INTERMEDIATE MANUAL

Print 1

AUTOMATIC CODING SYSTEM

FOR THE IBM 705

Contents

This intermediate manual is issued to accompany the distribution of limited versions of the PRINT I system. The structure of the system and the operation list is final at this time. Not included in this manual are descriptions of certain of the mathematical sub-routines in various mantissa lengths, which will be further revised with respect to memory occupancy and internal structure. The tinkertoy appendix is also not available at this time.

Existing pre-edit and executive routines will be furnished in card form upon written request, automatically placing those installations on the mailing list for subsequent revisions, particularly to include all mantissa lengths. The symbolic listing of the pre-edit routine will not normally be furnished, except upon special request. Coding sheets may also be obtained upon request. Assistance in programming and operating the PRINT I system may be obtained from Applied Science representatives.

Working committee

Bemer, R. W.
Glans, T. B.
Krasnow, E.
Hira, G. R.
Michels, L.
Hoggatt, A.
Levitan, R.

Programming Research Department
International Business Machines Corporation
590 Madison Avenue
New York 22, N. Y.

PRINT I

Purpose

The PRINT I (PRe-edited INTerpretive) system has been primarily designed to meet the engineering and scientific computing needs of those 705 installations where such work is a secondary computing requirement.

General characteristics

PRINT I is an automatic coding system of the interpretive type, designed to make the 705 itself do the major portion of the coding and clerical work. It is designed for ease of learning and operation by personnel with little or no previous programming experience. It has the following desirable features:

1. *Floating point arithmetic.* The programmer need not concern himself with the position of decimal points throughout calculation. Entry of fixed point numbers and production of fixed point printed output may be made without the operator concerning himself with the fact that internal calculation was in the floating point mode.

2. *Matching mathematical functions.* All functions operate near optimum speed and are computed to an accuracy which is consistent with the arithmetic used. Facility is made for the user to insert his own sub-routines, by using the "tinkertoy" appendix (Appendix III). The library of functions is greatly extended by the floating sub-routine feature, which allows non-standard functions in tape storage to be used as though they were standard functions in core memory.

3. *Variable address and instruction format.* The instructions in this system are of varying length and contain a variable number of specified addresses, depending on the amount of information each instruction must carry. This is consistent with the variable length features which enhance the 705. Coding is done in a variable field, with the multiple addresses and other information separated by commas.

4. *Advanced instruction set.* Many useful combinatorial instructions are incorporated to give greater flexibility to calculations. Among these are vector multiply-adds, polynomial multiply-adds, special operations for convergence testing, indirect address features, counting switches, counting printing instructions and a completely automatic table search operation with an adjustable block feature. All of these are performed by the use of a single instruction.

5. *Index registers.* An incremental type of indexing is used for address modification, substantially reducing the number of program steps to be written, by factors of from 2-1 up to 50-1. Each address may be indexed by the sum of the contents of up to three registers, greatly facilitating internal loops. Index registers may also be used as counters for control purposes, without actually being used for address modification.

6. *Repeat instruction.* This instruction controls automatic repetition of a following instruction, allowing grouped data to be handled with very few instructions. This is advantageous in converting input and output data from fixed decimal to floating and vice versa, in table searching, in matrix calculations, etc. It also permits a secondary form of indexing.

7. *Facility.* PRINT I may be thought of as a means of using the 705 as a giant but convenient desk calculator. Elapsed time between problem statement and production of answers can now be a matter of hours, rather than days or weeks. The instruction set is straightforward and restrictions are minor; many logical errors in the written program are automatically detected and typed out to the operator in the form of an error message.

8. *Interpretive system.* PRINT I is operated by an executive routine which is always in memory during the running of a problem. This routine fabricates the requisite 705 instructions as it computes, finding the various components in the pattern of the converted PRINT instruction. There is no necessity for developing expert machine language programmers; the intricate coding is already built in. The executive routine, under various options, occupies

from 4000 to 6000 characters in memory (equivalent to 800 to 1200 705 instructions), but experience with the system has shown that for mathematical work one PRINT instruction is the equivalent of about 40 705 instructions. The break-even point is therefore at around 30 PRINT instructions, which is a relatively small program. Interpretation is not generally time-consuming in PRINT, because the repeat instruction enables the following instruction to be performed n times in succession with only a single interpretation. For the remaining $n-1$ times, the instruction operates, in general, even faster than the most expert coder or compiler could generate the program. This statement may appear contradictory unless it is understood that, due to the possibility of selecting the most advantageous fixed locations in memory, certain machine characteristics may be utilized to decrease the operating times. These same routines, if compiled in random memory locations, would be incapable of operating correctly.

705 components

The only memory components required to operate the PRINT system are the magnetic core memory and sufficent tape units (> 3) to handle expected problem size. An on-line printer and on-line card reader are assumed to be available, although they may be dispensed with by certain modifications to the system.

System components

When operating in the PRINT system, the 705 is for all practical purposes changed to a different machine, that is in a non-physical sense. Certain simulated hardware exists in the system, as:

1. *Index registers.* There are three of these registers. They are addressable by certain instructions for setting and augmenting their contents. They are effectively addressable in the body of other instructions to enable their contents to be used to modify addresses.

2. *Limit registers.* There are three of these, one for each index register. They are for maintaining limits to the contents of the index registers, which are

used for automatic termination of loops of indexed instructions.

3. *Line image.* This is an image in memory of the printer type wheels, such that each of the type wheels is effectively addressable. All printing and error correcting routines associated with printing are automatically associated with this line image.

4. *Heading image.* This is also an image in memory of the printer type wheels, but is used exclusively for heading printed pages of reports in any format the programmer desires. The programmer merely uses two cards in his program to specify this heading format.

5. *Card image.* This is an image in memory of the card columns. All columns are effectively addressable. All card reading, whether from the card reader or tape, enters this area; all card writing, whether on tape or to the card punch, is done from this area.

6. *Fixed symbolic locations.* There are six fixed locations in memory. Although addressed symbolically, they are automatically interpreted as actual addresses:

For numbers (data word length)	*For addresses*
PAC1 (Pseudo-ACcumulator 1)	LAR1 (Location of ARG1)
PAC2 (Pseudo-ACcumulator 2)	LAR2 (Location of ARG2)
ARG1 (ARGument 1)	
ARG2 (ARGument 2)	

PAC1 is the basic component for the multi-address instructions, for which it is the understood address. All arithmetic operations send the result to PAC1 as a secondary result storage or temporary working area. The other locations are mainly pertinent to the table search operations.

Overall mode of operation

The use of PRINT to solve a problem falls into four basic steps. They are described very generally here; the actual details of each of these steps are contained in the full description of each which follows later in the manual.

1. The programmer writes, on the symbolic coding form for this system, a

sequence of PRINT and/or 705 instructions designed to bring in data, do arithmetic and logical operations, and finally prepare and produce output data. He does this knowing the function of each of the PRINT instructions, as described in detail under individual sections.

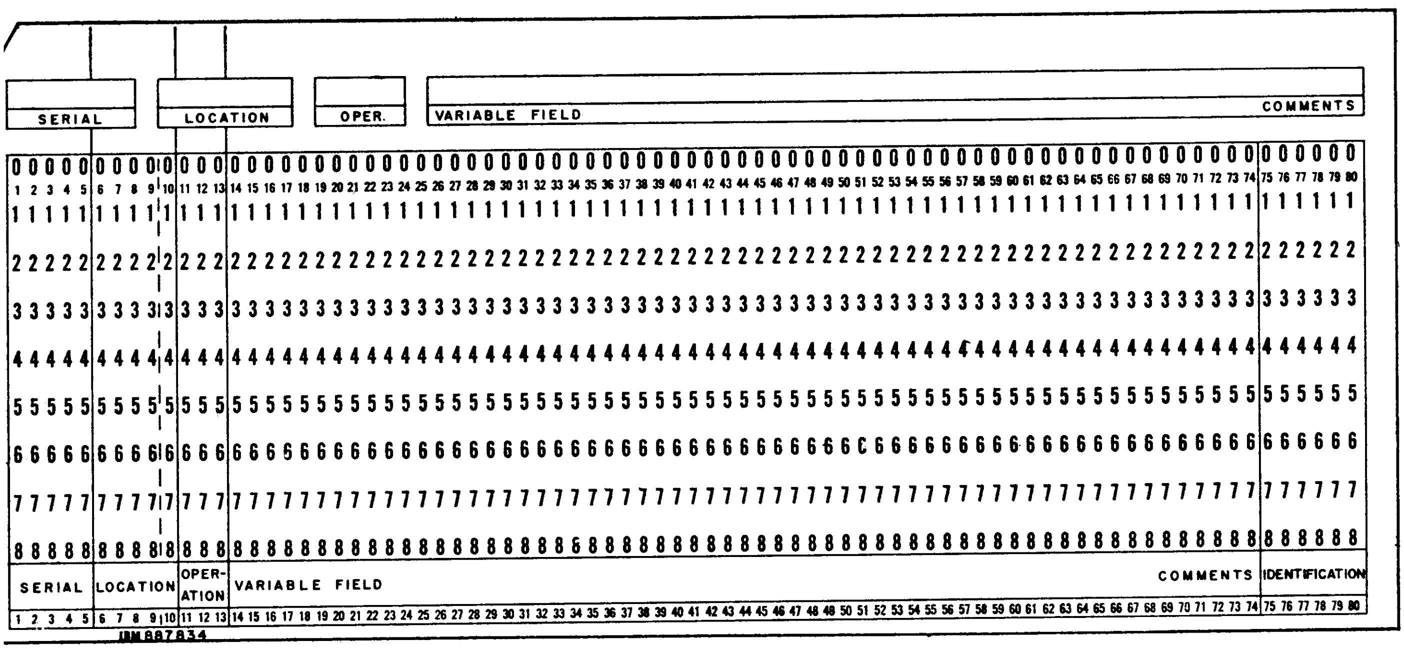
2. Cards are punched from this coding form, each line of coding representing a single card. Punching is done in consecutive columns and may be done without a drum card, as the format is variable. The only column skipped before the end of punching is that defining the end of the variable field and the beginning of the comments.

3. These cards are read into the 705 along with the PRINT I system, which consists of two independent parts. The first of these is the pre-edit routine, which will process the program cards and convert them to pseudo-instructions in card or tape form for actual running of the problem. The second part is the executive routine, which is always in core memory during the operation of a program prepared for this system. The pre-edit routine is not maintained in memory after performing its function and is destroyed by entry of the executive routine and the program. It is possible to pre-edit at one time and save the execution of the problem until a later time, as these are entirely separate functions. Pre-editing is a triple function of assembling, compiling and conversion to a form more convenient to the executive routine. For each card with its mnemonic instruction and variable field, pre-edit produces a corresponding pseudo-instruction especially tailored for the fabrication of 705 instructions from its components. These are of varying length of characters according to the operation specified. Matched sets of mnemonic and pseudo-instructions may be printed at pre-edit time, at the option of the operator, together with the comments punched in the right hand part of the variable field. This should be his permanent coding record.

4. The actual running of the problem is under the control of the executive routine, which may be called from tape immediately after pre-editing. The executive routine fills from 4000 to 6000 characters in memory, including the floating sub-routine position and input-output images. Overflow or sign check indicators are not used in PRINT as decision elements. They are reserved for stops while operating with 705 instructions and the switches may therefore be set to automatic stop during the operation of PRINT. Any entry to PRINT sets up the ASU's as required for its operation. All ASU's are therefore available for use in 705 language. Their settings should be noted from the ENTer sub-routine (see Page 12) to avoid redundant resetting for 705 usage.

General coding instructions

Addressing in the PRINT I system is entirely symbolic; that is, the address nomenclature can be descriptive of the contents. The symbolic address of a location must be a sequence of one alphabetic character followed by three or four alphanumeric characters. If these following characters are all numeric the address is said to be "regional", which is a sub-class of symbolic notation with certain useful properties (see the next section). A "region" is indentifiable by the first or first two characters (i.e., the "G" region, the "F3" region). If the sequence begins with a numeric character, the following characters must be all numeric and this signifies an actual 705 address. The symbolic locations may be coded in any sequence desired. The format of the PRINT coding card is:



Columns 1-5 — *Serial number.* This provides for sequence control and the collation of change cards. Serial numbers must be in ascending sequence. A convenient convention is the use of the first two columns for coding page number, the second two for line number and the last for inserts.

Columns 6-10 — *Symbolic location.* This field provides a referral name for the entry; that is, if the entry is not referred to by any other instruction in the program, the field can and should be left unpunched. This will reduce the size of the table of symbolic and actual address correspondence, thus decreasing the running time of pre-editing by minimizing search time. If the field is punched it must follow the rules for a symbolic address, with the alphabetic character in column 6. Punching is optional in column 10.

Columns 11-13 *Operation code.* This field must be punched with a 3 character (including blanks as characters) mnemonic code which describes the function of the entry. This may be a PRINT operation, a 705 operation or one of the several special operations for constants, memory reservations, origins or headings.

Columns 14-74 *Variable field.* This field is punched as the requirements of the particular instruction dictate. The first blank column indicates the beginning of the comments field, which may actually extend through to column 80 if no identification is required.

Columns 75-80 *Identification.* Any 6 character alphanumeric designation may be punched (ganged) here to identify the program. An identification obtained from the *first* card of the symbolic program deck will be punched in the first 6 columns of the 705 load cards produced by pre-edit for reloading, and will also appear in the heading of the pre-edit listing.

Regional notation

An alternate method of using symbolic addressing is available if the programmer desires to code with "unitized" components. If the symbolic address is regional, the serial number in columns 1 to 5 may be omitted, in which case the numbering sequence within the regional address controls instruction sequence in the program. Columns 6 to 10 will always be filled with a regional address, regardless of referral status, and referral will now be indicated by punching an 11-punch in column 1. The drum card of the keypunch should be arranged to skip to column 6 for the next punching. If the programmer fears becoming careless in noting referral addresses, he may gang the 11-punch in column 1 of every card, but this could retard the pre-edit process by carrying a complete table of referrals.

Regional addressing is convenient for quick replacement of identifiable components with a specific function in the program. To illustrate, consider that in a program to compute airplane performance the calculation of engine thrust is assigned to region T1. This region receives as input data certain information produced by other regions, computes thrust with this data and certain equations, finally putting this resultant thrust value in a location usable to other regions. For several different engines, or several different ways of computing thrust, different T1 regions would be coded. All of these receive and store data in addresses not common to the computing regions, but accessible to all. If the programmer wishes to compute performance for a certain configuration he selects one form for each region involved and processes this combination through the pre-edit routine. He is thus guaranteed that housekeeping is perfect and that no pattern of computation will have been erroneously disrupted.

Coding PRINT instructions

The variable field of PRINT instructions is coded according to the context of the instructions. Each operation is described as having a certain number of positions in the variable field. Each of these positions are separated by commas. An exception occurs for indexable instructions, where a position is defined to contain both the address and its index register tag, although separated by a comma. The tag is therefore in a sub-position immediately following the address it modifies. An address is a field of four or five characters. It is symbolic if it begins with an alphabetic character; if the first character is numeric, all must be numeric and the address is actual. An index tag is a field composed of the digits 1, 2 and 3 not repeated which designate the index register or registers which are to affect the address in context. If an address is not to be indexed, no tag field is written. In the first example the address in the second position is the only one indexed; in the second example the address in the third position is also indexed.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
	ADD	P 202, R 532, 12, QYR5	A blank column terminates
	ADD	P 202, R 532, 12, QYR5, 2	the instruction and starts
	MPY	RATE, 2, TIME, 23, DIST, 3	the comments

Both numeric and alphabetic characters are used in coding for this system.

As a standard precautionary practice, always write the letters Ø, I and Ƶ as shown, with slashes and cross-bars to safely distinguish them from the numerals 0, 1 and 2.

Every program will begin with either a 705 instruction or an ENT (ENTer) instruction. Every transfer of control from 705 to PRINT instructions and back will be called for by the programmer. Consequently, every block of PRINT instructions must be preceded by an ENT, which is compiled by the pre-edit into three 705 instructions:

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
BADD-11	SET	4, 1	
BADD-6	LØD	BADD-6, 1	
BADD-1	TR	(to the address of the first instruction in the ENT Sub-routine in the PRINT Executive Routine)	

The basic address of the first PRINT instruction is at BADD. This is computed by the ENT sub-routine from BADD-6 in ASU01. ASUs are set to length and control is transferred to the fetch sub-routine, which brings in the first PRINT instruction in the interpretation cycle.

Following an ENT, all entries are considered by pre-edit as PRINT instructions until the instruction LVE (LeaVE) is encountered. LVE is a PRINT instruction whose address is normally pre-edited as the location of the next 705 instruction. When executed, it will cause the executive routine to transfer control to that instruction. All succeeding entries will then be considered as 705 operations or special operations until another ENT is encountered. Thus ENT and LVE are normally coded without addresses in the variable field. When LVE is coded with the address of a 705 instruction, pre-edit gives that address in conversion rather than that of the next 705 instruction in sequence.

An asterisk in the variable field of an ENT indicates that this is the point at which the operation of the program will be commenced, rather than the first ENT or 705 instruction encountered by the pre-edit. If more than one ENT contains an asterisk in the variable field, the first encountered takes precedence. An ENT must not precede a 705 or special instruction, else a compiling error will occur in memory assignment. When successive entries change from PRINT to 705 instructions or vice versa, without intervening ENT or LVE entries, pre-edit will type out a mode change error message.

Upon executing a LVE instruction, advantage may be taken in 705 operations of the fact that the ASUs are left with known length settings, as follows:

ASU	*Length*	*ASU*	*Length*	*ASU*	*Length*
01	4	06	4	11	2
02	1	07	4	12	3
03	2	08	Word Length	13	Indeterminate
04	4	09	1	14	5
05	4	10	1	15	18

Two successive commas imply that the intervening address is that of the main pseudo-accumulator PAC1, which is a field in memory reserved for this function. PAC1 is incapable of being indexed, even if tagged; a zero indicator is automatically inserted for it by the pre-edit routine. PAC1 may also be addressed by the symbol PAC1. If fewer addresses are coded than required by a particular instruction, the remaining addresses will be interpreted to be PAC1 by the pre-edit. For example, the following instructions are equivalent, incidentally doubling the contents of PAC1.

ADD PAC1, PAC1, PAC1 ADD , , b ADD

An exception to this rule occurs in the SAC operation. If the result addresses are not specified in the second and third positions of the variable field, the second position is interpreted as PAC1 and the third as PAC2.

Coding 705 instructions

705 instructions are coded either before the first ENT or between LVE and the next ENT. Standard 705 mnemonic codes are used. The first field after the operation code is interpreted as the address. An actual address can be any combination of 4 or 5 numeric digits, as the leading zero does not have

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
	TR	CYCLE	
	RAD	FØL3-52, 2	FØL3-52 zoned for ASU 02
	SET	9, 13	Set ASU 13 to length of 0009

to be punched. A symbolic address must satisfy the same criteria as the addresses of PRINT instructions do. If the address is terminated in a sign, the next field is interpreted as an increment. The following field is the ASU designation. The instruction refers to the 00 accumulator if no ASU coding is present.

Special operations

PRINT I uses various mnemonic special operation codes for initial organization of a program. These are illustrated at the end of this section. These operations are static and do not create working PRINT or 705 instructions.

ADC (ADdress Constant) produces a 4 character constant which is the 705 address determined by the symbolic address, increment and ASU coding in the variable field.

ORG (ORiGin) controls the actual memory assignment of subsequent instructions or areas. These are four types of ORG entries, and pre-edit will handle up to 100 ORGs with addresses in the variable field.

1. When the address in the variable field is actual (i.e. numeric), the first character of the next entry will be at that specified address.
2. When the variable field is blank, all following entries will be assigned in order following the highest location assigned previously.
3. When the address in the variable field is identical to the symbolic address of the ORG itself, the location of the previous entry will be stored in the table of origins for later reference.
4. When the address in the variable field is a symbolic address stored under the conditions of type 3, the succeeding entries will be assigned following the location stored by type 3. This is a device for remembering and continuing an interrupted series.

CON (CONstant) and BLK (BLocK) reserve filled or unfilled memory space. For either entry, the first position in the variable field is a number of from 1 to 3 digits specifying the length of the entry, which in the case of CON is limited to 50 characters. If an asterisk precedes the length specification the entry starts with a memory position ending in 0 or 5. If a constant is signed, the plus or minus sign follows just after the length; plus signs may not be omitted. A blank column and the actual constant follow. A record or group mark may appear only in the first character of a constant. The address assigned by pre-edit is that of the highest memory position (or right-hand character) in the field.

FLC (FLoating Constant) is a PRINT entry corresponding to CON for the 705. Coded in the variable field is the sign and the 1 or 2 character exponent, followed by the mantissa sign and as many digits of the mantissa as the coder cares to write. These are not separated by commas. The initial number of characters in the mantissa is not limited; pre-edit automatically converts to internal format, without rounding if the number of characters exceeds mantissa length for the system. A blank variable field is considered an error by pre-edit.

DEL (DELete) is a special operation for program correction and is explained in the operation of pre-edit and system entry.

REG (REGister reservation) is a PRINT entry corresponding to BLK for 705 entries. It is used to reserve memory space for floating point PRINT numbers (words). For reservation of a single word or number space the variable field is normally left blank, as the length is already specified by the system. The word length area is addressed in other instructions by the symbolic address of the REG entry. If the address is regional, a lower address in that same region may be written in the variable field, signifying reservation for all addresses within those limits. For indexing purposes, all randomly symbolic addresses in an operational sequence must be reserved sequentially and individually. It is therefore preferable to reserve addresses for indexable instructions in the regional mode. Pre-edit will accommodate up to 60 of these multiple reservations.

SAY (SAY it) will enter a line of comment into the pre-edit listing.

HDG (HeadDinG) is a PRINT entry for inserting a page heading for printed reports. Coded in the variable field are a blank (column 14) and up to 60 characters in columns 15 to 74. One or two cards may be used, the second corresponding to type wheels 61 to 120.

FIN (FINish) is an entry which signifies termination, in the card reader, of the program to be pre-edited. It has the same effect as an end-of-file signal. This permits data cards to be loaded into the reader simultaneously with the program, without being considered as entries to be pre-edited. The entire program may thus be run in a continuous fashion.

IBM

FORM 19 6905 0

PRINT I SYMBOLIC CODING FORM

PROBLEM: ILLUSTRATING THE VARIOUS USAGES FOR SPECIAL OPERATIONS

CODER | DATE | PAGE | OF

SERIAL 1-5	LOCATION 6-10	OPERATION CODE 11-13	VARIABLE FIELD 14-	COMMENTS -80
	MANT	ADC	PØWER - 2,9	(,09 also correct)
		ØRG	5040	(05040 also correct)
	A123	RAD	CØL08,2	(instruction located at 5044)
		ØRG	35040	
	CØN45	CØN	2+b45b.....	(the number +45, located at address 35041)
	XNYC	BLK	56	(reserves 56 characters in memory)
		CØN	* 28bFOURbSCOREb.....	
	ZERØ	FLC	+0+0	0×10^{0} = ZERO
	LØC1	FLC	+1+1	$.1 \times 10^{1}$ = ØNE
		FLC	-12+528	$.528 \times 10^{-12}$
		FLC	+3-2	$-.2 \times 10^{3} = -200$
		FLC	+3-200	"
10000		DEL	10006	
	F121	REG	F119	
	F119	REG	F121	
	F119	REG		Identical effects
	F120	REG		
	F121	REG		
	F121	REG		Will cause normal indexing
	F120	REG		to occur in
	F119	REG		reverse order
		SAY	THE FØLLØWING 3 ADDRESSES ARE DESIGNED FØR INDEXING	
	JØNES	REG		Pre-edit assigns memory positions
	SMITH	REG		in the order in which
	BRØWN	REG		they are encountered
	A001A	REG	A100A	Useful technique for greatly
	A001B	REG	A100B	expanding the number
	B001A	REG	B100A	of available regions
		SAY	THE FØLLØWING LINES SHØW ILLEGAL USAGE	
10006		DEL	10001	Will delete 10006 only
20684		DEL	A106,13,,F*/4	(Dangerous to maintain old variable field)
	F 120 1	REG		(Will not be included in F119 - F121 sequence)
		REG	SMITH	
	SMITH	REG	JØNES	
	TEMP2	REG	TEMP1	

Indexing

A system of indexing is simulated within the PRINT framework. Most 705 programmers are already familiar with one means of specifying the location of a number without using the actual address. This is symbolic addressing, where the actual address is determined by searching a table of symbolic addresses, each of which has a corresponding actual address. Indexable addressing is one further step up conceptually. If either a symbolic or actual address, not only is the corresponding actual address determined from the symbolic, but the address which the instruction really refers to is that actual address plus the number contained in the index register specified. If that index register has a different number in it every time that the same instruction refers to it, then the same instruction obviously uses a different address every time, although the instruction itself never changes. The examples in Appendix II show how the same instruction may be used repetitively to advantage. The justification for indexing is the resultant economy in the number of instructions that the programmer must write.

In the out-of-context example shown, the angle whose sine is placed in PAC1, and whose cosine is placed in PAC2, is not the angle in the address P220. Since R2 (register 2) and R3 have been set to 3 and 8 word lengths respectively, it is rather the angle in address P231, which is P220 plus 3 plus 8.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
P240	REG	P220	Reserve sequential addresses
	ENT		
	SR 2	3	Set R2 to 3 word lengths
	SR 3	8	Set R3 to 8 word lengths
	SAC	P220, 23	Sine to PAC 1, cosine to PAC 2

The 23 tag after the address in the first position indicated that the address was to be incremented by the sum of the contents of R2 and R3. In PRINT I, the contents are added to the address and indexing is said to be incremental. By contrast, 704 indexing is decremental. There are 3 index registers, referred to as R1, R2 and R3. Any address in an indexable instruction may be incremented by the contents of any of these registers or the arithmetic sum of the contents of any two or all three. This alteration takes place in a work area before fabricating the necessary machine language instructions from the address portions of the PRINT instruction involved. The original PRINT instruction in operating sequence is never altered. Loops formed by transfer on index instructions will therefore be re-indexed from the original instructions. Such transfer is dependent upon the contents of the index register not having exceeded a specified limit.

The contents of index registers are used only for address modification with 705 add-to-memory instructions. The contents are unsigned and 4 digits in length. Increments are carried in memory as true numbers, decrements as the complements of 40,000. To increase operating speeds, all possible sums of contents of index registers are carried along in memory. When any register is altered, the contents of each combination in which that register participates are altered by the same amount. This permits indexing any address by a single add-to-memory instruction.

Direct access to the contents and limits of index registers may be had (in 705 machine language) by using the actual addresses of 4-character unsigned numeric fields as follows:

R1	0718	R1 limit	0723
R2	0728	R2 limit	0733
R1+R2	0738		
R3	0748	R3 limit	0753
R1+R3	0758		
R2+R3	0768		
R1+R2+R3	0778		

Diagnostic routines

Two types of diagnostic methods for error finding are used with PRINT instructions. The first of these is a memory print associated with the system control, the operation of which is described under the section on pre-editing (page 44). This method is used primarily to determine cause of machine stops during computation.

The second type of diagnostic is that commonly called "snapshot", and is entirely under the selective control of the programmer. This is accomplished by inserting extra instructions in the program to be pre-edited. These instructions are designed by the programmer to view selected intermediate results or logical path indications. When the program is ascertained to be correct, these snapshot instructions have their operation codes changed to DEL and are re-collated in with the previously assembled program, thus removing them from the operating program. This feature is possible only because of the fast re-assembly time in the PRINT system. Most programs will take from 30 seconds to 1 minute for tape re-assembly.

For detail work, a 705 machine language tracing routine is furnished. This routine (primarily developed by Mrs. Helen Meek of the Hughes Aircraft Company, Culver City, California) may be used as a separate diagnostic tool for all work encountered by the installation, including commercial problems. The basic principle of this routine is the temporary displacement (and storage for later replacement) of certain operating instructions in the working program by transfers to the tracing routine. This permits high speed operation to various points of interest, at which time detailed tracing occurs. High speed operation of the program may be resumed at specified points. Determination of the local regions to be traced is under card control.

Arithmetic operations

All PRINT I arithmetic operations use numbers in floating decimal form as the operands. All 705 operations are in fixed decimal form. A floating decimal number is essentially a piece of data and is referred to as a "word". This floating point word is comprised of two parts, a proper decimal fraction (called the "mantissa") with a non-zero leading digit and a power of 10 multiplying that fractional number (called the "power", although "characteristic" is an alternate term). Floating point words are

stored in memory as:	although written for input as FLCs:
$XXX. . . . XX\overset{\pm}{X}\overset{\pm}{P}P$	±PP±XXX. . . .

The X's represent the digits of the mantissa and the P's represent the two digits of the power, which may range from −99 to +99. The dots signify that PRINT I is furnished in separate forms for 8, 10 and 12 digit mantissas. The 12 digit system (word length = 14 characters) will be furnished originally with 12 digit arithmetic but having the mathematical functions normally provided with the 10 digit system. A 12 digit system to consistent accuracy and a 20 digit system will be available about January 1957.

Each of these systems will then be complete in itself for all operations, having all sub-routines designed to an accuracy equivalent to the length of the mantissa. Sample words in input format, floating point format and their equivalent fixed decimal form are:

Input format	*Internal format*	*Actual number*	
+0+12345678	+ + 1234567800	.12345678	
+5—1234567899	— + 1234567805	12345.678—	
—5+1234567899	+ — 1234567805	.0000012345678	
+1+6	+ + 6000000001	6	
+8+6	+ + 6000000008	60,000,000	
—10+6	+ — 6000000010	.00000000006	
+6+1	+ + 1000000006	100,000	$(=10^5)$
+2—3579	— + 3579000002	35.79—	
+0+0	+ + 0000000000	0	

The second example is —.12345678 times 10^5. It can be seen from the examples that when the power is positive, it represents the number of whole number digits; when the power is negative, it represents the number of zeros to be placed after the decimal point before the actual number begins. A power of zero means that the number is a decimal number just as it is without using the power. A true zero is always signed positively.

Whereas as 705 instructions refer to the contents of a single address, PRINT instructions are in a multi-address form. All PRINT operations except FPR are performed without rounding, to save operation time. If this should ever cause inconvenience, use a system with a longer mantissa. Arithmetic instructions which are found to refer to zero operands will operate in accelerated fashion, since all operands are first tested for zero in the arithmetic subroutines.

If an error occurs during execution because of the impossibility of foreseeing certain illegal conditions, an error message will be written on the typewriter. The "tinkertoy" appendix provides options for this type-out.

If the programmer wishes to conserve memory space he may select the option which types out the letter E followed by a 2 digit code number; referral to the manual will tell him the type of error which has occurred. If economy of memory is not vital, he may select the option of typing out an expository message. Under either option, the actual address of the failing instruction is also typed out. Error messages are:

E01: Division by zero
E02: Logarithm of zero or a negative number
E03: Sine and cosine of an angle greater than $\pm 318\pi$
E04: Square root of a negative number
E05: Power overflow (>99)
E06: Power underflow (<-99) (only if desired by user)
E07: Line image overflow
E08: Too many whole numbers
E09: Echo check, 0901 error, channel 12 in tape
E10: Line or HDG won't write correctly 0902
E11: Read card error or card punch error
E12: Card won't punch correctly 0902
E13: 0901 error on write tape
E14: Tape won't read/write correctly
E15: End-of-file before read/write tape completed
E16: Exponential to the base 10 of $|ARG| \geq 99$
Exponential to the base e of $|ARG| \geq 225.65334$

Summary of mnemonic codes

Non-indexable operations

ATR	Alternating TRansfer	TNZ	Transfer on Non-Zero
BSi	BackSpace tape "i"	TRM	TRansfer on Minus
LVE	LeaVE PRINT	TRP	TRansfer on Plus
RCD	Read a CarD	TRU	TRansfer Unconditionally
RPL	RePLace	TRZ	TRansfer on Zero
RPT	RePeaT	TXi	Transfer testing indeX limit, augmenting "i"
RWi	ReWind tape "i"	WCD	Write a CarD
RWR	Repeat With Reset (PACl)	WHi	Write a Heading, space "i"
SRi	Set index Register "i"	WLi	Write a Line, space "i"
TMi	write Tape Mark on tape "i"	XTP	eXTract Power
TNi	Transfer Not testing limit, augmenting "i"		

Special operations

ADC	ADdress Constant	FLC	FLoating Constant
BLK	BLocK	HDG	HeaDinG
CON	CONstant	ORG	ORiGin
DEL	DELete	REG	REGister reservation
FIN	FINish	SAY	SAY it

Indexable operations

ADD	ADD	MPM	Minus Polynomial Mult.—add
ART	ARcTangent	MPY	MultiPlY
DIV	DIVide	PMA	Polynomial Multiply—Add
EXD	EXponential, Decimal base	RTi	Read Tape "i"
EXE	EXponential, base E (e)	SAC	Sine And Cosine
FLO	FLOat	SQR	SQuare Root
FPR	Fix for Printing Rounded	SUB	SUBtract
FXP	FiX for Printing	TAB	Transmit ABsolute
LGD	LoGarithm to Decimal base	TMT	TransMiT
LGE	LoGarithm to base E (e)	TNA	Transmit Negative Absolute
MAD	Multiply – ADd	TRC	TRansfer on Comparison
MDV	Minus DiVide	TRE	TRansfer on Equality
MMA	Minus Multiply – Add	TSC	Table Search on Comparison
MMY	Minus MultiplY	WTi	Write Tape "i"

Summary of indexable operations

OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
ADD	ØPER 1, ØPER 2, SUMM	(ØPER 1) + (ØPER 2) ⟶ SUMM
SUB	ØPER 1, ØPER 2, DIFF	(ØPER 1) - (ØPER 2) ⟶ DIFF
MPY	MLPLR, MCAND, PRDCT	(MLPLR) (MCAND) ⟶ PRDCT
MMY	MLPLR, MCAND, NGPRD	-(MLPLR) (MCAND) ⟶ NGPRD
DIV	DVDND, DVSØR, QUØT	(DVDND) ÷ (DVSØR) ⟶ QUØT
MDV	DVDND, DVSØR, NGQUØ	-(DVDND) ÷ (DVSØR) ⟶ NGQUØ
MAD	MLPLR, MCAND, CRSFT	(MLPLR) (MCAND) + (PAC 1) ⟶ CRSFT
MMA	MLPLR, MCAND, CRSFT	-(MLPLR) (MCAND) + (PAC 1) ⟶ CRSFT
PMA	ADDND, MCAND, RSULT	(ADDND) + (PAC 1) (MCAND) ⟶ RSULT
MPM	ADDND, MCAND, RSULT	(ADDND) - (PAC 1) (MCAND) ⟶ RSULT
SQR	SXTY 4, EIGHT	$\sqrt{(\text{SXTY 4})}$ ⟶ EIGHT
SAC	ANGLE, SINE, CØSIN	sin (ANGLE) ⟶ SINE, cos (ANGLE) ⟶ CØSIN
ART	TNGNT, ANGLE	$\tan^{-1}$ (TNGNT) ⟶ ANGLE
LGD	NUMBR, DECLG	$\log_{10}$ (NUMBR) ⟶ DECLG
LGE	NUMBR, NATLG	$\log_{e}$ (NUMBR) ⟶ NATLG
EXD	EXPØN, TEN2X	antilog (EXPØN) ⟶ TEN2X
EXE	EXPØN, E2THX	antilog (EXPØN) ⟶ E2THX
(FSR)	ARGUM, RSULT	function (ARGUM) ⟶ RSULT
TMT	HERE, THERE	(HERE) ⟶ THERE
TAB	MINUS, PLUS	|(MINUS)| ⟶ PLUS
TNA	PL/MN, MINUS	-|(PL/MN)| ⟶ MINUS
TRC	TRADD, THIS, THAT	Transfer to TRADD if (THIS) ≥ (THAT)
TRE	TRADD, THIS, THAT	Transfer to TRADD if (THIS) = (THAT)
TSC	± Δ, TABLE, ARGUM	Search argument table for first number ≥ (ARGUM), be-
		ginning at TABLE. f(TABLE) is ± Δ word lengths away.
WTi	BEGIN, ENDD, TRADD, TM	Write all successive words from BEGIN to ENDD, inclus-
		ive, as 1 record on tape i. Transfer to TRADD if end-of-
		file is reached, write tape mark if TM is written.
RTi	START, TRADD	Read record from tape i, filling as many successive locat-
		ions as on record, beginning with START. Transfer to
		TRADD if a tape mark is encountered.
FXP	FLNUM, t, wW, dD, s	Fix (FLNUM) x 10^{s} for print in line image, decimal point
		in type wheel t, with w whole numbers and d decimals.
FPR	FLNUM, t, wW, dD, s	Same as FXP, except round the number when fixing.
FLØ	CØLXX, n, R/L s, FLNUM	Take the n digit number with units position in column XX.
		Move the decimal point R(ight) or L(eft) s positions. Put
		in floating point format in FLNUM.

Summary of non-indexable operations

OPERATION CODE 11-13	VARIABLE FIELD 14-	COMMENTS -80
TRZ	TRADD, TEST	Transfer to TRADD if (TEST) are zero
TNZ	TRADD, TEST	Transfer to TRADD if (TEST) are non-zero
TRP	TRADD, TEST	Transfer to TRADD if (TEST) are plus
TRM	TRADD, TEST	Transfer to TRADD if (TEST) are minus
TRU	TRADD	Transfer to TRADD unconditionally
RPL	ADDR1, INSTR	Replace the 1st address in INSTR by ADDR1
XTP	FIRST, SECND	Give (SECND) the same power as (FIRST)
SRi	± n, ± lim	Set contents of R_i to ±n, limit to ± lim
TNi	TRADD, ± Δ	Augment R_i by ± Δ, transfer to TRADD
TXi	TRADD, ± Δ	Augment R_i by ± Δ, transfer to TRADD only if
		new $(R_i) < lim_i$. Otherwise proceed.
RPT	n, ± i, ± j, ± k	Repeat (perform) the next instruction n times, index-
		ing its 1st, 2nd, and 3rd addresses, as they exist,
		by i, j, and k words lengths respectively.
RWR	n, ± i, ± j, ± k	Reset PAC1 to zero, then operate same as RPT. ± i,
		± j and ± k may all be prefaced in RPT and RWR
		by an * to indicate indexing by number of char-
		acters, not word lengths.
LVE	TRADD	Leave PRINT. Next instruction is next 705 instruct-
		ion if TRADD is not written, TRADD if written.
BSi	n	Backspace tape i for n records.
RWi		Rewind tape i.
TMi		Write a tape mark on tape i.
WLi	UNIT, n, TRADD	Write a line. UNIT is tape t or printer. i is the
		space control after writing. n, TRADD is optional
		Write n lines, transfer to TRADD rather than
		write the (n+1)th line.
WHi	UNIT, n, TRADD	Write a heading. (Equivalent to WLi).
WCD	UNIT	Write a card. UNIT is either tape t or punch.
RCD	UNIT, TRADD	Read a card. UNIT is either tape t or printer.
		Transfer to TRADD on end-of-file condition.
		(Optional specification of TRADD).

Indexable computing operations

Arithmetic operations

These operations are largely self-explanatory from the operation summary preceding this section. It should be noted that MAD, MMA, PMA and MPM are compound, or double, arithmetic operations, although they are still written with only three positions in the variable field. The understood operand is always the contents of PAC1, the primary pseudo-accumulator. Although these accumulative operations may be used singly, their design purpose is for repetitive arithmetic. As such, the result of each operation may be found in PAC1 as well as in the normal result address. The Multiply-ADds are designed for vector products. The Polynomial-Multiply-Adds are designed for evaluation of polynomials with the argument addressed in the second position. Although no index register modification is shown in the summary, all of these operations may have a sub-position for each address, indicating this.

Mathematical function operations

These operations are also self-explanatory. SAC (Sine And Cosine) is the only operation with three positions in the variable field, all others having two positions. Each position may have a sub-position for index register indication. The first position address for all of these operations is that of the argument.

Data transmission operations

TMT (TransMiT), TAB (Transmit ABsolute), and TNA (Transmit Negative Absolute) are operations for moving blocks of data from one group of locations to another. The address in the first position of the variable field is that of the original location; the address in the second position is that of the location to which the data is moved. Both positions may have sub-positions for index register modification. Unless PAC1 is specified in either position it will be unaffected by the transmittal. Unless destroyed by a later operation, the original contents will be unaffected. TAB guarantees that the contents will be positively signed in the new location, TNA that they will be negatively signed. The first example shows the 40 numbers in locations M001 through M040 being transmitted in a reverse fashion, with a blank location between each number, to the locations P080 down to P002. The second example shows that ANY word-length block of characters may be transmitted by use of this instruction.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
BLNKS	CØN	(word length)	
P080	REG	P001	
	ENT		
	RPT	40, 1, -2	
	TMT	M001, P080	
	TMT	BLNKS, AREA	

Comparison transfer operations

TRC (TRansfer on Comparison) and TRE (TRansfer on Equality) are conditional transfer instructions which make an algebraic comparison of two operands. They are written with three positions in the variable field, the first of which is the address to be transferred to if the condition is met. TRC takes place when the number addressed in the second position is equal to or algebraically greater than the number addressed in the third position. TRE takes place only when these two numbers are equal. All three addresses may be modified by index registers.

These operations have special characteristics when preceded by a RPT or RWR operation. The number of repetitions may be set at a maximum by a positive number or to an indefinite repeat by a negative number in the first position of the RPT instruction. In either case, transfer may occur before the repeat tally is reduced to zero in normal fashion. The tally is therefore automatically reset to zero on a transfer. Considering for purposes of identification that the general instruction is:

TRC/TRE   TRADD, THIS, THAT

transfer to TRADD will occur when the contents of THIS, as indexed by the RPT, are greater than or equal to the contents of THAT, as also indexed. When the transfer occurs, the following quantities are left in specialized symbolic locations:

Location	*Contents*
ARG1 (working position)	Last THIS used
ARG2 " "	Last THAT used
LAR1 (Location of ARG1)	Address of last THIS used
LAR2 (Location of ARG2)	Address of last THAT used

LAR1 and LAR2 are usable only by the RPL operation. Using LAR1 or LAR2 as an address in any other instruction will cause an error message in pre-edit. None of these special addresses is indexable by either index registers or RPT or RWR instruction increments. Their index indicators are automatically set to zero by pre-edit. RPT or RWR increments, if used, must be coded as zero by the programmer or a machine stop will occur. When TRC or TRE is used with RPT or RWR the second position in RPT or RWR, which would normally be considered to modify the transfer address, must be coded as zero by the programmer or a machine stop will occur. Although TRC and TRE are indexable instructions, transfer addresses are obviously not indexable in a system using variable length instructions.

Table search operations

TSC (Table Search on Comparison) is a special variation of TRC which is especially designed for fast and flexible table search. Rather than a transfer address, the first position in the variable field contains the differential number of word lengths between the arguments of the table and the corresponding functions of these arguments. No transfer is made after TSC; the next instruction in sequence is executed.

A special RPT or RWR instruction must precede TSC. The first position contains a negative number for indefinte repetition. The second and fourth positions must contain zeros. The third position contains the interval of gross search. Table Search automatically consists of two parts. Letting N symbolize the gross search interval, the first part compares the first argument and successive arguments in intervals of N against the test argument. When one of these is found to equal or exceed the test argument, the search automatically backs up to the previous grouped argument. The second part of the search consists of a comparison of successive arguments in this localized area against the test argument in intervals of *one*. N may be the integer 1 or any other integer, PROVIDED that the number of arguments in the table equals (some multiple of this integer plus one). For example, consider the case of a table with 65 arguments. Valid values of N would be 16, 8, 4 and 2. The first, 17th, 33rd, etc. arguments would be compared against the test arguments if N were assigned as 16. In practice, the most effective interval of gross search is that which most closely approximates the square root of the number of arguments in the table, in this case, 8.

Further suppose that the 33rd argument was found to exceed the test argument. Comparison is now made to the 18th, 19th, 20th, etc., until some argument between the 17th and 33rd is found to exceed the test argument, or equal it. When this is found, the Table Search is discontinued and the following items are to be found:

If for any argument X_{test}, X_n is defined as the first argument greater than or equal to X_{test} and the previous argument X_{n-1} is less than X_{test}

Location		*Contents*
ARG1	(ARGument 1)	X_n
LAR1	(Location of ARgument 1)	Address of X_n argument
PAC1	(Pseudo-ACcumulator 1)	Corresponding function f (X_n)
ARG2	(ARGument 2)	X_{n-1}
LAR2	(Location of ARgument 2)	Address of X_{n-1} argument
PAC2	(Pseudo-ACcumulator 2)	Corresponding function f (X_{n-1})

All of these are useful as addresses, although LAR1 and LAR2 may be used only with the RPL operation, and none of the addresses is indexable. *Caution!* Arithmetic operations use PAC1 as a result address, so the contents of PAC1 after a TSC will have to be used before any arithmetic operation or else transmitted to a temporary location.

It should be standard practice to make the last argument in any table equal to the highest number in the PRINT system (+99+999999 . . . .). This dummy number ensures against overrunning the table with an unexpectedly high argument. In a table of 398 entries, for example, it would also be very practical to make the last 3 entries to be this dummy number, thus increasing the number of arguments to 401 and allowing a gross search interval of 20, which is the most efficient.

Let △ symbolize the number of words lengths between the table of arguments and the corresponding functions. If △ were set to zero, PACi would not contain the functions of the argument; the contents would be identical to the contents of ARGi, which are the arguments again. Sliding sets of tables might be constructed with this feature, using a variable △. Furthermore, using +△ in one case and −△ in another permits interchange of the dependent and independent variables.

A standard method of coding is shown in the example, illustrating Table Search and linear interpolation, according to the formula:

$$f(X_{test}) = f(X_{n-1}) + \frac{f(X_n) - f(X_{n-1})}{X_n - X_{n-1}} (X_{test} - X_{n-1})$$

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
	RPT	- 1, 0, (interval), 0	
	TSC	Δ, XSUB 1, XTEST	
	SUB	, PAC 2, TEMP 1	$f(X_n) - f(X_{n-1})$
	SUB	XTEST, ARG 2, TEMP 2	$X_{test} - X_{n-1}$
	SUB	ARG 1, ARG 2	$X_n - X_{n-1}$
	DIV	TEMP 2	(PAC I implied as divisor)
	PMA	PAC 2, TEMP 1, RSULT	$= F(X_{test})$

Non-indexable computing operations

Transfer operations

There are four conditional and one unconditional operations in this group. Conditional transfer commands are written with mnemonic symbol and two positions in the variable field. The address in the first position is always that of the instruction to which the transfer is to be made if the condition is met. The address in the second position is that of the number whose condition is to be tested. The mnemonic symbols TRZ, TNZ, TRP and TRM signify respectively that this condition is to be zero, non-zero, plus or minus. TRU signifies that transfer is to be made unconditionally to the instruction whose address is in the single position in the variable field. In the example shown, the program will operate in normal sequence if the contents of JONES is a positive non-zero number; otherwise control transfers to the instruction in B006 and proceeds sequentially from there.

SERIAL 1- -5	LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
03041		TRZ	B006, JØNES	
03042		TRM	B006, JØNES	
03043		SQR	JØNES, SMITH	

Replace operation

The instruction for this operation is written with the mnemonic RPL (RePLace) and two positions in the variable field. This operation causes the instruction in the address specified by the second position to have its first position address replaced by the first position address of the RPL instruction. If the first position address is written "LAR1" or "LAR2" the replacement address is not LARi but the address in LARi. This indirect address feature is used in conjunction with the TRC, TRE and TSC operations. This operation has three other usages. It may be used as a "flip-flop" or sequencing switch, for direct exiting from sub-routines and for command modification by replacement rather than by indexing.

RPL will operate only on the arithmetic, mathematical function and data transmission operations, all transfer operations, ATR, FXP and FPR. In addition, it will operate on the transfer addresses of the WLi, WHi, RCD, WTi and RTi operations, although these addresses are not in the first position of the variable field. The example shown depicts the condition of the instruction in TEXAS before and after a RPL. Further examples of usage may be found in Appendix II.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
TEXAS	MAD	CØMIC, 1, SMITH, 2, RSLT3	
	RPL	CAPS, TEXAS	
TEXAS	MAD	CAPS, 1, SMITH, 2, RSLT3	(new form of TEXAS)

Extract operation

The instruction for this operation is written with the mnemonic XTP (eXTract Power) and two positions in the variable field. The first position contains the address of the floating point number whose power is to be extracted. The second position contains the address of another floating point number whose power is to be replaced by the power extracted from the first number. This operation is designed for convergence testing, since in floating point the size of a number during the course of calculation may not be predicted. The example shown illustrates the programming of a convergence test on (JANES), where it is desired that the valid value of (JANES) shall not differ from the previous value of (JANES) by more than 3 in the 7th digit of the mantissa. The power of (JANES) is assigned to mantissa of .3 in TEST and scaled by 10^{-6}. After step CONV2 the number in TEST is the proper value for testing for convergence. When the difference between the present and previous value becomes less than this increment, the iterative loop is abridged by the TRC command. Without such an instruction, an oscillatory condition in the last digit of an iterated number might make it impossible to exit from the loop. It also provides for a less exacting matching than all of the digits in the mantissa. An appreciable speed-up of computing time may also be realized in slowly converging operations, if less stringent accuracies are made acceptable.

LOCATION 6- -10		OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
TEST		FLC	- 0 + 3	Power has no significance
SCAL	E	FLC	- 5 + 1	10^{-6}
RSTR	T	TMT	JANES, SMITH	Send old JANES to SMITH
		⋮		Compute new value for JANES
		⋮		with proper expression
CØNV	1	XTP	JANES, TEST	
CØNV	2	MPY	TEST, SCALE, TEST	
CØNV	3	SUB	JANES, SMITH	Differential in PAC I
CØNV	4	TAB		Absolute value of differential
CØNV	5	TRC	RSTRT,, TEST	To RSTRT if not converged

Set index register operations

The instruction for this operation is written with the mnemonic SRi (Set Register) and two positions in the variable field. The third character in the mnemonic symbol is written 1, 2 or 3, thus specifying the number of the index register to be set. It is set to the number of plus or minus word lengths in the first position.

The limit tally of that register is set to the number of word lengths written in the second position. If the second position is blank, the limit tally will automatically be set to zero. The limit tally is always a positive quantity and when converted (by multiplying by data word length, which is 2 plus the mantissa length) must be less than or equal to the memory capacity of the 705 minus 10,000.

Non-test transfer operations

The instructions for these operations are written with the mnemonic TNi

(Transfer No test) and two positions in the variable field. The third character in the mnemonic symbol is written 1, 2 or 3, thus specifying the index register to be operated upon. The first position contains the address of the instruction to which unconditional transfer is made after augmenting the contents of the index register with the number of word lengths written in the second position. This transfer address will, in many cases, merely be the next instruction. The increment for the index register may be either plus or minus; the minus sign is written before the number and no sign is written for plus numbers.

Test transfer operations

The instructions for these operations are written with the mnemonic TXi (Transfer testing indeX) and two positions in the variable field. The third character in the mnemonic symbol is written 1, 2 or 3, thus specifying the index register to be operated upon. This operation functions in the same manner as TNi except that the transfer (to the instruction whose address is specified in the first position) is nullified if the contents of the index register, as now incremented, are equal to or greater than the limit tally previously specified. If this is so, the program does not transfer but rather proceeds to the next instruction in sequence.

Repeat operations

The instructions for these operations are written with the mnemonic RPT (RePeaT) or RWR (Repeat With Reset) and four positions in the variable field. RWR is equivalent to RPT except that PAC1 (the primary pseudo-accumulator) is reset to zero. A RPT signifies that the next instruction in sequence is to be repeated (i.e. performed) the number of times specified in the first position of the RPT. This instruction is to be performed as written the first time, but for each repetition the numbers or addresses in the first, second and third positions of that next instruction are to be additionally augmented by the numbers respectively in the second, third and fourth positions of the RPT. If the next instruction does not have a third address, it is not necessary to specify a fourth position for RPT.

RPT and RWR apply only to the next instruction, not to any sequence of instructions. Their purpose is to both minimize the number of instructions written by the programmer and reduce operating time on repetitive instructions. This is accomplished by letting the executive routine know in advance that the next instruction is repetitive so that interpretation and command fabrication is performed only once.

Indexing by RPT is secondary and subordinate to indexing by the contents of index registers and the simultaneous use of both is possible. All four positions of the variable field may be written as 1 or 2 digit numbers and all may be signed both plus and minus. The first position, however, is normally plus, for when it is signed minus it indicates indefinite repeat and as such must be used with caution. Indefinite RPT is designed to be used with the TRC, TRE and TSC operations. When an exit is made for any of these, the repeat tally is automatically reset to zero so as not to influence the next instruction in sequence. A leading asterisk in any of the second through fourth positions indicates that incrementation will be by that integer number rather than by that number of word lengths. This is mainly used for indexing the card column on FLO and the decimal position in the type wheels for FXP.

All indexable instructions and only indexable instructions are repeatable. Each of these interrogates the number in the first position (serving as a count) before storing the result. If this is non-zero, the count is reduced by 1 and the operation is automatically repeated with further indexing by the RPT increments. If is it zero, it signifies either that the instruction was not intended to be repeated or that it has been performed for the last time. In either case, the program proceeds to the next instruction in sequence. For the MAD, MMA, PMA and MPA instructions, a special condition exists under RPT control. If the address to which the result is sent is not indexed by the RPT, the result is stored intermediately in PAC1 only, and not sent to the result storage until the repeat tally is zero.

Advantage may be taken of the fact that RPT and RWR do not alter the contents of PAC1. The following example illustrates the calculation of

$\left(\frac{A}{B}\right)^n$, n being an integer, which result is then available in PAC1.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
TEMP	REG		
	ENT		
	DIV	LØCA, LØCB, TEMP	
	RPT	(n-1)	
	MPY	TEMP	

Switching operation

The instruction for this operation is written with the mnemonic ATR (Alternating TRansfer) and two positions in the variable field, each of which is tagged. In operation, an unconditional transfer is made to the address in the first position each time the instruction is executed, up to the number of times designated by the tag for that position. After this limit is reached, succeeding executions of this instruction cause unconditional transfer to the address in the second position, up to the number of times designated by its tag. The instruction then reverts to the original condition for further alternation as required. Execution is dynamic and it is impossible to return to the initial condition without performing the entire cycle; thus a conditional transfer exit from the cycle destroys the utility of the ATR unless the program is read in again to restore the initial conditions.

Tags for both positions are unsigned positive numbers, from 1 to 400. A zero is an illegal tag for which pre-edit will substitute a 1. For purposes of counting only, this operation is generally more efficient than using index registers with their transfer instructions.

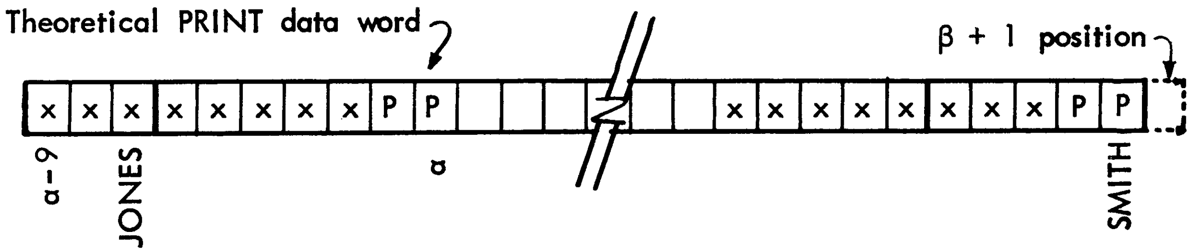
The first example shows the preferable, but not the only, method for executing *n* times the routine commencing with the operation in the address "START". The second example illustrates a method for simulating the general extended case when the desired tags for both addresses exceed the limit 400 and are not prime.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
START			
	ATR	START, (n-1), NXTCM, 1	
NXTCM			
START	ATR	FIRST, a, SECOND, c	SIMULATES:
FIRST			
			ATR FIRST, ab, SECND, cd
	ATR	FIRST, (b-1), START, 1	
SECND			WHERE:
			ab > 400, NOT PRIME
	ATR	SECND, (d-1), START, 1	cd > 400, NOT PRIME

Generating print instructions

As programmers become more experienced in using the PRINT system, the translation charts on pages 51 and 52 will become increasingly useful. PRINT instructions, due to their variable length and specialized format for maximum operating speeds, may not be modified directly except by indexing. Very often specific coding for a problem will depend upon parameter values. An excellent example of this is the martix inversion kernel in Appendix II. Rather than recode the problem each time for a different order of matrix and a different number of column vectors, it would be advantageous to have 705 instructions preceding the general coding which would generate the necessary variable portions of the PRINT instructions, given only the values of *n* and *b*. To do this, the structure of the translated PRINT instructions must be known.

As an example, consider the tape instructions RTi and WTi, which are specifically designed to operate with PRINT data words in fixed lengths. The coding kernel below shows how to make use of these same instructions to write and read irregular blocks of memory on tape, taking advantage of the error-correction routines contained in these PRINT instructions. The record as formed is illustrated below:



LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
JØNES	BLK	3	
	⋮		
SMITH	BLK	5	
BRØWN	ADC	JØNES - 2	α - 9 portion
BLACK	ADC	SMITH + 1, 9	β + 1 portion
	LØD	BRØWN, 4	
	UNL	PUTT + 12, 4	
	UNL	TAKE + 12, 4	
	LØD	BLACK, 4	
	UNL	PUTT + 16, 4	
	ENT		
PUTT	WT8	0000, 0000	Zeros are dummy addresses, later replaced
	⋮		
TAKE	RT8	0000	Zeros are dummy addresses, later replaced

Fixed symbolic locations

Actual location for the fixed symbolic data words defined on page 4 are:

	Address of right-hand digit		
Mantissa:	*8-digit*	*10-digit*	*12-digit*
PAC1	0254	0256	0258
PAC2	0244	0244	0244
ARG1	2439	2441	2443
ARG2	2449	2453	2457
LAR1	2044	2044	2044
LAR2	2019	2019	2019

In addition, all line, heading and card image positions are considered fixed symbolic locations if addressed as:

TWxxx . . . . . TW stands for type wheel, xxx for 001 to 120
HDxxx . . . . . HD stands for heading, xxx for 001 to 120
CØLxx . . . . . CØL stands for column, xx for 01 to 80

Locations of these images are:

LINE	3265-3385	Address=TW+3265
⌗	3386	
HDG	3387-3507	Address=HD+3387
⌗	3508	
CARD	3509-3588	Address=CØL+3508
⌗	3589	

The position referenced in the coding is considered as:

1. BADD for PRINT data words as called for in PRINT instructions.

   xxx. . . .$\overset{\pm}{\text{xxx}}$ P$\overset{\pm}{\text{P}}$ Pre-edit must make conversions of BADD as required for the several types of PRINT instructions.

2. The addressed character for 705 commands.

   Examples: UNL CØL06=UNL (3271)
   ST TW086+4, 15

Image addresses must contain all five characters. In the above examples, CØL6 or TW86 would not be allowable.

Input–output operations

Printing operations

Much emphasis has been placed, within the PRINT system, upon ease of entering data and recording results. Both on-line and off-line operations are equally feasible. There are several operations especially designed for simplified control of printed output.

FXP (FiX for Printing) and FPR (Fix for Printing Rounded) are indexable operations which will convert the floating point number in a specified address to a fixed decimal condition and store it in the specified position in the line image in memory just as it should be printed.

The variable field of these instructions consists of the address of the number to be converted, index tag (if any), a 1 to 3 digit type wheel position for the decimal point, a 1 or 2 digit maximum number of expected whole numbers and a W, the number of decimal positions and a D, and a 1 or 2 digit power of 10 by which the number is to be scaled. The last position should be left blank if there is no scaling. (The heading should note this scaling if it exists). Typical commands and results are:

FXP JONES, 28, 2W, 10D, –2	$123456787\overset{+}{8}0\overset{+}{3}$	bl. 2345678780b
FPR G116, 13, 9, 4W, 2D	$438692617\overset{-}{8}0\overset{+}{4}$	4386.93–
FPR G116, 3, 4W, 2D	$438692617\overset{-}{8}0\overset{+}{4}$	86.93– (ERROR)

In the first example, 2, 20, and +20 would have been acceptable as scale factors. The error shown is due to calling for 4 whole numbers to the left of type wheel 3. Similar errors can occur by exceeding type wheel 120. Only the address in the first position is modifiable by index registers. Both the address and the type wheel position are modifiable by RPT or RWR. An example is shown where more than one number is converted to the same specifications; this will occur quite often in matrix output. In the example, (RØW01) are fixed with the decimal point in type wheel image 8, (RØW01+1) with the decimal point in type wheel image 19, etc., thus using only three instructions to convert the entire line and print it. 0W and 0D must be used to indicate absence of either whole numbers or decimals.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
	RPT	8, 1, *11	
	FXP	RØW01, 1, 8, 1W, 6D	
	WLS	PRINTER	

There are many combinatorial instructions for output control, as illustrated. All are non-indexable. Either the printer or a tape unit may be specified in the variable field. Either the full word description or the initials P or T may be used. An asterisk as the preceding character signifies a fast skip under carriage tape control.

Both on-line and off-line printers must be set to program control to use the PRINT I system. Except for special skip instructions, there should be a punch in only channel 1 of the carriage tape; a channel 12 punch is illegal and will cause an error message. All carriage control should be built into the program to eliminate most tape-changing.

Headings are put into the heading image with the HDG entries; for multiple usage of heading in a single run, reserve extra positions for heading components in memory with CON operations, and transmit these to the heading image as required. When a line is written successfully, without echo checks or other errors, the line image is erased to blanks. This allows a flexible line format, as the programmer makes provision for positioning only those numbers which he wishes to print, regardless of the make-up of the previous line. Card and heading images are NOT erased after writing.

WLN	PRINTER	Write a Line - No spacing
WLS	PRINTER	Write a Line - Single spacing
WLD	TAPE 4	Write a Line - Double spacing
WHD	T 7	Write Heading - Double spacing
WHT	P	Write Heading - Triple spacing
WL 2	TAPE 7	Write a Line - Skip to channel 2 punch
WH 4	* TAPE 8	Write Heading - Fast skip to channel 4

These operations may have counting transfers added by using two more positions in the variable field. The second position is a 1 or 2 digit number specifying the number of times the write line or heading operation is to be executed, up to a limit of 98. The third position contains the address of the instruction to which control is to be transferred when writing is attempted after the limit is reached. This transfer restores the initial condition of the instruction so that the same process may be repeated. The following example shows the control operations for writing 20 pages, each with a heading and 50 lines of answers, grouped in 5 groups of 10.

LOCATION 6- -10		OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
HEAD		WHT	T6, 20, PAGES	PAGES, LINES and LASTL are used
CØMP	U	— — —	- - - -	as convenient mnemonic names for
		WLS	T6, 9, LINES	the associated instructions. The first
		TRU	CØMPU	line therefore reads:
LINE	S	WLD	T6, 4, LASTL	
		TRU	CØMPU	"Write a Heading, Triple space,
LAST	L	WLI	T6	on Tape 6 – write 20 PAGES."
		TRU	HEAD	
PAGE	S			(continues computation after 20 pages are written)

Tape data storage operations

WTi (Write Tape) and RTi (Read Tape) are indexable operations provided for storage and retrieval of data words on tape, which is essentially a function of increasing memory capacity. The third character of the mnemonic symbol is the dial setting of the tape unit addressed. WTi is written with two positions in the variable field, each of which may have an index register tag. These positions contain the first and last addresses of a consecutive series of words in memory which are to now constitute a record on tape. RTi is written with one position in the variable field, which is the starting address in memory for reading one record from tape. As many words will be replaced in memory as the record itself contains, so the programmer is cautioned to know the pattern of his tape operations very thoroughly, to avoid destruction of wanted data. Discretion should also be maintained in using these instructions with the 4 tape units normally associated with pre-edit and library.

It is possible to add other positions in the variable field of WTi and RTi.

The third position may be a transfer address or the two letters TM. The fourth position must be TM, and exist only if there is an address in the third position. The transfer address of WTi is that of a sequence of instructions defining procedure in case the physical end of tape is reached before completing the write instruction. TM puts a tape mark as the next record after completion of writing. The transfer address of RTi is that of a sequence of instructions defining procedure in case where the record consists of a tape mark written by a WTi.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
	WT6	B001, B020	
	WT6	B001, 1, B020, 12, PATCH	
	WT5	G136, 1, G136, 12, TM	
	RT5	RAND	
	RT9	FIRST, TRØUT	
	BS8	20	(backspace tape 8 by 20 records)
	RW8		(rewind tape 8)
	TM8		(write a tape mark on tape 8)

BSi (BackSpace tape) and RWi (ReWind tape) are non-indexable operations for positioning records to be read or written. In the variable field of BSi is written the number of records to be backspaced. This is a 1 to 3 digit number; the programmer may not specify more records than exist on the tape from that point back. RWi has no information in the variable field, nor does TMi, which is a separate instruction for writing a tape mark unconnected with other operations.

Card operations

The card image in memory is used for both reading and punching. Special facilities are provided for reading both floating point and fixed point data, but punching is restricted to floating point form unless special handling is made in 705 language. This is based on the assumption that actual punched cards will be produced for local re-loading only, in which case there is no purpose in refloating data which may be had already in floating form. Suggested floating point loader cards are as follow:

8 digit mantissa	8 words per card	addressed at 10(10)80
10 digit mantissa	6 words per card	addressed at 20(12)80
12 digit mantissa	5 words per card	addressed at 24(14)80

The card for the 8 digit mantissa may be reduced at option to 7 words, thus allowing the first 8 columns in any system to be indicative information. For fixed data, there is no specified format and commands are so designed that it is not necessary. Typical movement of data and production of a punched card might be:

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
	RPT	7, 1, *10	
	TMT	A001, CØL20	
	WCD	T4	

WCD (Write CarD) and RCD (Read CarD) are operations for reading and writing 80 character records. Reading may be from either the card reader or a numbered tape unit, and this is written in the variable field. Writing is onto either the punch or a numbered tape unit, and this is specified in the variable field. The use of tape for these operations is designed for peripheral equipment, although it is another method of temporary data storage in fixed decimal format. The unit in the variable field may be written with the full name or the initials T, P or R as required. The transmission to or from the card image in memory is implicit in all of these instructions. The second position of RCD is a transfer address for an end-of-file condition.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
	WCD	TAPE 8	
	WCD	PUNCH	
	WCD	P	
	RCD	T6	
	RCD	READER, TRADD	

FLO (FLOat) is an operation for converting a fixed point number of any specified length to floating point form, thus making it suitable as an operand in the PRINT system. It is written with these four positions in the variable field:

1. The symbolic address of the units position of the number to be converted. This will most often be CØLxx. A sign for this number must exist over the units position for negative numbers only.

2. The number of digits comprising the number. Must be <2 mantissa lengths.

3. The direction (L or R) for shifting the decimal point to put it in the true position and the number of places to shift, considering the number to be originally comprised of whole numbers. (See examples). For no shift, either L0 or R0 must be coded.

4. The symbolic location where the number is to be stored after conversion, with an index register tag if required.

Only the address in the fourth position is indexable by the contents of index registers, but both it and the address in the first position may be indexed by a RPT instruction. The first position address will most commonly be CØLxx, and the RPT increment will most commonly be asterisked to indicate number of character positions rather than word lengths. The indications in the comments field of the examples show the true decimalization of the numbers. The third example is a program for reading in 200 pieces of 4 digit data for processing, condensed for loading purposes on 10 cards. When converted to floating point form, the data words occupy PRINT memory addresses A001 to A200 inclusive.

SERIAL 1- -5	LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
		RPT	4,1,1	
		FLØ	FXWRD,6,R2,INPUT,3 XXXXXX00.	
		RPT	6,*5,1	
		FLØ	CØL48,5,L2,TAX4 XXX.XX	
01010	A200	REG	A001	
01020		SR1	0,200	
01030	RDATA	RCD	READER	
01040		RPT	20,*4,1	
01050		FLØ	CØL04,4,L3,A001,1 X.XXX	
01060		TX1	RDATA,20	

Pre-edit and system entry

If a system tape does not exist by virtue of previous usage of the PRINT I system, operations are commenced by placing the PRINT I program deck in the card reader, followed by symbolic cards for the program to be pre-edited. The system is then initiated from the console by:

	Address selector	*Typewriter key*
1. Clear memory		
2. Place in manual instruct status		
3. Select the card reader	0100	2
4. Read into lowest memory position	0000	Y
5. Depress the start key		

After loading these cards, the PRINT I system will be on tape 0200, in 9 sections:

1. System control 1
2. Memory print (13 records)
3. Tape print
4. First pass of pre-edit
5. Intermediate pass of pre-edit
6. Last pass of pre-edit
7. System control 2
8. PRINT I executive routine
9. Non-standard library

If a system tape does exist, it is loaded on tape 0200, and the system initiated from the console by:

	Address selector	*Typewriter key*
1. Clear memory		
2. Place in manual instruct status		
3. Select the system tape unit	0200	2
4. Rewind the system tape	0002	3
5. Turn off IOF	0000	3
6. Read into lowest memory position	"	Y
7. Depress the start key		

Programs may be pre-edited from card, tape or combined card and tape input. Tape input will be on 0202 if Alteration Switch 0915 is OFF, on 0203 if it is ON. The combined card and tape input feature exists for purposes of repairing programs. Steps may be inserted or deleted by change cards. A complete reorganization of the program within memory takes place every time this is done.

Pre-editing starts with system control 1 and proceeds to the first pass. Card and tape input are checked for sequence and merged on serial number (col. 1-5 of the card). When the serial of a card matches that of a card image on tape, the card record replaces that tape record unless the card carries the mnemonic operation code DEL (DELete). In this case, that record is deleted and so are all subsequent records up to and including the record whose serial is punched in the variable field of the DEL card. Tape records having DEL for operation and DEL cards without matching tape records are both deleted. Non-DEL cards whose serials do not match with any card images on tape are collated with them.

The output of the first pass is on 0202 or 0203. Records contain the actual locations of the instructions. Other conversion is deferred until the last pass. In the event that the assignment table overflows available memory, the overflow blocks will be on 0201. The intermediate pass is executed only when 3 or more overflow blocks occur; this pass finds actual addresses for the symbolic addresses referring to the overflow blocks. When there are 3 or less blocks of the assignment table yet to be searched, the last pass is called into memory for operation. This last pass completes conversion of the mnemonic intructions, writing:

1. A program tape on 0201, consisting of actual 705 instructions and converted pseudo-instructions. If Alteration Switch 0913 is ON, standard 705 load cards will be punched for reloading by card rather than the 0201 tape. This is

suitable for short programs, or where a permanent record of the program is desired for storage in a more flexible medium. This should be done only when the program is known to be correct and working.

2. A master symbolic tape on 0203 (or 0202 for referrals > 1000) which contains the corrected and updated program in symbolic form, just as the original punched cards were. This is suitable as input for re-editing and further correction. The selection of either 0203 or 0202 as the proper input tape for re-editing is automatic. A concluding typewriter message will indicate the correct location of tapes.

3. A listing tape on 0202 (or 0203 for referrals > 1000) which is the permanent record of coding, pre-editing and assembly of the program. If Alteration Switch 0914 is ON, the listing will be written on the line printer during the pre-edit process, in which case this tape need not be saved unless more copies are desired. If the switch is OFF, the availability of an auxiliary printer is normally assumed. For both printing methods, time will be lost if the comments exceed 25 characters, since an extra line will be printed just to accommodate the overflow. Comment characters above 50 will not be printed on this listing.

If the program is to be executed without pre-editing, Alteration Switches 0911 and 0912 are both OFF, thus diverting to system control 2. Alteration Switch 0913 is then interrogated. If it is ON, the edited program will be read from punched cards; if it is OFF, the program will be read from tape 0201. The combination of both card and tape input is not possible here. System control 2 reads the edited program and the executive routine into memory, activating a typewriter message calling for setting Alteration Switches for the program to be executed and stopping on HLT 1111. Depressing the start key will cause execution of the program starting at the first instruction, which is either a 705 instruction or the compiled ENTer instruction in PRINT. If the 0902 indicator is turned on, loading is in error. Press the start key to reload from tape. Cards must be reloaded; reset, start and read again. Pre-edit will blank unused memory before reading in the program.

Memory print and tape print routines are incorporated in the system tape. They are called into use by setting Alteration Switch 0916 ON and depressing the reset and start keys. A typewriter message will give instructions for next setting of 0916 (OFF to bypass memory print). The tape to be printed is selected by setting Alteration Switches 0911 thru 0914 in a binary representation of the units position of the tape unit desired, as:

Alteration switch	*Value if ON*	*Value if OFF*
0911	8	0
0912	4	0
0913	2	0
0914	1	0

For example, if 0912 and 0914 were the only switches ON, it would signify that tape 0205 would be printed, as $4 + 1 = 5$. Configurations which sum more than 9 are in error. Printing of the tape selected continues until a tape mark is sensed or operation is changed to manual. Tapes may be selected and printed successively but in any order, by changing the Alteration Switch configuration and depressing the start key each time. The return to system control 1 is effected by turning 0916 OFF, reset and start.

Pre-edit conversion

Two types of addresses are recognized by pre-edit. The basic address of a floating point data word is that of its highest (or right-hand) memory position. Pre-edit allocates memory in $(m + 2)$ modules, where m is the mantissa length. FLC and REG are the two operations which cause memory to be reserved this way.

The basic address of a PRINT pseudo-instruction, which is of variable length, is that of its lowest (or left-hand) memory position. BADD = (the basic address of the previous instruction) + (the length of the previous instruction), since they are normally obeyed in order of ascending memory position.

When either REG or FLC is encountered by pre-edit, a test is made to see if the previous instruction was either REG or FLC. If not, (and a previous ORG falls in this category), the location counter leaves a blank preceding the entry to insure definition of a numeric field. If an initial origin is supplied in the program it will take precedence over the standard origin supplied by pre-edit, which follows immediately after the executive and loading routines.

The typewriter may operate during pre-edit to send error messages about system restrictions which have been ignored in coding. Each message is identified with the serial and symbolic location of the erroneous instruction. Some of these are for:

1. RPT or RWR tally > 99.

2. Non-repeatable instruction following a RPT or RWR.

3. Actual address for 705 instructions TR and 00 TMT not ending in 4 or 9.

4. Infraction of rules for symbolic or actual location addresses.

5. Minus index limit for any register, or a converted limit > (memory—10,000).

6. More than 2 HDG cards.

7. Problem overflows memory.

8. Non-PRINT or non-705 operation codes.

9. Attempting to increment non-indexable address (i.e. PAC1, PAC2, etc.)

10. ATR tally greater than 400.

11. Non-indexable entry tagged (i.e. PAC1, decimal location in FLO, etc.)

There may be instances when the programmer has a definite and legitimate purpose in ignoring these restrictions. Error messages do not necessarily indicate that revision must be made; they exist to warn the programmer to be certain that this was his true intent.

When a floating sub-routine symbol is coded, the pre-edit knows that the symbol has no assigned operation code number in the table of correspondence. The operation code for all floating sub-routines is assigned to it (this code comes from the last two digits of the address of the first instruction in the FSR area). Pre-edit automatically compiles the 705 instructions necessary to bring the proper sub-routine (if it exists on the library tape 0200) into the floating position in PRINT during the course of computation. Such a linkage is compiled only the first time that function is needed, or if another function has superseded it before it was to be used again. The criterion for compiling the linkage is thus change of requirement only. If only one non-standard sub-routine is used for a particular problem, the net effect is as though it were a permanent component of the executive routine in memory.

No floating sub-routines will be furnished with this manual. They are primarily the responsibility of the user, although IBM will distribute any routine contributed. The "tinkertoy" appendix in the supplement will show various means of extending this feature so that the programmer may specify the amount of memory he is willing to expend for floating sub-routines. Replacement would then be set up only if the desired sub-routine exceeded in size the amount of available specified memory left.

Summary—System operation

Tape Assignments	0200–System Tape	0202–Listing
	0201–Actual Program	0203–Symbolic (updated)

Function	*Operation*	*Alteration Switches* 0911	0912	0913	0914	0915	0916[1]
INPUT.	Card and tape 0202	ON	ON			OFF	OFF
	Card and tape 0203	ON	ON			ON	OFF
	Card	ON	OFF				OFF
	Tape 0202	OFF	ON			OFF	OFF
	Tape 0203	OFF	ON			ON	OFF
OUTPUT . . .	On-line printer listing[2]	–	–		ON		
	On-line punched program[2]	–	–	ON			
	Memory and tape print[3]	8	4	2	1		ON-(ON)
	Memory print only[4]						ON-ON
	Tape print only[5]	8	4	2	1		ON-OFF
PROGRAM . .	Start key[6]						
	Start key	(As required for subject program)					
PROGRAM . . (Without pre-edit)	Card-loaded program	OFF	OFF	ON			OFF
	Tape-loaded program	OFF	OFF	OFF			OFF

[1]Alteration Switch 0916 must be OFF if re-entry to system is by reset and start, except as noted under memory and tape print instructions.

[2]These will be on tapes 0202 and 0201 respectively, regardless of settings.

[3]Reset-start. Start again after typed message. Memory print will occur first, then a tape print for each start until 0916 is turned OFF. Select tape units by binary representation, in 0911 thru 0914, of units digit of desired unit.

[4]Reset-start. Start again after typed message. Turn 0916 OFF, reset and start to return to system control 1.

[5]Reset-start. Turn 0916 OFF after typed message. Select tape unit by Alteration Switch combination. Selected tapes will print for each start until reset.

[6]Brings executive routine and subject program into memory to operate. After typed message and HLT 1111, set switches as required and depress start key.

Appendix I—Operation execution times

The execution times for certain operations are given here to indicate the general speed range for the PRINT I system, in running time. It should be noted that these times cannot reflect elapsed problem time, and that in general they bear the burden of flexibility and convenience. As the system is still in process, final times for other mantissa lengths than those shown here may be expected to vary. The final manual will contain a complete list of guaranteed execution times for all operations not associated with input-output equipment.

The times given here are for complete multi-address operation and include all interpretation and miscellaneous times. All times are given, in milliseconds, for a 10 digit mantissa system, except as noted for 8 digit.

	Non-zero operands		*Zero operands*	
	Single execution	*10 time average*	*Single execution*	*10 time average*
ADD, SUB	4.9	4.2	3.1	2.4
MPY, MMY	7.1	6.3	4.0	3.2
DIV, MDV	18.6	17.5	3.1	2.0
PMA, MPM	8.6	7.4		
MAD, MMA	8.8	7.6		

SQR	16.4 (8 digit)
ART	26.9 (8 digit)
EXE	20.1 (8 digit)
EXD	18.0 (8 digit)
SAC	25.7 (8 digit)
LGD	13.9
LGE	16.8

ENT	2.2	TMT	1.9
LVE	.8	TAB, TNA	2.1
TRU	.8	XTP	1.1
TRZ, TNZ, TRP, TRM	1.4	RPL	1.7
TRC	2.3	SRi	1.3
ATR	1.5	TXi, TNi	1.0
RPT, RWR	1.3		

Translation chart for non-indexable (non-repeatable) operations

	0283	0284	0285	0286	0287	0288	0289	0290	0291	0292	0293	0294	0295	0296	0297	0298	0299	0300	COMMENTS
	0	$\overset{+}{4}$	This operation sacrificed for system control.																
	0	$\overset{+}{9}$																	
	1	$\overset{+}{4}$																	
TRM TNZ	1	$\overset{+}{9}$	α x	$\overset{+}{x}$	$\overset{+}{x}$	x	β − 2 x	x	$\overset{\pm}{x}$	x	M N								Tens position of β − 2 is signed + for TRP and TRZ, signed − for TRM and TNZ
TRU	2	$\overset{+}{4}$																	
RPT	2	$\overset{+}{9}$	n − 1 x	$\overset{\pm}{x}$	i x	x	x	x	j x	x	x	x	k x	x	x	x	‡		n − 1 is either $\overset{\pm}{xx}$ or $0\bar{1}$ for indefinite RPT. i, j and k follow the rules for SRi except = actual number only when asterisked.
RWR	3	$\overset{+}{4}$																	
TN1 TX1	3	$\overset{+}{9}$	α x	$\overset{+}{x}$	$\overset{+}{x}$	x	1 2	Increment x	x	x	x								α = basic address (BADD) of the command to which transfer is made. Increment follows the rules for SRi.
TN2 TX2	4	$\overset{+}{4}$					1 2												
TN3 TX3	4	$\overset{+}{9}$					1 2												
RPL	5	$\overset{+}{4}$	α − 9/11/13 x	$\bar{x}$	x	$\overset{+}{x}$	β + 2 x	x	x	x									β = command basic address. If RPL is indirect for LAR 1 or LAR 2, the α position is replaced by $204\bar{4}$ or $201\bar{9}$, respectively.
XTP	5	$\overset{+}{9}$	α − 1 x	$\bar{x}$	$\overset{+}{x}$	x	β − 1 x	x	x	x									α and β are basic addresses of data words.
ATR	6	$\overset{+}{4}$	α x	$\overset{+}{x}$	$\overset{+}{x}$	x	i x	x	β x	$\overset{+}{x}$	$\overset{+}{x}$	x	j x	x	tally 0	0			i and j are zoned in the 10s position for count up to 399, in ADM collating sequence.
WLi WHi	6	$\overset{+}{9}$	0	$\overset{+}{0}$ or	$\overset{+}{0}$	0 (if not specified)	LC+1 x	$\overset{+}{x}$	tally 0	$\overset{+}{0}$	unit x	0	$\overset{\pm}{x}$	2 3	6 8	N G	u	s	s is the spacing control character. u is 4 for tape, 5 for printer. UNIT is 20i or 400, with units pos. zoned − for triple space, + for none, single or double. Line count (LC + 1) is $0\bar{0}$ if not specified.
BSi RWi TMi	7	$\overset{+}{4}$	i	3	0	0	0	D B A	n 0	n 0	n A								
CD punch	7	$\overset{+}{9}$	0	$\overset{+}{0}$	$\overset{+}{0}$	0	3	0	0	R	9								
CD reader			x	$\overset{+}{x}$	$\overset{+}{x}$	x	1			Y	4								
SR1	8	$\overset{+}{4}$	− limit x	x	x	x	‡	set to x	x	x	x								set = f(first position), − limit = f(second position). Both are unsigned true (or 40,000 complements) products of (number) times the (word length).
SR2	8	$\overset{+}{9}$																	
SR3	9	$\overset{+}{4}$																	
LVE	9	$\overset{+}{9}$	1	α x	x	x	x												α is first following 705 command location if not specified.

Translation chart for indexable (repeatable) operations

	0283	0284	0285 0286 0287 0288	0289 0290 0291	0292 0293 0294 0295	0296 0297 0298 0299	0300	COMMENTS
ADD / SUB	0	$\bar{4}$	α - 9/11/13	α_i β_i γ_i	β - 9/11/13	γ - 9/11/13	H / Q	
MPY / MMY	0	$\bar{9}$	x $\bar{x}$ x x ↓	x x x ↓	x $\bar{x}$ x x ↓	x x x x ↓	H / Q	
DIV / MDV	1	$\bar{4}$	↓	↓	↓	↓	H / Q	
MAD / MMA	1	$\bar{9}$	↓	↓	↓	↓	H / Q	
PMA / MPM	2	$\bar{4}$	↓	↓	↓	↓	H / Q	
SAC	2	$\bar{9}$	↓	↓	x x x x	↓		
SQR	3	$\bar{4}$	↓	x x 0	↓	0		
LGD / LGE	3	$\bar{9}$	↓	↓	↓	0 / 1		
EXD / EXE	4	$\bar{4}$	↓	↓	↓	0 / 1		
ART	4	$\bar{9}$	↓	↓	↓	0		
(FSR)	5	$\bar{4}$	↓	↓	↓	0		
TMT / \B TNA	5	$\bar{9}$	↓	↓	↓	0 / s		s is a + sign for TAB, a - sign for TNA
i WCDt / i RCDt	6	$\bar{4}$	γ x $\overset{+}{x}$ $\overset{+}{x}$ x	t α_i β_i	α - 9/11/13 x x x x	x $\bar{x}$ x x / 0 0 0 0 (β + 1)	t / t	t = $\bar{0}$ for (TM)(WCD + WTi), = $\bar{0}$ for RTi + RCD + (TM)(WCD + WTi)
	6	$\bar{9}$						γ is the BADD of the next command if not specified. α address is xxxx for WTi and RTi, 3509 for RCD and WCD. β+1 is x$\bar{x}\mathring{x}$x for WTi, 3NY9 for WCD. α_i =0 for RCD+WCD, x for RTi+WTi. β_i =0 for RCD+WCD+RTi, x for WTi.
	7	$\bar{4}$						
	7	$\bar{9}$			Δ(word length) + 19/21/23			
TSC / TRC	8	$\bar{4}$	$\overset{+}{x}$ $\bar{x}$ x x	0 x 0	β - 9/11/13	γ - 9/11/13	A	
TRE	8	$\bar{9}$	α x $\overset{+}{x}$ $\overset{+}{x}$ x	0 x x	x $\bar{x}$ x x	x $\bar{x}$ x x	1	M-L or M+R
FLO	9	$\bar{4}$	α x x x x	0 x 0	β - 9/11/13 x x x x	N x $\overset{+}{x}$ x $\overset{\pm}{x}$		written: FLO, α units, N, R or L n, β, β index; |L or R| < 80 N ≤ 2 mantissa lengths
FXP / FPR	9	$\bar{9}$	α - 9/11/13 x $\bar{x}$ x x	x $\bar{0}$/$\bar{0}$ 0	F x $\overset{\pm}{x}$; TW+265 x x $\overset{+}{x}$	D+1 x $\overset{+}{x}$; D+W-M+1 x $\overset{\pm}{x}$		command is written: FXP α, α index, TW, wW, dD, F

PRINT I 8 DIGIT MANTISSA

14019	1	1	10404	10434	10369	10359	10909	10939	10974
12479	10494	12854	13064	13974	12944	10784	10819	10849	10289
11314	11324	11379	11349	11339	10534	10529	10529	10529	10529
10529	10529	12574	1	1	1	11974	11964	11124	12244
0007	0009	001 | 0 | 011	0014	0017	0018	•00A	(12) ¦ (10) ¦	PAC2	PAC1
PAC1	(10) ¦ (12) ¦ ‡	C | W.L. 1 0̄ | 2 0	legal zero 0 0 0 0	0 0 0̄ 0 0̄	0 0̄ 0 0̄ ‡ Z	0 H | A	STANDARD COMMAND POSITION		
S C P | 0 0̄ 8 C (m)	n-1 0̄ 0̄ | 0 0 0 0 (i)	0 0 0 0 (j)	0 0 0 0 ‡ (k)	FETCH 6 0 3 T 4	U 0 2 8 3	9 (BADD)	Q 0 K H 4	M 0 Ø B 4	7 0 L E D
1 0 0̄	RWR U 0 2 4 9	9 0 2 6 9	RPT U 0 3 0 9	9 0 2 8 9	8 0 T / 0	8 0 T J 4	8 0 T A 8	8 0 2 S 2	1 0 3 2 4
TRP, M, Z, NZ T 0 2 9 1	8 0 B R 3	M 0 4 4 9	7 0 D K 5	H	0 4 6 4	TRU U 0 3 3 1	9 0 2 Y 5	1 0 3 2 9	7 0 D N 5
H	0 4 3 4	8 0 2 / 4	1 0 3 2 D	-m-1 0 0 R | D	0 9 9 9 9	9 9 9 I 9	I 9 I | open	XTP U 0 5 1 6	9 0 2 Y 5
U 0 5 1 1	9 0 2 Y 9	U	9 - +	1 2 5 6 4	GEN'L S-R 8 0 2 / 8	U 0 5 6 1	9 0 2 Y 5	U 0 5 8 1	9 0 2 Z 2
7 0 N F 9	U 1 9 2 0	9	1 3 0	U 1 9 2 4	9 0 2 6 9	U	9 1 R 2 0	Q 0 L 0̄ 6	N 0 L B 4
6 0 3 0̄ 6	6 0 V W 4	6 0 V Q 4	1 0 5 5 9	INDEXING 8 0 B 9 1	N 0 G 0 9	8 0 K Y 9	N 0 Ø V 9	7 0 Ø U 8	8 0 X 8
6 0 S 8 8	8 0 K Z 0	N 0 Ø Y 4	7 0 Ø X 3	8 0 X 8	6 0 S 9 5	8 0 K Z 1	N 0 P ‡ 9	7 0 Ø Z 8	8 0 X 8
6 0 S 9 9	7 0 P A 4	1 0 1	R_1 ‡	R_1 limit ‡	R_2 ‡	R_2 limit ‡	$R_1 + R_2$ ‡	0 0 0 0 ‡	R_3 ‡
R_3 limit ‡	$R_1 + R_3$ ‡	0 ‡ 0̄ 9 ‡	$R_2 + R_3$ ‡	0 0̄ 0̄ 6	$R_1+R_2+R_3$ ‡	SR1 U 0 7 3 9	9 0 7 2 9	U 0 7 5 9	9 0 7 4 9
8 0 7 W 8	U 0 7 1 9	1 0 8 7 9	SR2 U 0 7 3 9	9 0 7 1 9	8 0 X 4 8	8 0 7 V 8	U 0 7 2 9	1 0 8 7 4	SR3 U 0 7 5 9
9 0 7 1 9	8 0 X 2 8	8 0 7 T 8	U 0 7 4 9	7 0 X 6 8	7 0 7 X 8	9 0 7 4 4	9 0 2 8 9	4 0 L G 4	L 0 9 3 9
K 0 9 7 4	TN1, TX1 8 0 2 Z 3	6 0 7 / 8	6 0 7 V 8	6 0 7 S 3	8 0 P S 0	1 0 9 6 4	TN2, TX2 8 0 2 Z 3	6 0 7 S 8	6 0 7 W 8
6 0 7 T 3	8 0 P T 0	6 0 7 T 8	1 1 0 0 4	TN3 TX3 8 0 2 Z 3	6 0 7 U 8	6 0 7 V 8	6 0 7 W 8	6 0 7 V 3	8 0 P V 0
6 0 7 X 8	4 0 2 Q 9	L 0 4 3 4	K 0 4 6 4	4 0 Ø V 8	K 0 4 6 4	1 0 4 3 4	ENT I 1 0 4 4	7 0 3 T 4	8 0 7 X 3
6 0 3 T 4	B 0 0̄ A 8	B 0 0̄ 0̄ 5	B 0 0̄ 0 3	B 0 0̄ 0̄ 1	B 0 0̄ ‡ 1	B 0 0̄ 1 0	B 0 ‡ 0̄ 4	B 0 ‡ 0̄ 4	B 0 ‡ ‡ 4
B 0 ‡ 0 4	B 0 0 0̄ 2	H 0 2 Q 2	1 0 3 2 9	FLO U 1 1 5 6	9 0 2 Y 5	H 0 K W 0	8 0 K I 7	7 1 J E 4	P 0 K H 1
B 0 0	8	7 1 9 5 4	G 1 2 2 9	X 1 2 9 4	7 1 K C 4	M 1 J I 4	P 0 K Y 2	N 1 2 0 9	G 0 K I 9
1 1 2 1 4	8 0 K 0̄ 1	F 0 K E 4	8 0 2 Z 5	7 1 2 V 4	F 1 K T 0̄	0 0	7 0 2 5 2	T 1 R V 4	6 0 K V 2
U	9 0 K 4 5	Q 0 L 0̄ 6	N 0 L I 4	6 0 3 0̄ 6	6 1 / V 9	6 0 S R 5	1 1 1 3 4	P 0 K H 2	1 1 1 7 D

PRINT I 8 DIGIT MANTISSA

1300	AAABB	BDDDH	ARITHMETIC T1450	11369	T1540	61460	11364	T1460	11359	T(0010)
1350	61460	61540	61450	61440	11384	T1440	U1421	902Z2	U1721	902Z6
1400	U1426	902Y5	90L‡0	U192+0	9 -	9 -	1927	H1J+0H	A1809	G1RBI
1450	1154I	U192+0	11474	91R30	90K45	N1544	419K0	K1534	V1927	C0007
1500	X1519	C000A	11524	P0KH2	G1RB9	11544	H1927	81RBB	11669	N1784
1550	419L0	K1669	P1RC9	40L+03	K1779	71ØB9	71ØE4	M1ØA4	B0-0+00	F1927
1600	H1937	11624	U1920	91R30	B0009	D0	G1927	N1679	G0KC0	X1764
1650	C0	C0001	G1RC9	X1ØI4	M4-0E9	U0249	90269	11704	F0252	F0KE4
1700	Q0L+06	N1PB4	40SF8	L1744	U	90K45	N3KB4	61XB4	61US9	61UK4
1750	603-06	11419	D0001	P0KH2	11649	M1PI9	U0245	91R30	11704	G1RC9
1800	11669	N4039	419L0	K1679	F1927	C0003	F1954	H0483	W1954	F1954
1850	Q1954	V1927	C0003	G0272	V1937	E0009	V1954	X1894	E0004	X1914
1900	E0001	G0KH2	P1RB9	1166D	TEMPORARY STORAGE					
1950	TEMP. STORAGE		TRC, TRE, TSC 80NZ5	11979	80LU0	72-0X0	U2145	90L00	80S95	72‡44
2000	U2016	902Z6	U1936	9 -	H1945	G0352	71935	U1923	9 -	H1932
2050	G0352	71922	H1930	P1943	2139	82‡44	U2440	91R23	Q0L+06	N3KB4
2100	M2JA4	603-06	62‡M4	80TA8	N2‡C9	62‡A9	12014	U2430	91R23	12224
2150	41TM9	L2179	81TM9	72‡44	12039	60S88	72‡19	80S88	72S04	B0020
2200	8	70254	603-06	13224	91R36	80KG8	70L+06	10434	FXP, FPR U2286	902Y5
2250	H0B96	72B94	G0B98	43+012	K4079	72D04	U1920	9 -	U3	90KS7
2300	U2392	904X0	60SR6	H0KI3	G1RB9	M2LC9	62LI4	G0KI8	G0MG2	72LI9
2350	M2LG9	T0KZ0	60KZ0	623R5	Q02Q2	P0L00	M4-0H9	H1927	B0	0
2400	53	Q0L+06	N3KB4	603-06	42SY9	1225D	ARG1		ARG2	
2450	ARG2 / E	‡	ERROR ROUTINE U4241	92X51	14204	RPL H0288	M2499	72494	8	70288
2500	802Z2	725V4	725S9	816V9	625S9	T	M2544	12554	807W3	602Y8
2550	U	902Y5	802/1	10324	RTi, WTi 81KQ3	T0289	80L‡0	72ØV4	72PY9	U2701

PRINT I 8 DIGIT MANTISSA

(RTi, WTi) 902Z2	T2Q/5	T2PU5	M2629	62PU5	T2654	M2649	62Q/5	80ML8	72P0̄0
2020	30000	80S99	72W84	72W89	72X19	8	U	90PW4	B0003
	Q0L0†6	I2759	7	N2PD9	62X‡4	60SR9	6030̄6	12664	30001
13224	Ø2804	20902	Ø2779	14139	X2789	14149	2020	30004	12704
30000	N2QA9	14269	30001	10434	(TMT, TAB, TNA) 80KZ6	N0NY4	T1927	61RS7	10584
(ATR) 80KI0	40KI8	L2874	12904	B00†‡6	80BZ6	U0291	90BY5	70BZ0	90KF4
80KA0	60KI8	U2921	903T1	U	90BH3	10434	5 blanks	(RCD, WCD (NON-TAPE)) U2967	90S89
U2999	90KZ3	H1A‡8	20	3509	I2989	10464	Ø0434	20902	Ø300
14119	20300	38505	X2IX4	14129	(SAC) 16059	(SQR) 13594	(LGE, LGD) 14474	(EXE, EXD) 14874	(ART) 15469
(FSR) 1	(TMT) 12829	(WLi, WHi) 80KI0	602R2	U3081	903T1	U	40KI2	K3109	81J0†8
70KI2	90BH3	L0434	H0295	73159	73249	U3254	90KZ9	H1A‡8	H0B98
73A64	20	R3	I3234	U3387	90L‡0	U3265	90L‡0	M3B14	B0026
92IL5	X3204	M3224	R2458	802S6	10324	20902	Ø3249	14099	20
3000	X3AW4	14109							
LINE IMAGE									
							‡		
HEADING IMAGE									
		‡							
CARD IMAGE									
							‡	(SQR) H1583	V1929
F1930	C0001	F1929	Q1927	N0574	M4049	H1RT0	N3ØX4	M3ØV4	13659
619K9	81J/1	B0009	13679	D0001	F1939	71RT0	C0005	F1953	C0002
80LH0	U3723	910̄C7	63PB4	439	K3719	U3748	93PB3	U1940	93R
Q1943	V1953	C0002	G1946	F3898	Q1939	C0001	W3898	T3QR8	G3900
F1945	H1583	V1945	C0001	F3906	V3906	C0001	G1939	B0006	D0005
W1945	T3R‡6	P3910	E0001	X3884	10584	71927	61RS7	1058D	0

P R I N T I 8 D I G I T M A N T I S S A

3900	+0		0 0 0 +0	(SQR TABLE) 0 1 4.A 0 6 +0	0 2 3.B 0 7 H		0 7 2.B 1 1 +0	1 7 1.D 1 7 D		2 9 1.0 2 4
3950	G	5 9 0 H 3 0 D	9 9 0 F 4 0 I	+0 ‡ R	(RW, BS, TM) U 3 9 9 4	9 0 K Y 5	9 0 B Q 6	H 0 B Z 3	2 0 2 0	3
4000	P 0 B Y 2	N 0 D W 4	1 3 9 9 9	(SYSTEM) 2 0 2 0 0	3 0 0 0 2	Y 0 0 0 0	1 0 0 0 4	(ERROR ROUTINES) 8 0 2 A 0	1 2 4 6 4	8 0 0 A 9
4050	1 4 1 6 4	8 0 1 E 2	1 2 4 6 4	8 0 9 F 0	1 2 4 6 4	8 0 2 0 3	1 2 4 6 4	8 0 3 H 0	1 2 4 6 4	8 0 2 0 7
4100	1 2 4 6 4	8 0 1 E 1	1 2 4 6 4	8 1 2 0 1	1 4 1 7 9	8 1 2 0 2	1 4 1 7 9	8 0 1 0 2	1 2 4 6 4	8 1 2 0 4
4150	1 2 4 6 4	8 1 8 0 2	U 4 2 4 1	9 2 V 6 6	1 4 2 0 4	U 4 2 4 1	9 2 Z 8 1	1 4 2 0 4	U 4 2 4 1	9 6 W 3 6
4200	7 2 4 E 7	U 2 4 5 0	9 0 T 3 1	T 2 4 5 1	T 2 4 5 2	2 0 5 0 0	R 2 4 5 0	J 0 0 0 1	1	(SYSTEM) 8 0 4 0 2
4250	1 4 1 6 4	8 0 6 +0 2	1 4 1 9 4	8 1 1 B 3	1 2 4 6 4	2 0 1 0 0	Ø 4 3 1 4	J 0 1 0 0	A 4 3 3 4	2 0 2 0 1
4300	3 0 0 0 4	1 4 3 3 9	J 0 1 0 1	5 6 9 8 9	B 0 0 ‡ 2	B 0 0 0̄ 4	2 0 1 0 0	Y 4 3 9 0	I 4 2 7 9	8 4 4 ‡ 4
4350	N 4 4 ‡ 9	7 4 3 X 4	U 4 3 7 6	9 4 3 R 9	B 0 ‡ 0 0	U	9 4 U 0 5	1 4 3 3 9		
4400										
4450					(LGE,LGD) B 0 +0 ‡ 0	B 0 +0 / 0	Q 1 9 2 7	M 4 2 4 9	F 1 9 2 7	C 0 0 0 7
4500	V 4 8 4 5	P 4 8 5 8	7 4 5 3 4	7 4 5 3 9	8 3 I 6 9	6 4 E 3 4	G	H	V 1 9 2 7	E 0 0 0 1
4550	X 4 4 9 4	B 0 0 0 8	F 1 9 2 7	F 1 I T 9	Q 2 8 5 8	V 1 9 2 7	E 0 0 0 5	G 4 7 4 0	V 1 9 2 7	E 0 0 0 6
4600	G 4 7 4 7	V 1 9 2 7	E 0 0 0 7	G 4 7 5 5	V 1 9 2 7	E 0 0 0 7	P 1 9 3 9	F 1 9 3 9	H 1 9 2 9	D 0 0 0 9
4650	G 1 9 3 9	E 0 0 0 1	8 0 K Z 6	N 4 Ø Y 4	V 4 8 6 8	E 0 0 0 9	B 0 0 1 1	N 0 5 7 4	8 0 0̄ C 2	X 4 7 2 4
4700	E 0 0 0 3	F 1 9 2 7	F 1 R B 9	1 0 5 8 4	D 0 0 0 1	P 0 L H 8	1 4 6 9 9	1 4 3 2 P	2 1 7 0 9 8 J	4 3
4750	4 2 9 3 9 M	5 E	7 4 0 3 6 2 6 9 +0	3 F	5 5 6 3 0 2 5 0 A	2 G	4 3 1 3 6 3 7 6 D	2 B	3 4 2 4 2 2 6 8 A	
4800	1 H	2 5 5 2 7 2 5 0 E	1 E	1 7 6 0 9 1 2 5 I	1 C	1 1 3 9 4 3 3 5 B	1 B	0 7 9 1 8 1 2 4 F	1 A	0 4 1 3
4850	9 2 6 8 E	4 7 4 F	2 3 0 2 5 8 5 0 9 C		(EXE, EXD) H 1 9 2 7	D 0 0 0 2	8 0 K Z 6	N 4 R ‡ 9	V 5 4 3 2	X 5 2 6 9
4900	E 0 0 0 8	H 1 I S 9	P 5 C Z 9	M 4 A V 9	G 4 H U 5	M 4 I T 9	1 5 2 8 9	P 0 B W 2	7 4 I U 9	C 0 0
4950	B 0 0 1 0	F 1 9 2 9	B 0 0 0 8	T 1 R S 9	6 1 R S 1	6 1 R S 9	H 1 I S 1	4 3 +0 Z 5	K 4 1 5 9	M 5 0 2 9
5000	G 5 4 0 8	B 0 0 0 8	F 1 9 2 9	H 5 D V 9	6 1 I S 1	8 1 R S 2	7 5 0̄ U 4	7 5 J Z 4	Q 5 3 9	V 5 4 1 6
5050	G 1 9 2 9	B 0 0 0 8	V 5 4 2 4	C 0 0 0 7	B 0 0 0 8	F 1 9 2 9	C 0 0 0 7	V 4 8 4 5	G 5 4 6 3	7 5 1 9 9
5100	G 0 3 3 8	7 5 1 1 4	Q	G 1 9 2 9	F 1 9 2 9	V 5 3 8 0	C 0 0 0 7	G 5 4 3 8	V 1 9 2 9	C 0 0 0 6
5150	G 5 3 8 8	V 1 9 2 9	C 0 0 0 8	G 5 4 4 7	V 1 9 2 9	C 0 0 0 8	G 5 4 0 8	F 1 9 3 0	H 1 3 0	V

PRINT I 8 DIGIT MANTISSA

5200	(EXE, EXO) V1930	C0002	B0010	H5CZ6	X5254	G1IS1	F1IS9	E0002	F1927	10584	
5250	P5CZ3	D0001	15224	E0007	Q5CZ3	61IS9	14909	U1920	95M48	1058D	
5300	10E	0487901F	11F	1484200A	12H	2468600H	14B	3506568G	15G	451	
5350	0756B	17C	5481214A	19B	6523251I	416G	50000000		$\overset{+}{0}\overset{+}{0}\overset{+}{0}$AAABBBC		
5400	10000000$\overset{+}{0}$	30103000	2302585A	4342944H	16668H	09999999I	10				
5450	0000000A	0J	530B		(ART) H1927	N0584	M5489	Q1927	F6908	H19B9	40303
5500	K5894	402A0	K5864	M55E9	B0009	755C4	C00	B0008	H0EZ3	F1IU4	
5550	15724	755F4	D00	C0006	B0003	U5596	96Z46	80S22	65V99	4	
5600	K5594	85V99	G3Z39	75W29	B0014	85	71944	H1934	V6908	C0002	
5650	F1953	756F9	H5408	C00	G1953	F1953	Q1934	D0006	756I9	C00	
5700	G6908	B0009	D0008	W1953	F1953	V1953	C0008	F6908	H6020	V6908	
5750	C0005	G6027	V6908	C0006	G6037	V1953	C0008	G1944	N0584	H5DV7	
5800	X5849	C0002	F6908	F1IS9	T1IS7	M5HT9	Q6908	F1927	10584	P0BY2	
5850	D0001	15804	M5874	15524	B00/4	U1931	960Z4	15639	M5904	10584	
5900	H5DV7	H6045	15814	027 01C	012927500D	058 04A	038909723A	1			
5950	00 07G	065617871H	173 13$\overset{+}{0}$	091510070A	373 24A	117747926C					
6000	999 76$\overset{+}{0}$	143996893A	195G	333296M	099999992B	1570796C	5900				
6050	729		(SAC) 86PZ7	76ØT0	H1929	46805	K6719	P1306	M4259	G1300	76124
6100	H1927	M6119	Q1927	D0003	C00	B0011	V6877	E0008	B0009	F1938	
6150	H6886	P1938	M6224	G6895	M6199	H1938	P6835	B0009	16244	H6886	
6200	P1938	86PU4	76ØT0	16244	P6895	M6239	16199	H1938	V1306	B0010	
6250	F1939	U6B82	96$\overset{+}{0}$50	H6H98	C0008	66B84	46	K6279	L6279	U6C12	
6300	96B82	B0019	86	71959	H1939	M6349	Q1IS7	F1IS7	Q1939	P1951	
6350	E0001	B0008	F1939	V1939	E0008	F1951	H1309	V1951	E0005	G6859	
6400	V1951	E0005	G6868	V1939	E0008	F1939	H6847	V1951	E0006	G6854	
6450	V1951	E0006	G5408	F1951	H1954	V1951	E0002	F6908	H1957	V1939	

PRINT I 8 DIGIT MANTISSA

6500	SAC E0002	G6908	N6689	H2BV2	X6659	F1IS9	C0002	H1IS7	M6EV9	F6908					
6550	Q6908	F1927	H1954	V1939	E0002	F6908	H1957	V1951	E0002	P6908					
6600	N6704	H2BV2	X6674	C0002	F1937	F1IT9	16644	16914	Q1937	F1937					
6650	16639	D0001	P5CZ3	16524	D0001	P5CZ3	16614	U1920	90K65	16564					
6700	U1930	90K65	16639	M4259	B0000	16129		000000000$\overset{+}{0}$	00$\overset{+}{0}$	10$\overset{+}{0}$	01				
6750	018066894A	02H	09F	03		040966553$\overset{+}{0}$	06$\overset{+}{0}$	08$\overset{+}{0}$	05		059033447A	08			
6800	$\overset{+}{0}$	06$\overset{+}{0}$	07		081933105I	09F	02H	09		100000000$\overset{+}{0}$	10$\overset{+}{0}$	00$\overset{+}{0}$	9I	253B	12
6850	3369Q	6459P	15707963G	15915494C	500000000$\overset{+}{0}$	25000000$\overset{+}{0}$	01I								
6900	WORK AREA	SAC 802S2	U6926	902Z6	U	91R30	66ZB9	10584							

ENTER ROUTINE

```
Y-10 SET 0004 01                    THESE 3 COMMANDS                  6
Y-05 LOD Y-05 01                       COMPILED BY THE                6
Y    TR  1039                          PRE-EDIT ROUTINE               2

1039 TRA 1044                                                         2
1044 UNL 0334 01                    BADD-6  ZONED FOR ASU 01          6
1049 LOD 0773 01          0006                                        6
1054 ADM 0334 01                    BADD  ZONED FOR ASU 15            6
1059 SET 0018 15                                                     20
1064 SET 0005 14                                                      7
1069 SET 0003 12                                                      5
1074 SET 0001 10                                                      3
1079 SET 0001 09                                                      3
1084 SET 001  08 —— 0/2/4           DIFFERENT FOR 20 DIGITS    12,14,16
1089 SET 0004 07                                                      6
1094 SET 0004 06                                                      6
1099 SET 0004 05                                                      6
1104 SET 0004 04                                                      6
1109 SET 0002 03                                                      4
1114 RAD 0282 02             1                                        3
1119 TR  0329                       TO FIRST COMMAND                  2
```

FETCH AND SUB-ROUTINE SWITCH

```
 324 ADM 0334 01                    NEW BADD EQUAL OLD PLUS LENGTH    6
 329 RCV 0283                       MOVE VARIABLE LENGTH              2
 334 TMT BADD 15                       COMMAND TO STD. POSITION      20
 339 RSU 0284 11            YY      40 CODES   04 BY 5 TO 99          4
 344 TRP 0624 11                    ADDR. IS OPTIONALLY 0634          2
 349 UNL 0354 11            YY      ACTUAL PLUS OP, NON-INDEXABLE     4
 354 TR  00YY                       YY VARIES 04 TO 99                2
```

INDEXING

```
 624 LOD 0291 12           XXX      TEST FOR ANY INDEXING             5
 629 TRZ 0709 12                                                      2

 634 LOD 0289 09             X      ALPHA INDICATOR    0 TO 7         3
 639 TRZ 0659 09                                                      2
 644 UNL 0648 09                                                      3
 649 LOD 07□8 04                    CONTENTS OF INDEX REGISTER        6
 654 ADM 0288 04                    INDEX FUNCTION OF ALPHA           6

 659 LOD 0290 09             X      BETA INDICATOR                    3
 664 TRZ 0684 09                                                      2
 669 UNL 0673 09                                                      3
 674 LOD 07□8 04                    CONTENTS OF INDEX REGISTER        6
 679 ADM 0295 04                    INDEX FUNCTION OF BETA            6

 684 LOD 0291 09             X      GAMMA INDICATOR                   3
 689 TRZ 0709 09                                                      2
 694 UNL 0698 09                                                      3
 699 LOD 07□8 04                    CONTENTS OF INDEX REGISTER        6
 704 ADM 0299 04                    INDEX FUNCTION OF GAMMA           6

 709 UNL 0714 11                                                      4
 714 TR  01YY                       YY VARIES 04 TO 99                2
```

CONDITIONAL TRANSFER COMMANDS TRZ,TNZ,TRP,TRM

```
404 SGN 0291 00                                   STRIP INDICATOR SIGN              4
409 LOD 0293 14                          XXXXX    BETA ADDRESS AND OP CODE          7
414 TRP 0449 00                                   TRANSFER TO TRP/TRZ TEST          2

419 UNL 0425 14                                   INSERT OP CODE AND BETA-2         7
424 RAD      00      BETA-2                       MANTISSA OF BETA             10,12,14
429 [M/N] 0464 00                                 TO NEXT COMMAND IF PLUS OR 0      2
```

UNCONDITIONAL TRANSFER COMMAND   TRU

```
434 RCV 0331                                      FIRST STEP OF  TRU.  REPLACE      2
439 TMT 0285 01                                      BADD BY ALPHA                  6
444 TR  0329                                      NEXT COMMAND, BYPASS LENGTH      -4

449 UNL 0455 14                                   INSERT OP CODE AND BETA-2         7
454 RAD      00      BETA-2                       MANTISSA OF BETA             10,12,14
459 [M/N] 0434 00                                 TO ALPHA IF PLUS OR 0             2

464 LOD 0214 01                           0011    LENGTH FOR TRZ,TNZ,TRP,TRM        6
469 TR  0324                                      TO NEXT COMMAND IN SEQUENCE       2
```

EXTRACT POWER COMMAND   XTP

```
494 RCV 0516                                      SET UP TMT FOR                    2
449 TMT 0285 01                                      POWER OF ALPHA                 6
504 RCV 0511                                      SET UP RCV FOR                    2
509 TMT 0289 01                                      POWER OF ALPHA                 6
514 RCV            BETA-1                         DUPLICATE POWER OF                2
519 TMT      11    ALPHA-1                           ALPHA IN BETA                  4
524 TR  2564                                      TO NEXT COMMAND VIA RPL           2
```

GENERAL SUB-ROUTINE HOUSEKEEPING

```
529 LOD 0218 01                          0014     LENGTH FOR SUB-ROUT. COMMANDS     6
534 RCV 0561                                      SET UP TMT TO FETCH               2
539 TMT 0285 01                                      CONTENTS OF ALPHA              6
544 RCV 0581                                      SET UP RCV TO STORE               2
549 TMT 0292 01                                      RESULT IN BETA                 6
554 UNL 0569 11                       29 - 64     SET SECONDARY OP SWITCH           4

559 RCV 1920                                      MOVE CONTENTS OF ALPHA TO         2
564 TMT      08      ALPHA-9/11/13                   FIXED WORKING POSITION    12,14,16
569 TR  30YY                                      TO SPECIFIC ROUTINE               4

574 RCV 1924                                      INSERT ZERO                       2
579 TMT 0269 00                                      FOR RESULT                     3
584 RCV              BETA-9/11/13                 STORE RESULT                      2
589 TMT 1920 08                                      IN BETA                   12,14,16

594 RSU 0306 11                            XX     REPEAT TALLY                      4
599 TRZ 0324 11                                   TO NEXT COMMAND IN SEQUENCE       2

604 ADM 0306 02                                   DIMINISH RPT TALLY BY 1           4
609 ADM 0564 05                                   AUGMENT ALPHA ADDRESS BY I        6
614 ADM 0584 06                                   AUGMENT BETA ADDRESS BY J         6
619 TR  0559                                      TO REPEAT THIS COMMAND            2
```

SET INDEX REGISTER COMMANDS SRI

```
 784 RCV 0739                                CONTENTS OF R2 TO              2
 789 TMT 0729 00                                R1 PLUS R2                  3
 794 RCV 0759                                CONTENTS OF R3 TO              2
 799 TMT 0749 00        SR1                     R1 PLUS R3                  3
 804 LOD 0768 01                             R2 PLUS R3                     6
 809 RCV 0719                                MAC II TO R1                   2
 814 TR  0879                                                               2

 819 RCV 0739                                CONTENTS OF R1 TO              2
 824 TMT 0719 00                                R1 PLUS R2                  3
 829 LOD 0748 04        SR2                  CONTENTS OF R3                 6
 834 LOD 0758 01                             CONTENTS OF R1 PLUS R3         6
 839 RCV 0729                                MAC II TO R2                   2
 844 TR  0874                                                               2

 849 RCV 0759                                CONTENTS OF R1 TO              2
 854 TMT 0719 00                                R1 PLUS R3                  3
 859 LOD 0728 04        SR3                  CONTENTS OF R2                 6
 864 LOD 0738 01                             CONTENTS OF R1 PLUS R2         6
 869 RCV 0749                                MAC II TO R3                   2

 874 UNL 0768 04                             TO R2 PLUS R3                  6
 879 UNL 0778 01                             TO R1 PLUS R2 PLUS R3          6
 884 TMT 0744 00                             SET REGISTER TO 0              3
 889 TMT 0289 00                             PLACE LIMIT IN REGISTER        3
 894 CMP 0374 11                             COMPARE OP CODE TO 89          4
 899 TRE 0939                                TO TN2 ON OP CODE 89           2
 904 TRH 0974                                TO TN3 ON OP CODE 94           2
```

TRANSFER ON INDEX COMMANDS TXI, TNI

```
 909 LOD 0293 01                  XXXX       R1 INCREMENT                   6
 914 ADM 0718 01        TN1                  AUGMENT R1                     6
 919 ADM 0758 01        TX1                  AUGMENT R1 PLUS R3             6
 924 ADM 0723 01                             AUGMENT R1 LIMIT TALLY         6
 929 LOD 0720 09                     X       1000 POS. OF LIMIT TALLY       3
 934 TR  0964                                                               2

 939 LOD 0293 01                  XXXX       R2 INCREMENT                   6
 944 ADM 0728 01        TN2                  AUGMENT R2                     6
 949 ADM 0768 01        TX2                  AUGMENT R2 PLUS R3             6
 954 ADM 0733 01                             AUGMENT R2 LIMIT TALLY         6
 959 LOD 0730 09                     X       1000 POS. OF LIMIT TALLY       3
 964 ADM 0738 01                             AUGMENT R1 PLUS R2             6
 969 TR  1004                                                               2

 974 LOD 0293 01                  XXXX       R3 INCREMENT                   6
 979 ADM 0748 01        TN3                  AUGMENT R3                     6
 984 ADM 0758 01        TX3                  AUGMENT R1 PLUS R3             6
 989 ADM 0768 01                             AUGMENT R2 PLUS R3             6
 994 ADM 0753 01                             AUGMENT R3 LIMIT TALLY         6
 999 LOD 0750 09                     X       1000 POS. OF LIMIT TALLY       3

1004 ADM 0778 01                             AUGMENT R1 PLUS R2 PLUS R3     6
1009 CMP 0289 02                             1 FOR TNI, 2 FOR TXI           3
1014 TRE 0434                                TNI EQUIVALENT TO TRU          2
1019 TRH 0464                                TO NEXT COMMAND ON SRI         2
1024 CMP 0658 09                             1000 POS. WITH Z IN MEMORY     3
1029 TRH 0464                                HI, 0-9 IN 1000S, TO NEXT COMM 2
1034 TR  0434                                TRU TO ALPHA                   2
```

FLOAT COMMAND   FLO

```
1124 RCV 1156                                    INSERT ALPHA                        2
1129 TMT 0285 01                                    ADDRESS                          6
1134 RAD 0260 09                            3    NUMERICAL PART OF SHR OP CODE       3
1139 LOD 0297 11                           XX    N                                   4
1144 UNL 1154 11                                 INSERT N IN SET INSTRUCTION         4
1149 SUB 0281 11                                 N-M                                 4
1154 SET 00NN 00                                 SET 00 FOR LOADING A
1159 LOD      00  ALPHA                          LOAD A
1164 UNL 1954 00                                 UNLOAD A TO TEMPORARY
1169 ADD 1229 00                                 UNSIGN A BY ADDING -0
1174 NTR 1294 00
1179 UNL 1234 11                                 N-M-S IN SHIFT ADDRESS              4
1184 TRP 1194 11                                 TR IF N > (M PLUS S)                2
1189 SUB 0282 09                                 CONVERT TO NUMER. PART OF LNG       3
1194 TRZ 1209 00                                 TRANSFER IF A EQUAL 0               2
1199 ADD 0299 11                                 N-M-S    PLUS    P-N PLUS M         4
1204 TR  1214                                    EQUALS  P-S  EQUALS POWER OF A      2
1209 LOD 0201 11                           00    SET AP TO ZERO                      4
1214 ST  0254 11                                 STORE AP IN PAC1                    4
1219 LOD 0295 01                         XXXX    ASU SIGN IS PLUS                    6
1224 UNL 1254 01                                 INSERT BETA-9/11/13 AS ADDRESS      6
1229 ST  1230 09                       3 OR 4    C OR D STORED IN SHIFT INSTR.       3
1234 [C/D] 00 00  N-M-S                          SHIFT A TO MANTISSA LENGTH
1239 UNL 0252 00                                 UNLOAD UNSIGNED A INTO PAC1 10,12,14
1244 SGN 1954 09                                 STRIP SIGN FROM MANTISSA OF A       4
1249 ADM 0252 09                                 PLACE SIGN OVER MANTISSA OF A       3
1254 RCV          BETA-9/11/13                   SEND PAC1 TO                        2
1259 TMT 0245 08                                    RESULT POSITION            12,14,16
1264 RSU 0306 11                                 REPEAT TALLY                        4
1269 TRZ 0394 11                                 TO LOAD LENGTH AND TO NEXT          2
1274 ADM 0306 02                                 REDUCE REPEAT TALLY BY 1            4
1279 ADM 1159 05                                 AUGMENT ALPHA ADDRESS               6
1284 ADM 0295 06                                 AUGMENT BETA ADDRESS                6
1289 TR  1134                                    TO REPEAT FLO COMMAND               2
1294 SUB 0282 11                           XX    N-M - (S-1) - 1                     4
1299 TR  1174                                    RETURN TO NORMALIZE                 2
```

REPLACE COMMAND   RPL

```
2479 RAD 0288 00              XX OR 2044-        PLUS ALPHA ADDRESS INDICATES        6
2484 TRP 2499 00                                    FIRST ORDER TYPE                 2
2489 UNL 2494 00                                 REPLACE 285-288 BY CONTENTS         6
2494 LOD 20   00  19/44                             OF LAR1, LAR2 OR UNZONED         6
2499 UNL 0288 00                                    UNITS OF ALPHA-9/11/13           6
2504 LOD 0292 01                         XXXX    BETA PLUS 2                         6
2509 UNL 2554 01                                 INSERT AS RCV ADDRESS               6
2514 UNL 2529 01                                 INSERT AS SGN ADDRESS               6
2519 LOD 1659 01                         0001                                        6
2524 ADM 2529 01                                 CONVERT SGN TO BETA PLUS 3          3
2529 SGN      00  BETA PLUS 3                    TEST IF ALPHA OF COMMAND IN         4
2534 TRP 2544 00                                    BETA IS ZONED FOR ASU 15         2
2539 TR  2554                                    TR IF BETA CONTAINS ALPHA-9/11/13   2
2544 LOD 0763 01              0009/0011/0013     CONSTANT                            6
2549 ADM 0288 01                                 CONVERT ALPHA TO ASU 15 ZONING      6
2554 RCV          BETA PLUS 2                    REPLACE ALPHA ADDRESS OF BETA       2
2559 TMT 0285 01                                    BY ALPHA OR ALPHA-9/11/13        6
2564 LOD 0211 01                         0010    LENGTH FOR RPL COMMAND   /XTP/      6
2569 TR  0324                                    TO NEXT COMMAND IN SEQUENCE         2
```

TRANSMIT COMMANDS TMT,TAB,TNA

```
2829 LOD 0296 09          PLUS, - OR 0   TMT IS 0, TAB IS PLUS, TNA IS -  3
2834 TRZ 0584 09                         TO TMT ONLY ON 0                  2
2839 SGN 19   00    27/29/31             REMOVE MANTISSA SIGN              4
2844 ADM 19   09    27/29/31             APPLY DESIRED SIGN TO MANTISSA    3
2849 TR  0584                            RETURN TO GENERAL ROUTINE         2
```

SWITCHING COMMAND ATR

```
2854 LOD 0290 11                    XX   ALPHA LIMIT                       4
2859 CMP 0298 11                         COMPARE LIMIT TO TALLY            4
2864 TRE 2874                            TO SWAP ADDRESSES IF EQUAL        2
2869 TR  2904                            TO AUGMENT TALLY IF NOT EQUAL     2
2874 SET 0006 13                         LOAD BETA AND ITS                 8
2879 LOD 0296 13                            LIMIT INTO ASU 13              8
2884 RCV 0291                            MOVE ALPHA AND ITS LIMIT          2
2889 TMT 0285 13                            TO BETA POSITION               8
2894 UNL 0290 13                         SWAP BETA AND ITS LIMIT           8
2899 TMT 0264 11                    00   SET TALLY TO ZERO                 4
2904 LOD 0210 11                    01   AUGMENT TALLY BY 1                4
2909 ADM 0298 11                                                           4
2914 RCV 2921                            INSERT LOCATION OF THIS           2
2919 TMT 0331 01                            COMMAND IN RCV                 6
2924 RCV BADD                            SEND MODIFIED COMMAND TO          2
2929 TMT 0283 15                            ORIGINAL LOCATION             20
2934 TR  0434                            TO TRU EITHER ALPHA OR BETA       2
```

CARD COMMANDS RCD, WCD

```
2944 RCV 2967                            INSERT UNIT DESIGNATION           2
2949 TMT 0289 04                            AND READ/WRITE OP CODE         6
2954 RCV 2999                            INSERT UNIT DESIGNATION           2
2959 TMT 0293 09                            FOR ERROR ROUTINE              3
2964 RAD 1108 13                   000   MONITOR FOR ALLOWABLE ERRORS      5
2969 SEL 0[ ]                                                              2
2974 [Y/R] 3509                          READER OR PUNCH
2979 TRA 2989                                                              2
2984 TR  0464                            TO LENGTH AND NEXT COMMAND        2
2989 TRS 0434                            TRU ON END OF FILE                2
2994 SEL 0902                            TEST FOR                          2
2999 TRS 300[ ]                             ERROR TYPE                     2
3004 TR  4119        E11                 READ OR PUNCH ERROR               2
3009 SEL 0300                            MEMORY TO PUNCH BUFFER            2
3014 SUP 0005                             ERROR,TRY TO CORRECT             2
3019 NTR 2974 13                           3 TIMES
3024 TR  4129        E12                 JUST CANT GET IT RIGHT            2
```

```
1314 SGN 1450 00       ADD/SUB              SET SW 1 TO TR                   4
1319 TR  1369                                                                2
1324 SGN 1540 00       MPY/MMY              SET SW 3 TO TR                   4
1329 ADM 1460 00                            SET SW 2 TO NOP                  3
1334 TR  1364                                                                2
1339 SGN 1460 00       PMA/MPM              SET SW 2 TO TR                   4
1344 TR  1359                                                                2
1349 SGN 00   00       MAD/MMA              SET SW 2 TO NOP                  4
1354 ADM 1460 00       10/12/14             WORD LENGTH                      3
1359 ADM 1540 00                            SET SW 3 TO NOP                  3
1364 ADM 1450 00                            SET SW 1 TO NOP                  3
1369 ADM 1440 00                            SET SW 4 TO NOP                  3
1374 TR  1384                                                                2
1379 SGN 1440 00       DIV/MDV              SET SW 4 TO TR                   4
1384 RCV 1421                               INSERT BETA                      2
1389 TMT 0292 01                               ADDRESS                       6
1394 RCV 1721                               INSERT GAMMA                     2
1399 TMT 0296 01                               ADDRESS                       6
1404 RCV 1426                               INSERT ALPHA                     2
1409 TMT 0285 01                               ADDRESS                       6
1414 TMT 0300 09                            INSERT PLUS OR - CODE TYPE       3

1419 RCV 1920                               SEND TO TEMPORARY STORAGE THE    2
1424 TMT      08       BETA-9/11/13            WORD IN BETA AND THE   12,14,16
1429 TMT      08       ALPHA-9/11/13           WORD IN ALPHA          12,14,16
1434 H/Q 19   00       27/29/31             BETA MANTISSA, PROPER SIGN 10,12,14
1439 RAD 1108 11                   000      SET ASU 11 TO OVERFLOW LENGTH    5
1444 A/1 1809                               SWITCH 4, TR TO DIVIDE           2
1449 ADD 19   11       29/31/33    0XX ±    POWER OF BETA                    5
1454 A/1 1549                               SWITCH 1, TR TO ADD/SUB X=±B     2
1459 RCV 1920                               SET MAC II TO RECEIVE Z          2
1464 A/1 1474                               SW. 2, TR TO SEND PAC1 TO Z      2
1469 TMT 193  08       0/2/4                SEND A EQUAL Z TO 1920 ON 12/14/16
1474 TMT 0245 08                            SEND PAC1 TO Z OR Y AREA  12/14/16
1479 TRZ 1544 00                            TO SWITCH 3 IF B EQUAL 0         2
1484 CMP 1920 02                            IF Z EQUAL 0, TR                 3
1489 TRH 1534                                  TO SET X EQUAL TO 0           2
1494 MPY 19   00       27/29/31           ± B.Z EQUAL X, Z IS A OR PAC1 98,142,194
1499 SHR 00   00       07/09/11             X MANTISSA AT M PLUS 1    10,12,14
1504 NTR 1519 00                            NORMALIZE OR SHORTEN
1509 SHR 0001 00                               MANTISSA OF X                 4
1514 TR  1524                                  TO M DIGITS                   2
1519 SUB 0282 11                            X POWER - 1 IF NORMALIZED        4
1524 ADD 19   11       29/31/33             ADD Z POWER TO GET X POWER       5
1529 TR  1544                                                                2
1534 RAD 19   00       27/29/31             SET MANTISSA AND POWER OF X 10/12/14
1539 LOD 1922 11                               TO 0 IF Z EQUALS 0            5
1544 A/1 1669                               SW 3,  TR IF C EQUALS X IN 00    2
1549 TRZ 1784 00                            TR TO SET C EQUAL Y IF X IS 0    2
1554 CMP 193  02       0/2/4                TR TO SET C EQUAL X              3
1559 TRH 1669                                  IF Y IS 0                     2
1564 SUB 19   11       39/43/47             X POWER - Y POWER                4
1569 CMP 0303 11                            TR IF ABS. DIFF. OF POWERS       4
1574 TRH 1779                                  EXCEEDS MANTISSA LENGTH       2
1579 UNL 1629 11                            INSERT ADDRESSES OF              5
1584 UNL 1654 11                               SHIFT INSTRUCTIONS            5
1589 TRP 1614 11                            TR IF ABS. X > ABS. Y            2
1594 SET 0000 11                               AND SET ON NEGATION           2
1599 ST  19   00       27/29/31             STORE X MANTISSA AS SMALLER 10,12,14
1604 RAD 19   00       37/41/45             Y MANTISSA AS LARGER IN 00  10,12,14
1609 TR  1624                                                                2
1614 RCV 1920                               Y MANTISSA AS SMALLER, X IS      2
1619 TMT 193  08       0/2/4                   ALREADY IN 00 AS LARGER  10,12,14
```

```
1624 SET 00   00    09/11/13         LEADING 0 BEFORE LARGER       11,13,15
1629 LNG 0    00    DIFF. POWER      EXTEND TO ADD SMALLER
1634 ADD 19   00    27/29/31         ADD SMALLER MANTISSA
1639 TRZ 1679 00                     TR TO SET PAC1 TO 0 IF C IS 0        2
1644 ADD 0230 11                     001 OR DIFF. POWER PLUS 1            5
1649 NTR 1764 00                     NORMALIZE
1654 SHR 0    00    DIFF. POWER      SHORTEN C MANTISSA TO M PLUS 1
1659 SHR 0001 00                     SHORTEN C MANTISSA TO M              4
1664 ADD 19   11    39/43/47         PLUS Y POWER EQUAL C POWER           5
1669 NTR 1694 11                     NORMALIZE LEGAL POWER AND TR         6
1674 TRP 4059 11    E05              TEST ILLEGAL POWER FOR OFLO          2
1679 RCV 0249                        SET PAC1 TO 0 ON 0 RESULT            2
1684 TMT 0269 00                        OR POWER UNDERFLOW                4
1689 TR   1704                       TO INTERROGATE RPT TALLY             2
1694 ST   025 00    2/4/6            STORE C MANTISSA IN PAC1      10,12,14
1699 ST   025 11    4/6/8            STORE C POWER IN PAC1                4
1704 RSU 0306 11                     REPEAT TALLY                         4
1709 TRZ 1724 11                     TR TO SEND PAC1 TO GAMMA IF 0        2
1714 CMP 0268 07                     IF K IS 0 AND RPT TALLY ISNT,        6
1719 TRE 1744                        DO NOT SEND PAC1 TO GAMMA            2
1724 RCV            GAMMA-9/11/13    SEND C TO GAMMA IF RPT TALLY         2
1729 TMT 0245 08                        IS 0 OR K IS NOT 0         12,14,16
1734 TRZ 3224 11                     TO LENGTH AND NEXT COMMAND           2
1739 ADM 1724 07                     AUGMENT GAMMA ADDRESS BY K           6
1744 ADM 1429 05                     AUGMENT ALPHA ADDRESS BY I           6
1749 ADM 1424 06                     AUGMENT BETA ADDRESS BY J            6
1754 ADM 0306 02                     DIMINISH REPEAT TALLY                4
1759 TR   1419                       TO REPEAT THIS COMMAND               2

1764 LNG 0001 00                     RESTORE TO DETERMINED LENGTH         4
1769 SUB 0282 11                     DECREASE C POWER BY 1                4
1774 TR   1649                       FOR RE-NORMALIZATION                 2

1779 TRP 1799 11                     TR IF ABS. X ≥ ABS. Y                2
1784 RCV 0245                        SEND C EQUAL Y TO PAC1 IF            2
1789 TMT 193  08    0/2/4               ABS. Y > ABS. X            12,14,16
1794 TR   1704                       TR TO INTERROGATE RPT TALLY          2

1799 ADD 19   11    39/43/47         RESTORE X POWER TO ASU 11            5
1804 TR   1669                       TR TO NORMALIZE LEGAL POWER          2

1809 TRZ 4039 00    E01              DIVISION BY 0 ERROR                  2
1814 CMP 193  02    0/2/4            IF NUMERATOR IS 0, TR TO             3
1819 TRH 1679                           SET PAC1 TO 0                     2
1824 ST   19  00    27/29/31         STORE B MANTISSA IN TEMP      10,12,14
1829 SHR 000  00    3/4/5            SHORTEN B TO B1                  6,7,8
1834 ST   1954 00                    STORE B1                         7,8,9
1839 RAD 048  00    3/5/7            0.(M PLUS 1) NINES            12,14,16
1844 DIV 1954 00                     N1 EQUAL RECIPROCAL OF B1        ,383,
1849 ST   1954 00                    STORE N1                         7,8,9
1854 RSU 1954 00                     - N1                             7,8,9
1859 MPY 19   00    27/29/31         - N1B                             ,86,
1864 SHR 000  00    3/4/5            TO M PLUS 2                      6,7,8
1869 ADD 027  00    2/4/6            2 - N1B                       12,14,16
1874 MPY 19   00    37/41/45         A TIMES 2 - N1B                  ,170,
1879 RND 00   00    09/11/13         TO M PLUS 1                   14,16,18
1884 MPY 1954                        N1A TIMES 2 - N1B                ,112,
1889 NTR 1894 00                                                       ,20,
1894 RND 000  00    4/5/6            MANTISSA AT 1.5 M              9/10/11
1899 NTR 1914 00                     POSSIBLE SECOND
1904 RND 0001 00                     ROUNDING AND                         6
1909 ADD 0282 11                     ADJUST C POWER                       5
1914 SUB 19   11    29/31/33         - B POWER  OR - B POWER PLUS 1       5
1919 TR   1664                       TO COMPLETE AND TEST C POWER         2
```

COMPARISON AND SEARCH COMMANDS   TRC,TRE,TSC

```
1964 LOD 0595 09                     N    OP CODE FOR TRE                    3
1969 TR  1979                                                               2
1974 LOD 0340 09                     M    OP CODE FOR TRC OR TSC             3
1979 UNL 2070 09                          INSERT OP CODE FOR TRZ/TRP         3
1984 RCV 2145                             SET SWITCH FOR TABLE               2
1989 TMT 0300 10                             SEARCH OR TRANSFER              3
1994 LOD 0295 04                          INSERT BETA-9                      6
1999 UNL 2044 04                             ADDRESS                         6
2004 RCV 2016                             INSERT GAMMA-9                     2
2009 TMT 0296 01                             ADDRESS                         6

2014 RCV 1936                             SEND GAMMA, I.E. TESTNUMBER,       2
2019 TMT      08    GAMMA-9/11/13            TO TEMPORARY STORAGE     12,14,16
2024 RAD 1945 00                          POWER OF GAMMA                     4
2029 ADD 0352 00                          POWER PLUS 100                     5
2034 UNL 1935 00                          POWER PLUS 100 IN FRONT            5
2039 RCV 1923                             SEND BETA                          2
2044 TMT      08    BETA-9/11/13             TO TEMPORARY STORAGE     12,14,16
2049 RAD 1932 00                          POWER OF BETA                      4
2054 ADD 0352 00                          POWER PLUS 100                     5
2059 UNL 1922 00                          POWER PLUS 100 IN FRONT            5

2064 RAD 1930 00                          PSEUDO-BETA                 13,15,17
2069 SUB 1943 00                          PSEUDO-BETA - PSEUDO-GAMMA  13,15,17
2074 [N/M] 2139 00                        TEST                               2
2079 LOD 2044 04                          LAR2 ADDRESS                       6
2084 RCV 244        0/2/4 ARG2            N-1 TH BETA                        2
2089 TMT 1923 08                                TO ARG2               12,14,16
2094 RSU 0306 11                          REPEAT TALLY                       4
2099 TRZ 3224 11                          LENGTH AND TO NEXT COMMAND         2

2104 TRP 2114 11                          TEST FOR INDEFINITE REPEAT         2
2109 ADM 0306 02                          DIMINISH REPEAT TALLY IF PLUS      4
2114 ADM 2044 06                          AUGMENT TO NTH BETA                6
2119 LOD 0318 07                          LOAD K INDEXING REGISTER           6
2124 TRZ 2039 07                                                             2
2129 ADM 2019 07                          AUGMENT TO NTH GAMMA               6
2134 TR  2014                                                               2

2139 RCV 2430       ARG1                  MOVE NTH BETA                      2
2144 TMT 1923 08                             TO ARG1                  12,14,16
2149 [A/1] 2224                           TR OUT IF NOT TABLE SEARCH         2
2154 CMP 1349 06                          TR IF TABLE SEARCH IS IN           6
2159 TRE 2179                                ONE WORD LENGTH JUMP STATUS     2
2164 LOD 1349 06                          REPLACE INCR BY 1 WORD LENGTH      6
2169 UNL 2044 04                          N-1 TH BETA ADDRESS TO LAR1        6
2174 TR  2039                             RETURN FOR REFINED SEARCH          2

2179 ADM 0288 04                                                             6
2184 UNL 2019 04                        (N-1TH BETA - 9) IN LAR2             6
2189 LOD 0288 04                          INSERT LOCATION OF FUNCTION        6
2194 UNL 2204 04                             IN LOD ADDRESS                  6
2199 SET 002  00    0/4/8                 TO DOUBLE WORD LENGTH             22
2204 LOD      00                          LOAD FUNCTIONS OF X AND X-1       22
2209 UNL 0254 00                          FUNCTIONS TO PAC1 AND PAC2        22
2214 ADM 0306 02                          DIMINISH REPEAT TALLY BY 1         4
2219 TR  3224                             TO LENGTH AND NEXT COMMAND         2

2224 TMT 1936 08                          CN IN ARG2                  12,14,16
2229 LOD 0278 11                          RESET REPEAT TALLY TO 0            4
2234 UNL 0306 11                             IF TR BEFORE EXHAUSTED          4
2239 TR  0434                             TO TRU TO ALPHA                    2
```

REPEAT COMMANDS   RPT, RWR

```
 359 RCV 0249                              SET PAC1                        2
 364 TMT 0269 00                              TO ZERO                      4

 369 RCV 0309                              SEND REPEAT INFORMATION         2
 374 TMT 0289 00                              TO STANDARD POSITION         4
 379 LOD 0310 05                           ALPHA INDEX INCREMENT           6
 384 LOD 0314 06                           BETA INDEX INCREMENT            6
 389 LOD 0318 07                           GAMMA INDEX INCREMENT           6
 394 LOD 0222 01                    0017   LENGTH FOR RPT AND RWR          6
 399 TR  0324                              TO NEXT COMMAND, WHICH REPEATS  2
```

CONVERSION COMMANDS   FXP, FPR

```
2244 RCV 2286                              SET UP TMT TO FETCH             2
2249 TMT 0285 01                              CONTENTS OF ALPHA            6
2254 RAD 0296 12                     XXX   INSERT TYPE-                    2
2259 UNL 2294 12                              WHEEL ADDRESS                5
2264 ADD 0298 12                           TW PLUS D PLUS 1                5
2269 CMP 3012 12                           TEST FOR LINE OVERFLOW BY       5
2274 TRH 4079        E07                      COMPARING TO 385             2
2279 UNL 2404 12                           INSERT SPR ADDRESS              5
2284 RCV 1920                              MOVE FLOATING POINT NUMBER      2
2289 TMT      08      ALPHA-9/11/13           TO WORKING POSITION     12,14,16
2294 RCV 3 [TW]                            INSERT DECIMAL POINT            2
2299 TMT 0227 09                              IN LINE IMAGE                3
2304 RCV 2392                              INSERT ADDRESS FOR SET          2
2309 TMT 0470 01                              AND SHIFT CODE               6
2314 ADM 0296 06                           AUGMENT TW COMMAND ADDRESS      6
2319 RAD 0293 11                     OXX   SCALE FACTOR                    5
2324 ADD 19   11      29/31/33             SCALED POWER                    5
2329 TRP 2339 11                           TEST FOR P ≥ 0                  2
2334 ADM 2394 11                           M PLUS 1 PLUS ABS. P IN SET     5
2339 ADD 0298 11                           P PLUS D PLUS 1                 5
2344 ADD 0472 11                           P PLUS D - M                    5
2349 UNL 2399 11                           INSERT SHIFT ADDRESS            5
2354 TRP 2379 11                           TEST IF P PLUS D ≥ M            2
2359 SGN 0290 09                           SIGN OF TAG FOR FXP/FPR         4
2364 ADM 0290 09                           RESTORE TAG                     3
2369 ADM 2395 02                           CONVERT D OP TO C OR E          3
2374 RSU 0282 02                           RESTORE ASU SIGN TO -           3
2379 SUB 0300 11                           P - W - 1                       5
2384 TRP 4089 11     E08                   ERROR IF P ≥ W PLUS 1           2
2389 RAD 19   00      27/29/31             MANTISSA OF NUMBER       10,12,14
2394 SET 0XXX 00                           SET M PLUS 1, PLUS ABS. P
2399 [C/D/E] 0 00     P PLUS D - M         SHIFT D PLUS 1, PLUS P
2404 SPR 3    00     TW PLUS D PLUS 1      STORE IN LINE IMAGE

2409 RSU 0306 11                           REPEAT TALLY                    4
2414 TRZ 3224 11                           LENGTH AND TO NEXT COMMAND      2

2419 ADM 0306 02                           DIMINISH REPEAT TALLY BY 1      4
2424 ADM 2289 05                           AUGMENT ALPHA ADDRESS BY I      6
2429 TR  2254                              TO REPEAT THIS COMMAND          2
```

READ AND WRITE TAPE COMMANDS   WTI, RTI
READ AND WRITE CARD ON TAPES   RCD, WCD

```
2574 LOD 1283 10                          R    LOAD OP CODE FOR WRITE           3
2579 SGN 0289 00                               WTM/SHR TAG FOR SWITCH 2         4
2584 LOD 0300 09                          I    LOAD TAPE UNIT IDENT.            3
2589 UNL 2654 09                                                                3
2594 UNL 2789 09                                                                3
2599 RCV 2701                                  SET UP FIRST ADDRESS             2
2604 TMT 0292 01                                  FOR READ OR WRITE COMMAND     6
2609 SGN 2815 09                               SET SW. 1 TO WTM FOR WRITE       4
2614 SGN 2745 09                               SET SW. 2 TO WTM                 4
2619 TRP 2629 00                                                                2
2624 ADM 2745 09                               SET SW. 2 TO SHR IF TAG IS -     3
2629 SGN 2654 00                               READ/WRITE TAG FOR SW. 1         4
2634 TRP 2649 00                                                                2
2639 ADM 2815 09                               SET SW. 1 TO SHR FOR READ        3
2644 LOD 0438 10                          Y    CHANGE OP IN ASU 10 TO READ      3
2649 UNL 2700 10                        Y/R    INSERT PROPER OP CODE            3
2654 SEL 020[]                                 SELECT ITH TAPE UNIT             2
2659 IOF 0000                                  TURN OFF I/O INDICATOR           2
2664 LOD 0299 04                               BETA PLUS 1 ADDRESSES            6
2669 UNL 2684 04                                                                6
2674 UNL 2689 04                                                                6
2679 UNL 2719 04                                                                6
2684 LOD      09      BETA PLUS 1              CHARACTER FROM BETA PLUS 1       3
2689 RCV      09      BETA PLUS 1              REPLACE END CHARACTER            2
2694 TMT 0764 09                                  BY GROUP MARK                 3
2699 SET 0003 00                               SET MONITOR                      5
2704 [Y/R]            ALPHA-9/11/13            READ/WRITE
2709 RSU 0306 11                               REPEAT TALLY                     4
2714 TRA 2759                                                                   2
2719 UNL      09      BETA PLUS 1              RESTORE LAST CHARACTER           3
2724 TRZ 2749 11                               TEST FOR NON-REPEAT              2
2729 ADM 2704 05                               AUGMENT ALPHA ADDRESS            6
2734 ADM 0299 06                               AUGMENT LAST CHARACTER           6
2739 ADM 0306 02                               DIMINISH REPEAT TALLY            4
2744 TR  2664                                                                   2
2749 [3/C] 0001                                WTM OR SHR 00
2754 TR  3224                                  LENGTH AND TO NEXT COMMAND       2
2759 TRS 2804                                  TEST I/O INDICATOR               2
2764 SEL 0902                                                                   2
2769 TRS 2779                                  IF 0902 IS ON                    2
2774 TR  4139          E13                     ERROR - CHECK INDICATOR 0901     2
2779 NTR 2789 00                                                                3
2784 TR  4149          E 14                    READ/WRITE ERROR MESSAGE         2
2789 SEL 020                                   SELECT ITH TAPE UNIT             2
2794 BSP 0004                                  BACKSPACE ONE RECORD
2799 TR  2704                                  TRANSFER TO TRY AGAIN            2

2804 IOF 0000                                  TURN OFF INDICATOR               2
2809 TRZ 2819 11                               TEST REPEAT TALLY                2
2814 TR  4269          E15                     EOF BEFORE RPT EXHAUSTED         2
2819 3/C 0001                                  WTM OR SHR
2824 TR  0434                                  TR TO TRU TO ALPHA               2
```

```
                        WRITE COMMANDS          WL-, WH-

                                        +
3064 LOD 0290 11                 0̄0 OR XX    LINE COUNT                       4
3069 ADM 0292 02                             AUGMENT LINE TALLY               4
3074 RCV 3081                                INSERT  BADD                     2
3079 TMT 0331 01                                AS AN ADDRESS                 6
3084 RCV [BADD]                              SET TO RECEIVE MODIFIED COMM.    2
3089 CMP 0292 11                             COMPARE LINE COUNT WITH TALLY    4
3094 TRH 3109                      +         WRITE LINE IF HIGH               2
3099 LOD 1108 11                   00        RESET TALLY IF EQUAL LINE COUNT  4
3104 UNL 0292 11                                                              4
3109 TMT 0283 15                             RESTORE MODIFIED COMMAND        20
3114 TRE 0434                                TRU TO ALPHA ON NO LINE          2
3119 RAD 0295 00                             UNIT DESIGNATION                 5
3124 UNL 3159 00                                                              5
3129 UNL 3249 00                                                              5
3134 RCV 3254                                INSERT BSP OR SUP CONTROL        2
3139 TMT 0299 09                                ADDRESS CHARACTER             3
3144 RAD 1108 13                 000         SET MONITOR FOR ERROR            5
3149 RAD 0298 12                 XXX±        LINE OR HDG ADDRESS INSERT       5
3154 UNL 3164 12                               INSERTED IN WRITE COMMAND      5
3159 SEL 0[XXX]                              WRITE A LINE OR HDG              2
3164 WR  3[XXX] 00                              ON TAPE OR PRINTER
3169 TRA 3234                                                                 2
3174 RCV 3387                                INSERT CARRIAGE CONTROL          2
3179 TMT 0300 09                                CHARACTER IN                  3
3184 RCV 3265                                   LINE AND                      2
3189 TMT 0300 09                                HEADING IMAGES                3
3194 TRP 3214 12                             TEST IF LINE OR HDG              2
3199 SET 0026 00                             SET LINE IMAGE                  28
3204 TMT 2935 14                                TO BLANK IF A                 7
3209 NTR 3204 00                                LINE WAS WRITTEN
3214 TRP 3224 00                             TEST FOR TRIPLE SPACING          2
3219 WR  2458 00                             EXTRA SPACE FOR TRIPLE
3224 LOD 0226 01                 0018        COMMAND LENGTH                   6
3229 TR  0324                                TO NEXT COMMAND IN SEQUENCE      2

3234 SEL 0902                                TEST FOR WRITING ERROR           2
3239 TRS 3249                                   FROM MEMORY TO BUFFER         2
3244 TR  4099        E09                     ERROR MESSAGE FOR I/O,901,903    2
3249 SEL 0XXX                                SELECT TAPE OR PRINTER           2
3254 [  ] 000 ──────── 4 OR 5                BSP OR SUP
3259 NTR 3164 13                             TEST ERROR .ONITOR
3264 TR  4109        E10                     E MESSAGE ON EXHAUSTED MONITOR   2


                      CONTROL COMMANDS      BSI, RWI, TMI

3974 RCV 3994                                INSERT SELECT                    2
3979 TMT 0285 09                                ADDRESS AND                   3
3984 TMT 0286 14                                CONTROL INSTRUCTION           7
3989 RAD 0293 13                             COUNT FOR BACKSPACING            5
3994 SEL 020                                                                  2
3999  3  000
4004 SUB 0282 13                             REDUCE COUNT BY 1                5
4009 TRZ 0464 13                             LENGTH AND TO NEXT COMMAND       2
4014 TR  3999                                TO BACKSPACE AGAIN               2
```

8-DIGIT SQUARE ROOT ROUTINE   SQR

```
3594 RAD 1583 00                         .5                                        3
3599 MPY 1929 00                      PP.X    X IS 0 OR 5    PLUS OR -             8
3604 ST  1930 00                              POWER DIVIDED BY 2                   5
3609 SHR 0001 00                        PP                                         4
3614 ST  1929 00                              UNITS AND DECIMAL BOTH SIGNED        4
3619 RSU 1927 00               .XXXXXXXX∓     MINUS MANTISSA                      10
3624 TRZ 0574 00                              SQUARE ROOT EQUAL 0                  2
3629 TRP 4049 00   E04                        ERROR MESSAGE FOR SQR OF -           2
3634 RAD 1930 09                 5 OR 0±      PLUS OR -                            3
3639 TRZ 3674 09                              TR ON EVEN POWER                     2
3644 TRP 3654 09                              INCREASE ON ODD AND PLUS ONLY        2
3649 TR  3659                                                                      2
3654 ADM 1929 02                              SADM, CORRECTED POWER                4
3659 LOD 1111 09                              ASU 09 IS 0 IN BOTH CASES            3
3664 SET 0009 00               .0XXXXXXXX-                                        11
3669 TR  3679                                                                      2
3674 LNG 0001 00               .XXXXXXXX0-                          AVERAGE 13     4
3679 ST  1939 00                                                                  11
3684 UNL 1930 09                              0.                                   3
3689 SHR 0005 00                    .XXXX-                                         8
3694 ST  1953 00                              - ARG FOR LINEAR APPROX.             6
3699 SHR 0002 00                      .XX-    ARGUMENT FOR TLU                     5
3704 LOD 0380 11                        08    INCREMENT FOR TLU                    4
3709 RCV 3723                                 INITIALIZE CMP COMMAND               2
3714 TMT 1037 11                                 TO ADDRESS OF 3904                4
3719 ADM 3724 11                              START SEARCH AT 3912                 4
3724 CMP 39[ ] 00                                                                  4
3729 TRH 3719                                                       AVERAGE 45     2
3734 RCV 3748                                 INSERT ADDRESS OF SEGMENT            2
3739 TMT 3723 11                                 INTO TMT COMMAND                  4
3744 RCV 1940                                 MOVE SEGMENT COEFFICIENTS TO         2
3749 TMT 39[ ] 08                                WORKING POSITION (W .L.)         12
3754 RSU 1943 00                      X.X-    -A                                   4
3759 MPY 1953 00                   0.XXXXX    AX                                  18
3764 SHR 0002 00                     0.XXX                                         5
3769 ADD 1946 00                     X.XXX    1ST APPROX., EQUALS AX PLUS B        6
3774 ST  3898 00                              A1                                   6
3779 RSU 1939 00               0.XXXXXXXXX    ARGUMENT                            12
3784 SHR 0001 00                0.XXXXXXXX                                         4
3789 DIV 3898 00                    .XXXXX    Q1                                 245
3794 SGN 3898 10                              LNG A1 0002 IN MEMORY                4
3799 ADD 3900 00                   X.XXXXX    A1 PLUS Q1 APPROX. 2A2               8
3804 ST  1945 00                                                                   8
3809 RAD 1583 00                        .5                                         3
3814 MPY 1945 00                  X.XXXXXX                                        12
3819 SHR 0001 00                   X.XXXXX    A2 WITHIN .0005 OF SQR N             4
3824 ST  3906 00                              A2                                   8
3829 MPY 3906 00            0X.XXXXXXXXXX    A2 SQUARED                           62
3834 SHR 0001 00             0X.XXXXXXXXX                                          4
3839 ADD 1939 00             00.0000XXXXX    A2 SQUARED - N                       13
3844 SET 0006 00                    0XXXXX                                         8
3849 LNG 0005 00               0XXXXX00000                                         8
3854 DIV 1945 00                     XXXXX    -.5 DELTA                          321
3859 SGN 3906 09                              LNG A2 0004 IN MEMORY                4
3864 SUB 3910 00               0.XXXXXXXXX-   -A2 -.5 DELTA IS APPROX. SQR N      12
3869 RND 0001 00                X.XXXXXXXX-                                        6
3874 NTR 3884 00                 .XXXXXXXX-                                       11
3879 TR  0584                                 SQR .99999999 WITH EVEN POWER
3884 UNL 1927 00                              UNSIGNED MANTISSA                   10
3889 ADM 1927 09                              PLUS SIGN                            3
3894 TR  0584                                 RETURN TO GENERAL ROUTINE            2
```

```
4019 SET 0000 13                                                                     2
4024 SET 0010 13              0.000000000   INITIALIZE MPLR LOG ACCUM.              12
4029 RSU 1927 00               .XXXXXXXX    - MANTISSA OF ARGUMENT                  10
4034 TRP [4034] 00  E02                     ERROR FOR LOG OF 0 OR - NUM.             2
4039 ST  1927 00                            START OF MPLR LOOP                      10
4044 SHR 0007 00                       X    LEADING DIGIT                           10
4049 MPY 4390 00                            BY TABLE INCREMENT OF 11                 8
4054 SUB 4403 00                            MAKE TABLE ADDRESS                       5
4059 UNL 4079 00                            INSERT ADDRESS FOR                       5
4064 UNL 4084 00                              MPLR AND LOG OF MPLR                   5
4069 LOD 3969 12                            ADDRESS CONVERSION INCREMENT             5
4074 ADM 4079 12                            BOTH ADDRESSES COMPLETE                  5
4079 ADD [    ] 13                          ACCUMULATE LOGS OF MPLRS                12
4084 RAD [    ] 00                   X.X    MULTIPLIER                               4
4089 MPY 1927 00                            CONVERT ARGUMENT                        26
4094 RND 0001 00                            TO MANTISSA LENGTH                       6
4099 NTR 4039 00                            RETURN IF LESS THAN 1.0                 11
4104 SET 0008 00               .0XXXXXXX    ARGUMENT FOR                            10
4109 ST  1927 00                              POLYNOMIAL APPROXIMATION              10
4114 ST  1939 13                            STORE LOG SUM                           12
4119 RSU 2858 00                     .09    4TH ORDER COEFFICIENT                    3
4124 MPY 1927 00                                                                    14
4129 RND 0005 00                                                                    10
4134 ADD 4285 00                            3RD ORDER COEFFICIENT                    7
4139 MPY 1927 00                                                                    62
4144 RND 0006 00                                                                    11
4149 ADD 4292 00                            2ND ORDER COEFFICIENT                    9
4154 MPY 1927 00                                                                    86
4159 RND 0007 00                                                                    12
4164 ADD 4300 00                            1ST ORDER COEFFICIENT                   10
4169 MPY 1927 00                                                                    98
4174 RND 0007 00                                                                    11
4179 SUB 1939 00            X.XXXXXXXXX     LOGARITHM OF                            11
4184 ST  1939                .XXXXXXXXX       DECIMAL NUMBER                        11
4189 RAD 1929 00                            POWER                                    4
4194 LNG 0009 00           XX.000000000                                             12
4199 ADD 1939              XX.XXXXXXXXX                                             13
4204 RND 0001 00            XX.XXXXXXXX                                              6
4209 LOD 0296 09                            TAG TO DETERMINE BASE                    3
4214 TRZ 4229 09                            BYPASS CONVERSION IF BASE 10             2
4219 MPY 4413 00                                                                   142
4224 RND 0009 00                            BACK TO 8 DECIMALS                      14
4229 SET 0011 00           XXX.XXXXXXXX                                             13
4234 TRZ 0574 00                            IF LOG IS ZERO                           2
4239 LOD 0032 11                      03    BASE POWER                               4
4244 NTR 4269 00                                                                    14
4249 RND 0003                  .XXXXXXXX    TO MANTISSA LENGTH                       8
4254 ST  1927 00                            STORE MANTISSA                          10
4259 ST  1929 11                      XX    STORE ADJUSTED POWER                     4
4264 TR  0584                               EXIT TO GENERAL SUB-ROUTINE              2

4269 LNG 0001 00                            RESTORE TO 11 CHAR. AFTER NTR            4
4274 SUB 0388 11                            REDUCE BASE POWER                        4
4279 TR  4244                               TO TRY NTR AGAIN                         2
```

8-DIGIT EXPONENTIAL

```
4874 RAD 1927                      MANTISSA                  10
4879 LNG 0002                                                 5
4884 LOD 0296 09                   0 OR 1 TAG FOR BASE        4
4889 TRZ 4909 09                   0 FOR DECIMAL              2
4894 MPY 5432                      BASE E MPY BY            122
4899 NTR 5269                      LOG E BASE 10             21
4904 RND 0008                                                13
4909 RAD 1929 13                   TEST EXPONENT              4
4914 SUB 5399 13                   MINUS 3                    4
4919 TRP 4159 13    E16            ERROR MESSAGE              2
4924 ADD 4845 13                   EXP-3+11                   4
4929 TRP 4939 13                                              2
4934 TR  5289                      BY PASS CALC ANS=1         2
4939 SUB 0262 13                   USE EXP-2                  4
4944 UNL 4949 13                   TO CONVERT TO              4
4949 SHR 00[ ]                     FIXED POINT               13
4954 SET 0010                                                12
4959 ST  1929                                                12
4964 SET 0008                                                10
4969 SGN 1929 09                   PLACE SIGN ON              3
4974 ADM 1921 09                   EXOPNENT                   3
4979 ADM 1929 09                   MANTISSA                   3
4984 RAD 1921 13                   MAX ARG =98.XXXX           4
4989 CMP 3095 13                                              4
4994 TRH 4159       E16            ERROR MESSAGE              2
4999 TRP 5029                      MANTISSA SIGN              2
5004 ADD 5408                      NEG ADD ONE               11
5009 SET 0008                                                10
5014 ST  1929                      1-X                       10
5019 RAD 5459 13                   DECREASE EXP               4
5024 ADM 1921 13                   BY 1                       4
5029 LOD 1922 09                   LEADING DIGIT LOOK-UP      3
5034 UNL 5044 09                                              3
5039 UNL 5194 09                                              3
5044 RSU 539[]                     0,1,2, OR 3 TIMES          3
5049 MPY 5416                      LOG 2 BASE 10 SUBTRACT    14
5054 ADD 1929                      FROM ARGUMENT             11
5059 SET 0008                                                10
5064 MPY 5424                      LOG 10 BASE E             98
5069 SHR 0007                      FOR                       10
5074 SET 0008                      BASE E                    10
5079 ST  1929                                                10
5084 SHR 0007                      TLU 0,1,2...,6            10
5089 MPY 4845                      TIMES TABLE INCREMENT      8
5094 ADD 5463                      PLUS TABLE ORIGIN          6
5099 UNL 5199                      FOR ADDRESSES TO           6
5104 ADD 0338                      EXTRACT                    6
5109 UNL 5114                      TABLE ENTRIES              6
5114 RSU [    ]                    TABLE VALUE SUBT          10
5119 ADD 1929                      FROM ARGUMENT             10
5124 ST  1929                      POLYNOMIAL ARGUMENT       10
5129 MPY 5380                      5TH ORDER COEFF           74
5134 SHR 0007                                                10
5139 ADD 5438                      4TH ORDER COEFF            8
5144 MPY 1929                                                74
5149 SHR 0006                                                 9
5154 ADD 5388                      3RD ORDER COEFF           10
5159 MPY 1929                                                98
5164 SHR 0008                                                11
5169 ADD 5447                      2ND ORDER COEFF           11
```

```
5174 MPY 1929                                        110
5179 SHR 0008                                         11
5184 ADD 5408            1ST ORDER COEFF              11
5189 ST  1930                                         11
5194 RAD 130[]           1,2,4, OR 8                   3
5199 MPY [    ]          TIMES TABLE VALUE             9
5204 MPY 1930            TIMES POLYNOMIAL VALUE       54
5209 SHR 0002                                          5
5214 SET 0010                                         12
5219 RAD 5396 13         FLOATING PT CONVERSION        3
5224 NTR 5254                                         13
5229 ADD 1921 13                                       4
5234 ST  1929 13         EXPONENT                      4
5239 RND 0002                                          7
5244 ST  1927            MANTISSA                     10
5249 TR  0584            EXIT                          2
5254 SUB 5393 13         NORMALIZING                   3
5259 LNG 0001                                          4
5264 TR  5224                                          2
5269 RND 0007            ARGUMENT SHIFTED             12
5274 RSU 5393 13         EXPONENT DECREASED            3
5279 ADM 1929 13         BY                            3
5284 TR  4909                                          2
5289 RCV 1920            SET ANSWER TO ONE             2
5294 TMT 5448 08                                      12
5299 TR  0584                                          2
```

8-DIGIT ARC TANGENT

```
5469 RAD 1927                                         10
5474 TRZ 0584            EXIT ON ZERO                  2
5479 TRP 5489                                          2
5484 RSU 1927                                         10
5489 ST  6908                                         10
5494 RAD 1929 03         TEST RANGE OF EXP             4
5499 CMP 0303 03         8                             4
5504 TRH 5894                                          2
5509 CMP 0210 03         1                             4
5514 TRH 5864                                          2
5519 TRP 5559 03                                       2
5524 SET 0009                                         11
5529 UNL 5534 03         USE EXPONENT OT               4
5534 SHR 00[ ]           CONVERT TO FXD PT             6
5539 SET 0008            LET G=X                      11
5544 RAD 0593 13         ZERO FOR                      3
5549 ST  1944 13         ARCTAN X SUB N                3
5554 TR  5724            BY-PASS TLU                   2
5559 UNL 5564 03         USE EXP TO                    4
5564 LNG 00[ ]           CONVERT TO FXD PT             4
5569 SHR 0006            PREPARE ARGUMENT              9
5574 SET 0003            FOR TLU                       5
5579 RCV 5596            INITIALIZE                    2
5584 TMT 6047 04         TLU ADDRESS                   5
5589 LOD 0222 04         TABLE INCREMENT (17)          5
5594 ADM 5599 04         TLU                           5
5599 CMP [    ]                                        5
5604 TRH 5594                                          2
5609 LOD 5599 04         EXTRACT AND                   5
5614 ADD 3939 04         INCREASE                      5
5619 UNL 5629 04         TABLE ADDRESS                 5
5624 SET 0014            EXTRACT ENTRY                16
```

```
5629 LOD [   ]                                      16
5634 UNL 1944                                       16
5639 RAD 1934         TABLE ARGUMENT                 5
5644 MPY 6908         TIMES MANTISSA                38
5649 SHR 0002                                        5
5654 ST  1953                                       11
5659 UNL 5669 03      EXPONENT TO ADJUST             4
5664 RAD 5408         POWER OF TEN                  11
5669 SHR 00[ ]        IN SCALING                     4
5674 ADD 1953         PLUS PRODUCT                  11
5679 ST  1953                                       11
5684 RSU 1934         MINUS TABLE VALUE              5
5689 LNG 0006         SHIFTED LEFT                   9
5694 UNL 5699 03      EXPONENT TO                    4
5699 SHR 00[ ]        SCALE TABLE VALUE              4
5704 ADD 6908         PLUS ARGUMENT                 11
5709 SET 0009         PREPARE                       11
5714 LNG 0008         NUMERATOR TO DIVIDE           11
5719 DIV 1953         QUOTIENT IS G                688
5724 ST  1953         STORE G                       10
5729 MPY 1953         G SQUARED                     98
5734 SHR 0008                                       11
5739 ST  6908         POLYNOMIAL                    10
5744 RAD 6020         A2                             6
5749 MPY 6908         TIMES G SQUARE                50
5754 SHR 0005         SHIFTED                        8
5759 ADD 6027         A1                             9
5764 MPY 6908         TIMES G SQUARE                86
5769 SHR 0006         SHIFTED                      009
5774 ADD 6037         A0                            11
5779 MPY 1953         TIMES G                      110
5784 SHR 0008         SHIFTED                       11
5789 ADD 1944         PLUS TABLE ARCTAN             11
5794 TRZ 0584         ARCTAN X=X                     2
5799 RAD 5457 13      CONVERT TO                     3
5804 NTR 5849         FLOATING PT                   13
5809 SHR 0002         RETAIN 8 DIGIT MANT            5
5814 ST  6908         STORE MANTISSA                10
5819 ST  1929 13      EXPONENT                       4
5824 SGN 1927 13      TEST INPUR ARG SIGN            3
5829 TRP 5839 13      NEG INPUT                      2
5834 RSU 6908         NEG ANSWER                    10
5839 ST  1927         MANTISSA STORED               10
5844 TR  0584         EXIT                           2
5849 SUB 0282 13      DECREASE EXPONENT              3
5854 LNG 0001         SHIFT MANTISSA                 4
5859 TR  5804         RETURN TO NORMALIZE            2
5864 TRP 5874         TEST SIGN OF EXP               2
5869 TR  5524         NEG G=X                        2
5874 SET 0014 13      EXTRACT LAST                  15
5879 RCV 1931         TABLE ENTRY                    2
5884 TMT 6004 13      FOR G CALCULATION             15
5889 TR  5639         BYPASS TLU                     2
5894 TRP 5904 03      TEST SIGN OF EXP               2
5899 TR  0584         EXIT F(X)=X                    2
5904 RAD 5457 13      ANSWER IS                      4
5909 RAD 6045         PI/2                          10
5914 TR  5814         GO TO OUTPUT ROUTINE           2
```

SINE AND COSINE COMMAND   SAC

```
6059 LOD 6797 09                    COSINE SWITCH IS           3
6064 UNL 6630 09                    INITIALIZED TO NOP         3
6069 RAD 1929                       TEST EXPONENT              4
6074 CMP 6805                       FOR 7 OR LESS              4
6079 TRH 6719                                                  2
6084 SUB 1306                       X-46                       4
6089 TRP 4259        E03            ERROR                      2
6094 ADD 1300                                                  4
6099 UNL 6124                                                  4
6104 RAD 1927                       MANTISSA                  10
6109 TRP 6119                                                  2
6114 RSU 1927                       USE POSITIVE ARGUMENT     10
6119 LNG 0003                       PREPARE TO                 6
6124 SHR 00                         CONVERT TO FIXED PT       11
6129 SET 0011                                                 13
6134 MPY 6877                       2 PI INVERSE             123
6139 RND 0008                                                 13
6144 SET 0009                       DISCARD INTEGRAL PT       11
6149 ST  1938                       FRACTION OF 2PI           11
6154 RAD 6886                       QUADRANT REDUCTION        11
6159 SUB 1938                       .5-X                      11
6164 TRP 6224                                                  2
6169 ADD 6895                                                 11
6174 TRP 6199                                                  2
6179 RAD 1938                       4TH QUAD                  11
6184 SUB 6835                       X-1                       12
6189 SET 0009                                                 11
6194 TR  6244                                                  2
6199 RAD 6886                       2ND OR 3RD QUAD           11
6204 SUB 1938                       .5-X                      11
6209 LOD 6744 09                    COSINE SWITCH TO TR        3
6214 UNL 6630 09                    FOR NEG ANSWER             3
6219 TR  6244                                                  2
6224 SUB 6895                                                 11
6229 TRP 6239                                                  2
6234 TR  6199                                                  2
6239 RAD 1938                       1ST QUAD USE X            11
6244 MPY 1306                       4X FOR                    47
6249 SET 0010                       FRACTION OF PI/2          12
6254 ST  1939                                                 12
6259 RCV 6282 12                    INITIALIZE TLU             2
6264 TMT 6050 12                    CMP ADDRESS                5
6269 RAD 6898 12                    TABLE INCREMENT            5
6274 SHR 0008                       TABLE LOOK UP DIGITS      10
6279 ADM 6284 12                    TLU ON                     5
6284 CMP 6                          0.1,0.3,0.5,0.7,0.9,9.9    5
6289 TRH 6279                                                  2
6294 TRE 6279                                                  2
6299 RCV 6312 12                    USE COMP ADDRESS           2
6304 TMT 6282 12                    TO EXTRACT                 5
6309 SET 0019                       TABLE                     21
6314 LOD 6                                                     5
6319 UNL 1959                                                  5
6324 RAD 1939                       FRACTION OF PI OVER 2     12
6329 TRP 6349                                                  2
6334 RSU 1927 13                    FOR QUAD 3,4              10
6339 ST  1927 13                    SINE WILL BE MINUS        10
6344 RSU 1939                       ARGUMENT                  12
6349 SUB 1951                       MINUS TABLE VALUE         12
```

```
6354 RND 0001                                EQUALS                  6
6359 SET 0008                                POLY NOMIAL            10
6364 ST  1939                                ARGUMENT Z             10
6369 MPY 1939                                SQUARE OF              98
6374 RND 0008                                POLYNOMIAL             10
6379 ST  1951                                ARGUMENT Z             10
6384 RAD 1309                                SINE POLYNOMIAL         3
6389 MPY 1951                                                       14
6394 RND 0005                                                        7
6399 ADD 6859                                                        7
6404 MPY 1951                                                       62
6409 RND 0005                                                        7
6414 ADD 6868                                                       11
6419 MPY 1939                                                      110
6424 RND 0008                                                       10
6429 ST  1939                                                       11
6434 RAD 6847                                COSINE POLYNOMIAL       6
6439 MPY 1951                                                       50
6444 RND 0006                                                        8
6449 ADD 6854                                                        9
6454 MPY 1951                                                       79
6459 RND 0006                                                        8
6464 ADD 5408                                                       11
6469 ST  1951                                                       11
6474 RAD 1954                                SINE Y                  5
6479 MPY 1951                                TIMES COS Z            41
6484 RND 0002                                                        4
6489 ST  6908                                                       12
6494 RAD 1957                                COS Y                   5
6499 MPY 1939                                TIMES SINE Z           41
6504 RND 0002                                                        4
6509 ADD 6908                                SINE X                 12
6514 TRZ 6689                                                        2
6519 RAD 2252 13                             PREPARE TO              4
6524 NTR 6659                                NORMALIZE              13
6529 ST  1929 13                             STORE EXPONENT          4
6534 SHR 0002                                TRUNCATE TO 8 DIGITS   10
6539 RAD 1927 13                             TEST INPUT             10
6544 TRP 6559 13                             FOR SIGN                2
6549 ST  6908                                CHANGE                 10
6554 RSU 6908                                SIGN                   10
6559 ST  1927                                STORE MANTISSA         10
6564 RAD 1954                                SINE Y                  5
6569 MPY 1939                                TIMES SIGN Z           41
6574 RND 0002                                                        7
6579 ST  6908                                                       12
6584 RAD 1957                                COS Y                   5
6589 MPY 1951                                TIMES COS Z            41
6594 RND 0002                                                        7
6599 SUB 6908                                COS X
6604 TRZ 6704
6609 RAD 2252 13                             PREPARE TO
6614 NTR 6674                                NORMALIZE
6619 SHR 0002                                TRUNCATE TO 8 DIGITS
6624 ST  1937                                STORE MANTISSA
6629 ST  1939 13                             EXPONENT
6634 A/1 6644                                SWITCH FOR COS IN 2,3Q
6639 TR  6914                                EXIT
6644 RSU 1937                                CHANGE COS SIGN
```

```
6649 ST   1937                                   FOR QUAD 2,3
6654 TR   6639
6659 LNG 0001                                    NORMALIZE
6664 SUB 5393 13                                 FOR SINE X
6669 TR   6524
6674 LNG 0001                                    NORMALIZE
6679 SUB 5393 13                                 FOR COS X
6684 TR   6614
6689 RCV 1920                                    SET
6694 TMT 0265 08                                 SIN X=0
6699 TR   6564
6704 RCV 1930                                    SET
6709 TMT 0265 08                                 COS X=0
6714 TR   6639
6719 TRP 4259         E03                        ERROR MSG ON PLUS
6724 SET 0000                                    SET X=0
6729 TR   6129                                   ON MINUS
 .

6914 LOD 0282 01                                 COMMAND LENGTH          6
6919 RCV 6926                                    SET UP RCV TO STORE     2
6924 TMT 0296 01                                 COSINE IN GAMMA         6
6929 RCV              GAMMA-9/11/13                                      2
                                                                 12,14,16
6934 TMT 1930 08
6939 ADM 6929 07                                 AUGT. GAMMA ADDS. BY K  6
6944 TR   0584                                   TO EXIT                 2
```

Appendix II—Example I

Generalized matrix multiplication

The coding symbolized here is a basic method to effect the multiplication of a (k by m) matrix and an (m by n) matrix to produce the product matrix (k by n). It is valid when sufficient memory locations can be reserved for the elements of all three matrices. Many of the features of PRINT I as applied to repetitive operations are illustrated in this general method.

The common practice in assigning memory to the elements of a matrix is to store them row-by-row in sequential addresses. In this case these sequential addresses are regional for convenience. The general elements of these matrices are to be in locations defined as:

Matrix	*Location*
(k by m)	A001 + (row—1) (m) + (col—1)
(m by n)	B001 + (row—1) (n) + (col—1)
(k by n)	C001 + (row—1) (n) + (col—1)

The program is generally written as below, with the proper numbers replacing k, m and n. All operations are shown with referrals in the location column, but steps 3, 4 and 8 are the only ones which need it.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14- COMMENTS -80
STEP 1	SR 1	0
STEP 2	SR 2	0, km
STEP 3	SR 3	0, n
STEP 4	RWR	m, 1, n
STEP 5	MAD	A001, 2, B001, 3, C001, 123
STEP 6	TX 3	STEP 4, 1
STEP 7	TN 1	STEP 8, (n - m)
STEP 8	TX 2	STEP 3, m

If either of the two matrices to be multiplied are square, it should be used as the multiplicand. Under these conditions the 1st and 7th steps may be eliminated because m = n. Elements are computed row-wise in the example, for efficient storage, but the problem could be re-coded to develop them column-wise. Printing of a row at a time during calculation is possible by inserting the necessary operations between steps 6 and 7, or 7 and 8.

A coding kernel is given here for matrix multiplication using tapes for input of elements. This is again for the multiplication of a (k by m) matrix by a (m by n) matrix. The (k by m) matrix is assumed to be stored on tape 0207 in k records, each record consisting of the m elements of a row. The (m by n) matrix is assumed to be stored on tape 0208 in n records, each record consisting of the m elements of a column. The first row and column are the first records of the respective tapes, etc. The product matrix is to be stored in row records on tape 0209.

LOCATION -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
RØWS	RT7	A001	m elements in each of k rows
	SR2	0, n	
CØLS	RT8	B001	m elements in each of n columns
	RWR	m, 1, 1	
	MAD	A001, B001, C001, 2	
	TX2	CØLS, 1	
	WT9	C001, C00n	
	RW8		
	ATR	RØWS, (k-1), NXTCM, 1	
NXTCM			

Appendix II—Example II

Three examples of efficient coding for polynomial evaluation are illustrated below. As a general rule, requiring more instructions than are shown here will indicate inefficient assignment of memory for arguments and coefficients. The programmer who has occasion to use this type of coding may profit by extension of these examples.

1. *Evaluating a Single Polynomial at N Arguments (N = 14)*

$$P(X) = \sum_{0}^{6} a_i X^i$$

Arguments X located in	K203 (2) K229
P (X)s to be sent to	K204 (2) K230
Coefficients a_i in	(D006 + i)

2. *Evaluating N Polynomials at a Single Argument (N = 5)*

$$P_j(X) = \sum_{0}^{6} (a_j)_i X^i$$

Argument X located in	K203
P_j (X)s to be sent to	(D013 + 8j)
Coefficients $(a_j)_i$ in	(D006 + i + 8j)

3. *Evaluating the Bi-variate Surface*

$$Z = \sum_{i=0}^{i=4} a_i Y^i \quad \text{where } a_i = \sum_{j=0}^{j=3} b_{ij} X^j$$

a_i are in (C005 + 5i), b_{ij} are in (C001 + 5i + j), X is in C026, Y is in C027 and the answer Z is to be placed in C028.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
	SR 1	0, 28	
NXTAR	RWR	7, -1	
	PMA	D012, K203, 1, K204, 1	
	TX 1	NXTAR, 2	
	SR 1	0, 40	
NXTPN	RWR	7, -1	
	PMA	D012, 1, K203, D013, 1	
	TX 1	NXTPN, 8	
	SR 2	0, 25	
NXTCØ	RWR	4, -1	
	PMA	C004, 2, C026, C005, 2	
	TX 2	NXTCØ, 5	
	RWR	5, -5	
	PMA	C025, C027, C028	

Appendix II—Example III

Matrix inversion

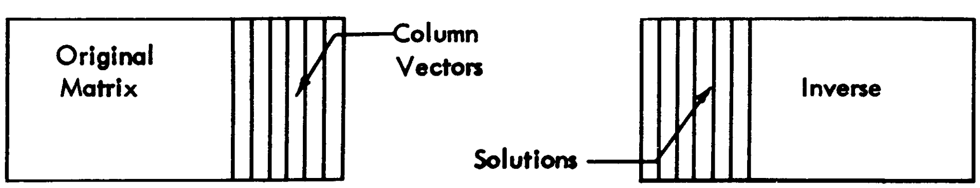
The coding kernel below gives a simple method for the inversion of a matrix and solution of simultaneous equations. It is adopted from R. DeSio's 650 program, using Jordan's method. The nth order matrix with b column vectors furnishes the array:

$$\begin{matrix} a_{11} & a_{12} & a_{13} & a_{14} \ldots\ldots\ldots\ldots a_{1n} & V_{11} & V_{12} & V_{13} \ldots\ldots\ldots\ldots V_{1b} \\ a_{21} & a_{22} & a_{23} & a_{24} \ldots\ldots\ldots\ldots a_{2n} & V_{21} & V_{22} & V_{23} \ldots\ldots\ldots\ldots V_{2b} \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & & a_{nn} & V_{n1} & V_{n2} & \ldots\ldots\ldots\ldots\ldots V_{nb} \end{matrix}$$

This array is stored row-wise in memory from A001 to A000 + n(n + b), each row starting at A001 + (row — 1) (n + b). A001 + n (n + b) thru A000 + (n + 1) (n — b) are reserved as working storage, for a total of (n + 1) (n + b) words.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
REDUC	DIV	LØC 1, A001, A000 + (n+1)(n+b)	(Reciprocal of first element)
	RPT	(n+b-1), 1, 0, 1	
	MPY	A002, A000+(n+1)(n+b), A001 + n(n+b)	
	SR1	0, (n-1)(n+b)	
UPRØW	RPT	(n+b), 0, 1, 1	
	MMY	A001+(n+b), 1, A001 + n(n+b), A001, 1	
	RPT	(n+b-1), 1, 1, 1	
	ADD	A002 + (n+b), 1, A001, 1, A001, 1	
	TX1	UPRØW, (n+b)	
	RPT	(n+b), 1, 1	
	TMT	A001 + n(n+b), A001 + (n-1)(n+b)	
	ATR	REDUC, (n-1), NXTCM, 1	(Counting control)
NXTCM			

n reductions are required, giving a new array in the identical block of memory positions, according to the following schematic:



Orders of matrices which may conveniently be inverted in memory are:

Memory size	*8-digit mantissa*	*10-digit*	*12-digit*
20,000	38	35	33
40,000	58	53	49

A 10th order matrix may be inverted in 10 seconds, 20th in 1 min., 20 sec., and 25th in 2 min., 36 sec. The 50th order in memory takes 20 min., 40 sec., and approximately 25 min. on tape. The program kernel for inversion of a matrix stored on tape is shown below. Each record on tape 0208 is a row of the combined array. As shown, the coding is limited to 99th order because of the RPTs; insertion of multiple RPTs will increase the order capability.

LOCATION 6- -10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
RTAP8	RT8	A001	
	ATR	RØW1, 1, RØWN, (n-1)	
RØW1	DIV	LØC1, A001, A000 + 3(n+b)	
	RPT	(n+b-1), 1, 0, 1	
	MPY	A002, A000 + 3(n+b), A001+2(n+b)	
	ATR	RTAP8, 1, RTAP9, 1	
RØWN	RPT	(n+b), 0, 1, 1	
	MMY	A001, A001 + 2(n + b), A001 + (n + b)	
	RPT	(n + b - 1), 1, 1, 1	
	ADD	A002, A001 + (n + b), A001 + (n+b)	
	ATR	WTAP9, (n - 1), WTAP8, (n - 1)	
WTAP9	WT9	A001 + (n + b), A000 + 2(n + b)	
	ATR	RTAP8, (n - 2), NRØW1, 1	
NRØW1	WT9	A001 + 2(n + b), A000 + 3(n + b)	
RWTPS	RW8		
	RW9		
	ATR	RTAP9, 1, RTAP8, 1	
RTAP9	RT9	A001	
	ATR	RØW1, 1, RØWN, (n - 1)	
WTAP8	WT8	A001 + (n + b), A000 + 2(n + b)	
	ATR	RTAP9, (n - 2), NRØW2, 1	
NRØW2	WT8	A001 + 2(n + b), A000 + 3(n + b)	
	ATR	RWTPS, n, NXTCM, 1	

Appendix II—Example IV

This program is to prepare a table of (X + cos Y)(X — sin Z), Y and Z being radian arguments. Each line is for 1 value of X, each page for 1 value of Z. Columns are for Y.

SERIAL 1- -5	LOCATION 6-	-10	OPERATION CODE 11- -13	VARIABLE FIELD 14-	COMMENTS -80
01010			HDG		Housekeeping. The line descriptions and head-
01020			HDG		ings are not coded for purposes of this example.
01030	R008		REG	R001	Reserve
01040	SINE	Z	REG		working
01050	TEM	P	REG		storage
01060	X020		REG	X001	Reserve storage
01070	Y008		REG	Y001	for loading
01080	Z010		REG	Z001	input data
01090			ENT		
01100			RCD	READER	Read 2 data cards. The first con-
01110			RPT	20, *4, 1	tains 20 values of X as (xx.xx),
01120			FLØ	CØL04, 4, L2, X001	stored in X001 to X020. The sec-
01130			RCD	READER	ond contains 8 values of Y as
01140			RPT	8, *5, 1	(xx.xxx), stored in Y001 to Y008,
01150			FLØ	CØL05, 5, L3, Y001	and 10 values of Z as (x.xxx),
01160			RPT	10, *4, 1	stored in Z001 to Z010.
01170			FLØ	CØL44, 4, L3, Z001	
01180			SR 3	0, 10	Control for number of pages (Z)
01190	PAG	E	WHT	TAPE 4	Heading, 2 spaces before lines
01200			SR 1	0, 20	Control for number of lines (X)
01210			SAC	Z001, 3, SINEZ	Compute sinZ for each page
01220			TRU	RSETY	
01230	LINE		WLS	TAPE 4	Write single-spaced result line
01240	RSET	Y	SR 2	0, 8	Control for number of columns (Y)
01250			SUB	X001, 1, SINEZ, TEMP	X - sinZ in TEMP (for each line)
01260	CØLM	N	SAC	Y001, 2	cosY in PAC 2
01270			ADD	X001, 1, PAC 2	X + cosY in PAC 1
01280			MPY	,TEMP, R001, 2	R001 through R008
01290			TX 2	CØLMN, 1	Re-cycle inner loop
01300			RPT	8, 1, *11	Fix for print rounded the line of 8
01310			FPR	R001, 8, 4W, 2D	values, R001 to R008 (xxxx.xx±)
01320			TX 1	LINE, 1	To write present line, ≠ 20th
01330			WL 1	TAPE 4	Write 20th line, skip over fold
01340			TX 3	PAGE, 1	to new page on channel 1 control
01350			LVE		To 705 HLT on completion of 10th page.

IBM®

INTERNATIONAL BUSINESS MACHINES CORPORATION, 590 MADISON AVE., NEW YORK 22, NEW YORK

32-7334

Litho in U.S.A.