.  THE FOLLOWING
          DELAY CALCULATIONS APPLY TO AN INDIVIDUAL VEHICLE.  THE
          DELAY FOR A TRAIN IS EQUAL TO THAT OF THE SLOWEST VEHICLE
          IN THE TRAIN.
               LET THE DELAY = N(U,V), VIZ., BE A NORMALLY DISTRIBUTED
                   RANDOM VARIABLE WITH MEAN = U AND STANDARD DEVIATION
                   = V.
               THEN:
                 (1) THE TIME FOR THE SEPERATE DEBOARD EVENT
                        = N(UD,STDBSD)
                        WHERE: UD=STDBA*NO. OF PASS DEBOARDING + STDBC
                 (2) THE TIME FOR THE SEPERATE BOARD EVENT
                        = N(UB,STBSD)
                        WHERE: UB=STBA*NO. OF PASS BOARDING + STBC
                 (3) THE TIME FOR THE JOINT DB & B EVENT
                        = MAX( N(UD,STDBSD), N(UB,STBSD)+STDLAY )
                      WHERE:
                        UD=STDBA*NO. OF PASS DEBOARDING
                           + STDBB*NO. PASS BOARDING
                           + STDBC
                           + FLDI4(1)*NO.PASS BOARDING*NO.PASS DEBOARDING
                        UB=STBA*NO. OF PASS BOARDING
                           + STBB*NO. OF PASS DEBOARDING
                           + STBC
                           + FLDI4(2)*NO.PASS BOARDING*NO.PASS DEBOARDING

STDBA         -/I4    DEBOARD TIME PER DEBOARDING PASSENGER USED IN
                        COMPUTING THE DEBOARD TIME DELAY FOR THE DEBOARD
                        AND JOINT EVENTS.(OPTIONAL; USED WHEN THE STATION
                        HAS LINKS CONTAINING THE JOINT OR DEBOARD EVENTS.)
```

		IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0.,)
STDBB	-/I4	DEBOARD TIME PER BOARDING PASS. USED IN COMPUTING THE DEBOARD DELAY FOR THE JOINT EVENT.(OPTIONAL; USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT EVENT.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0.,)
STDBC	-/I4	DEBOARD TIME PER DEBOARD VEHICLE USED IN COMPUTING THE DEBOARD TIME DELAY FOR THE DEBOARD AND JOINT EVENTS.(OPTIONAL; USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT OR DEBOARD EVENTS.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0.,)
FLDI4 FLDI4(1)	10/I4	FLDI4(1)=QUADRATIC COEFFICIENT USED IN COMPUTING THE DEBOARD TIME DELAY FOR THE JOINT EVENT. (OPTIONAL; USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT EVENT.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0.,)
FLDI4 FLDI4(2)		FLDI4(2)=QUADRATIC COEFFICIENT USED IN COMPUTING THE BOARD TIME DELAY FOR THE JOINT EVENT. (OPTIONAL; USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT EVENT.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0.,)
STBA	-/I4	BOARD TIME PER BOARDING PASSENGER USED IN COMPUTING THE BOARD TIME DELAY FOR THE BOARD AND JOINT EVENTS.(OPTIONAL; USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT OR BOARD EVENTS.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0.,)
STBB	-/I4	BOARD TIME PER DEBOARDING PASS. USED IN COMPUTING THE BOARD DELAY FOR THE JOINT EVENT.(OPTIONAL; USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT EVENT.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR.

		(0,,)
STBC	-/I4	BOARD TIME PER BOARD VEHICLE USED IN COMPUTING THE BOARD TIME DELAY FOR THE BOARD AND JOINT EVENTS.(OPTIONAL; USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT OR BOARD EVENTS.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0,,)
STDBSD	-/R4	STANDARD DEVIATION OF DEBOARD DELAY TIME USED IN COMPUTING THE DEBOARD TIME DELAY FOR THE DEBOARD AND JOINT EVENTS. (OPTIONAL: USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT OR DEBOARD EVENTS.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0,0,)
STBSD	-/R4	STANDARD DEVIATION OF BOARD DELAY TIME USED IN COMPUTING THE BOARD TIME DELAY FOR THE BOARD AND JOINT EVENTS. (OPTIONAL: USED WHEN THE STATION HAS LINKS CONTAINING THE JOINT OR BOARD EVENTS.) IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (0,0,)
STDLAY	-/I4	DELAY BETWEEN THE TIME THE DEBOARD EVENT IS TO START AND THE BOARD EVENT IS TO START WHEN COMPUTING THE BOARDING TIME DELAY FOR THE JOINT EVENT. (REQUIRED ONLY WHEN THE STATION HAS LINKS CONTAINING THE JOINT EVENT.) WHEN A VALUE OF 0 IS ENTERED, COMMON DEBOARD/BOARD IS IMPLIED. WHEN A VALUE GREATER THAN ZERO IS ENTERED, FLUSH DEBOARD/BOARD IS IMPLIED. IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (-1,-1,)
SLSTOR	-/I2	THE STATION LINK NUMBER OF THE STATION LINK DESIGNATED AS THE STORAGE LINK. THIS VALUE IS INITIALIZED BY THE INPUT PROCESSOR TO THE STATION LINK CONTAINING THE STORE EVENT.
PL2IND	-/L1	USER'S INDICATION AS TO WHETHER OR NOT THE DELAY TO PLAN THE LOCAL MERGE OF THE VEHICLE GOING ON THE OUTPUT RAMP WITH THOSE ON THE BYPASS LINK IS TO BE INCLUDED IN THE LAUNCH DELAY. F===>DO NOT INCLUDE "LOCAL" MERGE DELAY T===>INCLUDE

Table 3-1. Global Variables -- SCISYS (Page 19 of 58)

		THE STATION MUST BE PROPERLY CONFIGURED FOR LOCAL MERGING TO BE PLANNED, E.G., HAVE AN OUTPUT RAMP AND A BYPASS LINK (THE SECOND LONGER THAN THE FIRST), BOTH OF WHICH FEED INTO THE DOWNSTREAM LINK. HEADWAY INFORMATION MUST BE SUPPLIED FOR THE BYPASS LINK. (F,,)
PENTS	-/L1	WHEN POLSER=1 OR 2: INDICATOR OF WHETHER OR NOT ENTRAINMENT AND DETRAINMENT ARE TO BE DONE IN THE STATION. F===>NO ENTRAINMENT/DETRAINMENT TO BE DONE T===>ENTRAINMENT/DETRAINMENT TO BE DONE THE STATION MUST CONTAIN AT LEAVE ONE DEBOARD OR JOINT EVENT AND AT LEAST ONE LAUNCH EVENT WHEN PENTS=T SINCE DETRAINMENT IS DONE BEFORE DEBOARDING AND ENTRAINMENT IS DONE AFTER LAUNCH. (F,,)
VCAP	-/I2	THE MAXIMUM NUMBER OF PASSENGERS A VEHICLE CAN ACCOMMODATE (6,0,)
PMXTRL	-/I2	THE MAXIMUM NUMBER OF VEHICLES IN A TRAIN. THIS MUST BE AT LEAST AS LARGE AS THE LARGEST TRAIN GENERATED BY THE INPUT PROCESSOR (KMTLEN). (1,,)
AKSEED	-/I4	THE STARTING SEED TO THE RANDOM NUMBER GENERATOR MUST BE AN ODD INTEGER GREATER THAN OR EQUAL TO THREE. (3,3,)
ASTATU	-/I2	NUMBER OF SAMPLING INTERVALS PER INTERMEDIATE SAMPLING REPORT (5,1,)
PTSPLT	-/I2	TRIP SPLIT SIZE; ANY TRIP ENTERING THE STATION WHICH IS LARGER THAN PTSPLT WILL BE SPLIT INTO AS MANY TRIPS OF PTSPLT PASSENGERS AS POSSIBLE AND ONE SMALLER TRIP WITH THE REMAINING PASSENGERS. PTSPLT SHOULD BE NO LARGER THAN THE CAPACITY OF THE TRIP LINKS (UCAP) AND THE CAPACITY OF THE VEHICLES (VCAP). (VCAP,1,VCAP)
ASAMPI	-/I4	SAMPLING INTERVAL AT WHICH STATISTICS ARE RECORDED. A VALUE OF ZERO IMPLIES NO SAMPLES ARE TAKEN. IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR.

		(60,0.)
ACKPTI	-/I4	PERIODIC CHECKPOINT INTERVAL AT WHICH A CHECKPOINT IS TAKEN. DEFAULT IMPLIES NO CHECKPOINT IS TAKEN. IT IS INPUT BY THE USER IN SECONDS AND CONVERTED TO CLOCK UNITS BY THE INPUT PROCESSOR. (9999999,0.)
AFLAG	KMFLAG/ L1	DBUG FLAGS (INPUT VIA ASYNCHRONOUS DATA READ) (F,,)
AFAIL	4/I2	FAILURE DATA: AFAIL(1)= NOT APPLICABLE IF STATION LINK: AFAIL(2) IS THE STATION LINK NUMBER OF THE STATION LINK BEING FAILED, DEGRADED, OR RECOVERED. IF TRIP LINK: AFAIL(2) IS THE TRIP LINK NUMBER OF THE TRIP LINK BEING FAILED, DEGRADED, OR RECOVERED. IF STATION LINK: AFAIL(3) = 1 IF STATION LINK ENTRY IS FAILED AFAIL(3) = 2 IF STATION LINK EXIT IS FAILED AFAIL(3) IS NOT APPLICABLE IF DEGREDATION OR DEGRADATION RECOVERY. IF STATION LINK OR IF TRIP LINK: AFAIL(4) = 1 IF FAILURE AFAIL(4) = 2 IF RECOVERY AFAIL(4) = 3 IF DEGRADATION AFAIL(4) = 4 IF DEGRADATION RECOVERY THIS DATA IS ENTERED ASYNCHRONOUSLY AT FAILURE TIME AND OFTEN FOLLOWED BY OTHER DATA ON GDIP FORMAT TO UPDATE ADDITIONAL DATA ITEMS SUCH AS NUMBER OF SERVERS (USERV) FOR TRIPS, AND PENALTY TRAVEL FACTOR (SLPENT) FOR VEHICLES. (0,,)

AVSOUR	3/L1	VEHICLE ARRIVAL SOURCE; THESE VALUES FOR AVSOUR(1), AVSOUR(2), AND AVSOUR(3) ARE INTIALIZED BY THE INPUT PROCESSOR BASED ON THE UPSTREAM STATION LINK LIST (SLUSL) OR CONFIGUATOR. TO ENTER THEM DIRECTLY TO THE MODEL PROCESSOR THE FOLLOWING DEFINITIONS APPLY:
		AVSOUR(1) IS THE UPSTREAM LINK (UL)
		AVSOUR(2) IS THE MODAL INPUT BEFORE PROCESSING (MIB)
		AVSOUR(3) IS THE MODAL INPUT AFTER PROCESSING (MIA)

		AVSOUR FOR THE THREE SOURCES CAN HAVE THE FOLLOWING VALUES: T===>VEHICLES ENTERING THIS WAY F===>NO VEHICLES ENTERING THIS WAY
SLSOUR	3/I2	STATION LINK SOURCE; THESE VALUES FOR SLSOUR(1), SLSOUR(2), AND SLSOUR(3) ARE INITIALIZED BY THE INPUT PROCESSOR BASED ON THE UPSTREAM STATION LINK LIST (SLUSL) OR CONFIGURATOR. TO ENTER THEM DIRECTLY INTO THE MODEL PROCESSOR THE FOLLOWING DEFINITIONS APPLY: SLSOUR(1) IS THE UL STATION LINK NUMBER SLSOUR(2) IS THE MIB STATION LINK NUMBER SLSOUR(3) IS THE MIA STATION LINK NUMBER
ATVF	-/L1	INDICATOR AS TO WHETHER OR NOT THE TRIP AND VEHICLE FILE IS REQUESTED AS OUTPUT. T===>WRITE FILE F===>DO NOT WRITE FILE (F,,)
FLDI4(3) SEE FLDI4		FLDI4(3) MINIMUM EDGE OF FEL STATISTICS HISTOGRAM REPORTED IN THE FINAL MODEL REPORT AND USED TO FINE TUNE THE FUTURE EVENTS LIST FOR PROCESSING EFFICIENCY. THIS VALUE IS INITIALIZED BY THE INPUT PROCESSOR. TO ENTER IT DIRECTLY INTO THE MODEL PROCESSOR ENTER IT IN CLOCK UNITS. (CLSMAL*KMCLTA,0,)
FLDI4(4) SEE FLDI4		FLDI4(4) WIDTH EDGE OF FEL STATISTICS HISTOGRAM REPORTED IN THE FINAL MODEL REPORT AND USED TO FINE TUNE THE FUTURE EVENTS LIST FOR PROCESSING EFFICIENCY. THIS VALUE IS INITIALIZED BY THE INPUT PROCESSOR. TO ENTER IT DIRECTLY INTO THE MODEL PROCESSOR ENTER IT IN CLOCK UNITS. (CLSMAL*KMCLTA,0,)
FLDI4 FLDI4(5)-(10)	10/I4	FLDI4(5) THRU FLDI4(10) UNUSED (0,,)
FLDI2	10/I2	FLDI2(1) THRU FLDI2(10) UNUSED (0,,)
FLDL1	10/L1	FLDL1(1) THRU FLDL1(10) UNUSED (F,,)
FLDR4	10/R4	FLDR4(1) THRU FLDR4(10) UNUSED (0,,)

```
SCITL:        TRIP LINK DATA
-----------------------------------------------------------------------------
VAR NAME      DIM      DESCRIPTION
-----------   -------  ------------------------------------------------------
                  ***NOTE:  THERE ARE THREE TRIP LINKS (KMTL = 3).  THEY
                       ARE KNOWN AS:
                         1 = TICKETING LINK (TKL)
                         2 = TURNSTILE LINK (TSL)
                         3 = BOARDING LINK (BGL)

UCAP          KMTL/I2 CAPACITY OF THE TRIP LINK IN PASSENGERS.  THIS MUST
                         BE AT LEAST AS LARGE AS THE LARGEST TRIP THAT WILL
                         BE WALKING INTO THE STATION OR TRANSFERRING FROM
                         A VEHICLE.
                         (1,1,)

UTIM          KMTL/I4 WALK TIME ON TRIP LINK.
                         IT IS INPUT BY THE USER IN SECONDS AND CONVERTED
                         TO CLOCK UNITS BY THE INPUT PROCESSOR.
                         (0,0,)

                  ***NOTE:  TO DETERMINE THE AVERAGE AMOUNT OF TIME
                       FOR A TRIP TO GO THROUGH TICKETING (LINK 1
                       PROCESSING) AND THE TURNSTILE (LINK 2 PROCESSING),
                       THE FOLLOWING EQUATIONS ARE USED:
                         PROCESSING TIME =
                           ((UTIMA * NO. OF PASS. IN TRIP / USERV)
                            + UTIMB).
                       PROCESSING TIME IS NOT APPLICABLE TO THE BOARDING
                       LINK (LINK 3).

UTIMA         KMTL/I4 COEFFICIENT TERM FOR THE NUMBER OF PASSENGERS IN
                         THE TRIP USED IN CALCULATING PROCESSING TIME ON
                         THE TRIP LINK.
                         IT IS INPUT BY THE USER IN SECONDS AND CONVERTED
                         TO CLOCK UNITS BY THE INPUT PROCESSOR.
                         (0,0,)

UTIMB         KMTL/I4 CONSTANT TERM FOR THE TRIP USED IN CALCULATING
                         PROCESSING TIME ON THE TRIP LINK.
                         IT IS INPUT BY THE USER IN SECONDS AND CONVERTED
                         TO CLOCK UNITS BY THE INPUT PROCESSOR.
                         (0,0,)

USERV         KMTL/I2 NUMBER OF SERVERS CURRENTLY ACTIVE ON THE
                         TRIP LINK (FOR TICKETING, LINK 1, AND TURNSTILE,
                         LINK 2, ONLY.) NOT APPLICABLE TO THE BOARDING
                         LINK (LINK 3).  WHEN A LINK IS FAILED, USERV IS
                         AUTOMATICALLY SET TO ZERO IN THE MODEL.  TO
                         DEGRADE THE LINK, A REDUCED VALUE FOR USERV MUST
```

```
BE SPECIFIED IN ADDITION TO AFAIL(2) AND AFAIL(4).
TO RECOVER FROM A DEGRADED OR FAILED LINK, AN
INCREASED VALUE FOR USERV MUST BE SPECIFIED IN
ADDITION TO AFAIL(2) AND AFAIL(4).
(0,0,)
```

Table 3-1. Global Variables -- SCMFEL (Page 24 of 58)

SCMFEL: FUTURE EVENT LIST INTERNAL DATA

VAR NAME	DIM	DESCRIPTION
CLPOS	-/I4	CURRENT POSITION IN CLOCK TABLE (1=<CLPOS=<KMCLTA)
CLBASE	-/I4	BASE TIME VALUE FOR FIRST ENTRY IN CLOCK TABLE (CU*10)
CLMINI	-/I4	TIME (CU*10) OF CURRENT CLOCK TABLE INTERVAL GIVEN BY CLPOS (CLMINI=CLBASE+CLSMAL(CLPOS-1))
CLSCAN	-/L1	FLAG FOR USE IN SCANNING CLOCK TABLE WHEN RESCAN OF CLOCK TABLE REQUIRED UPON REACHING END(POS<CLSIZE) 0=F,1=T
CLTABL	KMCLTA /I2	CLOCK TABLE - LIST HEAD POINTER TO XTNS ACTIVE IN CLOCK TABLE INTERVAL CLMINI
CLSIZE	-/I4	NUMBER OF ENTRIES IN CLOCK TABLE
CNFEL	-/I4	NUMBER OF ENTRIES IN FUTURE EVENT LIST
CLOCK	-/I4	SIMULATION CLOCK - CURRENT TIME
CLSTAT	3,10/I4	FUTURE EVENTS TIMING STATISTICS(I,J) I=1 NUMBER OF ENTRIES X(I) =2 SUM X(I) =3 SUMSQ X(I) J=1 CLOCK TABLE INSERTIONS =2 MULTIPLE THREAD CHAIN INSERTIONS =3 DELTA T FOR (1) =4 DELTA POSITIONS SKIPPED FOR (1) =5 NUMBER OF M/T RELOADS OF CLOCK TABLE =6 DELTA POSITIONS SKIPPED FOR (2) =7 M/T LOOP SIZE (#XTNS) =8 UNUSED =9 UNUSED =10 UNUSED
CMTHRD	-/I2	MULTIPLE THREAD LIST HEAD

Table 3-1. Global Variables -- SCMFS (Page 25 of 58)

SCMFS: FEL STATISTICS INTERNAL DATA

VAR NAME	DIM	DESCRIPTION
FELHST	10/I4	HISTOGRAM OF THE DELTA T'S SCHEDULED ON THE FEL WITH SAPFEL;FELHST(10)=TOTAL NUMBER PUT ON FEL
MCLBIG	-/I4	MINIMUM EDGE OF FEL STATISTICS HISTOGRAM; SET EQUAL TO FLDI4(3) IN SANFEL
WIDTH	-/I4	WIDTH OF FEL STATISTICS HISTOGRAM; SET EQUAL TO FLDI4(4) IN SANFEL

Table 3-1. Global Variables -- SCMSL (Page 26 of 58)

```
SCMSL:    STATION LINK DATA - MODEL PROCESSOR
--------------------------------------------------------------------------------
VAR NAME    DIM       DESCRIPTION
----------- -------- ----------------------------------------------------------
SLMEMT      KMSL/I2  POINTER TO THE TAIL OF THE MEMBERSHIP
                       CHAIN OF THIS STATION LINK

SLHZF       KMSL/L1  HEADWAY ZONE FLAG FOR THIS STATION LINK
                       F===>HEADWAY ZONE IS NOT OCCUPIED
                       T===>HEADWAY ZONE IS OCCUPIED

SLEXIT      KMSL/L1  INDICATES WHETHER EXIT OF STATION LINK IS FAILED
                       OR ACTIVE
                         F===>ACTIVE
                         T===>FAILED

SLENT       KMSL/L1  INDICATES WHETHER ENTRY OF STATION LINK IF FAILED
                       OR ACTIVE
                         F===>ACTIVE
                         T===>FAILED

SLPOCC      KMSL/I2  PSUEDO-OCCUPANCY IS MAINTAINED ONLY FOR
                       STATION LINK*S WITH DB/B EVENTS AND EQUAL TO THE
                       CAPACITY MINUS THE NUMBER OF AVAILABLE
                       UPSTREAM BERTHS
```

Table 3-1. Global Variables -- SCMSYS (Page 27 of 58)

SCMSYS: SYSTEM DATA MODEL

VAR NAME	DIM	DESCRIPTION
ADONES	-/L1	RETURNED BY SSMOD AND USED TO INDICATE: F===>NOT_DONE - THERE ARE MORE VEHICLE EVENTS TO OCCUR TO THE VEHICLE ON ITS SL T===>DONE - THERE ARE NO MORE VEHICLE EVENTS TO OCCUR TO THE VEHICLE ON ITS SL
AENTRS	-/L1	RETURNED BY SSTEST AND USED TO INDICATE: F===>CANNOT_ENTER T===>CAN_ENTER
ADONET	-/L1	RETURNED BY SUMOD AND USED TO INDICATE: F===>NOT_DONE - THERE ARE MORE TRIP EVENTS TO OCCUR TO THE TRIP ON ITS TL T===>DONE - THERE ARE NO MORE TRIP EVENTS TO OCCUR TO THE TRIP ON ITS TL
AENTRT	-/L1	RETURNED BY SUTEST AND USED TO INDICATE: F===>CANNOT_ENTER T===>CAN_ENTER
AVNXSL	-/I2	NEXT SL OF THE VEHICLE BEING PROCESSED IS TO USE
ATNXTL	-/I2	THE NEXT TL THAT THE TRIP BEING PROCESSED IS TO USE
ARANDN	-/R4	THE RANDOM NUMBER GENERATED BY SMRNG; IT IS UNIFORMLY DISTRIBUTED BETWEEN 0 & 1; (REAL*4)
ATREC	-/I2	NUMBER OF TRIP SYSTEM SERVICE TRANSACTION
AVREC	3/I2	NUMBER OF VEHICLE SYSTEM SERVICE TRANSACTION FOR FROM EACH SOURCE 1 = ON GUIDEWAY UPSTREAM OF STATION 2 = MODAL ENTRY BEFORE PROCESSING 3 = MODAL ENTRY AFTER PROCESSING
ACARD	-/I2	NUMBER OF ASYNCHRONOUS SYSTEM SERVICE TRANSACTION
AEND	-/L1	LOGIC VARIABLE TO INDICATE THAT THE SIMULATION IS TO END
ASLLST	KMSLDS/ I2	LIST OF DOWNSTREAM SL'S PASSED BETWEEN SSTEST AND SMDIVF
SBQTL	-/I4	POINTER TO TAIL OF TRIPS READY TO BOARD VEHICLES (TRIPS WILL BE QUEUED TO THIS USING TQUECH)
NAME	-/I4	NAME OF THE MOST RECENTLY READ ASYNCHRONOUS HEADER CARD
SBEVTL	-/I4	HEAD USED TO CHAIN ALL VEHICLE IN THE THAT ARE IN THE BOARD EVENT
ADLST	KMTLEN /I2	LIST OF VEHICLES IN DETRAINED TRAIN; END OF LIST MARKED BY 0
AGPLST	KMSECAP ,2/I4	THIS TABLE IS A WRAP-AROUND LIST OF GAPS IN THE BYPASS LINK ELIGIBLE FOR VEHICLES AWAITING A LOCAL MERGE TO ATTEMPT TO MERGE INTO. EACH PAIR OF TIMES IS THE GAP'S START AND END TIME --- THE TIMES AT WHICH THE START OF THE GAP AND THE END OF THE GAP WILL HAVE FINISHED TRAVELING

```
                         IN THE BYPASS LINK.
GPAVTL       -/I4     POINTER TO THE NEXT AVAILABLE GAP
                        AFTER MODULO OPERATION

GPADTL       -/I4     POINTER TO THE LAST GAP ADDED TO AGPLST
                        AFTER MODULO OPERATION

SCMIDX:     INDEX DATA FOR MODEL PROCESSOR
-----------------------------------------------------------------------------
VAR NAME     DIM      DESCRIPTION
------------ -------- ---------------------------------------------- --------
AMNAME       10,8/L1 MEMBER NAMES PARSED FROM PARM FIELD
AMFLAG       10/L1    INDICATORS AS TO WHETHER OR NOT FILES
                        ARE USED
```

Table 3-1. Global Variables -- SCMT (Page 29 of 58)

SCMT: TRIP DATA - MODEL PROCESSOR

VAR NAME	DIM	DESCRIPTION
TARRT	KMT/I4	ARRIVAL TIME OF THE TRIP
TORIG	KMT/I2	ORIGIN STATION OF THE TRIP
TDEST	KMT/I2	DESTINATION OF THE TRIP (FINAL)
TPASS	KMT/I2	NUMBER OF PASSENGERS ON THE TRIP
TMEMCH	KMT/I2	USED TO CHAIN TRIPS THAT ARE A MEMBERS OF A TRIP LINK
TQREAS	KMT/I2	REASON THE TRIP IS QUEUED: 0===>TRIP ON FEL; ONLY HEAD TRIP ON TRIP LINK CAN BE ON FEL 1===>TRIP QUEUED DUE TO CONGESTION OR FAILURE; ONLY THE HEAD TRIP ON A TRIP LINK CAN HAVE THIS TQREASON 2===>TRIP QUEUED DO TO TRIP IN FRONT OF IT & OTHERWISE DONE --- NO TRIP CAN HAVE THIS TQREASON SINCE THE TRIP MUST BE AT THE HEAD OF ITS TRIP LINK IN ORDER TO START ITS PROCESSING EVENT 3===>TRIP QUEUED DUE TO WAITING TO START ITS PROCESSING EVENT 4===>TRIP QUEUED IN BOARDING QUEUE OR ON VEHICLE'S TRIP QUEUE

Table 3-1. Global Variables -- SCMTL (Page 30 of 58)

SCMTL: TRIP LINK DATA - MODEL PROCESSOR

VAR NAME	DIM	DESCRIPTION
UMEMTL	KMTL/I2	POINTER TO TAIL OF CHAIN OF TRIPS THAT ARE MEMBERS OF THIS TRIP LINK
UOCC	KMTL/I2	CURRENT NUMBER OF PASSENGERS IN THE TRIP LINK

Table 3-1. Global Variables -- SCMV (Page 31 of 58)

```
SCMV:       VEHICLE DATA - MODEL PROCESSOR
------------------------------------------------------------------------
VAR NAME     DIM      DESCRIPTION
----------- ------   ---------------------------------------------------
VNXSTN       KMV/I2   NEXT STOP OF THE VEHICLE - A STATION NUMBER

VDIVST       KMV/I2   INDICATES IF THE VEHICLE IS TO DIVERT TO
                        STORAGE AT THE NEXT AVAILABLE OPPORTUNITY
                          0===>DO NOT DIVERT TO STORAGE
                          1===>DIVERT TO STORAGE

VSINK        KMV/I2   SINK THROUGH WHICH THE VEHICLE IS TO EXIT
                        THE MODEL
                          1===>ON GUIDEWAY
                          2===>MODAL EXIT BEFORE PROCESSING
                          3===>MODAL EXIT AFTER PROCESSING

VROUTE       KMV/I2   FOR SCHEDULED:  THE NUMBER OF THE
                        SCHEDULED ROUTE TO WHICH THE VEHICLE
                        HAS BEEN ASSIGNED (POINTER TO VRPTR)

VNPASS       KMV/I2   CURRENT OCCUPANCY OF THE VEHICLE - NUMBER
                         OF PASSENGERS ON BOARD

VTOTP        KMV/I2   THE NUMBER OF PASSENGERS TO BOARD OR
                         DEBOARD THE VEHICLE

VTRLEN       KMV/I2   NUMBER OF VEHICLES IN THE TRAIN
                        -SET FOR ALL VEHICLES IN TRAIN BY IP;
                           USE FOR HEAD VEHICLE
                        -COUNT INCLUDES HEAD VEHICLE
                        -SET TO 1 IF NO TRAIN

VMEMCH       KMV/I2   USED TO CHAIN ALL VEHICLES ON AN STATION LINK IN THE
                        ORDER THEY ARRIVED (TO FORM A MEMBERSHIP
                        CHAIN)

VTRNCH       KMV/I2   CHAIN WORD FOR MAINTAINING VEHICLE
                        ENTRAINMENT
                          =0===>NOT ENTRAINED
                          >0===>IS ENTRAINED
                          (LAST VEH IN TRAIN POINTS TO HEAD VEH)

VBEVCH       KMV/I2   USED TO CHAIN ALL VEHICLES IN THE BOARD
                        EVENT
                        ALSO USED TO SUPPORT THE COLLECTION OF
                          STATISTICS REGARDING THE FEL/QUEUED
                          ORIGIN OF THE VEHICLE
                          =KMX+I===> V IS A FOLLOWER VEHICLE
                                     IN A TRAIN LEAD BY A
```

```
                                         VEHICLE WHICH CAME OFF THE FEL
                            =KMX+2===> THE FOLLOWER VEHICLES IN THE
                                         TRAIN LEAD BY V HAD BEEN
                                         QUEUED WHILE V HAD COME OFF
                                         THE FEL

VARRT         KMV/I4    ARRIVAL TIME OF THE VEHICLE IN THE MODELLED
                          AREA

VQREAS        KMV/I2    1===>VEHICLE AT HEAD OF STATION LINK & QUEUED DUE
                               TO:
                                 (A) CONGESTION
                                 (B) EXIT THIS LINK FAILED; OR
                                 (C) ENTRY NEXT LINK FAILED
                        2===>VEHICLE QUEUED DUE TO OTHER VEH IN FRONT
                               & OTHERWISE 'DONE'
                        3===>VEHICLE QUEUED DUE TO OTHER VEH IN FRONT
                               & WAITING TO START LAUNCH EVENT
                        4===>VEHICLE QUEUED IN STORAGE
                        0===>NONE OF THE ABOVE <===> ON FEL

VTRIPQ        KMV/I2    VEHICLE'S TRIP QUEUE - WHERE TRIPS RESIDE
                          WHILE ON THE VEHICLE; POINTER TO CHAIN
                          OF TRIPS; 0 WHEN THERE ARE NO TRIPS ON
                          VEHICLE

VBLTL         KMV/I2    POINTER TO TAIL OF CHAIN OF TRIP THAT ARE
                          ABOUT TO BOARD VEHICLE

VDBLTL        KMV/I2    POINTER TO TAIL OF CHAIN OF TRIP THAT ARE
                          ABOUT TO DEBOARD VEHICLE AND LEAVE STATION

VDXLTL        KMV/I2    POINTER TO TAIL OF CHAIN OF TRIP THAT ARE
                          ABOUT TO DEBOARD VEHICLE AND TRANSFER

VLAGAN        KMV/L1    TO INDICATE THAT THE L2 EVENT HAS TO BE
                          PERFORMED AGAIN SINCE IT WAS NOT POSSIBLE
                          TO FIND AN ADEQUATE OPENING ON THE BYPASS
                          LINK
                          T===>DO AGAIN
                          F===>DO NOT REPEAT

VRES          KMV/L1    INDICATES WHETHER OR NOT THE VEHICLE IS
                          RESERVED
                          T===>RESERVED
                          F===>NOT RESERVED
```

Table 3-1. Global Variables -- SCMXTN (Page 33 of 58)

SCMXTN: XTN HEADER DATA - MODEL PROCESSOR

VAR NAME	DIM/TYPE	DESCRIPTION

XTIME/
VTIME/
TTIME
	KMX/I4	IF VTIME > CLOCK, THEN THIS IS THE TIME AT WHICH THE VEHICLE/TRIP/TRANSACTION IS TO COME OFF THE FEL;
		IF VTIME < CLOCK, THEN THIS IS THE TIME AT WHICH THE VEHICLE/TRIP/TRANSACTION CAME OFF THE FEL AND WAS PUT IN A QUEUE

XSEVNT/ VSEVNT/ TSEVNT	KMX/I2	SYSTEM EVENT - WHERE TO GO IN MAIN ROUTINE WHEN COME OFF FEL --- TO BE DISTINGUISHED FROM VMEVNT/TMEVNT WHCH TELL WHERE TO GO IN SSMOD/SUMOD
XMEVNT VMEVNT TMEVNT	KMX/I2	TRANSACTION/VEH/TRIP EVENT - WHERE TO GO IN ASYNCH/ SSMOD/SUMOD (CURRENT STATION LINK/TRIP LINK EVENT) STATION LINK EVENTS PROCESSED BY SSMOD ARE: 1===>HEADWAY ZONE 2===>TRAVEL 3===>DEBOARD 4===>BOARD 5===>JOINT 6===>STORE 7===>LAUNCH 0===>END TRIP LINK EVENTS PROCESSED BY SUMOD ARE: 1===>WALK 2===>PROCESSING ASYNCHRONOUS EVENTS PROCESSED BY SAASYN ARE: 1===>DATA 2===>PARAM 3===>OPTION 4===>SELECT 5===>FAIL 6===>FLAG 7===>TEXT 8===>CKPT 9===>EOD 10===>STOP 11===>TRIP 12===>VEH
XFELCH/ VFELCH/ TFELCH/	KMX/I2	CHAIN WORD TO PUT TRANSACTION/VEHICLE/TRIP IN FEL OR CHAIN WORD USED TO CHAIN TRANSACTIONS/VEHICLES/ TRIPS INTO A QUEUE WHEN NOT ON THE FEL.

```
XQUECH/
VQUECH/
TQUECH                   TQUECH===>TRIPS ON A VEHICLE, TRIPS IN BOARDING
                           QUEUE

XEXTR1/        KMX/I2  EXTRA HALFWORD FOR MISCELLANEOUS DATA
                         IMMEDIATELY WHEN TRANSACTION COMES OFF FEL;
VCURR/                   THE NUMBER OF THE STATION BEING SIMULATED;
TCURR                    THE TRIP LINK ON WHICH THE TRIP IS CURRENTLY
                           LOCATED:
                             0===>JUST ARRIVING
                             1===>TICKETING LINK
                             2===>TURNSTILE LINK
                             3===>BOARDING QUEUE
                             4===>AT END OF TRIP LINK EVENTS.

XEXTR2/        KMX/I2  EXTRA HALFWORD FOR MISCELLANEOUS DATA
                         IMMEDIATELY WHEN TRANSACTION COMES OFF FEL;
  VSL                    THE STATION LINK ON WHICH THE VEHICLE IS
                           CURRENTLY LOCATED

    ***NOTE: FOR THE ABOVE 6 DATA ITEMS:
             SINCE VEH, TRIPS, AND TRANSACTIONS CAN BE PLACED ON THE FEL,
             THEY REQUIRE UNIQUE ID NUMBERS; THAT IS, VEH ID # 1,
             TRIP ID #1 AND TRANSACTION ID #1 CANNOT EXIST SIMULTANEOUSLY
             SINCE ON THE FEL THERE WOULD BE NO WAY OF
             DIFFERENTIATING BETWEEN THEM.  THEREFORE, TRANSACTION ID,
             REGARDLESS OF TYPE MUST BE SEQUENTIAL:
               VEHICLES:  1          THRU KMV
               TRIPS:     KMV+1 THRU KMV+KMT
               TRANSACTIONS: KMV+KMT+1 THRU KMX (SYSTEM SERVICE TRANSACT-
                             IONS)
             EQUIVALENCE BETWEEN NAMES ALLOWS VEHICLE TRANSACTIONS TO
             BE INDEXED INTO BY VEHICLE NUMBER (1---KMV) AND TRIP
             TRANSACTIONS TO BE INDEXED INTO BY TRIP NUMBER (1---KMT)
             IN THE MODEL CODE WHILE BEING REFERRED TO BY UNIQUE
             TXN NUMBER (1---KMX) IN THE CODE THAT PUTS THEM ON
             AND TAKES THEM OFF THE FEL.
             THE EQUIVALENCE RELATIONSHIPS ARE AS FOLLOWS:
               TTIME(1) =VTIME(KMV+1) =XTIME(KMV+1)
               TSEVNT(1)=VSEVNT(KMV+1)=XSEVNT(KMV+1)
               TMEVNT(1)=VMEVNT(KMV+1)=XMEVNT(KMV+1)
               TFELCH(1)=VFELCH(KMV+1)=XFELCH(KMV+1)
             XEXTR1 AND XEXTR2 ARE EQUIVALENCED TO TRIP AND
             VEHICLE DATA

             THE AVAILABLE CHAIN OF SYSTEM SERVICE TRANSACTIONS
             INCLUDES TRANSACTIONS IN THE FOLLOWING RANGES
             OF TRANSACTION ID (1---KMX):
               KMV+KMT+1 --------- KMX
```

```
KNV+1 -------------- KMV (UNUSED VEHICLE TRANSACTIONS)
KMV+KNT+1 --------- KMV+KMT (UNUSED TRIP TRANSACTIONS)
```

XAVAIL	-/I2	POINTER TO THE AVAILABLE CHAIN OF SYSTEM SERVICE TRANSACTIONS
VAVAIL	-/I2	POINTER TO THE AVAILABLE CHAIN OF VEHICLE TRANSACTIONS
TAVAIL	-/I2	POINTER TO THE AVAILABLE CHAIN OF TRIP TRANSACTIONS
XACTIV/ VACTIV/ TACTIV	-/I4	THE ID OF THE CURRENT TRANSACTION BEING PROCESSED, THE ONE THAT HAS MOST RECENTLY COME OFF THE FEL. XACTIV/VACTIV/TACTIV ARE EQUIVALENCED SINCE THERE IS ONLY ONE ACTIVE TRANSACTION AND SINCE IT IS OFTEN CONVENIENT TO REFER TO IT AS A VEHICLE OR TRIP.

Table 3-1. Global Variables -- SCNMAX (Page 36 of 58)

NAME: SCNMAX CATEGORY: INPUT PROCESSOR RUNTIME LIMITS

VARIABLE	DIM	TYPE	DESCRIPTION
KNIAT	–	I*2	NUMBER OF ENTRIES IN TRIP INTER-ARRIVAL TIME DISTRIBUTION (,1,KMIAT)
KNIAV	–	I*2	NUMBER OF ENTRIES IN VEHICLE INTER-ARRIVAL TIME DISTRIBUTION (,1,KMIAT)
KNNP	–	I*2	MAXIMUM NUMBER OF PASSENGERS/TRIP (,1,KMNP)
KNNT	–	I*2	MAXIMUM NUMBER OF TRIPS/VEHICLE (,0,KMNT)
KNS	–	I*2	MAXIMUM NUMBER OF DESTINATION STATIONS (,1,KMS)
KNTLEN	–	I*2	MAXIMUM TRAIN LENGTH IN VEHICLES (,1,KMTLEN)

AGT COMMON AREA DEFINITION

NAME: SCNSYS   CATEGORY: INPUT PROCESSOR SIMULATION SYSTEM CHARACTERISTICS

VARIABLE	DIM	TYPE	DESCRIPTION
ABCD	31	I*2	CONTENTS OF COLS 11-72 OF DATA HEADER CARD
ADATE	7	L*1	DATE OF CURRENT RUN FROM INDEX INPUT
AEOD	-	L*1	IF ON, END OF INPUT DATA HAS BEEN FOUND
AEOF	-	L*1	IF ON, END OF FILE FOUND ON INPUT DATA SET
AHDR	-	L*1	IF ON, DATA HEADER CARD HAS BEEN PROCESSED
AINDEX	-	L*1	IF ON, INDEX DATA HAS BEEN READ
AMNAME	6,8	L*1	NAMES OF MEMBERS WHICH MAY BE WRITTEN TO OUTPUT FILES BY IP. SUPPLIED BY USER IN PARM FIELD
ARANDN	-	R*4	RANDOM NUMBER BETWEEN 0-1
ARNTIM	-	L*1	IF ON, RUNTIME DATA HAS BEEN WRITTEN
ARNTMI	-	L*1	IF ON, RUNTIME DATA HAS BEEN READ
ASYSI	-	L*1	IF ON, SYSTEM DATA HAS BEEN READ
ASCHAR	-	L*1	IF ON, SYSTEM CHARACTERISTICS HAVE BEEN READ
ASETUP	-	L*1	IF ON, MODEL SETUP HAS BEEN REQUESTED BY USER
ATDGEN	-	L*1	IF ON, TRIP GENERATION HAS BEEN REQUESTED BY USER
ATEXT	18	I*2	DATA CARD IMAGE
ATIME	-	R*4	SIMULATED TIME ASSOCIATED WITH INPUT DATA CURRENTLY BEING PROCESSED (SECS)
ATIMG	-	R*4	VALUE OF TIME FILLD ON DATA HEADER CARD. SIMULATED TIME QT WHICH INPUT DATA IS TO BE READ (SECS)
ATITLE	12	I*4	TITLE OF CURRENT RUN FROM INDEX DATA
ATYPE	-	I*4	FIRST FOUR CHARACTERS OF DATA TYPE ON HEADER CARD
AVDGEN	-	L*1	IF ON, VEHICLE GENERATION REQUESTED BY USER
AVTYPE	-	L*1	SOURCE OF VEHICLE DEMAND (G=GUIDEWAY, A=MODAL INPUT BEFORE PROCESSING,

Table 3-1. Global Variables -- SCNSYS (Page 38 of 58)

			B=MODAL INPUT AFTER PROCESSING)
			SUPPLIED BY USER IN PARM FIELD FOR INDEX
AUSER	7	I*2	USER IDENTIFICATION FOR INDEX FILE

Table 3-1. Global Variables -- SCNTDM (Page 39 of 58)

NAME: SCNTDM CATEGORY: TRIP DEMAND GENERATION (INPUT)

VARIABLE	DIM	TYPE	DESCRIPTION
DRANDN	-	R*4	RANDOM NUMBER BETWEEN 0-1
DTARIV	-	R*4	ARRIVAL TIME OF TRIP (,0,DTENDT)
DTDESD	KMS	R*4	PROBABILITY OF A TRIP ENDING AT EACH OF OTHER STATIONS (,0,1) SUM OF DISTRIBUTION MUST = 1.0
DTDEST	-	I*2	DESTINATION OF TRIP (,1,KNS)
DTENDT	-	I*4	SIMULATED TIME AT WHICH TRIP GEN. IS TO END (SECS) (,0+,LVENDG)
DTIASW	-	L*1	SPECIFIES WHETHER EXPONENTIAL(0) OR USER (1) IAT DISTRIBUTION IS TO BE USED (,0,1)
DTIATD	KMIAT, 2	R*4	USER'S IAT DISTRIBUTION. COL1 = PROBABILITY, COL2=TIME SUM OF DISTRIBUTION MUST = 1.0 COL1 : (,0,1) COL2 : (,0,LTIATH)
DTLMDA	-	R*4	MEAN ARRIVAL RATE (TRIPS/TIME) (,0,LTIATH)
DTPASD	KMNP	R*4	PROBABILITY OF A TRIP HAVING 1,2,,, KMNP PASSENGERS (,0,1) SUM OF DISTRIBUTION MUST = 1.0
DTPASS	-	I*2	NO. PASSENGERS IN A TRIP (,1,KNNP)

Table 3-1. Global Variables -- SCNTDM (Page 40 of 58)

DTSTAN	—	I*2	STATION BEING SIMULATED (,1,KNS)
DTSTAT	2,KMS	I*2	TRIP STATISTICS BY DESTINATION ROW1=NO. TRIPS, ROW2=NO. PASSENGERS
TKSEED	—	I*4	RANDOM NO. SEED (TKSEED ODD INTEGER >= 3)

Table 3-1. Global Variables -- SCNVDM (Page 41 of 58)

NAME:SCNVDM CATEGORY:VEHICLE DEMAND GENERATION (INPUT)

VARIABLE	DIM	TYPE	DESCRIPTION
DRANDO	-	R*4	RANDOM NUMBER BETWEEN 0-1
DVARIV	-	R*4	CURRENT ARRIVAL TIME (,C+,DVENDT)
DVARVR	KMR	R*4	FOR SCHED. ENV., ARRIVAL TIME OF NEXT VEHICLE ON ROUTE (,0+,DVENDT)
DVCAPP	-	I*2	VEHICLE CAPACITY IN PASSENGERS (,1,)
DVDESD	KMS	R*4	PROBABILITY OF TRIP ENDING AT EACH OF THE OTHER STATIONS (,1,KMS) SUM OF DISTRIBUTION MUST = 1.0
DVDEST	-	I*2	DESTINATION OF TRIP BEING GENERATED (,1,KMS)
DVENDT	-	I*4	TIME AT WHICH DEMAND GENERATION IS TO STOP (SECS) (,0+,LVENDT)
DVHDWY	-	I*4	MINUMUM HEADWAY BETWEEN CURRENT AND PREVIOUS TRAINS/VEHICLES (,0,)
DVIASW	-	I*2	INDICATOR SPECIFYING WHETHER EXP(0) OR USER(1) IAT DIST. SHOULD BE USED (,0,1)
DVIATD	KMIAT, 2	R*4	USER'S INTERARRIVAL TIME DIST. COL1=PROBABILITY, COL2=TIME SUM OF DISTRIBUTION MUST = 1.0) COL1 : (,0,1) COL2 : (,0+,DVENDT)

Table 3-1. Global Variables -- SCNVDM (Page 42 of 58)

DVIATM	–	R*4	CURRENT INTERARRIVAL TIME (,0+,DVENDT)
DVLMDA	KMR	R*4	MEAN ARRIVAL RATE(TRANS/TIME) BY RTE (,0+,LTIATH)
DVNXID	–	I*2	INDICATOR SPECIFYING NEXT STOP SELEC- TION METHOD DVRTST(0),DVRSCH(1) (FOR SCHEDULED SERVICE) (,0,1)
DVNXST	–	I*2	NEXT STOP FOR VEHICLE BEING GENERATED (,0,KNS)
DVPASD	KMNP	R*4	PROBABILITY OF TRIP HAVING 1,2,,,PASS (,0,1) SUM OF DISTRIBUTION MUST = 1.0
DVPASS	–	I*2	NO. OF PASSENGERS IN TRIP (,1,KNNP)
DVPMIN	–	R*4	PROBABILITY OF VEHICLE ENTERING AT MINIMUM HEADWAY (,0,1)
DVROUT	–	I*2	ROUTE OF VEHICLE BEING GENERATED (,1,KNR)
DVRPTR	KMR	I*2	POINTER TO STARTING ENTRY FOR EACH ROUTE
DVRSCH	KMRT	I*2	CONCATENATION OF ALL SCHEDULED ROUTES. FIRST ENTRY TO EACH IS POINTED TO BY DVRPTR (,1,KMS)
DVRTST	KMR	I*2	FOR EACH ROUTE AN INDICATOR SPECIFY- ING WHETHER DVSTAN IS ON THE ROUTE(1) OR NOT(0) (,0,1)
DVSERV	–	I*2	SERVICE POLICY IN USE - 1=DEMAND RESPONSIVE SINGLE PARTY 2= DEMAND RESPONSIVE MULTI PARTY 3= SCHEDULED SERVICE (,1,3)
DVSINK	–	I*2	SINK (,1,3)

Table 3-1. Global Variables -- SCNVDM (Page 43 of 58)

DVSLOT	–	I*2	LENGTH OF SLOT IN TIME (,0+,LVSLOT)
DVSNKD	3	R*4	PROB. OF VEHICLE LEAVING AT 1 OF 3 SINKS (,0,1) SUM OF DISTRIBUTION MUST = 1
DVSNSW	–	I*2	INDICATOR SPECIFYING WHETHER VEHICLES ARE BEING GEN'D FOR SYNC(1) ENV OR NOT(0) (,0,1)
DVSTAN	–	I*2	STATION BEING SIMULATED (,1,KMS)
DVSTAT	6,KMR	I*4	VEHICLE GENERATION SUMMARY DATA ROWS1-4 =TRAINS,VEHICLES,TRIPS,PASS STOPPING ROWS 5-6=TRAINS, VEHICLES NOT STOPPING
DVSTPP	–	R*4	FOR DEMAND RESP. ENF., PROB. OF STOPPING AT THE STATION BEING SIM'D (,0,1)
DVTLND	KMTLEN	R*4	PROBABILITY OF HAVING 1,2,,,VEHICLES IN TRAIN (,0,1) SUM OF DISTRIBUTION MUST = 1.0
DVTLNR	KMR	I*2	LENGTH OF TRAINS ON EACH ROUTE (,1,KNTLEN)
DVTRIP	2 KMNT	I*2	TRIP FOLLOWER RECORDS FOR A VEHICLE-COLS ARE DEST AND NO. PASSENGERS
DVTRLN	–	I*2	TRAIN LENGTH IN VEHICLES (,1,KNTLEN)
DVTRPD	KMNT	R*4	PROB OF HAVING 0,1,2,,,TRIPS/VEHICLE (,0,1) SUM OF DISTRIBUTION MUST = 1.0
DVVPAS	–	I*2	NO. PASSENGERS IN VEHICLE (,0,DVCAPP)
DVVTRP	–	I*2	NO. TRIPS IN VEHICLE (,0,KNNT)

Table 3-1. Global Variables -- SCNVDM (Page 44 of 58)

VKSEED	–	I*4	RANDOM NUMBER SEED (VKSEED >= 3 ODD INTEGER)

Table 3-1. Global Variables -- SCZ (Page 45 of 58)

VAR NAME	DIM	DESCRIPTION
KMNST		NUMBER OF STATION STATES (DEFINED IN SMAXSIZE)
KMSST		NUMBER OF STATION LINK STATES(")
KMTST		NUMBER OF TRIP LINK STATES (")
---STATISTICS ON VEHICLES IN STATION STATES		
ZNVNE	KMNST /I2	NUMBER OF VEHICLES ENTERING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNVNL	KMNST /I2	NUMBER OF VEHICLES LEAVING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNVNI	KMNST /I2	NUMBER OF VEHICLES IN STATE I OF THE STATION AT THE END OF THE LAST SAMPLING INTERVAL
ZNVMNI	KMNST /I2	MAXIMUM NUMBER OF VEHICLES IN STATE I OF STATION DURING THE LAST SAMPLING INTERVAL
ZNVTIN	KMNST /I4	INTEGRAL OF VEHICLE-TIME IN STATE I IN STATION DURING THE LAST SAMPLING INTERVAL
ZNVSTL	KMNST /I4	SUM OF TIMES IN STATE I OF VEHICLES LEAVING DURING THE LAST SAMPLING INTERVAL
ZNVMTL	KMNST /I4	MAXIMUM TIME IN STATE I OF VEHICLES LEAVING DURING THE LAST SAMPLING INTERVAL
ZNVANI	KMNST /R4	AVERAGE NUMBER OF VEHICLES IN STATE I DURING THE LAST SAMPLING INTERVAL
ZNVATL	KMNST /R4	AVERAGE TIME IN STATE I OF VEHICLES LEAVING DURING THE LAST SAMPLING INTERVAL
---STATISTICS ON TRIPS IN STATION STATES		
ZNTNE	KMNST /I2	NUMBER OF TRIPS ENTERING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNTNL	KMNST /I2	NUMBER OF TRIPS LEAVING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNTNI	KMNST /I2	NUMBER OF TRIPS IN STATE I OF THE STATION AT THE END OF THE LAST SAMPLING INTERVAL
ZNTMNI	KMNST /I2	MAXIMUM NUMBER OF TRIPS IN STATE I OF STATION DURING THE LAST SAMPLING INTERVAL
ZNTTIN	KMNST /I4	INTEGRAL OF TRIP-TIME IN STATE I IN STATION DURING THE LAST SAMPLING INTERVAL
ZNTSTL	KMNST /I4	SUM OF TIMES IN STATE I OF TRIPS LEAVING DURING THE LAST SAMPLING INTERVAL
ZNTMTL	KMNST /I4	MAXIMUM TIME IN STATE I OF TRIPS LEAVING DURING THE LAST SAMPLING INTERVAL
ZNTANI	KMNST /R4	AVERAGE NUMBER OF TRIPS IN STATE I DURING THE LAST SAMPLING INTERVAL
ZNTATL	KMNST /R4	AVERAGE TIME IN STATE I OF TRIPS LEAVING DURING THE LAST SAMPLING INTERVAL
---STATISTICS ON PASSENGERS IN STATION STATES		
ZNPNE	KMNST /I2	NUMBER OF PASS. ENTERING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL

Table 3-1. Global Variables -- SCZ (Page 46 of 58)

```
ZNPNL          KMNST    NUMBER OF PASS. LEAVING STATE I OF THE STATION
                /I2       DURING THE LAST SAMPLING INTERVAL
ZNPNI          KMNST    NUMBER OF PASS. IN STATE I OF THE STATION
                /I2       AT THE END OF THE LAST SAMPLING INTERVAL
ZNPMNI         KMNST    MAXIMUM NUMBER OF PASS. IN STATE I OF STATION
                /I2       DURING THE LAST SAMPLING INTERVAL
ZNPTIN         KMNST    INTEGRAL OF PASS.-TIME IN STATE I IN STATION
                /I4       DURING THE LAST SAMPLING INTERVAL
ZNPSTL         KMNST    SUM OF TIMES IN STATE I OF PASS. LEAVING
                /I4       DURING THE LAST SAMPLING INTERVAL
ZNPMTL         KMNST    MAXIMUM TIME IN STATE I OF PASS. LEAVING
                /I4       DURING THE LAST SAMPLING INTERVAL
ZNPANI         KMNST    AVERAGE NUMBER OF PASS. IN STATE I
                /R4       DURING THE LAST SAMPLING INTERVAL
ZNPATL         KMNST    AVERAGE TIME IN STATE I OF PASS. LEAVING
                /R4       DURING THE LAST SAMPLING INTERVAL

---STATISTICS ON VEHICLES IN STATION LINK (SL) STATES
ZSVNE          KMSST    NUMBER OF VEHICLES ENTERING STATE I OF SL J
                KMSL/I2  DURING THE LAST SAMPLING INTERVAL
ZSVNL          KMSST    NUMBER OF VEHICLES LEAVING STATE I OF SL J
                KMSL/I2  DURING THE LAST SAMPLING INTERVAL
ZSVNI          KMSST    NUMBER OF VEHICLES IN STATE I OF SL J
                KMSL/I2  AT THE END OF THE LAST SAMPLING INTERVAL
ZSVMNI         KMSST    MAXIMUM NUMBER OF VEHICLES IN STATE I ON SL J
                KMSL/I2  DURING THE LAST SAMPLING INTERVAL
ZSVTIN         KMSST    INTEGRAL OF VEHICLE-TIME IN STATE I ON SL J
                KMSL/I4  DURING THE LAST SAMPLING INTERVAL
ZSVSTL         KMSST    SUM OF TIMES OF VEHICLES LEAVING STATE I ON SL J
                KMSL/I4  DURING THE LAST SAMPLING INTERVAL
ZSVMTL         KMSST    MAXIMUM TIME OF VEHICLES LEAVING STATE I ON SL J
                KMSL/I4  DURING THE LAST SAMPLING INTERVAL
ZSVANI         KMSST    AVERAGE NUMBER OF VEHICLES IN STATE I ON SL J
                KMSL/R4  DURING THE LAST SAMPLING INTERVAL
ZSVATL         KMSST    AVERAGE TIME OF VEHICLES LEAVING STATE I ON SL J
                KMSL/R4  DURING THE LAST SAMPLING INTERVAL

---STATISTICS ON TRIPS IN TRIP LINK (TL) STATES
ZTTNE          KMTST    NUMBER OF TRIPS ENTERING STATE I OF TL J
                KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTTNL          KMTST    NUMBER OF TRIPS LEAVING STATE I OF TL J
                KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTTNI          KMTST    NUMBER OF TRIPS IN STATE I OF TL J
                KMTL/I2  AT THE END OF THE LAST SAMPLING INTERVAL
ZTTMNI         KMTST    MAXIMUM NUMBER OF TRIPS IN STATE I ON TL J
                KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTTTIN         KMTST    INTEGRAL OF TRIP-TIME IN STATE I ON TL J
                KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTTSTL         KMTST    SUM OF TIMES OF TRIPS LEAVING STATE I ON TL J
                KMTL/I4  DURING THE LAST SAMPLING INTERVAL
```

Table 3-1. Global Variables -- SCZ (Page 47 of 58)

```
ZTTMTL        KMTST    MAXIMUM TIME OF TRIPS LEAVING STATE I ON TL J
               KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTTANI        KMTST    AVERAGE NUMBER OF TRIPS IN STATE I ON TL J
               KMTL/R4  DURING THE LAST SAMPLING INTERVAL
ZTTATL        KMTST    AVERAGE TIME OF TRIPS LEAVING STATE I ON TL J
               KMTL/R4  DURING THE LAST SAMPLING INTERVAL

---STATISTICS ON PASS. IN TRIP LINK (TL) STATES
ZTPNE         KMTST    NUMBER OF PASS. ENTERING STATE I OF TL J
               KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTPNL         KMTST    NUMBER OF PASS. LEAVING STATE I OF TL J
               KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTPNI         KMTST    NUMBER OF PASS. IN STATE I OF TL J
               KMTL/I2  AT THE END OF THE LAST SAMPLING INTERVAL
ZTPMNI        KMTST    MAXIMUM NUMBER OF PASS. IN STATE I ON TL J
               KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTPTIN        KMTST    INTEGRAL OF PASS.-TIME IN STATE I ON TL J
               KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTPSTL        KMTST    SUM OF TIMES OF PASS. LEAVING STATE I ON TL J
               KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTPMTL        KMTST    MAXIMUM TIME OF PASS. LEAVING STATE I ON TL J
               KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTPANI        KMTST    AVERAGE NUMBER OF PASS. IN STATE I ON TL J
               KMTL/R4  DURING THE LAST SAMPLING INTERVAL
ZTPATL        KMTST    AVERAGE TIME OF PASS. LEAVING STATE I ON TL J
               KMTL/R4  DURING THE LAST SAMPLING INTERVAL

---THE FOLLOWING STATISTICS DO NOT FIT INTO THE ABOVE SCHEME
    AND ARE REFERRED TO AS MISCELLANEOUS

ZM            280/R4   MISCELLANEOUS STATISTICS; THE FIRST
                         218 OF THESE ARE USED TO GENERATE THE
                         PERFORMANCE SUMMARY FILE BY THE OP
 SUBSCRIPT
           1 VEHICLE CAPACITY(=VCAP)
           2 AVERAGE VEHICLE LOAD ENTERING STN FROM GUIDEWAY
           3 AVERAGE VEHICLE LOAD ENTERING STN FROM MODAL INPUT BEFORE
           4 AVERAGE VEHICLE LOAD ENTERING STN FROM MODAL INPUT AFTER
           5 AVERAGE VEHICLE LOAD LEAVING STN FROM GUIDEWAY
           6 AVERAGE VEHICLE LOAD LEAVING STN FROM MODAL INPUT BEFORE
           7 AVERAGE VEHICLE LOAD LEAVING STN FROM MODAL INPUT AFTER
           8 NUMBER OF VEHICLE ENTERING STN FROM GUIDEWAY
           9 NUMBER OF VEHICLE ENTERING STN FROM MODAL INPUT BEFORE
          10 NUMBER OF VEHICLE ENTERING STN FROM MODAL INPUT AFTER
          11 NUMBER OF VEHICLE LEAVING STN FROM GUIDEWAY
          12 NUMBER OF VEHICLE LEAVING STN FROM MODAL INPUT BEFORE
          13 NUMBER OF VEHICLE LEAVING STN FROM MODAL INPUT AFTER
          14 NUMBER OF VEHICLES REJECTED AT INPUT RAMP
          15 NUMBER OF VEHICLES ACCEPTED AT INPUT RAMP
          16 NUMBER OF EMPTIES GOTTEN FROM LOCAL STORAGE
```

Table 3-1. Global Variables -- SCZ (Page 48 of 58)

```
     17 NUMBER OF EMPTIES GOTTEN FROM UPSTREAM SLS
     18 NUMBER OF EMPTIES GOTTEN FROM ELSEWHERE IN NET
     19 NUMBER OF TRIPS ARRIVING AT BOARD QUEUE
     20 NUMBER OF TRIPS BOARDING
     21 NUMBER OF TRIPS DEBOARDING TO LEAVE
     22 NUMBER OF TRIPS DEBOARDING TO TRANSFER
     23 NUMBER OF PASSENGERS ARRIVING AT BOARD QUEUE
     24 NUMBER OF PASSENGERS BOARDING
     25 NUMBER OF PASSENGERS DEBOARDING TO LEAVE
     26 NUMBER OF PASSENGERS DEBOARDING TO TRANSFER
                           SLTYPE      MEANING
                              1           IR
                              2           IQ
                              3           D (THE DEBOARD/BOARD/JOINT EVENTS
                                             CAN APPEAR ONLY ON THIS TYPE)
                              4           OQ
                              5           OR
                              6           S -
                              7           IS
                              8           SI
                              9           DS
                             10           SO
                             11           UL
                             12           BL
                             13           DL
                             14           MIB
                             15           MIA
                             16           MOB
                             17           MOA
                             18           UNUSED
  27- 44 FOR EACH 'SLTYPE' AVERAGE # OF VEHICLES IN SL OF THAT TYPE
  45- 62 FOR EACH 'SLTYPE' MAXIMUM # OF VEHICLES IN SL OF THAT TYPE
  63- 80 FOR EACH 'SLTYPE' AVERAGE TIME SPENT IN SL OF THAT TYPE
  81- 98 FOR EACH 'SLTYPE' MAXIMUM TIME SPENT IN SL OF THAT TYPE
  99-116 FOR EACH 'SLTYPE' AVERAGE # OF VEH IN SL QUEUE OF THAT TYPE
 117-134 FOR EACH 'SLTYPE' MAXIMUM # OF VEH IN SL QUEUE OF THAT TYPE
 135-152 FOR EACH 'SLTYPE' AVERAGE TIME SPENT IN SL QUEUE OF THAT TYPE
 153-170 FOR EACH 'SLTYPE' MAXIMUM TIME SPENT IN SL QUEUE OF THAT TYPE
 171-173 FOR EACH TL AVERAGE # OF TRIPS IN TL
 174-176 FOR EACH TL MAXIMUM # OF TRIPS IN TL
 177-179 FOR EACH TL AVERAGE TIME SPENT IN TL
 180-182 FOR EACH TL MAXIMUM TIME SPENT IN TL
 183-185 FOR EACH TL AVERAGE # OF TRIPS IN TL QUEUE
 186-188 FOR EACH TL MAXIMUM # OF TRIPS IN TL QUEUE
 189-191 FOR EACH TL AVERAGE TIME SPENT IN TL QUEUE
 192-194 FOR EACH TL MAXIMUM TIME SPENT IN TL QUEUE
 195-197 FOR EACH TL AVERAGE # OF PASSENGERS IN TL
 198-200 FOR EACH TL MAXIMUM # OF PASSENGERS IN TL
 201-203 FOR EACH TL AVERAGE TIME SPENT IN TL
 204-206 FOR EACH TL MAXIMUM TIME SPENT IN TL
```

```
207-209 FOR EACH TL AVERAGE # OF PASSENGERS IN TL QUEUE
210-212 FOR EACH TL MAXIMUM # OF PASSENGERS IN TL QUEUE
213-215 FOR EACH TL AVERAGE TIME SPENT IN TL QUEUE
216-218 FOR EACH TL MAXIMUM TIME SPENT IN TL QUEUE
    219 NUMBER OF TRIPS REJECTED AT TICKETING LINK
```

Table 3-1. Global Variables -- SMAXSIZE (Page 50 of 58)

COMPILE-TIME MAXIMA:
THE FOLLOWING VARIABLE NAMES DEFINE THE MAXIMUM NUMBER OF ENTITIES AVAILABLE WITHOUT RECOMPILING. ARRAYS ARE DIMENSIONED USING THESE VARIABLES. THEY ARE PREPROCESSOR VARIABLES AND ASSIGNED VALUES IN ONE CENTRALLY LOCATED MEMBER.

NAME	VALUE	DESCRIPTION - MAXIMUM NUMBER OF:
KMV	200	VEHICLES THERE CAN BE IN THE SIMULATOR SIMULTANEOUSLY
KMV1	201	KMV+1
KMT	1000	TRIPS THERE CAN BE IN THE SIMULATOR SIMULTANEOUSLY
KMX	1500	XTNS (=KMV+KMT+NO. OF SYSTEM SERVICE TRANSACTIONS)
KMCLTA	1000	ENTRIES IN CLOCK TABLE
KMMSGS	106	MESSAGES OF ANY KIND THAT CAN BE ISSUED BEFORE TERMINATION
KMMSGI	100	INFORMATION MESSAGES BEFORE TERMINATION
KMMSGW	6	WARNING MESSAGES BEFORE TERMINATION
KMMTYP	100	ANY ONE MESSAGE THAT CAN BE ISSUED PRIOR TO
KMSL	50	STATION LINKS
KMR	20	ROUTES
KMRT	100	ENTRIES IN SCHEDULED ROUTE LIST (PVRLST)
KMS	100	ENTRIES IN STATION ROUTE ASSIGNMENT TABLE (PRASGN)
KMEVP	10	ENTRIES IN USER'S ORDERED EMPTY VEHICLE PRIORITY LIST OF WHERE TO PUT EMPTIES (PVEPR) (=2)
KMSVP	10	ENTRIES IN USER'S ORDERED LIST OF WHERE TO SEARCH FOR EMPTIES (PVSPR) (=3)
KMNMD	50	ENTRIES IN NETWORK MERGE DELAY DISTRIBUTION (PNMDDT)
KMEVD	50	ENTRIES IN EMPTY VEHICLE DELAY DISTRIBUTION (PEVDDT)
KMSLE	200	ENTRIES IN EVENT LIST (SLEVL)
KMSLD	200	ENTRIES IN DOWNSTREAM SL LIST (SLDSL)

Table 3-1. Global Variables -- SMAXSIZE (Page 51 of 58)

KMSLU	200	ENTRIES IN UPSTREAM SL LIST (SLUSL)
KMTL	3	TRIP LINKS
KMFLAG	300	DBUG FLAGS = ENTRIES IN AFLAG TERMINATION
KMSLDS	20	ENTRIES IN THE ORDERED DOWNSTREAM SL LIST (ASLLST)
KMASYN	5	NUMBER OF THE INPUT UNIT FOR ASYNCHRONOUS DATA
KMTF	15	NUMBER OF THE INPUT UNIT FOR THE TRIP FILE
KMVF1	24	NUMBER OF THE INPUT UNIT FOR THE VEHICLE FILE ASSOCIATED WITH SOURCE 1. (THE UNIT NUMBER ASSOCIATED WITH SOURCE 2 IS ASSUMED TO BE KMNF1+1 AND THE UNIT NUMBER ASSOCIATED WITH SOURCE 3 IS ASSUMED TO BE KMNF1+2.)
KMRAW	6	NUMBER OF OUTPUT UNIT FOR RAW STATISTICS FILE
KMHDR	15	NUMBER OF TYPES OF HEADER CARDS(USED BY IP ONLY)
KMIAT	100	ENTRIES IN USER'S INTERARRIVAL TIME DISTRIBUTION
KMNP	100	ENTRIES IN NUMBER OF PASSENGER'S DISTRIBUTION
KMNT	100	ENTRIES IN NUMBER OF TRIP'S PER VEHICLE DISTRIBUTION
KMNST	4	NUMBER OF STATION-WIDE STATISTICS STATES
KMSST	3	NUMBER OF STATION LINK STATISTICS STATES
KMTST	3	NUMBER OF TRIP LINK STATISTICS STATES
KMTLEN	30	NUMBER OF VEHICLES IN A TRAIN
KMSLCAP	100	NUMBER OF VEHICLES ON ANY STATION LINK

Table 3-1. Global Variables -- SODCLS (Page 52 of 58)

SODCLS: DECLARE COMMON AREAS UNIQUE TO THE SOP

VAR NAME	DIM	TYPE	DESCRIPTION
(BLANK) COMMON			
A	40000	R4	BIN AREA
BASIC COMMON			
LOC	5	I4	CONTAINS POINTER TO BINS IN 'A' ARRAY
SYSCOM COMMON			
NLINES	-	I4	NUMBER OF LINES
KNSL	-	I4	NUMBER OF STATION LINKS
KNTL	-	I4	NUMBER OF TRIP LINKS
KNR	-	I4	UNUSED
KNB	-	I4	UNUSED
KNA	-	I4	UNUSED
CLOCK	-	I4	SAMPLE TIME
CSAMPL	-	I4	SAMPLE INTERVAL IN CU
CSIZE	-	I4	CU PER MINUTE
PERFS	-	L1	INDICATOR FOR PERFORMANCE SUMMARY DATA COLLECTION
SUB COMMON			
JN	-	I4	NUMBER OF BINS SET
ITOT	-	I4	NUMBER OF WORDS IN BIN AREA
KN	-	I4	NUMBER OF WORDS IN USE
JK	-	I4	UNUSED - INIT TO 0
AFLAG	99	L1	INTERMEDIATE OUTPUT CONTROL FLAGS
SYSCM1 COMMON			
NUMS	-	I4	UNUSED
NUML	-	I4	UNUSED
NUMSL	-	I4	NUMBER OF STATION LINKS
NUMR	-	I4	UNUSED
VCAP	-	I2	VEHICLE CAPACITY
STNNO	400	I2	UNUSED
KTYPE	2	I2	STATION LINK TYPE
OUTPT COMMON			
TITLES	1800	I4	OUTPUT STATISTIC TITLES. EACH TITLE IS COMPOSED OF 16 CHARACTERS ALLOCATED TO 4 FULLWORDS. THE ENTRIES IN THE TABLE ARE ORGANIZED SEQUENTIALLY BEGINNING WITH THE FIRST TITLE FOR THE FIRST VARIABLE IN A PARTICULAR MAJOR CATEGORY.

MAJC	400	I2	MAJOR CATEGORY OF REQUESTS
NCAT	120	I2	TABLE OF CUMULATIVE INDICIES FOR RETRIEVING 16 CHARACTER TITLES FOR OUTPUT STATISTICS. EACH VALUE IN THE TABLE CORRESPONDS TO A COLUMN IN MCOTAB AND CONTAINS THE CUMULATIVE COUNT OF THE NUMBER OF VARIABLES THAT PROCEEDED THAT CATEGORY TYPE.
PERFM COMMON			
PSUMM	400	R4	SUM OF PERFORMANCE SUMMARY VALUES OVER REPORT PER.
PMIN	400	R4	MIN OF PERFORMANCE SUMMARY VALUES OVER REPORT PER.
PMAX	400	R4	MAX OF PERFORMANCE SUMMARY VALUES OVER REPORT PER.
NSAMP	-	I4	COUNT OF NUMBER OF SAMPLES IN REPORT PERIOD.
FORMAT COMMON			
IFORMS	400	I2	OUTPUT FORMAT NUMBER FOR A PARTICULAR REQUEST
SOIDX COMMON			
AMNAME	4,8	L1	PARSED NAME FIELDS FROM PARM LIST
AMFLAG	4	L1	INDICATES WHICH FILES WERE USED
MISC			
ZCAP	5	I2	5 INPUT-LIKE VARIABLES TO BE PASSED TO PERFORMANCE SUMMARY

```
SODEFS: DECLARE COMMON AREAS CONTAINED IN SODCLS & ZODCLS WITH FULL
        DIMENSIONS FOR USE IN THE MAIN ROUTINE 'SOUTPT'.  SODCLS &
        ZODCLS ARE USED IN ALL OTHER ROUTINES AND CONTAIN SMALLER
        DUMMY DIMENSIONS IN THEIR DECLARATIONS.  THIS ALLOWS A USER
        TO CHANGE THE DIMENSIONS OF SOME OF THE MAJOR ARRAYS IN THE
        OP AND RECOMPILE ONLY SOUTPT AND NOT ALL THE OTHER ROUTINES
        THAT USE THE ARRAYS(SINCE IN THESE OTHER ROUTINES THESE
        ARRAYS ARE DECLARED WITH JUST DUMMY DIMENSIONS WHICH INDICATE
        TO THE FORTRAN COMPILER JUST THAT IT IS AN ARRAY (IT GETS THE
        DIMENSIONS FROM THE OTHER DECLARATION). THE DIMENSION TAKEN
        IS DECIDED BY HAVING SOUTPT FIRST IN THE LINK EDIT.
-----------------------------------------------------------------------
VAR NAME   DIM   TYPE DESCRIPTION
---------- ---   ---- -------------------------------------------------

THE ONLY DIFFERENT DIMENSIONS ARE:
ZREQUE     12,400     OUTPUT REQUEST STORAGE TABLE

LOC        400        BIN AREA LOCATION FOR BIN ASSIGNED TO SPECIFIC
                        REQUEST

KTYPE      KMSL       LINK TYPE DESIGNATIONS FOR SL'S

THE ONLY ADDITIONS (LOCAL TO SOUTPT) ARE:
KMSUB                 INIT=450

NAMEL      5     I4   CONTAINS NAMES OF THE VALID HEADER CARD TYPES
                        ACCEPTED BY DSM

FORMS      2,8   I4   CONTAINS REQUEST CARD KEYWORDS

WIDTHS     400   R4   HISTOGRAM CLASS INTERVAL FOR CORRESPONDING REQUEST

SYMBOL     4     I4   CONTAINS 4 PLOTTING SYMBOLS

COMMON NAMES
TSER       -     I4   CONTAINS 'TSER'

STATS      -     I4   CONTAINS 'STAT'

PLOT       -     I4   CONTAINS 'PLOT'

SYSTEM     -     I4   CONTAINS 'SYST'

TRIP       -     I4   CONTAINS 'TRIP'

LIST       -     I4   CONTAINS 'LIST'

HIST       -     I4   CONTAINS 'HIST'

STN        -     I4   CONTAINS 'STN '
```

Table 3-1. Global Variables -- ZCAMSG (Page 55 of 58)

ZCAMSG: ERROR MESSAGE DATA

VAR NAME	DIM	DESCRIPTION
KNMSG	3/I2	NUMBER OF MESSAGES ISSUED DURING A RUN, BY CLASS: 1 = INFORMATION 2 = WARNING 3 = SEVERE
NMSGS	-/I2	TOTAL NUMBER OF MESSAGES OF ANY CLASS ISSUED DURING A RUN
MSGC	KMMSGS /I2	MESSAGE NUMBERS ISSUED DURING RUN
MSGCN	KMMSGS /I2	NUMBER OF REMAINING MESSAGES OF THIS TYPE ALLOWED PRIOR TO TERMINATION
TERM	-/L1	INDICATOR TO SIGNAL TERMINATION DUE TO EXCESSIVE MESSAGES
MFLAG	-/L1	ERROR PROCESSING IN PROGRESS INDICATOR TO HALT RECURSIVE ERROR PROCESSING
MSGID	-/L1	ID OF MODEL BEING EXECUTED (1=DSM OUTPUT PROCESSOR)

Table 3-1. Global Variables -- ZODCLS (Page 56 of 58)

ZODCLS: DECLARE COMMON AREAS COMMON TO ALL OP

VAR NAME	DIM	TYPE	DESCRIPTION
REQUES COMMON			
ZREQUE	12,2	I4	OUTPUT PROCESSOR REQUEST TABLE
IENDS COMMON			
IEND	-	I4	NUMBER OF REQUESTS ENTERED
READER COMMON			
MSTART	-	I4	START OF REPORT PERIOD IN CU
MSTOP	-	I4	END OF REPORT PERIOD IN CU
MSOT	-	I4	RAW STATISTICS FILE UNIT NUMBER
MSOTX	-	I4	RAW STATISTICS FILE UNIT NUMBER
MNAME	-	R8	ALPHA DESIGNATION OF RECORD TYPE(HEADER,FOLLOWER)
CUSEC	-	R4	CU PER SEC
MCLOCK	-	I4	TIME VALUE OF CURRENT RAW STATISTICS RECORD BEING PROCESSED
EOF	-	I4	END OF FILE ON RAW STATISTICS FILE
MBYTES	-	I2	NUMBER OF BYTES IN FOLLOWER RECORD
MFOLL	-	I2	NUMBER OF FOLLOWER RECORDS
MTYPE	-	I2	TYPE OF FOLLOWER RECORDS
TABLES COMMON			
MAINTA	120	I4	MAIN CATEGORIES TABLE. IS INDEXED BY TYPE NUMBER. ENTRIES FOR MAIN CATEGORIES ARE BCD. A ZERO ENTRY IMPLIES NO MAIN CATEGORY HAS BEEN DEFINED.
MCOTAB	120	I2	TABLE OF SUBCATEGORIES. EACH COLUMN CONTAINS ALL OF THE SUBCATEGORIES CORRESPONDING TO A CERTAIN TYPE NUMBER. THE COLUMN ENTRIES IS SPECIFIED BY THE ENTRIES IN MCOTAB. A ZERO COLUMN ENTRY INDICATES THE END OF THE LIST. SINCE COLUMNS ARE CONTIGUOUS IN CORE, #ROWS IS NON-LIMITING SO LONG A MCOTAB CONSIDERES THAT ONE SUBCATEGORY USES >1 COLUMN.

Table 3-1. Global Variables -- ZODCLS (Page 57 of 58)

```
MSUTAB       15,18 I4    CONTAINS THE STATISTIC REQUEST NAMES.

MSUTYP       15,18 I2    CONTAINS THE STATISTIC TYPE NUMBER. SEE SODATA.

--------------------------------------------------------------------------------
MATCH COMMON
MATTAB       120   I2    COMPUTATIONAL MATCH TABLE INIT. FROM MATTAX

MATTAX       120   I2    MATCH TABLE. IS INDEXED BY TYPE NUMBER
                           CODES ARE:
                           -1 = DO NOT KNOW
                            0 = DO NOT WANT
                            X = DO WANT. X IS A POSITIVE INTEGER WHICH
                                  SERVES AS A POSITIVE INTEGER TO THE
                                  XTH ROW OF THE REQUEST TABLE

--------------------------------------------------------------------------------
MISCELLANEOUS: IN ZODCLS, BUT NOT IN ANY COMMON
KMSUB        -     I4    MAX NUMBER OF UNIGUE STATISTICS;OVERRIDDEN BY
                           DEFINITIONS IN SODCLS;INIT TO 270
```

Table 3-1. Global Variables -- ZSYSMAX (Page 58 of 58)

COMPILE-TIME MAXIMA:
THE FOLLOWING VARIABLE NAMES DEFINE THE MAXIMUM NUMBER OF ENTITIES AVAILABLE WITHOUT RECOMPILING. ARRAYS ARE DIMENSIONED USING THESE VARIABLES. THEY ARE PREPROCESSOR VARIABLES AND ASSIGNED VALUES IN ONE CENTRALLY LOCATED MEMBER.

NAME	VALUE	DESCRIPTION – MAXIMUM NUMBER OF:
KMMSGS	25	MESSAGES OF ANY KIND THAT CAN BE ISSUED BEFORE TERMINATION
KMMSGI	15	INFORMATION MESSAGES BEFORE TERMINATION
KMMSGW	15	WARNING MESSAGES BEFORE TERMINATION
KMMTYP	10	ANY ONE MESSAGE THAT CAN BE ISSUED PRIOR TO TERMINATION

SECTION 4. DEBUG TOOLS

Tables 4-1, 4-2, and 4-3 list the associations between debug flag numbers and code segments for the IP, MP, and OP, respectively. This intermediate debug output is turned on by the use of a FLAG card in the IANDD.RNTIM input to each processor. The format of the FLAG card is given in the User's Manual. Turning on such flags causes one or more debug messages to be printed. Each message contains the flag number, a short line of text, and the name (first six characters) and value of as many as ten variables.

Table 4-1. Input Processor Debug Flags

ROUTINE	ENTRY/EXIT FLAGS	BODY FLAGS
SICUMP	250	
SIGIAT	253	266
SINPUT		200
		290
		296
SITDGN	255	265
SIVDGN	256	261
SMRNG	83	262
SMRSEL		270

Table 4-2. Model Processor Debug Flags (Page 1 of 2)

ROUTINE	BODY FLAGS	ENTRY/EXIT FLAGS
SAASYN	103	143
SACKR	44	74
SADADD	45	75
SAFAIL	104	144
SAFINM	132	117
SAFINS	101	141
SAFLAG	109	149
SAINIT	20	70
SAMAIN	102	142
SANFEL	47	77
SANMDL	108	148
SANTSA	48	78
SANXTN	49	79
SAPFEL	50	80
SARFEL	51	81
SASAMP	127	157
SASCTL	161	160
SASPRM	121	151
SATORG	105	145
SATRD	120	150
SAUCTL	181	180
SAUPRM	125	155
SAVORG	106	146
SAVRD	107	147
SAWTIX	52	82
SAZNIT	128	158
SERROR	42	72
SMBRD	169	168
SMDBRD	191	190
SMDETR	195	194
SMENTR	197	196
SMEUM	193	192
SMLTIM	173	172
SMNXST	171	170
SMTABG	189	188
SMDIVF	126	156
SMDIVO	123	153
SMDIVS	124	154
SMRNG	53	83
SMRSEL	54	270
SSLEAV	167	166
SSMOD	175	174

Table 4-2. Model Processor Debug Flags (Page 2 of 2)

ROUTINE	BODY FLAGS	ENTRY/EXIT FLAGS
SSMODA	163	162
SSMOBB	177	176
SSMODN	165	164
SSTEST	179	178
SUMOD	185	184
SULEAV	187	186
SZHDR	131	118
SZINT	129	159
SZSTAT	122	152
SZZERO	130	119

Table 4-3. Output Processor Debug Flags

ROUTINE	FLAGS
DAYTIM	--
SHIST	1
SLIST	1
SONTIX	--
SOPSUM	5
SOUTPT	1,20
SOWTIX	1
SOZNIT	1
SREAD02	1
SREAD03	1
SREAD04	1
SREQTLU	50
SSETUP	16
SZPLOT	1
SPREAD	1,3,40
ZABIN	1
ZBINL	9
ZBNCHK	--
ZDBIN	--
ZDUMBIN	--
ZERROR	--
ZFLAG	--
ZGRAPH	98
ZHEADER	40
ZHIST	--
ZLIST	--
ZMNMX	--
ZRCLEAN	--
ZREQU	1
ZSHIFT	--
ZSKIPFO	--
ZSTORE	--

SECTION 5. SUBPROGRAM LOGIC TABLES

Tables 5-1, 5-2, and 5-3 contain subprogram name, entry points, called by calls, and functions for the subprograms of the IP, MP, and OP, respectively.

Table 5-1. Input Processor -- Subroutine Logic Table (Page 1 of 3)

SUBROUTINE LOGIC TABLE - INPUT PROCESSOR

CSECT	ENTRY	CALLED BY	CALLS	FUNCTION
DAYTIM	DAYTIM	SINPUT SISWRT	TIMES	GET DATE AND TIME OF DAY.
ERROR (1)	ERROR	GDIP4 SAFLAG SICHCK SINERR SINPUT SIREPT SISCFG SITDGN SIVDGN		WRITE INFORMATION, WARNING, OR SEVERE ERROR MESSAGE. COUNT MSG OCCURRENCES BY TYPE AND MSG NO. AND TERMINATE WHEN COUNT(S) EXCEED LIMITS.
GDIP-SECT	GDIP4	NDBOR	ERROR GDIPF4 GDIPH4 GDIPX4	READ INTO USER DEFINED INPUT AREA USING USER SPECIFIED FORMAT.
GDIPF4 (1)	GDIPF4	GDIP4		READ FULL WORD GDIP DATA
GDIPH4 (1)	GDIPH4	GDIP4		READ HALF WORD GDIP DATA
GDIPX4 (1)	GDIPX4	GDIP4		READ BYTE SIZE GDIP DATA
NDBOR (1)	NDBOR	SINPUT	GDIP4	READ FORMAT DATA FOR GDIP DATA
SACOMN	SACOMN			USED BY IP AND MP TO INSURE IDENTICAL ORDERING OF INPUT COMMON AREAS. NO ROUTINE CALLS SACOMN; IT ACCOMPLISHES ITS FUNCTION BY BEING LINK EDITED AHEAD OF ANY RTN WHICH USES THE COMMONS BEING ORDERED.
SAFLAG	SAFLAG	SINPUT	ERROR	SET FLAGS FOR INTERMEDIATE OUTPUT.
SIADDR	SIADDR	SIINIT	SISADD	PROVIDE ADDRESSABILITY TO SYSTEM CHARACTERISTICS COMMONS SO THEY CAN BE WRITTEN BY IP TO STRUCTURED DATA FILE
SISWRT	SISADD	SIADDR		SAVE ADDRESS AND LENGTH OF SYSTEM CHARACTERISTICS COMMON AREA
	SISWRT	SINPUT	DAYTIM	WRITE SYSTEM CHARACTERISTICS STRUCTURED DATA FILE
SICHCK	SICHCK	SINPUT	ERROR SINERR	CHECK REASONABLENESS OF INPUT DATA. SET INITIAL VALUES FOR MP. CONVERT INPUT TIMES FROM SECONDS TO CLOCK UNITS.
SICUMP	SICUMP	SIREPT SITDGN SIVDGN		CONVERT PROBABILITY DISTRIBUTION TO CUMULATIVE PROBABILITY DISTRUBUTION.
SIGIAT	SIGIAT	SIVDGN	SMRNG SMRSEL	COMPUTE TRAIN INTERARRIVAL TIME FOR VEHICLE DEMAND GENERATION.

Table 5-1. Input Processor -- Subroutine Logic Table (Page 2 of 3)

SIINIT	SIINIT	SINPUT	LODCOM SIADDR SIPLST	INITIALIZE / SET DEFAULT PARAMETERS FOR USER INPUT. INVOKE ROUTINES TO ESTABLISH ADDRESS AND SIZE OF SYS CHAR COMMON AREA FOR MP AND DETERMINE USER SPECIFIED FILE MEMBER NAMES.
SIMNAM	SIMNAM	SIPLST	DAYTIM	SCAN PARM FIELD CHARACTER STRING, SEPARATE INTO FILE MEMBER NAMES, AND WRITE LOAD MODULE DATE AND TIME.
	SIWNAM	SINPUT		LIST USED MEMBERS IN INDEX.
SINERR	SINERR	SICHCK SINPUT SIREPT	ERROR	ACCEPT IP MSG NO. AND SEVERITY CODE AND CALL ERROR TO WRITE THE MESSAGE.
SINPUT	SINPUT	SIPARM	DAYTIM ERROR NDBOR SAFLAG SICHCK SIINIT SINERR SIREPT SISCFG SISWRT SITDGN SIVDGN SPIEL SIWNAM	CONTROL INPUT PROCESSING - READ USER INPUT GENERATE TRIP STRUC FILE GENERATE VEH STRUC FILE GENERATE SYS CHAR STRUC FILE
SIPARM	SIPARM	SYSTEM JOB STEP TASK	SINPUT	SAVE ADDRESS OF PARM FIELD PASSED BY SYSTEM. CALL MAIN IP ROUTINE.
	SIPLST	SIINIT	SIMNAM	PASS PARM FIELD AND PARM FIELD LENGTH TO ROUTINE WHICH DIVIDES FIELD INTO FILE MEMBER NAMES AND VEH SOURCE.
SIPSAV	SIPSAV	COMMON DEFINED IN SIINIT		PROVIDE STORAGE LOCS(COMMON SIPSAV) FOR ADDRESSES OF IP COMMONS, SYS CHAR COMMONS, AND END OF SYS CHAR COMMONS.
	LODCOM	SIINIT		CAUSE ABOVE ADDRESSES TO BE LOADED INTO SIPSAV BY EXECUTING RETURN TO CALLER.
SIREPT	SIREPT	SINPUT	ERROR SICUMP SINERR	WRITE INITIAL CONDITIONS REPORT FOR STATION LINK CHARAC., SYSTEM CHARAC., AND SERVICE CHARAC CHECK PARAMETERS FOR ERRORS.
SISCFG	SISCFG	SINPUT	ERROR	BUILD STATION STRUCTURED DATA TABLES FROM USER INPUT. DETER- UPSTRM AND DNSTRM LINKS. CHECK USER DATA FOR ERRORS.
SITDGN	SITDGN	SINPUT	ERROR SICUMP SMRNG SMRSEL	GENERATE TRIP ARRIVAL FILE AND TRIP SUMMARY REPORT.
SIVDGN	SIVDGN	SINPUT	ERROR SICUMP SIGIAT	GENERATE VEHICLE ARRIVAL FILE AND VEHICLE SUMMARY REPORT.

Table 5-1. Input Processor -- Subroutine Logic Table (Page 3 of 3)

CSECT	ENTRY	CALLED BY	CALLS	FUNCTION
			SMRNG SMRSEL	
SMRNG	SMRNG	SIGIAT SITDGN SIVDGN SMRSEL		GENERATE RANDOM NUMBER BETWEEN 0-1.
SMRSEL	SMRSEL	SIGIAT SITDGN SIVDGN	SMRNG	CHOOSE RANDOM ENTRY IN CUMULATIVE PROBABILITY DISTRIBUTION.
TIMES (1)	TIMES	DAYTIM		GET DAY AND DATE FROM SYSTEM CLOCK
TRACBK (1)	TRACBK	INTRPTD PGM	TRCKI TRCBKV TRCBKR	GET REGISTER AND ARGUMENT TRACE INFORMATION
	SPIEL	SINPUT		SET INTERRUPT FLAGS TO GET CONTROL AT PGM INTERRUPT TIME
TRCBKP (1)	TRCBKI	TRACBK		PRINT PGM INT HEADING
	TRCBKV	TRACBK		PRINT 2 LINES FOR ARGUMENT
	TRCBKR	TRACBK		PRINT 3 LINES FOR GEN REG

NOTES:
(1) SEVERAL OF THE CSECTS ABOVE ARE KNOWN BY DIFFERENT SOURCE NAMES. SINCE OTHER DOCUMENTATION MAY REFER TO SUBROUTINES BY SOURCE NAME RATHER THAN CSECT NAME, THE CSECTS WITH THEIR SOURCE NAMES ARE LISTED BELOW.

CSECT	SOURCE MEMBER
ERROR	SIERROR
GDIPF4	XGDIPF4
GDIPH4	XGDIPH4
GDIPSECT	SIGDIP4
GDIPX4	XGDIPX4
NDBOR	XNDBOR
TIMES	DTIMEL
TRACBK	XTRACBK
TRCBKP	XTRCBKP

Table 5-2. Model Processor -- Subroutine Logic Table (Page 1 of 5)

SUBROUTINE LOGIC TABLE - MODEL PROCESSOR

CSECT	ENTRY	CALLED BY	CALLS	FUNCTION
DAYTIM	DAYTIM	SAWTIX	TIMES	CONVERT DATE & TIME TO YY/MM/ DD/HH/MM/SS
ERROR (1)	ERROR	GDIP4 SAFLAG SAMAIN SMBRD SMDETR SMENTR SMTABQ SSMODA SSMODB SSMODN SUMOD SAASYN SZSTAT SACKR SAVORG SMDIVS SMDIVF SAVRD SANMDL SATRD SADADD SAINIT SAPFEL	SAPINS TRACBK	WRITE INFORMATION, WARNING, OR SEVERE ERROR MESSAGE. COUNT MSG OCCURRENCES BY TYPE AND MSG NO. AND TERMINATE WHEN COUNT(S) EXCEED LIMITS.
GDIP-SECT	GDIP4	NDBOR	ERROR GDIPF4 GDIPH4 GDIPX4	READ INTO USER DEFINED INPUT AREA USING USER SPECIFIED FORMAT.
GDIPF4 (1)	GDIPF4	GDIP4		READ FULL WORD GDIP DATA
GDIPH4 (1)	GDIPH4	GDIP4		READ HALF WORD GDIP DATA
GDIPX4 (1)	GDIPX4	GDIP4		READ BYTE SIZE GDIP DATA
NDBOR (1)	NDBOR	SAASYN	GDIP4	READ FORMAT DATA FOR GDIP DATA
PSEUDO (1)	PSEUDO	GDIP4		XPSEUDO-MAIN ENTRY
	SUDOGO	GDIP4		INITIALIZE PSEUDO-I/O
SAASYN	SAASYN	SAMAIN	ERROR NDBOR SAFAIL SAFLAG SACKPT SAPFEL SATORG SAVORG	ASYNCHRONOUS DATA READ
SACKR	SACKR	SANSAV		CHECKPOINT & RESTART PROCESSING
	SACKPT	SAASYN SAMAIN		WRITE CHECKPOINT RECORD

Table 5-2. Model Processor -- Subroutine Logic Table (Page 2 of 5)

	SAREST	SAINIT	ERROR SAPFEL	READ CHECKPOINT RECORDS & RESET FILES
SACOMN	SACOMN	NONE	NONE	USED BY IP AND MP TO INSURE IDENTICAL ORDERING OF INPUT COMMON AREAS. NO ROUTINE CALLS SACOMN; IT ACCOMPLISHES ITS FUNCTION BY BEING LINK EDITED AHEAD OF ANY RTN WHICH USES THE COMMONS BEING ORDERED.
SADADD	SADADD	SANSAV	ERROR	INITIALIZE INPUT AREA ADDRESSES AND MESSAGES.
	SANDTA	SAINIT		READ INPUT DATA INTO INPUT COMMONS
SAFAIL	SAFAIL	SAASYN	SACKPT	FAILURE ACTIVITY PROCESSING
SAFINM	SAFINM	SAMAIN SASAMP		WRITE ONLINE MODEL REPORT & FINAL MODEL REPORT
SAFINS	SAFINS	SAMAIN ERROR	SAWTIY	FEL USAGE REPORT
SAFLAG	SAFLAG	SAASYN	ERROR	SET FLAGS FOR INTERMEDIATE OUTPUT.
SAINIT	SAINIT	SAMAIN	SANTSA SAREST SANDTA SAFLAG MDBOR SANXTN SANFEL SANMDL SAPFEL ERROR SAUPTX	INITIALIZE SIMULATION
SAMAIN	SAMAIN	SANTIX	SZSTAT SSMOD SSTEST SSLEAV ERROR SUMOD SMTABQ SAINIT SAASYN SASAMP SACKPT SATRD SAVRD SAVORG SAVRD SAPFEL SAFINM SAFINS SZINT	MAIN CONTROL LOOP
SANFEL	SANFEL	SAINIT		INITIALIZE FUTURE EVENTS LIST
SANMDL	SANMDL	SAINIT	ERROR	MODEL VARIABLE INITIALIZATION
SANSAV	SANSAV	SANTSA	SADADD SACKR	INIT CKPT & SYSTEM DATA READ PROCESSES
SANTIX	SANTIX	OPER. SYSTEM	SAMAIN	INIT MEMBER NAME STRING FOR INDEX FILE

Table 5-2. Model Processor -- Subroutine Logic Table (Page 3 of 5)

	SAUPTX	SAINIT	SAWTIX	PASS MEMBER NAME STRING TO SAWTIX
SANTSA	SANTSA	SAINIT	SANSAV	INIT SYSTEM STATUS AREA ADDRESSES
SANXTN	SANXTN	SAINT		INIT XTN HEADER DATA & AVAIL- ABLE LISTS
SAPFEL		SATABQ SSMODB SUMOD SAREST SAMAIN SAASYN SAVRD SATRD	SZSTAT ERROR	PUT XTN ON FUTURE EVENTS LIST
SASAMP	SASAMP	SAMAIN	SZINT SZHDR SZZERO	SAMPLE EVENT PROCESSING
SATORG	SATORG	SAMAIN SAASYN	SZSTAT SUMOD	MOVE ARRIVING TRIP
SATRD	SATRD	SAMAIN	ERROR	READ TRIP FROM TRIP FILE
SAVORG	SAVORG	SAMAIN SAASYN	ERROR SZSTAT SSMOD	MOVE ARRIVING VEHICLE
SAVRD	SAVRD	SAMAIN	ERROR	READ VEHICLE FROM VEHICLE FILE
SAWTIX	SAWTIX SAWTIY	SAUPTX SAFINS	DAYTIM	PARSE PARM LIST CHARACTER STRING, SEPARATE INTO FILE MEMBER NAMES, AND WRITE LOAD MODULE DATE AND TIME. LIST USED MEMBERS IN INDEX.
SAZNIT	SAZNIT	SAINIT	SZZERO SZHDR	INITIALIZE STATISTICAL VARIA- BLES
SMBRD	SMBRD	SSMODB	SMRSEL ERROR	PLANNING TRIP BOARDING
SMDETR	SMDETR	SSMODB	ERROR	DETRAIN VEHICLE FROM LEAD VEHI- CLE OF A TRAIN
SMDIVF	SMDIVF	SSTEST	ERROR SMDIVO SMDIVS	DETRAIN VEHICLE FROM LEAD VEHI- CLE OF A TRAIN
SMDIVO	SMDIVO	SMDIVF		DETRAIN VEHICLE FROM LEAD VEHI- CLE OF A TRAIN
SMDIVS	SMDIVS	SMDIVF	ERROR	DETRAIN VEHICLE FROM LEAD VEHI- CLE OF A TRAIN
SMENTR	SMENTR	SSMODA	ERROR	DETRAIN VEHICLE FROM LEAD VEHI- CLE OF A TRAIN
SMRNG	SMRNG	SMRSEL		GENERATE RANDOM NUMBER BETWEEN 0-1.
SMRSEL	SMRSEL	SMBRD SMTABQ SSMODA SSMODB	SMRNG	CHOOSE RANDOM ENTRY IN CUMULA- TIVE PROBABILITY DISTRIBUTION.

Table 5-2. Model Processor -- Subroutine Logic Table (Page 4 of 5)

SMTABQ	SMTABQ	SAMAIN SSMODA	ERROR SAPFEL SMRSEL	PREPARE A TRIP FOR BOARDING
SSASAV	SSASAV	NONE	NONE	INIT ARRAY SYSTEM STATUS AREA WORDS
SSLEAV	SSLEAV	SAMAIN	SZSTAT SSMOD	PROCESSING A VEHICLE/TRAIN LEAVING A SL
SSMOD	SSMOD	SAMAIN	SSMODA SSMODN SSMODB	MODEL THE VEHICLE ON ITS CURRENT STATION LINK
SSMODA	SSMODA	SSMOD	SZSTAT SMTABQ SMENTR SMRSEL ERROR	VEHICLE PROCESSING AFTER A STATION LINK EVENT
SSMODB	SSMODB	SSMOD	SMDETR SMBRD SZSTAT SMRSEL ERROR SAPFEL SMRSEL	VEHICLE PROCESSING BEFORE A STATION LINK EVENT
SSMODN	SSMODN	SSMOD	SZSTAT ERROR	VEHICLE'S NEXT SL EVENT DETERMINATION
SSTEST	SSTEST	SAMAIN	SMDIVF	STATION LINK ENTRY TESTING & NEXT LINK DETERMINATION
SUMOD	SUMOD	SAMAIN	SZSTAT ERROR SAPFEL	MODEL THE TRIP ON ITS CURRENT TRIP LINK
SZHDR	SZHDR	SASAMP		WRITE SAMPLING HEADER RECORD
SZINT	SZINT	SASAMP		CALCULATE INTEGRALS, AVERAGES, & MISCELLANEOUS STATISTICS
SZSTAT	SZSTAT	SAMAIN SSLEAV SSMODA SSMODB SSMODN SUMOD SAPFEL	ERROR	COLLECT STATISTICS
SZZERO	SZZERO	SASAMP SAZNIP		RESET STATISTICS
TIMES (1)	TIMES	DAYTIM		GET DAY AND DATE FROM SYSTEM CLOCK
TRACBK (1)	TRACBK	ERROR	TRCKI TRCBKV TRCBKR	GET REGISTER AND ARGUMENT TRACE INFORMATION
	SPIEL	SAINIT		SET INTERRUPT FLAGS TO GET CONTROL AT PGM INTERRUPT TIME
TRCBKP (1)	TRCBKI	TRACBK		PRINT PGM INT HEADING
	TRCBKV	TRACBK		PRINT 2 LINES FOR ARGUMENT
	TRCBKR	TRACBK		PRINT 3 LINES FOR GEN REG

Table 5-2. Model Processor -- Subroutine Logic Table (Page 5 of 5)

CSECT	ENTRY	CALLED BY	CALLS	FUNCTION

NOTES:
(1) SEVERAL OF THE CSECTS ABOVE ARE KNOWN BY DIFFERENT SOURCE NAMES. SINCE OTHER DOCUMENTATION MAY REFER TO SUBROUTINES BY SOURCE NAME RATHER THAN CSECT NAME , THE CSECTS WITH THEIR SOURCE NAMES ARE LISTED BELOW.

CSECT	SOURCE MEMBER
ERROR	SERROR
GDIPF4	XGDIPF4
GDIPH4	XGDIPH4
GDIPSECT	SMGDIP4
GDIPX4	XGDIPX4
NDBOR	XNDBOR
PSEUDO	XPSEUDO
TIMES	DTIMEL
TRACBK	XTRACBK
TRCBKP	XTRCBKP

Table 5-3. Output Processor -- Subroutine Logic Table (Page 1 of 3)

SUBROUTINE LOGIC TABLE - OUTPUT PROCESSOR

CSECT	ENTRY	CALLED BY	CALLS	FUNCTION
ABIN (1)	ABIN	STOFLO	ERROR SHIFT	BIN REALLOCATION
BNCHK (1)	BNCHK	ZHIST ZREQU	ERROR SHIFT	BIN EXPANSION
DAYTIM	DAYTIM	SOWFIX	TIMES	CONVERT DATE & TIME TO YY/MM/DD/HH/MM/SS
DBIN	DBIN	SOZNIF	-	ALLOCATE BIN STORAGE
DUMBIN (1)	DUMBIN	SOUTPT	-	PRINT BIN AREA HEADERS
ERROR (1)	ERROR	ZREQU ZREAD ABIN BNCHK HEADER READ02 READ03 READ04 SHIFT SKIPFO SOUTPT REQTLU	TRACBK	WRITE INFORMATION, WARNING, OR SEVERE ERROR MESSAGE. COUNT MSG OCCURRENCES BY TYPE AND MSG NO. AND TERMINATE WHEN COUNT(S) EXCEED LIMITS.
GRAPH (1)	GRAPH	ZPLOT	-	PRINT TIME SERIES PLOT
HEADER (1)	HEADER	ZREAD	ERROR	READ NEXT HEADER RECORD
HIST (1)	HIST	ZHIST	-	PRINT HISTOGRAM
LIST (1)	LIST	ZLIST	-	LIST VALUES
MNMX (1)	MNMX	ZHIST	-	COMPUT MINIMUM & MAXIMUM OF BIN
RCLEAN	RCLEAN	ZREAD	-	RESET BIN ADDRESSES
READ02 (1)	READ02	ZREAD	ERROR STOFLO	READ SYSTEM STATISTICS
READ03 (1)	READ03	ZREAD	ERROR STOFLO	READ STATION LINK STATISTICS
READ04 (1)	READ04	ZREAD	ERROR STOFLO	READ TRIP LINK STATISTICS
REQTLU (1)	REQTLU	ZREAD	ERROR	RECORD/REQUEST CORRELATION
SETUP (1)	SETUP	ZREAD	-	INITIALIZE OP TABLE VALUES
SHIFT (1)	SHIFT	BNCHK	ERROR	REALLOCATE BIN STORAGE ASSIGNMENTS
SKIPFO	SKIPFO	ZREAD	ERROR	SKIP A FOLLOWER RECORD

Table 5-3. Output Processor -- Subroutine Logic Table (Page 2 of 3)

(1)				
SODATA	SODATA	-	-	INITIALIZE MAJOR COMMON AREAS
SONTIX	SONTIX	OPER. SYSTEM	SOUTPT	ESTABLISH PARM FIELD ADDRESS-ABILITY
	SOUPTX	SOUTPT	-	PASS MEMBER NAME TO SOWTIX
SOPSUM	SOPSUM	SOUTPT	-	PERFORMANCE SUMMARY PROCESSING
SOUTPT	SOUTPT	SONTIX	SOZNIT DUMBIN ZFLAG ZREAD ZLIST ZHIST ZPLOT ZREQU ERROR SOJPTX	DSM-OP MAIN CONTROL
SOWTIX	SOWTIX	SOUPTX	DAYTIM	SCAN PARM FIELD CHARACTER STRING, SEPARATE INTO FILE MEMBER NAMES, AND WRITE LOAD MODULE DATE AND TIME.
	SOWTIV	SOUTPT		LIST USED MEMBERS IN INDEX.
	SOWTIY	SOUTPT		WRITE PERSUM MEMBER NAME IN PS.
SOZNIT	SOZNIT	SOUTPT	SPIEL DBIN ZREQU ZREAD	INITIALIZATION OF OP
STORE (1)	STORE	-	-	STORE DATA IN BIN
	STOFLO	READ02 READ03 READ04	ABIN	STORE DATA IN BIN
TIMES (1)	TIMES	DAYTIM	-	GET DAY AND DATE FROM SYSTEM CLOCK
TRACBK (1)	TRACBK	ERROR	TRCKI TRCBKV TRCBKR	GET REGISTER AND ARGUMENT TRACE INFORMATION
	SPIEL	SOZNIT	-	SET INTERRUPT FLAGS TO GET CONTROL AT PGM INTERRUPT TIME
TRCBKP (1)	TRCBKI	TRACBK	-	PRINT PGM INT HEADING
	TRCBKV	TRACBK	-	PRINT 2 LINES FOR ARGUMENT
	TRCBKR	TRACBK	-	PRINT 3 LINES FOR GEN REG
ZBINL	ZBINL	SOUTPT	-	FIND BIN LENGTH
ZFLAG	ZFLAG	SOUTPT	-	SET INTERMEDIATE FLAGS.
ZHIST	ZHIST	SOUTPT	BNMX BNCHK HIST	HISTOGRAM OUTPUT CONTROL
ZLIST	ZLIST	SOUTPT	LIST	LIST OUTPUT CONTROL
ZPLOT (1)	ZPLOT	SOUTPT	GRAPH	PLOT OUTPUT CONTROL
ZREAD	ZREAD	SOZNIT	HEADER	ACQUIRE SYSTEM CONSTANTS

Table 5-3. Output Processor -- Subroutine Logic Table (Page 3 of 3)

(1)		SOUTPT ERROR SETUP REQTLU READ02 READ03 READ04 RCLEAN	SKIPFO	
ZREQU	ZREQU	SOZNIT SOUTPT	ERROR ENCHK	REQUEST HANDLING
CSECT	ENTRY	CALLED BY	CALLS	FUNCTION

NOTES:
(1) SEVERAL OF THE CSECTS ABOVE ARE KNOWN BY DIFFERENT SOURCE NAMES. SINCE OTHER DOCUMENTATION MAY REFER TO SUBROUTINES BY SOURCE NAME RATHER THAN CSECT NAME , THE CSECTS WITH THEIR SOURCE NAMES ARE LISTED BELOW.

CSECT	SOURCE MEMBER
ERROR	ZERROR
TIMES	DTIMEL
TRACBK	XTRACBK
TRCBKP	XTRCBKP
ABIN	ZABIN
ENCHK	ZENCHK
DUMBIN	ZDUMBIN
GRAPH	ZGRAPH
HEADER	ZHEADER
HIST	SHIST
LIST	SLIST
MNMX	ZMNMX
READ02	ZREAD02
READ03	ZREAD03
READ04	ZREAD04
REQTLU	ZREQTLU
SETUP	SSETUP
SHIFT	ZSHIFT
SKIPFO	ZSKIPFO
STORE	ZSTORE
ZPLOT	SZPLOT
ZREAD	SZREAD

SECTION 6. DSM SUBPROGRAM DESCRIPTIONS

This section describes the components of the DSM Input Processor, Model Processor, and Output Processor. These components include subroutines, macros, and included code segments. They are identified by their source library member names. The global variables used in these PARAFOR, ASSEMBLER, and PL/I components are defined in Section 3. Local variables which are arguments in the calling sequence of these modules are listed in the component's Argument Dictionary of each description. All arguments are assumed to be input only unless "OUTPUT" or "INPUT and OUTPUT" has been explicitly stated. Other local variables are listed in the Local Variable Dictionary of each description.

For each local variable the following is provided:

- o Variable name: the name by which the variable is known in its module. Since arguments may not be named in Assembly Language routines, an arbitray name has been assigned.

- o Dimension: A hyphen indicates that there is only one variable (a scalar) by the variable name. A number, n, e.g., 2, indicates that multiple variables of that name are defined with subscripts from 1 to n, e.g., variable(1), variable(2).

- o Type: FORTRAN notation is used to identify the type and length of the variable, e.g., I*4 = full word integer, R*8 = double word real number. When a character string or variable name is an argument, the letter "C" is specified. When the character string must be a specified length, that too is shown, e.g., "T" = C*1.

- o Description: A brief definition of the variable is given. If it is an optional argument in the calling sequence, that is stated and its default value is given.

In addition to local variables, a description of the module's logic is provided as are any supporting decision tables and algorithms. The descriptions parallel the PDL (Program Design Language), which is the detailed logic of the program making reference to local and global variables. The PDL is given in Appendix A.

6.1 INPUT PROCESSOR

This section outlines the subprogram descriptions for the DSM-Input Processor.

6.1.1 DAYTIM

See subsection 6.2.1, DAYTIM.

6.1.2 ERROR

See subsection 6.2.43, SERROR.

6.1.3 GDIPSECT

See subsection 6.2.52, SMGDIP4.

6.1.4 SACOMN

See subsection 6.2.15, SACOMN.

6.1.5 SAFLAG

See subsection 6.2.20, SAFLAG.

6.1.6 SIADDR

6.1.6.1 Identification

- o SIADDR - System Characteristics Address Save
- o IBM/FSD - July 1, 1977
- o Assembler H

6.1.6.2 Argument Dictionary

PARAMETER	DIM	TYPE	DESCRIPTION
Parm 1	-	A	Address of start of System Characteristics commons (Input)
Parm 2	-	F	Length in words of System Characteristics commons (Input)

6.1.6.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
ARGA	-	F	First item in System Characteristics common area
ARGB	-	F	Length in words of System Characteristics commons

6.1.6.4 Description - SIADDR receives the address and length of the System Characteristics common area and passes them to SISADD where they are saved for use in the structured data file write of SIBWRT.

6.1.6.5 PDL - See Appendix A.

6.1.6.6 Decision Tables and Algorithms - None.

6.1.7 SIBWRT

6.1.7.1 Identification

- o SIBWRT - Structured Data File Write
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.7.2 Argument Dictionary

ENTRY	PARAMETER	DIM	TYPE	DESCRIPTION
SISADD	ADDR1	LEN1	I*4	System Characteristics structured data area (Input)
	LEN1	-	I*4	Length (in full words) of System Characteristics structured data area (Input)
SISWRT	NONE			

6.1.7.3 Local Variable Dictionary - None.

6.1.7.4 Description - SIBWRT has two entry points:

1. SISADD is called by SIADDR to save the address and length of the System Characteristics common area.

2. SISWRT writes the System Characteristics to the Structured Data File using the address and length saved by SISADD.

6.1.7.5 PDL - See Appendix A.

6.1.7.6 Decision Tables and Algorithms - None.

6.1.8 SICHCK

6.1.8.1 Identification

- o SICHCK - Parameter Checking and Initialization
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.8.2 Argument Dictionary - None.

6.1.8.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
DE	-	I*4	Deboard event type (3)
BE	-	I*4	Board event type (4)
DBE	-	I*4	Deboard/board event type (5)
SE	-	I*4	Storage event type (6)
LE	-	I*4	Launch event type (7)
IDB	-	I*4	If = 1, deboard/board event(s) found
IVS	-	I*4	Event list pointer
IVT	-	I*4	Points to previous or next event for a link
IERR	-	I*4	If = 1, serious error found. Terminate
TCNVRT	-	R*4	Used to convert seconds to clock units (CU'/sec)

6.1.8.4 Description - SICHCK converts several groups of input time parameters from seconds to clock units for the model processor.

- o Station link travel time
- o Station link headway
- o Vehicle headway and spacing
- o Vehicle delay time
- o Deboard/board time
- o Trip link travel time
- o Deboard Exit Walk and Transfer Walk Times.

Following the time conversions, SICHCK verifies that certain station link-event combinations occur correctly.

- o Launch and store events must be last where they occur
- o A store event must be the last event on a storage link
- o Deboard/board events and downstream station link must occur together.

Finally, SICHCK finds the source station links if the station configuration has not done so.

6.1.8.5 PDL - See Appendix A.

6.1.8.6 Decision Tables and Algorithms - None.

6.1.9 SICUMP

6.1.9.1 Identification

- o SICUMP - Cumulative Probability Distribution Conversion
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.9.2 Argument Dictionary

PARAMETER	DIM	TYPE	DESCRIPTION
DISTR	DENTS	R*4	Probability distribution to be converted (Input)
DENTS	-	I*2	Number of entries in distribution (Input)
DERR	-	I*4	Return code (0 = no error, 1 = error, invalid probability distribution) (Output)
DMEAN	-	R*4	If = 1 on input, SICUMP returns mean entry number of probability distribution (Input and Output)

6.1.9.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
DMEVAL	-	R*4	Used to compute mean entry number of distribution

6.1.9.4 Description - Beginning with distribution entry two, SICUMP adds each entry to the previous one and saves the result in the current entry so that the final entry is the sum of the whole distribution. If the mean is requested, SICUMP computes the sum of each entry number times the entry value. If any entry is less than 0 on input or if the sum of the entries is greater than 1.0, processing stops and an error indicator is returned to the caller.

6.1.9.5 PDL - See Appendix A.

6.1.9.6 Decision Tables and Algorithms - None.

6.1.10 SIGIAT

6.1.10.1 Identification

- o SIGIAT - Vehicle Interarrival Time Generation
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.10.2 Argument Dictionary - None.

6.1.10.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
DENTY	-	I*2	Entry no. in user IAT dist. (from SMRSEL)
DHSTAR	-	R*4	Intermediate value in computation of IAT
DHLOGN	-	R*4	Intermediate value in computation of IAT

6.1.10.4 Description - SIGIAT computes vehicle interarrival time in one or two ways as requested by the user.

- o User distribution option

  SIGIAT calls SMRSEL to sample the user interarrival time distribution.

- o Exponential distribution option

SIGIAT gets a random probability from SMRNG. If the probability is less than the probability of minimum headway, the interarrival time is set to minimum headway. Otherwise, interarrival time is computed as follows:

$$\text{IAT} = \text{MINH} - \frac{\text{MEANH} - \text{MINH}}{1 - \text{PMINH}} * \text{LN}\left(\frac{1 - \text{PRAND}}{1 - \text{PMIN}}\right)$$

Where:

IAT = Interarrival time

MINH = Minimum headway

MEANH = Mean headway

PMINH = Probability of exactly minimum headway

PRAND = Random probability.

In both cases if the environment is synchronous, interarrival time is really interarrival slots and SIGIAT multiplies the slots by slot length in seconds to get a time value.

6.1.10.5 <u>PDL</u> - See Appendix A.

6.1.10.6 <u>Decision Tables and Algorithms</u> - The following is the derivation of the interarrival time algorithm used above in Description.

$$h = h_m - \frac{\bar{h} - h_m}{1 - F_0} \ln\left(\frac{1 - R}{1 - F_0}\right) \tag{1}$$

Where:

h_m = Minimum headway (input by user as DVHDWY)

$\bar{h}$ = Mean headway (input by user as DVLMDA)

F_0 = Probability of exactly minimum headway (input by user as DVPMIN)

R = Random number uniformly distributed between 0 and 1

h = Interarrival time = headway for specific vehicle.

This is derived from the following equation given in <u>Adaptive Merging Under Cap-Follower Control</u> by S. J. Brown, Jr. (APL/JHU CP038/TPR029 October 1974), Page 49.

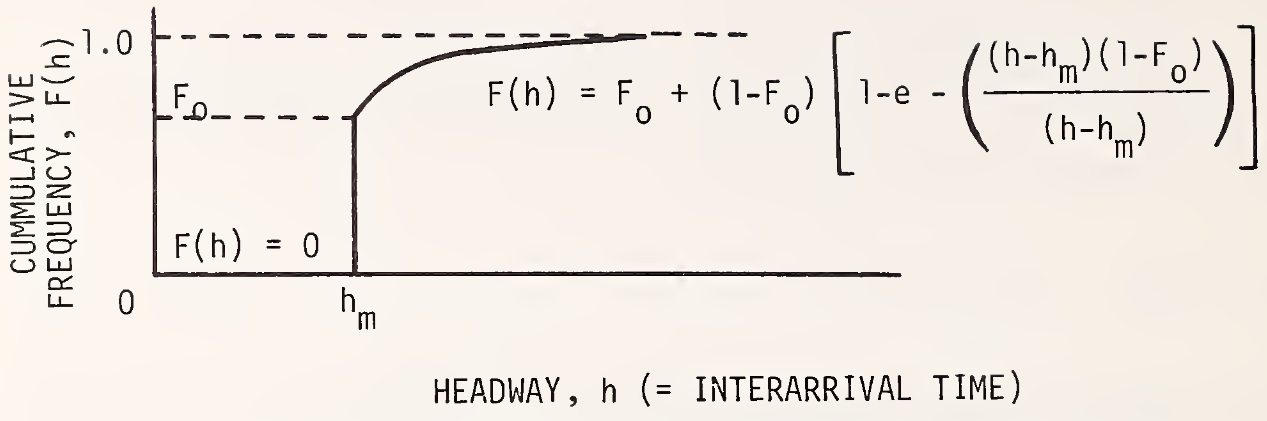
$$F(h) = 0, \; 0 \leq h < h_m$$

$$F(h) = F_0 - (1 - F_0)\left[1 - e - \left(\frac{(h - h_m)(1 - F_0)}{(h - h_m)}\right)\right] \;, \; h_m \leq h \; \infty \tag{2}$$

Where:

F(h) = probability that vehicle enters with headway $\leq$ h. Equation (1) is derived from (2) by substituting R for F(h) and solving for h.

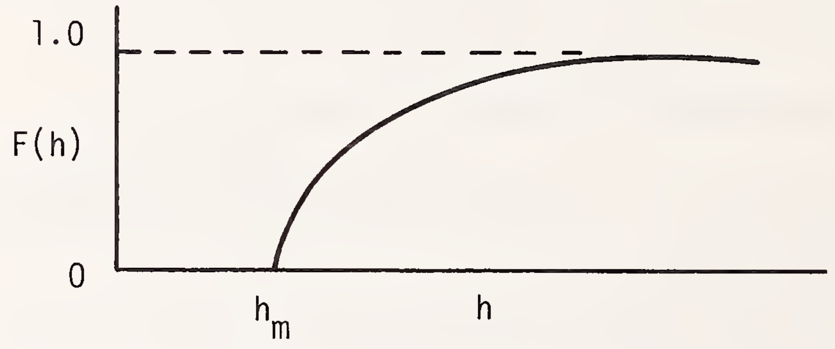
The graph of (2) has the form:



In the special case where $F_0 = 0$ (the fraction of vehicles arriving at exactly minimum spacing is zero), equation (2) reduces to

$$F(h) = 1 - e^{-\left(\frac{h - h_m}{\bar{h} - hm}\right)} \quad , h_m \leq h \infty \tag{3}$$
$$= 0 \quad , 0 \leq h < h_m$$

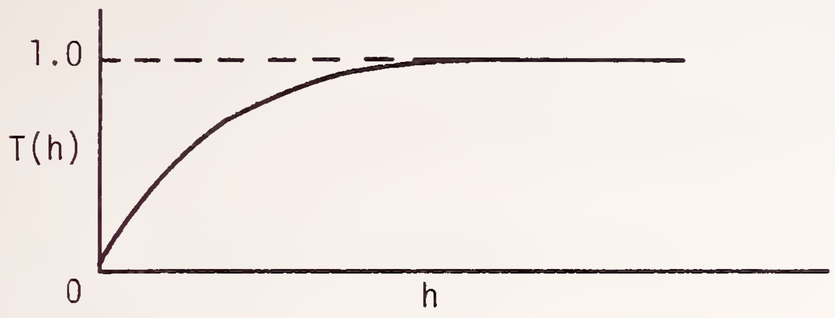
given in Martin and Whol Traffic System Analysis, page 507. This has the graph



In the special case where $h_m = 0$, equation (3) reduces to

$$F(h) = 1 - e^{-\frac{h}{\bar{h}}} \tag{4}$$

the standard exponential interarrival time distribution with graph



6.1.11 SIINIT

6.1.11.1 Identification

- o SIINIT - Input Initialization
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.11.2 Argument Dictionary - None.

6.1.11.3 Local Variable Dictionary - None.

6.1.11.4 Description - Depending on the nature of the parameter being initialized, SIINIT either clears the parameter or sets it to a default value. SIINIT also controls the processing that establishes the address and length of the System Characteristics commons by calling LODCOM which sets up a common (SIPSAV) containing the address and length and SIADDR which passes the address and length to SISADD where it is saved for the structured data file write.

6.1.11.5 PDL - See Appendix A.

6.1.11.6 Decision Tables and Algorithms - None.

6.1.12 SIMNAM

6.1.12.1 Identification

- o SIMNAM - Parameter List Scan
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.12.2 Argument Dictionary

PARAMETER	DIM	TYPE	DESCRIPTION
COUNT	-	I*2	Length of character string in PARM field of EXEC statement (Input)
STRING	COUNT	L*1	PARM field of EXEC statement (Input)

6.1.12.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
PTR	-	I*4	Index to STRING array

6.1.12.4 Description - The PARM field of the IP EXEC statement contains seven fields separated by commas. They are:

1. Module Name
2. System Characteristics input member name
3. System Characteristics member name
4. Runtime member name
5. Trip Demand member name
6. Vehicle Demand member name
7. Source of vehicle demand indicator.

SIMNAM separates the PARM field into seven fields by scanning the list for the field delimiter -- a comma. Each field is saved for later IP use in writing Run Index data.

6.1.12.5 PDL - See Appendix A.

6.1.12.6 Decision Tables and Algorithms - None.

6.1.13 SINERR

6.1.13.1 Identification

- o SINERR - Error Message Generation
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.13.2 Argument Dictionary

PARAMETER	DIM	TYPE	DESCRIPTION
IMSG	-	I*4	Message ID (Input)
ISEV	-	I*4	Message severity (Input) 1 = information 2 = warning 3 = severe (termination)

6.1.13.3 Local Variable Dictionary - None.

6.1.13.4 Description - SINERR calls subroutine ERROR to write the error message and if the severity code indicates, terminate the run.

6.1.13.5 PDL - See Appendix A.

6.1.13.6 Decision Tables and Algorithms - None.

6.1.14 SINPUT

6.1.14.1 Identification

- o SINPUT - Input Processor Control
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.14.2 Argument Dictionary - None.

6.1.14.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
AEND	-	I*4	'EOD' - Indicates end of GDIP input
ATYPES	15	I*4	Character fields representing all possible input data types. Used to check validity of a GDIP input type
DVRELS	-	I*4	Number of stations in a route (user input) (,1,)
IEND	-	I*4	Used in reading VEH demand data. PTR to end of 1 RTE's components in list of route lists (DVRSCH) (,1, KMRT)
IERR	-	I*4	If = 1, serious error found in trip or vehicle input data (not used)
INVAL	-	I*4	Temporary loc for maximum number of stations entered in VEH demand data (,1, KMS)

6.1.14.4 Description - After calling SIINIT to initialize system parameters, SINPUT reads System Characteristics input into the system common areas. Next, SINPUT reads the Runtime File decoding and processing each valid entry. Any nonzero time Runtime input is copied to the Runtime output file for the Model Processor. Any input not in time sequence causes termination.

SINPUT uses the runtime OPTION data to determine which of the main IP functions are required.

- o Trip demand generation
- o Vehicle demand generation
- o Model Processor setup.

If trip demand generation is requested, SINPUT reads the Trip Demand Input and Description File containing the trip demand generation data and calls SITDGN to generate the trips.

If vehicle demand generation is requested, SINPUT reads the Vehicle Demand Input and Description File containing the vehicle demand generation data and calls SIVDGN to generate vehicles.

If the Model Processor (MP) preparation is required, SINPUT calls several MP setup subroutines.

- o SISCFG to do station configuration
- o SICHCK to do parameter checking
- o SIREPT to write the Initial Conditions Report
- o SISWRT to write the Structured Data System Characteristics File.

SINPUT calls SIWNAM to list members in the index.

6.1.14.5 PDL - See Appendix A.

6.1.14.6 Decision Tables and Algorithms - None.

6.1.15 SIPARM

6.1.15.1 Identification

- o SIPARM - Parameter List Processor
- o IBM/FSD - July 1, 1977
- o Assembler H

6.1.15.2 Argument Dictionary

ENTRY	PARAMETER	DIM	TYPE	DESCRIPTION
SIPARM	PARMAD	-	A	Address of parameter list from EXEC statement (Input)
SIPLST	None			

6.1.15.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
PARMAD	-	A	Address of parameter list from EXEC statement
ARG1	-	A	Address of length of character string in PARM field of EXEC statement
ARG2	-	A	Address of character string in PARM field of EXEC statement

6.1.15.4 Description - SIPARM has two entry points each with a separate function.

1. SIPARM is the first program to receive control when the DSM IP is executed. It saves the address of the parameter list for future processing and gives control to SINPUT, the main IP processor.

2. SIPLST is called by SIINIT later in the IP. It passes the length and content of the parameter list to subroutine SIMNAM where the list is scanned and divided into member names of I/O files.

6.1.15.5 PDL - See Appendix A.

6.1.15.6 Decision Tables and Algorithms - None.

6.1.16 SIPSAV

6.1.16.1 Identification

- o SIPSAV - Input and System Characteristics Commons Address Save
- o IBM/FSD - July 1, 1977
- o Assembler H

6.1.16.2 Argument Dictionary - None.

6.1.16.3 Local Variable Dictionary - None.

6.1.16.4 Description - SIPSAV saves in a common (SIPSAV) the starting and ending addresses of the Input Processor Commons and the System Characteristics Commons generated at Link Edit time by the following overlay structure:

```
OVERLAY
  BEGCOM                                Start of IP Commons
OVERLAY
  Input Processor Commons
  .
  .
  .
OVERLAY                                 End of IP Commons
  IPSYS                                 Start of Sys Char Commons
OVERLAY
  System Characteristics Commons
  .
  .
  .
OVERLAY
  ENDCOM                                End of Sys Char Commons
```

The Linkage Editor generates the addresses (BEGCOM, IPSYS, ENDCOM) surrounding the two sets of commons. Entry point SIPSAV (equivalent to common SIPSAV defined in SIINIT) defines storage for the addresses; and entry point LODCOM causes SIPSAV (i.e., the addresses above) to be loaded into core. SIBWRT later uses IPSYS and ENDCOM to locate the System Characteristics Commons which it writes to the structured data file.

6.1.16.5 PDL - See Appendix A.

6.1.16.6 Decision Tables and Algorithms - None.

6.1.17 SIREPT

6.1.17.1 Identification

- o SIREPT - Initial Conditions Report
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.17.2 Argument Dictionary - None.

6.1.17.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
Station Link Types			
BL	-	I*4	Bypass station link (12)
D	-	I*4	Dock (3)
DS	-	I*4	Dock to storage (9)
IQ	-	I*4	Input queue (2)
IR	-	I*4	Input Ramp (1)
IS	-	I*4	Input to storage (7)
OQ	-	I*4	Output queue (4)
OR	-	I*4	Output ramp (5)
SI	-	I*4	Storage to input (8)
SO	-	I*4	Storage to output (10)
MOA	-	I*4	Modal output after processing (17)
MOB	-	I*4	Modal output before processing (16)
List of List Pointers			
DPTR	-	I*4	Index to downstream links
EPTR	-	I*4	Index to events
UPTR	-	I*4	Index to upstream links
IERR	-	I*4	If = 1, severe found. Terminate

Output Character Arrays

LKEVNT	-	I*4	Character representation of link events
LKTYPE	17	I*4	Character representation of station link types
TPLINK	9	I*4	Character representation of trip link types. 3 entries=1 trip link type
CYES	-	I*4	'YES' and 'NO' for link availability
CNO	-	I*4	
IAVAIL	-	I*4	CYES or CNO for link availability
OFFON	2	I*4	'ON' and 'OFF' for station type
LKORDR	2	I*4	'PRIO' and 'FIFO' for dequeue order
UDLNKS	17	I*4	Used to count occurrences of each link type
ERFLDS	13	I*4	List of fields in error in station link data

Page Control

LINREM	-	I*4	Lines remaining on page
LINE	-	I*4	Current line number within page
PAGE	-	I*4	Current page number

Time Conversions from Clock Units to Seconds

CVRSN	-	R*4	Used to convert CUs to secs (CU's/sec)
PTIMA	-	R*4	Used to hold trip link time in seconds from CUs
PTIMB	-	R*4	Same
TWALK	-	R*4	Same
PSECS	-	R*4	Used to hold DBD/BD times in seconds from CUs
SSECS	-	R*4	Same
ESECS	-	R*4	Same
LHDWYA	-	R*4	Used to hold link times in seconds from CUs
LHDWYB	-	R*4	Same
LTRAVL	-	R*4	Same
ISAMP	-	I*4	Used to hold sampling intvl in seconds from CUs
ICHK	-	I*4	Used to hold CKPT intvl in seconds from CUs

Miscellaneous indices where to Columns are Printed

C1LIM	-	I*4	Where 2 cols of data are printed, the limit of column 1
C2STR	-	I*4	Where 2 cols of data are printed, the starting index of column 2
RT1	-	I*4	Indices to route output using two columns
RT2	-	I*4	
DNLINK	-	I*4	Upstream link ID
UPLINK	-	I*4	Downstream link ID
DERR	-	I*4	Error return from SICUMP (0=no error)
DES1	-	I*4	Index to route assignment table
IADL	-	I*4	Index to events, upstream links, or downstream links
ISTR	-	I*4	Start and end Ptrs to a route list in
IEND	-	I*4	List of route lists array (PVRLST)

6.1.17.4 Description - SIREPT writes and validates each of the following DSM initial conditions:

1. Vehicle, trip, and train length capacities
2. Deboard/Board method and times
3. Vehicle launch delay
4. Static entrainment option
5. Vehicle sources
6. Service type
7. Empty vehicle and merge delay
8. Empty vehicle management
9. Vehicle headway and spacing by route
10. Route lists and route assignments by destination
11. Station link summary of upstream/downstream links, events, diverge functions, type, capacity, and travel time
12. Trip link summary
13. Simulation control summary.

If SIREPT finds errors in any of the initial conditions, it writes error messages within the report and terminates the run at the end of the report.

6.1.17.5 PDL - See Appendix A.

6.1.17.6 Decision Tables and Algorithms - None.

6.1.18 SISCFG

6.1.18.1 Identification

- o SISCFG - Station Configurator
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.18.2 Argument Dictionary - None.

6.1.18.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
Pointers to Fields in Input Array SLCFIG (13, KNL)			
LT	-	I*2	Link type fld (1)
TT	-	I*2	Link travel time fld (2)
LL	-	I*2	Link length fld (3)
CAP	-	I*2	Link capacity (VEH) fld (4)
EV1	-	I*2	Event 1 fld (5)
EV2	-	I*2	Event 2 fld (6)
EV3	-	I*2	Event 3 fld (7)
EV4	-	I*2	Event 4 fld (8)
EV5	-	I*2	Event 5 fld (9)
FN	-	I*2	Diverge function fld (10)
ORD	-	I*2	Upstream link ordering fld (11)
HT	-	I*2	Headway fld (12)
ET	-	I*2	Headway fld (13)
Event Types			
HEVENT	-	I*2	Headway event type (1)
TEVENT	-	I*2	Travel event type (2)
DEVENT	-	I*2	Deboard event type (3)
BEVENT	-	I*2	Board event type (4)
JEVENT	-	I*2	Joint board/deboard event type (5)
SEVENT	-	I*2	Storage event type (6)
LEVENT	-	I*2	Launch event type (7)

IDs of Certain Link Types Found

BLFLK	-	I*2	ID of bypass link
DBFLK	-	I*2	ID of board dock
DDBFLK	-	I*2	ID of deboard dock
DLFLK	-	I*2	ID of downstream station link
DSFLK	-	I*2	ID of dock to storage link
IQFLK	-	I*2	ID of input queue link
IRFLK	-	I*2	ID of input ramp link
ISFLK	-	I*2	ID of input to storage link
MIAFLK	-	I*2	ID of modal input after processing link
MIBFLK	-	I*2	ID of modal input before processing link
MOAFLK	-	I*2	ID of modal output after processing link
MOBFLK	-	I*2	ID of modal output before processing link
OQFLK	-	I*2	ID of output queue link
ORFLK	-	I*2	ID of output ramp link
STFLK	-	I*2	ID of storage link
SIFLK	-	I*2	ID of storage to input link
SOFLK	-	I*2	ID of storage to output link
ULFLK	-	I*2	ID of upstream station link

Count of Certain Link Types

IQCNT	-	I*2	Number of input queue links
DDBCNT	-	I*2	Number of deboard docks
DBCNT	-	I*2	Number of board docks
OQCNT	-	I*2	Number of output queue links
CCT	-	I*2	Pointer used to build list of lists tables
LINK	-	I*2	Link number
ORDSEL	-	I*2	Upstream link ordering option (1 of 6)
RTTIM	-	R*4	Link travel time (CUs)
RVEL	-	R*4	Link velocity (ft/sec)
SEQ	-	I*2	Used in upstream link ordering
SQGSM	18	I*2	Each 3 entries specifies 1 of 6 ways of ordering upstream links
USLOR	-	I*2	ID of link with launch event

6.1.18.4 Description - As a result of reading the system characteristics data, the station configuration parameters are read into an IP named-Common area SLCFIG. Each link definition prepared by the user consists of parameters describing the station link in terms of the following attributes:

Link Definition Attributes	Description
Repetition Factor	Standard GDIP field
Link Type	Numeric code for type link

Link Definition Attributes	Description
Link Travel Time	Time in seconds to travel link
Link Length	Length of link in feet to compute a travel line based on station velocity
Link Capacity	Capacity in vehicles per link
Link Events (five maximum)	Ordered events to occur on link
Link Diverge Function	Numeric code to select desired diverge function for link
Link Order for Dequeue	Ordering of link for dequeue numeric code to select order
Link Headway Time	Time in seconds to travel the headway zone
Link Headway Entrainment	Time factor in seconds per vehicle in train

The SLCFIG station link parameter values are used to build the structured data required by the DSM-MP. The processing performed and the resultant structured data prepared in the process is described in the following steps.

1. Establish the number of station links entered and set the runtime number of station links KNSL.

2. For each link travel time equal to 0, compute travel time from link length value x station velocity. When both travel time and link length are zero, then travel time is set to zero. Otherwise, all time values whether given or computed are converted into corresponding clock values prior to building the travel time table SLTTIM.

3. For each link, the capacity parameter value is used to build the capacity table SLCAP.

4. For each link, headway times are used to build the headway zone travel time tables SLHTA and SLHTB.

5. For each link, the link type value is used to build the link type table SLTYPE.

6. For each link, the link event value sequence is used to build the event sublist table and a sublist pointer table for each sublist in the sublist table SLEVL and SLEVP, respectively.

7. For each link, the link diverge function value is used to build the diverge function table SLDIVC.

8. For each link, the process computes and builds the system pointer value and upstream link ID for the link sublist in SLUSP and SLUSL, respectively. The link IDs are ordered by the value selection given in the ORDER parameter. There are six possible order combinations for the three link types (Main SL, Storage SL, and Modal SL) that occur upstream from another link. Upstream links are defined in Table 6-1.

9. For each link, the process computes and builds the downstream pointer and downstream link sublist SLDSP and SLDSL, respectively. Downstream links are defined in Table 6-1.

6.1.18.5 <u>PDL</u> - See Appendix A.

6.1.18.6 <u>Decision Tables and Algorithms</u> - Options for ordering of upstream links where

GW = Guideway link

ST = Storage link

MO = Modal link

OPTION	ORDER		
1	GW	ST	MO
2	ST	GW	MO
3	MO	GW	ST
4	GW	MO	ST
5	ST	MO	GW
6	MO	ST	GW

Table 6-1. Link Connectivity

Upstream Links	Link Type	Downstream Links
UL	IR	IS, MOB (IQ or DOCK)
SI, IR, MIB, or UL	IQ	DOCK (D)
IQ, or IR, SI, MIB, or UL	DOCK (D) only	DOCK (B)
DOCK (D)	DOCK (B) only	DS, MOA OQ or OR or DL
IQ or IR, SI, MIB, or UL	DOCK (D/B)	DS, MOA OQ or OR or DL
Dock (B) or Dock (D/B)	OQ	OR or DL
SO, MIA OQ or Dock	OR	DL
DS, IS	ST	SI, SO
IR	IS	ST
ST	SI	IQ or DOCK
Dock (B) or Dock (D/B)	DS	ST
ST	SO	OR
-1	UL	BL, IR or IQ or DOCK
UL	BL	DL
BL, OR or OQ or Dock	DL	-1
-2	MIB	IQ or Dock
-3	MIA	OR
IR	MOB	-2
Dock (B) or Dock (D/B)	MOA	-3

Mnemonic-Definition

IR -- Input Ramp	OQ -- Output Queue	DS -- Dock-to-Storage
IQ -- Input Queue	OR -- Output Ramp	SU -- Storage-to-output
Dock (D) -- Deboard	ST -- Storage	UL -- Upstream Station Link
Dock (B) -- Board	IS -- Input-to-Storage	BL -- Bypass Station Link
Dock (D/B) -- Deboard/ Board	SI -- Storage-to-Input	DL -- Downstream Station Link
MIB -- Modal Input Before	MIA -- Modal Input After	MOB -- Modal Output Before
MOA -- Modal Output After		

6.1.19 SITDGN

6.1.19.1 Identification

- o SITDGN - Trip Demand Generation
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.19.2 Argument Dictionary - None.

6.1.19.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
EXPN	-	I*4	Exponent, IAT character indic
USER	-	I*4	User IAT character indic
TYPE	-	I*4	Either exp or user IAT character indic
DIATMN	-	R*4	Computed mean of input user IAT dist.
DTERM	-	I*4	If = 1, serious error found, terminate
DTGNER	10	I*2	Indicators for 10 errors checked for in trip generation
DTIMNA	-	R*4	Actual mean of user IAT dist. Computed as trips are generated
DTPMNA	-	R*4	Actual mean trip size. Computed as trip gen'd
DTPREV	-	R*4	Time of previous trip
DTRPMN	-	R*4	Input mean trip size
DTSCNT	-	I*4	Total passengers generated
DTTCNT	-	I*4	Total trips generated
MEANTM	-	R*4	Actual exponential interarrival time mean
DTARUS	-	I*4	Start time

6.1.19.4 Description - After first verifying the input it has received from the Trip Demand File, SITDGN generates trips until the end time specified is reached. The components of each trip are generated as follows:

- o The trip arrival time equals the time of previous arrival plus interarrival time where the interarrival time is randomly chosen from a user defined distribution or from an exponential distribution with a user defined mean (interarrival time = mean interarrival time x log (random number between 0-1). Initially the previous arrival time is set to the start time.

- o The trip's origin is the simulated station.

- o The trip's destination is chosen from a user defined destination distribution.
- o The number of passengers in the trip is chosen from a user defined trip size distribution.

Each trip is written to the Structured Data Trip Sequence File.

At the completion of trip generation, SITDGN writes a trip summary report.

6.1.19.5 PDL - See Appendix A.

6.1.19.6 Decision Tables and Algorithms - SITDGN uses the following algorithm to generate trip arrival times.

Exponential arrival rate

ARIV = PREV + (-MEAN(ln(RAND)))

where

ARIV	=	Arrival time
PREV	=	Time of previous arrival
MEAN	=	Mean interarrival time (user input)
RAND	=	Random number between 0 and 1

User arrival rate

ARIV = PREV + DTIATD (RAND,y)

where

ARIV	=	See above
PREV	=	See above
RAND	=	See above
DTIATD (x, y)	=	User distribution where x is a cumulative probability and y is an interarrival time

Trip destination is chosen from a cumulative probability distribution.

Destination Dist.	Meaning
DTDESD(1)	P (destination is station 1)
DTDESD(2)	P (destination is station 2 or 1)
.	
.	
.	
DTDESD (KNS)	P (destination is station KNS, ...2, 1)

SITDGN chooses a random probability between 0 and 1 and finds the first entry in DTDESD whose value is greater than or equal to the random probability. The entry number (1 to KNS) is the destination.

Trip size is also chosen from a cumulative probability distribution, DTPASD.

6.1.20 SIVDGN

6.1.20.1 Identification

- o SIVDGN - Vehicle Demand Generation
- o IBM/FSD - July 1, 1977
- o FORTRAN IV (H Extended) with PARAFOR

6.1.20.2 Argument Dictionary - None.

6.1.20.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
CEXP	-	I*4	Exponential IAT character indicator
CUSER	-	I*4	User IAT character indicator
CIATYP	-	I*4	Either exponential or user IAT character indi
DERR	-	I*4	If = 1, an error has been found
DIATMN	-	R*4	Input mean interarrival time
DMEAN	-	R*4	Actual mean interarrival time computed as vehicles are generated
DNXRTE	-	I*4	Next arrival time on current route
DNXTIM	-	R*4	Time of next vehicle arrival
DPASMN	-	R*4	Input mean no. passengers/trip
DPSMNA	-	R*4	Actual mean no. passengers/trip. Computed as trips are generated
DSTP1	-	I*4	% of trains stopping at simulated station
DSTP2	-	I*4	% of vehicles stopping at simulated station
DSTP3	-	I*4	% of trains not stopping at simulated station
DSTP4	-	I*4	% of vehicles not stopping at simulated station
DTERM	-	I*4	If = 1, a serious error has been found. Terminate
DVGNER	15	I*2	Indicators for 15 input error types checked in vehicle generation
DVPREV	KMR	R*4	Previous vehicle arrival time/route
DVTMNA	KMR	R*4	Current IAT mean/rte.computed as vehicles are generated for output purpose
DTPMNA	-	R*4	Actual no. trips/vehicle. Computed as trips and vehicles are generated
DTRMNA	-	R*4	Actual no. vehicles/train. Computed as vehicles and trains are generated

DTRPMN	-	R*4	Input no. trips/veh mean
DVEHMN	-	R*4	Input no. vehicles/train mean.
IRTEND	-	I*4	End of route list of lists
IVEH	-	I*4	Index no. for vehs/train
PASTOT	-	I*4	Total passengers stopping at sim'd station
TR1TOT	-	I*4	Total trains stopping at sim'd station
TR2TOT	-	I*4	Total trains not stopping at sim'd station
TRPTOT	-	I*4	Total trips stopping at sim'd station
VH1TOT	-	I*4	Total vehicles stopping at sim'd station
VH2TOT	-	I*4	Total vehicles not stopping at sim'd station
DVARVS	-	I*4	Start time

6.1.20.4 <u>Description</u> - After verifying/initializing the input parameters received from the vehicle demand file, SIVDGN generates vehicles and onboard trips until the arrival time of the next vehicle is greater than the end generation time. This process starts with a user-specified start time.

If the service policy is scheduled SIVDGN does the following:

1. Select the route with the next arrival time.

2. Get train length for chosen route.

3. Set the next stop to the simulated station or not after examining a route next stop indicator or scanning all the route stops for the simulated station.

4. Call subroutine SIGIAT to determine the next arrival time for the chosen route.

If the service policy is demand responsive, SIVDGN does the following:

1. Select train length from probability distribution.

2. Use next stop probability to determine if train should stop at simulated station.

3. Call subroutine SIGIAT to determine arrival time of next vehicle.

For all vehicles SIVDGN chooses a sink from the user sink distribution.

If the train is to stop at the simulated station, SIVDGN generates onboard trips (0 or more), choosing the maximum number of trips and number of passengers per trip from user defined probability distributions. Trips are generated until either the vehicle capacity is reached or the maximum number of trips is generated.

For each vehicle generated a vehicle record and for each onboard trip a trip record are written to the Structured Data Vehicle Arrival File.

When all vehicles have been generated, SIVDGN writes a vehicle summary report.

6.1.20.5 PDL - See Appendix A.

6.1.20.6 Decision Tables and Algorithms - See SIGIAT (subsection 6.9) for a discussion of interarrival time computation.

SIVDGN chooses several trip/vehicle characteristics from cumulative probability distributions.

Trip destination	DVDESD (KNS)
Trip size	DVPASD (KNNP)
Sink	DVSNKD (3)
Train length (demand)	DVTLND (KNTLEN)
Trips/veh	DVTRPD (KNNT)

The following illustrates the process for trip size selection.

Distribution	Meaning
DVPASD(1)	P (trip = 1 passenger)
DVPASD(2)	P (trip $\leq$ 2 passengers)
.	
.	
.	
DVPASD (KNNP)	P (trip $\leq$ KNNP passengers)

SIVDGN chooses a random probability between 0 and 1 and finds the first distribution entry whose contents is greater than or equal to the random probability. The entry number chosen is the number of passengers/trip. The only exception to this process in the selection of trips/vehicle where the selection of entry 1 means 0 trips, entry 2 means 1 trip, entry 3 means 2 trips, etc.

6.1.21 SMRNG - See subsection 6.2.55, SMRNG.

6.1.22 SMRSEL - See subsection 6.2.56, SMRSEL.

6.1.23 TIMES - See subsection 6.2.6, DTIMEL.

6.2 MODEL PROCESSOR

This section contains the subprogram descriptions for the DSM-Model Processor.

6.2.1 DAYTIM

6.2.1.1 Identification

- o DAYTIM - Convert Date and Time
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.1.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
MM	–	I*2	(OUTPUT) MONTH
DD	–	I*2	(OUTPUT) DAY
YY	–	I*2	(OUTPUT) YEAR
HH	–	I*2	(OUTPUT) HOURS
MM	–	I*2	(OUTPUT) MINUTES
SS	–	I*2	(OUTPUT) SECONDS

6.2.1.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
YEAR	2	I*4	Century and year of century
HMS	3	I*4	Hours, minutes, seconds
SS	-	I*2	Seconds
LEAP	-	I*4	Indicates leap year

6.2.1.4 Description - The purpose of DAYTIM is to get Julian date and time from system clock and return calendar date and time. DAYTIM first calls DTMEL via entry point TIMES to get the Julian date and time from the system clock. The returned year is then tested for leap year with the MOD function to determine which calendar routine to use. The calendar routine then uses the day of the year to find the month of the year and the day of the month.

6.2.1.5 PDL - See Appendix A.

6.2.1.6 Decision Tables and Algorithms - None.

6.2.2 DBUG

6.2.2.1 Identification

- o DBUG - Intermediate Output Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.2.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
ID	-	C	MAXIMUM 31 CHARACTERS OF TEXT TO BE OUTPUT IN DBUG MESSAGE. ENCLOSED IN SINGLE QUOTES WHEN IMBEDDED BLANKS ARE USED.
FLAG	-	I*4	ARRAY SUBSCRIPT FOR A LOGICAL VARIABLE TESTED DURING EXECUTION TO DETERMINE IF MESSAGE DISPLAY IS REQUIRED. AFLAG MUST LIST THIS FLAG DURING EXECUTION.
C1	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C2	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C3	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C4	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C5	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C6	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C7	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C8	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C9	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)
C10	-	C	VARIABLE NAME TO BE DISPLAYED WITH VALUE WHEN DBUG MESSAGE IS ISSUED. MUST BE FULLY QUALIFIED. (OPTIONAL)

6.2.2.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUTA	-	C	Constructed FORTRAN code
OUTB	-	C	Constructed FORTRAN code
FMT	-	C	Format statement number

6.2.2.4 Description - The purpose of DBUG is to provide a trace facility within the DSM simulator. This macro generates IF, WRITE, and FORMAT statements. When the flag is turned on at execution time, the write statement is executed and prints the first six characters of the variable and its value for as many as ten variables in addition to a message.

6.2.2.5 PDL - See Appendix A.

6.2.2.6 Decision Tables and Algorithms - None.

6.2.3 DQUE

6.2.3.1 Identification

- o DQUE - Dequeue Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.3.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	-	C*1	ENTITY TYPE TO BE DEQUEUED (X,V,T)
HEAD	-	C	QUEUE LIST HEAD/TAIL WORD MUST BE FULLY QUALIFIED
INDEX	-	C	(INPUT) VARIABLE NAME TO BE ASSIGNED TO ENTITY REMOVED FROM THE QUEUE LIST. (OUTPUT) NON-ZERO ENTITY NUMBER OF THE XTN REMOVED OR ZERO IF THE QUEUE WAS EMPTY.
YCHAIN	-	C	TRANSACTION CHAIN WORD (OPTIONAL; DEFAULT QUECH/FELCH)

6.2.3.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
XCHAIN	-	C	'QUECH' default string

6.2.3.4 Description - The purpose of DQUE is to remove an entity from FIFO or LIFO queue. This macro generates code which when executed takes the first entity off a queue. If the queue was specified as being FIFO in NQUE, then the entity queued for the longest time (the one at the head of the link) is removed. Otherwise, the entity queued for the shortest time is removed. The chain is then closed and the head is set to the next entity. If no entities remain in the chain, the head is set to zero.

6.2.3.5 PDL - See Appendix A.

6.2.3.6 Decision Tables and Algorithms - None.

6.2.4 DQUEM

6.2.4.1 Identification

- o DQUEM - Dequeue a Particular Entity for Anywhere in a Queue Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.4.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	—	C*1	ENTITY TYPE TO BE DEQUEUED (X,V,T)
HEAD	—	C	QUEUE LIST HEAD/TAIL WORD MUST BE FULLY QUALIFIED
INDEX	—	C	(INPUT) VARIABLE NAME OF ENTITY TO BE REMOVED FROM THE QUEUE LIST. (OUTPUT) VALUE GREATER THAN 0 IF XTN WAS NOT ROUND OR 0 IF INDEX WAS FOUND.
YCHAIN	—	C	TRANSACTION CHAIN WORD (OPTIONAL; DEFAULT QUECH/FELCH)

6.2.4.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
XCHAIN	-	C	'QUECH' default string
LOC1	-	C	Pointer to head of queue
LOC2	-	C	Chainword

6.2.4.4 Description - The purpose of DQUEM is to remove a given entity from anywhere in a queue. This macro generates code which when executed steps through the queue until the desired entity is found, removes it from the queue, and repairs the chain.

6.2.4.5 PDL - See Appendix A.

6.2.4.6 Decision Tables and Algorithms - None.

6.2.5 DQUEMID

6.2.5.1 Identification

- o DQUEMID - Remove a Specific Entity from a Queue Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.5.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	-	C*1	ENTITY TYPE TO BE DEQUEUED (X,V,T)
TAIL	-	C	QUEUE LIST HEAD/TAIL WORD MUST BE FULLY QUALIFIED
LOOPVAR	-	C	VARIABLE NAME OF THE ENTITY IN THE QUEUE POINTED TO BY "QLOOP"
PRED	-	C	VARIABLE NAME OF THE ENTITY WHOSE CHAIN WORD POINTS TO LOOPVAR
CHAINWD	-	C	TRANSACTION CHAIN WORD (OPTIONAL; DEFAULT XQUECH/VQUECH/TQUECH)

6.2.5.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer

6.2.5.4 Description - The purpose of DQUEMID is to remove a specific entity from a queue in conjunction with the QLOOP macro. This macro generates code which when executed removes the currently active entity in a QLOOP operation from its chain and repairs the chain. DQUEMID can only be used inside a QLOOP/ENDQLOOP code segment with other code that is to be performed on each entity in the queue and so is able to allow QLOOP to effectively locate the entity.

6.2.5.5 PDL - See Appendix A.

6.2.5.6 Decision Tables and Algorithms - None.

6.2.6 DTIMEL

6.2.6.1 Identification

- o DTIMEL - Read System Clock for Date and Time
- o IBM/FSD - July 1, 1977
- o ASM

6.2.6.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TIMES:			
YEAR	2	I*4	(OUTPUT) YEAR, JULIAN DAY
HMS	3	I*4	(OUTPUT) HOURS, MINUTES, AND SECONDS
SEC	—	I*4	(OUTPUT) TIME OF DAY IN SECONDS
DELT	—	I*4	(OUTPUT) ELAPSED TIME SINCE LAST CALL TO TIMES (IN SECONDS)

6.2.6.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TA	-	I*4	Seconds of the day
DBL	2	I*4	Packed decimal rate and time
YIM	2	I*4	Century any year of century
HIM	3	I*4	Hours, minutes, seconds

6.2.6.4 Description - The purpose of DTIMEL is to get the Julian date and time from the system clock. DTIMEL is called by DAYTIM to read the system clock and return the current date and time. DTIMEL calls the system TIME macro to get the date and time in EBCDIC. The routine then converts the date and time to binary and returns to the calling program.

6.2.6.5 PDL - See Appendix A.

6.2.6.6 Decision Tables and Algorithms - None.

6.2.7 ENDQLOOP

6.2.7.1 Identification

- o ENDQLOOP - Terminate a QLOOP Code Segment Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.7.2 Argument Dictionary - None.

6.2.7.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer

6.2.7.4 Description - The purpose of ENDQLOOP is to terminate the code segment of a QLOOP. This macro generates @ENDIF and @ENDDO statements which close-off corresponding @IF and @DOWHILE statements generated by the QLOOP macro.

6.2.7.5 PDL - See Appendix A.

6.2.7.6 Decision Tables and Algorithms - None.

6.2.8 FREE

6.2.8.1 Identification

- o FREE - Return an Entity to an Available List Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.8.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	-	C*1	ENTITY TYPE TO BE FREED (X,V,T)
INDEX	-	C	VARIABLE NAME OF ENTITY TO BE RETURNED TO THE LIST OF AVAILABLE TRANSACTIONS.
XLIST	-	C	HEAD TO THE AVAILABLE LIST (OPTIONAL; DEFAULT XAVAIL/VAVAIL/TAVAIL)

6.2.8.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
LIST	-	C	'AVAIL' default string
TYPE1	-	C	Null string

6.2.8.4 Description - The purpose of FREE is to return an entity to the available list of corresponding entities. This macro generates code which when executed checks to see if the entity is in a chain and if so, generates an error message. Otherwise, it changes the entity's chain word to point to the current top of the available transactions list and changes the list head to point to it.

6.2.8.5 PDL - See Appendix A.

6.2.8.6 Decision Tables and Algorithms - None.

6.2.9 GET

6.2.9.1 Identification

- o GET - Remove an Entity from the Available List Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.9.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	–	C*1	ENTITY TYPE TO BE REQUESTED (X,V,T)
INDEX	–	C	VARIABLE NAME OF ENTITY TO BE GOTTEN FROM THE LIST OF AVAILABLE TRANSACTIONS.
XLIST	–	C	HEAD TO THE AVAILABLE LIST (OPTIONAL; DEFAULT XAVAIL/VAVAIL/TAVAIL)

6.2.9.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
LIST	-	C	'AVAIL' default string
TYPE1	-	C	Null string

6.2.9.4 Description - The purpose of GET is to remove an entity from the available list. This macro generates code which when executed checks to see if there are any more entities in the available list. If not, an error message is generated. If so, the code changes the list head to the value of the current list head, sets the chainword of this now-old top entity to zero, and returns its ID.

6.2.9.5 PDL - See Appendix A.

6.2.9.6 Decision Tables and Algorithms - None.

6.2.10 MULTICK

6.2.10.1 Identification

- o MULTICK - Test if Currently Enqueued Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.10.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	-	C*1	ENTITY TYPE TO BE REQUESTED (X,V,T)
INDEX	-	C	VARIABLE NAME OF ENTITY
XCHAIN	-	C	ENTITY CHAIN WORD
NAME		C	TEXT DESIGNATION OF CHAIN TO APPEAR IN ERROR MESSAGE (OPTIONAL)

6.2.10.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code

6.2.10.4 Description - This macro is used by other macros to ensure that an entity is queued in only one list at a time. It generates code which when executed tests the chainword of the entity. If it is non-zero, an error message is generated.

6.2.10.5 PDL - See Appendix A.

6.2.10.6 Decision Tables and Algorithms - None.

6.2.11 NQUE

6.2.11.1 Identification

- o NQUE - Place an Entity into a Queue Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.11.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	—	C*1	ENTITY TYPE TO BE ENQUEUED (X,V,T)
HEAD	—	C	QUEUE LIST HEAD/TAIL WORD MUST BE FULLY QUALIFIED
INDEX	—	C	ENTITY ID OF ENTITY TO BE ENQUEUED REMOVED OR ZERO IF THE QUEUE WAS EMPTY.
LIFO	—	I*4	TYPE OF ENQUEUE REQUIRED: NUMBER GREATER THAN ZERO = LIFO NULL = FIFO
YCHAIN	—	C	TRANSACTION CHAIN WORD (OPTIONAL; DEFAULT QUECH/FELCH)

6.2.11.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
XCHAIN	-	C	'QUECH' default string

6.2.11.4 Description - The purpose of NQUE is to place an entity into a queue. This macro generates code which first uses MULTICK to determine if the entity is already enqueued. If not, it then proceeds to alter chainwords and the queue head in order to LIFO/FIFO enqueue the entity.

6.2.11.5 PDL - See Appendix A.

6.2.11.6 Decision Tables and Algorithms - None.

6.2.12 QLOOP

6.2.12.1 Identification

- o QLOOP - Loop Through a Queue Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.12.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	–	C*1	ENTITY TYPE IN QUEUE (X,V,T)
TAIL	–	C	QUEUE LIST HEAD/TAIL WORD MUST BE FULLY QUALIFIED
LOOPVAR	–	C	(INPUT) VARIABLE NAME TO BE ASSIGNED TO AN ENTITY. (OUTPUT) THE ENTITY IN THE QUEUE THAT WAS ADVANCED TO BY QLOOP.
PRED	–	C	(INPUT) VARIABLE NAME TO BE ASSIGNED TO THE ENTITY POINTING TO "LOOPVAR."(OPTIONAL) (OUTPUT) THE ENTITY IN THE QUEUE WHOSE CHAIN WORD POINTS TO THE ENTITY IN "LOOPVAR."
CHAINWD	–	C	TRANSACTION CHAIN WORD (OPTIONAL; DEFAULT IS QUECH/FELCH)
FINI	–	L*1	VARIABLE NAME OF A LOGICAL VARIABLE THAT CAN BE SET WITHIN THE QLOOP/ENDQLOOP PROCESSING TO CAUSE THE LOOP PROCESSING TO TERMINATE AT ENDQLOOP IF THE VALUE IS TRUE. (IT IS INITIAL-IZED TO FALSE BY THE MACRO AND IS OPTIONAL.)

6.2.12.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
FINI1	-	C	Constructed FORTRAN code
FINI2	-	c	Constructed FORTRAN code
LOC1	-	C	Queue head
LOC2	-	C	Queue chainword

6.2.12.4 Description - The purpose of QLOOP is to loop through a queue, in order from head (longest waiting entity) to tail, performing a code segment on every entity in the queue. This macro generates code which when executed together with ENDQLOOP allows a code segment located between these two macro names to be performed on every entity in the queue. It effectively returns the ID of every entity in the queue to the code segment. DQUEMID can be in this code segment to allow any entity to be removed. An early exit flag can be set to true in the code segment to immediately drop through the loop.

6.2.12.5 PDL - See Appendix A.

6.2.12.6 Decision Tables and Algorithms - None.

6.2.13 SAASYN

6.2.13.1 Identification

- o SAASYN - Process Asynchronous Commands
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.13.2 Argument Dictionary - None.

6.2.13.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NAMES	12	I*4	List of legal HEADER CARD names
TEXT	18	I*4	Holds 72 characters of data from TEXT FOLLOWER CARD
END	-	I*4	Holds keyword END searched for on cards
FIND	-	L*1	Indicates legal header name found
SKIP	-	L*1	Indicates next card read to be skipped since EOD or STOP CARD found
AEOF	-	L*1	End of file on asynchronous card file
CU	-	I*4	Clock in seconds

6.2.13.4 Description - The purpose of SAASYN is to perform the processing required to input the asynchronous data. This data can be commands which cause status changes within the simulation system or change specifications which modify the value of system parameters and data. It takes the form of a CASE block where the data item that determines the case to be performed is the event type of the transaction XMEVNT (XACTIV). XMEVNT is analogous to VMEVNT and TMEVNT in that they are all "subvent types" of the system level events used at the SAMAIN level. The asynchronous data associated with the event is processed according to the header card which initially caused the scheduling of the asynchronous event as follows:

1. DATA/OPTION/PARAM/SELECT Header Cards initiate successive data change requests to update the global data variables and parameters as required. Reading is done using SMGDIP4 (GDIP).

2. FAIL header cards initiate the reading of failed selected data with GDIP and the calling of SAFAIL to perform failure related processing.

3. FLAG header cards initiate intermediate debug output used in program maintenance. SAFLAG sets the flags based on information provided on follower cards.

4. TEXT header card initiates the writing of one line of text from one follower card to the system output device.

5. CKPT header card initiates the writing of one checkpoint record on demand using SACKR.

6. EOF or STOP header cards terminate the simulation by causing the asynchronous data read transaction to be scheduled as the termination transaction.

7. TRIP header cards cause a number of trips defined on follower cards to be read.

8. VEH header cards cause a vehicle and onboard trips as defined in follower cards to be read in as a vehicle arriving on the guideway.

9. INDEX header cards initiate the reading of index followers and the writing of them to the index file until an END card is encountered.

6.2.13.5 <u>PDL</u> - See Appendix A.

6.2.13.6 <u>Decision Tables and Algorithms</u> - None.

6.2.14 SACKR

6.2.14.1 Identification

- o SACKR - Checkpoint and Restart Processing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.14.2 Argument Dictionary

```
|-----------------------------------------------------------------------|
|VARIABLE  | DIM  | TYPE  | DESCRIPTION                                 |
|-----------------------------------------------------------------------|
SACKR:
   CAREA      -      I*4     STARTING ADDRESS OF COMMONS
   LEN        -      I*4     LENGTH OF THE COMMONS
SACKPT:   NONE
SAREST:
   ZTIME      -      R*4     TIME OF REQUESTED RESTART
```

6.2.14.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NAMES	12	I*4	List of legal header card names
END	-	I*4	Holds keyword END searched for on cards
FIND	-	L*1	Indicates legal header name found
AEOF	-	L*1	End of file on asynchronous card file
TEOF	-	L*1	End of file on trip file
VEOF	3	L*1	End of file on vehicle files
CEOF	-	L*1	End of file on checkpoint file
TTTIME	-	I*4	Requested restart time in CU

6.2.14.4 Description - Checkpointing is performed to save the status of a simulation experiment at any point during the simulation run. Checkpointing can occur at periodic intervals or via an asynchronous data request or at failure. The checkpoint data can be used to restart the simulation by reinitialization of system status as saved by the checkpoint. This code segment contains entry points to perform both checkpointing

(SACKPT) and restart (SAREST). The writing of system status during a checkpoint involves a sequential binary write of core storage beginning at the symbolic address defining the beginning of global common data for the simulation system. The address of this area and its length are defined to checkpoint processing during initialization. This is accomplished by causing definition of the checkpoint area and its length to be established by issuing a call to the checkpoint I/O routine (SANTSA) during initialization. The actual checkpoint is performed by an entry point (SACKPT) defined in the checkpoint I/O routine to which control is transferred when a checkpoint is required.

Restart is performed by reading the checkpoint file until a record is read with a clock value equal to that requested. When this record is found, it is used to reposition the run time, trip, and vehicle files. This process essentially takes the place of initialization. Control then returns to the main routine (SAMAIN) when processing will continue.

6.2.14.5 PDL - See Appendix A.

6.2.14.6 Decision Tables and Algorithms - None.

6.2.15 SACOMN

6.2.15.1 Identification

- o SACOMN - Input Common Area Sequencing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.15.2 Argument Dictionary - None.

6.2.15.3 Local Variable Dictionary - None.

6.2.15.4 Description - This routine is used by both the IP and MP to force an identical ordering of input area commons. This is done by including it as the first object module at link edit time in both the IP and MP that contains the input commons. This ordering is necessary to ensure that the ordering of these commons in the IP from which AGT.STRUC.SYSTEM is the same as that in the MP into which they are read. This is a linkage edit time device.

6.2.15.5 PDL - See Appendix A.

6.2.15.6 Decision Tables and Algorithms - None.

6.2.16 SADADD

6.2.16.1 Identification

- o SADADD - Initialize Input Area Addresses and Message Common
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.16.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SADADD:			
IPAREA	–	I*4	STARTING ADDRESS OF INPUT COMMONS
LEN2	–	I*4	LENGTH OF INPUT COMMON
SANDTA: NONE			

6.2.16.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
EOF	-	L*1	End of file on system characteristics file

6.2.16.4 Description - The purpose of SADADD is to initialize input area address and message common (SCAMSG). SADADD is first called from SANSAV to inform the routine of the starting address of the input commons and the length of this area. These two factors are then used by the routine when its entry point SANDTA is called from SAINIT to actually perform the reading. The variables of SCAMSG are also initialized at this time.

6.2.16.5 PDL - See Appendix A.

6.2.16.6 Decision Tables and Algorithms - None.

6.2.17 SAFAIL

6.2.17.1 Identification

- o SAFAIL - Failure, Degradation, and Recovery Processing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.17.2 Argument Dictionary - None.

6.2.17.3 Local Variable Dictionary - None.

6.2.17.4 Description - This routine serves to set and reset variables associated with failures, recoveries, degradation, and degradation recoveries and take checkpoints of failure. When this routine is called, GDIP formatted data containing the input data has already been read. The various activities are modeled as follows:

1. Station Link Entry Failure -- A flag is set for the subject link that will indicate to other parts of the simulation that the link cannot be entered. A checkpoint is taken.

2. Station Link Exit Failure -- A flag is set for the subject link to indicate to other parts of the simulation that the link cannot be exited. A checkpoint is taken.

3. Station Link Entry Recovery -- The flag is set at failure time is turned off and an upstream prompt is done to attempt to get waiting vehicles into the link.

4. Station Link Exit Recovery -- The flag is set at failure is turned off and a self prompt is done to attempt to get waiting vehicles off the subject link.

5. Station Link Degradation -- The degradation factor, SPLENT, has already been read from the run time file (it typically has been set greater than one by the user) and will be used from then on as a multiplicative factor in the travel event time calculation.

6. Station Link Degration Recovery -- The degration factor has already been read from the run time file (it typically has been set equal to one by ther user) and will be used from then on in the travel event time calculation.

7. Trip Link Failure -- The number of servers on the subject link is set equal to zero and used fron then on.

8. Trip Link Recovery -- The number of servers has already been read in from the run time file (it typically is the full number of servers) and will be used from then on. A prompt is done to get trips moving again off the subject link.

9. Trip Link Degradation -- As opposed to Trip Link Failure, the number of servers has already been read in from the run time file (it typically is less than the full number of servers and must be greater than zero) and is used from then on.

10. Trip Link Degration Recovery -- As opposed to Trip Link Recovery, the number of servers has already been read in from the run time file (it typically is equal to the full number of servers) and is used from then on. No prompting is done.

6.2.17.5 PDL - See Appendix A.

6.2.17.6 Decision Tables and Algorithms - None.

6.2.18 SAFINM

6.2.18.1 Identification

- o SAFINM - Snapshot and Final Model Report
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.18.2 Argument Dictionary - None.

6.2.18.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	20	I*4	Abbreviations for SL and TL types for reports
HEAD1	5,10	I*4	Report line headings
TIME	-	R*4	Current SIM time in seconds
KNTL	-	I*4	Number of trip links

6.2.18.4 Description - The purpose of this routine is to write a report that itemizes statistics collected since the last sample. This routine simply prints out the statistics from the statistics common with appropriate headings.

6.2.18.5 PDL - See Appendix A.

6.2.18.6 Decision Tables and Algorithms - None.

6.2.19 SAFINS

6.2.19.1 Identification

- o SAFINS - Final System Report
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.19.2 Argument Dictionary - None.

6.2.19.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
A	-	R*4	Percentage of total count
INFINY	-	R*4	Largest integer*4 possible
J	-	I*4	Edge of histogram cells

6.2.19.4 Description

The purpose of this routine is to provide event statistics. The event statistics reflect usage demands placed on the event scheduling mechanism of the simulator as defined by user input data for establishing a clock table and multiple thread list definitions. These statistics provide the basis upon which more efficient definitions of the FEL can be based in future simulation experiments to increase the run time speed.

The report is based on the following analysis. The two primary measures of simulator operation are:

M_1 = CPU time/SIM time $\sim$ RUNTIME

and

M_2 = No. of Events Processed/CPU Time = SIMULATOR EFFICIENCY

The complementary measure is:

M_3 = No. of Events Processed/SIM Time = DENSITY OF FEL (Future Event List)

Note:

$$M_1 = M_3/M_2$$

$$M_2 = M_3/M_1$$

$$M_3 = M_1M_2$$

M_1 can be reduced by decreasing M_3 or increasing M_2.

M_3 can be decreased by:

1. Decreasing the size of the model (i.e., number of traffic units and size of traffic network); and

2. Decreasing the number of events which traffic units must encounter.

M_2 can be increased by:

3. Improving the efficiency of the event program code; and

4. Improving the efficiency of the Future Event List (FEL) code.

For any given model run 1, 2, and 3 are fixed and only 4 is open to improvement by varying a parameter called CLBIG. (CLSMAL = CLBIG/1000, by definition.) The remainder of this section is devoted to optimizing CLBIG, which in turn for any given model run will optimize M_1 and M_2.

The current structure of the Future Event List (FEL) is such that is the Δt of a TXN is less than CLBIG, then the TXN is put into a clock table (C/T) while if $\Delta t >$ CLBIG, the TXN is put into a multiple thread chain (M/T). That is,

If	Then TXN Put Into
$\Delta t <$ CLBIG	C/T
$\Delta t >$ CLBIG	M/T

This is shown in Figure 6-1.

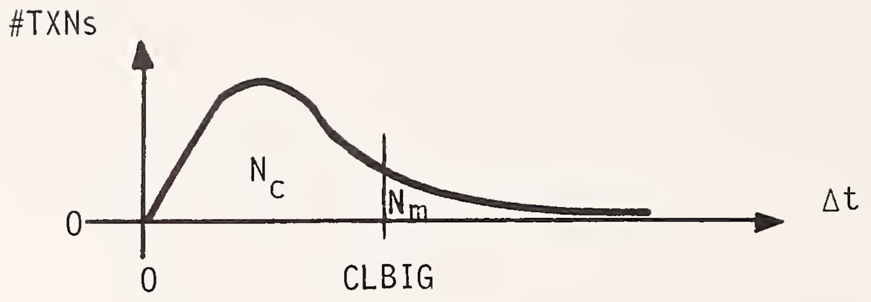


Figure 6-1. Distribution of Number of Transactions versus Their Delta Time

Thus, CLBIG forms a cutoff determining how many transactions are processed by the clock table mechanism versus the number acted on by the M/T mechanism.

Let:

N_c = Number of TXNs processed via C/T

N_m = Number of TXNs processed via C/T

$N = N_c + N_m$ = Number of TXNs processed from the Future Events List (FEL)

$R = N_c/N$

W_c = Work (Number of instructions to be executed or execution time) required to put a TXN in C/T and remove it for processing.

W_m = Work required to put a TXN in M/T and remove it for processing.

Objective: If too many TXNs are put into the C/T, then the average number of TXNs per C/T entry will increase to a point where too much work is being done in managing the TXNs in C/T entries. On the other hand, if too many TXNs are put in the M/T, then too much work will be done searching through M/T TXNs and the direct indexing facility of the C/T will be under-utilized. So the objective is to find a balance between N_c and N_m so as to minimize the expected amount of work associated with processing a TXN. This expected amount of work is given by the following weighted average:

$$W = \frac{N_c W_c + N_m W_m}{N}$$

Now, note that W_c is itself a function of N_c, since a larger N_c implies a larger average number of TXNs per C/T entry which implies a longer search during insertion. A linear function seems to be a reasonable approximation:

$$W_c = a + bN_c$$

Where

> a = Amount of work (execution time) required to put a TXN in C/T and remove it for processing <u>regardless of the number of TXNs in the C/T</u>.

and

> b = The average incremental incrase in work required to put a TXN in C/T and remove it for processing <u>caused by each (one) TXN added to the C/T</u>.

Similarly for W_m

$$W_m = c + dN_w = c + dN - dN_c$$

So our objective function becomes:

$$W = \frac{1}{N}\left[(a + bN_c)N_c + (c + dN - dN_c)(N - N_c)\right]$$

$$\frac{1}{N}\left[(aN_c + bN_c^2 + cN - cN_c + dN^2 - DNN_c - dNN_c + dN_c^2\right]$$

Differentiating W with respect to N_c to find the value of N_c for which W is a minimum:

$$\frac{dW}{dN} = \frac{1}{N}\left[\bar{a} + 2bN_c - c - 2dN + 2dN_c\right] = 0$$

$$N_c = \frac{2dN + c - a}{2(b+d)} = \frac{d}{b+d} N + \frac{c - a}{2(b+d)} = N_c \qquad (1)$$

$$R = \frac{N_c}{N} = \frac{d}{b+d} + \frac{c-a}{2N(b+d)} \qquad (2)$$

Since,

$$\left.\frac{d^{(2)}W}{dN_c^{(2)}}\right|_{N_c=N'_c} = \frac{2(b+d)}{N} > 0$$

Equation (1), in fact, provides the value of N_c yielding a minimum W.

Thus, R in equation (2) gives us the desired balance between N_c and N_m. But it does not give us a value for CLBIG. The value of CLBIG will depend on the magnitude of the Δt's in the simulation. In some simulations Δt could be on the order of microseconds, while in others it could be on the order of hours, depending on the system being simulated via this C/T-M/T method. Note that R is primarily a function of the "work" coefficients (a, b, c, and d), that is, the C/T-M/T code, not the model code or Δt's of the model. CLBIG is a function of the Δt's and should be selected such that a fraction, R, of the total number of TXNs processed fall in the C/T.

The distribution of the number of transactions as a function of t will vary from model to model as shown in Figure 6-2.

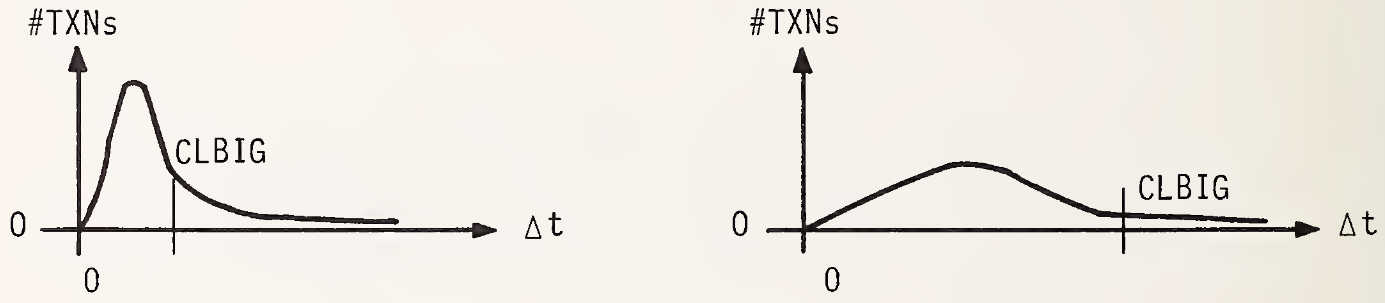


Figure 6-2. Contrasting Distributions from Two Models

Now assuming a particular value of R* of R is known, CLBIG (and in turn CLSMAL = CLBIG/DMCLTA = CLBIG/1000, which the user inputs) can be found by inspecting the histogram in the system output report and finding the time value in the second column that corresponds to R* in the fourth column. By varying the base of the histogram (FLDI4(3)) and the width of the cells (FLDI4(4)), the user can "home-in" on CLBIG and, thus, finer values of CLSMAL.

To initially estimate or to try to improve on the R* value that should be used, the histogram in the report should be used as follows. A series of runs should be made that vary CLBIG (by varying CLSMAL) and the execution time noted. The run with the minimum execution time should give the best value of R*. R* is calculated by finding the value of CLBIG in the second column of the histogram and then finding the corresponding percentage in the fourth column. The user should set the base and width of the histogram to insure that it contains the value of CLBIG used during the run. By the base and width the user can "home-in" on R*. The execution time of any given run may vary due to contention with other jobs in a multiprogramming system. Thus a plot execution time versus CLSMAL with a finer point may not produce a clear parabola as suggested by the analysis above. Thus, this should be done over a many runs as possible.

Lastly SAWTIN is called to list in the index file the members that were used in the run.

6.2.19.5 PDL - See Appendix A.

6.2.19.6 Decision Tables and Algorithms - None.

6.2.20 SAFLAG

6.2.20.1 Identification

- o SAFLAG - Intermediate Output Flag Setting
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.20.2 Argument Dictionary - None.

6.2.20.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
FINI	-	L*1	Indicates last card found
TEMP	18	I*4	Flag numbers from current card
AEOF	-	L*1	End of file on asynchronous card file
L	-	I*4	Lower bound of range of flags to be set true
U	-	I*4	Upper bound of range of flags to be set true

6.2.20.4 Description - This routine sets flags associated with getting intermediate output generated using the DBUG macro. This routine first turns off all flags and reads cards containing the numbers of flags to be set until a zero field is found. The requests can contain ranges of the flag value, e.g., 47-54. The requested flags are turned on and used henceforth.

6.2.20.5 PDL - See Appendix A.

6.2.20.6 Decision Tables and Algorithms - None.

6.2.21 SAINIT

6.2.21.1 Identification

- o SAINIT - System Initialization
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.21.2 Argument Dictionary - None.

6.2.21.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
REST	-	I*4	Restart card name
DATA	-	I*4	Data card name
OPTI	-	I*4	Option card name
SELE	-	I*4	Select card name
PARA	-	I*4	PARAM card name
FLAG	-	I*4	Flag card name
ZTIME	-	R*4	Time read from header card
TIME	-	R*4	Time read from trip and vehicle records
CU	-	I*4	Clock in seconds
AEOF	-	L*1	End of file on asynchronous card file
TEOF	-	L*1	End of file on trip file
VEOF	3	L*1	End of file on vehicle files
KUNIT	-	I*4	I/O unit number from which to read vehicle

6.2.21.4 Description - System initialization is performed to establish the initial conditions for a simulation experiment. System initialization begins by calling SANTSA to initialize the system status area addresses. Then the first asynchronous data card is read from AGT.STRUC.RNTIM. If this card is a restart card entry point SAREST of SACKR is called to perform restart. Otherwise, initialization proceeds as follows. Entry point SANDTA of SADADD is called to read in the binary system characteristics data from AGT.STRUC.SYSTEM and to initialize message counters. Then any zero time DATA, OPTION, PARAM, SELECT, or FLAG cards and their associated follower cards are read from AGT.STRUC.RNTIM. Next SANXTN is called to initialize transaction data, SANFEL to initialize the FEL, and SANMDL to initialize model related data. After this the first trip and vehicle records are read and scheduled. Then, unless their requested intervals are zero, the periodic sampling and checkpointing transactions are scheduled. The entry point SAUPTIX of SANTIX is called to update the index file with load module name, date and time. Lastly, the next asynchronous read transaction is scheduled.

Index cards are processed as in SAASYN. Trip records are skipped over until a trip with an origin equal to the station being simulated is found.

6.2.21.5 PDL - See Appendix A.

6.2.21.6 Decision Tables and Algorithms - None.

6.2.22 SAMAIN

6.2.22.1 Identification

- o SAMAIN - Model Processor Control
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.22.2 Argument Dictionary - None.

6.2.22.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OLDSL	-	I*4	Previous link current vehicle was on
AEND	-	L*1	End of SIM XTN found
LOOP	-	I*4	Count of number of times XTN was removed from FEL without clock changing; compare with CLOOP
NOW	-	I*4	Time of current transaction

6.2.22.4 Description - The purpose of SAMAIN is to serve as the main control loop of the simulator. This code segment runs code segment SAINIT to initialize the simulator. After this, SAMAIN continues to perform the following operation until a termination transaction is encountered. It gets the most imminent transaction off the Future Event List (FEL), updates the clock to the time specified in this transaction, and then proceeds according to the type of event as specified in the transaction. The following are the different types of system events. They should not be confused with events that can occur to a vehicle or a trip -- these latter types of events can be viewed as "subevents" to these system events.

1. Vehicle Event -- Something is about to happen to a vehicle.

2. Trip Event -- Something is about to happen to a trip.

3. Asynchronous Event -- Something is about to happen (such as failure) that required the reading of more data into the simulator.

4. Sampling Event -- It is now time to write out the values of the simulation output variables.

5. Periodic Checkpoint -- It is now time to take a checkpoint of the system.

6. Trip Origination -- A trip is about to arrive at the simulated station.

7. Vehicle Origination -- A vehicle is about to arrive in the simulated station area.

8. Station Link Prompt -- It is now time to try to get a vehicle moving that was queued due to congestion or failure on a station link.

9. Trip Link Prompt -- It is now time to try to get a trip moving that was queued due to congestion or failure on a trip link.

10. End of Simulation -- It is now time to terminate the simulation.

For each of these system events, an appropriate code segment is run.

6.2.22.5 PDL - See Appendix A.

6.2.22.6 Decision Tables and Algorithms - None.

6.2.23 SANFEL

6.2.23.1 Identification

- o SANFEL - Future Event List Initialization
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.23.2 Argument Dictionary - None.

6.2.23.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
INFINY	-	I*4	Largest possible I*4

6.2.23.4 Description - This routine initializes the FEL. It begins by getting a transaction and initializing it to be the infinite time multiple thread transaction. Then the clock table and timing statistics are set to zero. Internal timing control parameters are set based on input timing control parameter values.

6.2.23.5 PDL - See Appendix A.

6.2.23.6 Decision Tables and Algorithms - None.

6.2.24 SANMDL

6.2.24.1 Identification

- o SANMDL - Model Variable Initialization
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.24.2 Argument Dictionary - None.

6.2.24.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
KNTL	-	I*4	Number of trip links
IND	-	I*4	Improper configuration for local merge
TLMAX	-	I*4	Maximum train length

6.2.24.4 Description

- This routine initializes internal (non-input) variables used by the model. This routine sets the variables in the internal commons (SCMSYS, SCMSL, SCMTL, SCMT, SCMV) to their appropriate initial values. Care is taken to avoid resetting values in the first trips and vehicles that were set at the initial reads. SAZNIT is called to initialize model statistics. Lastly, if the delay associated with local merge is requested, tests are done to insure the required configuration of a downstream link having upstream of bypass and output links upstream of it is present; that the bypass link headway is greater than zero so that the slot width on the bypass link is greater than zero; and that the lengths of the bypass and output links are consistent. If these tests fail, the local merge option is turned off, a message is printed, and the run proceeds.

6.2.24.5 PDL - See Appendix A.

6.2.24.6 Decision Tables and Algorithms - None.

6.2.25 SANSAV

6.2.25.1 Identification

- o SANSAV - Initialize Checkpointing and System Data Read Processor
- o IBM/FSD - July 1, 1977
- o ASM

6.2.25.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
ARGA	–	I*4	ADDRESS OF BEGINNING OF COMMONS
ARGB	–	I*4	LENGTH OF COMMONS IN WORDS
ARGC	–	I*4	ADDRESS OF BEGINNING OF SYSTEM CHARACTERISTICS DATA
ARGD	–	I*4	LENGTH OF SYSTEM CHARACTERISTICS IN WORDS

6.2.25.3 Local Variable Dictionary - None.

6.2.25.4 Description - The purpose of SANSAV is to inform checkpointing and system data read routines of addresses and lengths of common areas which they need to perform their functions. This routine calls SACKR to initialize it with the starting address of all the commons and their total length and calls SADADD to initialize it with the starting address of the input commons and their length. Before making these calls, this routine also performs the function of converting these addresses of addresses into simple addresses so that the FORTRAN routines SACKR and SADADD can proceed.

6.2.25.5 PDL - See Appendix A.

6.2.25.6 Decision Tables and Algorithms - None.

6.2.26 SANTIX

6.2.26.1 Identification

- o SANTIX - Initialize Member Name String
- o IBM/FSD - July 1, 1977
- o ASM

6.2.26.2 Argument Dictionary - None.

6.2.26.3 Local Variable Dictionary - None.

6.2.26.4 Description - SANTIX saves the address of the member name string from the PARM field of the EXEC card and makes it available when necessary to update the index file. The model processor is entered from the system through this routine. When it is first entered, it saves the address of the contents of the PARM field of the EXEC card. The system had put this address in register 1. After this save, control is passed to SAMAIN. When the routine is called again later through its entry point SAUPTX from SAINIT, the address of the string is restored to register 1 ready for SAWTIX to use it and then SAWTIX is called to actually update the index file using this string. Upon return from SAWTIX, control returns to SAINIT without reentering SAUPTX.

6.2.26.5 PDL - See Appendix A.

6.2.26.6 Decision Tables and Algorithms - None.

6.2.27 SANTSA

6.2.27.1 Identification

- o SANTSA - Initialize System Status Area Addresses
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.27.2 Argument Dictionary - None.

6.2.27.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
ACORE	3	I*4	Addresses of start of input commons, start of model commons, and end of model commons

6.2.27.4 Description - The purpose of SANTSA is to initialize system status area addresses. The input area commons (SCIFEL, SCIMAX, SCISL, SCISYS, and SCITL) and the internal model commons (SCMFEL, SCMSL, SCMSYS, SCMSL, SCMT, SCMV, SCMXTN, SCMFS, SCAMSG) lie in storage as two contiguous blocks. The routine SADADD that reads the binary system data from AGT.STRUC.SYSTEM must know the starting address and length of these commons in order to do the read. Also, SACKR, the routine that writes or reads a checkpoint record (which contains both blocks), must know the starting address and length to do its reads and writes. This is accomplished in the following way. At linkage edit time, these two blocks of commons are put in an overlay structure so that the addresses, their starting, middle, and end addresses can be given names BEGCOM, IPCOM, and ENDCOM. Also at linkage edit time, SSASAV serves to capture these addresses of addresses. At execution time, this routine SANTSA is called from SAINIT and contains a common called SSASAV containing one array ACORE of dimension 3. Thus SANTSA knows the addresses. It then proceeds to calculate starting addresses and lengths of the two blocks from the addresses and passes them to SANSAV which will call SACKR to inform it of the addresses and lengths it needs to do checkpointing and restarting and SADADD to read the binary system data.

6.2.27.5 PDL - See Appendix A.

6.2.27.6 Decision Tables and Algorithms - None.

6.2.28 <u>SANXTN</u>

6.2.28.1 <u>Identification</u>

- o SANXTN - Initialize Transaction Data
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.28.2 <u>Argument Dictionary</u> - None.

6.2.28.3 <u>Local Variable Dictionary</u> - None.

6.2.28.4 <u>Description</u> - SANXTN initializes the data in SCMXTN for vehicle, trip, and system service transactions. First, all the variables in SCMXTN are set to zero, except the chain word of each which is set to point to the next. Then the trip, vehicle, and system service available lists are initialized.

6.2.28.5 <u>PDL</u> - See Appendix A.

6.2.28.6 <u>Decision Tables and Algorithms</u> - None.

6.2.29 SAPFEL

6.2.29.1 Identification

- o SAPFEL - Put Transaction on FEL
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.29.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
XTN	—	I*4	ID OF TRANSACTION TO BE SCHEDULED
GOTO	—	I*4	SYSTEM EVENT OF THE NEXT EVENT FOR XTN
DELTA	—	I*4	THE TIME INTERVAL (IN C.U.'S) THAT XTN IS TO STAY ON THE FEL
PRTY	—	I*4	THE PRIORITY ORDER IN WHICH IT IS TO COME OFF THE FEL

6.2.29.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
LINK	-	I*2	Number of link of current trip/vehicle
TIME	-	I*4	Maximum of new time of XTN and CLMINI
FMTHRD	-	I*4	Holds ID of current XTN when looping thru M/T
NMTHRD	-	I*4	Holds ID of next XTN when looping thru M/T
FIRST	-	I*4	Holds ID of current XTN when looping thru C/T
NEXT	-	I*4	Holds ID of next XTN when looping thru C/T
DTIME	-	I*4	Delay time in queue or on FEL
XID	-	I*4	XTN ID
TXN	2	L*1	Transaction type trip or vehicle
WASQD	-	L*1	Indicates XTN was queued
VID	-	I*4	ID of vehicle being processed

6.2.29.4 Description - The purpose of SAPFEL is to put a transaction on the FEL in correct time order. SAPFEL performs the scheduling of a transaction on the future events list. SAPFEL is invoked by either the scheduling of a transaction via the SCHED macro or via a direct call. The transaction to be placed on the future events list is either placed in the clock table or on the multiple thread list depending upon whether the schedule time is within the current clock table interval or at some extended time in the future. Scheduling on the clock table involves finding the correct position for insertion and adding the transaction ID to the clock table. Multiple thread scheduling requires either the addition of the transaction to an existing multiple thread loop or the creation of a new multiple thread loop. Concurrent with scheduling trip and vehicle transactions, trip next event data is written to the trip and vehicle file when required. A history of the trip's/vehicle's last queued status is also written to the file.

6.2.29.5 PDL - See Appendix A.

6.2.29.6 Decision Tables and Algorithms - None.

6.2.30 SARFEL

6.2.30.1 Identification

- o SARFEL - Removes Next Most Imminent Transaction from FEL
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.30.2 Argument Dictionary - None.

6.2.30.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
XMTHRD	-	I*4	Holds ID of next XTN when looping thru M/T
XFIRST	-	I*4	Holds ID of current XTN when looping thru C/T
XNEXT	-	I*4	Holds ID of next XTN when looping thru C/T

6.2.30.4 Description - The purpose of SARFEL is to obtain the next imminent event to be performed from the Future Events List and update the clock table and multiple thread list as necessary. A sequential scan of successive entries in the clock table, beginning with the currently active interval, is performed until a non-empty interval or the end of the clock table is reached. If a non-empty interval pointer is found, the first transaction chained within the interval is removed and returned as the currently active transaction requiring event processing. If the end of the table is reached during the scan, the first available multiple thread FEL list is removed from the multiple thread chain and reloading of the FEL is performed. The base time value of the FEL is reestablished to the time of the multiple thread transaction.

Each transaction chained on the multiple thread list is removed and chained in time order within the clock table interval given by:

$$I = 1 + \frac{XTIME_{XTN} - CLBASE}{CLSIZE}$$

Where:

$XTIME_{XTN}$ = Scheduled time of transaction

CLSIZE = Time value encompassed by a clock table interval

CLBASE = Base time value for the clock table.

Once loading of the clock table is complete, the first available transaction within the current table interval (first reloaded clock interval) is returned as the currently active transaction requiring event processing.

6.2.30.5 PDL - See Appendix A.

6.2.30.6 Decision Tables and Algorithms - None.

6.2.31 SASAMP

6.2.31.1 Identification

- o SASAMP - Sample Event Processing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.31.2 Argument Dictionary - None.

6.2.31.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
FOLLOW	-	R*8	Keyword 'FOLLOWER' to write in record
KNTL	3	I*4	Number of trip links
NBY	-	I*4	Number of bytes in follower
NFOLL	-	I*4	Number of followers
A	92	R*4	Data for performance summary average times

6.2.31.4 Description - A sampling event causes statistics reflecting modeling subsystem's status accumulated over an interval of time to be recorded in the raw statistics file. This routine first calls SZINT to calculate integrals and averages. It then checks to see if this is a user specified multiple of intervals at which snapshot reports are to be written and, if so, writes them using SAFINM. Next values are collected in array A for the OP to use in computing average times for the performance summary. Next the stationwide statistics are written, followed by those relating to each station link and each trip link. Finally, SZZERO is called to reset the statistics values so that they are ready for accumulation during the next sampling interval.

Time integrals are computed during the sampling interval by:

1. Initializing the integral at the start of the sampling interval to the negative of the product of the current time (clock) value and the current occupancy of the element to which the integral is associated. This is done in SZZERO.

2. Whenever the occupancy decreases during the sampling interval, the integral is increased by the product of the current time value and the size of the occupancy decreases. This is done in SZSTAT.

3. Whenever the occupancy increases during the sampling interval, the integral is decreased by the product of the current time and the size of the increase. This is done in SZSTAT.

4. At the end of the interval, the integral value is corrected by increasing it by the product of the current time value and the current occupancy. This is done in SZINT. At this point, the integral contains the desired value.

Once sampling processing is completed, the sampling transaction is scheduled to occur at the next sample time.

6.2.31.5 PDL - See Appendix A.

6.2.31.6 Decision Tables and Algorithms - None.

6.2.32 SASCTL

6.2.32.1 Identification

- o SASCTL - Control for Vehicle Event Processing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.32.2 Argument Dictionary - None.

6.2.32.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OLDSL		I*4	THE VEHICLE'S CURRENT STATION LINK.
PASCNT		I*4	THE NUMBER OF PASSENGERS ON A TRAIN.
CHEAD		I*4	POINTER TO THE HEAD OF THE TRAIN CHAIN.

6.2.32.4 Description - The purpose of SASCTL is to control the transition of a vehicle transaction when the vehicle is moving from one station link to another. SASCTL is given control by SAMAIN when the transaction that comes off the FEL indicates that some event is about to happen to a vehicle. First it runs SSMOD to perform the processing associated with the event that has come off the FEL. Next, it determines if the vehicle in question has completed all the events on the station link on which it is currently traveling. If the vehicle is still undergoing event processing, SASCTL does nothing since SSMOD put the vehicle back on the FEL to wait for its next event. If the vehicle has completed all the events associated with the link on which it is traveling, SASCTL then tries to get the vehicle onto the next station link. It does this by using SSTEST to determine the next link that should be entered and to determine if that link can be entered. If the next link cannot be entered, SSTEST queues the vehicle at the end of its current link. Otherwise, it runs SSLEAV to perform processing associated with leaving the link on which it is traveling. In the case where the next link is a sink and the vehicle is leaving the simulated area, final trip and vehicle statistics are recorded and the transactions used to represent the vehicle, and its onboard trips are returned to the available list so that data areas can be reused as other vehicles and trips. When the next station link is not a sink, but another station link, SASCTL resets the vehicle event number to indicate that the vehicle should undergo the processing associated with the first event on the next link and calls SSMOD which will perform that processing and put the vehicle back on the FEL.

6.2.32.5 <u>PDL</u> - See Appendix A.

6.2.32.6 <u>Decision Tables and Algorithms</u> - None.

6.2.33 SASPRM

6.2.33.1 Identification

- o SASPRM - Station Link Prompt Event Processing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.33.2 Argument Dictionary - None.

6.2.33.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
LIST	KMSL	I*4	List of vehicles to attempt to move
FLAG	-	L*1	Indicates self prompt is to be done
SL	-	I*4	Station link being prompted
TEMP	-	I*4	Intermediate variable for bubble sort
PASCNT	-	I*4	Passenger count
QHEAD	-	I*4	Pointer to head of train
VID	-	I*4	ID of vehicle currently being processed
I	-	I*4	Index of misc. statistic to update
J	-	I*4	Index of misc. statistic to update

6.2.33.4 Description - This routine serves to get vehicles moving (viz., schedule them to spend time on the FEL), that had been queued. The station link in question is recovered from a word of the active (prompt) system service transaction which was scheduled by SSPMAC. Then a list is built of all vehicles that may now be able to move. If the prompt is a self prompt, then the list contains at most one vehicle, the head vehicle on the subject link; otherwise, it may contain the head vehicle on each link immediately upstream of the subject link. A vehicle is put in the list if and only if it is queued and done with processing on the link.

After the list is built, if the user has specified FIFO dequeuing from upstream of the link (by using SLPF), the list is reordered on the basis of the times the vehicles finished their last event on the link. Otherwise, a priority situation exists and the list, which was originally built in the order of priority that the user specified in listing upstream links, is not reordered.

After the list of candidate vehicles is ordered, a control structure similar to SASCTL is run through for each vehicle in the list. SSTEST is run to determine if the vehicle can leave its current link. If it can, SSLEAV is run to perform station link leave processing. If a sink follows the current link, the vehicle transactions and onboard trip transactions are returned to the available lists and statistics are collected. Otherwise, SSMOD is called to commence station link processing on the next link.

6.2.33.5 PDL - See Appendix A.

6.2.33.6 Decision Tables and Algorithms - None.

6.2.34 SATORG

6.2.34.1 Identification

- o SATORG - Move Arriving Trip
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.34.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TN	-	I*4	ID OF TRIP THAT IS ORIGINATING

6.2.34.3 Local Variable Dictionary - None.

6.2.34.4 Description - The purpose of SATORG is to initialize a transaction for an arriving trip and run SUMOD to get it moving. First a test is made to see if there is adequate room in the ticketing link to accommodate the trip. If not, the trip is rejected and the rejection recorded. If there is room and the trip is larger than a user specified split size, then the trip is split into subtrips. For each such subtrip a transaction is acquired from the available list, initialized to the characteristics of the trip, and SUMOD is run to get the trip moving.

6.2.34.5 PDL - See Appendix A.

6.2.34.6 Decision Tables and Algorithms - None.

6.2.35 SATRD

6.2.35.1 Identification

- o SATRD - Read Trip from Trip File
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.35.2 Argument Dictionary - None.

6.2.35.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TIME	-	R*4	Time read from trip file
TEOF	-	L*1	End of file on trip file
TN	-	I*4	ID of transaction gotten for new trip

6.2.35.4 Description - The purpose of this routine is to read a trip record and initialize its transaction. A transaction is acquired from the trip available chain. The trip record is read into the fields of the transaction. If the origin of the trip is not equal to the station being simulated, it is skipped over and the next trip is read. Arrival time is converted to clock units. The trip arrival system service transaction is updated to contain the transaction number of the newly arrived trip.

6.2.35.5 PDL - See Appendix A.

6.2.35.6 Decision Tables and Algorithms - None.

6.2.36 SAUCTL

6.2.36.1 Identification

- o SAUCTL - Control of Trip Event Processing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.36.2 Argument Dictionary - None.

6.2.36.3 Local Variable Dictionary - None.

6.2.36.4 Description - The purpose of SAUCTL is to control the transition of a trip from one trip link to another. SAUCTL is given control via SAMAIN when a trip transaction comes off FEL and requires processing for a trip link event. First it runs SUMOD to perform the processing associated with the event for which the trip has spent time on the FEL. Next it determines if SUMOD has set the data item ADONET to indicate that the trip being processed has completed all of the events on its current trip link. If the trip has not yet completed all events, SAUCTL processing is complete for the time being since SUMOD returned the trip to the FEL for the duration of its event. Otherwise, SAUCTL tries to advance the trip to its next trip link. This is done by using SUTEST to identify the next link and determine that the trip can be accommodated thereon. If entry is precluded, SUTEST indicates that the trip has been queued. Otherwise, when the trip can enter, SULEAV is invoked to process the trip leaving its current trip link. In the case where there are no further trip links, the trip is prepared to enter the station's boarding queue and SMTABQ is run to assure that a vehicle is moving toward the trip in the case of a demand responsive environment. In the case of scheduled service the trip is allowed to board a vehicle undergoing boarding if the destination is compatible. When there are further trip links, SAUCTL advances the trip to the first event as the next link. Then SUMOD is invoked to perform the processing associated with the first event on the next link and return the trip to the FEL for the duration of that trip link event.

6.2.36.5 PDL - See Appendix A.

6.2.36.6 Decision Tables and Algorithms - None.

6.2.37 SAUPRM

6.2.37.1 Identification

- o SAUPRM - Trip Link Prompt Event Processing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.37.2 Argument Dictionary - None.

6.2.37.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TL	-	I*4	Number of trip link being prompted
TLP	-	I*4	Number of trip link upstream of TL
T	-	I*4	ID of the trip to try to move

6.2.37.4 Description - This routine serves to get trips moving, viz., scheduled on the FEL, that had been queued. The trip link in question is recovered from a word of the active (prompt) system service transaction which was scheduled by SUPMAC. If the head trip on the link upstream of the one in question is queued and done, then SAUPRM's control structure (similar to SAUCTL) is run to try to get the trip moving again. SUTEST is entered to determine if the trip can leave its current link. If it can, SULEAV is run to perform trip link processing. Then SSMOD is called to commence trip link processing on the next link.

6.2.37.5 PDL - See Appendix A.

6.2.37.6 Decision Tables and Algorithms - None.

6.2.38 SAVORG

6.2.38.1 Identification

- o SAVORG - Move Arriving Vehicle
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.38.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
VN	-	I*4	ID OF THE VEHICLE ORIGINATING

6.2.38.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
ASY	-	L*1	Indicates SAVORG called from SAASYN instead of SAMAIN
EOF	-	L*1	End of file on vehicle file prematurely
TIME	-	R*4	Time read from vehicle file
SOUR	-	I*4	Source number of current vehicle
QHEAD	-	I*4	Head vehicle in train
N	-	I*4	Number of vehicles in train
VNT	-	I*4	Number of onboard trips
FN	-	I*4	Number of unit vehicle file is on
TN	-	I*4	Number of trip transaction
VNV	-	I*4	Number of following vehicle in a train

6.2.38.4 Description - The purpose of SAVORG is to initialize a transaction for an arriving vehicle and run SSMOD to get the vehicle moving. First, the source of the vehicle is determined. For vehicles read asynchronously, the source is assumed to be the guideway. Next the vehicle data associated with the vehicle transaction that was acquired at the last execution of SAVRD for the source is initialized. Any onboard trips are read from the vehicle file and trip transactions are acquired and initialized for them. If the vehicle was the first of a train, then the follower vehicles and their onboard trips are read, acquired, and initialized. Then statistics are collected and SSMOD is called to get the vehicle moving on its source link.

6.2.38.5 PDL - See Appendix A.

6.2.38.6 Decision Tables and Algorithms - None.

6.2.39 SAVRD

6.2.39.1 Identification

- o SAVRD - Read Vehicle from Vehicle File
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.39.2 Argument Dictionary - None.

6.2.39.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TIME	-	R*4	Time read from vehicle file
J	-	I*4	Source of the vehicle
FN	-	I*4	Number of unit vehicle file is on
VEOF	3	L*1	End of file on vehicle files
VN	-	I*4	Number of transaction gotten for this vehicle

6.2.39.4 Description - Read a vehicle record and initialize its transaction. A transaction is acquired from the vehicle available list chain. The vehicle record is read into the fields of the transaction. Arrival time is converted to clock units. The vehicle arrival system service transaction has the transaction number of the newly arrived vehicle stored in its associated data words.

6.2.39.5 PDL - See Appendix A.

6.2.39.6 Decision Tables and Algorithms - None.

6.2.40 SAWTIX

6.2.40.1 Identification

- o SAWTIX - Write Index File Update
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.40.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
COUNT	–	I*2	NUMBER OF CHARACTERS IN STRING
STRING	–	R*8	UP TO 8 CHARACTER NAME OF SAMPLE & CKPT FILES

6.2.40.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
MONTH	-	I*2	Month of year
DAY	-	I*2	Day of year
YEAR	-	I*2	Year
HOUR	-	I*2	Hour of day
MIN	-	I*2	Minute of hour

6.2.40.4 Description - SAWTIX first parses the parm field to get individual names. Then DAYTIM is called to get the date and time. NExt the load module name, date and time are written to the index file. When entry SAWTIW is called (from SAFINS) the files that were used in the run are listed in the index.

6.2.40.5 PDL - See Appendix A.

6.2.40.6 Decision Tables and Algorithms - None.

6.2.41 SAZNIT

6.2.41.1 Identification

- o SAZNIT - Initialize Statistical Variables
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.41.2 Argument Dictionary - None.

6.2.41.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NBY	-	I*2	Number of bytes in first follower record
FOLLOW	-	R*8	Keyword 'FOLLOWER' to write in record
KNTL	-	I*4	Number of trip links
KNSL1	-	I*4	KNSL

6.2.41.4 Description - SAZNIT initializes statistical variables and writes the first records to the raw statistics file. This routine begins by initializing the status type statistical variables. These are the only statistics that will not be reset in SZZERO every sample. Here they are the number of entities in each state. Then SZZERO is called to initialize the remaining variables. Next, SZHDR is called to write the first header record to the raw statistics. Lastly, the first follower record containing the number of station links, trip links, clock units per minute, clock units per sampling interval, station link types and five input parameters describing the configuration are written to the raw statistics file. This data is needed by the output processor in determining the location of data in subsequent records written by SASAMP.

6.2.41.5 PDL - See Appendix A.

6.2.41.6 Decision Tables and Algorithms - None.

6.2.42 SCHED

6.2.42.1 Identification

- o SCHED - Schedule a Transaction for an Event Completion Time Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.42.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
INDEX	–	I*4	VARIABLE NAME OF ENTITY TO BE SCHEDULED.
TYPE	–	C*1	TYPE OF SCHEDULED EVENT (OPTIONAL): S = STATION LINK EVENT FOR A VEHICLE T = TRIP LINK EVENT FOR A TRIP NULL = OTHER.
MEVNT	–	I*4	EVENT NUMBER FOR WHICH INDEX IS ON THE FEL
DELTA	–	I*4	TIME ON THE FEL IN CLOCK UNITS (ZERO ASSUMED IF DELTA IS NEGATIVE.
YRTY	–	I*4	ORDER RELATIVE TO OTHER TXNS COMMING OFF THE FEL AT THE SAME TIME. PRIORITY VALUES FROM 0 THROUGH 9. 0 = HIGHEST PRIORITY AND 9 IS LOWEST PRIORITY.(OPTIONAL; DEFAULT = 0)

6.2.42.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
PRTY	-	C	'0' default priority

6.2.42.4 Description - The purpose of SCHED is to schedule a transaction for an event completion time to come off of the FEL. This macro generates code which when executed uses MULTICK to ensure that the entity is not already enqueued and if not calls SAPFEL to put the entity on the FEL.

6.2.42.5 PDL - See Appendix A.

6.2.42.6 Decision Tables and Algorithms - None.

6.2.43 SERROR

6.2.43.1 Identification

- SERROR - Write Error Message and Continue or Terminate
- IBM/FSD - July 1, 1977
- PARAFOR

6.2.43.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
ERROR:			
MSGNO	-	I*4	ERROR MESSAGE NUMBER
MSG	2	L*1	MESSAGE TEXT
MSEVER	-	I*4	MESSAGE SEVERITY (1=I,2=W,3=S)

6.2.43.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
PGM	2	I*4	Error message prefix
MCLOCK	-	I*4	Clock in seconds
MSG	2	L*1	Message text character, used to count characters until semicolon
SCLN	-	L*1	Semicolon (used to indicate end of message)
TYPE	-	L*1	Message level character (info, warning, severe)
MSGTYP	-	L*1	Message type
TERM	-	L*1	Terminate simulation

6.2.43.4 Description - The purpose of SERROR is to write an error message when an anomalous situation arises and continue or terminate. This begins by determining the length of the message in characters. (When called, the message text is required to be in quotes and terminated by a semicolon.) Next, this text is printed together with the standard text line appropriate to each severity:

SEVERITY	TEXT
1 (Information)	This condition may be acceptable to the user.
2 (Warning)	This condition must be corrected prior to the next run.
3 (Severe)	Execution cannot proceed beyond this point.

This is followed by the value of the clock. Next the number of messages issued by ID number and severity class are incremented. If either the message type was severe or the number of informative, warning, or both type messages exceeded a compile time maximum (KMMSGI, KMMSGN, KMMSGS) or the number of messages of any one given ID number exceeded a compile time maximum (KMMTYP), then the simulation is terminated; otherwise, it is continued.

6.2.43.5 PDL - See Appendix A.

6.2.43.6 Decision Tables and Algorithms - None.

6.2.44 SMBRD

6.2.44.1 Identification

- o SMBRD - Planning Trip Boarding
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.44.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	–	I*4	SOLITARY VEHICLE OR LEAD VEHICLE OF A TRAIN

6.2.44.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
MATCH	–	I*2	INDICATES WHETHER OR NOT THE VEHICLE AND TRIP ARE COMPATIBLE: 1 = MATCH 2 = NO MATCH
QHEADV	–	I*4	TAIL OF THE TRAIN CHAIN
CHECK	–	L*1	INDICATES WHETHER OR NOT THE COMPATIBILITY OF TRIP AND VEHICLE HAS BEEN TESTED: T = WAS TESTED AND THEREFORE PROCEED TO NEXT TRIP. F = WAS NOT TESTED THEREFORE TRY THE NEXT VEHICLE.
FULL	–	L*1	NOT USED
TBEFOR	–	I*4	PREDECESSOR TRIP IN QLOOP BOARDING QUEUE
TID	–	I*4	A TRIP IN THE BOARDING QUEUE
VBEFOR	–	I*4	PREDECESSOR VEHICLE IN QLOOP TRAIN
VID	–	I*4	A VEHICLE IN THE TRAIN
STPTR	–	I*2	POINTER TO A STATION IN THE VEHICLE'S STATION ROUTE LIST
ONE	–	I*2	INTEGER*2 VERSION OF THE CONSTANT 1
TWO	–	I*2	INTEGER*2 VERSION OF THE CONSTANT 2

6.2.44.4 Description - The purpose of SMBRD is to build a list of trips to board the vehicle being processed or each vehicle in the train. In the case of demand responsive single party service, the trip at the head of the boarding queue is selected (if there is one); it is dequeued from the boarding queue and enqueued into the vehicle's boarding list.

In the case of demand responsive multiparty service, if the vehicle is empty, then the trip at the head of the boarding queue is put on the list.

For all other trips in the boarding queue, a test is made to see if the trip can fit on the vehicle and if so, a compatibility test is made using a random number and a user specified probability of compatibility. If it is determined that the trip is compatible, then the trip is dequeued from the boarding queue and enqueued into the boarding list of the vehicle. This process continues until either there are no more trips in the boarding queue or the vehicle is at capacity.

In the case of scheduled service, starting with the first trip in the station's boarding queue and the first vehicle in the train, each vehicle in the train is checked to see if the trip can fit on the vehicle. When a fit is found, a further check for destination compatibility is made. Should there be insufficient space on the train for the trip or incompatibility, the same search proceeds for the next trip in the boarding queue, and so on.

When a trip can fit onto the train, one of three compatibility tests can be selected by the user. The sampling test uses a probability of compatibility to determine if the trip is compatible. The second test, the route test using the route assignment table, is invoked whenever this one-route-per-destination table has been provided by the user. The route of the vehicle is compared to the one route allowed for the trip based on the trip's destination. When the route assignment table has not been specified by the user, the third method to check compatibility is used. The list of stations on the vehicle's route is evaluated to see if any one of them is the trip's destination. A trip than can fit on the vehicle and has a compatible destination is dequeued from the boarding queue and enqueued onto the boarding list of the vehicle. During this processing the total number of passengers that are to board each vehicle is maintained for later use in computing vehicle boarding time.

6.2.44.5 PDL - See Appendix A.

6.2.44.6 Decision Tables and Algorithms - None.

6.2.45 SMDBRD

6.2.45.1 Identification

- o SMDBRD - Planning Trip Deboarding
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.45.2 Argument Dictionary - None.

6.2.45.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
XFER	—	I*2	INDICATES WHETHER OR NOT A TRIP WILL TRANSFER: 1 = TRANSFER 2 = NO TRANSFER
TBEFOR	—	I*4	PREDECESSOR TRIP IN GLOOP VEHICLE TRIP QUEUE
TID	—	I*4	A TRIP IN THE VEHICLE'S TRIP QUEUE

6.2.45.4 Description - SMDBRD plans the deboarding of one vehicle building two lists of trips:

1. Those trips that deboard and leave the system.
2. Those trips that deboard and transfer; a count of passengers deboarding is also maintained.

For each trip onboard the vehicle, a test is made to see if the destination of the trip is equal to the station being simulated. If so, the trip is dequeued from the onboard queue and enqueued into the deboard and leave list. If this is not so, then a test is made to see if the trip is to transfer at this station. A user specified probability of transfer is used together with a random number to determine if the trip is to transfer. If it is to transfer, it is dequeued from the onboard queue and enqueued into the deboard and transfer list. During this process, the total number of deboarding passengers is accumulated.

6.2.45.5 PDL - See Appendix A.

6.2.45.6 Decision Tables and Algorithms - None.

6.2.46 SMDETR

6.2.46.1 Identification

- o SMDETR - Detrain Vehicles from Lead Vehicle of a Train
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.46.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	–	I*4	LEAD VEHCLE OF A TRAIN (OR COULD BE AN INDIVIDUAL VEHICLE)

6.2.46.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
GHEAD	–	I*4	TAIL TO A TEMPORARY STATION LINK MEMBERSHIP LIST
GHEADZ	–	I*4	TAIL TO THE TRAIN CHAIN
VID	–	I*4	VEHICLE BEING PROCESSED

6.2.46.4 Description - The purpose of SMDETR is to detrain all vehicles from the lead vehicle of a train. Detrain the following vehicles from the lead vehicle in a train maintaining their original order, adding each follower to the station link membership list, and assigning them the attributes of the lead vehicle.

6.2.46.5 PDL - See Appendix A.

6.2.46.6 Decision Tables and Algorithms - None.

6.2.47 SMDIVF

6.2.47.1 Identification

- o SMDIVF - Diverge Functions
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.47.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	-	I*4	ID OF THE VEHICLE BEING PROCESSED

6.2.47.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SLN	KMSL	I*4	List of station links found in search

6.2.47.4 Description - The purpose of SMDIVF is to create a list downstream links that is ordered by preference and contains only feasible candidates for entry. The structure of SMDIVF is a CASE block where the data item that controls which case is run is the user's specification of the diverge function to be used when exiting the current link. There are six diverge functions within SMDIVF corresponding to its six cases.

A diverge function is a rule by which the simulation will decide which station link a vehicle will enter when the vehicle is at a diverge. For example, in Figure 6-3, the diverge function used for link A will decide which of links B, C, or D a vehicle will enter. (In the case where there is just one link downstream of another, no diverge function is used since there is no decision to be made.)

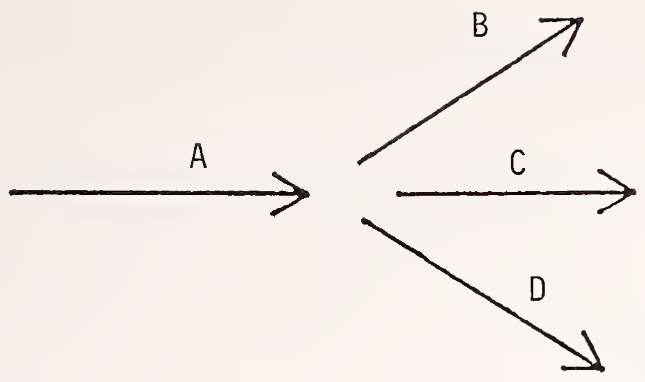


Figure 6-3. Sample Diverge

Often there are a number of diverges in a station to be modeled. The user specifies the number of a preprogrammed diverge function on each station link that has a multiple number of links downstream of it. The following discussion describes the six available diverge functions comparing their common input data and processing methodologies and contrasting their link ordering decision rules. In the event other rules are desired, the user may develop other diverge functions, add them to the simulator code, and request them at execution time.

All six diverge functions can use the following data:

1. Next stop of the vehicle (station number)

2. Divert-to-dock indicator (0 = divert to dock, 1 = go the other way)

3. Sink of the vehicle, viz., the vehicle's station exit mode, (1 = guideway, 2 = modal exit before dock, 3 = modal exit after dock).

All six diverge functions have the same input and output methodology:

1. Input -- The main input to a diverge function can be thought of as the list of all links immediately downstream of the link on which the vehicle is currently located.

2. Output -- The main output of a diverge function can be thought of as the input list with:

   a. Incompatible links omitted (e.g., the input-to-storage link eliminated for a vehicle that is to divert to the dock)

b. The remaining links ordered in order of preference (e.g., minimum occupancy first).

This output list is then used in the following way. Each link on the list is tested in the returned order to see if it can be entered. If the entry test fails due to failure at link entry, congestion, headway zone occupancy, etc., then the next link on the list is tested and so on until either a link is successfully entered or until the list is exhausted (in which case the vehicle queues on its current link). This ability to test other links if one is impassible allows the vehicle to be cleared out of the way when it would otherwise be caught waiting for that one link to recover and thus determines where vehicles will travel in these alternate link situations.

The first four diverge functions use a search function (SMDIVS) as a service routine to look for links of a specific type. The service routine uses:

1. The list of downstream links with their associated link types

2. The link type for which the diverge function is searching

3. An arming indicator.

This search routine builds a list of all downstream links of the requested type. This list is returned to the diverge function. If no links of the requested type are found and the indicator is armed, then the simulation is terminated. So, for example, if the user accidentally input vehicles to divert to storage (from, say a DESM run) and there was no storage, this condition would activate the arming indicator to terminate the simulation. This generalized search process is used in the following diverge functions and will terminate the run if the required link type is not found.

<u>Diverge Function No. 1</u> -- This function is for the diverge at the entrance to the station. If the vehicle's next stop is the station being simulated, the input ramp is found and made the first item in the list to be returned. Next, a bypass link is found and added to the list. If the vehicle is not stopping at the station, bypass link alone is listed. See Figure 6-4.

<u>Diverge Function No. 2</u> -- The function is for use at the end of the input ramp, modal input before processing, and storage to input link. If the sink of the vehicle is the modal output before processing, then this link is found and listed.

Use: End of approach link to station

Ordering:

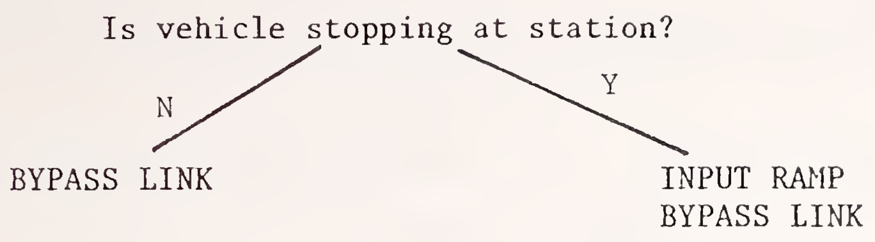


Figure 6-4a. Diverge Function #1

Use: End of Input ramp

End of modal-input-before-processing link

End of storage-to-input link

Ordering:

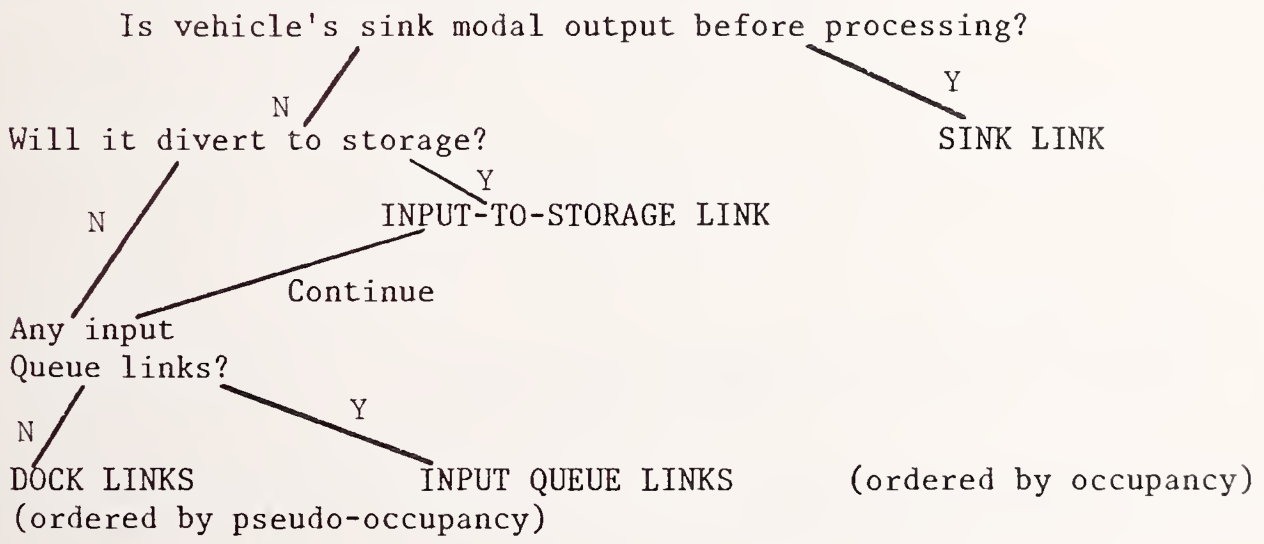


Figure 6-4b. Diverge Function #2

Use: End of dock links (after board)

Ordering:

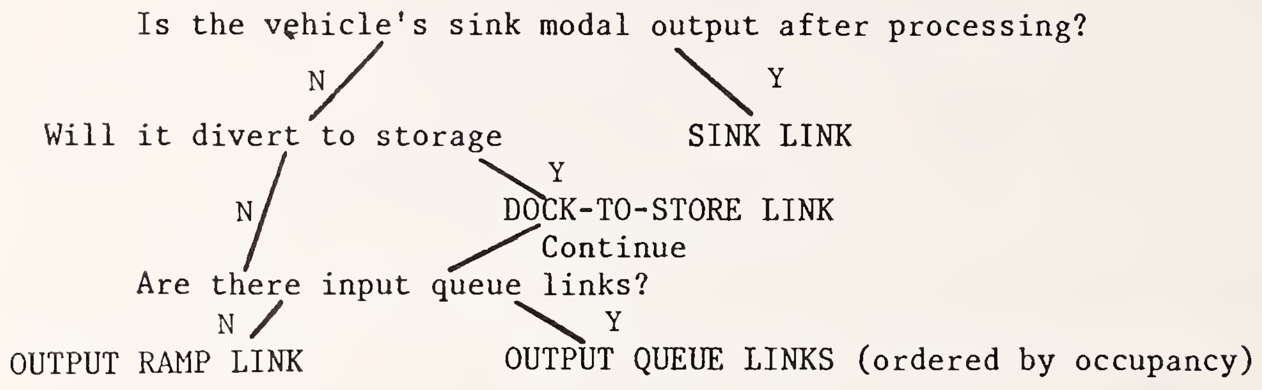

Is the vehicle's sink modal output after processing?
N
Y
Will it divert to storage
SINK LINK
N
Y
DOCK-TO-STORE LINK
Continue
Are there input queue links?
N
Y
OUTPUT RAMP LINK
OUTPUT QUEUE LINKS (ordered by occupancy)

Figure 6-4c. Diverge Function #3

Use: End of Storage

Ordering:

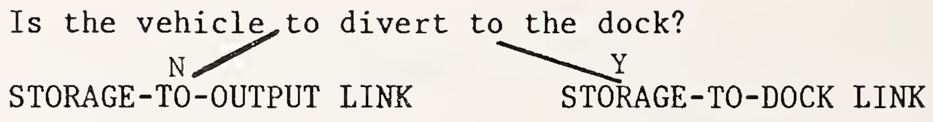


Figure 6-4d. Diverge Function #4

Use: As_applicable

Ordering: DOWNSTREAM LINKS (by occupancy)

Figure 6-4e. Diverge Function #5

Use: As applicable (generally before docking links)

Ordering: DOWNSTREAM LINKS (by pseudo-occupancy)

Figure 6-4f. Diverge Function #6

Otherwise, the input queue links are found and ordered by occupancy. If none were found then dock links are found and ordered by pseudo-occupancy (the number of blocked positions on the link). If the vehicle is to divert to storage, then this list of input queue links or dock links is prefaced by an input-to-storage link. See Figure 6-4.

Diverge Function No. 3 -- This function is for use at the end of dock links (after the board event). If the vehicle's sink is modal output after processing, then this link type is found and listed.

Otherwise, the output queue links are found and ordered by occupancy. If none are found then an output ramp found. If the vehicle is to divert to storage then a dock-to-store link is found and inserted above the list of output queue or ramp links. See Figure 6-4.

Diverge Function No. 4 -- This function is for use at the end of the storage link. If the vehicle is to divert to the dock, then a store-to-dock link is found and listed alone. Otherwise, a store-to-output link is found and listed alone. See Figure 6-4.

Diverge Function No. 5 -- This function orders the downstream links by occupancy regardless of type. See Figure 6-4.

Diverge Function No. 6 -- This function orders the downstream links by pseudo-occupancy regardless of type. See Figure 6-4.

6.2.47.5 PDL - See Appendix A.

6.2.47.6 Decision Tables and Algorithms

6.2.48 SMDIVO

6.2.48.1 Identification

- o SMDIVO - Order Station Links for Diverge Function
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.48.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SLN	KMSL	I*4	LIST OF LINKS TO BE ORDERED
IND	-	I*4	INDICATOR AS TO WHETHER ORDERING SHOULD BE BY OCCUPANCY OR PSEUDO-OCCUPANDY

6.2.48.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SLN	KMSL	I*4	List of station links found in search
TEMP	-	I*4	Intermediate variable used in bubble sort

6.2.48.4 Description - Order a list of station links by occupancy or pseudo-occupancy. This routine does a bubble sort on the station links in the input list based on either their occupancy or pseudo-occupancy. The links with the minimum occupancy will be the first on the returned list.

6.2.48.5 PDL - See Appendix A.

6.2.48.6 Decision Tables and Algorithms - None.

6.2.49 SMDIVS

6.2.49.1 Identification

- o SMDIVS - Search for Link of Specific Type for Diverge Function
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.49.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	–	I*4	ID OF THE VEHICLE BEING PROCESSED
TYPE	–	I*4	TYPE ('SLTYPE') OF LINK TO BE SEARCHED
ARMED	–	I*4	INDICATOR AS TO WHETHER OR NOT SIMULATION SHOULD STOP IF AT LEAST ONE LINK OF GIVEN TYPE IS NOT FOUND
SLN	KMSL	I*4	(OUTPUT) LIST OF LINKS OF GIVEN TYPE THAT ARE IMMEDIATELY DOWNSTREAM OF THE CURRENT LINK OF THE VEHICLE BEING PROCESSED

6.2.49.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SLN	KMSL	I*4	List of station links found in search
NEXT	-	I*4	Pointer to next downstream link

6.2.49.4 Description - SMDIVS forms a list of all links of a given type that are immediately downstream of a given link. The list of links downstream of the current link of the vehicle being processed is scanned for links of the requested type. As such links are found they are noted in the output list. If there are none found and an input indicator is set, the simulation terminates.

6.2.49.5 PDL - See Appendix A.

6.2.49.6 Decision Tables and Algorithms - None.

6.2.50 SMENTR

6.2.50.1 Identification

- o SMENTR - Entrain Following Vehicles to a Lead Vehicle
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.50.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	-	I*4	THE LEAD VEHICLE ON THE LINK (IT CAN'T BE QUEUED) TO WHICH FOLLOWING VEHICLES WILL BE ENTRAINED.

6.2.50.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
GHEAD	-	I*4	TAIL TO A TEMPORARY STATION LINK MEMBERSHIP LIST
GHEADZ	-	I*4	TAIL TO THE TRAIN CHAIN

6.2.50.4 Description - The purpose of SMENTR is to entrain as many vehicles as possible (up to a user specified limit) to the head vehicle on the link at launch time, provided they have the same next stop. For each queued vehicle which is either done or awaiting launch and which is immediately behind the head vehicle, chain it to the lead vehicle until either the limit on the number of vehicles in a train is reached or the vehicles have different next stops. For each entrained vehicle the train length of the head vehicle is increased by one, the trailing vehicle is chained to the one in front of it, and the trailing vehicle is removed from the membership chain of the current link.

6.2.50.5 PDL - See Appendix A.

6.2.50.6 Decision Tables and Algorithms - None.

6.2.51 SMEVM

6.2.51.1 Identification

- o SMEVM - Empty Vehicle Management
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.51.2 Argument Dictionary - None.

6.2.51.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NEED	-	I*2	INDICATOR THAT VEHICLE IS NEEDED AT ANOTHER STATION: 1 = NEEDED 2 = NOT NEEDED
ONE	-	I*2	INTEGER*2 VERSION OF THE CONSTANT 1
TWO	-	I*2	INTEGER*2 VERSION OF THE CONSTANT 2

6.2.51.4 Description - The purpose of SMEVM is to determine whether an empty vehicle is to be sent to local storage or out of the station. If policy dictates that all vehicles be sent out of the station, an indicator associated with the vehicle is set to indicate to the diverge function (SMDIVF) that the vehicle is not to be diverted to local storage.

If policy dictates that an attempt should be made to send the vehicle to local storage, then a test is made to see if the link representing local storage is at capacity. If it is at capacity, then the vehicle is marked not to divert into local storage. If space is available on the storage link, then a test is made to determine if there is a simulated need for the vehicle at another station. This test is made by randomly sampling a user specified distribution of the vehicle being needed elsewhere. The vehicle is then marked accordingly to divert to local storage or not.

6.2.51.5 PDL - See Appendix A.

6.2.51.6 Decision Tables and Algorithms - None.

6.2.52 SMGDIP4

6.2.52.1 Identification

- o SMGDIP4 - Generalized Data Input Package - Define Layout of Input Common Areas
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.52.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
GDIP4:			
NAME	-	R*8	PARAMETER NAME
FMT	-	R*8	FORMAT OF DATA
IRAL	-	I*4	FIRST DIMENSION LOWER BOUND
IRAH	-	I*4	FIRST DIMENSION UPPER BOUND
IRBL	-	I*4	SECOND DIMENSION LOWER BOUND
IRBH	-	I*4	SECOND DIMENSION UPPER BOUND
IRCL	-	I*4	THIRD DIMENSION LOWER BOUND
IRCH	-	I*4	THIRD DIMENSION UPPER BOUND
IRDL	-	I*4	FOURTH DIMENSION LOWER BOUND
IRDH	-	I*4	FOURTH DIMENSION UPPER BOUND

6.2.52.3 Local Variable Dictionary - None.

6.2.52.4 Description - SMGDIP4 (GDIP) is the Generalized Data Input Package. The GDIP is a collection of routines that provides the user with the capability of reading data into COMMON variables with a minimum of programming effort. GDIP eliminates the need for pre-initializing data areas prior to program execution and provides the ability to change data formats without requiring modification to embedded read statements contained in executable program modules. The GDIP provides the following features, which are controlled by the user at program execution time:

1. Any rectangular section of any array may be modified.

2. The data items to be loaded are on input cards of the user's own format, which is specified at execution time.

3. A "repetition factor" allows the loading of consecutive data elements with a single value specification.

The package can accommodate arrays having up to four subscripts (dimensions).

When the GDIP is invoked by a CALL, input cards supplied by the user of the CALLING program are read, and the desired data loading functions are performed. Two categories of functions are provided: end-input and read-data. The card formats for these functions are defined in the User's Manual.

The statement:

```
CALL NDBOR
```

invokes GDIP. This statement may be invoked as desired, but is typically issued during program initialization. However, in the DSM, GDIP is invoked each time asynchronous data initialization is requested to modify existing simulation data definitions. Each such CALL to GDIP results in one or more input cards being read from the standard system input data stream (FT05F001).

Figure 6-5 illustrates a sample ALC routine, by which the necessary definitions of variables and COMMON areas are made. This routine, when assembled, provides addressability of each data item in the common areas. Any format modifications required are easily accommodated by merely respecifying the definition data in the routine. No modifications are required to the I/O portion of GDIP or the invoking program. The significant features of this routine are:

1. The statement:

   ```
   NODIMENS 4
   ```

   defines the maximum number of dimensions on any array to be four. This value may be changed only with corresponding changes to NDBOR and by providing new routines named GDIPFn, GDIPHn, and GDIPXn (internal routines currently provided with n = 4) to allow data formatting into higher dimensioned arrays.

2. Each COMMON area requiring GDIP data loading is defined by a set of cards, consisting of the following:

   a. A card to name the COMMON area, in the form:

      ```
      common-name CSECT
      ```

```
GDIP     TITLE 'GENERALIZED DATA INPUT PROGRAM'
         NODIMENS 4
         SPACE 5
         LCLA  &KML,&KMS,&KMV,&KMX,&KMT,&KMCRT,&KMVEAT,&KMCRP,&KMFLAG,  X
               &KMS1,&KMM,&KM1
         LCLA  &KMCLTA
&KML     SETA  100
&KMS     SETA  40
&KMV     SETA  2500
&KMX     SETA  3500
&KMT     SETA  2500
&KMCRT , SETA  400
&KMVEAT  SETA  200
&KMCRP   SETA  61
&KMFLAG  SETA  300
&KMCLTA  SETA  1000
&KMM     SETA  40
&KM1     SETA  100
&KMS1    SETA  &KMS+1
         SPACE 5
LNKCOM   CSECT
         COMN  F,LTIMT,(&KML),LTIMHZ,,LTIMRE,
         COMN  H,LSPEED,,LCAP,(&KML),LOCC,(&KML),LENTY,(&KML,2),        X
               LSNEXT,(&KML),LEQHD,(&KML),LSLT,(&KML,&KMS),             X
               LDIST,(&KML),LMERGN,(&KML)
         COMN  X,LFAIL,(&KML),LPRIOR,(&KML),LFHZ,(&KML)
         SPACE 3
STNCOM   CSECT
         COMN  F,STIMHZ,,STIMEN,,STIMID,,STIMIS,,STIMDS,,STIMSD,,       X
               STIMEM,,STIMDE,,STIMEX,,SUSTN,(&KMS)
         COMN  H,SCAPIA,(&KMS),SCAPDA,(&KMS),SCAPSA,(&KMS),             X
               SCAPOA,(&KMS),SOCCIA,(&KMS),SOCCDA,(&KMS),SOCCSA,(&KMS),X
               SOCCOA,(&KMS),SQTI,(&KMS1,SQTD,(&KMS),SQTSS,(&KMS),      X
               SQTSD,(&KMS),SQTSE,(&KMS),SEQHD,(&KMS),SQTTRP,(&KMS),    X
               SILINK,(&KMS),SELINK,(&KMS),STRPT,(&KMS),STRPU,(&KMS),   X
               SALT,(&KMS)
         COMN  X,SFHZ,(&KMS)
         SPACE 3
SYSCOM   CSECT
         COMN  F,KNL,,KNS,,KNV,,VACTIV,,TACTIV,,KNCRT,,KNVEAT,,KNCRP,,  X
               CSAMPL,,CSIZE,,CLOOP,,CLOCK,,XFIRST,,CNFEC,,             X
               KSEED,,KWTIMW,,KNM,,KN1,,NUCLK,,KTSERV,
         COMN  F,KSATNO,,KTHRP,,KTHRN,,KROWL,
         COMN  H,KWTTAB,(&KM1,&KMM)
         COMN  X,CFLAG,(&KMFLAG),KSTATU,
         SPACE 3
TRPCOM   CSECT
         COMN  F,TAVAIL,,TTIME,(&KMT)
         COMN  H,TORIG,(&KMT),TDEST,(&KMT),TPASS,(&KMT),TCHAIN,(&KMT),  X
               TCASGN,(&KMS,&KMS)
         SPACE 3
VEHCOM   CSECT
         COMN  F,VTIME,(&KMX),VEACP,(&KMVEAT),VAVAIL,,VCLAST,(&KMCRP)
         COMN  H,VGOTO,(&KMX),VCHAIN,(&KMX),VCURR,(&KMX),VCASE,(&KMV),  X
               VOT,(&KMV1,VPASS,(&KMV),VNXSTN,(&KMV),VCYCNO,(&KMV),     X
               VCYCPO,(&KMV),VCLIST,(&KMCRT),VCPTR,(&KMCRP),            X
               VEARS,(&KMVEAT),VEAP,(&KMS1),VCAP,,                      X
               VCYCHW,(&KMCRP),VNVCYC,(&KMCRP)
         COMN  X,VT~~B~~OS,(&KMV),VQUED,(&KMV)
FECCOM   CSECT
         COMN  F,CLPASE,,CLBIG,,CLPOS,,CLSMAL,,CLSCAN,,CLMINI,,CLNUM,,  X
               CLSIZE,,CLSTAT,(3,10)
         COMN  H,CLTABL,(&KMCLTA)
SN2COM   CSECT
         COMN  F,SCAPSP,(&KMS)
V2COM    CSECT
         COMN  F,VCLASM,(&KMCRP,&KMS)
         COMN  H,VULTD,(&KMV1,TSST,(&KMS,&KMS),VCTIME,(&KMCRT1
         COMN  X,TCNIS,(&KMS,&KMS)
         ENDEFS
ERRORXIT CALLS ER,(F'888',C'ARRAY NAME NOT FOUND IN TABLE;',F'2')
         8     RETURN
         END
```

Figure 6-5. GDIP Common Data Definition

b. One or more cards to define the variables in COMMON area. These definitions must be in precisely the same order as in the corresponding FORTRAN COMMON statement since they are used to define a data map of each variable in the common area such that addressability can be established to any data position. The format of each card is:

```
COMN1,name,dimensions, (name-2,dimensions-2, . . . .)
```

Up to 20 variables may be defined per card, provided their data item lengths are the same. The field "1" must be one of the following:

<u>1</u>	<u>Variable type(s)</u>
F	REAL*4, INTEGER*4, and LOGICAL*4
H	INTEGER*2
X	LOGICAL*1

The dimensions field is of the form:

```
(first-dimension, . . ., fourth-dimension)
```

If a variable has fewer than four dimensions, only the necessary ones are given. If the variable has no dimensions (i.e., is unsubscripted), then this field is null. However, the comma must always be supplied, as illustrated by Figure 6-5.

3. The statement:

```
ENDEFS
```

marks the end of all the definitions and causes the data mapping to be established. Currently, up to 200 variables may be defined.

4. The pair of statements:

```
ERRORXIT . . .

          B RETURN
```

defines how errors are to be handled. If GDIP encounters an undefined variable in the input stream, control branches to ERRORXIT. The branch to RETURN causes GDIP to read and process the next input card.

5. The statement:

```
END
```

terminates the definition routine code.

6.2.52.5 PDL - See Appendix A.

6.2.52.6 Decision Tables and Algorithms - None.

6.2.53 SMLTIM

6.2.53.1 Identification

- o SMLTIM - Launch Time Delay Due to Schedule
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.53.2 Argument Dictionary - None.

6.2.53.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
B2	-	I*4	BOARDING DELAY DUE TO WAITING FOR THE SCHEDULED DEPARTURE TIME.

6.2.53.4 Description - The purpose of SMLTIM is to determine the time delay that the vehicle should wait until the scheduled departure time. In the case of scheduled service, a test is made to see if fixed departure times are used or the vehicles are to depart midway between the previous vehicle on the route and the following vehicle. In the case of fixed departure times, the time the current vehicle on the route is to leave is determined by adding the time the last vehicle on the route was scheduled to leave and the route headway. If this time has already passed, then the delay associated with waiting until scheduled departure time is set to zero. If the time has not already passed, then this delay is set to the difference between the current clock and the desired time.

In the case of scheduling departures midway between the time the previous vehicle on the route and the following vehicle, the time the current vehicle on the route is to leave is determined by computing the average of the time the next should leave and the time the last did leave. If this time has already passed, the delay is set to zero. Otherwise, it is set to the difference of the computed time and the current value of the clock. Next the time the last did leave is set to the sum of the value of the clock and the delay. Then the time the next should leave is computed by increasing its previous value by the route headway.

6.2.53.5 PDL - See Appendix A.

6.2.53.6 Decision Tables and Algorithms - None.

6.2.54 SMNXST

6.2.54.1 Identification

- o SMNXST - Vehicle Next Stop Determination
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.54.2 Argument Dictionary - None.

6.2.54.3 Local Variable Dictionary - None.

6.2.54.4 Description - The purpose of SMNXST is to determine the next station at which occupied vehicles will stop and turn empty vehicles over to empty vehicle management for the store-leave decision. For demand responsive service, a test is first made to determine if the onboard trip queue is empty. If it is empty, empty vehicle management (SMEVM) is run. Otherwise, the vehicle is marked so as not to divert into storage and the next stop is to be the destination of the first trip on the vehicle. In the case of scheduled service, next stop is not required since it is not used to support entrainment.

6.2.54.5 PDL - See Appendix A.

6.2.54.6 Decision Tables and Algorithms - None.

6.2.55 SMRNG

6.2.55.1 Identification

- o SMRNG - Generate Uniformly Distributed Random Numbers
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.55.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
MRSEED	—	I*4	RANDOM NUMBER SEED
MRANDN	—	R*4	RANDOM NUMBER BETWEEN 0 AND 1

6.2.55.3 Local Variable Dictionary - None.

6.2.55.4 Description - This routine is used to generate a random number that is uniformly distributed between 0 and 1.

6.2.55.5 PDL - See Appendix A.

6.2.55.6 Decison Tables and Algorithms - None.

6.2.56 SMRSEL

6.2.56.1 Identification

- o SMRSEL - Randomly Select Point from Cumulative Distribution
- o IBM/FSD ·· July 1, 1977
- o PARAFOR

6.2.56.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
DDISTR	DEND	R*4	ARRAY CONTAINING A CUM. PROB. DIST.
DSTRT	–	I*2	STARTING ENTRY IN DDISTR ARRAY
DEND	–	I*2	ENDING ENTRY IN DDISTR ARRAY
DKSEED	–	I*4	(INPUT AND OUTPUT) RANDOM NUMBER SEED (>=3)
DSLECT	–	I*2	(OUTPUT) PROBABILITY ENTRY SELECTED

6.2.56.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
DRANDN	-	R*4	Random number returned by SMRNG

6.2.56.4 Description - This routine is used to randomly select a point from a cumulative distribution. It does this by using SMRNG to generate a random number between 0 and 1 and then searching the cumulative distribution until a point on it larger than the random number is found. The index of that point is returned.

6.2.56.5 PDL - See Appendix A.

6.2.56.6 Decision Tables and Algorithms - None.

6.2.57 SMTABQ

6.2.57.1 Identification

- o SMTABQ - Prepare a Trip for Boarding
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.57.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
T	–	I*4	TRIP JUST FINISHED WITH ALL TRIP LINKS OR JUST TRANSFERRED OFF A VEHICLE.

6.2.57.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
T	–	I*4	TRIP BEING PROCESSED
TR	–	I*4	TRIP BEING PROCESSED
TIM	–	I*4	DELAY UNTIL EMPTY VEHICLE ARRIVES
V	–	I*4	LEAD VEHICLE BEING PROCESSED
VID	–	I*4	VEHICLE BEING PROCESSED
VBEFOR	–	I*4	LEAD VEHICLE PREDECESSOR OF TRAINS IN THE BOARD EVENT
LEADV	–	I*4	LEAD VEHICLE OF A TRAIN
VEH	–	I*4	VEHICLE IN TRAIN IN THE BOARDING EVENT
VEHBEF	–	I*4	PREDECESSOR OF VEHICLE IN TRAIN IN THE BOARD EVENT
QHEADZ	–	I*4	TAIL TO TRAIN CHAIN
STPTR	–	I*4	POINTER TO A STATION IN THE VEHICLE'S STATION ROUTE LIST
K	–	I*2	SUBSCRIPT TO THE EMPTY VEHICLE DELAY DISTRIBU- TION
ONE	–	I*2	INTEGER*2 VERSION OF THE CONSTANT 1
TWO	–	I*2	INTEGER*2 VERSION OF THE CONSTANT 2
FOUND	–	L*1	INDICATES VEHICLE CAN SERVICE THE TRIP BEING PROCESSED: T===>FOUND F===>NONE FOUND

```
MATCH      -    L*1    INDICATES WHETHER OR NOT THE VEH & TRIP ARE
                       COMPATIBLE:
                         1===>MATCH
                         2===>NO MATCH
CHECK      -    L*1    INDICATOR THAT VEHICLE HAS BEEN FOUND THAT
                       HAS SPACE FOR THE TRIP:
                         T = FOUND
                         F = NOT FOUND
```

6.2.57.4 Description - The purpose of SMTABQ is to get a vehicle moving to pick up a trip when the trip arrives at the boarding queue. Under certain circumstances a trip can immediately board a waiting vehicle. In the case of scheduled service, the arrival of a trip at the boarding queue causes it to actively seek out a vehicle which is undergoing boarding and is on the appropriate route and has space available. (See SMBRD for methodology.) In the case of demand responsive service, the user has the option of specifying any subset and any ordering of up to three places to "look" for an empty vehicle to service the trip. These three places are:

1. From local storage

2. From eligible user-specified station links upstream of the dock

3. From elsewhere in the network (always successful since it generates an empty vehicle).

In the case of trying to get a vehicle from local storage, all the vehicles on the link representing storage are searched until one is found that is still in the stored state. (There could be vehicles on the storage ramp that are queued waiting to depart but cannot due to congestion.) When a vehicle is found, if it is at the head of the storage link (i.e., no other vehicles in front of it), its queuing reason is set to indicate that it is done with processing on the link it is on and queued due to congestion and then SSPMAC is used to schedule a prompt on that link to get the vehicle moving. If it is not at the head of the storage link, its queuing reason is set to indicate that it is done with processing on the link it is on and waiting for the vehicle in front of it to leave before proceeding. If a vehicle is found this way, success is signalled and SMTABQ exited.

In the case of trying to get a vehicle from the station links upstream on the dock, SMTABQ searches all vehicles on the trains on all requested links until it finds an unreserved empty vehicle. When it finds one it marks it as reserved and signals success.

In the case where either or both of the above requested options fail or when the user specifies this methodology, SMTABQ will simulate the fetching of an empty vehicle from elsewhere in the network by generating an empty vehicle arriving on the guideway upstream of the station. It does this by getting an available transaction from the available list, initializing it to the characteristics of an empty vehicle, determining the delay until it appears upstream of the station from a user specified distribution of delay and a random number, and schedules the vehicle to arrive upstream of the station at that selected delay time in the future.

6.2.57.5 PDL - See Appendix A.

6.2.57.6 Decision Tables and Algorithms - None.

6.2.58 SSASAV

6.2.58.1 Identification

- o SSASAV - Initialize System Status Area Words
- o IBM/FSD - July 1, 1977
- o ASM

6.2.58.2 Argument Dictionary - None.

6.2.58.3 Local Variable Dictionary - None.

6.2.58.4 Description - The purpose of SSASAV is to initialize system status area words. It serves at linkage edit time to capture the address of the start of the input common area, start of model common area, and end of common areas. See discussion of SANTSA.

6.2.58.5 PDL - See Appendix A.

6.2.58.6 Decision Tables and Algorithms - None.

6.2.59 SSLEAV

6.2.59.1 Identification

- o SSLEAV - Process a Vehicle/Train Leaving a Station Link
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.59.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	–	I*4	VEHICLE OR LEAD VEHICLE OF A TRAIN WHICH IS LEAVING THE STATION LINK

6.2.59.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NEXTV	–	I*4	VEHICLE BEHIND V IN THE STATION LINK MEMBER-SHIP LIST
SL	–	I*2	V'S CURRENT STATION LINK

6.2.59.4 Description - SSLEAV performs processing associated with a vehicle leaving a station link. When it has been guaranteed that the next link can be entered, SSLEAV decreases the link occupancy (and pseudo-occupancy if necessary) of the current link by the length of the train and dequeues the train from the link's membership list thereby facilitating the departure of the vehicle/train.

SSLEAV next tries to get the following vehicle moving if it had been queued. If it has completed events on the link, it is prompted; if it is waiting to start its launch event, it is modeled.

Finally, SSLEAV tries to get vehicles on upstream links moving since the leaving vehicle might have made sufficient space available to accommodate them.

6.2.59.5 PDL - See Appendix A.

6.2.59.6 Decision Tables and Algorithms - None.

6.2.60 SSMOD

6.2.60.1 Identification

- o SSMOD - Model the Vehicle/Train on its Current Station Link
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.60.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	-	I*4	VEHICLE TO BE MODELED ON THE STATION LINK

6.2.60.3 Local Variable Dictionary - None.

6.2.60.4 Description - The purpose of SSMOD is to direct the use of three other code segments: SSMODA, SSMODN, and SSMODB, thereby controlling the transitional processing from one station link event to another. SSMOD is a code segment whose function is to direct the use of three other code segments:

1. SSMODA -- Perform processing associated with a vehicle's station link event immediately after that vehicles comes off the FEL for that event.

2. SSMODN -- Perform processing to determine the next event to occur to the vehicle.

3. SSMODB -- Perform processing associated with the vehicle's next station link event and put the vehicle on the FEL for that event.

These three code segments are commonly called in the order after-next-before. This structure gives the simulator the flexibility to represent any station link that can be derived from the canonical station link. However, in the case when the vehicle is entering the link for the first time, SSMODA is skipped. Also, in the case where the vehicle has completed all events on the current link, SSMODB is skipped. SSMODA is also skipped when a vehicle has been waiting to start the launch event but is unable to, and so control has been transferred back to SAMAIN with the vehicle left in a queued state. In this case, the after-time-segment processing that SSMODA does has already been performed and should not be performed again. Additionally, SSMODB is also skipped when SSMODN had to queue the vehicle awaiting launch due to another vehicle being in front of it.

6.2.60.5 PDL - See Appendix A.

6.2.60.6 Decision Tables and Algorithms - None

6.2.61 SSMODA

6.2.61.1 Identification

- o SSMODA - Vehicle Processing After a Station Link Event
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.61.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	–	I*4	INDIVIDUAL VEHICLE OR LEAD VEHICLE OF THE TRAIN

6.2.61.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
VID	–	I*4	VEHICLE BEING PROCESSED IN THE TRAIN
TRIP	–	I*4	TRIP BEING PROCESSED
DTIME	–	I*4	DELTA TIME BETWEEN TIME THE TRIP LAST CAME OFF THE FEL AND THE CURRENT TIME
TID	–	I*4	TRIP BEING PROCESSED
VEH	–	I*4	LEAD VEHICLE IN THE TRAIN
QHEADV	–	I*4	TAIL TO THE TRAIN CHAIN
VBEFOR	–	I*4	PREDECESSOR VEHICLE IN THE TRAIN
SL	–	I*2	VEHICLE'S CURRENT STATION LINK
CHECK	–	L*1	NOT USED

6.2.61.4 Description - The purpose of SSMODA is to do processing after the time segment that the vehicle has just spent on the FEL. After the headway zone travel event, the headway zone flag is turned off so as to indicate to other vehicles that the link can now be entered. Then in order to insure that any vehicle that was waiting to enter but could not because the headway zone was occupied, SSPMAC is run to schedule a prompt to get vehicles moving on upstream station links.

After the main travel event, no processing is necessary.

After the deboard event, the following processing is done for each vehicle in the train. First, for every trip that is to deboard and leave the system (as determined by SMBRD when SSMODB was run), the trip is dequeued from the deboard and leave list, trip statistics are collected, and the trip is scheduled for the deboard exit walk time. Second, for each trip that is to deboard and transfer (as determined by SMBRD when SSMODB was run), the trip is dequeued from the deboard and transfer list, statistics are collected and it is scheduled for the transfer exit walk.

After the board event, the following processing is done for each vehicle in the train. For each trip that is to board that vehicle (as determined by SMBRD when SSMODB was run), the trip is dequeued from the board list, enqueued onto the onboard trip queue and the occupancy of the boarding queue is decreased by the size of the trip. The SMNXST is run to determine the next stop of the vehicle. Then SUPMAC is run to schedule prompt to insure that any trip that has been waiting in the turnstile area to enter the boarding link but could not (since the boarding link was at capacity) does not enter since some trips have left the boarding queue.

After the joint event the processing done after the deboard and board events is done.

After the store event, no processing is necessary.

After the launch event, SMENTR is run when the entrainment policy is in effect in order to attach other waiting vehicles to the launched one.

6.2.61.5 <u>PDL</u> - See Appendix A.

6.2.61.6 <u>Decision Tables and Algorithms</u> - None.

6.2.62 SSMODB

6.2.62.1 Identification

- o SSMODB - Vehicle Processing Before a Station Link Event
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.62.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	–	I*4	INDIVIDUAL VEHICLE OR LEAD VEHICLE OF THE TRAIN WHOSE NEXT EVENT HAS BEEN DETERMINED AND WHO NEEDS "BEFORE" PROCESSING DONE FOR THAT EVENT.

6.2.62.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
B1	–	I*4	BOARDING DELAY BASED ON NUMBER OF TRIPS BOARDING
B2	–	I*4	BOARDING DELAY BASED ON THE SCHEDULED DEPARTURE TIME
B3	–	I*4	BOARDING DELAY BASED ON THE FORWARD VEHICLE'S BOARDING DURATION
D1	–	I*4	DEBOARDING DELAY BASED ON NUMBER OF TRIPS DEBOARDING
DBBMAX	–	I*4	MAXIMUM DEBOARD/BOARD TIME
VEH	–	I*4	LEAD VEHICLE IN THE TRAIN
VID	–	I*4	VEHICLE BEING PROCESSED
NOT	–	I*4	NOT USED
DBTOTP	KMTLEN	I*2	NUMBER OF PASSENGERS DEBOARDING THE VEHICLE
ORTTIM	–	I*4	OUTPUT RAMP TRAVEL TIME
TIM	–	I*4	THE TIME THE VEHICLE WILL SPEND ON THE FEL
T1	–	I*4	DELAY DUE TO MERGES IN THE REST OF THE NETWORK
T2	–	I*4	DELAY DUE TO LOCAL MERGES
THWAY	–	I*4	HEADWAY TRAVEL TIME
TLMAX	–	I*4	HEADWAY TIME REQUIRED ON THE BYPASS LINK FOR THE LONGEST POSSIBLE TRAIN

TMAX	–	I*4	TIME A VEHICLE WILL SPEND ON THE FEL FOR THE BOARD/DEBOARD/JOINT EVENT
TMEAN	–	R*4	MEAN TIME OF A NORMAL DISTRIBUTION
T	–	R*4	REAL VARIABLE CONTAINING THE DEBOARD/BOARD/ JOINT EVENT TIME
G	–	I*4	INDICATOR WITH A VALUE OF 1 WHEN THE CURRENT STATION LINK HAS A HEADWAY EVENT AND A VALUE OF 0 WHEN IT HAS NONE. USED TO MAKE TRAVEL TIME A FUNCTION OF HEADWAY TIME.
VBEFOR	–	I*4	PREDECESSOR VEHICLE ON THE STATION LINK MEMBERSHIP LIST.
ONE	–	I*2	INTEGER*2 VERSION OF THE CONSTANT 1
TWO	–	I*2	INTEGER*2 VERSION OF THE CONSTANT 2
K	–	I*2	SUBSCRIPT TO THE LOCAL MERGE DELAY DISTRIBUTION
SL	–	I*2	VEHICLE'S CURRENT STATION LINK
CHECK	–	L*1	TRUE INDICATES THAT THE SEARCH OF VEHICLES ON THE STATION LINK MEMBERSHIP LIST HAS HIT THE VEHICLE GOING ON THE FEL FOR THE BOARD OR JOINT EVENT.

6.2.62.4 Description - The purpose of SSMODB is to perform processing that is to be done before the time segment for which the vehicle is about to be put on the FEL. With the exception of the store event it also determines the amount of time that a vehicle is to spend on the FEL, and then actually puts the vehicle transaction on the FEL.

SSMODB consists of the processing to be performed for each of the seven events that can occur on a station link before a vehicle goes on the FEL for any one of those events.

In the case of traveling the headway zone, the headway zone flag is turned on to indicate to the other vehicles that this link cannot be entered until the flag is turned off. The headway zone travel time is then calculated from the form ax+b, where x is the number of vehicles in the train that the vehicle being processed is leading and a and b are user specified times. The vehicle being processed is then put on the FEL to remain there for that amount of time.

In the case of traveling the main body of a link, the travel time is computed as the difference between the user-supplied station link travel time and the headway zone travel time. Furthermore, this difference is multiplied by a user-supplied penalty factor used to degrade the link and then the resulting time is the time which the vehicle will spend on the FEL for the travel event. In addition to travel time, empty slots on the bypass link must be updated whenever the vehicle is traveling on the bypass link and the local merge policy is in effect. The head of the train beginning travel delimits the end of the open slot in front of that train and the end of the train delimits the start of the slot behind the train. Hence, the table of empty slots on the bypass link, used before launch by vehicle's attempting local merge, is updated to reflect the presence of a new vehicle traveling the bypass link.

In the case of deboarding, SMDETR is run when the entrainment/detrainment policy is in effect and the vehicle beginning the deboarding event is actually the lead vehicle of the train. Once that is done, the following processing is done for each vehicle before it is separately scheduled to spend time on the FEL.

SMDBRD is run to determine the total number of passengers that will be deboarding the vehicle at the station. This count is multiplied by the standard deboarding time per passenger and then added to a deboarding time constant. The resultant "mean" deboarding time for the vehicle is then randomized using a user-specified standard deviation and the result is the time that the vehicle will spend deboarding.

When the deboarding time for a train is required, the above procedure is followed for each vehicle in the train and the time required for the train is the maximum of the times required for each vehicle in the train. The lead vehicle of the train is then scheduled to spend that maximum time in the FEL.

In the case of boarding, SMBRD is run to determine the number of trips that will be boarding either an individual vehicle, or each vehicle in a train. Once the counts are made, borading time is computed for each vehicle individually. The count for a vehicle is multiplied by the standard boarding time per passenger and then added to a boarding time constant. The resultant "mean" boarding time for the vehicle is then randomized using a user-specified standard deviation and the result is the time that the vehicle will spend boarding trips. When the boarding time for a train is required, the maximum of the times for each vehicle in the train is used.

When the service policy is demand responsive, the vehicle is ready to be scheduled to spend the above boarding time on the FEL. When the service policy is scheduled, it is possible that the vehicle will spend more time in the boarding event than that computed for boarding. Such is the case when SMLTIM is run and the schedule delay is found to be greater than the boarding time. The greater of the two times is chosen. In addition, when the vehicle/train is behind another vehicle/train which is in the board event, the following vehicle's time in boarding will extend at least as long as that of the preceding one so that trips can continue to board it while it is held up by the vehicle/train in front of it. Hence, in scheduled service, the maximum of the following three times is the time for which the vehicle/ train is scheduled to spend time on the FEL for the board event: passenger boarding time, schedule delay, and delay due to the preceeding vehicle held up in boarding.

In the case of the joint event, a combination of processing done for the deboard event and the board event is done. As in deboard processing, trains are detrained (using SMDETR) when necessary and the number of passengers getting off each vehicle is determined (using SMDETD). As in board processing, the number of passengers getting on each vehicle is determined (using SMDBRD). The calculations to determine the time to spend in the joint event do, however, differ. First the deboard and board times are computed separately using equations that contain interaction terms. The deboard time equals a + b + c + d where a equals the deboard time per passenger times the number of passengers deboarding; b equals an interaction constant times the product of trips boarding and trips deboarding; c equals a constant times the number of trips boarding; and d equals a deboarding constant. The board time equation is of the same form, however the coefficients for each term are user-specified specifically for the boarding case. The two "mean" times, thus found, are then randomized using separate user-specified standard deviations. The resultant times are then compared. If the randomized deboard time is greater than the randomized board time plus the waiting delay (for deboard to get a 'head-start'), then the deboard time becomes the joint time. Otherwise, the board time plus that delay is the joint time.

When the joint time for a train is required, the above procedure is followed for each vehicle in the train and the time required for the train is the maximum of the times required for each vehicle in the train.

As in the case in the board event, when the service policy is demand responsive, the vehicle is ready to be scheduled to spend the joint event time on the FEL. However, when the service policy is scheduled, it is possible that the vehicle will spend more time in the joint event than that already computed. Hence, in joint scheduled service as in the board scheduled service processing, the maximum of the following three times is the time for which the vehicle/train is scheduled to spend time on the FEL for the joint event: joint time, schedule delay, and delay due to the preceeding vehicle being held up in the joint event.

For the deboard, board and joint events, if a trip and vehicle event file has been requested a record is written for each trip deboarding to leave, deboarding to transfer, and boarding.

In the case of a store event, the vehicle is simply marked as queued for storage and not put on the FEL because it is not known when it will be unstored.

In the case of the launch event, the delay time due to merges in the rest of the network is selected randomly from the user-specified network merge delay distribution. This is the only launch delay when the local merge policy is not in effect. However, when local merging is done, an attempt is also made to find a slot on the bypass link to accommodate the vehicle. When a slot cannot be found, retry is attempted after a suitable time delay which includes the delay due to merges in the rest of the network. The second and subsequent times local merging is attempted, delays due to merges in the rest of the network are not considered.

Under two circumstances local merge delay is retried: when the bypass link has no slots, and when a vehicle is queued at the end of the bypass link. When the bypass link has no slots, the vehicle awaiting launch is required to wait a minimum delay time of t until a gap, if created now, were to reach a point on the bypass link such that a vehicle starting to travel the output ramp t time later would be able to fit into that slot. When a vehicle is queued at the end of the bypass link, launch is retried a nominal time t later where t equals the difference between the time to travel the bypass link and that required to travel the output ramp.

Local merge delay is found when a table of slots on the bypass link has been checked and a slot that can accommodate the vehicle is found far enough from the end of the bypass link such that a vehicle could have time to travel the output ramp before merging with the slot. The table of slots on the bypass link must be updated to reflect the loss of a slot when one has been used. To be considered eligible, all slots must be greater than or equal to the maximum headway required on the bypass link. Hence, any length train can be merged into an eligible slot.

6.2.62.5 PDL - See Appendix A.

6.2.62.6 Decision Tables and Algorithms - None.

6.2.63 SSMODN

6.2.63.1 Identification

- o SSMODN - Vehicle Next Station Link Event
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.63.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	–	I*4	VEHICLE OR LEAD VEHICLE OF A TRAIN, WHOSE NEXT STATION LINK EVENT IS TO BE DETERMINED

6.2.63.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SL	–	I*2	VEHICLE'S CURRENT STATION LINK
WASQD	–	I*2	REASON THE VEHICLE WAS FORMERLY QUEUED
NXEVP	–	I*2	POINTER TO THE NEXT EVENT IN THE SL EVENT LIST
EVFND	–	L*1	USED TO INDICATE: F===>SKIP OVER DEBOARD AND BOARD EVENTS IN THE CASE OF AN ON-LINE STATION WHERE THE VEHICLE IS NOT TO STOP T===>DO NOT SKIP OVER DEBOARD & BOARD EVENTS

6.2.63.4 Description - The purpose of SSMODN is to determine the next event to occur in the vehicle being processed. In the case where the link is being entered for the first time, the occupancy of the link is increased by the length of the train and the vehicle is enqueued onto the station link's membership list. When the link is one that contains deboard, board, or joint events (i.e., where the vehicle must be stationary and thus effectively blocks berths downstream of it when they become empty), a pseudo-occupancy is also maintained by increasing it by the train length. This pseudo-occupancy will be maintained equal to the capacity minus the number of available (upstream) berths.

In most cases the next event for the vehicle/train is the next event listed on the station link's event list. Such is the case with headway travel, main travel, and store, where no special processing is required.

In the cases of deboard and board events, a test is made to see if this is on online station and if the vehicle is not to stop here thus allowing these two events to be skipped when possible.

In the case of the launch events, a test is made to insure that the vehicle that is about to attempt launch is the head vehicle on the link (i.e., no other vehicles in front of it). If there are other vehicles in front of it, then the vehicle is marked as queued for that reason and control passes out of SSMODN.

When there are not more events to be processed on the link, a test is made to see if the vehicle had been waiting for launch or if it is back to retry launch. (Recall that the launch event is the last event on the canonical station link.) In this case, the next event is the launch event. Otherwise the vehicle is done with the station link.

6.2.63.5 PDL - See Appendix A.

6.2.63.6 Decision Tables and Algorithms - None.

6.2.64 SSPMAC

6.2.64.1 Identification

- o SSPMAC - Station Link Prelininary Prompt Test Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.64.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SL	-	C	IDENTIFIER OF THE STATION LINK BEING PROMPTED.
FLAG	-	C	CHARACTER STRING TO INDICATE THAT THE SL ITSELF SHOULD BE PROMPTED. (OPTIONAL; DEFAULT = NULL. VEHICLES ON EACH LINK IMMEDIATELY UPSTREAM OF AL SHOULD BE PROMPTED)

6.2.64.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
X	-	C	'X' character string
LOC1	-	C	Indicates queued vehicle has been found
LOC2	-	C	Link number being searched
LOC3	-	C	Number of system service XTN gotten for PROMPT

6.2.64.4 Description - The purpose of SSPMAC is to schedule a special purpose transaction zero time in the future which, when it comes off the FEL, will run SASPRM. SSPMAC is run at various points in the other station link code segments to schedule a special purpose transaction zero time in the future which, when it comes off the FEL, will run SASPRM. This mechanism of scheduling a prompt to occur immediately rather than immediately calling SASPRM at that point in the code is done since SASPRM calls SSLEAV which would call SASPRM and so on.

SSPMAC first does some preliminary prompt testing to see if it is necessary to schedule a prompt at all. Next it gets a free transaction and initializes it to call SASPRM when it comes off the FEL. It then schedules it on the FEL zero time in the future.

6.2.64.5 PDL - See Appendix A.

6.2.64.6 Decision Tables and Algorithms - None.

6.2.65 SSTEST

6.2.65.1 Identification

- o SSTEST - Station Link Entry Testing and Next Link Determination
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.65.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
V	-	I*4	VEHICLE OR LEAD VEHICLE OF A TRAIN WHICH IS DONE WITH ITS STATION LINK EVENTS AND READY TO PROCEED TO ITS NEXT STATION LINK

6.2.65.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
VLEN	-	I*4	VEHICLE'S TRAIN LENGTH
SL	-	I*2	VEHICLE'S CURRENT STATION LINK
DONE	-	L*1	NOT USED
FOUND	-	L*1	NOT USED

6.2.65.4 Description - The purpose of SSTEST is to determine the next station link to be entered and if it can be entered. Station link entry testing is comprised of a series of its next station link.

The first test insures that the vehicle in question is at the head of the link it is on (i.e., there are no other vehicles in front of it). If there are other vehicles in front of it, the vehicle is marked as done with all events on its current link and queued. If this test is passed, then a test is made to determine if the exit of the vehicle's current link is failed or not. If that exit is failed, the vehicle is marked as queued due to the congestion/failure.

Once the tests are passed on the current link, downstream links are examined. If there is no diverge at the end of the current link, then a test is made to determine if the next link is a sink (i.e., there are no more station links). In this case, the next station link to be entered is noted to be a sink and the "can enter" indicator is set. If the next link is not a sink and there is no diverge, a list of possible links to enter is initialized to the single downstream link. In the case where there is a diverge downstream of the current link, SMDIVF is run to narrow down the list of possible downstream links and also to order that shortened list in an order of preference. This reduced and ordered list is used by SSTEST.

Once the list is built, the following tests are made for each link on the list until an adequate link is found or the list is exhausted. First a test is made to insure that the link is available (i.e., has not been "turned off" by the user for this simulation run) and that the link entry is not failed. If these tests are passed, a test is made to see if the headway zone of the link in question is occupied and if the capacity of the link would be violated by allowing the train of the vehicle being processed to enter. The capacity check uses occupancy or pseudo-occupancy as appropriate (and as explained in SSMODN).

SSTEST signals "can enter" or "cannot enter" as appropriate. When the vehicle cannot enter the reason it is queued is set. When it can enter, the next station link is set. Miscellaneous statistics are also collected.

6.2.65.5 PDL - See Appendix A.

6.2.65.6 Decision Tables and Algorithms - None.

6.2.66 SULEAV

6.2.66.1 Identification

- o SULEAV - Processing a Trip Leaving a Trip Link
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.66.2 Argument Dictionary - None.

6.2.66.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NEXTT	-	I*4	TRIP BEHIND-1 ON THE TRIP LINK MEMBERSHIP LIST
TL	-	I*2	I'S CURRENT TRIP LINK

6.2.66.4 Description - SULEAV performs processing associated with a trip leaving a trip link. When it has been guaranteed that the next link can be entered, SULEAV decreases the occupancy of the trip link by the size of the trip and dequeues it from the-link's membership list. The waiting trip behind the leaving trip is gotten moving again by being modeled. The upstream links are prompted so trips on it may also have the opportunity to get moving again.

6.2.66.5 PDL - See Appendix A.

6.2.66.6 Decision Tables and Algorithms - None.

6.2.67 SUMOD

6.2.67.1 Identification

- o SUMOD - Model a Trip on its Current Trip Link
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.67.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
T	-	I*4	TRIP TO BE MODELED ON ITS TRIP LINK

6.2.67.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TIM	-	I*4	TIME TO PERFORM THE PROCESSING EVENT
TL	-	I*2	T'S CURRENT TRIP LINK

6.2.67.4 Description - SUMOD performs processing associated with each of the events a trip can undergo on a trip link. When SUMOD is entered for the first time by a trip transaction, the occupancy of the link is increased by the size of the trip, the trip is enqueued in the membership list of the link (to record the order of entry), the next event number is set to 1, the walk time on the link is used when scheduling the trip on the FEL.

After the walk event on the ticketing or turnstile link, a test is made to see if the trip is at the head of its trip link (i.e., there are no other trips in front of it). An indicator associated with the trip is set to indicate that it cannot proceed with its next event (namely processing through ticketing/turnstile mechanisms). When the trip is at the head of its trip link, the processing time through the ticketing/turnstile mechanism is computed from the form ax/y+b, where x is the number of passengers in the trip, y is the number of active servers (mechanisms), and a and b are user specified times. The next event number of the trip is then set and the trip is put on the FEL for the amount of computed time. After the walk event on the boarding link (i.e., there is no processing event to be performed), "done" is signalled immediately. For ticketing and turnstile links done is signalled after processing through the ticketing/turnstile mechanisms.

If SUMOD is entered with an event code indicating a deboard exit walk has been completed, final statistics are collected on the trip and its transaction is freed. If a transfer walk was completed, statistics on collected and SMTABQ is called.

6.2.67.5 PDL - See Appendix A.

6.2.67.6 Decision Tables and Algorithms - None.

6.2.68 SUPMAC

6.2.68.1 Identification

- o SUPMAC - Trip Link Preliminary Prompt Test Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.68.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TL	-	C	IDENTIFIER OF THE TRIP LINK BEING PROMPTED.

6.2.68.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
X	-	C	'X' character string
LOC1	-	C	Number of system service XTN gotten for PROMPT

6.2.68.4 Description - The purpose of SUPMAC is to schedule a special purpose transaction zero time in the future which, when it comes off the FEL, will run SAUPRM. SUPMAC is run at various points in the other trip link code segments to schedule a special purpose transaction zero time in the future, which, when it comes off the FEL, will run SAUPRM. This mechanism of scheduling a prompt to occur immediately rather than immediately calling SAUPRM at that point in the code is done since SAUPRM calls SULEAV which would call SAUPRM and so on.

SUPMAC first does some preliminary prompt testing to see if it is necessary to schedule a prompt at all. Next it gets a free transaction and initializes it to call SAUPRM when it comes off the FEL. It then schedules it on the FEL zero time in the future.

6.2.68.5 PDL - See Appendix A.

6.2.68.6 Decision Tables and Algorithms - None.

6.2.69 SUTEST

6.2.69.1 Identification

- o SUTEST - Trip Link Entry Testing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.69.2 Argument Dictionary - None.

6.2.69.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NXTL	-	I*2	T'S NEXT TRIP LINK

6.2.69.4 Description

- SUTEST is used to determine the next trip link to be entered and if it can be entered once the trip is on a trip link. In the case where the trip's current trip link is the ticketing link or turnstile link, a test is made to determine if the capacity limit of the next link would be violated by its entry. If so the trip is marked as queued and cannot enter. Otherwise the next link is set to the turnstile link or boarding link respectively so that the trip can enter.

In the case when the current link is the boarding link, the next link is set to four to indicate to SAUCTL that link processing is finished and the trip is ready to board.

6.2.69.5 PDL - See Appendix A.

6.2.69.6 Decision Tables and Algorithms - None.

6.2.70 SZHDR

6.2.70.1 Identification

- o SZHDR - Write Raw Statistics Header Record
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.70.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NFOLL	-	I*4	NUMBER OF FOLLOWER RECORDS
NTYPE	-	I*4	TYPE OF FOLLOWER RECORDS

6.2.70.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
HEADER	-	R*8	Keyboard 'HEADER' to write in records
MBYTES	-	I*2	Number of bytes in header
MFOLL	-	I*2	Number of followers
MTYPE	-	I*2	Type of followers

6.2.70.4 Description - The purpose of SZHDR is to write a header record on the raw statistics file that indicates the number and type of follower records to follow.

This routine formats and writes a header record that contains the word 'HEADER', its own length, the clock time, and number and type of follower records.

6.2.70.5 PDL - See Appendix A.

6.2.70.6 Decision Tables and Algorithms - None.

6.2.71 SZINT

6.2.71.1 Identification

- o SZINT - Calculate Integral Averages and Miscellaneous Statistics
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.71.2 Argument Dictionary - None.

6.2.71.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NBY	-	I*2	Number of bytes
COUNT	72	R*4	Count of SLS of each type
AINT	-	I*4	Length of last sample period
K	-	I*4	Number of clock units per second
KNTL	-	I*4	Number of trip links

6.2.71.4 Description - SZINT calculates endpoint integrals and calculates averages and miscellaneous statistics. It begins by endpointing integrals. This is done by adding to the integral the product of the clock and count of number of entities currently in state/ Then the average number in state is calculated by dividing the time integral ~~in~~ state by the length of the interval. The average time in state is calculated by dividing the sum of times in state of those leaving by the number leaving.

Miscellaneous statistics relating to averages and maxima within station link type are then calculated. Other miscellaneous statistics relating to trip link activity are set here for use in the performance summary file.

6.2.71.5 PDL - See Appendix A.

6.2.71.6 Decision Tables and Algorithms - None.

6.2.72 SZSTAT

6.2.72.1 Identification

- o SZSTAT - Collect Statistics
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.72.2 Argument Dictionary

```
|------------------------------------------------------------------------|
|VARIABLE  | DIM | TYPE | DESCRIPTION                                    |
|------------------------------------------------------------------------|
 I           -     I*4    TYPE OF ELEMENTS:
                            1 = VEHICLE
                            2 = TRIP
 J           -     I*4    ELEMENT NUMBER:
                            FOR VEHICLES:  1 - KNV
                            FOR TRIPS:  1 - KNT
 K           -     I*4    ENTITY TYPE:
                            1 = STATION (ENTIRE MODELLED AREA AS A
                                WHOLE APPLICABLE TO VEHICLES AND TRIPS)
                            2 = STATION LINK (APPLICABLE TO VEHICLES
                                ONLY)
                            3 = TRIP LINK (APPLICABLE TO TRIPS ONLY)
 L           -     I*4    DIRECTION:
                            1 = ENTERING STATE
                            2 = LEAVING STATE
 M           -     I*4    STATE:
                             FOR STATIONS:
                               1 = IN STATION
                               2 = IN BOARD EVENT
                               3 = IN DEBOARD EVENT
                               4 = IN LAUNCH EVENT
                             FOR STATION LINKS:
                               1 = ON STATION LINK
                               2 = ON FEL
                               3 = QUEUED
                             FOR TRIP LINKS:
                               1 = ON TRIP LINK
                               2 = ON FEL
                               3 = QUEUED
 N           -     I*2    LINK NUMBER:
                               FOR STATIONS:  0
                               FOR STATION LINKS: 1 - KMSL
                               FOR TRIP LINKS:  1 - KMTL
```

6.2.72.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
JJJ	-	I*4	Saved entering value of J
MM	-	I*4	Saved entering value of M
VID	-	I*4	ID of vehicle being processed
QHEAD	-	I*4	Pointer to head vehicle of train
IND	-	I*4	Indicates just entrained leaving FEL
DELTA	-	I*4	Difference between time clock was saved and now

6.2.72.4 Description - Collect statistics other than miscellaneous statistics. Based on the input arguments branches are taken to code to update the appropriate statistics. A study of the statistics in the commons (other than the miscellaneous) will show the structure of these variables. When entering a state, the number entering, number in, time integral, and maximum number are updated. When leaving a state, the number leaving, number in, time integral, sum of times in, and maximum time in are updated.

Table 6-1 is an outline of the data contents of the Raw Statistics File. This data is written in binary.

The following is an expansion on the statistical states shown in Table 6-2. The term state refers to the concept that a vehicle or trip is in a state or states for a period of time. When the state is entered

1. The number in the state is increased by 1
2. The number entering the state is increased by 1
3. The maximum number in the state is updated (if necessary)
4. The value of the clock is stored away for use when leaving the state to determine time in state
5. The time integral of number of entities (trips or vehicles) in state is adjusted.

When a trip/vehicle leaves a state

1. The number in the state is decreased by 1
2. The number leaving the state is increased by 1

Table 6-2. SZSTAT Statistics Descriptions (Page 1 of 7)

```
SCZ:     STATISTICS - MODEL PROCESSOR & OUTPUT PROCESSOR

SUMMARY:  THE FOLLOWING TABLES SUMMARIZE THE DEFINITIONS OF MOST OF
          THE STATISTICS

   STATISTIC NAMES:

     LETTERS      SIGNIFICANCE

        1         'Z' --- ALL STATISTIC VARIABLES BEGIN WITH 'Z'

       2-3        THE STATISTIC RELATES TO:
                    NV - VEHICLES IN STATION STATES
                    NT - TRIPS IN STATION STATES
                    NP - PASSENGERS IN STATION STATES
                    SV - VEHICLES IN STATION LINK(SL) STATES
                    TT - TRIPS IN TRIP LINK STATES
                    TP - PASSENGERS IN TRIP LINK STATES

       4-6        STATISTIC TYPE:
                    NE  - NUMBER ENTERING STATE DURING LAST SAMPLING
                            INTERVAL(HISTORICAL)
                    NL  - NUMBER LEAVING STATE DURING LAST SAMPLING
                            INTERVAL(HISTORICAL)
                    NI  - NUMBER IN STATE AT END OF LAST SAMPLING
                            INTERVAL(STATUS)
                    MNI - MAXIMUM NUMBER IN DURING LAST SAMPLING
                            INTERVAL(HISTORICAL)
                    TIN - TIME INTEGRAL OF NUMBER IN STATE DURING LAST
                            SAMPLING INTERVAL(HISTORICAL)
                    STL - SUM OF THE TIMES IN STATE OF THOSE LEAVING
                            STATE DURING LAST SAMPLING INTERVAL(HIST.)
                    MTL - MAXIMUM TIME IN STATE OF THOSE LEAVING STATE
                            DURING LAST SAMPLING INTERVAL(HISTORICAL)
                    ANI - AVERAGE NUMBER IN STATE DURING LAST SAMPLING
                            INTERVAL(HISTORICAL) (DERIVED AT SAMPLE
                            OUTPUT TIME BY DIVIDING 'TIN' BY LENGTH
                            OF THE SAMPLING INTERVAL(ASAMPI))
                    ATL - AVERAGE TIME IN STATE OF THOSE ELEMENTS
                            LEAVING STATE DURING LAST SAMPLING
                            INTERVAL(HISTORICAL) (DERIVED AT SAMPLE
                            OUTPUT TIME BY DIVIDING 'STL' BY 'NL')

     SUBSCRIPTS   SIGNIFICANCE
         1        STATE:
                    STATION STATES:
                      1 - IN STATION
                      2 - IN BOARD EVENT
                      3 - IN DEBOARD EVENT
                      4 - IN LAUNCH EVENT
```

Table 6-2. SZSTAT Statistics Descriptions (Page 2 of 7)

```
          STATION LINK STATES:
            1 - ON STATION LINK
            2 - ON FEL
            3 - QUEUED
          TRIP LINK STATES:
            1 - ON TRIP LINK
            2 - ON FEL
            3 - QUEUED
        LINK NUMBER:
          STATION STATISTICS  - OMITTED
          STATION LINKS - STATION LINK NUMBER
          TRIP LINKS - TRIP LINK NUMBER
```

Table 6-2. SZSTAT Statistics Descriptions (Page 3 of 7)

VAR NAME	DIM	DESCRIPTION
KMNST		NUMBER OF STATION STATES (DEFINED IN SMAXSIZE)
KMSST		NUMBER OF STATION LINK STATES(")
KMTST		NUMBER OF TRIP LINK STATES (")
---STATISTICS ON VEHICLES IN STATION STATES		
ZNVNE	KMNST /I2	NUMBER OF VEHICLES ENTERING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNVNL	KMNST /I2	NUMBER OF VEHICLES LEAVING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNVNI	KMNST /I2	NUMBER OF VEHICLES IN STATE I OF THE STATION AT THE END OF THE LAST SAMPLING INTERVAL
ZNVMNI	KMNST /I2	MAXIMUM NUMBER OF VEHICLES IN STATE I OF STATION DURING THE LAST SAMPLING INTERVAL
ZNVTIN	KMNST /I4	INTEGRAL OF VEHICLE-TIME IN STATE I IN STATION DURING THE LAST SAMPLING INTERVAL
ZNVSTL	KMNST /I4	SUM OF TIMES IN STATE I OF VEHICLES LEAVING DURING THE LAST SAMPLING INTERVAL
ZNVMTL	KMNST /I4	MAXIMUM TIME IN STATE I OF VEHICLES LEAVING DURING THE LAST SAMPLING INTERVAL
ZNVANI	KMNST /R4	AVERAGE NUMBER OF VEHICLES IN STATE I DURING THE LAST SAMPLING INTERVAL
ZNVATL	KMNST /R4	AVERAGE TIME IN STATE I OF VEHICLES LEAVING DURING THE LAST SAMPLING INTERVAL
---STATISTICS ON TRIPS IN STATION STATES		
ZNTNE	KMNST /I2	NUMBER OF TRIPS ENTERING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNTNL	KMNST /I2	NUMBER OF TRIPS LEAVING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNTNI	KMNST /I2	NUMBER OF TRIPS IN STATE I OF THE STATION AT THE END OF THE LAST SAMPLING INTERVAL
ZNTMNI	KMNST /I2	MAXIMUM NUMBER OF TRIPS IN STATE I OF STATION DURING THE LAST SAMPLING INTERVAL
ZNTTIN	KMNST /I4	INTEGRAL OF TRIP-TIME IN STATE I IN STATION DURING THE LAST SAMPLING INTERVAL
ZNTSTL	KMNST /I4	SUM OF TIMES IN STATE I OF TRIPS LEAVING DURING THE LAST SAMPLING INTERVAL
ZNTMTL	KMNST /I4	MAXIMUM TIME IN STATE I OF TRIPS LEAVING DURING THE LAST SAMPLING INTERVAL
ZNTANI	KMNST /R4	AVERAGE NUMBER OF TRIPS IN STATE I DURING THE LAST SAMPLING INTERVAL
ZNTATL	KMNST /R4	AVERAGE TIME IN STATE I OF TRIPS LEAVING DURING THE LAST SAMPLING INTERVAL
---STATISTICS ON PASSENGERS IN STATION STATES		
ZNPNE	KMNST /I2	NUMBER OF PASS. ENTERING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL

Table 6-2. SZSTAT Statistics Descriptions (Page 4 of 7)

ZNPNL	KMNST /I2	NUMBER OF PASS. LEAVING STATE I OF THE STATION DURING THE LAST SAMPLING INTERVAL
ZNPNI	KMNST /I2	NUMBER OF PASS. IN STATE I OF THE STATION AT THE END OF THE LAST SAMPLING INTERVAL
ZNPMNI	KMNST /I2	MAXIMUM NUMBER OF PASS. IN STATE I OF STATION DURING THE LAST SAMPLING INTERVAL
ZNPTIN	KMNST /I4	INTEGRAL OF PASS.-TIME IN STATE I IN STATION DURING THE LAST SAMPLING INTERVAL
ZNPSTL	KMNST /I4	SUM OF TIMES IN STATE I OF PASS. LEAVING DURING THE LAST SAMPLING INTERVAL
ZNPMTL	KMNST /I4	MAXIMUM TIME IN STATE I OF PASS. LEAVING DURING THE LAST SAMPLING INTERVAL
ZNPANI	KMNST /R4	AVERAGE NUMBER OF PASS. IN STATE I DURING THE LAST SAMPLING INTERVAL
ZNPATL	KMNST /R4	AVERAGE TIME IN STATE I OF PASS. LEAVING DURING THE LAST SAMPLING INTERVAL
---STATISTICS ON VEHICLES IN STATION LINK (SL) STATES		
ZSVNE	KMSST KMSL/I2	NUMBER OF VEHICLES ENTERING STATE I OF SL J DURING THE LAST SAMPLING INTERVAL
ZSVNL	KMSST KMSL/I2	NUMBER OF VEHICLES LEAVING STATE I OF SL J DURING THE LAST SAMPLING INTERVAL
ZSVNI	KMSST KMSL/I2	NUMBER OF VEHICLES IN STATE I OF SL J AT THE END OF THE LAST SAMPLING INTERVAL
ZSVMNI	KMSST KMSL/I2	MAXIMUM NUMBER OF VEHICLES IN STATE I ON SL J DURING THE LAST SAMPLING INTERVAL
ZSVTIN	KMSST KMSL/I4	INTEGRAL OF VEHICLE-TIME IN STATE I ON SL J DURING THE LAST SAMPLING INTERVAL
ZSVSTL	KMSST KMSL/I4	SUM OF TIMES OF VEHICLES LEAVING STATE I ON SL J DURING THE LAST SAMPLING INTERVAL
ZSVMTL	KMSST KMSL/I4	MAXIMUM TIME OF VEHICLES LEAVING STATE I ON SL J DURING THE LAST SAMPLING INTERVAL
ZSVANI	KMSST KMSL/R4	AVERAGE NUMBER OF VEHICLES IN STATE I ON SL J DURING THE LAST SAMPLING INTERVAL
ZSVATL	KMSST KMSL/R4	AVERAGE TIME OF VEHICLES LEAVING STATE I ON SL J DURING THE LAST SAMPLING INTERVAL
---STATISTICS ON TRIPS IN TRIP LINK (TL) STATES		
ZTTNE	KMTST KMTL/I2	NUMBER OF TRIPS ENTERING STATE I OF TL J DURING THE LAST SAMPLING INTERVAL
ZTTNL	KMTST KMTL/I2	NUMBER OF TRIPS LEAVING STATE I OF TL J DURING THE LAST SAMPLING INTERVAL
ZTTNI	KMTST KMTL/I2	NUMBER OF TRIPS IN STATE I OF TL J AT THE END OF THE LAST SAMPLING INTERVAL
ZTTMNI	KMTST KMTL/I2	MAXIMUM NUMBER OF TRIPS IN STATE I ON TL J DURING THE LAST SAMPLING INTERVAL
ZTTTIN	KMTST KMTL/I4	INTEGRAL OF TRIP-TIME IN STATE I ON TL J DURING THE LAST SAMPLING INTERVAL
ZTTSTL	KMTST KMTL/I4	SUM OF TIMES OF TRIPS LEAVING STATE I ON TL J DURING THE LAST SAMPLING INTERVAL

Table 6-2. SZSTAT Statistics Descriptions (Page 5 of 7)

```
ZTTMTL        KMTST    MAXIMUM TIME OF TRIPS LEAVING STATE I ON TL J
               KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTTANI        KMTST    AVERAGE NUMBER OF TRIPS IN STATE I ON TL J
               KMTL/R4  DURING THE LAST SAMPLING INTERVAL
ZTTATL        KMTST    AVERAGE TIME OF TRIPS LEAVING STATE I ON TL J
               KMTL/R4  DURING THE LAST SAMPLING INTERVAL

---STATISTICS ON PASS. IN TRIP LINK (TL) STATES
ZTPNE         KMTST    NUMBER OF PASS. ENTERING STATE I OF TL J
               KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTPNL         KMTST    NUMBER OF PASS. LEAVING STATE I OF TL J
               KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTPNI         KMTST    NUMBER OF PASS. IN STATE I OF TL J
               KMTL/I2  AT THE END OF THE LAST SAMPLING INTERVAL
ZTPMNI        KMTST    MAXIMUM NUMBER OF PASS. IN STATE I ON TL J
               KMTL/I2  DURING THE LAST SAMPLING INTERVAL
ZTPTIN        KMTST    INTEGRAL OF PASS.-TIME IN STATE I ON TL J
               KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTPSTL        KMTST    SUM OF TIMES OF PASS. LEAVING STATE I ON TL J
               KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTPMTL        KMTST    MAXIMUM TIME OF PASS. LEAVING STATE I ON TL J
               KMTL/I4  DURING THE LAST SAMPLING INTERVAL
ZTPANI        KMTST    AVERAGE NUMBER OF PASS. IN STATE I ON TL J
               KMTL/R4  DURING THE LAST SAMPLING INTERVAL
ZTPATL        KMTST    AVERAGE TIME OF PASS. LEAVING STATE I ON TL J
               KMTL/R4  DURING THE LAST SAMPLING INTERVAL

---THE FOLLOWING STATISTICS DO NOT FIT INTO THE ABOVE SCHEME
    AND ARE REFERRED TO AS MISCELLANEOUS

ZM            280/R4   MISCELLANEOUS STATISTICS; THE FIRST
                          218 OF THESE ARE USED TO GENERATE THE
                          PERFORMANCE SUMMARY FILE BY THE OP
 SUBSCRIPT
          1 VEHICLE CAPACITY(=VCAP)
          2 AVERAGE VEHICLE LOAD ENTERING STN FROM GUIDEWAY
          3 AVERAGE VEHICLE LOAD ENTERING STN FROM MODAL INPUT BEFORE
          4 AVERAGE VEHICLE LOAD ENTERING STN FROM MODAL INPUT AFTER
          5 AVERAGE VEHICLE LOAD LEAVING STN FROM GUIDEWAY
          6 AVERAGE VEHICLE LOAD LEAVING STN FROM MODAL INPUT BEFORE
          7 AVERAGE VEHICLE LOAD LEAVING STN FROM MODAL INPUT AFTER
          8 NUMBER OF VEHICLE ENTERING STN FROM GUIDEWAY
          9 NUMBER OF VEHICLE ENTERING STN FROM MODAL INPUT BEFORE
         10 NUMBER OF VEHICLE ENTERING STN FROM MODAL INPUT AFTER
         11 NUMBER OF VEHICLE LEAVING STN FROM GUIDEWAY
         12 NUMBER OF VEHICLE LEAVING STN FROM MODAL INPUT BEFORE
         13 NUMBER OF VEHICLE LEAVING STN FROM MODAL INPUT AFTER
         14 NUMBER OF VEHICLES REJECTED AT INPUT RAMP
         15 NUMBER OF VEHICLES ACCEPTED AT INPUT RAMP
         16 NUMBER OF EMPTIES GOTTEN FROM LOCAL STORAGE
```

Table 6-2. SZSTAT Statistics Descriptions (Page 6 of 7)

```
     17 NUMBER OF EMPTIES GOTTEN FROM UPSTREAM SLS
     18 NUMBER OF EMPTIES GOTTEN FROM ELSEWHERE IN NET
     19 NUMBER OF TRIPS ARRIVING AT BOARD QUEUE
     20 NUMBER OF TRIPS BOARDING
     21 NUMBER OF TRIPS DEBOARDING TO LEAVE
     22 NUMBER OF TRIPS DEBOARDING TO TRANSFER
     23 NUMBER OF PASSENGERS ARRIVING AT BOARD QUEUE
     24 NUMBER OF PASSENGERS BOARDING
     25 NUMBER OF PASSENGERS DEBOARDING TO LEAVE
     26 NUMBER OF PASSENGERS DEBOARDING TO TRANSFER
                             SLTYPE     MEANING
                                1         IR
                                2         IQ
                                3         D (THE DEBOARD/BOARD/JOINT EVENTS
                                             CAN APPEAR ONLY ON THIS TYPE)
                                4         OQ
                                5         OR
                                6         S
                                7         IS
                                8         SI
                                9         DS
                               10         SO
                               11         UL
                               12         BL
                               13         DL
                               14         MIB
                               15         MIA
                               16         MOB
                               17         MOA
                               18         UNUSED
  27- 44 FOR EACH 'SLTYPE' AVERAGE # OF VEHICLES IN SL OF THAT TYPE
  45- 62 FOR EACH 'SLTYPE' MAXIMUM # OF VEHICLES IN SL OF THAT TYPE
  63- 80 FOR EACH 'SLTYPE' AVERAGE TIME SPENT IN SL OF THAT TYPE
  81- 98 FOR EACH 'SLTYPE' MAXIMUM TIME SPENT IN SL OF THAT TYPE
  99-116 FOR EACH 'SLTYPE' AVERAGE # OF VEH IN SL QUEUE OF THAT TYPE
 117-134 FOR EACH 'SLTYPE' MAXIMUM # OF VEH IN SL QUEUE OF THAT TYPE
 135-152 FOR EACH 'SLTYPE' AVERAGE TIME SPENT IN SL QUEUE OF THAT TYPE
 153-170 FOR EACH 'SLTYPE' MAXIMUM TIME SPENT IN SL QUEUE OF THAT TYPE
 171-173 FOR EACH TL AVERAGE # OF TRIPS IN TL
 174-176 FOR EACH TL MAXIMUM # OF TRIPS IN TL
 177-179 FOR EACH TL AVERAGE TIME SPENT IN TL
 180-182 FOR EACH TL MAXIMUM TIME SPENT IN TL
 183-185 FOR EACH TL AVERAGE # OF TRIPS IN TL QUEUE
 186-188 FOR EACH TL MAXIMUM # OF TRIPS IN TL QUEUE
 189-191 FOR EACH TL AVERAGE TIME SPENT IN TL QUEUE
 192-194 FOR EACH TL MAXIMUM TIME SPENT IN TL QUEUE
 195-197 FOR EACH TL AVERAGE # OF PASSENGERS IN TL
 198-200 FOR EACH TL MAXIMUM # OF PASSENGERS IN TL
 201-203 FOR EACH TL AVERAGE TIME SPENT IN TL
 204-206 FOR EACH TL MAXIMUM TIME SPENT IN TL
```

Table 6-2. SZSTAT Statistics Descriptions (Page 7 of 7)

```
207-209 FOR EACH TL AVERAGE # OF PASSENGERS IN TL QUEUE
210-212 FOR EACH TL MAXIMUM # OF PASSENGERS IN TL QUEUE
213-215 FOR EACH TL AVERAGE TIME SPENT IN TL QUEUE
216-218 FOR EACH TL MAXIMUM TIME SPENT IN TL QUEUE
    219 NUMBER OF TRIPS REJECTED AT TICKETING LINK
```

3. The sum of times spent in state for those leaving the state is increased by the difference between the current value of the clock and the saved value of the clock at state entry.

4. The maximum time spent in state of those leaving is updated (if necessary)

5. The time integral of number of entitites (trips or vehicles) in state is adjusted.

These calculations at entry and exit to a state allow seven statistics to be compiled on the state each sampling interval. In addition, two averages can be calculated from these seven basic statistics at the end of each sampling interval after the seven have been collected. These nine statistics are:

- o Number entering state during last sampling interval (historical)
- o Number leaving state during last sampling interval (historical)
- o Number in state at end of last sampling interval (status)
- o Maximum number in during last sampling interval (historical)
- o Time integral of number in state during last sampling interval (historical)
- o Sum of the times in state of those leaving state during last sampling interval (historical)
- o Maximum time in state of those leaving state during last sampling interval (historical)
- o Average number in state during last sampling interval (historical) derived at sample output time by dividing 'TIN' by length of the sampling interval (ASAMPI))
- o Average time in state of those elements leaving state during last sampling interval (historical) derived at sample output time by dividing 'STL by 'NL').

There are three sets of states: those with respect to the station as a whole, those with respect to station links, and those with respect to trip links. The states with respect to the station as a whole include the following:

1. In station -- this is with respect to vehicles and trips (and passengers) and refers to the number that are in the entire

modeled area. For trips it includes those entering and leaving on foot and by vehicles. For vehicles it includes those entering and leaving via all sources and links.

2. In BOARD/JOINT event -- This is with respect to vehicles only (since trips do not have a "BOARD/JOINT event" -- vehicles do). It includes all vehicles that enter and leave the BOARD/JOINT event on any station link in the station that has the BOARD/JOINT event.

3. In DEBOARD event -- This is analogous to item b, but for the DEBOARD event.

4. In LAUNCH event -- This is analogous to item b but for the LAUNCH event.

The states with respect to station links are for vehicles only and include the following:

1. On station links -- These statistics are updated when a vehicle enters and leaves the station link.

2. In Processing (On FEL) -- These statistics are updated every time a vehicle enters or leaves an event on a FEL. Thus if there are several events on a link, a single vehicle will cause this to be updated several times.

3. Queued -- These statistics are updated every time a vehicle enters or leaves a queued state. In the majority of station links all queuing occurs at the end of the link (i.e., after all events are done) since the model directs vehicles to go from one event to the next until all events are done and then queue if it cannot leave the link. In this case, a vehicle can enter the queued state only once on a link. However, in the case of a link that contains the LAUNCH event, the rule that requires that event not to start until the vehicle is at the end of the link, causes a situation where a given vehicle may queue once before the LAUNCH event (waiting to get to the head of the link), go through the launch, and then queue again (waiting to get off the link due to congestion or failure). Thus on links with the LAUNCH event a vehicle in a heavily congested/failure situation may enter the queued state twice.

The states with respect to trip links are for trips (and passengers only) and include the following:

1. On trip link -- These statistics are updated when a trip enters or leaves a trip link. A trip is considered to leave the boarding link after the BOARD event has transpired.

2. In Processing (On FEL) -- These statistics are updated every time a trip enters or leaves an event on the FEL. Thus on the ticketing and turnstile links where there are two events (viz., walk and process), a single trip will cause this to be updated twice.

3. Queued -- These statistics are updated every time a trip enters or leaves a queued state. In the case of the ticketing and turnstile links that contain the processing event, the rule that requires that event not to start until the trip is at the end of the link (i.e., all other trips ahead of it have gone through the ticketing/turnstile mechanism) causes a situation where a given trip may queue and before the processing event (waiting to get to the head of the link), go through the processing event, and then queue again (waiting to get off the link due to congestion). Thus, on these two links in a heavily congested situation a trip may enter the queued state twice. In the case of the boarding link, a trip is considered to leave the queued state after the board event has transpired.

With respect to the miscellaneous statistics the first 26 are clear cut. The following eight groups of eighteen (numbers 27-170) relate to averages and maximum over all links of each station link type. For example, number 28 contains the average number of vehicles in station links of type 2 -- input queues; that is, the average of all input queues is averaged to come up with one input-queue-wide number. The miscellaneous statistics numbered 171 through 194 are just repetitions of ZTTANI(i,j), ZTTMNI(i,j), ZTTATL(i,j), and ZTTMTL(i,j) where i goes from 1 through 3 (over the three trip links) and j = 1 (on trip link) and 3 (queued) on trip link. Statistics 195 through 218 are analogous but for passengers and use 'ZIP' statistics instead of 'ZIT'. Statistics 171 through 218 are repeats of other statistics to make it easier for the output processor to locate statistics to do a performance summary by groupoing them all in one place.

All of this data in Table 6-1 is written to the raw statistics file each sampling interval.

6.2.72.5 PDL - See Appendix .

6.2.72.6 Decision Tables and Algorithms - None.

6.2.73 SZZERO

6.2.73.1 Identification

- o SZZERO - Reset Statistics
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.2.73.2 Argument Dictionary - None.

6.2.73.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
KNTL	-	I*4	Number of trip links

6.2.73.4 Description - The purpose of this routine is to reset statistical variables. All the statistical variables, except the status type variables (i.e., number in state) are reset. All of these are reset to zero, except the maximum number which is set to the current number in and the time integral in (the latter of which is set to the negative of the product of the number currently in times the current clock value).

6.2.73.5 PDL - See Appendix A.

6.2.73.6 Decision Tables and Algorithms - None.

6.2.74 VRAND

6.2.74.1 Identification

- o VRAND - Uniformity Distrubuted Random Number Generator Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.74.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SEED	–	I*4	(INPUT) NAME OF RANDOM NUMBER SEED WHICH MUST BE AN ODD INTEGER >= 3. (OUTPUT) UPDATED SEED.
VALUE	–	R*4	(INPUT) NAME OF RANDOM VARIABLE TO BE RETURNED. (OUPUT) A RANDOM VARIABLE BETWEEN 0 AND 1.

6.2.74.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer

6.2.74.4 Description - The purpose of VRAND is to generate uniformly distributed random numbers between zero and one. This macro generates code which when executed performs a function analagous to SMRNG. Its only use in DSM is in VRANDN.

6.2.74.5 PDL - See Appendix A

6.2.74.6 Decision Tables and Algorithms - None.

6.2.75 VRANDN

6.2.75.1 Identification

- o BRANDN - Normal Random Number Generation Macro
- o IBM/FSD - July 1, 1977
- o PL/I

6.2.74.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
SEED	-	I*4	(INPUT) NAME OF RANDOM NUMBER SEED WHICH MUST BE AN ODD INTEGER >= 3. (OUTPUT) UPDATED SEED.
MEAN	-	R*4	NAME OF MEAN VÁLUE.
SD	-	R*4	NAME OF STANDARD DEVIATION.
VALUE	-	R*4	(INPUT) NAME OF RANDOM VARIABLE TO BE RETURNED. (OUPUT) A NORMALLY DISTRIBUTED RANDOM VARIABLE WITH THE SPECIFIED MEAN AND STANDARD DEVIATION.

6.2.75.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	Constructed FORTRAN code
M	-	C	Hold margin pointer
LOC1	-	C	Loop index
LOC2	-	C	Accumulator for random numbers

6.2.75.4 Description - The purpose of VRANDN is to generate normally distributed random numbers. This macro generates code when executed generates 12 uniformly distributed random numbers using BRAND, computes their sum, subtracts 6, multiplies the result by the standard deviation and adds the mean.

6.2.75.5 PDL - See Appendix A.

6.2.75.6 Decision Tables and Algorithms - None.

6.3 OUTPUT PROCESSOR

This section contains the subprogram descriptions for the DSM-Output Processor.

6.3.1 CKFOLLOW

6.3.1.1 Identification

- CKFOLLOW - Check the Follower Record
- IBM/FSD - July 1, 1977
- PL/I

6.3.1.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NONE			

6.3.1.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
OUT	-	C	CONSTRUCTED FORTRAN STATEMENTS

6.3.1.4 Description - CKFOLLOW generates code which when executed simply tests if the first eight bytes of an alleged follower record contains the characters 'FOLLOWER' and if not stops the OP.

6.3.1.5 PDL - None since it is a macro.

6.3.1.6 Decision Tables and Algorithms - None.

6.3.2 DAYTIM

6.3.2.1 Identification

- DAYTIM - Convert Date and Time to YY/MM/DD/HH/MM/SS

See DAYTIM in MP section.

6.3.3 <u>DBUG</u>

6.3.3.1 <u>Identification</u>

- o DBUG - Write Intermediate Output

See SBUG in MP section.

6.3.4 <u>DTIMEL</u>

6.3.4.1 <u>Identification</u>

- DTIMEL - Get Date and Time from System

See DTIMEL in MP section.

6.3.5 SHIST

6.3.5.1 Identification

- o SHIST - Output Histogram of Data
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.5.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
B	–	R*4	LOCATION OF BIN CONTAINING DATA
C	–	R*4	LOCATION OF WORK BIN
DLT	–	R*4	CLASS INTERVAL WIDTH
NB	–	R*4	BIN NUMBER TO BE PROCESSED

6.3.5.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	17	I*4	4 CHARACTER DESIGNATION OF SL TYPE
K	–	I*4	START OF WORK BIN
IA	–	I*4	HISTOGRAM SLOT NUMBER TO BE INCREMENTED
IC	–	I*4	COUNT IN A PARTICULAR HISTOGRAM SLOT
I1	–	I*4	POINTER TO START OF DATA IN DATA BIN
I2	–	I*4	POINTER TO END OF DATA IN DATA BIN
I3	–	I*4	START OF WORK BIN
I4	–	I*4	LAST POSITION IN WORK BIN
JA	–	I*4	NUMBER OF MARKERS TO BE ASSOCIATED WITH A PARTICULAR HISTOGRAM SLOT
MK	–	I*4	THE CHARACTER 'X' USED TO PRINT HISTOGRAM
AMP	–	R*4	NUMBER OF MARKERS PER COUNT
AMX	–	R*4	BIN AMPLITUDE PER MARKER
DLX	–	R*4	CLASS INTERVAL WIDTH
DOT	–	R*4	THE CHARACTER '.'
SUM	–	R*4	SUM OF VALUES IN DATA BIN
ANNN	–	R*4	NUMBER OF SAMPLES
AVAR	–	R*4	VARIANCE OF VALUES IN DATA BIN
GRID	101	R*4	101 *'S
NDIS	–	I*4	POINTER TO 16 CHARACTER TITLE
XMAX	–	R*4	LARGEST COUNT IN A HISTOGRAM SLOT
AMEAN	–	R*4	MEAN OF VALUES IN DATA BIN
SUMSQ	–	R4	SUM OF SQUARES OF VALUES IN DATA BIN

COMMON HISTO SEE ZHIST

6.3.5.4 Description - SHIST cycles through the bin accumulating the sum, sum squared of each sampled item along with a frequency of occurrence within a given class of intervals. The mean and variance of the data is computed and the desired histogram is output.

6.3.5.5 PDL - See Appendix A.

6.3.5.6 Decision Tables and Algorithms - None.

6.3.6 SLIST

6.3.6.1 Identification

- o SLIST - List Items or Output Summary
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.6.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
B	–	R*4	POINTER TO BEGINNING OF DATA IN BIN TO BE PROCESSED
IP	–	I*4	THE INDEX FOR LISTING BIN ELEMENTS(IP=1LIST EVERY ELEMENT)
NB	–	I*4	BIN NUMBER TO BE PROCESSED

6.3.6.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	17	I*4	4 CHARACTER DESIGNATION OF SL TYPE
AN	–	R*4	NUMBER OF SAMPLES INCLUDING 0'S
BX	–	R*4	FIRST DATA ITEM IN BIN
I1	–	I*4	POINTER TO START OF DATA IN BIN
I2	–	I*4	POINTER TO END OF DATA IN BIN
ANO	–	R*4	NUMBER OF SAMPLES EXCLUDING 0'S
IBX	–	I*4	FIRST DATA ITEM IN BIN
AMIN	–	R*4	MINIMUM INCLUDING 0'S
NDIS	–	I*4	POINTER TO 16 CHARACTER TITLE
AMEAN	–	R*4	MEAN INCLUDING 0'S
INDEX	–	I*4	STATION/TRIP LINK NUMBER
STDEV	–	R*4	STANDARD DEVIATION INCLUDING 0'S
AMEANO	–	R*4	MEAN EXCLUDING 0'S
STDEVO	–	R*4	STANDARD DEVIATION EXCLUDING 0'S

6.3.6.4 Description - SLIST prints out the contents of any specified bin, listing every Kth element, or performs the computations necessary for producing a statistical summary of the data. If a statistical summary has been requested, the following items are computed and displayed for all sampled values including and exclusing zero values:

1. Number of samples
2. Sum of values
3. Mean per sample
4. Standard Deviation from the mean
5. Minimum value
6. Time of minimum (seconds)
7. Maximum value
8. Time of maximum (seconds).

6.3.6.5 PDL - See Appendix A.

6.3.6.6 Decision Tables and Algorithms - None.

6.3.7 SODATA

6.3.7.1 Identification

- o AODATA - Initialize Major Comment Areas
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.7.2 Argument Dictionary - None.

6.3.7.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TITLEM	144	I*4	USED TI INITIALIZE 'TITLES' THROUGH EQUIVALENCE
TITLM1	144	I*4	"
TITLM2	144	I*4	"
TITLM3	144	I*4	"
TITLM4	144	I*4	"
TITLM5	144	I*4	"
TITLE2	12	I*4	"
TITL21	72	I*4	"
TITL22	72	I*4	"
TITL23	72	I*4	"
TITL24	72	I*4	"
TITL25	72	I*4	"
TITL26	76	I*4	"
TITLE3	72	I*4	"
TITL31	40	I*4	"
TITLE4	72	I*4	"
TITL41	72	I*4	"
TITL42	76	I*4	"
STABM	150	I*4	USED TO INITIALIZE 'MSUTAB' THROUGH EQUIVALENCE
STABM1	60	I*4	"
STAB2	118	I*4	"
STAB3	28	I*4	"
STAB4	55	I*4	"
STYPM	150	I*4	USED TO INITIALIZE 'MSUTYP' THROUGH EQUIVALENCE
STYPM1	60	I*4	"
STYP2	118	I*4	"
STYP3	28	I*4	"
STYP4	55	I*4	"

6.3.7.4 Description - SODATA serves to simply initialize many of the tables by means of a block data subprogram.

6.3.7.5 PDL - None (there is no process).

6.3.7.6 Decision Tables and Algorithms - None.

6.3.8 SONTIX

6.3.8.1 Identification

- o SONTIX - Establish PARM Field Addressibility
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.8.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
ARG1	-	I*4	(OUTPUT) NUMBER OF CHARACTERS IN FIRST PARM FIELD
ARG2	-	I*4	(OUTPUT) ADDRESS OF FIRST PARM FIELD

6.3.8.3 Local Variable Dictionary - None.

6.3.8.4 Description - Entry SONTIX obtains and saves the address of PARM field defined in execution JCL and gives control to output processor main program. Entry SOUPTX gets the number of characters in the PARM field and passes character count and address of PARM field-to routine SOZNIT.

6.3.8.5 PDL - See Appendix A.

6.3.8.6 Decision Tables and Algorithms - None.

6.3.9 SOPSUM

6.3.9.1 Identification

- o SOPSUM - Performance Summary Processing
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.9.2 Argument Dictionary - None.

6.3.9.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
DIV	-	R*4	# HOURS IN REPORT INTERVAL
SUMM	-	R*4	INTERMEDIATE TOTAL
TYPE	20	I*4	4 CHARACTER ABBREVIATION OF LINK TYPE

6.3.9.4 Description

6.3.9.4 Description - SOPSUM computes the required performance summary measures from the sums and maximum values accumulated during the data acquisition process. This processing involves the computation of average rates/hour and system wide averages. For the case of average times, these are computed from data passed to the output process in each type 2 record. Once all values have been computed, they are formatted along with required maximum values for outputting to the performance summary file. Prior to actual writing of the file, the index file is updated to reflect performance summary computations.

6.3.9.5 PDL - See Appendix A.

6.3.9.6 Decision Tables and Algorithms - None.

6.3.10 SOUTPT

6.3.10.1 Identification

- o SOUTPT - Output Processor Control
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.10.2 Argument Dictionary - None.

6.3.10.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
IN	–	I*4	NUMERIC CARD TYPE(FLAG,REQU,READ,...)
AEND	–	L*1	END-OF-FILE REQUEST CARDS
FND	–	L*1	INDICATOR THAT CARD TYPE HAS BEN FOUND
TEMP	–	L*1	FIRST CHARACTER OF NAME OF DATA ITEM(IGNORED)
MC	–	I*4	MAJOR CATEGORY CODE(1=SYST/2=STN/3=TRIP)
XX	–	R*4	SAMPLE INTERVAL IN SECONDS
BIN	–	I*4	BIN USED TO HOLD DATA
SUB	–	I*4	NAME OF DATA ITEM=SUBCATEGORY REQUESTED ON CARD
FORM	–	I*4	OUTPUT FORMAT REQUESTED ON CARD(LIST,SUMM,PLOT, HIST,PERF)
IDHI	–	I*4	HIGH LINK NUMBER REQUESTED ON CARD
IDLO	–	I*4	LOW LINK NUMBER REQUESTED ON CARD
MAIN	–	I*4	MAIN CATEGORY REQUESTED ON CARD
XTERM	–	I*4	OUTPUT TO TERMINAL INDICATOR(UNUSED)
NAME	–	I*4	TYPE OF CARD(FLAG,REQU,READ,...)
IFORM	–	I*4	NUMERIC FORM REQUESTED

6.3.10.4 Description

6.3.10.4 Description - Output Processor Control provides the basic mechanism for recognizing user output requests and involving service components required to satisfy those requests. Control is passed to Output Processor Control from an auxilary entry point defined for saving PARM field information (PDL segment SONTIX) necessary for later index file updating. Upon entry, Output Processor Control (PDL segment SOUTPT) performs initialization of the bin storage areas (PDL segment SOZNIT). The basic control loop for recognizing user output requests is then started. The basic loop consists of the following processing which is performed until the last user request is satisfied:

1. Read user service request and classify it as to whether it specifies required data to be collected or the acquisition and display of data.

2. If the request is for data, determine the number of requests which must be filed for data acquisition and perform request filing)PDL segment ZREQU). Each entity specified in a data request requires a separate bin storage area for data acquisition. Thus, a range of entities specified on one data request causes the automatic generation of multiple internal data requests as does a request for performance summary output.

3. If the request if for data acquisition (READ Command), reading of the raw statistics file (PDL segment SZREAD) and data accumulation within the bin areas is performed. Once completed, the appropriate data manipulation and display is performed for each service request, previously filed in the request table (PDL segments ZHIST, ZLIST, SZPLOT).

Once data display has been completed, the control loop is recycled to begin processing of the next user specified group of service requests. Finally SOWTIW is called to list the members that were used in the index file.

6.3.10.5 PDL - See Appendix A.

6.3.10.6 Decision Tables and Algorithms - None.

6.3.11 SOWTIX

6.3.11.1 Identification

- o SOWTIX - Update Index File
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.11.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
COUNT	-	I*2	NUMBER OF CHARACTERS CONTAINED IN THE PARM FIELD
STRING	8	L*1	PARM FIELD INFORMATION SUPPLYING THE MEMBER BEING UPDATED IN THE PERFORMANCE SUMMARY FILE

6.3.11.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
MONTH	-	I*2	MONTH OF YEAR
DAY	-	I*2	DAY OF THE MONTH
YEAR	-	I*2	YEAR
MIN	-	I*2	MINUTE OF THE DAY
BLK	-	L*1	BLANK CHARACTER ' '

6.3.11.4 Description - SOWTIX parses the parm list to get individual names. Then DAYTIM is called to get the date and time. Next, the load module name is written with the date and time to the index. When SOWTIX is called from SOUTPT, it writes the member name of the performance summary file into that file. When SOWTIW is called by SOUTPT, it lists the members that were used during the run in the index.

6.3.11.5 PDL - See Appendix A.

6.3.11.6 Decision Tables and Algorithms - None.

6.3.12 SOZNIT

6.3.12.1 Identification

- o SOZNIT - Initialization of the Output Processor
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.12.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NAREA	–	I*4	TOTAL SIZE IN WORDS OF BIN STORAGE AREA
NBINS	–	I*4	NUMBER OF BINS REQUIRED
NREQU	–	I*4	MAXIMUM NUMBER OF REQUESTS
NLINE	–	I*4	NUMBER OF LINES/PAGE FOR OUTPUT FORMATTING

6.3.12.3 Local Variable Dictionary - None.

6.3.12.4 Description - Initialization is performed to establish initial conditions for the output processing of a Raw Statistics File. Initial bin allocations (PDL segment ZDBIN) is performed to create a default number of bins in the storage area. This includes cycling through the bin storage area and establishing each five locations in the area as a bin with the following characteristics defined:

1. Total number of words allocated to bin (=5)
2. Bin number
3. Starting index of bin data
4. Ending index of bin data
5. Identification mnemonic = 0.

Any remaining space in the bin storage area is defined as a large bin which serves as the basis for dynamic bin storage are allocated during data acquisition and manipulation processing.

Once the bin storage area is initialized, default parameters for raw statistics processing are established from header data (PDL segment

SZREAD) containing characteristics of the sampling experiment used in generating the Raw Statistics File as follows:

1. Number of station links
2. Number of trip links
3. Clock units used
4. Sampling interval.

These data are acquired from the file by filing a system service request and invoking the data acquisition process in a manner analogous to processing of user service commands.

6.3.12.5 PDL - See Appendix A.

6.3.12.6 Decision Tables and Algorithms - None.

6.3.13 SREAD02

6.3.13.1 Identification

- o SREAD02 - Read System Statistics
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.13.2 Argument Dictionary - None.

6.3.13.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
END	-	L*1	INDICATES END OF FILE IN WHILE READING FOLLOWER
FOLLOW	-	R*8	'FOLLOWER'
FSTAT	36	I*4	HOLD STATISTICS OF TYPE & LENGTH I*4
HSTAT	46	I*2	HOLD STATISTICS OF TYPE & LENGTH I*2
RSTAT	24	R*4	HOLD STATISTICS OF TYPE & LENGTH R*4
VAL	-	R*4	A PARTICULAR VALUE OF ONE STATISTIC
CUSAM	-	R*4	CU PER SAMPLE
VSUB	-	R*4	UNUSED
STAT	219	R*4	MISCELLANEOUS STATISTICS
STAT1	6	R*4	R*4 - IN STN
STAT2	6	R*4	R*4 - IN BOARD
STAT3	6	R*4	R*4 - IN DEBOARD
STAT4	6	R*4	R*4 - LAUNCH
STAT5	9	I*4	I*4 - IN STN
STAT6	9	I*4	I*4 - IN BOARD
STAT7	9	I*4	I*4 - IN DEBOARD
STAT8	9	I*4	I*4 - LAUNCH
STAT9	12	I*2	I*2 - IN STN
STAT10	12	I*2	I*2 - IN BOARD
STAT11	12	I*2	I*2 - IN DEBOARD
STAT12	12	I*2	I*2 - LAUNCH
IREQ	-	I*4	REQUEST TABLE ENTRY ASSOCIATED WITH ITEM
ISUB	-	I*4	SUBCATEGORY ASSOCIATED WITH ITEM
ISUB1	-	I*4	USED IN COMPUTING POSITION OF NEXT DESIRED STATISTIC IN RECORD
ISUB2	-	I*4	USED IN COMPUTING POSITION OF NEXT DESIRED STATISTIC IN RECORD
STLNL	-	R*4	USED TO READ DATA FOR COMPUTING AVERAGE TIMES FOR THE PERFORMANCE SUMMARY

6.3.13.4 Description - SREAD02 reads sampling records containing system statistics written to the raw statistics file each sample interval by the model processor into a buffer from which requested statistics can be retrieved. The requested items are retrieved by cycling through the request table and obtaining the appropriate sampled item from the buffer based on the subcategory index contained in the request table entry. If the request indicates that performance summary data is required, the sum, maximum and minimum values for the first 219 system statistics are automatically accumulated for later processing by SOPSUM. As each required value is retrieved, it is stored in an assigned bin storage location for later processing and outputting.

6.3.13.5 PDL - See Appendix A.

6.3.13.6 Decision Tables and Algorithms - None.

6.3.14 SREAD03

6.3.14.1 Identification

- o SREAD03 - Read Station Link Statistics
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.14.2 Argument Dictionary - None.

6.3.14.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
END	-	L*1	INDICATES END OF FILE IN WHILE READING FOLLOWER
FOLLOW	-	R*8	'FOLLOWER'
FSTAT1	900	I*4	HOLD STATISTICS OF TYPE & LENGTH I*4
HSTAT1	1200	I*2	HOLD STATISTICS OF TYPE & LENGTH I*2
RSTAT1	600	R*4	HOLD STATISTICS OF TYPE & LENGTH R*4
VAL	-	R*4	A PARTICULAR VALUE OF ONE STATISTIC
CUSAM	-	R*4	CU PER SAMPLE
VSUB	-	R*4	UNUSED
IREQ	-	I*4	REQUEST TABLE ENTRY ASSOCIATED WITH ITEM
ISUB	-	I*4	SUBCATEGORY ASSOCIATED WITH ITEM
ISUB1	-	I*4	USED IN COMPUTING POSITION OF NEXT DESIRED STATISTIC IN RECORD
ISUB2	-	I*4	USED IN COMPUTING POSITION OF NEXT DESIRED STATISTIC IN RECORD
SL	-	I*4	STATION LINK NUMBER

6.3.14.4 Description - SREAD03 reads sampling records containing station link stats written to the raw statistics file at each sample interval by the model processor into a buffer from which requested statistics can be retrieved. The requested items are retrieved by cycling through the request table and obtaining the appropriate sampled item from the buffer based on the subcategory index contained in the request table entry. As each required value retrieved it is stored in an assigned bin storage location for later processing and outputting.

6.3.14.5 PDL - See Appendix A.

6.3.14.6 Decision Tables and Algorithms - None.

6.3.15 SREAD04

6.3.15.1 Identification

- o SREAD04 - Read Trip Link Statistics
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.15.2 Argument Dictionary - None.

6.3.15.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
END	-	L*1	INDICATES END OF FILE IN WHILE READING FOLLOWER
FOLLOW	-	R*8	'FOLLOWER'
FSTAT1	900	I*4	HOLD STATISTICS OF TYPE & LENGTH I*4
HSTAT1	1200	I*2	HOLD STATISTICS OF TYPE & LENGTH I*2
RSTAT1	600	R*4	HOLD STATISTICS OF TYPE & LENGTH R*4
VAL	-	R*4	A PARTICULAR VALUE OF ONE STATISTIC
CUSAM	-	R*4	CU PER SAMPLE
VSUB	-	R*4	UNUSED
IREQ	-	I*4	REQUEST TABLE ENTRY ASSOCIATED WITH ITEM
ISUB	-	I*4	SUBCATEGORY ASSOCIATED WITH ITEM
ISUB1	-	I*4	USED IN COMPUTING POSITION OF NEXT DESIRED STATISTIC IN RECORD
ISUB2	-	I*4	USED IN COMPUTING POSITION OF NEXT DESIRED STATISTIC IN RECORD
TL	-	I*4	TRIP LINK NUMBER

6.3.15.4 Description - SREAD04 reads sampling records containing trip link statistics written to the raw statistics file for each sample interval by the model processor into a buffer from which requested statistics can be retrieved. The requested items are retrieved by cycling through the request table and obtaining the appropriate sampled item from the buffer based on the subcategory index contained in the request table entry. As each required value is retrieved, it is stored in an assigned bin storage location for later processing and outputting.

6.3.15.5 PDL - See Appendix A.

6.3.15.6 Decision Tables and Algorithms - None.

6.3.16 SREQTLU

6.3.16.1 Identification

- o SREQTLU - Record/Request Correlation
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.16.2 Argument Dictionary - None.

6.3.16.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
L	-	I*4	FIRST AVAILABLE SPACE IN BIN
NOW	-	I*4	USED TO LOOP THROUGH REQUEST TABLE CHAIN
SUB	-	I*4	SUBCATEGORY 4 CHARACTER ABBREVIATIONS
IDNO	-	I*4	SUBCATEGORY FROM REQUEST TABLE
IREQ	-	I*4	REQUEST NUMBER(INDEX TO ZREQUE)
ISUB	-	I*4	SUBCATEGORY
MAIN	-	I*4	MAIN CATEGORY 4 CHARACTER ABBREVIATION
NEXT	-	I*4	USED TO LOOP THRU REQUEST TABLE CHAIN
NREQ	-	I*4	REQUEST NUMBER
IFLAG	-	I*4	UNUSED
IMAIN	-	I*4	MAIN CATEGORY NUMBER

6.3.16.4 Description - SREQTLU is invoked each time a record of a particular type is encountered in the Raw Statistics File. The Record/Request Correlation process involves cycling through each request table entry. Each time a request requiring the particular record type is encountered, it is chained to the previous request requiring the record type and the major and subcategory indices are converted to numerical values.

6.3.16.5 PDL - See Appendix A.

6.3.16.6 Decision Tables and Algorithms - None.

6.3.17 SSETUP

6.3.17.1 Identification

- o SSETUP - Initialize Data Tables
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.17.2 Argument Dictionary - None.

6.3.17.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
INFINY	–	R*8	LARGEST I*4 NUMBER POSSIBLE
L	–	I*4	FIRST AVAILABLE SPACE IN BIN

6.3.17.4 Description - SSETUP reinitializes the match table which is used in establishing record request correlation to a specified initial state as described within the SODATA block data routine. The requested form of each entry in the request table is validated and optionally the tables used by the output processor are displayed.

6.3.17.5 PDL - See Appendix A.

6.3.17.6 Decision Tables and Algorithms - None.

6.3.18 SZPLOT

6.3.18.1 Identification

- o SZPLOT - Plot Output Control
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.18.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
XO	-	R*4	STARTING X VALUE
DELTAX	-	R*4	X INCREMENT
NDELTA	-	I*4	# POINTS TO BE PLOTTED
NY	-	I*4	# BINS TO BE PLOTTED
N1	-	I*4	BIN #1
N2	-	I*4	BIN #2
N3	-	I*4	BIN #3
N4	-	I*4	BIN #4
BOTTOM	-	R*4	LOWER LIMIT ON Y VALUES
TOP	-	R*4	UPPER LIMIT ON Y VALUES
SYMBOL	4	R*4	PLOTTING SYMBOLS

6.3.18.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TYPE	17	I*4	4 CHARACTER DESIGNATION OF SL TYPES
J1	-	I*4	# OF FIRST BIN TO BE PLOTTED
J2	-	I*4	UNUSED
J3	-	I*4	UNUSED
J4	-	I*4	UNUSED
TOP	-	R*4	LARGEST VALUE TO BE PLOTTED
NDIS	-	I*4	POINTER TO 16 CHARACTER TITLE
INDEX	-	I*4	STATION/TRIP LINK NUMBER
BOTTOM	-	I*4	SMALLEST VALUE TO BE PLOTTED
NDELTA	-	I*4	NUMBER OF VALUES TO BE PLOTTED

6.3.18.4 Description - SZPLOT is invoked to provide a time series plot of sampled data items. The actual data accumulation, scaling, and formatting is performed by GRAPH. It formats the required output by manipulating the contents of a bin and outputting the desired results. Format processing includes establishing necessary grids and titles, and establishing scaling factor applied to data for accommodating the image size on the output medium (page size).

6.3.18.5 <u>PDL</u> - See Appendix A.

6.3.18.6 <u>Decision Tables and Algorithms</u> - None.

6.3.19 SZREAD

6.3.19.1 Identification

- o SZREAD - Data Acquisition of System Constants
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.19.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
TAPE	-	I*4	FORTRAN UNIT NUMBER FOR RAW STATISTICS FILES
START	-	I*4	BEGIN TIME OF ACQUISITION INTERVAL
STOP	-	I*4	STOP TIME OF ACQUISITION INTERVAL

6.3.19.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
JUNK	-	I*4	DUMMY ARGUMENT USED TO CALL HEADER

6.3.19.4 Description - This routine reads Raw Statistics File to acquire data items from samples within a start/stop interval as required to service previous data requests. The data acquisition process is initiated by Output Processor Control in response to a Read Command. The data acquisition process (PDL segment SZREAD) is partitioned into three functions:

1. Obtain initial, critical data from the tape and perform various other initializations.

2. Skip to the beginning of the request interval.

3. Read groups of records from the tape, ascertain whether a group has requested data within it, obtain the requested data, and store it into the appropriate bin.

Since data acquisition serves to obtain system default parameters during initialization, a check is made to determine if this is the initial read of the Raw Statistics File. If it is the initial read, the default parameters are read from the initial file header. The time units specified for the simulation experiment acquired during this processing are used in subsequent data acquisition processing as described below. Entry into the data acquisition process for satisfying data requests begins with data table initialization (PDL segment SSETUP) and establishing request/record correlation (PDL segment SREQTLU), conversion of the request interval to simulator clock units and repositioning of raw statistics file at its beginning. The Raw Statistics File is read, processing each header record (PDL segment SHEADER) and skipping successive records (PDL segment SSKIPFO) until the file is positioned to the start of the read (acquisition) interval. Basically, in this process, record groups are read and their followers are skipped until one is found whose time is not less than the interval start time. Two important exceptions apply during the record skipping process:

1. The end of the tape is indicated by a special header record type number, which must be detected.

2. Those record groups containing critical information that must be read (indicated by a major category indicator of 1) are detected and their follower records are read as appropriate.

Once the file is positioned to the beginning of the read interval, subsequent records are read and one of three actions is taken based upon the initial setting of the major category indicator and summarized below:

1. 0 -- Meaning that records of type 0 are not needed, the follower records are skipped, and the next header is read.

2. 1 -- Meaning that following records of 1 are needed processing for acquiring and storing data is performed.

3. -1 -- Meaning that the records might be needed, but whether or not they are has yet to be determined. At this time, the program must determine if they are or are not needed by invoking the data matching function previously described. The result of determining whether this record type is required results in changing the major category indicator to 0, indicating the first request requiring data from the record type.

Actual data acquisition from required record types is performed by I/O processing based on individual record type for the major data category indicated in the record group header (PDL segments SREAD02, SREAD03, and SREAD04). This processing iterates upon each of the follower records in turn and then upon each of the requests in the request table associated with the particular record type (as defined by the chain beginning with the major category indicator).

If the main category is one that requires no entity index number (e.g., as for system as opposed to link, which does), then only one follower record exists and it contains a single set of data items. However, if the main category can have an associated entity number (e.g., a particular link number for the link category), then each follower contains several replications of data items, one each for several entity indexes. In this case, I/O processing (besides iterating on the followers and request lines) must also iterate upon the number of data item replications in a particular follower record.

For each iteration, the required appropriate read routine for the specific record type is called to store the data for processing as contained in a follower record. Each required data item is located within the record, retrieved, and stored in the appropriate bin area. In general, data position information is determined from the major category and sub-category indices contained in each request table entry as the result of performing the data mapping function. If during the store process (PDL segment SSTORE), a bin becomes full, it is automatically reallocated to contain more space (PDL segment SBNCHK). Thus, the file reading process does not require the user to "second guess" how much of each type of data actually resides in the Raw Statistics File. Once storage of a data item has been performed, the bin space pointers contained in the request table entry are updated to reflect bin usage.

6.3.19.5 PDL - See Appendix A.

6.3.19.6 Decision Tables and Algorithms - None.

6.3.20 ZABIN

6.3.20.1 Identification

- o ZABIN - Bin Reallocation
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.20.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NB	-	I*4	BIN NUMBER TO BE CHECKED
IP	-	I*4	REQUIRED BIN SIZE IN WORDS

6.3.20.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
K	-	I*4	POINTER TO AREA BEYOND BIN
L	-	I*4	CURRENT LOCATION OF BIN
I1	-	I*4	POINTER TO START OF OLD BIN
I2	-	I*4	POINTER TO START OF NEW BIN
KL	-	I*4	POINTER TO BLANK AREA BEYOND EXPANDED BIN
LMX	-	I*4	NUMBER OF BIN POSITIONS TO BE MOVED
IDIF	-	I*4	LEFT OVER BIN AREA
IREAL	-	I*4	REQUIRED BIN SIZE PLUS 4 FOR BIN HEADER
ISIZE	-	I*4	CURRENT SIZE OF BIN

6.3.20.4 Description - This component (PDL segment ZABIN) is invoked to ensure that proper bin space exists to support a completely new set of data (after bin area initialization or subsequent processing iterations for a new set of user requests). The following processing is performed:

1. If the bin has enough space allocated already, then:

   a. If the allocation is four positions or more than required, the extra space is made into a pseudo-bin (available space in bin storage area).

   b. If the allocation is within four positions of required, no changes are made.

2. If more space is needed, then the currently allocated bin area is changed to a pseudo-bin and an attempt is made to relocate the bin as:

   a. If the back of the bin storage area has enough unused space, the bin is placed there.

   b. If the back of the bin storage area does not have enough space, then:

      (1) All bins are moved towards the top of the bin area by eliminating any pseudo-bins that may be interspersed.

      (2) Test 2(a) above is repeated. If it fails this time, no additional space is available and processing terminates.

When any bin is relocated (including those moved up in Step (1) above), the corresponding entry in the bin location pointer is changed. If the specified bin is currently in use (data in it that must be preserved) the following processing is performed:

1. If sufficient space has been allocated, no changes are made.

2. If sufficient space has not been allocated, but a pseudo-bin immediately follows the bin being allocated, then:

   a. If the total space of the two bins (real plus pseudo) is within four positions of the requirements, the total space is allocated to the real bin; the pseudo-bin is eliminated.

   b. If the total space exceeds the requirement by at least four positions, then the excess psace over and above the required space is made into a pseudo-bin.

3. If the bin cannot remain where it is, then an attempt to find a new location of sufficient space is made. First, the empty area at the end of the bin storage area is checked.

   a. If the end of the area is large enough, the old bin contents are copied into it, the previous bin location is set to a pseudo-bin, and the array bin location pointer is updated.

   b. If the end of the bin storage area has sufficient space, all bins are moved up by eliminating pseudo-bins. Test 3(a) is then repeated. If it fails, step (4) is tried.

4. The amount of area covered by the bin itself plus the space available back of the bin storage is checked. If this is below the required space, processing terminates. Otherwise:

   a. If the bin being allocated and the free area are adjacent, the bin is simply enlarged by using part of the free area space in the bin storage area.

   b. If the two areas are not adjacent, then all bins between the current one and the free area are moved downward to provide the necessary space.

ZABIN checks if sufficient space has been allotted to a bin and if not provides the changes necessary to provide the required bin space. Either the original bin is left unchanged or its size is increased to some specified number of words. In either case, the previous contents of the bin are left unchanged. If the expansion of a bin requires a change of location in the bin storage area, all appropriate pointers are updated to reflect the new mapping of the bin storage area.

6.3.20.5 PDL - See Appendix A.

6.3.20.6 Decision Tables and Algorithms - None.

6.3.21 ZBINL

6.3.21.1 Identification

- o ZBINL - Get Length of Data in Bin
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.21.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NBIN	-	I*4	BIN NUMBER

6.3.21.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
LENGTH	-	I*4	LENGTH OF BIN

6.3.21.4 Description - ZBINL returns the length in bytes of a specified bin.

6.3.21.5 PDL - See Appendix A.

6.3.21.6 Decision Tables and Algorithms - None.

6.3.22 ZBNCHK

6.3.22.1 Identification

- o ZBNCHK - Bin Expansion
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.22.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
NB	–	I*4	BIN NUMBER TO BE CHECKED
IP	–	I*4	REQUIRED BIN SIZE IN WORDS

6.3.22.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
K	–	I*4	POINTER TO AREA BEYOND BIN
L	–	I*4	CURRENT LOCATION OF BIN
I1	–	I*4	POINTER TO START OF OLD BIN
I2	–	I*4	POINTER TO START OF NEW BIN
KL	–	I*4	POINTER TO BLANK AREA BEYOND EXPANDED BIN
LMX	–	I*4	NUMBER OF BIN POSITIONS TO BE MOVED
IDIF	–	I*4	LEFT OVER BIN AREA
IREAL	–	I*4	REQUIRED BIN SIZE PLUS 4 FOR BIN HEADER
ISIZE	–	I*4	CURRENT SIZE OF BIN

6.3.22.4 Description - This component (PDL segment ZBNCHK) is invoked to ensure the expansion of existing bins as necessary to support data acquisition requirements. The following processing is performed:

1. If the bin has enough space allocated already, then:

   a. If the allocation is four positions or more than required, the extra space is made into a pseudo-bin (available space in bin storage area).

   b. If the allocation is within four positions of required, no changes are made.

2. If more space is needed, then the currently allocated bin area is changed to a pseudo-bin and an attempt is made to relocate the bin as:

a. If the back of the bin storage area has enough unused space, the bin is placed there.

b. If the back of the bin storage area does not have enough space, then:

(1) All bins are moved towards the top of the bin area by eliminating any pseudo-bins that may be interspersed.

(2) Test 2(a) above is repeated. If it fails this time, no additional space is available and processing terminates.

When any bin is relocated (including those moved up in Step (1) above), the corresponding entry in the bin location pointer is changed. If the specified bin is currently in use (data in it that must be preserved) the following processing is performed:

1. If sufficient space has been allocated, no changes are made.

2. If sufficient space has not been allocated, but a pseudo-bin immediately follows the bin being allocated, then:

   a. If the total space of the two bins (real plus pseudo) is within four positions of the requirements, the total_space is allocated to the real bin; the pseudo-bin is eliminated.

   b. If the total space exceeds the requirement by at least four positions, then the excess space over and above the required space is made into a pseudo-bin.

3. If the bin cannot remain where it is, then an attempt to find a new location of sufficient space is made. First, the empty area at the end of the bin storage area is checked.

   a. If the end of the area is large enough, the old bin contents are copied into it, the previous bin location is set to a pseudo-bin, and the array bin location pointer is updated.

   b. If the end of the bin storage area has sufficient space, all bins are moved up by eliminating pseudo-bins. Test 3(a) is then repeated. If it fails, step (4) is tried.

4. The amount of area covered by the bin itself plus the space available back of the bin storage is checked. If this is below the required space, processing terminates. Otherwise:

   a. If the bin being allocated and the free area are adjacent, the bin is simply enlarged by using part of the free area space in the bin storage area.

b. If the two areas are not adjacent, then all bins between the current one and the free area are moved downward to provide the necessary space.

6.3.22.5 <u>PDL</u> - See Appendix A.

6.3.22.6 <u>Decision Tables and Algorithms</u> - None.

6.3.23 ZDBIN

6.3.23.1 Identification

- ZDBIN - Allocate Bin Storage
- IBM/FSD - July 1, 1977
- PARAFOR

6.3.23.2 Argument Dictionary - None.

6.3.23.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
KX	-	I*4	POINTER TO END OF CURRENT BIN
KNN	-	I*4	POINTER TO BEGINNING OF REMAINING BIN AREA
LONG	-	I*4	5 = INITIAL LENGTH OF BIN

6.3.23.4 Description - ZDBIN defines an initial number of bins in the bin area each having header information initialized to indicate the bin is currently empty and available for use. Any space remaining in the storage area after definition is complete is allocated to one large bin area.

6.3.23.5 PDL - See Appendix A.

6.3.23.6 Decision Tables and Algorithms - None.

6.3.24 ZDUMBIN

6.3.24.1 Identification

- o ZDUMBIN - Formatted Dump of Bin Area
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.24.2 Argument Dictionary - None.

6.3.24.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
J	–	I*4	POINTER TO START OF DATA IN BIN HEADER
I1	–	I*4	TOTAL WORDS ALLOCATED TO BIN
I2	–	I*4	LOGICAL BIN NUMBER
I3	–	I*4	POINTER TO START OF DATA
I4	–	I*4	POINTER TO END OF DATA
I5	–	I*4	LENGTH OF DATA IN BIN
JNN	–	I*4	JN-5
JTOT	–	I*4	NUMBER OF FREE WORDS
MTOT	–	I*4	NUMBER OF ITEMS IN BINS
NTOT	–	I*4	TOTAL NUMBER OF WORDS ALLOCATED TO BINS

6.3.24.4 Description - ZDUMBIN produces a formatted dump of the bin storage area as an aid to debugging.

6.3.24.5 PDL - See Appendix A.

6.3.24.6 Decision Tables and Algorithms - None.

6.3.25 ZERROR

6.3.25.1 Identification

- o ZERROR - Write Error Message and Continue/Terminate
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.25.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
MSGNO	-	I*4	ERROR MESSAGE NUMBER
MSG	2	L*1	MESSAGE TEXT
MSEVER	-	I*4	MESSAGE SEVERITY

6.3.25.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
PGM	3	I*4	PROCESSOR ABBREVIATION
SCLN	-	L*1	SEMICOLIN
TYPE	3	L*1	ALPHA SEVERITY DESIGNATIONS
MSGTYP	-	L*1	MESSAGE TYPE CHARACTER
XCLOCK	-	R*4	TIME OF CURRENT SAMPLE BEING PROCESSED
NUM	-	I*4	INDEX TO MSGC & MSGCN

6.3.25.4 Description - ZERROR issues a specified error message according to a fixed format consisting of number, type, descriptive and test. It accumulates counts of messages by type and number and gracefully terminates if error limits are exceeded by providing a trace of subroutine calls leading to termination (see SERROR in MP section).

6.3.25.5 PDL - See Appendix A.

6.3.25.6 Decision Tables and Algorithms - None.

6.3.26 ZFLAG

6.3.26.1 Identification

- o ZFLAG - Intermediate Output Flag Setting
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.26.2 Argument Dictionary - None.

6.3.26.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
FINI	-	L*1	INDICATES END OF FLAG FIELDS FOUND
TEMP	18	I*4	HOLD 18 FIELDS FROM FLAG FOLLOWER CARD

6.3.26.4 Description - ZFLAG initializes all flag setting to zero and then turns a specified set of flags as requested by the user (see SAFLAG in MP section).

6.3.25.5 PDL - See Appendix A.

6.3.26.6 Decision Tables and Algorithms - None.

6.3.27 ZGRAPH

6.3.27.1 Identification

- o ZGRAPH - Produce Time Series Plot
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.27.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
X0	-	R*4	STARTING X VALUE
DELTAX	-	R*4	X INCREMENT
NDELTA	-	I*4	# POINTS TO BE PLOTTED
NY	-	I*4	# BINS TO BE PLOTTED
Y1	-	R*4	BIN #1
Y2	-	R*4	BIN #2
Y3	-	R*4	BIN #3
Y4	-	R*4	BIN #4
BOTTOM	-	R*4	LOWER LIMIT ON Y VALUES
TOP	-	R*4	UPPER LIMIT ON Y VALUES
SYMBOL	4	R*4	PLOTTING SYMBOLS

6.3.27.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
PLOT	101	R*4	EVERY 10TH LINE TO BE PLOTTED
GRID	101	R*4	1ST THRU 9TH LINES TO BE PLOTTED
GRDMK	11	R*4	Y SCALE VALUES FOR TITLING
TITLE	21	R*4	UNUSED
LABEL	21	I*4	UNUSED
LABAS	21	I*4	UNUSED
Y1	-	R*4	VALUE OF FIRST GRAPH PLOTTED
Y2	-	R*4	UNUSED
Y3	-	R*4	UNUSED
Y4	-	R*4	UNUSED
BLANK	-	R*4	6 BLANKS ' '
DASH	-	R*4	6 DASHES '------'
CROSS	-	R*4	6 DOTS '......'
J	-	I*4	POINTER TO VALUE TO BE PLOTTED
X	-	R*4	SAMPLE NUMBER
SYM	-	R*4	SYMBOL(1)

LAB1	–	I*4	LABEL(1)
LAB2	–	I*4	LABEL(2)
LINE	–	I*4	COUNT OF LINES PRINTED
NOUT	–	I*4	SYSOUT UNIT NUMBER
YMIN	–	R*4	MINIMUM OF ALL BINS TO BE PLOTTED
YMAX	–	R*4	MAXIMUM OF ALL BINS TO BE PLOTTED
LIMIT	–	I*4	NUMBER OF POINTS TO BE PLOTTED
SAVE1	–	R*4	HOLD PREVIOUS VALUE OF GRID/PLOT
SAVE2	–	R*4	UNUSED
SAVE3	–	R*4	UNUSED
SAVE4	–	R*4	UNUSED
NLABEL	–	I*4	UNUSED
ORDSCL	–	R*4	ORDINAL SCALE USED TO COMPUTE LOCATION OF SYMB
SAMSCL	–	R*4	UNUSED

6.3.27.4 Description - ZGRAPH sets up grid lines to be displayed, computes scaling factors, scales data points and produces desired hardcopy output.

6.3.27.5 PDL - See Appendix A.

6.3.27.6 Decision Tables and Algorithms - None.

6.3.28 ZHEADER

6.3.28.1 Identification

- o ZHEADER - Read Next Header Record
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.28.2 Argument Dictionary - None.

6.3.28.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
HEADRE	–	R*8	*HEADER

6.3.28.4 Description - ZHEADER reads next record from the Raw Statistics File. If expected header is not found, it issues a warning message. If an I/O error is encountered, it issues a warning message.

6.3.28.5 PDL - See Appendix A.

6.3.28.6 Decision Tables and Algorithms - None.

6.3.29 ZHIST

6.3.29.1 Identification

- o ZHIST - Histogram Output Control
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.29.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
IQA	–	I*4	BIN NUMBER
ZA	–	R*4	CLASS INTERVAL WIDTH

6.3.29.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
DMY	5	R*4	USED LIKE BIN IN CALL TO ZMNMX; FIRST 2 POSITIONS UNUSED;3=MIN;4=MAX;5=RANGE
COMMON OUTPUT			
NBIN	10	I*4	NBIN(1)=BIN FOR HISTOGRAM/NBIN(2)=JN
PAR	7	R*4	PAR(1)=CLASS INTERVAL WIDTH
IPAR	7	I*4	UNUSED
COMMON HISTO			
MIN	–	I*4	UNUSED
MAX	–	I*4	UNUSED
AMAX	–	I*4	LARGEST VALUE IN HISTOGRAM
AMIN	–	I*4	SMALLEST VALUE IN HISTOGRAM
NSLOT	–	I*4	NUMBER OF SLOTS IN HISTOGRAM

6.3.29.4 Description - ZHIST is invoked to produce a histogram of sampled items contained in a bin storage area. The acquisition of data is performed by SHIST. Prior to displaying the histogram, the minimum and maximum values of the sampled items are determined (PDL segment ZMNMX).

It formats the required output by manipulating the contents of a bin and outputting the desired results. Format processing includes establishing necessary grids and titles.

6.3.29.5 PDL - See Appendix A.

6.3.29.6 Decision Tables and Algorithms - None.

6.3.30 ZLIST

6.3.30.1 Identification

- o ZLIST - List Output Control
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.30.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
IQA	–	I*4	BIN NUMBER
IZA	–	I*4	K (KTH ELEMENT LISTING INDICATOR)

6.3.30.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
COMMON OUTPUT			
NBIN	10	I*4	NBIN(I)=NUMBER OF BIN TO BE LISTED
PAR	7	R*4	UNUSED
IPAR	7	I*4	IPAR(I)=K (KTH ELEMENT LISTING INDICATOR)

6.3.30.4 Description - ZLIST is invoked to produce either listing of sampled data items or a statistical summary. The actual acquisition of the data is performed by SLIST which retrieves each required data values within a specific bin based on the start and stop indices contained in the bin header.

It formats the required output by manipulating the contents of a bin and outputting the desired results. Format processing includes establishing necessary titles.

6.3.30.5 PDL - See Appendix A.

6.3.30.6 Decision Tables and Algorithms - None.

6.3.31 ZMNMX

6.3.31.1 Identification

- o ZMNMX - Compute Minimum and Maximum Values
- o IBM/FSD - July 1, 1977.
- o PARAFOR

6.3.31.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
B	21	R*4	SPECIFIED BIN
C	21	R*4	BIN FOR STORING MIN, MAX & RANGE
IP	-	I*4	WORD IN BIN C FOR STORING COMPUTED VALUES

6.3.31.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
IC	-	I*4	C(1)
I1	-	I*4	START OF DATA
I2	-	I*4	END OF DATA
BMAX	-	R*4	MAXIMUM VALUE FOUND
BMIN	-	R*4	MINIMUM VALUE FOUND
RANGE	-	R*4	BMAX-BMIN

6.3.31.4 Description - ZMNMX cycles through a specified bin, determines the minimum, maximum data values and computes the range given by the difference and stores in some specified bin location.

6.3.31.5 PDL - See Appendix A.

6.3.31.6 Decision Tables and Algorithms - None.

6.3.32 ZRCLEAN

6.3.32.1 Identification

- ZRCLEAN - Reset Bin Addresses
- IBM/FSD - July 1, 1977
- PARAFOR

6.3.32.2 Argument Dictionary - None.

6.3.32.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
L	-	I*4	POINTER TO BIN

6.3.32.4 Description - ZRCLEAN cycles through the bin storage area and resets the end data location within each allocated bin as identified in the request table to its initial value.

6.3.32.5 PDL - See Appendix A.

6.3.32.6 Decision Tables and Algorithms - None.

6.3.33 ZREQU

6.3.33.1 Identification

- o ZREQU - Request Handling
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.33.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
FORM	–	I*4	FORM OF REQUESTED OUTPUT
MAIN	–	I*4	MAIN CATEGORY MNEMONIC
SUB	–	I*4	SUBCATEGORY MNEMONIC
IDNOA	–	I*4	LOW INDEX
IDNOB	–	I*4	HIGH INDEX
BINA	–	I*4	BEGINNING BIN NUMBER
SIZE	–	I*4	DUMMY ARGUMENT = 0

6.3.33.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
IA	40	I*4	FIRST 40 WORDS OF BIN AREA
FORMS	10	I*4	'TSER' 'STAT' & 8 0'S
FMSIZE	10	I*4	100, 11, & 8 0'S
L	–	I*4	POINTER TO BIN TO BE PROCESSED
LD	–	I*4	POINTER TO START OF DATA IN BIN
BIN	–	I*4	BIN TO BE USED TO STORE DATA
IDNO	–	I*4	STATION/TRIP LINK NUMBER
IFORM	–	I*4	NUMBER OF FORM SELECTED
ISIZE	–	I*4	SIZE OF FORM SELECTED

6.3.33.4 Description - Request processing is invoked by Output Processor Control for filing a data request in the request table. Requests are accumulated until a read command is encountered which causes initiation of the data acquisition process.

Request filing (PDL segment ZREQU) begins with creating an entry in the request table by initializing the following data associated with the request:

1. Assignment bin number (next available unused)

2. Initial bin size-assigned based on type of data display required. Initial bin size allocation is made to accommodate data acquisition and manipulation requirements. This allocation serves only as an initial size estimate of the bin area which may be expanded as required during data acquisition.

3. Main category of data (input mnemonic).

4. Subcategory of data (input mnemonic).

In addition, the required bin space allocation to accommodate the acquisition of data is performed (PDL segments ZBNCHK and ZSHIFT) and three other entries in the request table are initialized:

1. Next available position in the bin

2. Number of entries remaining in the bin

3. Request chain printer (=0).

Requests in the table are only erased after servicing (data acquisition, amnipulation, and display).

6.3.33.5 PDL - See Appendix A.

6.3.33.6 Decision Tables and Algorithms - None.

6.3.34 ZSHIFT

6.3.34.1 Identification

- o ZSHIFT - Reallocate Bin Storage Assignments
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.34.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
J	-	I*4	STARTING POSITION FOR BIN
L	-	I*4	CURRENT LOCATION OF BIN
K	-	I*4	END POSITION IN BIN TO BE MOVED

6.3.34.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
DIFF	-	L*4	INDICATES CURRENT POSITION & NEW POSITION ARE DIFFERENT
I1	-	I*4	L
I2	-	I*4	K-1

6.3.34.4 Description - ZSHIFT cycles through a given bin relocating contents in a new area and zero old bin entries.

6.3.34.5 PDL - See Appendix A.

6.3.34.6 Decision Tables and Algorithms - None.

6.3.35 ZSKIPFO

6.3.35.1 Identification

- o ZSKIPFO - Skip a Follower Record
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.35.2 Argument Dictionary - None.

6.3.35.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
FOLLOW	—	R*8	'FOLLOWER'

6.3.35.4 Description - ZSKIPFO reads next record from the Raw Statistics File. If expected follower is not found, it issues a warning message. If an I/O error is encountered, it issues a warning message.

6.3.35.5 PDL - See Appendix A.

6.3.35.6 Decision Tables and Algorithms - None.

6.3.36 ZSTORE

6.3.36.1 Identification

- o ZSTORE - Store Data in Bin
- o IBM/FSD - July 1, 1977
- o PARAFOR

6.3.36.2 Argument Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
L	-	I*4	REQUEST TABLE ENTRY ASSOCIATED WITH ITEM
ITEM	-	I*4	DATA VALUE TO BE STORED

6.3.36.3 Local Variable Dictionary

VARIABLE	DIM	TYPE	DESCRIPTION
XITEM	-	R*4	DATA ITEM TO BE STORED
YITEM	-	R*4	DATA ITEM TO BE STORED
TIME	-	R*4	TIME OF CURRENT RECORD BEING PROCESSED IN SEC
TUNITS	9	I*4	UNUSED
BIN	-	I*4	BIN TO BE USED TO STORE ITEMS
IREQ	-	I*4	REQUEST TABLE INDEX

6.3.36.4 Description - ZSTORE alters pointers into bin area from request table in order to reflect the storing of a sampled item. It ensures the bin receiving the data is large enough. If a statistical summary of this item is required, compute the sum of items and store the time of the minimum or maximum as required.

6.3.36.5 PDL - See Appendix A.

6.3.36.6 Decision Tables and Algorithms - None.

100 copies

☆U.S. GOVERNMENT PRINTING OFFICE : 1982—500-796/326

epartment
nsportation

arch and
cial Programs
ninistration

l Square
idge Massachusetts 02142

Postage and Fees Paid
Research and Special
Programs Administration
DOT 513

U.S. MAIL

Official Business
Penalty for Private Use $300