



**GENERAL PURPOSE SIMULATION
SYSTEM V (GPSS) V/6000
GENERAL INFORMATION MANUAL**

CDC® COMPUTER SYSTEMS:

CYBER 170 SERIES

CYBER 70 SERIES

6000 SERIES

LIST OF EFFECTIVE PAGES

New features, as well as changes, deletions, and additions to information in this manual, are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
Front Cover	-	6-9	B	C-1	D				
Title Page	-			C-2	D				
ii	D	7-1	D	C-3	D				
iii/iv	D	7-2	D						
v/vi	D	7-3	D	D-1	D				
vii	D	7-4	D	D-2	B				
viii	D	7-5	D	D-3	B				
		7-6	D	D-4	B				
1-1	-	7-7	D	D-5	B				
1-2	D	7-8	D	D-6	B				
1-3	D	7-9	D	D-7	B				
1-4	D	7-10	D	D-8	B				
		7-11	D						
2-1	D	7-12	D	E-1	D				
2-2	D	7-13	D	E-2	B				
2-3	D			E-3	B				
		8-1	D	E-4	B				
3-1	-	8-2	D	E-5	B				
3-2	-	8-3	D	E-6	B				
3-3	D	8-4	D	E-7	B				
3-4	D	8-5	D	E-8	B				
3-5	D	8-6	D	E-9	B				
3-6	D	8-7	D						
3-7	D			F-1	D				
3-8	D	A-1	D	F-2	B				
3-9	D	A-2	B	F-3	B				
3-10	D	A-3	-	F-4	B				
3-11	D	A-4	B	F-5	B				
3-12	D	A-5	B						
3-13	D	A-6	-	G-1	D				
3-14	D	A-7	B	G-2	B				
		A-8	D	G-3	B				
4-1	-	A-9	D	G-4	B				
4-2	B	A-10	-	G-5	B				
4-3	-	A-11	-	G-6	B				
4-4	D	A-12	-	G-7	B				
4-5	D	A-13	-	G-8	D				
4-6	D	A-14	-	G-9	D				
4-7	D	A-15	-	G-10	B				
		A-16	-	G-11	B				
5-1	D	A-17	-	G-12	B				
5-2	B	A-18	-	G-13	B				
5-3	-	A-19	-						
		A-20	-	H-1	D				
6-1	D	A-21	-						
6-2	-			Comment					
6-3	-	B-1	D	Sheet	D				
6-4	-	B-2	D						
6-5	-	B-3	D	Back					
6-6	B	B-4	D	Cover	-				
6-7	D	B-5	D						
6-8	D	B-6	D						



PREFACE

This manual is intended for use by the experienced General Purpose Simulation System (GPSS) V user as a supplement to the IBM General Purpose Simulation System V User's Manual (Pub. No. SH20-0851-1). This publication is a general information manual that presents detailed changes between Standard GPSS V and GPSS V/6000; there is no attempt made to explain the use of GPSS V/6000.

Appendix B of this manual describes differences between the two systems (GPSS V/6000 and the IBM Program Product 5734-XS2, GPSS V), and clarifies IBM implementation documentation ambiguities.

In general, any characteristic of GPSS V/6000 not covered in this manual can be assumed to conform to standard GPSS V conventions.

In various examples throughout the manual, column numbers are used to illustrate the GPSS V/6000 fixed format option. However, the user may also employ the free format option.

Additional information on GPSS in general can be found in the following manuals:

Simulation Using GPSS by Thomas J. Schriber
GPSS Primer by Stanley Greenberg

O

O

O

O

O

O

O

CONTENTS

1.	GENERAL SYSTEM DESCRIPTION	1-1		
	Transactions	1-1	REALLOCATE Card	4-1
	Blocks	1-1	JOB Card	4-2
	System Flow	1-3	CLEAR Card	4-3
	Core Allocation	1-3	RESET Card	4-3
			RMULT Card	4-4
			RN9	4-5
			SAVE Card	4-6
			READ Card	4-6
			REWIND Card	4-6
			SIMULATE Card	4-7
			START Card	4-7
2.	NOS SCOPE 3.4 - NOS/BE INTERFACE	2-1		
3.	GPSS ENTITIES	3-1		
	Standard Numerical Attributes	3-1	5.	UPDATE PROCEDURE
	Transactions	3-4		
	Blocks	3-5		Master File Creation
	PAUSE Block	3-5		Updating a Master File
	STOP Block	3-5		UPDATE Card
	GOTO Block	3-5		Update Operation Cards
	DISMISS Block	3-5		END UPDATE Card
	CALL Statement	3-6		Sample of Update Usage
	Modifications and Limitations			Old Master File
	to Standard Blocks	3-6		Update File
	EXECUTE Block	3-6		New Master File
	HELP Block from COMPASS	3-6		
	Variables	3-6	6.	PSEUDO-OPERATIONS
	Arithmetic Variables	3-7		
	Boolean Variables	3-8		END Card
	Functions	3-9		Block Number Assignment
	Function Definition Card Format	3-9		ICT Card
	Function Types	3-10		ORG Card
	Continuous Numerical			SYN Card
	Valued (C)	3-10		Entity Number Assignment
	Discrete Numerical Valued (D)	3-10		EQU Card
	List Numerical Valued (L)	3-10		Entity Function
	Discrete Attribute Valued (E)	3-10		Macros
	List Attribute Valued (M)	3-10		Listing Control Cards
	Function Follower Cards	3-11		COMMENT Card
	Tables	3-11		EJECT Card
	QTables	3-12		SPACE Card
	Storages	3-13		UNLIST Card
	Matrices	3-13		LIST Card
	Initial Cards	3-14		NOXREF Card
				XREF Card
				TITLE Card
4.	SIMULATION CONTROL	4-1		

7.	REPORT GENERATOR	7-1	GRAPH Card	7-9
			ORIGIN Card	7-10
	Selection of Statistics and Output		X Card	7-11
	Control	7-1	Y Card	7-12
	REPORT Card	7-1	STATEMENT Card	7-12
	ENDREPORT Card	7-3	ENDGRAPH Card	7-12
	TITLE Card	7-3	Sample Report Definitions	7-13
	INCLUDE Card	7-4		
	FORMAT Card	7-5	8. HELP BLOCKS	8-1
	TEXT Card	7-7	Procedure	8-1
	COMMENT Card	7-8	HELPA	8-2
	EJECT Card	7-8	HELPA	8-3
	SPACE Card	7-8	HELPC	8-4
	OUTPUT Card	7-8	ACCESS TO TABLES	8-6
	Graphic Output	7-8		

APPENDIXES

A.	ALLOCATION SCHEMES	A-1	E. BLOCK STATEMENT	
B.	GPSS V PROGRAM		FORMATS	E-1
	COMPATIBILITIES	B-1	F. CONTROL STATEMENT	
C.	JOBTAPE FORMAT	C-1	FORMATS	F-1
D.	REPORT GENERATOR		G. DIAGNOSTICS	G-1
	STATEMENTS	D-1		

TABLES

1-1	GPSS Entity Limits and Core	6-1	Entity Mnemonics	6-5
	Allocation	1-3	Entity Specification Mnemonics	7-3
2-1	GPSS Call Card Parameters	2-1	Standard Printout Column Numbers	7-4
3-1	Standard Numerical Attributes	3-1	Legal Format Card Mnemonics	7-6
3-2	Limits of Standard Numerical	7-4	Legal SNA Mnemonics for	
	Attributes	3-4	"Graph" Cards	7-9
4-1	Reallocate Card Entity Mnemonics	4-2		

GENERAL SYSTEM DESCRIPTION

1

The General Purpose Simulation System (GPSS) V/6000 is a general simulation package designed to simulate systems for analysis. The objects moving through the system are called transactions. The system is defined by means of blocks. By moving through the system (blocks), transactions obtain and release facilities, enter and leave queues, and obtain and use other resources. Statistics on the system are generated automatically, and additional statistics may be specified by the user.

GPSS V/6000 does not compile a program into object code for execution, but executes the program within itself.

TRANSACTIONS

Transactions are the entities which are processed by the system being simulated. Therefore, they can represent any number of actual entities, such as a customer of a business, units of traffic, or messages being processed. What the transactions represent is dependent on the system being simulated.

Transactions can possess up to 1020 parameters. These parameters accumulate information about the transaction as it passes through the system and direct the transaction's passage. Transactions can have fullword, halfword, byte or floating parameters with a maximum of 255 parameters of each type.

BLOCKS

Blocks are the entities used to represent the system being simulated. Action is initiated when a transaction enters a block. The action may be causing the transaction to enter or leave a queue, assigning a facility to the transaction, obtaining storage, or some other action, depending on the type of block entered. Some blocks are used to control the path a transaction follows when passing through the system or to delay a transaction until desired conditions are met.

There are also a number of entities which are not blocks but are defined in the same manner as blocks. These are definition cards, and are used to define functions, tables, matrices, and so forth.

The block and definition cards have a similar format*, which is:

2	8	19
name	operation	parameters

* The fixed format illustrated in this example, and throughout the manual is optional. GPSS V/6000 is free format.

GPSS V/6000 is a free field programming language.

Column 1 is used to designate a comment card (*).

Columns 1-5 are the location field which can contain a user symbol whose meaning is dependent on the type of card it appears on. The meaning of this field is discussed with each individual card type. A user symbol is 3-5 characters in length. The first 3 characters must be alphabetic and the last 2 alphanumeric. A number may also appear, and it too must start in column 1. For free field input the user symbol must start in column 1.

The operation field starts at least one blank after completion of the location field. If a location field is not used, the operation field may start in column two. The operation field determines the type of card used.

The auxillary field (such as on a gate block) may occur as the first column following the operation field, or it may be separated from it by one or more blanks.

The variable field is used for indicating information and options of the individual card. There are nine sub-fields labeled A-I. The meanings of these sub-fields are dependent on the type of card. The sub-fields start at least one blank after completion of the operation field (or auxillary field if it exists), and are separated by commas. After the first blank in the variable field, comments may be entered and they will be ignored by the system. If the card does not use any of the fields, comments may start in column 21. If a variable field is not required, comments starting before column 21 must be preceded by a (;).

For compatibility purposes, GPSS III/6000 input format decks may be used. (See PSEUDO OPS FIXED/FREE.) In this case the coding convention is as follows:

If column one contains an asterisk (*), the card is a comment card and is ignored by the system, although it is listed on the output.

Columns two through six define the location field. GPSS V/6000 automatically assigns a block number or entity name if this field is left blank. However, the programmer can assign symbolic names in this field, obviating the need for modification of block or entity references if blocks or entities are added or deleted at a later update. Symbolic names are three to five characters in length. The first three characters must be alphabetic; the last two, if used, alphanumeric.

Column seven is always blank.

Columns eight through 18 define the operation field. This field determines the type of block or entity. The operation code must be left-justified in the field. If column eight is blank, the card is assumed to be a comment card and is ignored by the system.

Columns 19 through 72 define the operand field. The number of operands is dependent on the type of block or entity defined. Operands must start in column 19 and are separated by commas. Embedded blanks are not allowed. The absence of an operand is signified by successive commas. Comments may be entered after the first blank in the field.

SYSTEM FLOW

Transactions are put into the system by GENERATE blocks. Transactions move sequentially through the blocks except where the path is modified by flow control blocks such as the TRANSFER and LOOP blocks. Simulation time is not incremented during transaction movement. During a scan, all transactions on the current events chain (all transactions able to move) are moved until blocked or until they encounter an ADVANCE block, which removes them from the current events chain until a specified positive (simulated) time has elapsed, at which point the transaction is placed back in the current events chain. If a transaction is blocked, it is put on a delay chain and is not examined again until the blocking condition has been removed, if possible. (i. e., there is no delay chain for the test block)

When no transaction can be moved, GPSS V/6000 advances the time increment to the next event scheduled to occur (first transaction on the future events chain). This is usually either the entry of one or more transactions from a GENERATE block, or a transaction leaving an ADVANCE block.

The scan then continues as before, until the number of transaction terminations (the number of transactions that have completed a path through the system) specified on the START block has been recorded. The simulation is then terminated, and statistics printed.

The ADVANCE block is the only block that inherently assigns a positive delay time to an entering transaction. A transaction may, however, be delayed at other blocks if that block will not allow entry. Otherwise, transactions move through the blocks with no incrementation of the simulated time.

CORE ALLOCATION

GPSS V/6000 will normally run in a field length of 32000₈ words. There are three standard allocation options. These are shown, with the core allocation per entity, in Table 1-1. The structure of each of the entities is given in Appendix A. The middle allocation scheme (R2) is the default allocation option. Note that there is no inherent limit on the number of transactions.

TABLE 1-1. GPSS ENTITY LIMITS AND CORE ALLOCATION

Entity	Maximum allocated			Core Allocation/Entity
	R1	R2	R3	
Blocks	80	500	1000	2 words + 1 word for every 2 parameters over 2 + 2 words for each matrix reference
Facilities	20	150	300	5 words
Functions	20	50	200	2 words + 1 word for each point (type L or M) or + 2 words per point (type C, D, or E)
Groups	5	10	25	1 word + 2 words for each transaction in the group or 1 word for each numeric value in the group.
Logic Switches	100	400	1000	1 word
Queues	35	150	300	4 words + 1 word for each current entry

TABLE 1-1. GPSS ENTITY LIMITS AND CORE ALLOCATION (CONT'D)

Macimum allocated				
Entity	R1	R2	R3	Core Allocation/Entity
Savevalues				
fullword	100	400	1000	1 word
halfword	50	200	500	1/2 word
byte	100	400	800	1/4 word
floating point	25	50	100	1 word
matrix fullword	5	10	25	1 word + 1 word/cell of the matrix
matrix halfword	5	10	25	1 word + 1/2 word/cell of the matrix
matrix byte	5	10	25	1 word + 1/4 word/cell of matrix
matrix floating point	5	10	25	1 word + 1 word/cell of matrix
Storages	20	150	300	7 words
Tables	15	30	100	9 words + 1 word for each frequency class + 1
Transactions	no limit			5 words + 1 word for each fullword or floating point parameter, + 1/2 word for each halfword + 1/4 word for each byte
User chains	20	40	100	3 words
Variables	20	50	200	1 word*
Boolean	5	10	25	1 word*

The allocation of individual entities can be changed by the REALLOCATE statement.

*Variables of all types are now compiled and the amount of storage needed to hold the compiled core is dependent on the type and complexity of this statement. Only approximate figures can be given (1 word per operation, 2 per SNA, 3 per indirect SNA). Constants are optimized out if possible. Multiple occurrences of floating point constants are stored only once.

NOS SCOPE 3.4-NOS/BE INTERFACE

2

The General Purpose Simulation System (GPSS) V/6000 is a standard software package running under the NOS, SCOPE 3.4, or NOS/BE operating system. The standard (default) GPSS V/6000 system can be called by the following control card:

1	8	19
GPSS.		

There are several options available which are specified by parameters on the control card. As many options as desired may be specified, but all the options must fit on one control card. Parameters may be in any order. The format of the control card is:

1	8	19
GPSS(P1, P2, P3, ...)		

The last character must be a period or closing parentheses, and parameters must be separated by commas. Embedded blanks are not allowed. If a parameter is not specified, it is assigned the default value. A list of the parameters, their default values, and their meanings is contained in Table 2-1.

TABLE 2-1. GPSS CALL CARD PARAMETERS

Parameter	Default Value	Meaning
A	none	Assembly phase only. SIMULATE card is ignored. No execution.
BB	unblocked	All jobtapes written will be in FORTRAN blocked binary format. Not available in CRM version of GPSS. CRM version writes with S type records.
C = lfn	COMPILE	File lfn contains the output of the update phase and is suitable for input to a GPSS run.
CL	none	Compressed list - no comment cards are listed, SPACE and EJECT cards are ignored in assembly phase.
EC	no ECS	ECS is used for all GPSS overlays.
FI = BBBBBB	1000 ₈	BBBBBB is minimum core obtained when more core is requested (expressed in octal).
I = lfn	INPUT	The record containing the program to be executed is on file lfn.
J1 = lfn	JOBTA1	lfn is used for JOBTAPE1.
J2 = lfn	JOBTA2	lfn is used for JOBTAPE2.

TABLE 2-1. GPSS CALL CARD PARAMETERS (CONT'D)

Parameter	Default Value	Meaning
J3 = lfn	JOBTAS	Lfn is used for JOBTAPE3
L = lfn	OUTPUT	Output from the system is written on file lfn.
LF = lfn	no HELP blocks	Lfn contains the reallocatable binary of all help blocks used. All routines on file lfn will be loaded in core before reading the input file. File lfn will be rewound unless file "INPUT" is used. The file will be read to an EOF or an empty record; single EOR's will be ignored.
L6	six lines per inch	Print listing at 6 lines per inch.
L8	six lines per inch	Output is formatted for 8 lines per inch; must be printed at a printer with 8 line-per-inch capability.
MF = BBBB	maximum field length available to the job	Maximum field length, in octal, that GPSS may request. If exceeded, run will terminate.
N = lfn	NEWPL	Lfn is the new program library (used with update).
NE	No ECS	GPSS will not use ECS for its overlays.
NB	none	System Bulletin will not be printed.
NX	none	Cross Reference map will not be printed.
P = lfn	OLDPL	Lfn is the old program library (used with update).
R	none	Perform simulation, even if no SIMULATE card.
RD = lfn	READ	Lfn is the file used for the read command.
RN = lfn	RN9 not available	File lfn contains one file of binary floating point numbers which will be used in order when RN9 is referenced.
SV = lfn	SAVE	Lfn is the file used for the save command.
FX	free field input	GPSS V input cards are to be processed with their fixed field format (GPSS III format).
FR	----	GPSS V input cards are to be processed with their free field format.
R1, R2, R3	R2	Entity allocation scheme choice.
U	none	Perform update phase only.
UB	unblocked	All jobtapes will be written in unblocked format, one logical record per transaction. Not available in CRM version of GPSS. CRM version writes with S type records.

Examples:

GPSS.

Input is from file INPUT.
Output is to file OUTPUT.
Free field format is allowed.
R2 allocation scheme is used.
Listing is 6 lines per inch.

GPSS,I=GPSIN,L=GPSOUT,FX,NB,NX,R3,LF=BIN,RN=RAND.

Input is from file GPSIN.
Output is to file GPSOUT.
Fixed field format is required.
System bulletin is not printed.
Cross-reference maps are not printed.
R3 allocation scheme is used.
FORTRAN binaries for HELP blocks are on file BIN.
RN9 may be used to access numbers on file RAND.

0

0

0

0

0

0

0

This section contains specific information concerning the various entities used by GPSS. While some attempt is made to demonstrate usage, none is made to demonstrate usefulness in actual simulation.

STANDARD NUMERICAL ATTRIBUTES

Standard Numerical Attributes (SNAs) are entities used to describe or reference information concerning other GPSS entities. They are used, for example, as the independent variable in specifying functions, as determining time increments in ADVANCE blocks, and numerous other areas. There are three ways of referencing SNAs: directly by name and number, directly by symbolic name, and indirectly.

Each SNA has a standard mnemonic associated with it. A list of these mnemonics, together with their explanation and the entities they reference, is contained in Table 3-1. To reference an SNA directly by name and number, the name followed by the number of the entity referred to are used. For example,

FN9

references the value of function 9.

To reference an SNA directly by name, the mnemonic followed by the symbolic name assigned the entity is used. For example, if MEM is the name of a storage, \$\$MEM references the current contents of that storage. Note that the mnemonic and symbolic name must be separated by a \$.

To reference an SNA indirectly is done by using the mnemonic and modifying it by a parameter number, rather than an entity number. The value of the parameter specified is the entity number which will be referenced. In order to differentiate between parameter and entity numbers, the parameter numbers in indirect reference must be separated from the mnemonic by an asterisk. For example, if parameter 3 contains the value 8,

X * 3

causes parameter 3 of the transaction being processed to be searched, and the value of this reference determines which savevalue will be referenced; in this case, 8.

A generalized indirect has been implemented. Any SNA may be referenced indirectly by any other SNA.

Example

XF * XB\$ALPHA fullword savevalue specified by byte
savevalue alpha

Not all SNAs have the same permissible range of values. The range permitted each SNA is given in Table 3-2.

TABLE 3-1. STANDARD NUMERICAL ATTRIBUTES

Entity	Mnemonic	Meaning
Transactions	PF	Fullword parameter
	PH	Halfword parameter
	PB	Byte parameter
	PL	Floating point parameter
	PR	Priority
	M1	Transit time
	MP	Parameter Transit time, suffix of PF, PH or PB required.
Blocks	N	Total entry count
	W	Current entry count
Facilities	F	Status of facility
	FR	Facility utilization (parts/1000)
	FC	Entry count
	FT	Average time per transaction
Storages	S	Current Contents
	R	Remaining contents
	SR	Utilization (parts/1000)
	SA	Average contents
	SM	Maximum contents
	SC	Entry count
	ST	Average time/transaction
Queues	Q	Current length of queue
	QA	Average contents
	QM	Maximum contents
	QC	Total entry count
	QT	Average time per transaction (including zero delays)
	QX	Average time per transaction (without zero delays)
	QZ	Number of zero entries
Tables	TB	Table mean
	TC	Entry count
	TD	Standard deviation

TABLE 3-1. STANDARD NUMERICAL ATTRIBUTES (CONT'D)

Entity	Mnemonic	Meaning
Savevalues	X or XF	Fullword savevalues
	H or XH	Halfword savevalues
	XB	Byte savevalues
	XL	Floating point savevalues
	MX	Matrix fullword savevalues
	MH	Matrix halfword savevalues
	MB	Matrix byte savevalues
	ML	Matrix floating point savevalues
Groups	G	Number of items in group
User chains	CA	Average number on chain
	CH	Current number on chain
	CM	Maximum number on chain
	CC	Total entries
	CT	Average time per entry
Functions	FN	Function value
Variables	V	Variable
	BV	Boolean variable
Random numbers	RN	Random number generator
Clock	C1	Current relative clock
	AC1	Current absolute clock
	TG1	Terminations to go (see start)
Constant	K	Positive integer $n < 2^{17}$

TABLE 3-2. LIMITS OF STANDARD NUMERICAL ATTRIBUTES

Limits*	Applicable SNAs
0-1	BV, F, L
0-127	PR
0-999	FR, RN**, SR
0-(2**12-1)	CA, CH, CM, G, W
0-(2**29-1)	AC1, CC, CT, C1, FC, FT, MP, M1, N, Q, QA, QC, QM, QT, QX, QZ, R, S, SA, SC, SM, ST, TC, TG1
0-(2**58-1)	TD
±(2**18-1)	MB, PB, XB
±(2**28-1)	H, MH, PH, XH
±(2**58-1)	FN***, MX, PF, TB, V, X, XF
±(10 ⁻³¹⁰ - 10 ³¹⁰)	ML, PL, XL - only where F. P. value allowed

TRANSACTIONS

Transactions are the entities which cause action by the simulated system. There is no limit on the number of entities that may be present in the system at one time, other than limitations of memory space. Transactions are created from available memory as needed, and destroyed when terminated. Hence, the memory used for a terminated transaction is available to the GPSS program.

Information concerning transactions is carried in the parameters. Each transaction is assigned a certain number of parameters by the program upon creation. This number is determined by the programmer. There are four types of parameters: halfword, fullword, byte and floating point. If not otherwise specified, transactions are assigned 12 halfword parameters.

Each transaction requires five words of memory, plus one word for each fullword or floating point parameter (or one-half word for each halfword parameter or 1/4 word for each byte). The usage of storage for transactions is given in Appendix A.

Available transactions are placed in a pushdown list and when additional transactions are needed, they are removed from the top of the stack and additional memory is requested only when the stack is empty. The transaction number is used to reference a transaction and has no other meaning. The number is a function of the transaction's location in core.

*The values of CA, CT, FT, QA, QT, QX, SA, ST, and TD are truncated to integers except in floating point variables or field requiring a floating point value.

**F. P. number in range (0, 1) if in field A of a function definition card.

***F. P. value retained if used as a function modifier.

BLOCKS

In general, blocks are defined and operate as in standard GPSS V; however, there are some changes. Four new blocks have been added: PAUSE, STOP, GOTO, and DISMISS. In addition, there are some modifications and limitations concerning some other blocks.

PAUSE BLOCK

Starting with the first non-blank after PAUSE, 10 characters of this card are placed in the dayfile and the simulation pauses until the operator intervenes. Exit is to the next block. If fixed format is observed it is as follows:

2	8	19
	PAUSE	message

STOP BLOCK

This block causes simulation to halt permanently. There are two options available in the use of the STOP block: A field blank and A field nonblank.

If the A field is blank, the termination count is set to zero, and the simulation stops. Since this is a normal exit, standard termination output is produced.

If the A field is nonblank, columns 19 through 28 are placed in the dayfile, and simulation ends with a fatal error. The format of the STOP block is:

2	8	19
	STOP	message

GOTO BLOCK

This block is a replacement for the unconditional transfer block. The only difference between GOTO ALPHA and TRANSFER ALPHA is that the GOTO block takes less core and executes faster.

DISMISS BLOCK

This block returns and/or releases any facility currently preempted and/or seized by the current transaction.

CALL STATEMENT

CALLA, CALLB, and CALLC are like the HELP blocks except that they are statements and not blocks. When they are encountered in the input stream the specified subroutine will be called before reading the next card. They can be used to call initialization subroutines prior to execution of the GPSS run or termination subroutines after GPSS execution. Since there is no transaction associated with the CALL statement, no SNA may be used which references a parameter.

MODIFICATIONS AND LIMITATIONS TO STANDARD BLOCKS

There are some changes to a few of the standard GPSS V blocks. While these are not major, they are important; without an understanding of how these blocks operate, the simulation could well be invalid.

EXECUTE BLOCK

The only change to the EXECUTE block is the elimination of recursive capability; that is, an EXECUTE block cannot execute another EXECUTE block.

HELP BLOCK FROM COMPASS

When a COMPASS call from a HELP block occurs, GPSS V/6000 executes an RJ jump to the called routine. The registers set are:

B1 = 1

A1 = Location of FTN parameter array

X1 = Location of first parameter

A1 and X1 must be set to these values, and B1 must be set to one, whenever a user calls a GPSS V/6000 subroutine. All GPSS V/6000 routines are available to the user. See internal documentation of system routines prior to use to determine how to use them. In general, only the following registers are used by the SNA routines: A1, A2, A6, A7, X1, X2, X6, X7, B7 and the integer result is in X6 and if a floating point result was calculated by the routine, it is in X7; otherwise X7 is zero. Normal return of a HELP block is through its entry point (HELP blocks are called by an RJ instruction).

A normal return from the help routine restores A0, B1, B4, and B5. The transaction exits to the next sequential block.

VARIABLES

Variables are used to evaluate arithmetic or logical expressions involving Standard Numerical Attributes. There are three types of variables, composed of two classes: The first class, arithmetic variables, has two variable types - integer and floating-point. The second class is composed of the boolean (logical) variables.

ARITHMETIC VARIABLES

The integer and floating point variables are defined in the same manner, with a slight modification of the operation field. The format for the integer variable definition is:

2	8	19
name	VARIABLE	exp

The format for floating-point variable definition is:

2	8	19
name	FVARIABLE	exp
name	symbolic name or number of the variable	
exp	expression to be evaluated, in both cases.	

There are five arithmetic operations defined in GPSS: Addition(+), subtraction(-), multiplication(*), division(/), and modulo division(// or @). Addition and subtraction have the same precedence. Multiplication, division, and modulo division have the same precedence in respect to each other, and all three have precedence addition and subtraction. With a precedence level, evaluation is left to right.

Multiplication can also be implied by the use of parentheses in the normal mathematical manner. This is made possible by the lack of subscripting in GPSS.

The nesting of expressions by use of parentheses can be carried to any level. However, there is a practical limit, since the entire expression must fit on a single card. To use longer expressions, more variable definitions must be used to evaluate subexpressions, and these variables used in place of the subexpressions. If, however, variables reference each other (that is, are cyclic in definition), execution errors will result. Most time in evaluating a VARIABLE is in the initialization of the recursive sections. To increase the speed, make as few references to other variables as possible.

If evaluation of an expression results in an indefinite or out-of-range value, the value is set to zero. Similarly, division by zero is defined, and yields a quotient of zero.

Both floating-point and integer variables return integer results. The difference lies in the method of evaluation. Integer variables are truncated to integers both before and after each operation in evaluating the expression. Floating-point variables are not truncated until the expression has been completely evaluated. As an example, the integer variable

2	8	19
1	VARIABLE	15-(18/4)*3

gives a result of 3, while the floating-point variable

2	8	19
2	FVARIABLE	15-(18/4)*3

gives a result of 1.

Integer and floating-point variables are referenced in precisely the same manner; the letter V followed by \$name or by number. Because of this single form of reference, each arithmetic variable, whether integer or floating-point, must have a unique name or number.

BOOLEAN VARIABLES

Boolean variable definition cards allow the user to evaluate logical combinations of SNAs, logical operators, conditional operators, and boolean operators. The result of a boolean (logical) expression is either true (=1) or false (=0). The SNA reference for Boolean variables is BV followed by the number or \$name of the variable. The format for variable definition is:

2	8	19
name	BVARIABLE	exp
name	name or number of the variable logical	
exp	expression to be evaluated	

There are three Boolean operators used in GPSS: the inclusive or (+), the exclusive or (-), and the logical and (*). The conditional operators are those used to compare numeric magnitude of SNAs. They are:

'G'	Greater than
'E'	Equal
'L'	Less than
'NE'	Not equal
'LE'	Less than or equal
'GE'	Greater than or equal

Logical operators are associated with equipment entities and are used to describe the status of these entities. They are either true (=1) or false (=0), depending on whether the condition for which they test is true or false. A list of the conditional operators and the conditions to which they refer is given below.

<u>operator</u>	<u>will be true if</u>
FUj	Facility j is seized or pre-empted
FNUj	Facility j is neither seized nor pre-empted
FIj	Facility j is pre-empted
FNIj	Facility j is not pre-empted
FVj	Facility j is available
FNVj	Facility j is not available
SFj	Storage j is full
SNFj	Storage j is not full
SEj	Storage j is empty
SNEj	Storage j is not empty
SVj	Storage j is available
SNVj	Storage j is not available
LRj	Logic switch j is reset
LSj	Logic switch j is set

For example, the Boolean variable

2	8	19
IOTA	BVARIABLE	(V2'G'5)*(FNI2 + LR\$LIGHT)

has the value true (=1) if facility 2 is not pre-empted or logic switch LIGHT is reset and if at the same time variable 2 is greater than 5.

Variable expressions terminate at the first blank.

FUNCTIONS

Function definition cards are used to describe the one-to-one correspondence between an independent SNA and a dependent function value. The numerical description of this correspondence is given by one or more function follower cards.

One of the most common uses of functions is in the generation of non-uniform random variates. This is accomplished by using one of the uniform random number generators as the independent variable and defining the desired cumulative density function in the function follower cards. Since it is being referenced by a function, the random number generator returns a value in the interval 0-1 instead of an integer on 0-999.

Functions are truncated to integer values after they are evaluated except when they are referenced in field B of GENERATE or ADVANCE blocks or field C of ASSIGN blocks.

FUNCTION DEFINITION CARD FORMAT

2	8	19
name	FUNCTION	indvar, type

name	function name or number
indvar	independent variable SNA
type	function type and number of points

Examples

2	8	19
1	FUNCTION	RN2, C3

Function number 1 is defined as a continuous numerical valued function of 3 points with random number generator 2 as its independent variable. References to the SNA FN1 cause the function to be evaluated.

2	8	19
ALPHA	FUNCTION	X\$VAL01, L4

Alpha is a 4 point list numerical valued function with savevalue VAL01 as the independent variable. The function is referenced by its SNA, FN\$ALPHA.

FUNCTION TYPES

CONTINUOUS NUMERICAL VALUED (C)

The function follower cards contain X, Y points defining the function in a piece-wise linear fashion.

When the value of the independent variable SNA lies between two consecutive X values on the function follower card, the value of the function is obtained by linear interpolation.

If the value of the independent variable SNA is outside the defined range of the function, the function takes the value of the first or last Y value (whichever is applicable).

DISCRETE NUMERICAL VALUED (D)

The X, Y points on the function follower cards define a discrete valued step function.

When the value of the independent variable SNA lies between two consecutive X values (say $X(I-1)$ and $X(I)$), the function is assigned the value of $Y(I)$. No linear interpolation is performed.

If the independent variable lies outside the defined range of the function, the function takes the value of the first or last Y point (whichever is applicable).

LIST NUMERICAL VALUED (L)

Only Y values must be specified on the function follower cards. X values, if entered, are ignored.

It is assumed that the independent variable takes on the integer values $1, 2, 3, \dots, n$ where n is the number of points in the function definition.

Values of the independent variable that are outside the range 1 to N will cause execution to terminate with an error.

List functions can be evaluated much faster than the C and D type functions.

DISCRETE ATTRIBUTE VALUED (E)

This type of function is analogous to the discrete numerical type except that the Y values on the function follower cards are SNA references.

LIST ATTRIBUTE VALUED (M)

Analogous to the L type function except that Y values are SNA references.

FUNCTION FOLLOWER CARDS

Format and Rules

Function follower cards contain the X, Y coordinates defining the function in the order of increasing X value.

The first 72 columns are available for use.

Any number of points may appear on a single card but the X and Y coordinates of a point may not be separated over 2 cards. Any number of cards may be used.

The total number of points defined on all the function follower cards for a single function must be equal to the field B value on the function definition card.

The limits on point definition are 2047 for C, D and E functions and 4095 for L and M functions.

The X and Y coordinates of a point are separated by commas and X, Y points are separated from each other by slashes. The last slash on any card is optional.

The data on the function follower cards must begin in column 1. A blank terminates the card (this is not compatible with GPSS III).

Comments may not appear between a function definition card and its follower cards. No comments are allowed either on or between function follower cards.

Examples

1	8	19
1	FUNCTION	RN2, C3
0, 0/.75, 100/.9999, 1000		
1	8	19
ALPHA	FUNCTION	X\$VAL01, L4
,16/,47/,39/,22		
1	8	19
FAC	FUNCTION	FN\$QUAN, E4
7, FRPUMP1/15, FR\$PUMP2		
23, FR\$PUMP3		
100, FR\$PREM		

Note the use of another function as the independent variable.

TABLES

Statistics concerning transaction movement are often desired in simulations. For gathering frequency distribution information, GPSS provides the user with TABLE cards for table definition, and TABULATE cards for causing table entries. The format of the table definition card is:

2	8	19
name	TABLE	type, upper, int, num, rt

name name or number of the table
 type SNA or table type specification
 upper upper limit of the first interval
 int interval width
 num number of intervals
 rt arrival rate time interval - only used by arrival rate tables

There are four types of tables: standard, difference, arrival rate, and interarrival rate.

Standard tables have an SNA in field A (type), and are used to gather statistics about that SNA. When a transaction enters a TABULATE block calling a standard table, the value of the SNA is obtained, and the proper frequency count in the table incremented by one or the weighting factor specified on the TABULATE card.

Difference tables are similar to standard tables, but the SNA in field A has a minus sign (-) appended. The quantity tabulated is the difference between the current SNA value and the previously tabulated SNA value.

Arrival rate tables tabulate the number of arrivals at the TABULATE block every n time units. Field A contains the letters RT, and field E gives the value of n.

Interarrival rate tables tabulate the time between arrivals at the tabulate block. The characters IA are put in field A of the table definition card.

If a table has a weighting factor specified on one of its TABULATE blocks, the number of intervals (field D) must have an alphabetic character as a prefix. As an example, the card

2	8	19
ALPHA	TABLE	FR\$UNIT, 100, 100, A10

defines a table which tabulates the facility utilization of facility UNIT. The upper limit of the first interval is 100, each interval is 100 units wide, there are ten units, and at least one TABULATE block referencing this table specifies a weighting factor. If ALPHA were not a weighted table, the table definition card would be

2	8	19
ALPHA	TABLE	FR\$UNIT, 100, 100, 10

QTABLES

The QTABLE card is provided to obtain the distribution of queue delay time for a particular queue. When a transaction enters a DEPART block of a queue for which a QTABLE has been defined, the delay time for the transaction in the queue is tabulated. Zero delay times are also tabulated. Since the QTABLE uses a table reference, the names or numbers of QTABLEs must not duplicate those of other tables. The format of the QTABLE definition card is:

2	name	8	QTABLE	19	nam1, upper, int, num
---	------	---	--------	----	-----------------------

name name or number of the table
 nam1 name or number of the associated queue
 upper }
 int } same meaning as for other tables
 num }

STORAGES

STORAGE cards are used to specify the capacities of storages in the user program. If no STORAGE card is entered for a particular storage, the default value is $(2^{**}29)-1$. If two or more capacities are defined for the same storage, the last capacity encountered will be used.

There are two formats for the STORAGE card: single storage and multiple storage. The format for single storage is:

2	name	8	STORAGE	19	capacity
---	------	---	---------	----	----------

name name or number of the storage
 capacity the capacity of the storage

If desired, the capacities of several storages can be defined on a single card by using the multiple storage definition card. When using this card, the first blank in the operand field capacities terminates the card. As an example, the card

2		8	STORAGE	19	S5, 10/S17-S25, 30/S\$ABC1-S\$ABC5, 25
---	--	---	---------	----	----------------------------------------

defines the capacity of storage 5 as being 10, storages 17 to 25 as being 30, and storages ABC1 to ABC5 as being 25.

MATRICES

The matrix definition card is used to describe the dimensions of a matrix of savevalues which are to be referenced by their row and column in the matrix. The format for the matrix definition card is:

2	name	8	MATRIX	19	p, rows, cols
---	------	---	--------	----	---------------

name name or number of the matrix
 p MX of X for fullword, MH or H for halfword, MB for byte, and ML for floating point
 rows number of rows in the matrix
 cols number of columns.

For example, the card

2	8	19
ALPHA	MATRIX	H, 3, 4

defines a matrix named ALPHA of halfword savevalues. Twelve savevalues are used in a three row by four column matrix.

INITIAL CARDS

INITIAL cards are used to preset the values of savevalues and matrix savevalues, and set the values of logic switches. If not initialized by means of INITIAL cards, all savevalues have an initial value of zero, and all logic switches are reset. The format of the INITIAL card is:

2	8	19
	INITIAL	values

values a list of savevalues, matrix savevalues, and logic switches, together with their initial values.

As an example, the card

2	8	19
	INITIAL	X1, 4/XH1-XH17, 9/MX4(6, 9), 7

sets the initial value of fullword savevalue 1 to be equal to four, the value of halfword savevalues 1 through 17 to be equal to nine, and cell 6, 9 of fullword matrix 4 to be equal to seven. The card

2	8	19
	INITIAL	LS\$ALPHA/X\$RHO, 17

sets logic switch ALPHA and initializes the value of fullword savevalue RHO to 17.

As with the multiple storage definition card, the first blank in the operand field terminates the card.

Matrices must be defined by a MATRIX card before initialization by an INITIAL card.

There are a number of cards used by GPSS to control a simulation. Most of them are used when more than a simple execution of an input deck is desired. They can be used to set up a simulation, save a simulation model, read back a previously saved model, or to change the allocation of the various entities.

REALLOCATE CARD

This card is used to specify limitations on the number of entities. The card is not required, as there are three standard allocation schemes available. If, however, none of these three adequately fit a particular model, the number allocated can be increased or decreased by means of the REALLOCATE card.

The card format is:

2	8	19	
	REALLOCATE	list	

list list of entity mnemonics and their new allocations.

The form of the list is

entity, number, entity, number, . . . , entity, number

As many REALLOCATE cards as needed can be used. A list of legal entity mnemonics is given in Table 4-1.

When a REALLOCATE card is encountered, all previous model definitions are erased, and all macro and report generator routines are destroyed.

As a sample of the use of the REALLOCATE card, the cards

2	8	19	
	REALLOCATE	BLO, 200, FAC, 10, LOG, 10, FSV, 500	
	REALLOCATE	GRP, 20, BVR, 20, STO, 50	

would cause the model to allocate room for 200 blocks, 10 facilities, 10 logic switches, 500 fullword savevalues, 20 groups, 20 boolean variables, and 50 storages. The allocation of other entities than those specified on a REALLOCATE card are unchanged from the previous allocation.

TABLE 4-1. REALLOCATE CARD ENTITY MNEMONICS

Entity	Mnemonic
Block	BLO
Boolean variable	BVR
Facility	FAC
Function	FUN
Group	GRP
Logic switch	LOG
Matrix savevalue - fullword	FMS
Matrix savevalue - halfword	HMS
Matrix savevalue - byte	BMS
Matrix savevalue - floating point	LMS
Queue	QUE
Savevalue - fullword	FSV
Savevalue - halfword	HSV
Savevalue - byte	BSV
Savevalue - floating point	LSV
Storage	STO
Table	TAB
User chain	CHA
Variable	VAR

The mnemonics XAC (transactions), MAC (macros), and COM (common storage) are not meaningful for GPSS V/6000, and are ignored if encountered.

JOB CARD

The JOB card is used to completely erase any previous model definitions in the system. The purpose of this card is to enable the running of unrelated simulations in a single run.

As with the REALLOCATE card, all macro definitions and report generator routines are destroyed. In addition, all random number generators are reset to their initial values, and the allocation scheme is reset to the type specified on the control card (or R2 by default).

The format of the JOB card is:

2		8		19	
		JOB		title	

title title of the next run. The title must be less than or equal to 49 characters in length.

CLEAR CARD

The CLEAR card is used to generate another run, possibly with modifications, using the same model. A CLEAR card before the first START card is not meaningful, since the system is initially cleared.

The format of the CLEAR card is:

2		8		19
		CLEAR		list

list a list of those savevalues which are not to be affected by the clearing operation. If the list is omitted, all savevalues are set to zero; otherwise, the savevalues specified will not be reset.

When a CLEAR card is encountered in the input stream, the following actions take place:

- The absolute and relative clocks are set to zero
- All current and incipient transactions are destroyed
- Current and total block counts are set to zero
- All statistics pertaining to chains, facilities, storages, queues, and tables are reset to zero
- All logic switches are reset
- Savevalues not specified on the CLEAR card are set to zero
- Matrices not listed on the CLEAR card are reset to zero. The matrices listed are not disturbed.
- Random number generators are not reset

As an example, the card

2		8		19
		CLEAR		X1-X10, X17, XH5-XH9, XH\$ALPHA-XH\$THETA

clears the simulation model except for fullword savevalues 1 to 10 and 17 and for halfword savevalues 5 to 9 and ALPHA to THETA. Note that fullword and halfword savevalues cannot be intermixed, and that the savevalues must be specified in ascending numerical order, including named savevalues and matrices as well.

Two or more CLEAR cards in a row are treated as a single card, allowing the specification of a number of savevalues to be saved. However, the rules concerning mixing types and numerical order apply across multiple card clearing operations.

After clearing operations are completed, any RT tables have their transaction operators recreated. Also, a new transaction is created at each GENERATE block. Since the number of previously generated transactions has been set to zero, GENERATE blocks with a creation limit generate a new set of transactions, regardless of whether the limit was or was not reached previously.

RESET CARD

A RESET card is used when another run of the same model is desired with some entities other than savevalues not being reset. These entities are specified on the RESET card. As many cards as necessary can be used, and are treated as a single card.

The RESET card affects all cards entered after the previous START card. Since it is executed immediately upon being encountered, it cannot affect cards following it in the input deck.

The format of the RESET card is:

2	8	19
	RESET	list

list the list of entities which are not to be reset.

When a RESET card is encountered, the following actions are taken:

- The relative clock is reset to zero
- Total block counts are set to zero, but current counts are unaffected
- Statistics accumulated in tables are set to zero
- Cumulative statistics for chains, facilities, storages, and queues not specified on the RESET card are set to zero. The total entry counts and maximum contents for the reset entities are set to the current contents.

Unlike the CLEAR card, transactions, savevalues and logic switches are not affected. Random number generators and the absolute clock are not reset.

As in the CLEAR card, entities specified on the RESET card must be given in increasing numerical order within entity type, and entity types cannot be mixed. An example of RESET card usage is:

2	8	19
	RESET	F5-F15, F\$ALPHA, Q3, Q\$AAA-Q\$CCC
	RESET	S2-S6, S\$PARK1-S\$PARK5
	RESET	S\$PAR19-S\$PAR27, CH3, CH6
	RESET	TB\$WAITT

Note that, due to the restriction to increasing numerical order, facility ALPHA must be number 16 or more, and similarly, storages PARK1 to PARK5 must number less than PAR19 and more than 6.

RMULT CARD

The RMULT card is used to replace the multipliers of the random number generators. The format is

2	8	19
	RMULT	r1, r2, r3, r4, r5, r6, r7, r8

where r1, ..., r8 are to replace the current seeds. Extreme care should be taken in choosing these numbers to ensure good random properties.

If one of the generators is not to have its seed reset, its position is indicated by consecutive commas. A 37 in any of the fields restores the original value of the seed for that generator. Up to 14 digits may be used for each seed.

As an example, the card

```
|| 2 | 8 | 19 |-----|
||   | RMULT | ,123579, , , , 37
```

sets the seed for generator 2 to 123579, restores the original seed value for generator 6, and leaves all others unaffected. Note that trailing unaffected generators may be omitted from the list. All eight random number generators initially produce the same sequence of random numbers.

RN9

The user also has the ability to read random numbers from a file that he creates. These values are referenced by RN9. The control card parameter, RN=fn, must be specified or RN9 will be considered an illegal SNA.

The file should be created by a FORTRAN program using unformatted WRITE statements. The file should contain numbers between 0 and 1, not including the endpoints. Use BT=C, RT=F, FL=10 on a file card before creation.

EORs will be ignored, but an EOF will cause the file to be rewound and the sequence repeated. The CLEAR and RESET cards will not affect the position of the file, but the JOB card will rewind the file.

Example

```
PROGRAM RND (RAND, TAPE6=RAND)
DO 10 I=1, 100
X=I * .003
WRITE (6) X
10 CONTINUE
STOP
END
```

The control cards to run this program are:

```
FTN.
FILE, RAND, BT=C, RT=F, FL=10.
LGO
```

To run this program to create the file and then use it in a GPSS model, use the following control cards:

```
.
.
.
FTN.
FILE, RAND, BT=C, RT=F, FL=10.
LGO.
REWIND, RAND.
GPSS, FX, RN=RAND.
.
.
.
```

*See Appendix H.

SAVE CARD

The SAVE card is used to save the current model on the save file. The save file is determined by the user on the GPSS call card. The format of the SAVE card is:

2		8		19
		SAVE		c

where c is any character, or blank. If c is omitted (blank), the save file is rewound before the current model is saved. If c is non-blank, the model is written as the next file on the save file.

READ CARD

The READ card is used to restore a model which has been saved on the read file (see SAVE card). The format of this card is:

2		8		19
		READ		num

where num indicates the number of files to skip on the restart file before reading in the desired saved model. Hence, if the model desired were the fifth on the restart file, the card would be

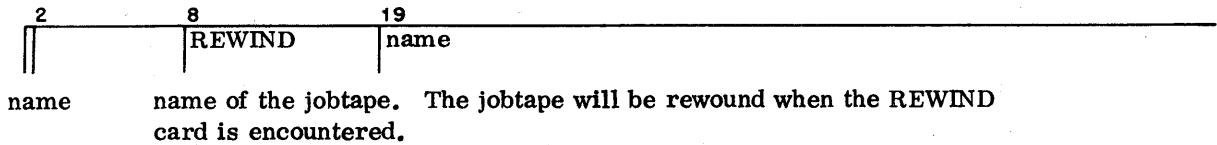
2		8		19
		READ		4

When the saved model is read in, all model definitions and blocks entered prior to the READ card are lost. Also, the allocation of entities will be that allocation used when the model was saved. Since any previous models are lost, any cards following a READ card can refer to the restored model only.

When a model is read in by the READ card, all FORTRAN binaries specified by the LF parameter on the control card for that run will also be lost. This occurs because the FORTRAN binaries are loaded into core before the READ statement is executed. Then, by executing the READ statement, all models, blocks, and definitions are destroyed. This loss may be avoided by using the LF parameter on the control card of the model being saved. Even if a HELP block is not used in the model the binaries will still be loaded into core. Then when the model is saved by the SAVE card, the FORTRAN binaries will be on the saved file, and can be used by the model which reads that file in.

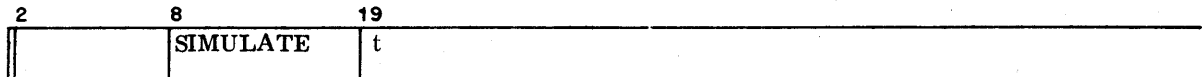
REWIND CARD

This card is used to rewind a jobtape. The format is:



SIMULATE CARD

The function of the SIMULATE card is to cause the execution of a GPSS model. If no SIMULATE card is present, execution does not occur, unless overridden by the GPSS program call card (see Section 2). The format is:

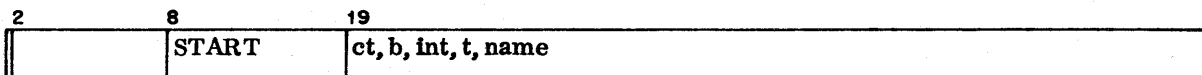


t = number of CP minutes to be used as a time limit on the model*

START CARD

The START card is used to indicate that a model is completely read in, and execution of the model should begin. The cards following a START card can be new or redefined blocks, simulation control cards, or other START cards. Execution resumes on the model, whether new, modified, or unchanged, when the next START card is encountered.

The format of the START card is:



ct	termination count for the model
b	printout indicator (if b = NP, end-of-run printout is suppressed)
int	snap interval (the number of terminations between printouts for intermediate statistics)
t	transaction printout indicator (t = 1 causes the transaction printout to be produced whenever a standard printout is obtained)
name	name of the report to be used (see Section 7)

The START card controls the length of model execution by means of the termination count. Transactions entering TERMINATE blocks in the model cause the termination count to be decremented appropriately. When the termination count becomes zero, execution is halted, and end-of-run statistics are printed, unless these statistics have been suppressed by an NP in field B of the START card.

*See Appendix H

00

0

0

0

0

0

UPDATE PROCEDURE

5

An update capability is provided to eliminate the need for handling large GPSS V/6000 source decks that undergo modification. This feature is provided for compatibility with standard GPSS V. UPDATE or MODIFY are better alternatives.

The update feature uses a master file and an update file to create a new master file. The new master file may be executed and punched without affecting the previous master. Also provided in the update capability is the ability to create a master file.

NOTE: Fixed field coding is required for an update.

MASTER FILE CREATION

A master file is created by the insertion of a CREATE card as the first card of the input deck. The user can specify the name to be given the master file on the GPSS V/6000 control card (see Section 2). If no name is specified, the file is given the name NEWPL. The new master file will not be rewound either before or after creation. The format of the CREATE card is:

```
||-----|-----|-----|
 2         8         19
||-----|-----|-----|
||         |CREATE  |         |
```

UPDATING A MASTER FILE

A master file is updated by means of an update file. An update file begins with an UPDATE card and ends with an ENDUPDATE card. There are three types of update operation cards that can be used in an update file.

UPDATE CARD

The UPDATE card must be the first card in an update file. The format of the card is:

```
||-----|-----|-----|
 2         8         19
||-----|-----|-----|
||         |UPDATE  |options|
```

There are two options available (PUNCH and NOASSEMBLE). Either or both options may be used, if desired. Specification of an option is not, however, necessary. PUNCH causes the new master file to be punched and written to the compile file (see control card option C=lfm in Section 2). NOASSEMBLE causes the new master file to be listed, but not executed. If neither is specified, the new master file is assembled and executed, but no other action is taken. The options, if specified, are order-independent.

UPDATE OPERATION CARDS

There are three cards that are used to effect update action. The format of these cards is:

2	8	19
	ADD	A
	DELETE	A
	DELETE	A - B
	REPLACE	A

The cards following the ADD operation card, up to the next update operation card, are inserted into the master file after line A.

The DELETE card has two forms. The first eliminates a single card from the master file. This is the DELETE A form. The second form deletes all cards from A to B, inclusive. Hence, the cards must be in increasing numerical order.

The REPLACE card functions as a combination of the single card DELETE card and the ADD card. Card A is deleted from the master file and the cards following the REPLACE card, up to the next update operation card, are inserted.

Adding cards after a card that has been deleted is not permissible.

ENDUPDATE CARD

An update file must end with an ENDUPDATE card. The format is:

2	8	19
	ENDUPDATE	

When the ENDUPDATE card is encountered, file update ends, and the options specified on the UPDATE card are implemented.

SAMPLE OF UPDATE USAGE

Here we will update a simple master file and show the resulting file. Note that on the update file no options have been specified on the UPDATE card. Therefore, the new master file is assembled and executed following the update.

OLD MASTER FILE

2	8	19
	SIMULATE	
	GENERATE	20, 5
	TRACE	
	SEIZE	ALPHA
	ADVANCE	10, 5
	RELEASE	ALPHA
	UNTRACE	
	TERMINATE	1
	START	100
	RESET	
	START	1000
	END	

UPDATE FILE

2	8	19
	UPDATE	
	DELETE	3
	REPLACE	5
	ADVANCE	20, 10
	DELETE	7
	ADD	11
	RESET	
	START	1000
	ENDUPDATE	

NEW MASTER FILE

2	8	19
	SIMULATE	
	GENERATE	20, 5
	SEIZE	ALPHA
	ADVANCE	20, 10
	RELEASE	ALPHA
	TERMINATE	1
	START	100
	RESET	
	START	1000
	RESET	
	START	1000
	END	

0

0

0

0

0

0

0

The operations described here are called pseudo-operations because they are directed to the assembler, rather than the system being simulated. Hence, they are considered as not "real" operations, since they actually are instructions on how to assemble the program. The ABS and ENDABS cards are not available in GPSS V/6000. Standard Numerical Attributes (SNAs) cannot be used on pseudo-operation cards.

END CARD

The END card is used to signal the end of the input deck. It must be present for normal program termination.

When an END card is encountered, control is returned to the operating system from the assembler.

The format of the END card is:



BLOCK NUMBER ASSIGNMENT

There are three methods of altering the numbering system of blocks. Under default conditions, blocks are numbered sequentially by the assembler. If a different numbering scheme is desired, one or more of these methods must be used.

ICT CARD

The format of this card is:



where inc is some integer. The value of inc is added to the current block count. This is the number assigned to the next block encountered. Numbering then proceeds sequentially from this new number.

As an example, if the block count is 10, the cards

2		8		19
		SEIZE		P5
		ICT		6
		RELEASE		P5

are assembled as

2		8		19
10		SEIZE		P5
17		RELEASE		P5

The block numbers skipped are now available for future use.

ORG CARD

This card is similar to the ICT card, but instead of incrementing the block counter, it resets the block counter. Numbering is then continued sequentially from the new number. The format is:

2		8		19
		ORG		number

For example, if the block counter is set at 15, the cards

2		8		19
		ENTER		S3
		ORG		20
		LEAVE		S3

are assembled as

2		8		19
15		ENTER		S3
20		LEAVE		S3

SYN CARD

This card is used to set a new user block name to the same value as a previously defined user block name. The format is:

2		8		19
new		SYN		old

new new name to be defined
 old old name with which the new name is synonymous

For example, if LOC12 is a name previously assigned in the program, the card

2		8		19
RET2		SYN		LOC12

makes the name RET2 synonymous with LOC12. Hence, the two names may be used interchangeably.

ENTITY NUMBER ASSIGNMENT

Normally, entities are numbered automatically by the system as they are encountered. They are numbered in increasing numerical order, starting with one. GPSS V/6000 is a one pass assembler, and because of this fact a slightly different numbering scheme is used from that of the IBM version. For all entities except blocks, names are assigned a value when they are encountered in the source deck. The value assigned is the next unused value of the specified entity at the current time. A couple of examples will make this easier to understand.

Given the following cards

```
1    VARIABLE  FN$ALP*V$PETER
PAUL VARIABLE  FN$BET*V1
BET  FUNCTION  RN1,D3
0,0/0.5,1/1.0,10
ALP  FUNCTION  RN2,C3
0,0/0.9,1/1.0,100
```

the first card would cause function symbol ALP to be assigned the value 1 and the variable symbol PETER to be assigned the value 2 (1 was already used). The second line would assign the value 3 (1 and 2 are used) to variable symbol PAUL and the value 2 to function symbol BET.

If both names and numbers are being used, a problem may arise. For example:

```
ALPHA  FUNCTION  P1,E2
1,V$BBB/2,V$CCC
1      VARIABLE  C1+1
BBB    VARIABLE  C1-1
CCC    VARIABLE  C1*2
```

line 2 above caused variable symbol BBB to be assigned the value 1 since no other references to variables preceded this line. Looking at lines 3 and 4 we see that this is probably not what the user intended. To avoid this problem we could place the function definition after the variable definitions or use the EQU and entity function cards. This problem will not occur if only names or only numbers are used; it only occurs when both are used together.

EQU CARD

This card sets a user-defined name equal to a given numeric value. The format is:

2	8	19
name	EQU	num, list

name user-defined name
num number to be assigned
list list of entity mnemonics; each mnemonic being set off by commas

A list of the entity mnemonics to be used is given in Table 6-1.

For example, the card

2	8	19
ALPHA	EQU	10, F, Q, L

causes the name ALPHA to be assigned the number 10 whenever it is encountered in referencing a facility, queue, or logic switch. If, however, a card such as

2	8	19
	ENTER	ALPHA

is encountered, where the entity referenced is not in the list, the value is generated by the system. Hence, if the ENTER card given above were the first time that a storage was referenced in the program, S\$ALPHA would be the same as S1.

Related to the different treatment for different entities explained in the previous paragraph is the ability to assign a name different numbers for different entity types. This is accomplished through the use of multiple EQU cards. For instance, if it is desired to assign the name ALPHA the number 15 when referring to a storage entity, but 10 when used to reference a facility or queue, the cards used would be

2	8	19
ALPHA	EQU	10, F, Q
ALPHA	EQU	15, S

As before, the name ALPHA is undefined when referring to any other type of entity until assigned a number by the system.

Another function of the EQU card is to assign a value to the untyped name. If an otherwise undefined name is referenced where the reference does not imply an entity type, the value assigned that name on the first applicable EQU card is used. Hence, the cards

2	8	19
ALPHA	EQU	10, F, Q, L
ALPHA	EQU	15, S
	ASSIGN	1, ALPHA

cause the ASSIGN card to be assembled as

2	8	19
	ASSIGN	1, 10

Otherwise an untyped name is assigned the same value as the block symbol of the same name. If there is no block with the same name, an assembly error is issued.

If desired, the user may assign a range of numbers in field A rather than a single number. The designated numbers will not be used by the system for automatic assignment. The format is best illustrated by an example:

2	8	19
SPAN	EQU	5(7), S, Z

This card reserves storage and function numbers 5 through 11. The name SPAN is associated with storage and function 5. Hence, the card

2	8	19
	ENTER	SPAN+3, 5

places five units into storage eight.

ENTITY FUNCTION

The entity function is a form of function used to assign entity numbers to a set of names, rather than a single name. The format is similar to that of other functions:

2	8	19
name	FUNCTION	var, Sn, list
name	name or number of the function	
var	independent variable	
n	an integer	
list	a list of entity mnemonics	

The mnemonics used are the same as those for the EQU card (see Table 6-1).

TABLE 6-1. ENTITY MNEMONICS

Entity	Mnemonic	Entity	Mnemonic
Boolean variable	B	Parameter	P
Facility	F	Savevalue (fullword)	X or XF
Function	Z	Savevalue (halfword)	H or XH
Group	G	Savevalue (byte)	XB
Logic Switch	L	Savevalue (floating point)	XL
Matrix (fullword)	M or MX	Storage	S
Matrix (halfword)	Y or MH	Table	T
Matrix (byte)	MB	User chain	C
Matrix (floating point)	ML	Variable	V
Queue	Q		

The function assigns entity numbers to the names that appear as y values on the function follower cards. For example, if queue 2 and facility 5 have previously been assigned through the use of EQU cards, the cards

1	8	19
ENTTY	FUNCTION	X\$ALPHA, S4, F, Q
1, STAT1/2, STAT2/5, STAT3/17, STAT4		

would assign values as follows:

STAT1 F1, Q1	
STAT2 F3, Q3	Q2 is taken, so cannot be assigned
STAT3 F4, Q4	
STAT4 F6, Q6	F5 is taken, so cannot be assigned

The equating of entity names and numbers is done during the assembly phase. However, the function can also be used during execution. In this case, the function is treated as a discrete valued function. The entity function given previously, for example, if used during program execution, is equivalent to

1	8	19
ENTTY 1,1/2,3/5,4/17,6	FUNCTION	X\$ALPHA, D4

No symbols already used in EQU cards, in other entity functions, or as block names can be used in entity functions.

MACROS

To avoid the use of repetitive coding, GPSS allows the use of macros. These are sections of coding which are used with little or no modification in more than one part of the program, with the modifications being passed in the form of parameters to the macro call.

Macros are defined with macro statements (STARTMACRO and ENDMACRO) to delimit the code, and are called with the macro call card. Macro definition has the form

2	8	19
name	STARTMACRO ----- ----- (body of macro) ----- ----- ENDMACRO	

and the format of the macro call card is:

2	8	19
name	MACRO	A, B, ..., J

name name of the macro being defined or called
A, B, ..., J macro parameters

If a particular parameter is not needed when the macro is called, the position of the parameter must be indicated by successive commas. For example, if parameter C is not necessary when calling macro ALPHA, the macro call card might be

2	8	19
ALPHA	MACRO	P1, X\$AAA, , P3, FN2

As indicated, a macro can have from zero to ten parameters. These are labeled from -A to -J in the macro definition. For example, if a transaction must pass through several facilities and queues with different specifications each time, the macro definition might be

2	8	19
LINE	STARTMACRO QUEUE =A SEIZE =B DEPART =A ADVANCE =C, =D RELEASE =B ENDMACRO	

and the macro call section

2	8	19
LINE	MACRO	1, P\$STAT, 10, 5
LINE	MACRO	2, P\$MACH, 300, FN5
LINE	MACRO	X16, P\$RTE, 27, 16
LINE	MACRO	X17, P\$CAPT, 500, 250

In this example, nine lines of coding are saved by use of the macro. Since the macros are expanded within the system, no actual saving of computer memory is realized.

Comments may appear on a macro call card after 2 blanks. A macro parameter may have one blank.

As a result of this, blanks can also be part of a macro parameter. An example of this is the macro call

2	8	19
ALPHA	MACRO	TEST LE, BETA

Here, the phrase TEST LE is a macro parameter, complete with blank. Trailing blanks are suppressed, so that the field B parameter is just BETA.

There is no limit on the number of macro definitions that can be made, but each macro can have no more than ten parameters. As illustrated in the example, the parameters can only be named =A, =B, ..., =J in the macro definition.

Any GPSS cards except STARTMACRO, JOB, and REALLOCATE may be used in a macro definition. MACROS cannot be used during GPSS UPDATE and are illegal during report generation.

As indicated, a macro cannot be defined within another macro. However, a macro can call another macro. The nesting obtained in this manner is unlimited.

The macro parameters are substituted for the = symbol and no blanks are eliminated. The expanded line must still conform to the coding specifications regarding 72 column maximum length and column 21 for start of comment field.

LISTING CONTROL CARDS

The following paragraphs give a summary of the listing control cards available to the user without the report generator.

COMMENT CARD

The comment card is indicated by an asterisk (*) in column one. Any comments desired can be entered on the rest of the card. The comments will be listed, but ignored by the system.

EJECT CARD

Encountering this card causes a skip to the top of the next page in the program listing. The card itself is not printed. The format is:

2	8	19
	EJECT	

SPACE CARD

This card is used to skip N lines in the listing. The card is not printed. The format is:

2	8	19
	SPACE	N

The operand field contains a line count.

UNLIST CARD

This card deletes the listing of all cards following it until a LIST card. The UNLIST card is not printed. The format is:

2	8	19
	UNLIST	

LIST CARD

The LIST card causes listing that has been stopped by an UNLIST card to resume. The LIST card is not printed. The format is:

2	8	19
	LIST	

NOXREF CARD

The NOXREF card causes no further cross references to be gathered. References gathered prior to the NOXREF card will be printed. This does not stop the cross reference map from being printed since some references are gathered during the input phase. In fact, this card can be dangerous to use since stopping the gathering of references can cause the interpreter to become confused. The cross reference map can be suppressed by the control card parameter - NX.

2	8	19
	NOXREF	

XREF CARD

The XREF card resumes gathering cross references and enables the cross reference map to be printed.

2	8	19
	XREF	

TITLE CARD

The TITLE card causes the rest of the card, starting with the first nonblank to be printed as a subtitle under the main title produced by the JOB card. All title cards after the first cause a page eject. The TITLE card is not printed. The format is:

2	8	19
	TITLE	SUBTITLE TEXT

This card is not to be confused with the TITLE card of the report generator.

O

O

O

O

O

O

O

The GPSS Report Generator is designed to allow the user to tailor the output statistics to his preference. There are several possibilities offered. A user can select the statistics to be printed, print section titles, insert comments, control the order of the output, insert blank lines and page ejects, or display SNAs graphically. The commands used are divided into two areas: graphic output and selection of statistics and formatting of output.

NOTE 1: Fixed field coding is required for report definitions.

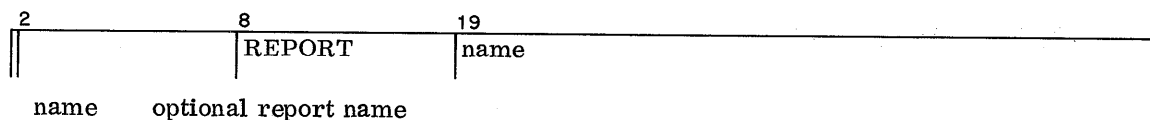
NOTE 2: Since GPSS V/6000 is a one pass assembler the entire report definition must appear before the START card. Otherwise the standard report is used.

SELECTION OF STATISTICS AND OUTPUT CONTROL

The commands used in this part of the report generator cause some of the standard output statistics to be moved or deleted entirely. The section begins with a REPORT card and ends either with an END-REPORT card or the first nonreport type card.

REPORT CARD

The REPORT card must be the first card in a report definition routine. The report may or may not be named. If it is, the name is indicated on this card. The card format is:



Only one unnamed report is allowed. An unnamed user-defined report is used whenever a simulation having a START card with a blank E field is executed. If there is no unnamed user-defined report, the standard output is produced.

If a report is given a name, it will be used only when called by that name. This is done by entering the name in field E of the appropriate START card.

A report routine can be redefined at any time; however, only the latest definition is kept. Hence, only one unnamed routine can be kept.

GPSS V/6000 automatically writes up to 32 different report generator routines on a local file named GPSSRPT during a GPSS run. This includes an unnamed report if one exists. These reports can be used in subsequent GPSS runs if the file GPSSRPT is saved (using appropriate operating system control cards) during the run in which it is created. Later, the file should be attached with the local file name GPSSRPT prior to executing GPSS models that will use these reports. This has the advantage of not requiring the user to redefine frequently used report generator routines. He simply names the report on the START card in order to use it.

Example

```
.  
. .  
GPSS,FX.  
SAVE,GPSSRPT.  
. .  
-EOR-  
SIMULATE  
. .  
REPORT RPT1  
. .  
ENDREPORT  
REPORT RPT2  
. .  
ENDREPORT  
. .  
START 100,,,RPT1  
END
```

Later:

```
. .  
GET,GPSSRPT.  
GPSS,FX.  
. .  
-EOR-  
SIMULATE  
. .  
START 100,,,RPT2  
END
```

ENDREPORT CARD

This card causes the end of a report routine to be generated. It is optional, since a report routine will also be terminated by the first nonreport card encountered.

TITLE CARD

The TITLE card is used to provide titles for sections of the statistical printout. The format is:

2	8	19
type	TITLE	entity, title

where type is the entity type, specified using the mnemonics given in Table 7-1, entity is the specific entity, or blank, and title is the title to be printed.

TABLE 7-1. ENTITY SPECIFICATION MNEMONICS

Entity	Mnemonic
Block Counts*	BLO
Savevalues - fullword	SAV or FSV
Savevalues - halfword	HSAV or HSV
Savevalues - byte	BSV
Savevalues - floating point	LSV
Matrix Savevalues - fullword	MSAV or FMS
Matrix Savevalues - halfword	MHSA or HMS
Matrix Savevalues - byte	BMS
Matrix Savevalues - floating point	LMS
Chain Statistics	CHA
Facility Statistics	FAC
Storage Statistics	STO
Queue Statistics	QUE
TABLE Statistics	TAB
Group Members	GRO
Clock Statistics*	CLO

If the entity specification (A field) is blank, all defined entities of the type specified in "type" are printed. If the specification is a number, the output for that specific entity is printed. The format is the standard output format.

The title as it appears in the B field of the TITLE card (including leading blanks) is printed in column 2 of the output file. GPSS places a blank in column 1 of the output file to be used as carriage control by the line printer. If the title is too long for a single card, it may be continued onto a second card by placing a nonblank character in column 72 of the first card and continuing with column 1 of the second card. Only one continuation card is allowed. The character used in column 72 will not be included as part of the title.

The title to be printed is left-justified, starting in column one. Therefore, care must be taken in title specification to ensure proper carriage control. If the title is too long for a single card, it may be continued onto a second card by placing a nonblank character in column 72 of the first card, and continuing beginning with column one of the second card. Only one continuation card may be used.

* May not specify specific entities in TITLE card.

INCLUDE CARD

The INCLUDE card is used to specify which columns of the standard output are to be printed. The format is:

2	8	19
type	INCLUDE	list
type	the entity type	
list	range of entity numbers followed by a list of the specific columns to be printed	

The mnemonics used to specify "type" are contained in Table 7-1. A list of the columns for each entity type is contained in Table 7-2. Other types may specify only a range of entity numbers.

TABLE 7-2. STANDARD PRINTOUT COLUMN NUMBERS

Entity (Mnemonic)	Number/Name
Facilities (F)	<ol style="list-style-type: none"> 1. Facility name/number 2. Average utilization 3. Number of entries 4. Average time per transaction 5. Seizing transaction number 6. Pre-empting transaction number
Queues (Q)	<ol style="list-style-type: none"> 1. Queue name/number 2. Maximum contents 3. Average contents 4. Total entries 5. Zero entries 6. Percentage zero entries 7. Average time per transaction 8. Average time per nonzero transaction 9. Table number 10. Current contents
Storages (S)	<ol style="list-style-type: none"> 1. Storage name/number 2. Capacity 3. Average contents 4. Average utilization 5. Number of entries 6. Average time per transaction 7. Current contents 8. Maximum contents

TABLE 7-2. STANDARD PRINTOUT COLUMN NUMBERS (CONT'D)

Entity (Mnemonic)	Number/Name
Tables (T)	<ol style="list-style-type: none"> 1. Table name/number 2. Entries - nonweighted 3. Mean-nonweighted 4. Standard deviation - nonweighted 5. Sum of arguments - nonweighted 6. Entries - weighted 7. Mean - weighted 8. Standard deviation - weighted 9. Sum of arguments - weighted 10. Upper limit 11. Observed frequency 12. Percent of total 13. Cumulative percentage 14. Cumulative remainder 15. Multiple of mean 16. Deviation from mean
User Chains (CH)	<ol style="list-style-type: none"> 1. User chain name/number 2. Total entries 3. Average time per transaction 4. Current contents 5. Average contents 6. Maximum contents

For example, the card

2	8	19
FAC	INCLUDE	F5-F11/1, 3, 4

specifies that the statistics for facilities five through 11 are desired, but that only columns one, three, and four of the standard output are to be printed.

If an INCLUDE card immediately follows a TITLE card and specifies the same entity type, then all ranges and restrictions on the INCLUDE card apply to the TITLE card as well.

FORMAT CARD

The FORMAT card allows the user to mix columns of output from various entity types in a single listing. The format is:

2	8	19
start	FORMAT	range/list

start	column in which printing starts
range	entity number range to which the FORMAT card applies
list	list of output statistics columns to be printed

A list of the legal mnemonics for "list" is contained in Table 7-3. Each column uses 18 print columns on the output sheet.

TABLE 7-3. LEGAL FORMAT CARD MNEMONICS

Mnemonic	Meaning
CH1 CH2 CH3 CH4 CH5 CH6	User chain name or number Total entries in chain Average time per transaction in chain Current contents of chain Average contents of chain Maximum contents of chain
F1 F2 F3 F4	Facility name or number Average facility utilization Number of entries in facility Average time per transaction in facility
Q1 Q2 Q3 Q4 Q5 Q6 Q7 Q8 Q10	Queue name or number Maximum contents of queue Average contents of queue Total entries in queue Zero entries in queue Percentage of zeros in queue Average time per transaction in queue Average time per nonzero transaction Current queue contents
S1 S2 S3 S4 S5 S6 S7 S8	Storage name or number Storage capacity Average contents of storage Average utilization of storage Number of entries in storage Average time per transaction Current storage contents Maximum storage contents
T1 T2 T3 T4 T6 T7 T8	Table name or number Entries in table - nonweighted Mean argument for table - nonweighted Standard deviation - nonweighted Entries in table - weighted Mean argument for table - weighted Standard deviation - weighted
X1 X2	Savevalue name or number Savevalue contents
XH1 XH2 XB1 XB2 XL1 XL2	Halfword savevalues name or number Halfword savevalues contents Byte savevalue name or number Byte savevalue value Floating point savevalue name or number Floating point savevalue value

For example, the card

2	8	19
5	FORMAT	5-10/F1, S8, X2, Q10

will produce six lines of output, each line having four columns. The first column contains the names or numbers of facilities five through 10, the second column the maximum contents of storages five through 10, the third column the contents of fullword savevalues five through 10, and the fourth column the current contents of queues five through 10. There are no column headings on formatted statistics.

TEXT CARD

The TEXT card is used to print data in sentence form. This is done by mixing alphanumeric text and data. The format is:

2	8	19
start	TEXT	text

start column in which the text is started
text alphanumeric text and data

All text is printed. The data is indicated by the phrase #data/format#, including the sharp marks. The data portion consists of an SNA name or number, followed by a comma, and completed by a number indicating the column number on the standard output statistics. The format portion indicates first the movement, if any, of the decimal point, and then a picture of the desired output format. An L indicates that the decimal point is to be moved to the left, and R indicates that it is to be moved to the right. Text may be continued on a second card by placing a non blank character in column 72. This character will not appear in the output.

For example, if the average utilization of facility five is .2734, the card:

2	8	19
11	TEXT	AVG. USAGE IS #F5, 2/2RXX#

would produce

AVG. USAGE IS 27

with the line beginning in column 11. Numbers are rounded, not truncated, if fewer digits are specified than actually exist. However, truncation on the left can occur. If, for example, the utilization in the previous example was 1.000, the output printed would be

AVG. USAGE IS 00

The column numbers used are those used for the FORMAT card, as shown in Table 7-3, except that column 1 is never used.

The following examples illustrate the use of the format portion:

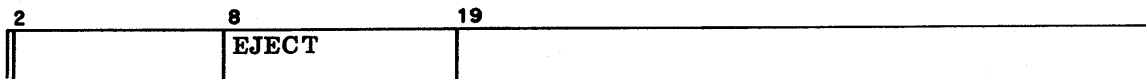
Original Data	Format	Printed Result
56742	3LXX.X	56.7
.29743	2RXX.XX	29.74
915.33	XXXX.X	915.3

COMMENT CARD

Comments are entered by putting an asterisk in column one. The card, from column 2 to column 71, is then printed, beginning in print column one. A comment can continue on the next card by entering a nonblank character in column 72 of the first card.

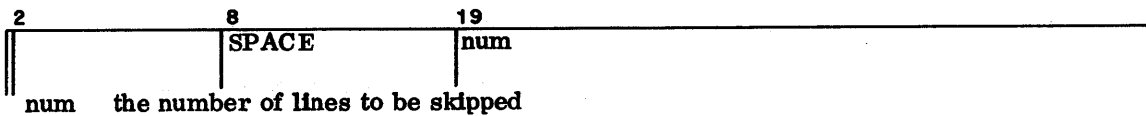
EJECT CARD

The EJECT card causes the printer to skip to a new page in the output. The format is:



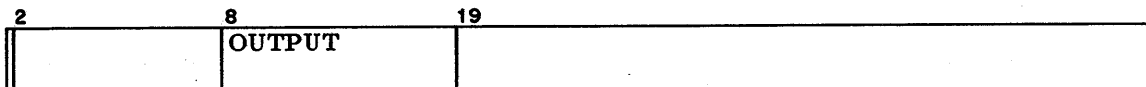
SPACE CARD

The number of lines specified in field A are skipped. The format is:



OUTPUT CARD

This card produces the standard statistical output printout in the report. The format is:



GRAPHIC OUTPUT

GPSS V/6000 provides the user with the capability of presenting some statistical output in graphic form. The graphical output is in the form of a histogram, with each box or bar representing an SNA and the height representing the actual value of the SNA.

When plotting graphic output, the graph area is a 60 row by 136 column raster. When locating a point on the graph area, rows are numbered from top to bottom, and columns are numbered from left to right. Column 1 is not printed, but is used for carriage control. Hence, it is usually better to leave column one blank.

GRAPH CARD

A GRAPH card must be the first card in a graph definition. The format is:

2	8	19
	GRAPH	sna, low, high, char
sna	the SNA to be plotted	
low	either the lower limit of the plot range or a table number	
high	upper limit of the plot range	
char	the character used in plotting.	

The term "plot range" refers to the range of entities that will have the appropriate SNA plotted, not to the range of values that the SNA may take. A list of legal SNA mnemonics for the GRAPH card is contained in Table 7-4. If no plotting character is specified, asterisks are used. As with the TEXT card, numbers are rounded, not truncated, for graphing.

Some of the table SNAs (those preceded by an asterisk in Table 7-4) actually require a separate graph for each table. In these cases, the table number is given as "low" and the C field is left blank.

TABLE 7-4. LEGAL SNA MNEMONICS FOR "GRAPH" CARDS

Entity	Mnemonic	Meaning
Blocks	N	Block count
Facilities	FC FR FT	Entry count Utilization Average time per transaction
Groups	G	Current contents
Queues	Q QA QC QM QT QX QZ	Current contents Average contents Entry count Maximum contents Average time per entry Average time per nonzero entry Zero entries
Savevalues	X XH	Fullword savevalues contents Halfword savevalues contents
Storages	S SA SC SM SR ST	Current contents Average contents Entry count Maximum contents Utilization Average time per entry

TABLE 7-4. LEGAL SNA MNEMONICS FOR "GRAPH" CARDS (CONT'D)

Entity	Mnemonic	Meaning
Tables	TB	Mean
	TC	Entry count
	*TD	Cumulative percent
	*TF	Observed frequency
	*TP	Percent of total
	*TR	Cumulative remainder
	TS	Standard deviation
User chains	CA	Average contents
	CC	Entry count
	CH	Current contents
	CM	Maximum contents
	CT	Average time per entry
	XB	Byted savevalues
	XL	Floating point savevalues (Integer values)

For example, the card

```

      2           8           19
      ||-----|-----|-----
      ||         |GRAPH   |QM, ALPHA, EPSOL, +
  
```

would plot the maximum contents of queues ALPHA through EPSOL, and the character + would be used to plot the axes and the histogram bars. As a further example, the card

```

      2           8           19
      ||-----|-----|-----
      ||         |GRAPH   |TF, WAIT, , $
  
```

causes the observed frequency of table WAIT to be plotted, using the character "\$".

ORIGIN CARD

The ORIGIN card is used to locate the origin of the graph to be plotted on the 60 by 136 raster of the plotting sheet. The format is:

```

      2           8           19
      ||-----|-----|-----
      ||         |ORIGIN  |row, col
  
```

row the row on which the x axis is printed
 col the column on which the y axis is printed

As noted above, rows are numbered from top to bottom, not bottom to top. Columns are numbered in the standard fashion, from left to right.

When specifying the position of the origin, care should be taken that sufficient room is allowed for labeling the axes. Labels for the y axis are positioned to the left of the axis, and labels for the x axis are positioned below the axis.

X CARD

The X card is used to describe the x axis, and also to describe the plotting specifications along the x axis. The format is:

```
||-----|-----|-----|
  2         8         19
  |         |X         |sym,width,space,,,no
```

sym either blank or SYM
width width of the bars being plotted
space the spacing between bars
no either blank or NO.

Note that fields D, E, and F are not used. As always, however, they must be indicated by successive commas.

If field A contains the characters SYM, symbolic labeling of the x axis is used; otherwise, numeric labeling is used. Field G causes axis labeling of the x axis to be suppressed if it contains the characters NO.

If the width is omitted, the default value is three. Similarly, if the space is omitted, the default value is one.

For the SNAs which require a separate graph for each SNA (that is, TD, TF, TP, and TR), a different format for the X card is required. This format is:

```
||-----|-----|-----|
  2         8         19
  |         |X         |,width,space,upper,class,inc,no
```

width bar width for SNAs TF and TP (blank for SNAs TD and TR)
space the space between bars for TF and TP or the space between points
for TD and TR
upper upper limit of the lowest frequency class
class number of classes per x axis increment
inc number of increments
no as in the previous format.

Note that this implies that SNAs TD and TR can have plotted bar widths of one; otherwise, fields A, B, C, and G are the same as in the other format.

Field D (upper) is the upper limit of the lowest frequency class to be plotted, and need not be the same as the value that appears on the table definition card.

Field E (class) is used to condense the information from several frequency classes into a single bar. For instance, if each bar of the graph is to represent three frequency classes, field E must contain the character 3. In this case, field D would have to be no less than the upper limit of class three in the table.

Field F (inc) indicates the number of bars that are to be plotted.

It is the user's responsibility that a sufficient number of entries are defined or exist in a table to fill the entire X card request.

Y CARD

The Y card is equivalent to the X card, but describes the y axis. Since the y axis is much simpler to describe in a histogram than the x axis, the format of the Y card is different. The format is:

2		8		19	
		Y			
			lim, size, num, rows		

lim lower limit of y, and will be used as the y axis label
size size of each increment
num number of increments
rows number of rows for each increment

The increments referred to are the space left between successive labeling of the y axis. Thus, the card

2		8		19	
		Y			
			25, 8, 4, 4		

would cause four labels to be put on the y axis. The labels would be four printer lines apart, and would be 25, 33, 41, and 49.

STATEMENT CARD

The STATEMENT card causes text to be printed on the graph being specified. The format is:

2		8		19	
col		STATEMENT	row, char, text		

col starting column number
row row number
char character count
text text to be printed

The statement may be continued onto the next card by putting a nonblank character in column 72.

ENDGRAPH CARD

This card is used to indicate the end of a graph definition and, unlike the ENDREPORT card, is not optional. The format is

2		8		19	
		ENDGRAPH			

SAMPLE REPORT DEFINITIONS

These are two examples of report definitions. The first is a very simple text report. The second is a graph definition. The cards in the graph definition example represent the required order. While the STATEMENT card is not required, it cannot be placed in a different portion of the report definition.

2	8	19
FAC 5	REPORT TITLE TEXT ENDREPORT	SUE GRAB,REPORT ON WIDGET MACHINE GRAB UTIL, IS UP TO 'F1, GRAB/2RXX'

The following is an example of a graph definition.

2	8	19
2	REPORT GRAPH ORIGIN X Y STATEMENT ENDGRAPH ENDREPORT	WALLY FR,1,10,\$ 50,10 SYM,3,5 0,30,33,1 5,20,FACILITY UTILIZATION

O

O

O

O

O

O

O

There are three types of HELP blocks that will allow the user to incorporate independently written FORTRAN routines.

HELPA - One-way communication via a six word read-only array.

HELPB - Two-way communication via six savevalues.

HELPC - One-way communication via a six word read-only array, plus access to the actual GPSS entity tables via an additional 19 arrays.

PROCEDURE

The procedure for using HELP blocks follows.

1. Code the desired routines following the usual coding procedures for FORTRAN. Blank common is not allowed in the HELP routines or anything they use.
2. If I/O is to be done in the routines, the following FORTRAN main program must be included:

```
PROGRAM name(filelist)
CALL FTNRET
END
```

name Any legal name for a FORTRAN main program.
filelist List of all files that are used by the subroutines. This should be just the same as an independently executed FORTRAN program.

Example

If the user wants the subroutine to read from INPUT via unit 5, write to OUTPUT via unit 6, and write to file FTNOUT via unit 9, the following main program would be needed:

```
PROGRAM MAIN (INPUT, OUTPUT, FTNOUT, TAPE5=INPUT, TAPE6=OUTPUT, TAPE9=FTNOUT)
CALL FTNRET
END
```

3. Compile the FORTRAN routines in the normal manner. Users must include the STATIC option on the compilation command to prevent CMM from being used. For example:
FTN,I=HELPI, L=0, B=BIN, STATIC.*
4. Execute the GPSS model using the LF = lfn option on the GPSS control card. lfn is the local file name of the file with the relocatable binaries of the FORTRAN routines.

*See Appendix H.

The main program may be named anything and appear anywhere in the file.

The main program is optional if none of the routines do I/O.

GPSS will dump all buffers and close all files, including those used by FORTRAN.

Data that is to be read from INPUT should be on a separate record immediately following the GPSS model. Data should not go after the START card.

Output from the FORTRAN routines should not go to the same file as the GPSS output file (in particular, OUTPUT). If this is done, the output from the two could be intermixed. The best results are obtained by having one write to file A and the other write to file B. Then copy file A to the end of file B. For example:

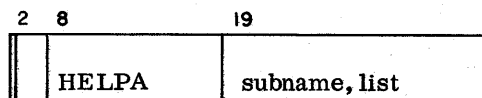
```
.  
. .  
. .  
GPSS, FX, L=ABC, LF=BIN.  
REWIND, ABC.  
COPY, ABC.
```

The OUTPUT file will then have the FORTRAN output followed by the GPSS output.

It is the user's responsibility to insure that his field length is large enough to hold both GPSS and the FORTRAN binaries. If not, the loader will not load the binaries, but GPSS will continue to execute. A message will be placed in the dayfile. It is quite possible to need a field length of more than 100,000. On NOS, the user should be in REDUCE, - mode.

HELPA

The FORTRAN HELPA subroutine is coded with only one argument. That argument must be an array dimensioned with a length of six. This is a read-only array; any changes to its contents by the FORTRAN routine will not effect the GPSS program.



subname Name of the subroutine to be executed.
list Up to six standard numerical attributes.

If less than six SNAs are listed, the remaining values of the array will be zero.

Example

2	8	19
	HELPA	STATS,AC1,TG1,Q,FC1,N17,N13

```
PROGRAM XYZ (FTNOUT, TAPE6=FTNOUT)
CALL FTNRET
END
```

```
SUBROUTINE STATS(IPAR)
DIMENSION IPAR(6)
DATA KOUNT/0/
KOUNT = KOUNT + 1
WRITE(6, 11) KOUNT, (IPAR(I), I=1, 6)
FORMAT(" ", 7(16, 5X))
RETURN
END
```

HELPB

The savevalues passed by the HELPB block are the actual savevalue locations; any changes of these savevalues will affect the GPSS program. Integer savevalues should be passed to integer variables and floating point savevalues should be passed to floating point variables.

Fields B through G of the block should be coded with savevalue prefixes instead of suffixes. Thus GPSS V/6000 accepts

```
HELPB ALPHA,XL1,XF$BETA,XL3
```

instead of

```
HELPB ALPHA,1XL,BETA$XF,3XL
```

2	8	19
	HELPB	subname, list

subname Name of the subroutine to be executed.
list Up to six savevalues, any type.

HELPC COMPUT, XF\$ALPHA, XF\$BETA, XL\$GAMMA

```
      SUBROUTINE COMPUT(IALPHA, IBETA, GAMMA)
C     COMPUTE THE SQUARE ROOT OF IALPHA
      IALPHA = SQRT(IALPHA)
C     COMPUTE THE AREA OF A CIRCLE WITH RADIUS IBETA
C     RETURN ANSWER IN GAMMA
      GAMMA = 3.14159*IBETA**2
      RETURN
      END
```

No main program is needed because no I/O is done.

HELPC

The six SNAs that are to be passed to the FORTRAN subroutine are listed in fields B through G. All other parameters are passed automatically. Arrays must be reserved for all parameters even if they are not used.

The six SNAs are a read-only array. Changes to them will not affect the GPSS program.

The additional nineteen arrays are the actual GPSS entity tables. Changes to these tables will affect the GPSS program. Great care should be taken when modifying these tables because GPSS does not check them upon return from the HELPC routine, but assumes they are in the correct format. Appendix A contains the structure of these tables. It should be studied carefully before any attempt is made to use the HELPC block.

Table 8-1 shows what the variables in the parameter list are used for.

2	8	19
	HELPC	subname, list

subname Name of the subroutine to be executed.
list Up to six standard numerical attributes.

If less than six SNAs are listed, the remaining values of the array will be zero.

Example

2	8	19
	HELPC	SUB3

```
PROGRAM MAIN(FTNOUT, TAPE9=FTNOUT)
CALL FTNRET
END
SUBROUTINE SUB3(SNA, BLO, VAR, FUN, BVA, CHA, FAC, GRO, HSA, LOG, QUE,
STO, $ TAB, FMA, FSA, HMA, BMA, LMA, BSA, LSA)
INTEGER SNA(6), BLO(2, 1), VAR(1), FUN(2, 1), BVA(1), CHA(3, 1), FAC(5, 1)
INTEGER GRO(1), HSA(1), LOG(1), QUE(4, 1), STO(7, 1), TAB(9, 1), FMA(1)
INTEGER FSA(1), HMA(1), BMA(1), LMA(1), BSA(1)
REAL LSA(1)

C
C PRINT TOTAL BLOCK COUNT AT BLOCK 77
C
      ICNT = SHIFT(BLO(1, 77), 30).AND.7777777777B
      WRITE(9, 101) ICNT
101  FORMAT(" BLOCK COUNT AT BLOCK 77 IS ", IS)

C
C CHANGE VALUE OF FULLWORD MATRIX 9 ELEMENT (3, 7) TO 37
C
C NUMBER OF COLUMNS
      ICOL = SHIFT(FMA(9), 24).AND.777777B

C
C CALCULATE ABSOLUTE LOCATION IN CORE
C
      LABSLOC = (ICOL*(3-1) (7-1))+(FMA(9).AND, 777777B)

C
C CHANGE LOCATION
C
      FMA(IABSLOC-LOCF(FMA)+1)=37

C
C FINISHED
C
      RETURN
      END
```

ACCESS TO TABLES

To access BLOCK information, first look in table 8-1 for the variable that holds the BLOCK information. The dummy variable is BLO with two words per block. Information is accessed as:

```
BLO(i, j)
i = 1, 2 corresponding to the desired word
j = n the block number
```

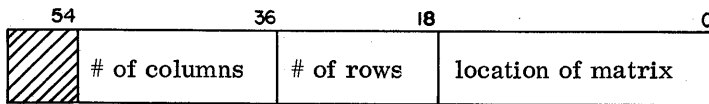
Appendix A shows how these two words are used. By using the FORTRAN SHIFT function and the Boolean .AND. with a mask of 7B's it is possible to extract the information.

Example

To obtain the total block count from bits 30-59 first shift the word 30 bits to the left to get the desired field in the right part of the word. Then form a mask of octal 7's (one seven for each three bits to be extracted) and do a logical .AND. between the shifted word and the mask.

```
KOUNT = SHIFT(BLO(i, j), 30).AND. 7777777777B
```

To access matrix savevalues a more complicated method is needed. FMA is the dummy variable name of the MSAVEVALUE array pointers. These pointers have the format:



First, get the number of columns by:

```
ICOL = SHIFT(FMA(i), 24).AND. 777777B
i is the number of the desired MSAVEVALUE
```

Use ICOL to compute the relative location of the matrix element (r,s) relative to the first word of the matrix by:

```
IRELLOC = ICOL*(r-1)+(s-1)
```

Compute the actual core location of element (r,s) of MSAVEVALUE i by:

```
IABSLOC = IRELLOC + (FMA(i).AND. 777777B)
```

Now that the absolute location in core is known, a way to address it relative to some other location that FORTRAN knows is needed. This is done by finding the distance between the first location of the matrix pointers and the location of the elements themselves. The result is an index for FMA that can be used to access the desired element.

```
VALUE = FMA(IABSLOC - LOCF(FMA)+1)
```


For halfword matrices the relative location from above must be divided by two, and then the correct 30 bits masked out.

ICOL = SHIFT(HMA(i), 24).AND. 777777B
IREL = ICOL*(r-1)+(s-1)
IRELLOC = IREL/2

The value is located within the word by using the MOD function.

IABSLOC = IRELLOC+(HMA(i).AND. 777777B)
IWORD = HMA(IABSLOC-LOCF(HMA)+1)
IMOD = MOD(IREL, 2)

case IMOD of
0: VALUE = IWORD.AND. 7777777777B
1: VALUE = SHIFT(IWORD, 30).AND. 7777777777B

For byte matrices the relative location must be divided by four, and the correct 15 bits masked out.

ICOL = SHIFT(BMA(i), 24).AND. 777777B
IREL = ICOL*(r-1)+(s-1)
IRELLOC = IREL/4

IABSLOC = IRELLOC+(BMA(i).AND. 777777B)
IWORD = BMA(IABSLOC - LOCF(BMA)+1)
IMOD = MOD(IREL, 4)

case IMOD of
0: VALUE = IWORD.AND. 777777B
1: VALUE = SHIFT(IWORD, 45).AND. 777777B
2: VALUE = SHIFT(IWORD, 30).AND. 777777B
3: VALUE = SHIFT(IWORD, 15).AND. 777777B

00



00

00

ALLOCATION SCHEMES

A |

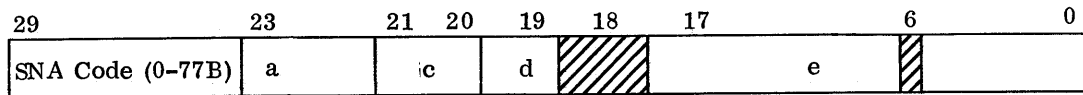
The following pages give the allocation schemes for the various GPSS V/6000 entities. With the exception of the Standard Numerical Attributes, which have 30 bits apiece allocated, all words are 60 bits long.

Any locations within words with contents not explicitly defined are reserved for system use.

Note that for group block and matrix entities, memory additional to the minimum may be required. In such a case, no more additional memory than is minimally required is used.

STANDARD NUMERICAL ATTRIBUTES (SNA) ALLOCATION

All standard numerical attributes are converted to a 30-bit interval form.

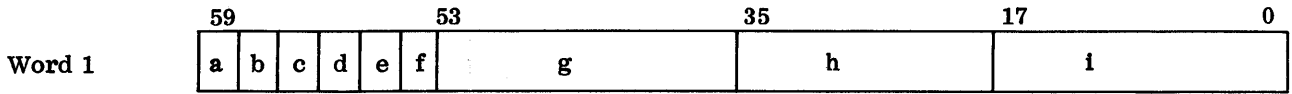


- a GPSS indirect SNA - bits 0 through 17 contain address of word containing indirect SNA in lower 30 bits
- c suffix field
 - 00 - half
 - 01 - byte
 - 10 - full
 - 11 - floating point
- d 1 = suffix field present
0 = suffix field not present
- e 18 bit signed constant. Entity number for SNA or user symbol location.

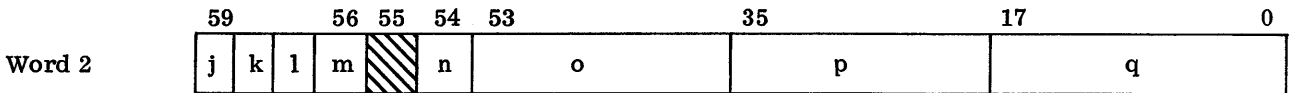
All 30 bits set to 1 indicates a blank field instead of an SNA.

TRANSACTION ALLOCATION

No table, comes from available storage as needed.



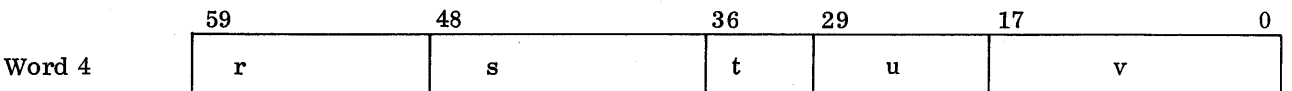
- a scan bit
- b interrupt chain
- c matching chain
- d trace flag
- e current events chain
- f future events chain
- g next block location
- h next entry in chain
- i previous entry in chain



- j = 0 ignore bit 58
= 1 in transfer all/both
- k = 0 in transfer both
= 1 in transfer all
- l SIM indicator
- m last block of all indicator
- n pre-empt status indicator
- o next assembly set member
- p current block location
- q parameter location

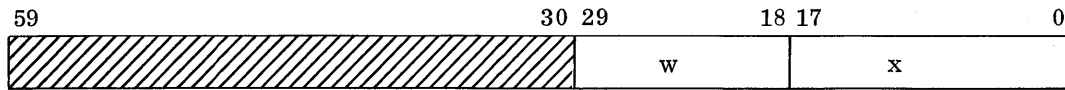


NOTE: When not in FEC (future events chain) Bits 0-29 are used as a scratch area



- r # words for parameters
- s pre-empt count
- t priority
- u queue in
- v queue link location

If a transaction is in more than one queue then one additional word of storage for each queue is requested and returned when the transaction departs the queue.



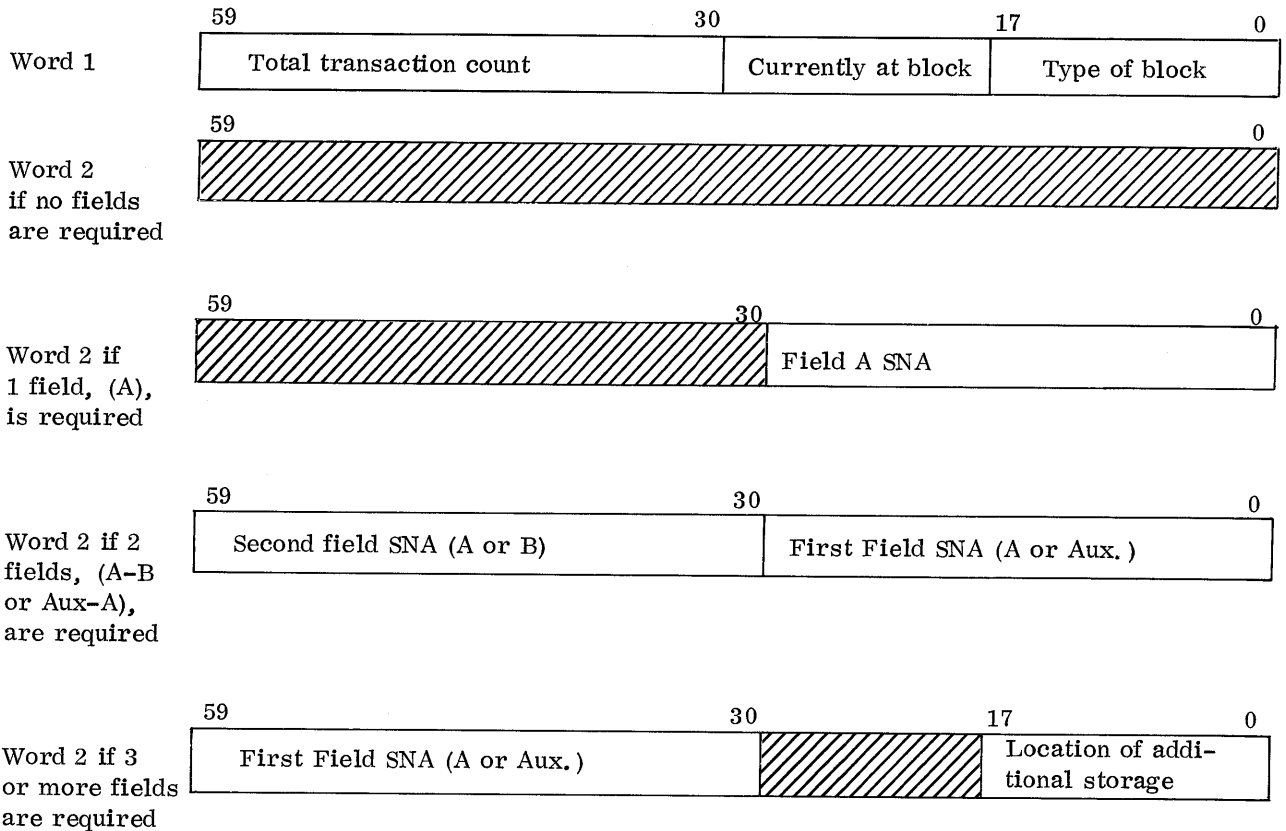
w queue number
 x next queue word or zero

Parameter storage is requested when the transaction is created and returned when it is destroyed. See parameter description in Job Tape Format (Appendix C).

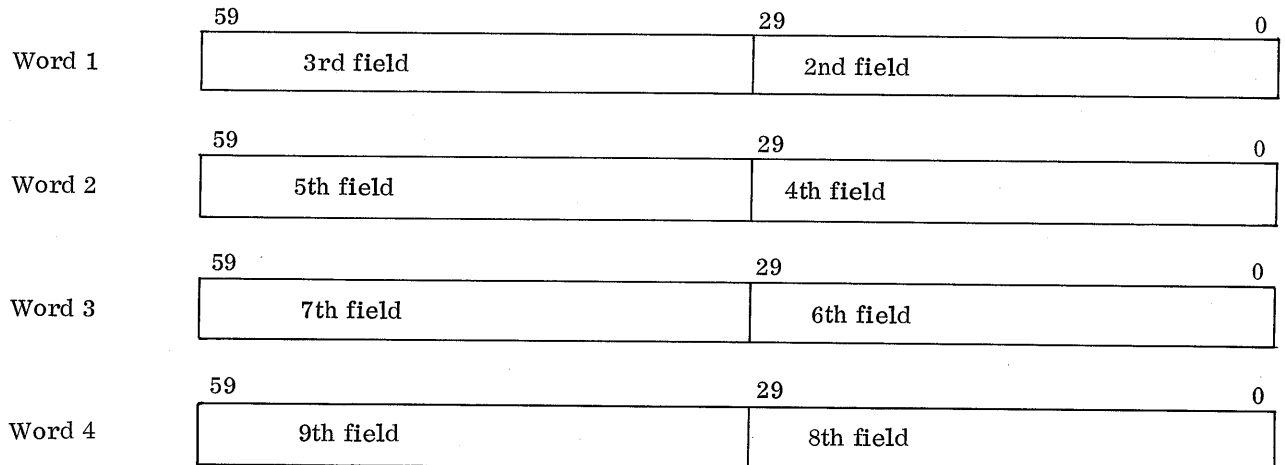
If a transaction is in a group then $u = 0$ and v points to another word as above except bits 30-59 contain the group number instead of 0 (for queue).

BLOCK ALLOCATION

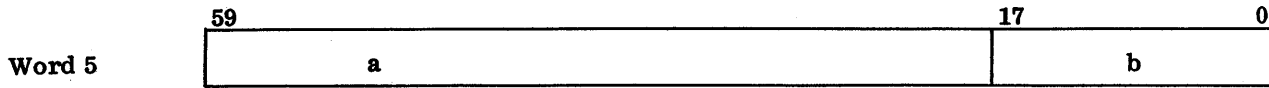
Every field on a block, whether used or not, requires 30 bits (Aux field, field A, ... field I). If a block requires 2 or less fields, then no additional storage is needed for the block. If more than 2 fields are required, then additional storage is requested for the extra blocks.



Additional storage is assigned as follows:

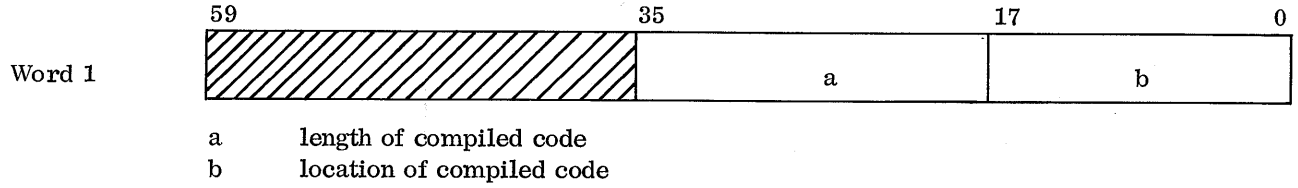


The generate block is a 5th word as follows:

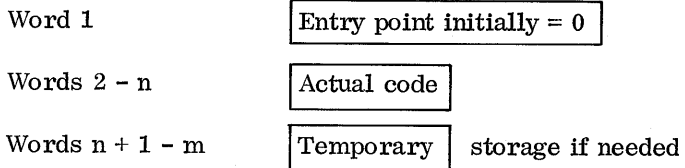


- a 41 bit creation limit. This is decremented by each transaction created by the block until zero, at which time the generate ceases.
- b link of generate blocks. Next generate block entry in block table, or zero for end.

VARIABLE ALLOCATION

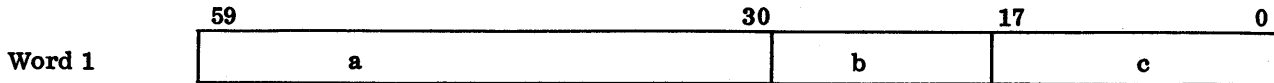


The compiled code looks like:



If word 1 is not zero, the variable is currently being executed and any attempt to reuse it will give a cyclic definition error.

FUNCTION ALLOCATION



- a field A SNA. (Independent variable)
- b number of words in definition. For C, D, and E functions this is the number of points *2. For L and M functions this is the number of points.
- c location of array containing points



- d = 0 not used in cyclic definition
= 1 used in cyclic definition
- e type of function
 - 1=C
 - 2=D
 - 3=E
 - 4=L
 - 5=M
- f location of temporary 9 word array used for recursive purposes

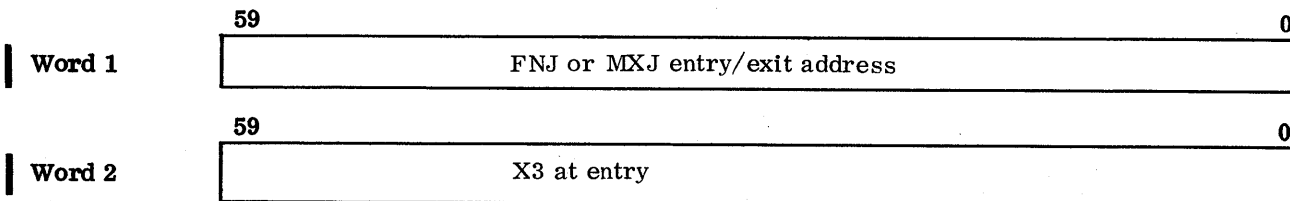
For C, D, and E functions the X and Y values are stored together in the following manner:

X1
Y1
X2
Y2
:
:
Xn
Yn

For L and M functions only the Y values are stored in order.

All values are stored as floating point numbers. SNAs are stored as negative indefinite with the lower 30 bits the SNA.

The recursive 9 word array is requested when function evaluation is started and returned when done.



Word 3	59	0	A3 at entry
Word 4	59	0	X4 at entry
Word 5	59	0	A4 at entry
Word 6	59	0	B6 at entry
Word 7	59	0	B2 at entry
Word 8	59	0	SVAA at entry
Word 9	59	0	SVA at entry

TABLE ENTITY ALLOCATION

	59		53			30		0
Word 1	a	b	c	d	e	f	Entity width	Entry count

a = 1 if weighted table
 b = 1 if IA table
 c = 1 if RT table
 d = 1 if difference table
 e = 1 if normal table
 f = 1 if queue table

	59							0
Word 2	Sum of arguments (floating)							

	59							0
Word 3	Sum of squares of arguments (floating)							

	59							0
Word 4	Sum of weighted arguments (floating)							

	59							0
Word 5	Sum of squares of weighted arguments (floating)							

	59				41			0
Word 6 (if RT table)	RT Link	Next RT table	Zero if none				Sum of RT Units	

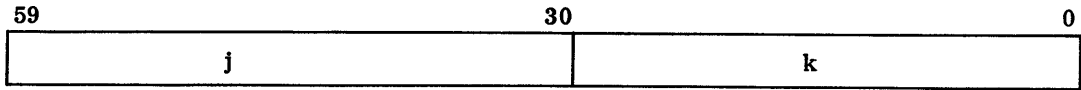
	59							0
Word 6 (If an IA or difference table)	Last tabulation (Integer)							

	59							0
Word 7	Sum of weighted overflow (floating)							

	59				30		17	0
Word 8	g				h		i	

g upper limit of lowest frequency class (signed 30 bit integer)
 h number of classes
 i location of classes

Word 9



j if an RT table, then non-weighted sum. Otherwise, not used.
k table argument (SNA) or RT interval

The frequency classes are a contiguous set of locations containing the integer tabulation of the classes. The last location is the overflow class count.

STORAGE ENTITY ALLOCATION

Word 1	59	29	0
	Available Contents		Current Contents

Word 2	59	0
	Cumulative time integral (floating)	

Word 3	59	29	0
	Maximum Contents		Last time cumulative time updated

Word 4	59	54	36	18	0
	a	b	c	d	

- a = 0 available
= 1 unavailable
- b reduction in storage delay chain, or zero if none
- c storage full/not full delay chain or zero
- d storage empty/not empty delay chain or zero

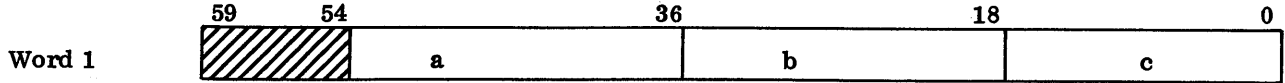
Word 5	59	48	30	0
	e	f		

- e storage available/unavailable delay chain
- f time unavailable

Word 6	59	29	0
	Total entries		Entry count

Word 7	59	0
	Unavailable cumulative time integral (floating)	

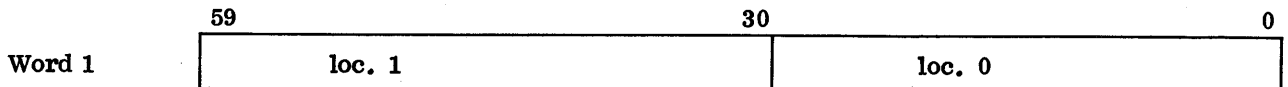
MATRIX ALLOCATION



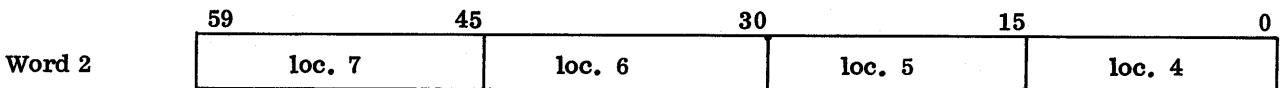
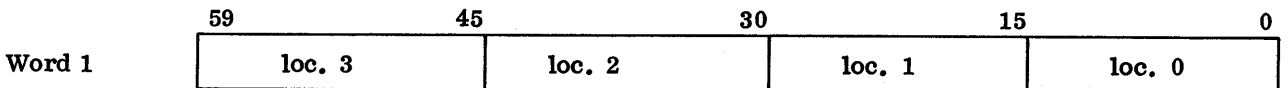
a columns in matrix
 b rows in matrix
 c location of matrix

For a matrix defined as MXN , item (i, j) is at location $N*(i-1)+(j-1)$
 For fullword and floating point matrices this is the location in the array

Halfword matrices from the preceding linear function are stored as follows:



Byte matrices from the preceding linear function are stored as follows:



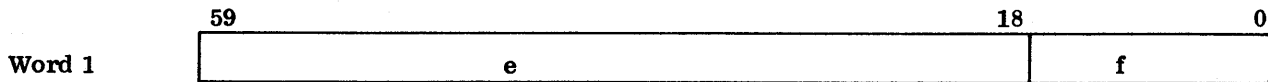
GROUP ALLOCATION



- a = 0 indicates group not used
= 1 indicates group used
- b = 0 indicates numeric group
= 1 indicates transaction group
- c number of items in group
- d link to chain of group items

For each item in group, 1 word of storage is requested and returned when the item is removed from the group.

For numeric groups:



- e value entered in group (integer)
- f next entry in group or zero

For transaction groups:



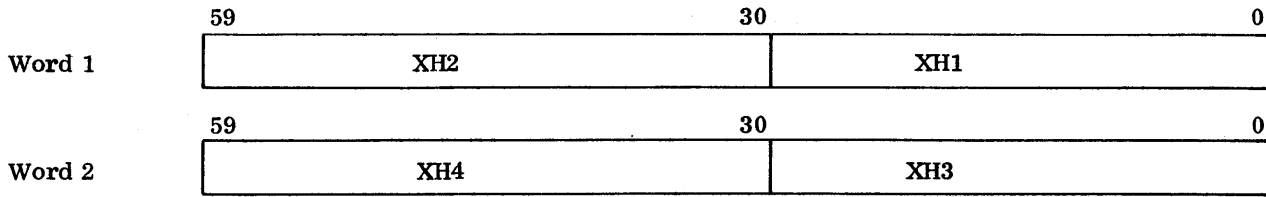
- g transaction location
- h next entry in group or zero

FULLWORD SAVEVALUES AND FLOATING POINT SAVEVALUES

One word per savevalue. (60 bit signed 6000 integer or floating point number.) Truncated to 48 bits for multiply, divide and several other operations.

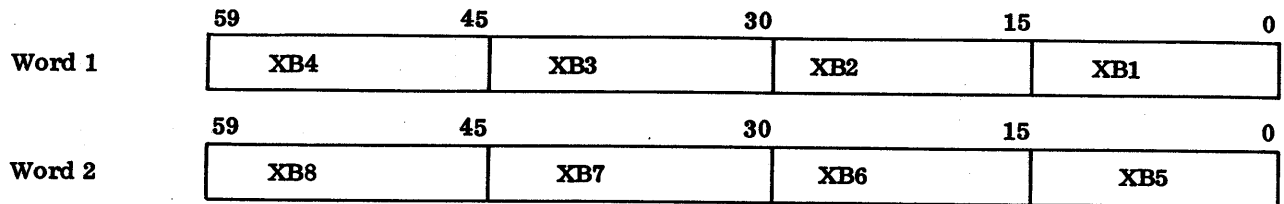
HALFWORD SAVEVALUES

30 bit signed integers stored 2 per 6000 word.

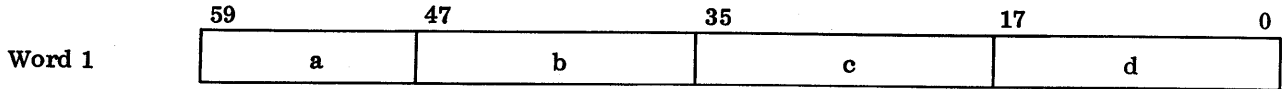


BYTE SAVEVALUES

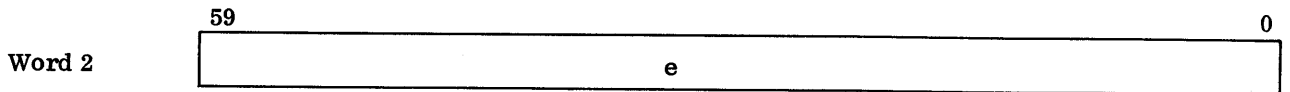
15 bit signed integers stored 4 per 6000 word.



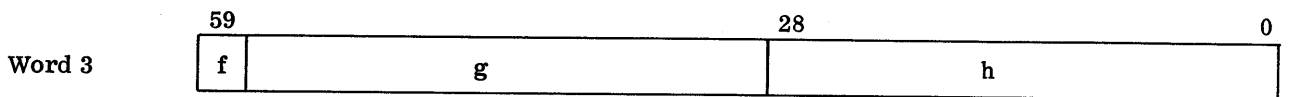
CHAIN ALLOCATION



- a maximum number of entries on chain
- b current number of entries on chain
- c backward link of chain of transactions
- d forward link of chain of transactions




- e cumulative time integral (floating point update every time a transaction is added or deleted from chain) increment = current number of items * length of time



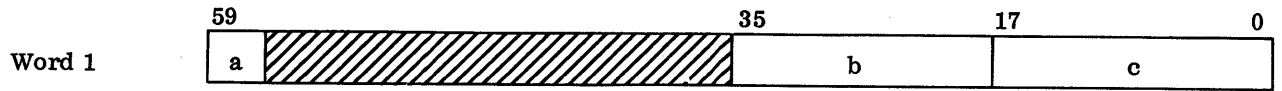
- f link indicator, 1 = on
- g last clock time cumulative word was updated
- h total count of items on chain

The user chain is a doubly linked list starting at (and including) the chain table entry using the FEC pointers in the transactions.

QUEUE ALLOCATION

Word 1	59	29	0
	Total Contents	Last time cumulative time integral updated	
Word 2	59	29	0
	Current Contents	Number of zero entries	
Word 3	59		0
	Cumulative time integral (floating)		
Word 4	59	47	17
		Maximum Contents	QTABLE location (if any)

LOGIC SWITCH TABLE ENTRIES



- a = 1 if switch is set
= 0 if switch is reset
- b switch reset delay chain or zero
- c switch set delay chain or zero

00

0

0

0

0

0

GPSS V PROGRAM COMPATIBILITIES

B I

GPSS V/6000 is intended to be compatible with IBM Program Product 5734-XS2, General Purpose Simulation System V. This appendix describes the differences between the two systems and clarifies various ambiguities in the documentation of the IBM implementation of GPSS V. In general many of the differences between the two programs stem from the removal of various restrictions imposed on IBM's GPSS V to allow compatibility with GPSS III/6000.

These notes are to be used with the IBM publication SH20-0851-1, General Purpose Simulation System V User's Manual. All page numbers quoted refer to that publication.

The minimum configuration for running GPSS V/6000 is a 6000 series computer and a NOS, NOS/BE, or SCOPE 3.4 operating system. The program will assemble and run in 32K and it will use ECS if available. GPSS V/6000 is a one pass assembler. Any mention of a second pass in the IBM manual should be ignored. Input cards are executed as they are read. Because of this a slightly different scheme is used in assigning entity numbers to entity symbols. See ENTITY NUMBER ASSIGNMENT in Chapter 6 above in this manual.

(Page 12) GPSS V/6000 has four data modes for transaction parameters, savevalues and matrix savevalues.

MODE	SIZE (bits)	SIG. DIGS. ¹	RANGE	SNAs
Fullword ²	60	18	$\pm(2^{59}-1)$	PF, XF, X, MX
Halfword	30	9	$\pm(2^{29}-1)$	PH, XH, H, MH
Byte	15	4	$\pm(2^{14}-1)$	PB, XB, MB
Float. Pt.	60 ³	15	$\pm 10^{320}$	PL, XL, ML

1. Number of significant digits is approximate.
2. Only the low order 48 bits are used in multiplication and division as in CDC FORTRAN.
3. One sign bit, eleven bits for the exponent and 48 bits for the mantissa.

GPSS V allows the use of the SNA P for compatibility with GPSS III/6000. The following scheme is used to determine the mode of a parameter when it is used as a block operand and its mode has not been specified.

1. If the entering transaction has any fullword parameters, then P will be interpreted as PF, a fullword parameter.
2. If the transaction has no F-parameters but it does have halfword parameters, the mode of P will be H (halfword).
3. If the transaction has no PH's but it has one or more PB's, P will be interpreted as a byte parameter.
4. Otherwise P will be assumed to be PL, a floating point parameter.

Note that each transaction that enters the block having the unspecified parameter as an operand will have to make the parameter-type decision again. This could conceivably mean that P would be interpreted as a different type of parameter for different transactions.

In addition to the use of P, GPSS V/6000 allows the naming of parameters (as in GPSS III/6000). The rules for parameter naming are the same as those for any SNA. However, since numbers are assigned to parameters in sequence regardless of type, users would be best advised to use the EQU statement for assigning numbers to named parameters.

(Page 15) GPSS V/6000 has only one chain for interrupted transactions and those waiting for assembly set MATCH. The IBM version has two separate chains for this purpose.

(Page 18) The two coding formats, fixed and free field, are available in GPSS V/6000. Two pseudo-operations are provided to enable the user to switch between the two formats within the same program. To go from free-field to fixed-field one would use the FIXED card (with the command FIXED appearing anywhere in columns 2-71). To go in the other direction one would use the FREE card.

NOTE: The word FREE must appear starting in column 8.

In addition to the FIXED and FREE cards, the user may specify the input card format by the parameters FR(free) and FX(fixed) on the GPSS program call card. If neither parameter is specified, the default format is free-field.

Both the REPORT GENERATOR and UPDATE processor must be in fixed-field format.

When the operand field on a card is optional or when no operands are required, comments may still be entered on the card by preceding them with a semi-colon(;) or by starting the comment in or after column 21.

All references to card columns 73-80 in the IBM GPSS V User's Manual should read 73-90 for GPSS V/6000, so that the CDC UPDATE program may be used. Card column 72 should always be left blank.

(Page 21) For compatibility with GPSS III, the following forms are allowed:

SNA*name
SNA*number

In these cases it will be assumed that the 'name' or 'number' refers to a parameter. Note that in neither case is the data mode of the parameter specified. In general, wherever the parameter suffix is not specified, the parameter scheme described earlier is used.

There are no restrictions in the use of indirect addressing.

(Page 23) If suffixes are not entered on SAVEVALUE blocks and in general where the SNA X is used, fullword integer savevalues are understood.

(Page 28) The following SNA's are also accepted by GPSS V/6000:

Kn	n is a numeric positive integer constant, $n \leq 2^{17}$
Lj	Logical status of Logic Switch j (0=Reset, 1=Set)
Pj	Value of Parameter j (data mode is assigned according to scheme described earlier)
Xj	Contents of Fullword Savevalue j
Hj	Contents of Halfword Savevalue j

(Page 30) Integer constants can be in the range $\pm(2^{59}-1)$, and floating point constants are in the approximate range $\pm 10^{310}$. Only the first 14 digits of a constant will be used, although GPSS V/6000 will accept longer constant specifications.

(Page 32) The three allocation schemes for GPSS V/6000 (R1, R2 and R3 on the GPSS program call card) are identical to the IBM specifications except that there are no restrictions on the number of transactions or MACRO definitions. The COMMON area allocation for IBM is irrelevant.

The notes to Table 3-1 are to be replaced by the following table:

Entity	Standard Storage Allocation
Transaction	5 words + 1 word for each PF + 1/2 word for each PH + 1/4 word for each PB + 1 word for each PL
Block	2 words (if less than 3 operand fields) or 1 word + 1 word for each 2 operand fields after the first
Facility	5 words + 1 word for each transaction preempted on priority
Storage	7 words
Queue	4 words + 1 word for each transaction in the queue
Logic Switch	1 word
Table	10 words + 1 word for each frequency class
Function	(C, D, E Type) 2 words + 2 words per point (L, M Type) 2 words + 1 word per point
Variables	2 words + 1 word for each operator or operand
Boolean Var.	2 words + 1 word for each operator or operand
Savevalues	(Fullword) 1 word (Halfword) 1/2 word (Byte) 1/4 word (Floating Pt.) 1 word
Chain	3 words
Group	1 word + 1 word for each group member
Matrix Savevalues	(Fullword) 1 word + 1 word for each matrix entry (Halfword) 1 word + 1/2 word for each matrix entry (Byte) 1 word + 1/4 word for each matrix entry (Floating Pt.) 1 word + 1 word for each matrix entry
MACRO	8 words per line (when stored) 11 words + parameter storage (when expanded)

(Page 34) Modulo division is denoted by a double slash // or @ .

The SNA's XL, PL and ML may be used in fixed-point VARIABLE definitions. They are truncated and treated as integers when the VARIABLE is evaluated.

(Page 38) The following logical operators are also defined for use in BOOLEAN VARIABLES:

FVj 1 if facility j available, otherwise 0.
 FNVj 1 if facility j unavailable, otherwise 0.
 SVj 1 if storage j available, otherwise 0.
 SNVj 1 if storage j unavailable, otherwise 0.

(Page 47) The Y-values on E (discrete attribute) and M (list attribute) functions may be matrix save-values.

(Page 59) GPSS V/6000 had eight pseudo-random number generators which are specified as SNA's RN1 through RN8. Each produces the same sequence of uniformly distributed numbers. The eight generators use the multiplicative congruential method in which each number is generated from the previous value in the sequence (the seed) by

$$U_{n+1} = MU_n \pmod{2^{48}}$$

where M is a multiplier chosen for its good statistical properties.

The seed value of any of the generators may be changed by using the RMULT card. The multiplier may not be changed.

GPSS V/6000 also has a ninth random number generator which will read numbers from a user specified file.

(Page 61) No more than 4095 transactions may reside at any single block.

(Page 65) An EXECUTE block may not execute a second EXECUTE block.

(Page 69) When referring to the mark time SNA which is denoted by MPjPx, the x may be an F (fullword), H (halfword) or B (byte), but it may not be L (floating point).

(Page 87) GPSS V/6000 supports the GPSS III version of the GENERATE block in which field F is interpreted as the number of parameters and field G contains an F, H or blank. This card can be distinguished from the GPSS V version because field F will have no alphanumeric suffix.

(Page 94) No warning messages are issued when modulo truncation occurs.

(Page 99) The MARK block also allows the B-type parameter along with F and H; however, the L-type (floating point) parameter may not be used.

(Page 109) The TRANSFER P statement is also allowed when using the parameter selection mode of the TRANSFER block. The standard parameter default scheme is used. For example,

If there are F-parameters, P becomes type F
If no F, try H.
If no H, try B.

If the default scheme is forced to treat P as a floating point (L) parameter, an execution error will result.

(Page 135) PLj may also be used as the LINK block criterion with the truncated integer value of the parameter being used. This is not recommended, as incorrect linking may result due to the truncation. PLj may also be used in field D of the UNLINK block (Page 137) with the truncated value being used here too.

(Page 162) Fields B through G of the HELPB block should be coded with savevalue prefixes instead of suffixes. Thus GPSS V/6000 accepts

```
HELPB ALPHA,XL1,XF$BETA,XL3
```

instead of

```
HELPB ALPHA,1XL,BETA$XF,3XL
```

(Page 168) There are no PL/I HELP blocks in GPSS V/6000.

(Page 218) For the PREEMPT block, case 3b at the top of page 218 and case 3D.2 on page 220 may be eliminated because these conditions are not identified by GPSS V/6000. Similarly Note 2 on page 230 may be ignored.

(Page 279) Only the first line of the TRACE output is printed. The transaction parameter values are not listed nor are the current and future event chain values printed for the traced transaction.

(Page 286) The column labelled ADV BLK DEPART will only contain meaningful information if the transaction is on the future events chain.

(Page 300) Multiple consecutive rows of matrix savevalues are not deleted when printing matrices.

(Page 302) The following additions should be made to the list of GPSS V Control Statements:

17. SPACE statement - insert one or more black lines in the program listing. The SPACE card is not printed.
18. EJECT statement - perform a skip-to-new page in the program listing. The EJECT card is not printed.

These cards are also used in the REPORT GENERATOR (see page 369).

(Page 302) The START card in GPSS V/6000 has an E-field for specifying a report name defined by the report generator routine. The named report will replace the standard statistical printout. If the E-field is left blank and there is an unnamed ("blank") report defined for the current run, the "blank" report will replace the standard output.

(Page 309) The RMULT card is used to redefine or reinitialize some or all of the random number generator seeds. If an operand placed in fields A-H is even, an error message will be issued. A thirty-seven (37) in any field will restore the original seed for that generator.

Any other odd integer number (up to 14 digits) placed in a field will be used to generate a new seed for the random number generator associated with that field. The low order 48 bits will be used as the mantissa of the new seed and the exponent will always be packed with 1717 (octal).

(Page 311) The JOB card in GPSS V/6000 has an A-field for entering titles on the program listing. The first 50 columns of the A-field are used as the title and are printed as part of all page headings.

Since the JOB card destroys any previous model definitions, care should be taken when using the JOB card with JOBTAPES and MACRO definitions since these will be lost after the JOB card is encountered. It is advisable to use the JOB card at the beginning of a model only and to combine multiple JOBs only when the different models do not make use of JOBTAPES and MACROs.

(Page 312) JOBTAPE transactions enter the model at time zero if the transaction offset time is left blank or is zero.

(Page 313) The GPSS V/6000 UNLIST card has no ABS option.

(Page 317) Field A on the SIMULATE card is interpreted as the number of CP minutes to be added to the current time to determine the new time limit for ending program execution. For example,

SIMULATE 2.5

would allow the simulation to run for an additional 150 CP seconds. Any previously defined time limit will no longer apply.

The B-field on the START card is not available.

When the time limit is reached the following may occur:

<u>Job is currently in</u>	<u>Action taken</u>
UPDATE or ASSEMBLY EXECUTION	Update or Assembly completed, but no final printout is made. Execution stops and the standard statistical printout is made.
OUTPUT	Output file is printed.
REPORT GENERATOR	Report generation stops and the standard statistical printout is made.

Users should be careful to make sure that the SBU limit set for the entire job (for example, by a SBULIM control card) is greater than the limit set by the A field of the SIMULATE card since no output is printed if the system SBU limit is exceeded.

(Page 318) The LOAD card is not available in GPSS V/6000.

(Page 319) The NOXREF card stops the collection of cross-references, not the printing of them.

(Page 320) The XAC, MAC, and COM mnemonics are irrelevant and therefore ignored on the REALLOCATE card. The allocation count (B-field) for any other entity may not exceed 4095.

REALLOCATE should be used carefully when JOBTAPES or MACROs are present, since they will be lost after a REALLOCATE. This card should only appear at the start of the model to be safe.

(Page 323) There is no AUXILIARY card in GPSS V/6000.

(Page 342) Since parameters may be named, they can appear on an EQU card. The letter P is used. For example:

GAMMA EQU 3,P

would mean that the name GAMMA would refer to the third parameter regardless of type (F, H, B or L).

To alleviate the problem described in the first paragraph on page 343, GPSS V/6000 assigns a typeless numeric value to the name on an EQU card. The name will retain the first numeric value it is assigned. Thus the following case no longer presents any ambiguity:

```
LINE1 EQU 2,F
LINE1 EQU 5,Q
      :
      :
ASSIGN 6,LINE1,PH
      :
      :
QUEUE PH6
SEIZE PH6
```

Here queue 2 would be entered and facility 2 seized. This is, of course, not recommended as a standard programming technique.

(Page 360) To obtain the services of the GPSS output editor, the requests must precede the START card.

See Appendix H.

JOBTAPE FORMAT

C

Each transaction is written on the file as 1 logical record of level zero. The first 2 words of each record contain transaction information, and words 3-n contain the parameters in a packed format, together with the parameter control word.

When GPSS V/6000 writes to jobtapes by a WRITE card, empty records are inserted between transactions. GPSS V/6000 expects the jobtapes to be in this format when a JOBTAPE card is executed. Jobtapes are written with S type records. FORTRAN programs that read or write these tapes must take these two facts into consideration.

TRANSACTION 1

⋮

-EOR-

-EOR-

TRANSACTION 2

⋮

-EOR-

-EOR-

TRANSACTION 3

⋮

-EOR-

-EOR-

⋮

WORD 1

Bits 59-30

Bits 29-0

Transit time

Interarrival time

WORD 2

Bit 59

Bits 58-49

Bits 48-37

Bits 36-30

Bits 29-0

Not used

Number of words in record excluding
first 2

Not used

Priority of transaction

Not used

WORD 3 - Parameter Control

Bits 59-51	Not used
Bits 50-42	Floating point parameters relative location
Bits 41-33	Byte parameter's relative location
Bits 32	Not used
Bits 31-24	Number of floating point parameters
Bits 23-16	Number of byte parameters
Bits 15-8	Number of halfword parameters
Bits 7-0	Number of fullword parameters

Assume a - Fullword
 b - Halfword
 c - Byte
 d - Floating point

WORDS 4 → 3 + a

60 bit 6000 integers
 The fullword parameters in order

$$\text{Words } 4+a \rightarrow 4+a+\frac{(b-1)}{2}$$

The half word parameters packed 2 per word

59	30	29	0
2	1		
4	3		
6	5		
⋮			

$$\text{Words } 4+a+\frac{(b-1)}{2}+1 \rightarrow 4+a+\frac{(b-1)}{2}+1+\frac{(c-1)}{4}$$

The byte parameters packed 4 per word

59	44	29	14	0
4	3	2	1	
8	7	6	5	
12	11	10	9	
16	15	14	13	

NOTE: The byte offset $a+\frac{(b-1)}{2}+1$ is stored in the parameter control word

$$\text{Words } 4+a+\frac{(b-1)}{2}+1+\frac{(c-1)}{4}+1 \rightarrow$$

$$4+a+\frac{(b-1)}{2}+1+\frac{(c-1)}{4}+1+(d-1)$$

The floating point parameters stored as 6000 floating point numbers in order

NOTE: The floating point offset $a + \frac{(b-1)}{2} + 1 + \frac{(c-1)}{4} + 1$ is stored in the parameter control word

If the transaction does not contain any parameters of a certain type the no storage is assigned to them

Examples: 4 - fullword
2 - byte

4	PF1	
5	PF2	
6	PF3	
7	PF4	
8	N. U.	PB2 PB1

5 - halfword
2 - floating point

4	PH2	PH1
5	PH4	PH3
6	N. U.	PH5
7	PL1	
8	PL2	

3 - floating point

4	PL1
5	PL2
6	PL3

O

O

O

O

O

O

O

REPORT GENERATOR STATEMENTS

D I

TABLE D-1. SUMMARY OF STATEMENTS FOR NONGRAPHIC OUTPUT

Location	Operation	Operands	Continuation Indication
<p>Not used</p> <p>Entity type FAC STO QUE TAB FSV HSV BSV LSV FMS HMS BMS LMS CHA GRP BLO CLO</p> <p>Entity type (See TITLE)</p> <p>Starting print position (Default is 1)</p>	<p>REPORT</p> <p>TITLE</p> <p>INCLUDE</p> <p>FORMAT</p>	<p>Not used</p> <p>A, B A: Identification of entity member to which title (B) applies. If n not specified, title applies to all statistics of this type. Operand A not applicable to entity type CLO and BLO. All of these types are obtained. B: Text</p> <p>A/B A: Range - numeric or symbolic Example: F1-F5 or F\$ONE-F\$FIVE Allowable mnemonics are: F, S, Q, T, CH, X, XH, XB, XL, MX, MH, MB, ML, G B: Specific columns - numeric only, separated by commas.* (This operand applicable only to F, S, Q, T, CH.)</p> <p>A/B A: Range of entity numbers - numeric only B: Output statistics-successive subfields of form Xn (X is entity code, n is column identification)* 18 print positions per type of statistics</p>	<p>Nonblank for continuation in position 1-71 of next statement.</p>

TABLE D-1. SUMMARY OF STATEMENTS FOR NONGRAPHIC OUTPUT (CONT'D)

Location	Operation	Operands	Continuation Indication
<p>Starting print position Default: position 1</p> <p>* in position 1</p>	TEXT	<p>A</p> <p>A: Intermixed text and data specifications Data Specification: X:entity mnemonic* n:entity number or symbol m:column identification*</p> <p>Format has 3 parts:</p> <ol style="list-style-type: none"> 1. Number of spaces decimal point to be moved. 2. L or R for direction decimal point to be moved. 3. XX.XX for format of data <p><u>Example:</u> #F1,2/2RXX.XX# means average utilization of facility 1, to be multiplied by 100 and printed in format XX.XX</p> <p>Comment statement positions 2-71 printed starting in position 1.</p>	<p>Nonblank for continuation in position 1-71 of next statement.</p> <p>Nonblank for continuation in position 1-71 of next statement</p>
	EJECT		
	SPACE	<p>A</p> <p>A: Number of lines to space: 1, 2, or 3.</p>	
OUTPUT			

* See Table D-2.

TABLE D-2. ENTRY CODES AND COLUMN NUMBERS FOR USE ON INCLUDE, FORMAT, AND TEXT STATEMENTS

Entity Type	Code	Column	Definition	Include	Format	Text
FACILITIES	F	1	Facility name or number	*	*	
		2	Average utilization	*	*	*
		3	Number of entries	*	*	*
		4	Average time per transaction	*	*	*
		5	Seizing transaction number	*		
		6	Preempting transaction number	*		
STORAGES	S	1	Storage name or number	*	*	
		2	Capacity	*		
		3	Average contents	*	*	*
		4	Average utilization	*	*	*
		5	Number of entries	*	*	*
		6	Average time per transaction	*	*	*
		7	Current contents	*	*	*
		8	Maximum contents	*	*	*
QUEUES	Q	1	Queue name or number	*	*	
		2	Maximum contents	*	*	*
		3	Average contents	*	*	*
		4	Total entries	*	*	*
		5	Zero entries	*	*	*
		6	Percent zero entries	*	*	*
		7	Average time per transaction	*	*	*
		8	Average time per transaction, excluding zero entries	*	*	*
		9	Table number associated with queue	*		
		10	Current contents of queue	*	*	*

*Indicates in which statement the code is legal

TABLE D-2. ENTRY CODES AND COLUMN NUMBERS FOR USE ON INCLUDE, FORMAT, AND TEXT STATEMENTS (CONT'D)

Entity Type	Code	Column	Definition	Include	Format	Text
TABLES	T	1	Table name or number	*	*	
		2	Entries in table -- nonweighted	*	*	*
		3	Mean argument -- nonweighted	*	*	*
		4	Standard deviation -- non- weighted	*	*	*
		5	Sum of argu- ments -- nonweighted	*	*	
		6	Entries in table -- weighted	*	*	*
		7	Mean argu- ment -- weighted	*	*	*
		8	Standard deviation -- weighted	*	*	*
		9	Sum of argu- ments -- weighted	*		
		10	Upper limit	*		
		11	Observed frequency	*		
		12	Percent of total	*		
		13	Cumulative percentage	*		
		14	Cumulative remainder	*		
		15	Multiple of mean	*		
		16	Deviation from mean	*		

*Indicates in which statement the code is legal

TABLE D-2. ENTRY CODES AND COLUMN NUMBERS FOR USE ON INCLUDE, FORMAT, AND TEXT STATEMENTS (CONT'D)

Entity Type	Code	Column	Definition	Include	Format	Text
USER CHAINS	CH	1	User chain name or number	*	*	
		2	Total entries	*	*	*
		3	Average time per transaction	*	*	*
		4	Current contents	*	*	*
		5	Average contents	*	*	*
		6	Maximum contents	*	*	*
SAVEVALUES	X	1	Savevalue name or number		*	
		2	Contents of savevalue		*	*
	XH	1	Halfword savevalue name or number		*	
		2	Contents of halfword savevalue		*	*
	XB	1	Byte savevalue name or number		*	*
		2	Contents of savevalue		*	*
	XL	1	Floating-point savevalue name or number		*	
		2	Contents of savevalue		*	*

*Indicates in which statement the code is legal

To obtain a graph, the following statements, which must appear in the order shown, are needed:

```

GRAPH
ORIGIN
X
Y
STATEMENT (As many as needed. These are optional)
ENDGRAPH

```

The output editor statements needed to prepare graphical output for the following SNAs (mnemonics used as A operand of GRAPH statement) are shown in Table D-3.

<u>Entity</u>	<u>SNA</u>	<u>Mnemonic</u>
Facilities	Utilization	FR
	Entry count	FC
	Avg. time/transaction	FT
Storages	Utilization	SR
	Avg. contents	SA
	Current contents	S
	Max. contents	SM
	Entry count	SC
	Avg. time/transaction	ST
Queues	Avg. contents	QA
	Current contents	Q
	Max. contents	QM
	Entry count	QC
	Zero entries	QZ
	Avg. time/transaction	QT
	Avg. time/transaction (excluding zero-delay entries)	QX
Savevalues	Fullword contents	X
	Halfword contents	XH
	Byte contents	XB
	Floating-point contents	XL
Tables	Entry count	TC
	Mean	TB
	Standard deviation	TS
Blocks	Block count	N
Groups	Current contents of group	G
User chains	User chain average contents	CA
	User chain current contents	CH
	User chain maximum contents	CM
	User chain entry count	CC
	User chain average time/transaction	CT

TABLE D-3. GRAPHICAL OUTPUT STATEMENTS FOR CERTAIN SNA'S EXCLUDING TF, TP, TD, AND TR

Location	Operation	Operands	Continuation Indication
	GRAPH	A, B, C, D A: SNA identification B: Lower limit of entity range (ranging from low to high) Numeric or symbolic range C: Upper limit of entity range D: Plotting character (Default is *.)	
	ORIGIN	A, B A: Row (from 1 to 60) B: Column (from 1 to 32)	
	X	A, B, C, , , , G A: SYM - entity symbols are used to label x-axis blank - entity numbers used B: Number of print positions for width of each bar of graph C: Number of print positions between bars of graph (B+C≥4) G: NO - no labeling for x-axis blank - x-axis is labeled	
	Y	A, B, C, D (all numeric) A: Value for y at origin B: Size of y-axis increment C: Total number of increments for y-axis (s*n=minimum value of SNA) D: Number of print lines per increment. (C*D is ≤ A operand of ORIGIN statement)	
Print pos. where statement should start (default is 1)	STATEMENT	A, B, C A: Row of graph (1-60) on which statement is to be printed B: Number of characters in text which follows C: Text	1 if continued in pos. 1-71 of next statement
NOTE: STATEMENT statements must be in ascending order by row. Any number of STATEMENTS may be used with each graph.			
		ENDGRAPH	

The output editor request statements needed to prepare graphical output for the following SNAs (mnemonics used as A operand of GRAPH statement) are shown in Table D-4.


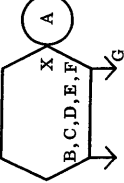
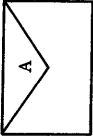
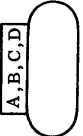

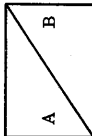

<u>Entity</u>	<u>SNA</u>	<u>Mnemonic</u>
Tables	Observed frequencies	TF
	Percent of total	TP
	Cumulative percentage	TD
	Cumulative remainder	TR

TABLE D-4. GRAPHICAL OUTPUT STATEMENTS FOR TF, TP, TD, AND TR

Location	Operation	Operands	Continuation Indication
	GRAPH	A, B C, D A: SNA identification B: Numeric or symbolic table identification. C: Not used D: Plotting character (Default is *.)	
	ORIGIN	A, B A: Row from 1 to 60 B: Column from 1 to 132	
	X	, B, C, D, E, F, G B: (TF and TP) - Number of print positions for width of each bar of graph (TD and TR) - Must be blank C: (TF and TP) - Number of print positions between bars of graph (TD and TR) - Number of print positions between points D: Upper limit of first class to be plotted E: Number of frequency classes per X-axis increment (width of X-axis increment given in C) F: Number of increments to be plotted (E*F = number of frequency classes to be plotted) G: NO - no labeling for X-axis blank - X-axis is labeled	
	Y	A, B, C, D (See explanation of Y statement in Table D-3.)	
	STATEMENT	See Table D-3.	
	ENDGRAPH	See Table D-3.	

BLOCK STATEMENT FORMATS

TABLE E-1. BLOCK STATEMENT FORMATS

Operation	A	B	C	D	E	F	G	H	I	BLOCK SYMBOL
ADVANCE	Mean time [k, SNA], [SNA*SNA]	Spread [k, SNA], [SNA*SNA] or Function Modifier FN], FN*SNA]								
ALTER G GE L LE E NE MIN MAX	Group no. k, SNA], SNA*SNA]	Count ALL or k, SNA], SNA*SNA]	Member attribute to be altered PR or kPx, SNA]Px, SNA*SNA]Px	Value to replace attribute k, SNA], SNA*SNA]	Matching member attribute PR or kPx, SNA]Px, SNA*SNA]Px	Matching SNA [k, SNA], [SNA*SNA]	Alternate exit [k, SNA], [SNA*SNA]			
ASSEMBLE	No. of transactions to assemble k, SNA], SNA*SNA]									
ASSIGN	Parameter no. or range k, SNA], SNA*SNA] [±]	SNA value to be assigned k, SNA], SNA*SNA]	No. of function modifier [k, SNA], [SNA*SNA]	Parameter type Px						
BUFFER										
CHANGE	"From" block no. k, SNA], SNA*SNA]	"To" block no. k, SNA], SNA*SNA]	Upper limit k, SNA], SNA*SNA]	Comparison value if conditional operator is specified [k, SNA], [SNA*SNA]	Mnemonic of SNA to be counted Any SNA except MX, MH, MB, ML					
COUNT G, GE L, LE E, NE U, NU I, NI SNE, SE SNF, SF LR, LS	Parameter in which to place count kPx, SNA]Px SNA*SNA]Px	Lower limit k, SNA], SNA*SNA]								

Note: The parameter type operand may optionally be coded at the C operand if a function modifier is not specified.

† Block operand where PL, XL, or ML is a valid SNA.
 [] Indicates optional operand
 { } Indicates that one of the items within
 { } the braces must be selected

TABLE E-1. BLOCK STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H	I	Block Symbol
DEPART	Queue no. k, SNAJ, SNA*SNAJ	No. of units [k, SNAJ,] [SNA*SNAJ]								
DISMISS										
ENTER	Storage no. k, SNAJ SNA*SNAJ	No. of units [k, SNAJ,] [SNA*SNAJ]								
EXAMINE	Group no. k, SNAJ, SNA*SNAJ	Numeric value- numeric mode [k, SNAJ,] [SNA*SNAJ]	Alternate exit k, SNAJ, SNA*SNAJ							
EXECUTE	Block no. k, SNAJ, SNA*SNAJ									
FAVAIL	Facility no. or range k, SNAJ, SNA*SNAJ									
FUNAVAIL	Facility no. or range k, SNAJ, SNA*SNAJ	Remove or continue option [RE] [CO]	Alternate block no. [k, SNAJ,] [SNA*SNAJ]	Parameter no. [kPx,] [SNAjPx,] [SNA*SNAjPx]	Remove or continue option [RE] [CO]	Alternate block no. [k, SNAJ,] [SNA*SNAJ]	Remove or continue option [RE] [CO]	Alternate block no. [k, SNAJ,] [SNA*SNAJ]		

[] Indicates optional operand

TABLE E-1. BLOCK STATEMENT FORMATS

Operation	A	B	C	D	E	F	G	H	I	Block Symbol
GATE { LS } { LR }	Logic switch no. k, SNAj, SNA*SNAj	Next block if condition is false [k, SNAj,] [SNA*SNAj]								
GATE { NI } { I } { NU } { U } { FV } { FNV }	Facility no. k, SNAj, SNA*SNAj	Next block if condition is false [k, SNAj,] [SNA*SNAj]								
GATE { SE } { SF } { SNE } { SNF } { SV } { SNV }	Storage no. k, SNAj, SNA*SNAj	Next block if condition is false [k, SNAj,] [SNA*SNAj]								
GATE { M } { NM }	Match block no. k, SNAj, SNA*SNAj	Next block if condition is false [k, SNAj,] [SNA*SNAj]								
GATHER	No. of trans- actions to be gathered k, SNAj, SNA*SNAj									

[] Indicates optional operand

{ } Indicates that one of the items within
the braces must be selected

TABLE E-1. BLOCK STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H	I	Block Symbol
GENERATE	Mean time [k, SNA], [SNA*SNA]	Spread [k, SNA], [SNA*SNA] or Function modifier FNj, [FN*SNA]	Initialization interval [k, SNA], [SNA*SNA]	Creation limit [k, SNA], [SNA*SNA]	Priority level [k, SNA], [SNA*SNA]	Fullword, halfword, byte & floating point in any sequence [kPx, SNA]Px, [SNA*SNA]Px	[kPx, SNA]Px, [SNA*SNA]Px	[kPx, SNA]Px, [SNA*SNA]Px	[kPx, SNA]Px, [SNA*SNA]Px	
GO TO	Next block k, SNA] SNA*SNA]									
HELP HALPA HELPA HELPC	Help routine name	B-G operands SNA values to be passed to help routine [k, SNA], [SNA*SNA]								
INDEX	Para meter no. kPx, SNA]Px, SNA*SNA]Px	Increment k, SNA], SNA*SNA]								
JOIN	Group) no. k, SNA], SNA*SNA]	Numeric value- numeric mode [k, SNA], [SNA*SNA]								
LEAVE	Storage no. k, SNA], SNA*SNA]	No. of units k, SNA], SNA*SNA]								

Note: Operands A-I may be a constant, FNj, Vj, Xj, XFj, XBJ, XHj, RNj, Cl, AC1, or Nj. Likewise, elements of functions or variables specified are restricted to these SNAs.

When using HELPB the B-G operands reference either fullword or floating-point savevalues. An XL (floating-point) or XF (fullword) should be used with each of these operands.

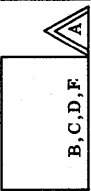
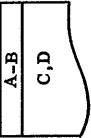
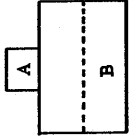

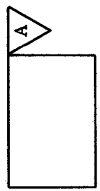
[] Indicates optional operand

TABLE E-1. BLOCK STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H	I	Block Symbol
LINK	User chain no. k, SNAj, SNA*SNAj	Ordering of chain LIFO, FIFO or parameter no. kPx, SNAjPx, SNA*SNAjPx	Alternate block exit [k, SNAj, SNA*SNAj]							
LOGIC { S } { R } { I }	Logic switch no. k, SNAj, SNA*SNAj									
LOOP	Parameter no. kPx, SNAjPx, SNA*SNAjPx	Next block if Pxj ≠ 0 k, SNAj, SNA*SNAj								
MARK	Parameter no. [kPx, SNAjPx, SNA*SNAjPx]									
MATCH	Conjugate MATCH block no. k, SNAj, SNA*SNAj									
MSAVEVALUE	Matrix no. or range k, SNAj, SNA*SNAj	Row no. or range k, SNAj, SNA*SNAj	Column no. or range k, SNAj, SNA*SNAj	SNA value to be saved k, SNAj, SNA*SNAj	Msavevalue type H, MH, MX, MB, ML					
PAUSE	MESSAGE									

[] Indicates optional operand
 { } Indicates that one of the items within the braces must be selected

TABLE E-1. BLOCK STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H	I	Block Symbol
PREEMPT	Facility no. k SNAI, SVA*SNAI	Priority option [PRI]	Block no. for pre-empted transaction [k, SNAI] [SNA*SNAI]	Parameter no. of pre-empted transaction [kPx, SNAI]Px, [SNA*SNAI]Px	Remove option [RE]					
PRINT	Lower limit [k SNAI], [SVA*SNAI]	Upper limit [k, SNAI], [SNA*SNAI]	Entity mnemonic	Paging indicator [Any alphanumeric character]						 
PRIORITY	Priority no. k SNAI, SVA*SNAI	Buffer option [BUFFER]								
QUEUE	Queue no. k SNAI, SVA*SNAI	No. of units [k, SNAI], [SNA*SNAI]								
RELEASE	Facility no. k SNAI, SVA*SNAI									

[] Indicates optional operand

TABLE E-1. BLOCK STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H	I	Block Symbol
REMOVE G GE L LE E NE MIN MAX	Group no. k, SNAJ, SNA*SNAJ	Count-no. of members to be re- moved- trans- action mode [k, SNAJ,] [SNA*SNAJ] [ALL]	Numeric value to be removed- numeric mode [k, SNAJ,] [SNA*SNAJ]	Member attribute for comparison- transaction mode [PR, or parameter no. kPx, SNAJPx, SNA*SNAJPx]	Comparison SNA [k, SNAJ,] [SNA*SNAJ]	Alternate exit-entering Xact [k, SNAJ,] [SNA*SNAJ]				
RETURN	Facility no. k, SNAJ, SNA*SNAJ									
SAVAIL	Storage no. or range k, SNAJ, SNA*SNAJ									
SAVEVALUE	Savevalue no. or range k, SNAJ, SNA*SNAJ [+]	SNA value to be saved k, SNAJ, SNA*SNAJ	Savevalue type X, XF, H, XH, XB, XL							
SCAN G GE L LE E NE MIN MAX	Group no. k, SNAJ, SNA*SNAJ	Member attribute for com- parison PR or parameter no. kPx, SNAJPx, SNA*SNAJPx	Comparison value for B operand k, SNAJ, SNA*SNAJ	Member attribute to be obtained if match is made [PR or parameter no. kPx, SNAJPx, SNA*SNAJPx]	Entering Xact parameter no. in which to place D oper- and value [kPx, SNAJPx, SNA*SNAJPx]	Alternate exit [k, SNAJ,] [SNA*SNAJ]				

[] Indicates optional operand

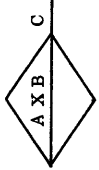

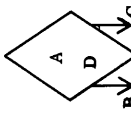
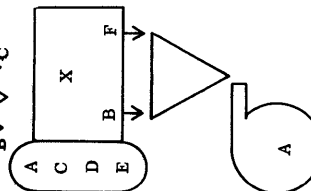
TABLE E-1. BLOCK STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H	I	Block Symbol
SEIZE	Facility no. k, {SNAj, SNA/*SNAj}	Lower limit k, SNAj, SNA/*SNAj	Upper limit k, SNAj, SNA/*SNAj	Comparison value if conditional operator is specified [k, SNAj, SNA/*SNAj]	SNA mnemonic to be examined if conditional operator is specified [Any SNA except MX, MH, MB, ML]	Alternate exit k, SNAj, SNA/*SNAj				
SELECT { G, GE L, LE E, NE U, NU I, NI SE, SNE SF, SNF LR, LS MIN, MAX }	Parameter in which to place entity number kPx, SNAj/Px, SNA/*SNAj/Px									
SPLIT	No. of copies k, {SNAj, SNA/*SNAj}	Next block for copies k, SNAj, SNA/*SNAj	Parameter for serial numbering [kPx, SNAj/Px, SNA/*SNAj/Px]	No. of fullword, halfword, byte & floating-point parameters in any sequence [kPx, SNAj/Px, SNA/*SNAj/Px]						
STOP	[MESSAGE]									
SUNAVAIL	Storage no. or range k, {SNAj, SNA/*SNAj}									
TABULATE	Table no. k, {SNAj, SNA/*SNAj}	Weighting factor [k, SNAj, SNA/*SNAj]								
TERMINATE	Termination count [k, SNAj, SNA/*SNAj]									

[] indicates optional operand

{ } indicates that one of the items within the braces must be selected

TABLE E-1. BLOCK STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H	I	Block Symbol
TEST <div style="display: flex; flex-direction: column; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">E</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">NE</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">GE</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">LE</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">G</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">L</div> </div>	First SNA k, SNAJ, SNA*SNAJ	Second SNA k, SNAJ, SNA*SNAJ	Next block if relation is false [SNAJ, SNAJ]							
TRACE										
TRANSFER	Selection mode	Next block A [k, SNAJ, SNA*SNAJ]	Next block B [k, SNAJ, SNA*SNAJ]	Indexing factor [k]						
UNLINK <div style="display: flex; flex-direction: column; align-items: center;"> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">G</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">GE</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">L</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">LE</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">E</div> <div style="border: 1px solid black; padding: 2px; margin-bottom: 2px;">NE</div> </div>	With ALL selection mode a block name or number is the only valid operand	Next block for the unlinked trans-action(s) k, SNAJ, SNA*SNAJ	Trans-action unlink count ALL or k, SNAJ, SNA*SNAJ	Parameter no. kPx, SNAJPx, SNA*SNAJPx, or BACK or BVj, BV*SNAJ	Match argument [k, SNAJ, SNA*SNAJ]	Next block B [k, SNAJ, SNA*SNAJ]				
UNTRACE										
WRITE	Jobtape no. { JOBT A1 JOBT A2 JOBT A3 }									

[] Indicates optional operand

{ } Indicates that one of the items within the braces must be selected

0

0

0

0

0

0

0

CONTROL STATEMENT FORMATS

TABLE F-1. CONTROL STATEMENT FORMATS

Operation	A	B	C	D	E	F	G	H
BVARIABLE		Combinations of elements, attributes, and operators: Elements k, SNAJ SNA*SNAJ		Logical attributes FUJ or FJ FNUJ FJ FNIJ FVJ FNVJ	Conditional operators 'G' 'L' 'E'	Boolean operators + (or) * (and) - (exclusive OR)		
CLEAR	Savevalues or ranges not to be cleared [X, XF, XH, XB, XL, MX, MH, MB, ML]	Delimiter if multiple entries []						
EJECT								
END								

[] Indicates optional operand

TABLE F-1. CONTROL STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H
FUNCTION	Function argument SNA], SNA*SNA] (any SNA except MX, MH, MB or ML)	Function type and no. of points $\left. \begin{matrix} C \\ D \\ E \\ L \\ M \\ S \end{matrix} \right\}^n$						
$x_1, y_1/x_2, y_2$ /etc.				(Note: Y values cannot be MX, MH, MB, or ML, and X and Y values must start in position 1.)				
INITIAL	Entity or range $\left\{ \begin{matrix} X, XF, XH, \\ XB, XL, MX, \\ MH, MB, ML, \\ LS \end{matrix} \right\}$	Value k	Delimiter if Multiple entries [/]					
JOB	TITLE							
JOBTAPE	Jobtape no. $\left\{ \begin{matrix} JOBT A1 \\ JOBT A2 \\ JOBT A3 \end{matrix} \right\}$	Next block for jobtape trans- actions [k]	Transaction offset time [k]	Scaling factor [k]				
LIST								

[] Indicates optional operand

{ } Indicates that one of the items within
the braces must be selected

TABLE F-1. CONTROL STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H
MATRIX	Matrix type {MX, MH, MB, ML }	No. of matrix rows k	No. of matrix columns k					
NOXREF QTABLE	Queue no. k	Upper limit of lowest fre- quency class k	Frequency class size k	No. of frequency classes k				
READ	No. of files to be skipped [k]							
REALLOCATE	Entity mnemonic to be reallocated	Total no. of that entity k	Delimiter if multiple entries ['']					

[] Indicates optional operand

{ } Indicates that one of the items within
the braces must be selected

TABLE F-1. CONTROL STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H
RESET	Entity or range not to be reset [Fj, Qj, Sj, CHj, TBj]	Delimiter if multiple entries [']						
REWIND	Jobtape no. { JOBT A1 JOBT A2 JOBT A3 }							
RMULT	Initial multiplier for RN1 [k]	Initial multiplier for RN2 [k]	Initial multiplier for RN3 [k]	Initial multiplier for RN4 [k]	Initial multiplier for RN5 [k]	Initial multiplier for RN6 [k]	Initial multiplier for RN7 [k]	Initial multiplier for RN8 [k]
SAVE	Reposition option [Any alphameric character]							
SIMULATE	Max. run length in minutes [k]							
SPACE	Lines to skip							

[] Indicates optional operand

{ } Indicates that one of the items within the braces must be selected

TABLE F-1. CONTROL STATEMENT FORMATS (CONT'D)

Operation	A	B	C	D	E	F	G	H
START	Run termination count k	Printout suppression [NP]	Snap interval [k]	Standard trans- action printout [I]	Report name			
STORAGE	Storage no. or range Sj	Capacity k	Delimiter if multiple entries [/]	No. of frequency classes k	Arrival rate time interval for RT mode table [k]			
TABLE	Table argument k, SNAJ, SNA*SNAJ RT, IA Any SNA except MX, MH, MB, ML, PL, XL SUBTITLE	Upper limit of lowest frequency class k	Frequency class size k					
TITLE								
UNLIST								
VARIABLE								
FVARIABLE		Combinations of elements and arithmetic operators: Elements k, SNAJ, SNA*SNAJ		Arithmetic operators + - / * // (VARIABLE only)				
XREF								

[] Indicates optional operand

00

0

0

0

0

0

DIAGNOSTICS

G I

The following error messages which can appear are catalogued by error number and description.

<u>Error Number</u>	<u>Description</u>
1	COLUMN 7 IS NOT BLANK
2	LOCATION FIELD MUST BE BLANK
3	ILLEGAL SYMBOL IN LOCATION FIELD
4	ILLEGAL SYMBOL IN OPERATION FIELD
5	ILLEGAL G FIELD ON ALTER BLOCK
6	VARIABLE STATEMENT ERROR
7	TOO MANY VARIABLE CARDS
8	TOO MANY FUNCTION CARDS
9	BAD A FIELD IN FUNCTION CARD
10	BAD B FIELD IN FUNCTION CARD
11	BAD FUNCTION TABLE ENTRY
12	ILLEGAL ENTITY FUNCTION MNEMONIC
13	ENTITY NUMBER PREVIOUSLY ASSIGNED
14	ENTITY NAME PREVIOUSLY USED
15	COMMA MUST SEPARATE X AND Y VALUES
16	EQU LOCATION FIELD PREVIOUSLY USED
17	MAXIMUM FIELD LENGTH EXCEEDED
18	EOF ON INPUT FILE
19	BLANK LOCATION FIELD ON EQU OR DEFINED SYMBOL
20	BAD A FIELD ON EQU CARD
21	ILLEGAL FIELD B-G ON EQU CARD
22	ICT CARD ERROR
23	ORG CARD ERROR
24	BAD LOCATION FIELD ON SYN CARD
25	BAD A FIELD ON SYN CARD

Error
Number

Description

26	ILLEGAL MATRIX TYPE SPECIFIED
27	MISMATCHED PARENTHESIS IN VARIABLE CARD
28	ILLEGAL B FIELD IN ADVANCE BLOCK
29	BAD ROW OR COLUMN VALUE ON MATRIX CARD
30	TOO MANY TABLE ENTRIES
31	BAD PARAMETER ON TABLE CARD
32	ILLEGAL C FIELD IN ASSIGN BLOCK
33	ILLEGAL STORAGE CONTENTS
34	BAD STORAGE FIELD A
35	ZERO CONTENTS ON A STORAGE CARD
36	ILLEGAL B FIELD ON START CARD
37	ILLEGAL TYPE ON INITIAL CARD
38	BAD MATRIX NUMBER
39	BAD MATRIX SUBSCRIPT
40	BAD MATRIX VALUE
41	BAD A FIELD ON ASSEMBLE BLOCK
42	EXPECTED SNA - DID NOT FIND ONE
43	BAD A FIELD ON EXECUTE BLOCK
44	BAD A FIELD ON GATHER BLOCK
45	BAD A FIELD ON MATCH BLOCK
46	BAD A FIELD ON RELEASE BLOCK
47	BAD A FIELD ON RETURN BLOCK
48	BAD A FIELD ON SEIZE BLOCK
49	WRITE BLOCK FIELD A ERROR
50	ILLEGAL D FIELD IN ASSIGN BLOCK
51	BAD A FIELD ON CHANGE BLOCK
52	BAD B FIELD ON CHANGE BLOCK
53	BAD A FIELD ON DEPART BLOCK
54	BAD A FIELD ON ENTER BLOCK
55	BAD A FIELD ON INDEX BLOCK
56	BAD B FIELD ON INDEX BLOCK
57	BAD A FIELD ON JOIN BLOCK
58	BAD A FIELD ON LEAVE BLOCK

<u>Error Number</u>	<u>Description</u>
59	BAD A FIELD ON LOGIC BLOCK
60	BAD AUX FIELD ON LOGIC BLOCK
61	BAD A FIELD ON LOOP BLOCK
62	BAD B FIELD ON LOOP BLOCK
63	BAD A FIELD ON PRIORITY BLOCK
64	BAD A FIELD ON QUEUE BLOCK
65	BAD A FIELD ON TABULATE BLOCK
66	ILLEGAL B FIELD IN DEPART BLOCK
67	BAD A FIELD ON ALTER BLOCK
68	BAD B FIELD ON ALTER BLOCK
69	BAD C FIELD ON ALTER BLOCK
70	BAD D FIELD ON ALTER BLOCK
71	BAD E FIELD ON ALTER BLOCK
72	BAD F FIELD ON ALTER BLOCK
73	BAD A FIELD ON ASSIGN BLOCK
74	BAD B FIELD ON ASSIGN BLOCK
75	BAD AUX FIELD ON COUNT BLOCK
76	BAD A FIELD ON COUNT BLOCK
77	BAD B FIELD ON COUNT BLOCK
78	BAD C FIELD ON COUNT BLOCK
79	BAD D FIELD ON COUNT BLOCK
80	BAD E FIELD ON COUNT BLOCK
81	BAD A FIELD ON EXAMINE BLOCK
82	BAD B FIELD ON EXAMINE BLOCK
83	BAD C FIELD ON EXAMINE BLOCK
84	BAD AUX FIELD ON GATE BLOCK
85	BAD A FIELD ON GATE BLOCK
86	BAD A FIELD ON GENERATE BLOCK
87	ILLEGAL D FIELD ON START CARD
88	ILLEGAL E FIELD ON START CARD
89	ILLEGAL USER SYMBOL
90	ILLEGAL F. P. NUMBER
91	ILLEGAL A FIELD IN FAVAIL BLOCK

Error
Number

Description

92 ILLEGAL A FIELD IN FUNAVAIL BLOCK
93 BAD A FIELD ON LINK BLOCK
94 BAD B FIELD ON LINK BLOCK
95 BAD C FIELD ON LINK BLOCK
96 BAD A FIELD ON MSAVEVALUE BLOCK
97 BAD B FIELD ON MSAVEVALUE BLOCK
98 BAD C FIELD ON MSAVEVALUE BLOCK
99 BAD D FIELD ON MSAVEVALUE BLOCK
100 BAD A FIELD ON PREEMPT BLOCK
101 BAD C FIELD ON PREEMPT BLOCK
102 BAD D FIELD ON PREEMPT BLOCK
103 BAD A FIELD ON PRINT BLOCK
104 BAD B FIELD ON PRINT BLOCK
105 BAD C FIELD ON PRINT BLOCK
106 BAD A FIELD ON REMOVE BLOCK
107 BAD B FIELD ON REMOVE BLOCK
108 BAD C FIELD ON REMOVE BLOCK
109 BAD D FIELD ON REMOVE BLOCK
110 BAD E FIELD ON REMOVE BLOCK
111 BAD F FIELD ON REMOVE BLOCK
112 BAD A FIELD ON SAVEVALUE BLOCK
113 BAD B FIELD ON SAVEVALUE BLOCK
114 BAD A FIELD ON SCAN BLOCK
115 BAD B FIELD ON SCAN BLOCK
116 BAD C FIELD ON SCAN BLOCK
117 BAD D FIELD ON SCAN BLOCK
118 BAD E FIELD ON SCAN BLOCK
119 BAD F FIELD ON SCAN BLOCK
120 BAD AUX FIELD ON SELECT BLOCK
121 BAD A FIELD ON SELECT BLOCK
122 BAD B FIELD ON SELECT BLOCK
123 BAD C FIELD ON SELECT BLOCK
124 BAD D FIELD ON SELECT BLOCK

<u>Error Number</u>	<u>Description</u>
125	BAD E FIELD ON SELECT BLOCK
126	BAD F FIELD ON SELECT BLOCK
127	BAD A FIELD ON SPLIT BLOCK
128	ILLEGAL B FIELD IN ENTER BLOCK
129	BAD C FIELD ON SPLIT BLOCK
130	BAD D FIELD ON SPLIT BLOCK
131	BAD AUX FIELD ON TEST BLOCK
132	BAD A FIELD ON TEST BLOCK
133	BAD B FIELD ON TEST BLOCK
134	BAD C FIELD ON TEST BLOCK
135	BAD A FIELD ON TRANSFER BLOCK
136	BAD B FIELD ON TRANSFER BLOCK
137	BAD C FIELD ON TRANSFER BLOCK
138	BAD D FIELD ON TRANSFER BLOCK
139	BAD A FIELD ON UNLINK BLOCK
140	BAD B FIELD ON UNLINK BLOCK
141	BAD C FIELD ON UNLINK BLOCK
142	BAD D FIELD ON UNLINK BLOCK
143	BAD E FIELD ON UNLINK BLOCK
144	BAD F FIELD ON UNLINK BLOCK
145	ILLEGAL B FIELD IN GATE BLOCK
146	TO NEXT LINE
147	2 DECIMAL POINTS IN ONE NUMBER
148	MORE THEN 5 CHARACTERS IN SYMBOL
149	BAD SNA IN FRONT OF A * OR \$
150	INCORRECT SNA
151	INCORRECT USE OF SNA
152	ILLEGAL NAME ON HELP BLOCK
153	ILLEGAL B FIELD IN JOIN BLOCK
154	ILLEGAL B FIELD IN LEAVE BLOCK
155	REWIND CARD HAS INCORRECT FILE NAME
156	BAD A FIELD ON START CARD
157	ILLEGAL A FIELD IN MARK BLOCK

Error
Number

Description

158	ILLEGAL E FIELD IN MSAVEVALUE BLOCK
159	ILLEGAL B FIELD IN PREEMPT BLOCK
160	ILLEGAL E FIELD IN PREEMPT BLOCK
161	ILLEGAL B FIELD IN QUEUE BLOCK
162	ILLEGAL INITIAL VALUE
163	ILLEGAL C FIELD IN SAVEVALUE BLOCK
164	ILLEGAL B FIELD IN SPLIT BLOCK
165	ILLEGAL D-G FIELDS IN SPLIT BLOCK
166	ILLEGAL B FIELD IN TABULATE BLOCK
167	ILLEGAL A FIELD IN TERMINATE BLOCK
168	ILLEGAL B-G FIELDS IN HELP BLOCK
169	ILLEGAL A FIELD ON ADVANCE BLOCK
170	BAD FIELD ON RESET CARD
171	BAD FIELD ON CLEAR CARD
172	BAD JOBTAPE CARD
173	CARD NOT LEGAL IN CURRENT CONTEXT
174	ILLEGAL MACRO PARAMETER
175	ILLEGAL B FIELD ON FUNAVAIL BLOCK
176	BAD STARTMACRO NAME
177	BAD MACRO NAME
178	ILLEGAL GENERATE REDEFINITION
179	ILLEGAL MATCH/ASSEMBLE/GATHER REDEFINITION
180	ILLEGAL MX OR MH SNA FORMAT
181	EVEN MULTIPLIER
182	ATTEMPTING TO PRESET AN UNDEFINED MATRIX
183	ILLEGAL REALLOCATE CARD
184	ILLEGAL C FIELD ON FUNAVAIL BLOCK
185	ILLEGAL SRQ REQUEST - SYSTEM ERROR
186	ILLEGAL SRT REQUEST - SYSTEM ERROR
187	LOAD FILE CONTAINS NO ENTRY POINTS OR SUBROUTINES
188	NOT ENOUGH ENTITIES TO DEFINE SYMBOL
189	ENTITY VALUE EXCEEDS CURRENT MAXIMUM
190	MAXIMUM BLOCK NUMBER EXCEEDED

<u>Error Number</u>	<u>Description</u>
191	HELP BLOCK NOT FOUND ON LOAD FILE
192	UNDEFINED SYMBOL FOR VALUE
193	ILLEGAL D FIELD ON FUNAVAIL BLOCK
194	ILLEGAL E FIELD ON FUNAVAIL BLOCK
195	ILLEGAL F FIELD ON FUNAVAIL BLOCK
196	ILLEGAL G FIELD ON FUNAVAIL BLOCK
197	ILLEGAL H FIELD ON FUNAVAIL BLOCK
198	ILLEGAL A FIELD ON SAVAIL BLOCK
199	ILLEGAL A FIELD ON SUNAVAIL BLOCK
200	ILLEGAL PARAMETER TERMINATOR
201	TOO MANY POINTS IN FUNCTION DEFINITION
202	X VALUES NOT IN INCREASING ORDER
203	NOT ENOUGH POINTS FOR FUNCTION DEFINITION
204	ILLEGAL B FIELD ON GENERATE BLOCK
205	ILLEGAL C FIELD ON GENERATE BLOCK
206	ILLEGAL D FIELD ON GENERATE BLOCK
207	ILLEGAL E FIELD ON GENERATE BLOCK
208	ILLEGAL F-I FIELDS ON GENERATE BLOCK
209	NEGATIVE OR ZERO ENTITY NUMBER
210	LOWER INDEX NUMBER EXCEEDS UPPER INDEX NUMBER
211	ILLEGAL GENERATE REDEFINITION
212	NO OPERATION FIELD ON CARD
213	COLUMN 1 MUST BE BLANK IN FIXED FIELD
214	RN9 DETECTED A F. P. VALUE OUTSIDE THE RANGE 0-1
300	REPORT INDEX FULL
301	FORMAT REQUEST ILLEGAL
302	ILLEGAL LOCATION FIELD IN TITLE/INCLUDE CARD
303	ILLEGAL TEXT CARD
304	EOF ENCOUNTERED ON INPUT FILE WHILE DEFINING A REPORT
305	ILLEGAL GRAPH CARD
306	LINE LONGER THAN 136 CHARACTERS REQUESTED
307	ORIGIN CARD CAN ONLY FOLLOW A GRAPH CARD
308	X CARD CAN ONLY FOLLOW AN ORIGIN CARD

Error
Number

Description

309 Y CARD CAN ONLY FOLLOW A X CARD
310 STATEMENT CARD CAN ONLY FOLLOW A Y OR ANOTHER STATEMENT
311 ENDGRAPH CARD CAN ONLY FOLLOW A Y OR STATEMENT CARD
312 CARD IS NOT LEGAL IN GRAPH DEFINITION
313 ILLEGAL TERMINATION OF GRAPH REPORT
314 ILLEGAL FLOATING POINT NUMBER IN CARD
400 REPORT NAME NOT FOUND IN INDEX
401 USER SYMBOL NOT DEFINED AT CURRENT TIME
402 GRAPHING ROUTINE X DETECTED AN ILLEGAL SNA
403 GRAPHING ROUTINE Y DETECTED AN ILLEGAL USE OF A TABLE
404 CAJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
405 CCJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
406 CHJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
407 CMJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
408 CTJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
409 FCJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
410 FRJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
411 FTJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
412 GJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
413 NJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
414 QJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
415 QAJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
416 QCJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
417 QMJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
418 QTJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
419 QXJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
420 QZJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
421 RJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
422 SJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
423 SAJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
424 SCJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
425 SMJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
426 SRJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
427 STJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE

<u>Error Number</u>	<u>Description</u>
428	TBJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
429	TCJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
430	TDJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
431	XJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
432	XHJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
433	XBJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
434	XLJ RPG ROUTINE DETECTED AN ILLEGAL J VALUE
500	CONTROL CARD ERROR
501	ECS READ PARITY ERROR
502	ECS WRITE PARITY ERROR
503	CURRENT ENTITY ALLOCATION SCHEME EXCEEDS MAXIMUM STORAGE
504	AT LEAST 2 BLOCKS MUST BE ALLOCATED TO RUN
505	GARBAGE IN HELP BLOCK LOAD FILE
600	CREATE WITH N=0
601	ILLEGAL UPDATE OPTION
602	NEWPL REQUESTED WITH N=0
603	P=0 FOR UPDATE RUN
604	EXPECTED UPDATE COMMAND
605	NO TEXT TO ADD OR REPLACE
606	TOO MANY UPDATE COMMANDS
607	DELETE RANGE MUST INCREASE
608	NOT ALL MODS PROCESSED
609	MORE THAN 1 MOD TO A CARD OR OLDPL ERROR
610	INTERNAL UPDATE ERROR 1
611	DELETE CARD RANGE NOT INCREASING
612	ILLEGAL UPDATE REQUEST NUMBER
700	EMPTY FUTURE EVENTS CHAIN
701	ILLEGAL LOGIC SWITCH
702	ILLEGAL STORAGE NUMBER
703	ILLEGAL TABLE NUMBER
704	MODULO DIVISION NOT VALID IN FVARIABLE
705	VALUE EXCEEDS LIST COUNT IN L FUNCTION
706	SAVEVALUE RANGE DECREASING

<u>Error Number</u>	<u>Description</u>
707	ILLEGAL SAVEVALUE SPECIFIED
708	TRANSACTION MODE GROUP IN NUMERIC JOIN
709	NUMERIC MODE GROUP IN TRANSACTION JOIN
710	TRANSACTION MODE GROUP IN NUMERIC REMOVE
711	NUMERIC MODE GROUP IN TRANSACTION REMOVE
712	TRANSACTION MODE GROUP IN NUMERIC EXAMINE
713	NUMERIC MODE GROUP IN TRANSACTION EXAMINE
714	NUMERIC MODE GROUP IN TRANSACTION SCAN
715	NUMERIC MODE GROUP IN TRANSACTION ALTER
716	ZERO CHAIN NUMBER ON LINK BLOCK
717	SOLE MEMBER OF ASSEMBLY SET ENTERED AN ASSEMBLY BLOCK
718	NEGATIVE OR ZERO COUNT AT ASSEMBLE BLOCK
719	SOLE MEMBER OF ASSEMBLY SET ENTERED A GATHER BLOCK
720	NEGATIVE OR ZERO COUNT AT GATHER BLOCK
721	NEGATIVE OR GREATER THEN 127 PRIORITY
722	REMOVING TOO MUCH FROM QUEUE
723	DEPARTING QUEUE TRANSACTION WAS NOT IN QUEUE
724	MSAVEVALUE ROWS DECREASING
725	MSAVEVALUE COLUMNS DECREASING
726	LOOP ENTRY PARAMETER IS ZERO OR NEGATIVE
727	LEAVE PRODUCES NEGATIVE CURRENT CONTENTS
728	ILLEGAL MATRIX SPECIFIED
729	TRY TO RELEASE FACILITY NOT SEIZED
730	TRY TO RETURN FACILITY NOT CURRENTLY PREEMPTING
731	QUEUE CALLS A TABLE WHICH IS NOT A QTABLE
732	BLOCK SYMBOL UNDEFINED
733	NO WRITE BLOCK FET - SYSTEM ERROR
734	BAD M FUNCTION VALUE
735	NO GENERATE BLOCKS TO START TRANSACTIONS
736	MATRICES DECREASING ON MSAVEVALUE BLOCK
737	SNA ERROR J VALUE OF CAJ
738	SNA ERROR J VALUE OF CCJ
739	SNA ERROR J VALUE OF CHJ
740	SNA ERROR J VALUE OF CMJ

<u>Error Number</u>	<u>Description</u>
741	SNA ERROR J VALUE OF CTJ
742	SNA ERROR J VALUE OF FJ
743	SNA ERROR J VALUE OF FCJ
744	SNA ERROR J VALUE OF FRJ
745	SNA ERROR J VALUE OF FTJ
746	SNA ERROR J VALUE OF GJ
747	SNA ERROR J VALUE OF NJ
748	COUNT/SELECT ILLEGAL B OR C FIELD
749	SNA ERROR J VALUE OF QJ
750	SNA ERROR J VALUE OF QAJ
751	SNA ERROR J VALUE OF QCJ
752	SNA ERROR J VALUE OF QMJ
753	SNA ERROR J VALUE OF QTJ
754	SNA ERROR J VALUE OF QXJ
755	SNA ERROR J VALUE OF QZJ
756	SNA ERROR J VALUE OF RJ
757	SNA ERROR J VALUE OF RNJ
758	SNA ERROR J VALUE OF SJ
759	SNA ERROR J VALUE OF SAJ
760	SNA ERROR J VALUE OF SCJ
761	SNA ERROR J VALUE OF SMJ
762	SNA ERROR J VALUE OF SRJ
763	SNA ERROR J VALUE OF STJ
764	SNA ERROR J VALUE OF TBJ
765	SNA ERROR J VALUE OF TCJ
766	SNA ERROR J VALUE OF TDJ
767	SNA ERROR J VALUE OF WJ
768	SNA ERROR J VALUE OF XJ
769	SNA ERROR J VALUE OF XHJ
770	COUNT/SELECT B AND C FIELDS IN WRONG ORDER
771	SNA ERROR J VALUE OF BVJ OR VJ
772	SNA ERROR J VALUE OF FNJ
773	SNA ERROR J VALUE OF LJ

<u>Error Number</u>	<u>Description</u>
774	ILLEGAL MSAVEVALUE ROW
775	ILLEGAL MSAVEVALUE COLUMN
776	ILLEGAL MX OR MH ROW SPECIFICATION
777	MX OR MH WITH UNDEFINED MATRIX
778	MSAVEVALUE WITH UNDEFINED MATRIX
779	ILLEGAL MX OR MH COLUMN SPECIFICATION
780	USER ERROR STOP
781	ILLEGAL QUEUE NUMBER
782	ABSOLUTE CLOCK TIME REQUEST EXCEEDS MAXIMUM
783	ILLEGAL FULLWORD PARAMETER NUMBER
784	ILLEGAL HALFWORD PARAMETER NUMBER
785	ILLEGAL BYTE PARAMETER NUMBER
786	ILLEGAL FLOATING POINT PARAMETER NUMBER
787	NEGATIVE VALUE FOR J OF SNA J
788	ILLEGAL BLOCK NUMBER
789	ILLEGAL VARIABLE NUMBER
790	TRANSACTION ATTEMPTED TO ENTER A GENERATE BLOCK
791	ILLEGAL FUNCTION NUMBER
792	ILLEGAL CHAIN NUMBER
793	ILLEGAL FACILITY NUMBER
794	TOO MANY PARAMETERS SPECIFIED ON GENERATE BLOCK
795	ILLEGAL GROUP NUMBER
796	VJ EVALUATION DETECTED A CIRCULAR CALL
797	FNJ EVALUATION DETECTED A CIRCULAR CALL
798	SNA ERROR J OF XLJ
799	SNA ERROR OF XBJ
800	SNA ERROR J VALUE OF FUJ
801	SNA ERROR J VALUE OF FNUJ
802	SNA ERROR J VALUE OF FIJ
803	SNA ERROR J VALUE OF FNIJ
804	SNA ERROR J VALUE OF FVJ
805	SNA ERROR J VALUE OF FNVJ
806	SNA ERROR J VALUE OF LRJ

<u>Error Number</u>	<u>Description</u>
807	SNA ERROR J VALUE OF LSJ
808	SNA ERROR J VALUE OF SFJ
809	SNA ERROR J VALUE OF SNFJ
810	SNA ERROR J VALUE OF SEJ
811	SNA ERROR J VALUE OF SNEJ
812	SNA ERROR J VALUE OF SVJ
813	SNA ERROR J VALUE OF SNVJ
814	ILLEGAL NUMBER OF PARAMETERS SPECIFIED ON A SPLIT BLOCK
815	SOLE MEMBER OF AN ASSEMBLY SET ENTERED A MATCH BLOCK
816	DECREASING STORAGE RANGE IN A SAVAIL BLOCK
817	DECREASING STORAGE RANGE IN A SUNAVAIL BLOCK
818	DECREASING FACILITY RANGE IN A FAVAIL BLOCK
819	DECREASING FACILITY RANGE IN A FUNAVAIL BLOCK
820	FUNAVAIL BLOCK REQUIRES C FIELD IF RE IN B FIELD
821	ATTEMPTING TO ENTER AN ILLEGAL BLOCK NUMBER
822	JOB EXCEEDED SIMULATED TIME LIMIT
823	NEGATIVE ADVANCE TIME CALCULATED
824	SENSE SWITCH ABORT FLAG
825	FIELD C DOES NOT EXCEED FIELD B BY AN EXACT MULTIPLE OF INDEX FIELD D
826	PREMATURE EOR/EOF ON READING BLOCKED BINARY JOBTAPE



CYBERNET USAGE

H

There are several minor differences with running GPSS V/6000 on CYBERNET. These are listed below.

The A-field on the SIMULATE card is interpreted as 1/60 of the number of SBUs allowed for the simulation. For example:

```
SIMULATE      3.5
```

would allow the model to use up to 210 SBUs.

The STATIC option should not be used when compiling FORTRAN routines for HELP blocks.

When creating a file of numbers for RN9, a LDSET±FILES= card is also needed in the load sequence.

O

O

O

O

O

O

O

COMMENT SHEET

MANUAL TITLE: General Purpose Simulation System (GPSS) V/600
General Information Manual

PUBLICATION NO.: 84003900

REVISION: D

NAME: _____

COMPANY: _____

STREET ADDRESS: _____

CITY: _____ STATE: _____ ZIP CODE: _____

This form is not intended to be used as an order blank. Control Data Corporation welcomes your evaluation of this manual. Please indicate any errors, suggested additions or deletions, or general comments below (please include page number references).

Please reply

No reply necessary

CUT ALONG LINE

AA3419 REV. 4/79 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE



FOLD

FOLD

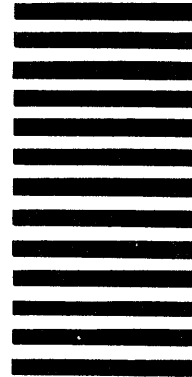


NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 8241 MINNEAPOLIS, MINN.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
Headquarters Publications Writing
Publications and Graphics Division
P.O. Box 0, HQC02C
Minneapolis, Minnesota 55440



CUT ALONG LINE

FOLD

FOLD

