File No. 360-25
GY28-2020-2

Program Logic

Version 8.1

IBM System/360 Time Sharing System FORTRAN IV Library Subprograms

This publication describes the internal logic of the IBM System/360 Time Sharing System FORTRAN IV mathematical and I/O libraries, including the mathematical, service, and I/O routines.

This material is intended for persons involved in program maintenance, and system programmers who are altering the program design. It can be used to locate specific areas of the program, and it enables the reader to relate these areas to the corresponding program listings. Program logic information is not necessary for the use and operation of the program.

PREFACE

This publication is organized into six sections and two appendixes.

Section 1 is an introduction to the FORTRAN mathematical and I/O libraries, including service subprograms, and provides a brief description of their contents and functions.

Section 2 contains descriptions, in figure form, of how the library routines interact in fulfilling user requests.

Section 3 describes each mathematical subprogram -- its entry names, function, attributes, entry, exit, storage requirement, error checks, and (where applicable) accuracy figures.

Section 4 describes the service routines residing in the mathematical and I/O libraries -- their subprograms, attributes, entry names and entry parameters, storage requirements, error checks, and their operation.

Section 5 describes each I/O routine -- its purpose, entry point and entry parameters, external references, and the details of its operation.

Section 6 gives the flowcharts of the service and I/O routines.

Appendix A describes those aspects of FORTRAN data management that are unique to the FORTRAN environment. Appendix B is a guide to external names of FORTRAN library routines.

PREREQUISITE PUBLICATIONS

Familiarity with the material contained in the following publications is essential to the use of this manual:

IBM System/360 Time Sharing System:
- Concepts and Facilities, GC28-2003
- IBM FORTRAN IV, GC28-2007
- FORTRAN Programmer's Guide, GC28-2025

REFERENCE PUBLICATIONS

Knowledge of the following publications will be helpful in understanding the concepts and logic of the FORTRAN Library routines:

IBM System/360 Time Sharing System:
- FORTRAN IV Library Subprograms, GC28-2026
- System Programmer's Guide, GC28-2008
- Assembler User Macro Instructions, GC28-2004
- FORTRAN IV Compiler PLM, GY28-2019
- System Control Blocks PLM, GY28-2011
- System Logic Summary, GY28-2009

Third Edition (September 1971)

Significant changes or additions to this publication will be provided in new editions or Technical Newsletters.

This edition is current with Version 8, Modification 1 of IBM System/360 Time Sharing System (TSS/360) and remains in effect for all subsequent versions or modifications of IBM System/360 Time Sharing System unless otherwise indicated. Before using this publication, refer to the latest edition of IBM System/360 Time Sharing System: Addendum, GC28-2043, which may contain information pertinent to the topics covered in this edition. The Addendum also lists the editions of all TSS/360 publications that are applicable and current.

This publication was prepared for production using an IBM computer to update the text and to control the page and line format. Page impressions for photo-offset printing were obtained from an IBM 1403 Printer using a special print chain.

Requests for copies of IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form is provided at the back of this publication for reader's comments. If the form has been removed, comments may be addressed to IBM Corporation, System/360 Time Sharing System Programming Publications, Department 643, Neighborhood Road, Kingston, New York 12401

ILLUSTRATIONS

The two FORTRAN IV libraries, mathematical and input/output (I/O), are written in assembler language and stored in the system library (SYSLIB). They are available to all users.

Each routine in the mathematical library is a collection of one or more subprograms. In this publication, the term subprogram means a routine or part of a routine that has a single entry point and performs or controls the performance of a single function. Mathematical library subprograms are generally mathematical or computational in nature, and generally return one answer (function value) to the calling program. Mathematical subprograms can be categorized by use:

1. Direct reference, as in reference to the sine subprogram in the statement

   X = SIN(Y)

2. Indirect reference, as in reference to an exponentiation subprogram in the statement

   X = Y**I

The I/O library is a group of routines that function as a single program complex for processing the I/O statements READ, WRITE, PRINT, PUNCH, BACKSPACE, REWIND, and END FILE. Processing of READ and WRITE statements can include list control, NAMELIST control, FORMAT control, or none of these controls. The routines within this complex can be categorized by function:

1. Language control routines, which analyze the user's I/O request.

2. Data conversion Routines, which convert data from internal to external form or from external to internal form.

Both libraries also contain service routines, each of which is a collection of subprograms. Service subprograms are called with CALL statements or are implicitly called by the occurrence of certain situations during execution. The service subprograms in the mathematical library handle machine exceptions and test program-simulated machine indicators. The service subprograms in the I/O library dump program data onto SYSOUTs and terminate execution of user programs.

Each of the two libraries also contains an error-handling routine.

This section explains the subdivisions of the FORTRAN IV mathematical and I/O libraries and gives in figure form an overview of each library. Note that the service subprograms are divided between the two libraries.

MATHEMATICAL LIBRARY

SUBDIVISIONS

The FORTRAN IV mathematical library consists of two types of relocatable routines: mathematical and service. Each routine contains one or more subprograms; there is a separate entry point for each subprogram.

Like a FORTRAN subprogram defined with a FUNCTION statement, a mathematical subprogram always returns an answer (function value) to the calling program. Mathematical subprograms can be categorized as direct reference or as indirect reference.

Since the user refers to direct-reference subprograms by name, they are explicitly called. For example, in the statement X = SIN(Y), the user invokes the SIN subprogram. The direct-reference mathematical subprograms are the logarithmic, trigonometric, hyperbolic, square root, absolute value, gamma function, and error function subprograms, and some exponentiation subprograms.

Since the user does not refer to indirect-reference subprograms by name, they are implicitly called. For example, to invoke an exponentiation subprogram, the user employs the exponentiation operator (i.e., **). The FORTRAN compiler then causes the user program to call the appropriate exponentiation subprogram. To invoke CHCBZA, the error-handling subprogram, the user invokes a mathematical subprogram without specifying the proper arguments. With the exception of CHCBZA, all indirect-reference mathematical subprograms are exponential (compute powers of numbers).

The mathematical library service subprograms can also be categorized as direct-reference or as indirect-reference. The direct-reference subprograms SLITE, SLITET, OVERFL, and DVCHK test program-simulated machine indicators. The indirect-reference subprograms CHCBD1, CHCBD2, CHCBD3, CHCBD4, CHCBD5, and CHCBE1 handle interruptions resulting from machine exceptions.

Each entry to an indirect-reference mathematical subprogram (and some entries to service subprograms) in the mathematical library is made with two entry names:

1. Compiler-generated entry name. When the user employs the exponentiation operator, the compiler examines the terms on both sides of the operator and determines which exponentiation subprogram to call. It then generates an entry name that is later translated into the system entry name.

2. System entry name. This is the name defined by the subprogram itself, the name of the entry point.

Each entry to a direct-reference subprogram is made with the name employed by the user. (The system entry name is the same as the user-employed name.)

INTERNAL RELATIONSHIPS

Figure 1 gives a functional overview of the mathematical library. The subprogram entry names of each routine are shown under the name of the routine. The entry names shown for the directly referenced subprograms are the user entry names. System entry names of these subprograms are shown in Table 7. The entry names shown for the indirectly referenced subprograms are the system entry names of these subprograms are shown in Table 8. The error-processing routine CHCBZ is not included in Figure 1.

I/O LIBRARY

Since the user has 10 entry points to the FORTRAN IV I/O library, this library can be thought of as 10 subprograms and a number of subroutines for these subprograms. The I/O library can also be described as a group of 21 routines that interact in various ways, depending upon the user's request. Except in "Section 4: Service Subprograms," this publication describes the I/O library in terms of its routines.

Three of the ten entry points to the I/O library are for processing the I/O statements READ, WRITE, REWIND, BACKSPACE, END FILE, PRINT, and PUNCH. Processing of READ or WRITE statements can include list control, NAMELIST control, FORMAT control, or none of these controls. The other seven entry points are to service subprograms, in two routines, that execute the statements STOP, PAUSE, CALL DUMP, CALL PDUMP, and CALL EXIT, and act upon requests by other library routines for termination of program execution.

SUBDIVISIONS

There are two groups of I/O routines: I/O language control routines and data conversion routines. These groups interact, in fulfilling an I/O request, by means of a common communication and work region.

I/O Language Control Routines

There are three types of I/O language control routines: I/O operation control, I/O list control, and I/O services control. These routines analyze the user's I/O requests to determine information such as: the type of I/O operation to be performed; the number and type of list items present, if any; the type of NAMELIST or FORMAT control, if any; and the I/O statement relationships with a user-specified DDEF command.

I/O OPERATION CONTROL ROUTINES: These routines control the I/O request by creating, if necessary, a data control block (DCB), and analyzing FORMAT and NAMELIST control specified by the user. After this information is processed, the I/O operation control routines interface with the TSS/360 data management routines that actually fulfill the I/O request. The interface with data management is accomplished by the routines CHCIB and CHCIC, via the data management macro instruction facilities.

I/O LIST CONTROL ROUTINES: These routines examine the list items, if any, in each I/O request to determine the type of conversion to be performed. After the type of data conversion is determined, control is given to the I/O operation control routines which in turn call the appropriate data conversion routines for final processing.

I/O SERVICES CONTROL ROUTINES: These routines are responsible for creating user-requested dumps, initiating a pause in the user program, and terminating a user program as a result of either normal processing completion or program errors. I/O services control routines also interact with the I/O operation control and with the group of data conversion routines for the preparation of the user's data in virtual storage for subsequent output on SYSOUT.

Data Conversion Routines

The data conversion routines are subdivided into routines used for input processing and routines used for the preparation of output. These routines can process all the permissible types of FORTRAN-formatted data specified in either a FORMAT or NAMELIST statement.

When converting a user's data, the data conversion routines interact with each other according to the requirements of the user-specified FORMAT or NAMELIST control. For example, for input data that is defined by a G-format conversion code, General Input Conversion (CHCIS) is called. This routine analyzes the data to determine whether it is integer, real, logical, or alphameric and calls the appropriate data conversion routine.

INTERNAL RELATIONSHIPS

Figure 2 gives a functional overview of the I/O library.

Table 1 shows the calling relationships between the user program, the FORTRAN I/O library routines, Data Management, and the Supervisor.

Since the routine functions and interrelationships vary, depending on the kind of I/O operation being performed, separate diagrams (Figures 3-14) describe the different kinds of I/O operations. Exceptions to the logical flows shown in Figures 3-14 are covered under the individual routine descriptions in "Section 5: I/O Routine Descriptions."

The types of I/O operations and their figure references are:

Type of Operation (Function)	Figure Reference
Formatted READ with List	Figure 3
Formatted READ without List	Figure 4
READ with NAMELIST	Figure 5
Unformatted READ with List	Figure 6
Unformatted READ without List	Figure 7
Formatted WRITE with List	Figure 8
Formatted WRITE without List	Figure 9
WRITE with NAMELIST	Figure 10
Unformatted WRITE with List	Figure 11
Unformatted WRITE without List	Figure 12
REWIND, BACKSPACE, and END FILE	Figure 13
EXIT and STOP	Figure 14

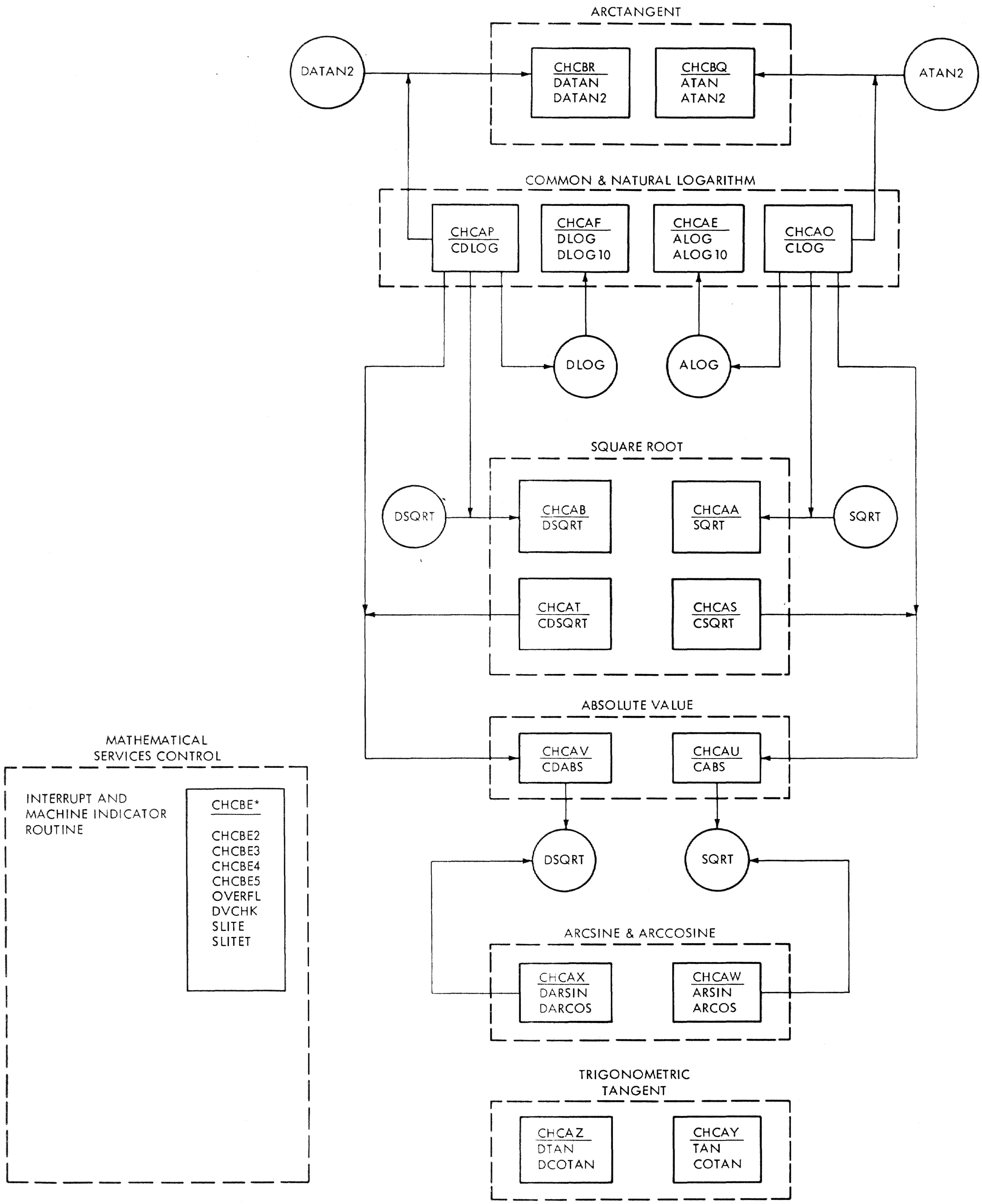


Figure 1. Functional flow of mathematical library subprograms (page 1 of 2)

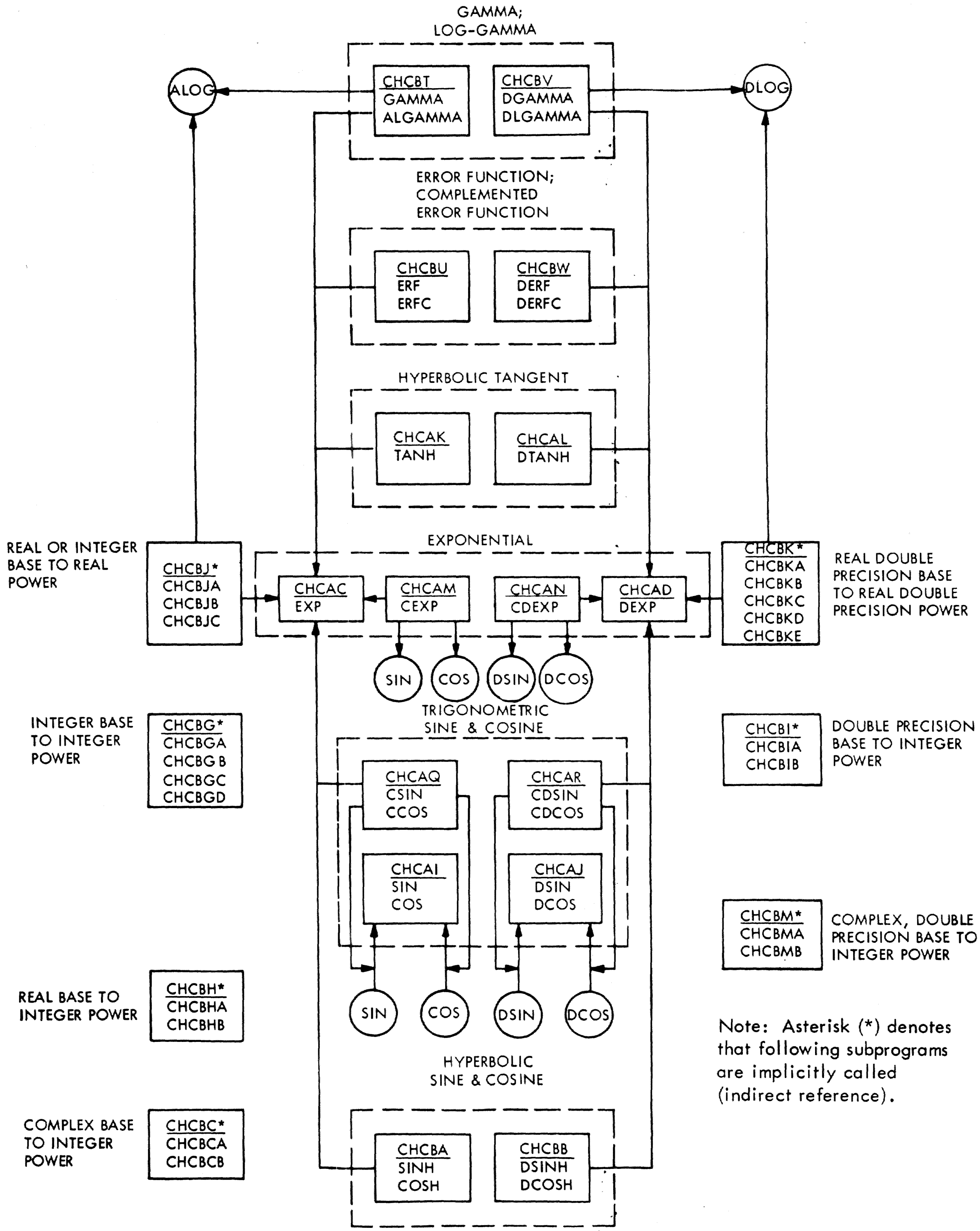


Figure 1. Functional Flow of Mathematical Library Subprograms (Page 2 of 2)

Table 1. I/O library calling relationships

Calling Routines \ Routines Called	CHCIA	CHCIB	CHCIC	CHCID	CHCIE	CHCIF	CHCIH	CHCII	CHCIJ	CHCIM	CHCIN	CHCIO	CHCIP	CHCIQ	CHCIR	CHCIS	CHCIT	CHCIU	CHCIV	CHCIW	CHCBD	Data Management	Supervisor
USER PROGRAM	X				X													X	X	X	X		
CHCIA		X	X	X		X														X			
CHCIB																				X		X	X
CHCIC		X																		X		X	X
CHCID			X							X	X					X	X			X			
CHCIE			X			X														X			
CHCIF			X				X	X	X	X	X	X	X	X	X	X	X			X			
CHCIH																							
CHCII																							
CHCIJ																							
CHCIM								X															
CHCIN																	X						
CHCIO																							
CHCIP																							
CHCIQ																							
CHCIR																							
CHCIS								X				X		X									
CHCIT							X		X						X								
CHCIU			X			X																	
CHCIV		X	X				X				X		X		X		X			X			
CHCIW		X	X																				X
CHCBD																							

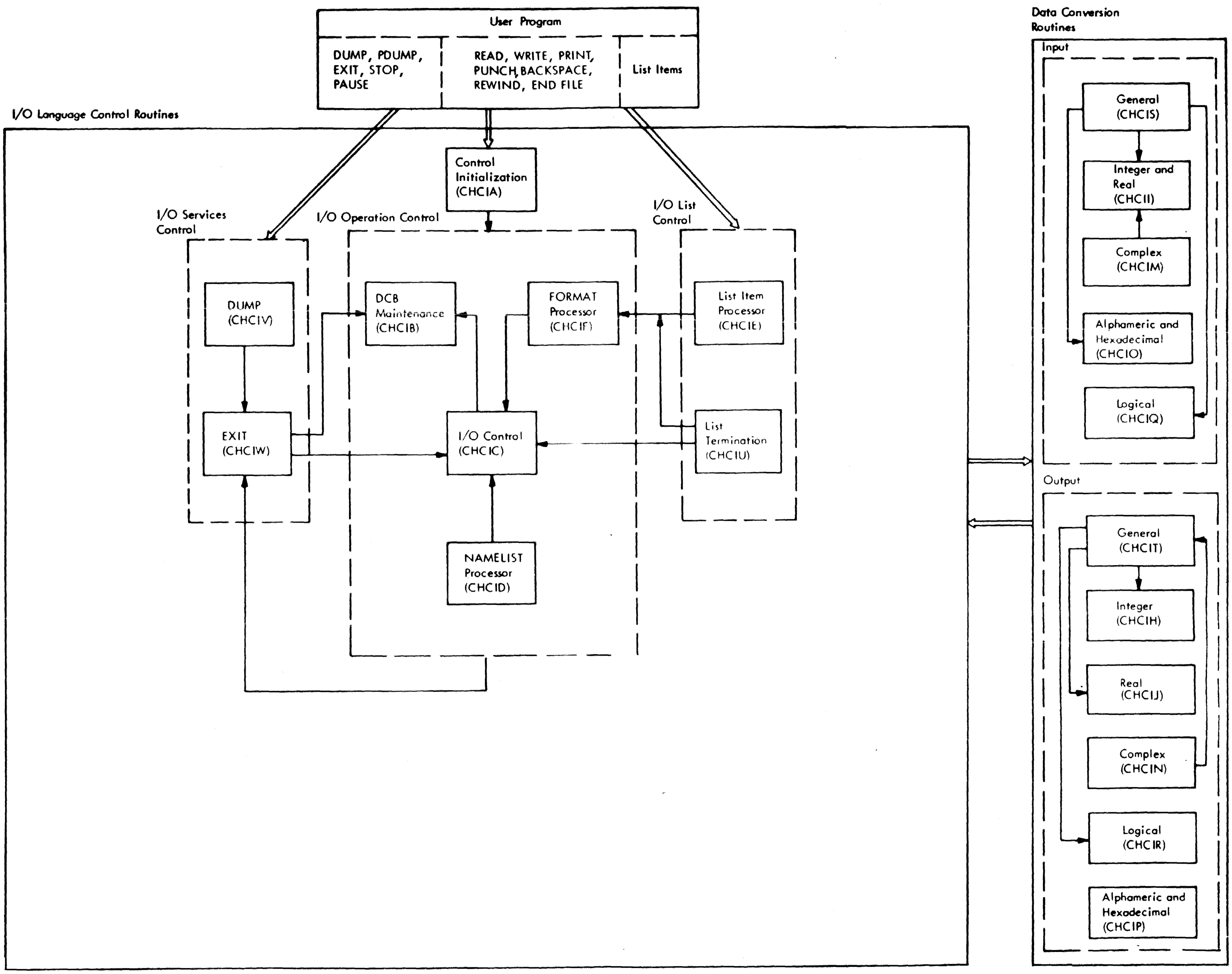


Figure 2. Functional Flow of I/O Library Routines

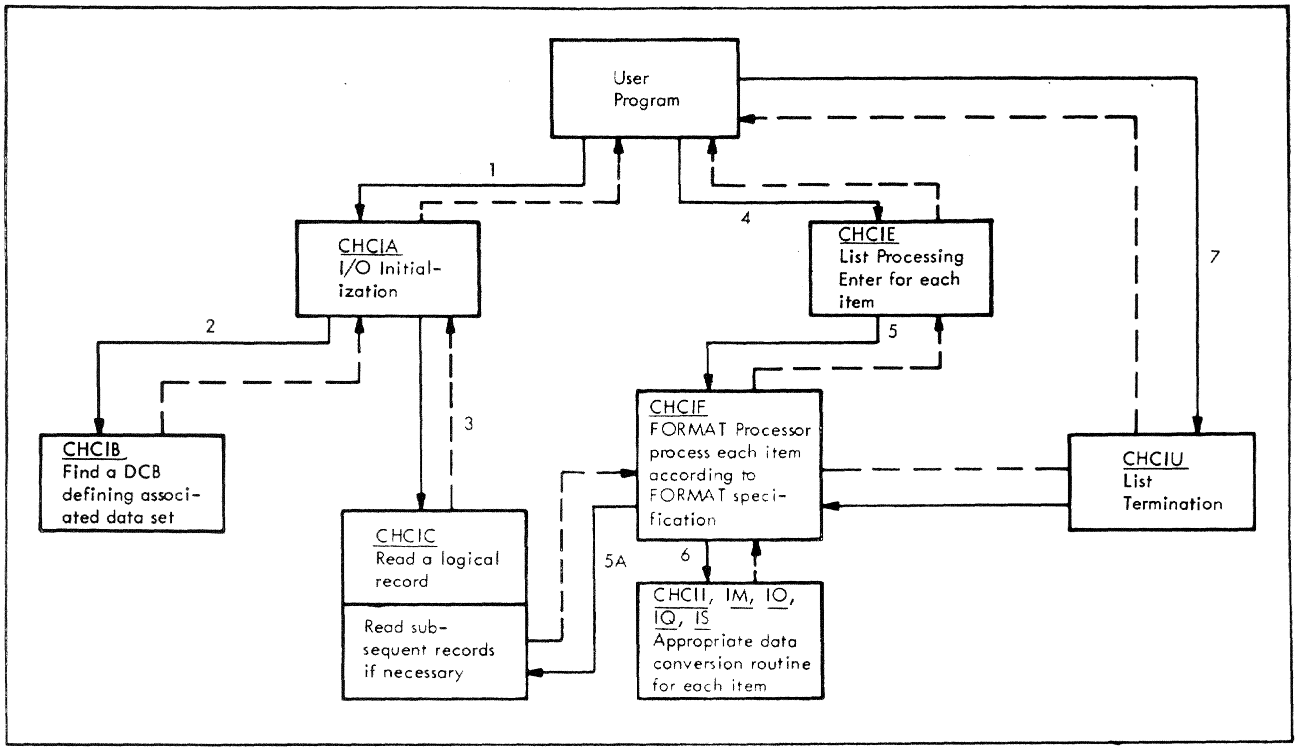


Figure 3. Formatted READ With List

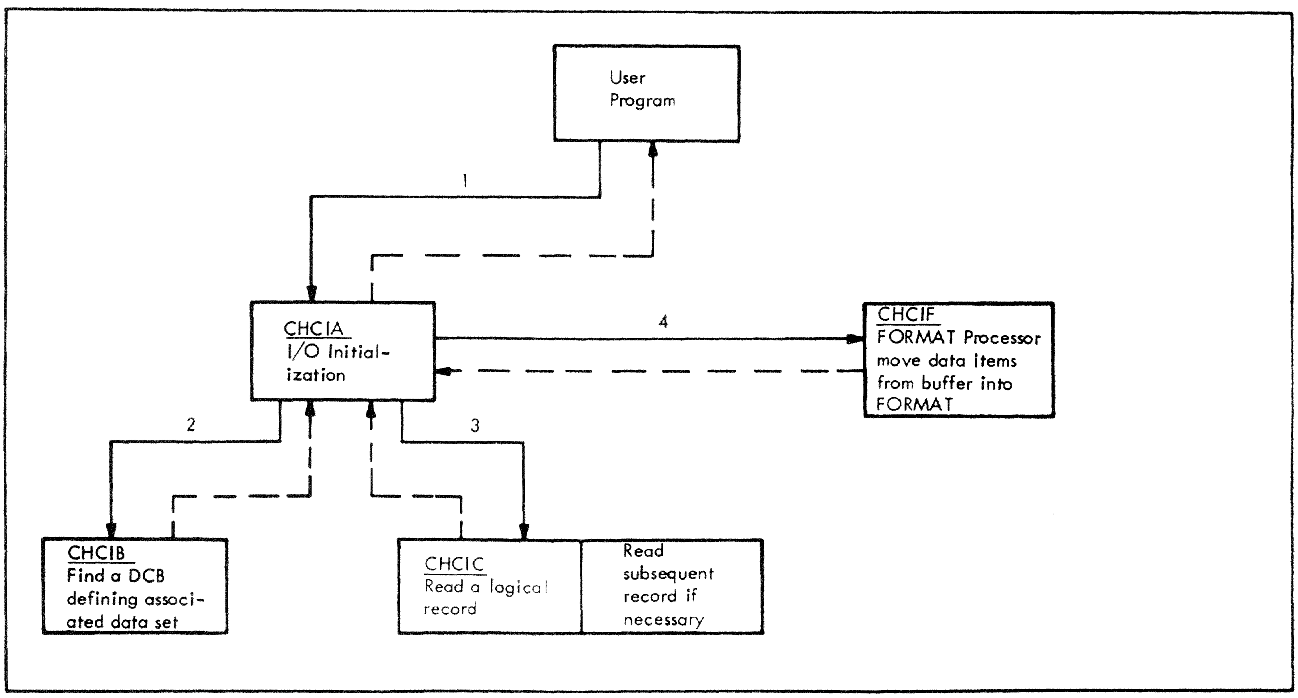


Figure 4. Formatted READ Without List

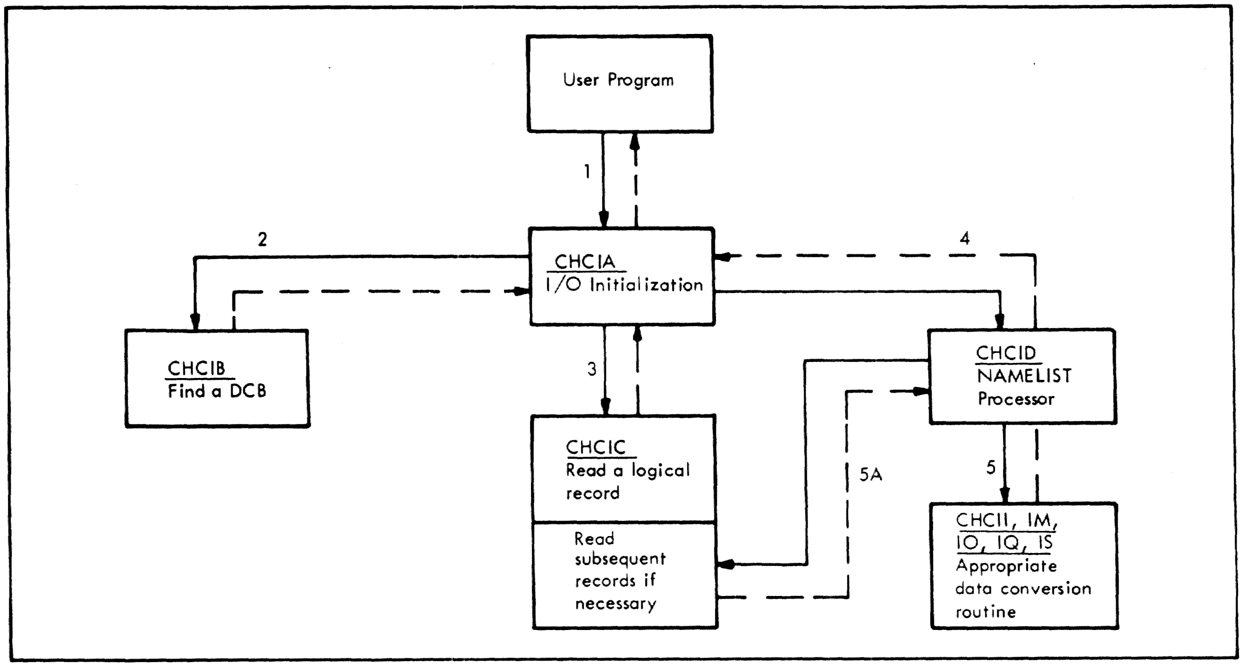


Figure 5. READ With NAMELIST

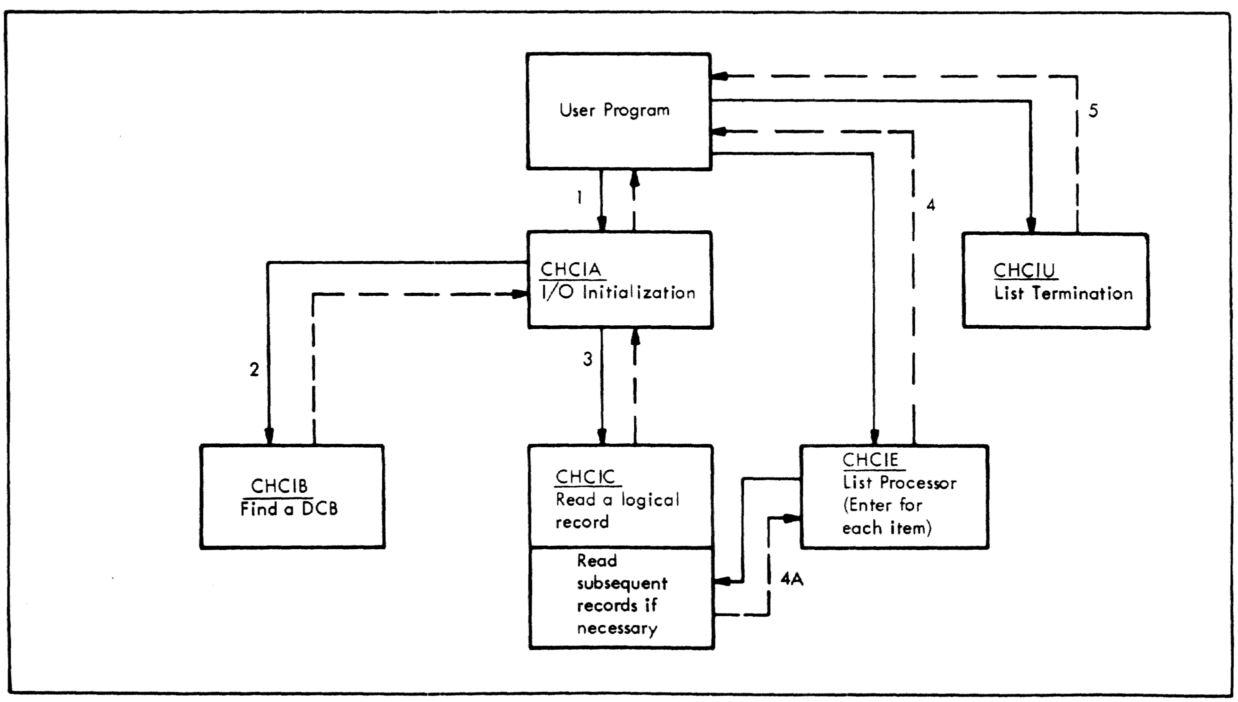


Figure 6. Unformatted READ With List

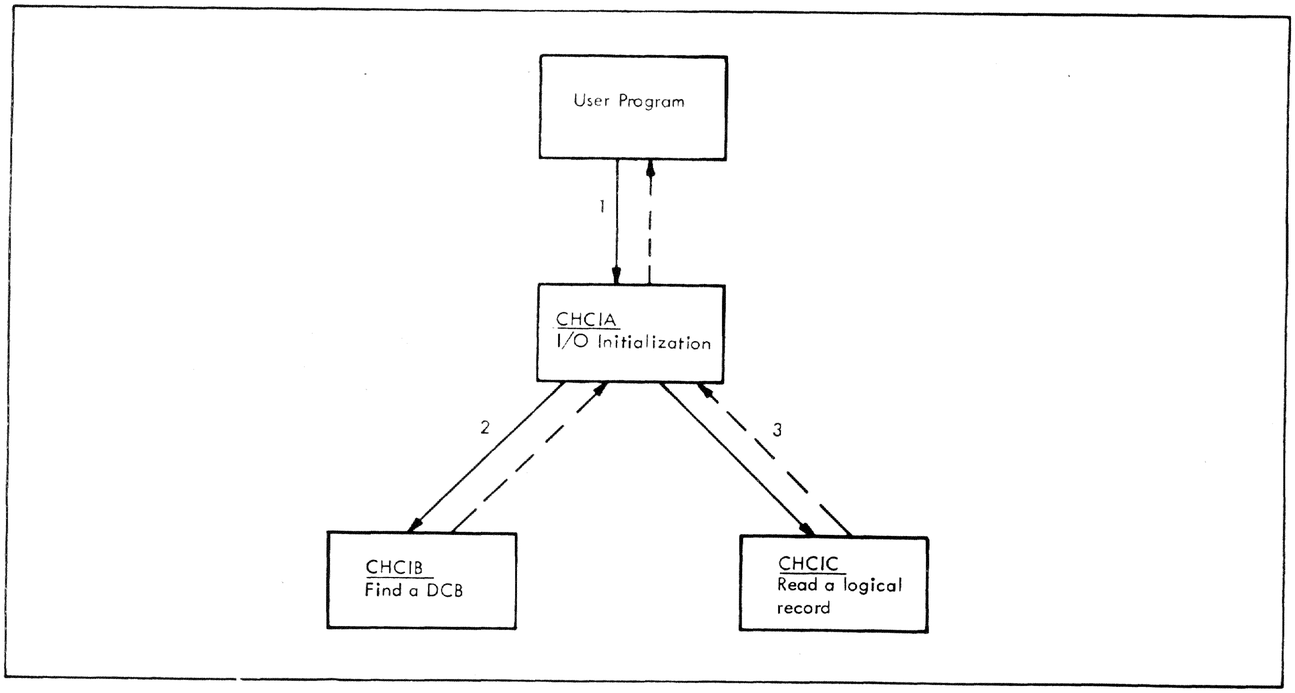


Figure 7. Unformatted READ Without List

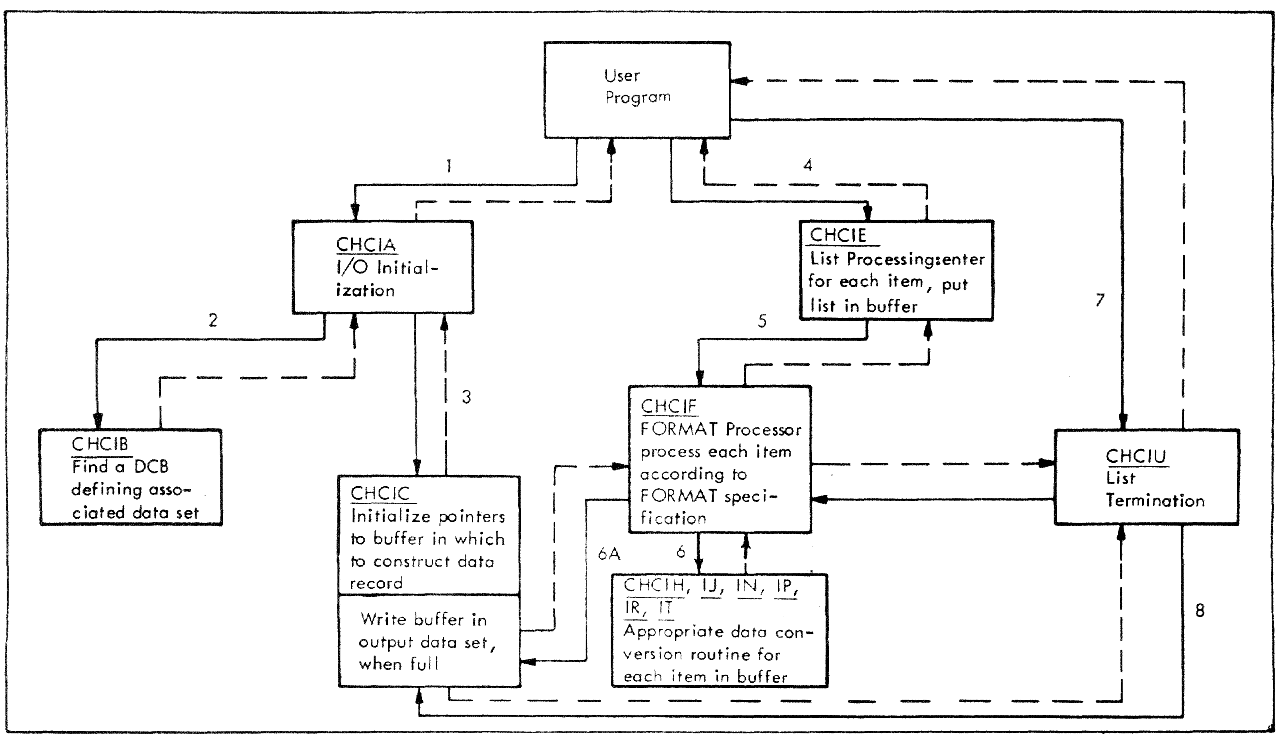


Figure 8. Formatted WRITE With List

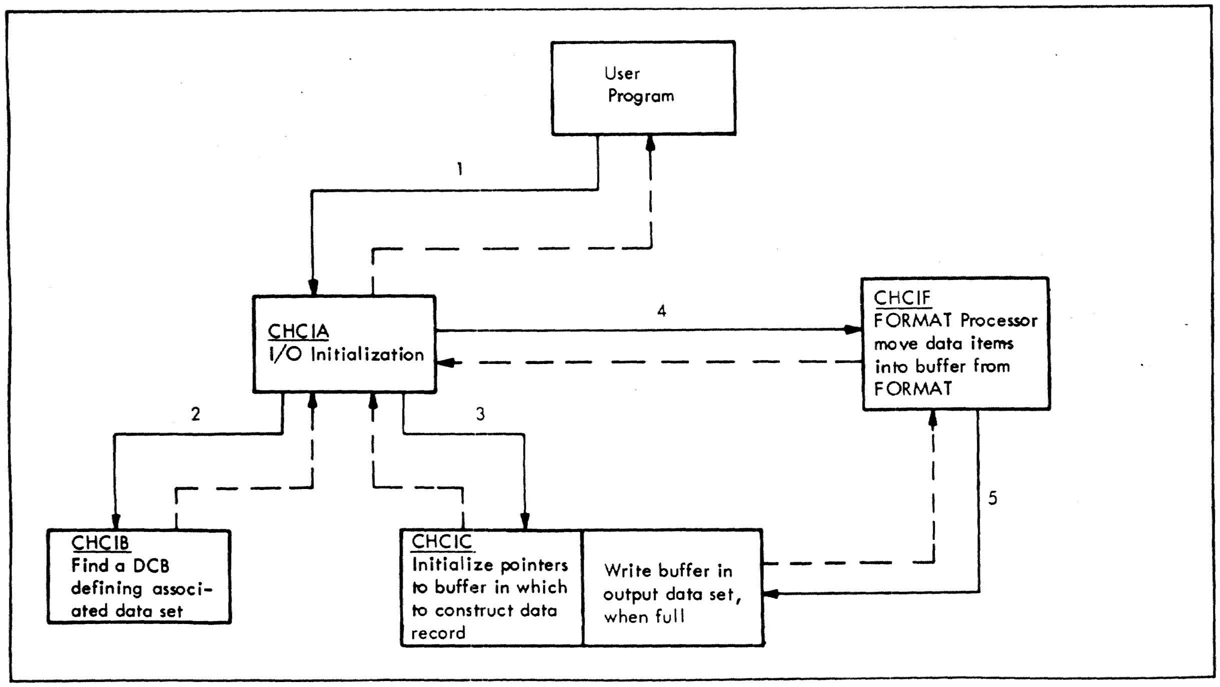


Figure 9. Formatted WRITE Without List

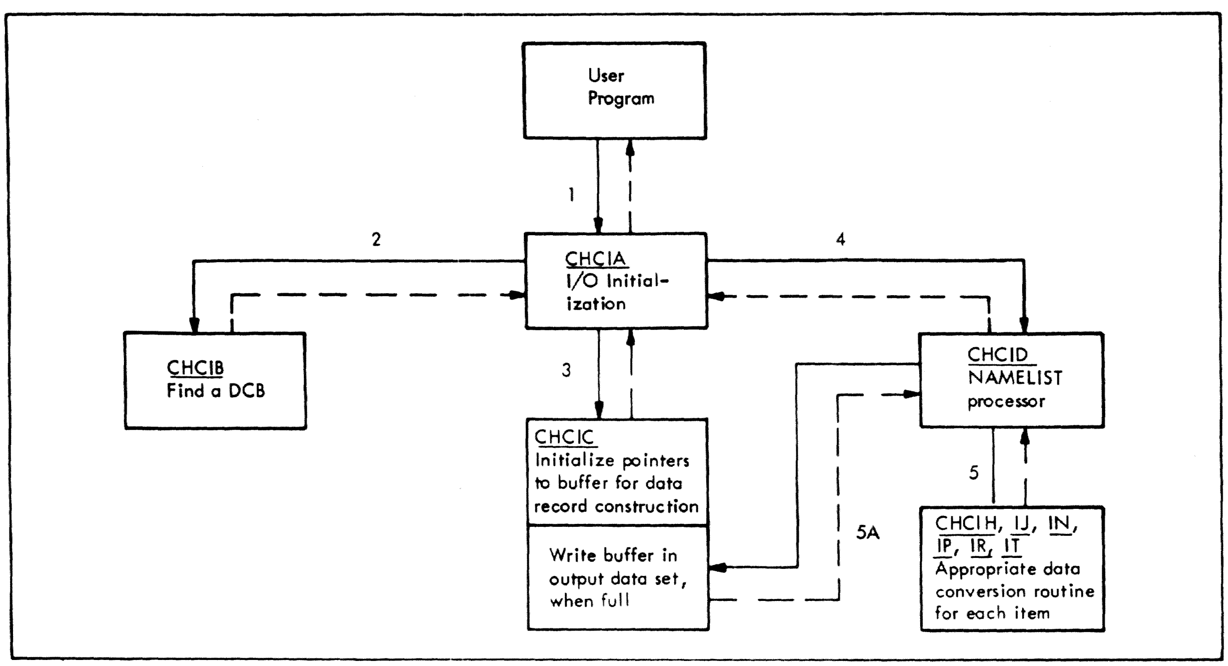


Figure 10. WRITE With NAMELIST

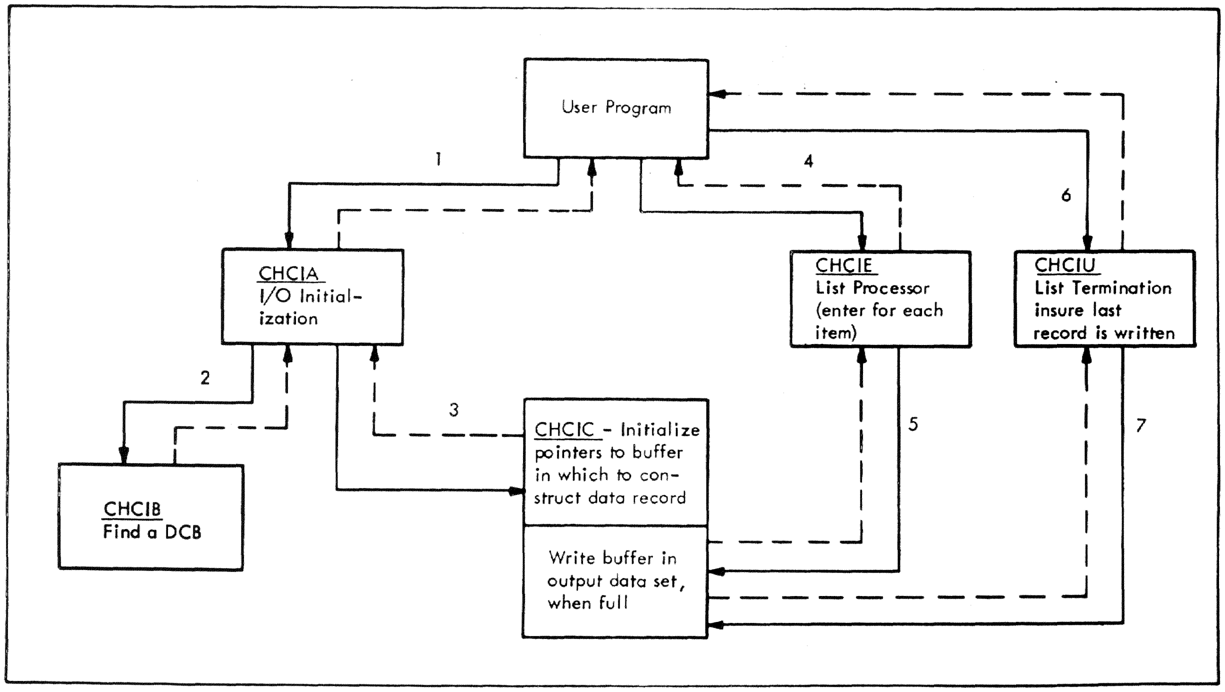


Figure 11. Unformatted WRITE With List

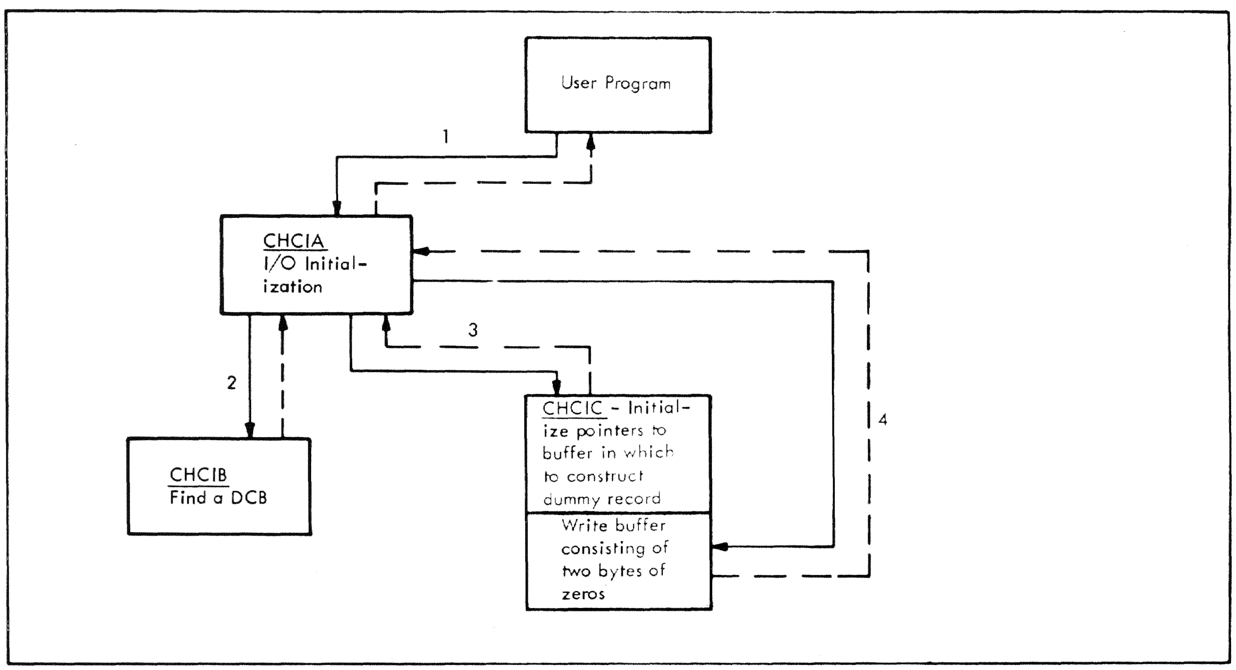


Figure 12. Unformatted WRITE Without List

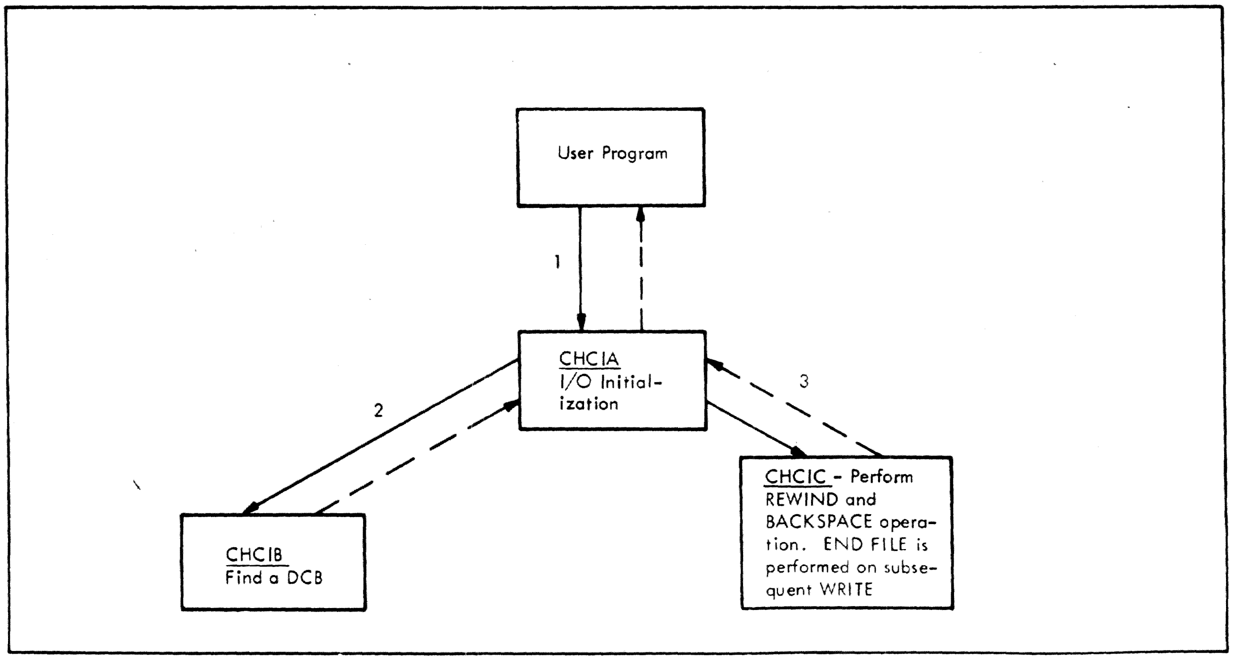


Figure 13. BACKSPACE, REWIND, and END FILE

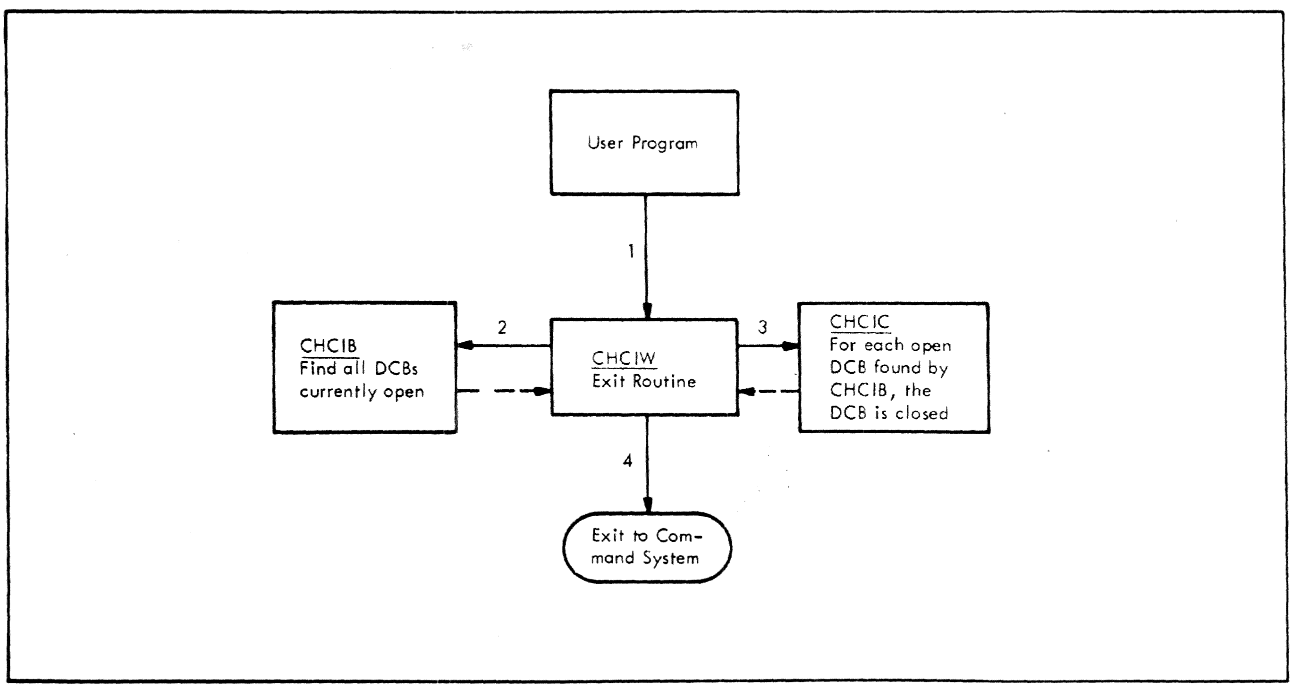


Figure 14. EXIT and STOP

This section gives the following information on each mathematical library subprogram, excluding the service subprograms, which are described in Section 4:

- Name
- Name of containing routine
- Function
- Attributes
- Entry
- Exit parameter (function value)
- Storage requirement
- Error check
- Accuracy figures (where applicable)

Since mathematical subprograms perform standardized computations, a detailed description of operation is given only for CHCBZ, the error-handling routine. The algorithms of direct-reference mathematical subprograms are described in FORTRAN IV Library Subprograms, GC28-2026.

GENERAL INFORMATION

Certain information is common to all mathematical routines and their subprograms. This information includes:

- Routine names
- Attributes
- Entry parameters
- Error action
- Exit parameters

Routine Names

All mathematical library routines have five-letter names beginning with 'CHCA' or 'CHCB'.

Attributes

All mathematical library routines use type-I linkage and are nonprivileged, reenterable, and closed.

Entry Parameters

Each mathematical subprogram receives one or two arguments from the calling program, in the form of a parameter list pointed to by register 1. The parameter list must contain the addresses of the arguments in the proper order:

- Directly referenced subprograms. The order is the same as that in the list of operands within the parentheses in the corresponding FORTRAN source statement. For example the source statement

  ANS=SIN(RADIAN)

  in FORTRAN coding corresponds to an assembler-language call containing one address in the parameter list -- the address of RADIAN. The FORTRAN statement

  ANS=ATAN2(X,Y)

  produces a linkage with a parameter list containing the addresses of X and Y, in that order. The assembler language programmer's linkage to ATAN2 must do the same.

- Indirectly referenced subprograms. The order for the exponentiation subprograms is: address of the number to be raised to a power and the address of the power itself.

Error Action

All mathematical subprograms that check for error call CHCBZ upon finding such error. CHCBZ then prints an error message and terminates execution. See the description of CHCBZ at the end of this section.

Exit Parameters

All subprogram results are returned in registers, as follows:

Integer	-	General register 0
Real	-	Floating register 0
Complex	-	Real part in floating register 0, complex part in floating register 2

SUBPROGRAM SUMMARIES

TABLES

Tables 2 and 3 give the following information concerning the mathematical subprograms:

FUNCTION: A brief description of the type of mathematical operation performed.

Table 2. Summary of directly referenced mathematical subprograms (page 1 of 3)

1	2	3	4			5	6	7		8	9	10					
Function	Entry Name	Definition	Argument(s)			Function Value Returned	Error Condition	Storage Estimates		Other Subprograms Required	Routine Name	Accuracy Figures					
												Argument Range	Sample E/U	relative		absolute	
			No.	Type	Range			Hex	Dec					$M(\epsilon)$	$\sigma(\epsilon)$	$M(E)$	$\sigma(E)$
COMMON AND NATURAL LOGARITHM	CDLOG	Ln (arg) or Log_e (arg) See Note 8	1	COMPLEX * 16	arg $\neq 0 + 0i$	COMPLEX * 16	Argument $= 0 + 0i$	1E8	488	CDABS, DLOG, DATAN2, DSQRT	CHCAP	The full range except (1 + 0i)	Note 1	2.72×10^{-16}	5.38×10^{-17}		
	CLOG	Ln (arg) or Log_e (arg) See Note 8	1	COMPLEX * 8	arg $\neq 0 + 0i$	COMPLEX * 8	Argument $= 0 + 0i$	1D0	464	CABS, ALOG, ATAN2, SQRT	CHCAO	The full range except (1 + 0i)	Note 1	7.15×10^{-7}	1.36×10^{-7}		
	DLOG	Ln (arg) or Log_e(arg)	1	REAL * 8	arg > 0	REAL * 8	Argument ≤ 0	21A	538		CHCAF	$0.5 \leq X \leq 1.5$	U			4.60×10^{-17}	2.09×10^{-17}
												$X < 0.5, X > 1.5$	E	3.32×10^{-16}	5.52×10^{-17}		
	DLOG10	Log_{10} (arg)	1	REAL * 8	arg > 0	REAL * 8	Argument ≤ 0	21A	538		CHCAF	$0.5 \leq X \leq 1.5$	U			2.73×10^{-17}	1.07×10^{-17}
												$X < 0.5, X > 1.5$	E	3.02×10^{-16}	6.65×10^{-17}		
	ALOG	Ln (arg) or Log_e (arg)	1	REAL * 4	arg > 0	REAL * 4	Argument ≤ 0	1D0	464		CHCAE	$0.5 \leq X \leq 1.5$	U			6.85×10^{-8}	2.33×10^{-8}
												$X < 0.5, X > 1.5$	E	8.32×10^{-7}	1.19×10^{-7}		
	ALOG10	Log_{10} (arg)	1	REAL * 4	arg > 0	REAL * 4	Argument ≤ 0	1D0	464		CHCAE	$0.5 \leq X \leq 1.5$	U			7.13×10^{-8}	2.26×10^{-8}
												$X < 0.5, X > 1.5$	E	1.05×10^{-6}	2.17×10^{-7}		
EXPONENTIAL	CDEXP	e^{arg}	1	COMPLEX * 16	real arg ≤ 174.673 $\lvert$imag arg$\rvert < 2^{50}\pi$	COMPLEX * 16	Real Argument > 174.673 $\lvert$Imaginary Argument$\rvert \geq 2^{50}\pi$	270	624	DEXP, DSIN, DCOS	CHCAN	$\lvert X_1\rvert \leq 1, \lvert X_2\rvert \leq \frac{\pi}{2}$	U	3.76×10^{-16}	1.10×10^{-16}		
												$\lvert X_1\rvert \leq 20, \lvert X_2\rvert \leq 20$	U	2.74×10^{-15}	9.64×10^{-16}		
	CEXP	e^{arg}	1	COMPLEX * 8	real arg 174.673 $\lvert$imag arg$\rvert < 2^{18}\pi$	COMPLEX * 8	Real Argument > 174.673 $\lvert$Imaginary Argument$\rvert \geq 2^{18}\pi$	250	592	EXP, SIN, COS	CHCAM	$\lvert X_1\rvert \leq 170, \lvert X_2\rvert \leq \frac{\pi}{2}$	U	9.93×10^{-7}	2.67×10^{-7}		
												$\lvert X_1\rvert \leq 170$, $\frac{\pi}{2} < \lvert X_2\rvert \leq 20$	U	1.07×10^{-6}	2.73×10^{-7}		
	DEXP	e^{arg}	1	REAL * 8	arg ≤ 174.673	REAL * 8	Argument > 174.673	2C0	704		CHCAD	$\lvert X\rvert \leq 1$	U	2.04×10^{-16}	5.43×10^{-17}		
												$1 < \lvert X\rvert \leq 20$	U	2.03×10^{-16}	4.87×10^{-17}		
												$20 < \lvert X\rvert \leq 170$	U	1.97×10^{-16}	4.98×10^{-17}		
	EXP	e^{arg}	1	REAL * 4	arg ≤ 174.673	REAL * 4	Argument ≥ 174.673	1A8	424	——	CHCAC	$\lvert X\rvert \leq 1$	U	4.65×10^{-7}	1.28×10^{-7}		
												$1 < \lvert X\rvert \leq 170$	U	4.42×10^{-7}	1.15×10^{-7}		
SQUARE ROOT	CDSQRT	$(arg)^{1/2}$ or $\sqrt{arg}$	1	COMPLEX * 16	Any	COMPLEX * 16	None	148	328	CDABS, DSQRT	CHCAT	The full range	Note 1	1.76×10^{-16}	4.06×10^{-17}		
	CSQRT	$(arg)^{1/2}$ or $\sqrt{arg}$	1	COMPLEX * 8	Any	COMPLEX * 8	None	138	312	CABS, SQRT	CHCAS	The full range	Note 1	7.00×10^{-7}	1.71×10^{-7}		
	DSQRT	$(arg)^{1/2}$ or $\sqrt{arg}$	1	REAL * 8	arg ≠ 0	REAL * 8	Negative Argument	160	352		CHCAB	The full range	E	1.06×10^{-16}	2.16×10^{-17}		
	SQRT	$(arg)^{1/2}$ or $\sqrt{arg}$	1	REAL * 4	arg ≠ 0	REAL * 4	Negative Argument	158	344		CHCAA	The full range	E	4.45×10^{-7}	8.43×10^{-8}		
ARCSINE AND ARCCOSINE	DARSIN	arcsine (arg)	1	REAL * 8	$\lvert$arg$\rvert \leq 1$	REAL * 8	$\lvert$Argument$\rvert > 1$	288	648	DSQRT	CHCAX	$-1 \leq X \leq +1$	U	2.04×10^{-16}	5.15×10^{-17}		
	DARCOS	arccosine (arg)	1	REAL * 8	$\lvert$arg$\rvert \leq 1$	REAL * 8	$\lvert$Argument$\rvert > 1$	288	648	DSQRT	CHCAX	$-1 \leq X \leq +1$	U	2.07×10^{-16}	7.05×10^{-17}		
	ARSIN	arcsine (arg)	1	REAL * 4	$\lvert$arg$\rvert \leq 1$	REAL * 4	$\lvert$Argument$\rvert > 1$	1F0	496	SQRT	CHCAW	$-1 \leq X \leq +1$	U	9.34×10^{-7}	2.06×10^{-7}		
	ARCOS	arccosine (arg)	1	REAL * 4	$\lvert$arg$\rvert \leq 1$	REAL * 4	$\lvert$Argument$\rvert > 1$	1F0	496	SQRT	CHCAW	$-1 \leq X \leq +1$	U	8.85×10^{-7}	3.19×10^{-7}		
ARCTANGENT	DATAN	arctan (arg)	1	REAL * 8	Any	REAL * 8	None	288	648		CHCBR	The full range	Note 7	2.18×10^{-16}	7.04×10^{-17}		
	DATAN2	arctan (arg_1/arg_2)	2	REAL * 8	arg ≠ 0	REAL * 8	$X_1 = X_2 = 0$	288	648		CHCBR	The full range	Note 7	2.18×10^{-16}	7.04×10^{-17}		
	ATAN	arctan (arg)	1	REAL * 4	Any	REAL * 4	None	1E8	488		CHCBQ	The full range	Note 7	1.01×10^{-6}	4.68×10^{-7}		
	ATAN2	arctan (arg_1/arg_2)	2	REAL * 4	arg ≠ 0	REAL * 4	$X_1 = X_2 = 0$	1E8	488		CHCBQ	The full range	Note 7	1.01×10^{-6}	4.68×10^{-7}		
TRIGONOMETRIC SINE & COSINE	CDSIN	sin (arg), arg in radians	1	COMPLEX * 16	$\lvert$real arg$\rvert < 2^{50}\pi$ $\lvert$imag arg$\rvert \leq 174.673$	COMPLEX * 16	$\lvert$Real Argument$\rvert \geq 2^{50}\pi$ $\lvert$Imaginary Argument$\rvert > 174.673$	340	832	DSIN, DCOS, DEXP	CHCAR	$\lvert X_1\rvert \leq 10, \lvert X_2\rvert \leq 1$	U	2.35×10^{-15} See Note 4	2.25×10^{-16}		
	CDCOS	cos (arg), arg in radians	1	COMPLEX * 16	$\lvert$real arg$\rvert < 2^{50}\pi$ $\lvert$imag arg$\rvert \leq 174.673$	COMPLEX * 16	$\lvert$Real Argument$\rvert \geq 2^{50}\pi$ $\lvert$Imaginary Argument$\rvert > 174.673$	340	832	DSIN, DCOS, DEXP	CHCAR	$\lvert X_1\rvert \leq 10, \lvert X_2\rvert \leq 1$	U	3.98×10^{-15} See Note 3	2.50×10^{-16}		
	CSIN	sin (arg), arg in radians	1	COMPLEX * 8	$\lvert$real arg$\rvert < 2^{18}\pi$ $\lvert$imag arg$\rvert \leq 174.673$	COMPLEX * 8	$\lvert$Real Argument$\rvert \geq 2^{18}\pi$ $\lvert$Imaginary Argument$\rvert > 174.673$	2F8	760	SIN, COS, EXP	CHCAQ	$\lvert X_1\rvert \leq 10, \lvert X_2\rvert \leq 1$	U	1.92×10^{-6} See Note 6	7.38×10^{-7}		

Table 2. Summary of directly referenced mathematical subprograms (page 2 of 3)

1	2	3	4			5	6	7		8	9	10					
Function	Entry Name	Definition	Argument(s)			Function Value Returned	Error Condition	Storage Estimates		Other Subprograms Required	Routine Name	Accuracy Figures					
			No.	Type	Range			Hex	Dec			Argument Range	Sample E/U	Relative		Absolute	
														$M(\epsilon)$	$\sigma(\epsilon)$	$M(E)$	$\sigma(E)$
TRIGONOMETRIC SINE & COSINE (Continued)	CCOS	cos (arg), arg in radians	1	COMPLEX * 8	$\lvert \text{real arg} \rvert < 2^{18}\pi$ $\lvert \text{imag arg} \rvert \leq 174.673$	COMPLEX * 8	$\lvert \text{Real Argument} \rvert \geq 2^{18}\pi$ $\lvert \text{Imaginary Argument} \rvert > 174.673$	2F8	760	SIN, COS, EXP	CHCAQ	$\lvert X_1 \rvert \leq 10, \lvert X_2 \rvert \leq 1$	U	2.50×10^{-6} See Note 2	7.66×10^{-7}		
	DSIN	sin (arg), arg in radians	1	REAL * 8	$\lvert arg \rvert < 2^{50}\pi$	REAL * 8	$\lvert \text{Argument} \rvert \geq 2^{50}\pi$	2B8	696		CHCAJ	$\lvert X \rvert \leq \frac{\pi}{2}$	U	3.60×10^{-16}	4.82×10^{-7}	7.74×10^{-17}	1.98×10^{-17}
												$\frac{\pi}{2} < \lvert X \rvert \leq 10$	U			1.64×10^{-16}	6.49×10^{-17}
												$10 < \lvert X \rvert \leq 100$	U			2.68×10^{-15}	1.03×10^{-15}
	DCOS	cos (arg), arg in radians	1	REAL * 8	$\lvert arg \rvert < 2^{50}\pi$	REAL * 8	$\lvert \text{Argument} \rvert \geq 2^{50}\pi$	2B8	696		CHCAJ	$0 \leq X \leq \pi$	U			1.79×10^{-16}	6.53×10^{-17}
												$-10 \leq X < 0$ $\pi < X \leq 10$	U			1.75×10^{-16}	5.93×10^{-17}
												$10 < \lvert X \rvert \leq 100$	U			2.64×10^{-15}	1.01×10^{-15}
	SIN	sin (arg), arg in radians	1	REAL * 4	$\lvert arg \rvert < 2^{18}\pi$	REAL * 4	$\lvert \text{Argument} \rvert \geq 2^{18}\pi$	1F8	504		CHCAI	$\lvert X \rvert \leq \frac{\pi}{2}$	U	1.32×10^{-6}	1.82×10^{-7}	1.18×10^{-7}	4.55×10^{-8}
												$\frac{\pi}{2} < \lvert X \rvert \leq 10$	U			1.15×10^{-7}	4.64×10^{-8}
												$10 < \lvert X \rvert \leq 100$	U			1.28×10^{-7}	4.52×10^{-8}
	COS	cos (arg), arg in radians	1	REAL * 4	$\lvert arg \rvert < 2^{18}\pi$	REAL * 4	$\lvert \text{Argument} \rvert \geq 2^{18}\pi$	1F8	504		CHCAI	$0 \leq X \leq \pi$	U			1.19×10^{-7}	4.60×10^{-8}
												$-10 \leq X < 0$ $\pi < X \leq 10$	U			1.28×10^{-7}	4.55×10^{-8}
												$10 < \lvert X \rvert \leq 100$	U			1.14×10^{-7}	4.60×10^{-8}
TRIGONOMETRIC TANGENT	DTAN	tan (arg), arg in radians	1	REAL * 8	$\lvert arg \rvert < 2^{50}\pi$	REAL * 8	$\lvert \text{Argument} \rvert \geq 2^{50}\pi$ Argument too close to a Singularity (i.e., too close to an odd multiple of $\pi/2$)	2F8	760		CHCAZ	$\lvert X \rvert \leq \frac{\pi}{4}$	U	3.41×10^{-16}	6.27×10^{-17}		
												$\frac{\pi}{4} < \lvert X \rvert \leq \frac{\pi}{2}$	U	1.43×10^{-12} See Note 5	2.95×10^{-14}		
												$\frac{\pi}{2} < \lvert X \rvert \leq 10$	U	2.78×10^{-13} See Note 5	7.23×10^{-15}		
												$10 < \lvert X \rvert \leq 100$	U	3.79×10^{-12} See Note 5	9.50×10^{-14}		
	DCOTAN	cotan (arg), arg in radians	1	REAL * 8	$\lvert arg \rvert < 2^{50}\pi$	REAL * 8	$\lvert \text{Argument} \rvert \geq 2^{50}\pi$ Argument too close to a Singularity (i.e., too close to a multiple of π)	2F8	760		CHCAZ	$\lvert X \rvert \leq \frac{\pi}{4}$	U	2.46×10^{-16} See Note 5	8.79×10^{-17}		
												$\frac{\pi}{4} < \lvert X \rvert \leq \frac{\pi}{2}$	U	2.78×10^{-13} See Note 5	8.61×10^{-15}		
												$\frac{\pi}{2} < \lvert X \rvert \leq 10$	U	5.40×10^{-13} See Note 5	1.13×10^{-14}		
												$10 < \lvert X \rvert \leq 100$	U	8.61×10^{-13} See Note 5	4.61×10^{-14}		
	TAN	tan (arg), arg in radians	1	REAL * 4	$\lvert arg \rvert < 2^{18}\pi$	REAL * 4	$\lvert \text{Argument} \rvert \geq 2^{18}\pi$ Argument too close to a Singularity (i.e., too close to an odd multiple of $\pi/2$)	288	648		CHCAY	$\lvert X \rvert \leq \frac{\pi}{4}$	U	1.71×10^{-6}	2.64×10^{-7}		
												$\frac{\pi}{4} < \lvert X \rvert \leq \frac{\pi}{2}$	U	1.05×10^{-6} See Note 5	3.59×10^{-7}		
												$\frac{\pi}{2} < \lvert X \rvert \leq 10$	U	6.49×10^{-6} See Note 5	3.38×10^{-7}		
												$10 < \lvert X \rvert \leq 100$	U	1.57×10^{-6} See Note 5	3.07×10^{-7}		
	COTAN	cotan (arg), arg in radians	1	REAL * 4	$\lvert arg \rvert < 2^{18}\pi$	REAL * 4	$\lvert \text{Argument} \rvert \geq 2^{18}\pi$ Argument too close to a Singularity (i.e., too close to a multiple of π)		648		CHCAY	$\lvert X \rvert \leq \frac{\pi}{4}$	U	1.07×10^{-6}	3.58×10^{-7}		
												$\frac{\pi}{4} < \lvert X \rvert \leq \frac{\pi}{2}$	U	1.40×10^{-6} See Note 5	2.56×10^{-7}		
												$\frac{\pi}{2} < \lvert X \rvert \leq 10$	U	1.30×10^{-6} See Note 5	3.11×10^{-7}		
												$10 < \lvert X \rvert \leq 100$	U	1.49×10^{-6} See Note 5	3.15×10^{-7}		
HYPERBOLIC SINE & COSINE	DSINH	sinh (arg)	1	REAL * 8	$\lvert arg \rvert < 175.366$	REAL * 8	$\lvert \text{Argument} \rvert \geq 174.673$	250	592	DEXP	CHCBB	$\lvert X \rvert \leq 0.88137$	U	2.06×10^{-16}	3.74×10^{-17}		
												$0.88137 < \lvert X \rvert \leq 5$	U	3.80×10^{-16}	9.21×10^{-17}		
	DCOSH	cosh (arg)	1	REAL * 8	$\lvert arg \rvert < 175.366$	REAL * 8	$\lvert \text{Argument} \rvert \geq 174.673$	250	592	DEXP	CHCBB	$-5 \leq X \leq +5$	U	3.63×10^{-16}	9.05×10^{-17}		
	SINH	sinh (arg)	1	REAL * 4	$\lvert arg \rvert < 175.366$	REAL * 4	$\lvert \text{Argument} \rvert \geq 174.673$	1F8	504	EXP	CHCBA	$-5 \leq X \leq +5$	U	1.26×10^{-6}	2.17×10^{-7}		
	COSH	cosh (arg)	1	REAL * 4	$\lvert arg \rvert < 175.366$	REAL * 4	$\lvert \text{Argument} \rvert \geq 174.673$	1F8	504	EXP	CHCBA	$-5 \leq X \leq +5$	U	1.27×10^{-6}	2.63×10^{-7}		

Table 2. Summary of directly referenced mathematical subprograms (page 3 of 3)

Function (1)	Entry Name (2)	Definition (3)	Argument(s) (4): No.	Argument(s) (4): Type	Argument(s) (4): Range	Function Value Returned (5)	Error Condition (6)	Storage Estimates (7): Hex	Storage Estimates (7): Dec	Other Subprograms Required (8)	Routine Name (9)	Accuracy Figures (10): Argument Range	Sample E/U	relative: $M(\epsilon)$	relative: $\sigma(\epsilon)$	absolute: $M(E)$	absolute: $\sigma(E)$
HYPERBOLIC TANGENT	DTANH	tanh (arg)	1	REAL * 8	Any	REAL * 8		130	304	DEXP	CHCAL	$\|X\| \leqq 0.54931$	U	1.91×10^{-16}	3.86×10^{-17}		
												$0.54931 < \|X\| \leqq 5$	U	1.54×10^{-16}	1.87×10^{-17}		
	TANH	tanh (arg)	1	REAL * 4	Any	REAL * 4		164	356	EXP	CHCAK	$\|X\| \leqq 0.7$	U	8.48×10^{-7}	1.48×10^{-7}		
												$0.7 < \|X\| \leqq 5$	U	2.44×10^{-7}	4.23×10^{-8}		
ABSOLUTE VALUE	CDABS	$\|arg\|$	1	COMPLEX * 16	Any See Note 9	REAL * 8		C8	200	DSQRT	CHCAV	The full range	Note 1	2.03×10^{-16}	4.83×10^{-17}		
	CABS	$\|arg\|$	1	COMPLEX * 8	Any See Note 9	REAL * 4		C0	192	SQRT	CHCAU	The full range	Note 1	9.15×10^{-7}	2.00×10^{-7}		
ERROR FUNCTION	ERF	$\frac{2}{\sqrt{\pi}} \int_0^x e^{-u^2} du$	1	REAL * 4	Any	REAL * 4		208	520	EXP	CHCBU	$\|X\| \leqq 1$	U	8.16×10^{-7}	1.10×10^{-7}		
												$1 < \|X\| \leqq 2.04$	U	1.13×10^{-7}	3.70×10^{-8}		
												$2.04 < \|X\| \leqq 3.9192$	U	5.95×10^{-8}	3.41×10^{-8}		
	DERF	$\frac{2}{\sqrt{\pi}} \int_0^x e^{-u^2} du$	1	Real * 8	Any	Real * 8		328	808	DEXP	CHCBW	$\|X\| \leqq 1$	U	1.89×10^{-16}	2.60×10^{-17}		
												$1 < \|X\| \leqq 2.04$	U	2.87×10^{-17}	9.84×10^{-18}		
												$2.04 < \|X\| < 6.092$	U	1.39×10^{-17}	8.02×10^{-18}		
COMPLEMENTED ERROR FUNCTION	ERFC	1 - erf (x) or $\frac{2}{\sqrt{\pi}} \int_x^\infty e^{-u^2} du$	1	REAL * 4	Any	REAL * 4		208	520	EXP	CHCBU	$-3.8 < X < 0$	U	9.10×10^{-7}	2.96×10^{-7}		
												$0 \leqq X \leqq 1$	U	7.42×10^{-7}	1.27×10^{-7}		
												$1 < X \leqq 2.04$	U	1.54×10^{-6}	3.78×10^{-7}		
												$2.04 < X < 4$	U	2.28×10^{-6}	3.70×10^{-7}		
												$4 \leqq X \leqq 13.3$	U	1.55×10^{-5}	8.57×10^{-6}		
	DERFC	1 - erf (x) or $\frac{2}{\sqrt{\pi}} \int_x^\infty e^{-u^2} du$	1	Real * 8	Any	Real * 8		328	808	DEXP	CHCBW	$-6 < X < 0$	U	2.08×10^{-16}	6.52×10^{-17}		
												$0 \leqq X \leqq 1$	U	1.40×10^{-16}	2.59×10^{-17}		
												$1 < X \leqq 2.04$	U	4.11×10^{-16}	8.86×10^{-17}		
												$2.04 < X < 4$	U	3.26×10^{-16}	8.65×10^{-17}		
												$4 \leqq X < 13.3$	U	3.51×10^{-15}	1.96×10^{-15}		
GAMMA (Γ)	GAMMA	$\int_0^\infty u^{x-1} e^{-u} du$	1	REAL * 4	$X > 2^{-252}$ $X < 57.5744$	REAL * 4	Real Argument > 57.5744 Real Argument < 2^{-252}	350	848	EXP, ALOG	CHCBT	$0 < X < 1$	U	9.86×10^{-7}	3.66×10^{-7}		
												$1 \leqq X \leqq 2$	U	1.13×10^{-7}	3.22×10^{-8}		
												$2 < X \leqq 4$	U	9.47×10^{-7}	3.79×10^{-7}		
												$4 < X < 8$	U	2.26×10^{-6}	8.32×10^{-7}		
												$8 \leqq X < 16$	U	2.20×10^{-5}	7.61×10^{-6}		
												$16 \leqq X < 57$	U	4.62×10^{-5}	1.51×10^{-5}		
	DGAMMA	$\int_0^\infty u^{x-1} e^{-u} du$	1	REAL * 8	$X > 2^{-252}$ $X < 57.5744$	REAL * 8	Real Argument > 57.5744 Real Argument < 2^{-252}	420	1056	DEXP, DLOG	CHCBV	$0 < X < 1$	U	2.14×10^{-16}	7.84×10^{-17}		
												$1 \leqq X \leqq 2$	U	2.52×10^{-17}	6.07×10^{-18}		
												$2 < X < 4$	U	2.21×10^{-16}	8.49×10^{-17}		
												$4 \leq X < 8$	U	5.05×10^{-16}	1.90×10^{-16}		
												$8 \leqq X < 16$	U	6.02×10^{-15}	1.78×10^{-15}		
												$16 \leqq X < 57$	U	1.16×10^{-14}	4.11×10^{-15}		
LOG - GAMMA	ALGAMA	$\log_e \int_0^\infty u^{x-1} e^{-u} du$	1	REAL * 4	$X > 0$ $X < 4.2913 \times 10^{73}$	REAL * 4	Real Argument > 4.2937×10^{73} Real Argument < 0	350	848	EXP, ALOG	CHCBT	$0 < X < 0.5$	U	1.16×10^{-6}	3.54×10^{-7}		
												$0.5 \leqq X < 3$	U			9.43×10^{-7}	3.42×10^{-7}
												$3 \leqq X < 8$	U	1.25×10^{-6}	3.04×10^{-7}		
												$8 \leqq X < 16$	U	1.18×10^{-6}	3.80×10^{-7}		
												$16 \leqq X < 500$	U	9.85×10^{-7}	1.90×10^{-7}		
	DLGAMA	$\log_e \int_0^\infty u^{x-1} e^{-u} du$	1	REAL * 8	$X > 0$ $X < 4.2913 \times 10^{73}$	REAL * 8	Real Argument > 4.2937×10^{73} Real Argument < 0	420	1056	DEXP, DLOG	CHCBV	$0 < X \leqq 0.5$	U	2.77×10^{-16}	9.75×10^{-17}		
												$0.5 < X < 3$	U			2.24×10^{-16}	7.77×10^{-17}
												$3 \leqq X < 8$	U	2.89×10^{-16}	8.80×10^{-17}		
												$8 \leqq X < 16$	U	2.86×10^{-16}	8.92×10^{-17}		
												$16 \leqq X < 500$	U	1.99×10^{-16}	3.93×10^{-17}		

Notes

1. The distribution of sample arguments upon which these statistics are based is exponential radially and is uniform around the origin.
2. The maximum relative error cited for the CCOS function is based upon a set of 2000 random arguments within the range. In the immediate proximity of the points $(n + 1/2)\pi + 0i$ (where $n = 0, \pm 1, \pm 2, \ldots,$) the relative error can be quite high, although the absolute error is small.
3. The maximum relative error cited for the CDCOS function is based upon a set of 1500 random arguments within the range. In the immediate proximity of the points $(n + 1/2)\pi + 0i$ (where $n = 0, \pm 1, \pm 2, \ldots,$) the relative error can be quite high although the absolute error is small.
4. The maximum relative error cited for the CDSIN function is based upon a set of 1500 random arguments within the range. In the immediate proximity of the points $n\pi + 0i$ (where $n = \pm 1, \pm 2, \ldots,$) the relative error can be quite high although the absolute error is small.
5. The figures cited as the maximum relative errors are those encountered in a sample of 2500 random arguments within the respective ranges. See the appropriate section in Appendix F for a description of the behavior of errors when the argument is near a singularity or a zero of the function.
6. The maximum relative error cited for the CSIN function is based upon a set of 2000 random arguments within the range. In the immediate proximity of the points $n\pi + 0i$ (where $n = \pm 1, \pm 2, \ldots,$) the relative error can be quite high although the absolute error is small.
7. The sample arguments were tangents of numbers uniformly distributed between $-\pi/2$ and $+\pi/2$.
8. The answer given is the principal value, i.e., the one whose imaginary part lies between $-\pi$ and $+\pi$.
9. Floating-point overflow can occur.

Table 3. Summary of indirectly referenced mathematical subprograms

	2	3	4		5	6	7		8	10
Function	Entry Name	Definition	Argument(s)		Function Value Returned	Error Condition	Storage Estimates		Other Subprograms Required	Routine Name
			No.	Type			Hex	Dec		
RAISE AN INTEGER BASE TO AN INTEGER POWER	CHCBGA	y = i * * i	2	i = INTEGER * 4	INTEGER * 4	Base is zero	1B4	436	—	CHCBG
	CHCBGB	y = j * * j	2	j = INTEGER * 2	INTEGER * 2	Base is zero and exponent is zero or negative	1B4	436	—	CHCBG
	CHCBGC	y = j * * i	2	j = INTEGER * 2 i = INTEGER * 4	INTEGER * 4	Base is zero and exponent is zero or negative	1B4	436	—	CHCBG
	CHCBGD	y = i * * j	2	i = INTEGER * 4 j = INTEGER * 2	INTEGER * 4	Base is zero and exponent is zero or negative	1B4	436	—	CHCBG
RAISE A REAL BASE TO AN INTEGER POWER	CHCBHA	y = a * * i	2	a = REAL * 4 i = INTEGER * 4	REAL * 4	Base is zero and exponent is zero or negative	144	324	—	CHCBH
	CHCBHB	y = a * * j	2	a = REAL * 4 j = INTEGER * 2	REAL * 4	Base is zero and exponent is zero or negative	144	324	—	CHCBH
RAISE A DOUBLE PRECISION BASE TO AN INTEGER POWER	CHCBIA	y = a * * i	2	a = REAL * 8 = INTEGER * 4	REAL * 8	Base is zero and exponent is zero or negative	14C	332	—	CHCBI
	CHCBIB	y = a * * j	2	a = REAL * 8 j = INTEGER * 2	REAL * 8	Base is zero and exponent is zero or negative	14C	332	—	CHCBI
RAISE A REAL BASE TO A REAL POWER	CHCBJA	y = a * * b	2	a = REAL * 4 b = REAL * 4	REAL * 4	Base is zero and exponent is zero or negative	1C0	448	EXP ALOG	CHCBJ
RAISE AN INTEGER BASE TO A REAL POWER	CHCBJB	y = i * * b	2	b = REAL * 4 i = INTEGER * 2	REAL * 4	Base is zero and exponent is zero or negative	1C0	448	EXP, ALOG	CHCBJ
	CHCBJC	y = i * * b	2	b = REAL * 4 i = INTEGER * 4	REAL * 4	Base is zero and exponent is zero or negative	1C0	448	EXP, ALOG	CHCBJ
RAISE A REAL OR INTEGER BASE TO A REAL POWER; BASE AND/OR EXPONENT DOUBLE PRECISION	CHCBKA	y = a * * b	2	a = REAL * 8 b = REAL * 8	REAL * 8	Base is zero and exponent is zero or negative	230	560	DEXP, DLOG	CHCBK
	CHCBKB	y = j * * b	2	b = REAL * 8 i = INTEGER * 2	REAL * 8	Base is zero and exponent is zero or negative	230	560	DEXD, DLOG	CHCBK
	CHCBKC	y = i * * b	2	b = REAL * 8 i = INTEGER * 4	REAL * 8	Base is zero and exponent is zero or negative	230	560	DEXP, DLOG	CHCBK
	CHCBKD	y = a * * b	2	a = REAL * 4 b = REAL * 8	REAL * 8 See Note.	Base is zero and exponent is zero or negative	230	560	DEXP, DLOG	CHCBK
	CHCBKE	y = a * * b	2	a = REAL * 8 b = REAL * 4	REAL * 8	Base is zero and exponent is zero or negative	230	560	DEXP, DLOG	CHCBK
RAISE A COMPLEX BASE TO AN INTEGER POWER	CHCBMA	y = a * * i	2	a = COMPLEX * 16 i = INTEGER * 4	COMPLEX * 16	Base is zero and exponent is zero or negative	274	628	—	CHCBM
	CHCBMB	y = a * * j	2	a = COMPLEX * 16 i = INTEGER * 2	COMPLEX * 16	Base is zero and exponent is zero or negative	274	628	—	CHCBM
	CHCBCA	y = a * * i	2	a = COMPLEX * 8 i = INTEGER * 4	COMPLEX * 8	Base is zero and exponent is zero or negative	24C	588	—	CHCBC
	CHCBCB	y = a * * j	2	a = COMPLEX * 8 i = INTEGER * 2	COMPLEX * 8	Base is zero and exponent is zero or negative	24C	588	—	CHCBC
PRODUCE ERROR MESSAGE AND TERMINATE EXECUTION	CHCBZA						E8	232	As required by use of the EXIT macro instruction	CHCBZ

NOTE: The REAL*8 function value returned by CHCBKD is not more accurate than the REAL*4 base given as an argument.

ENTRY NAME: In Table 2, this column shows the user entry name. In Table 3, this column shows the system entry name.

DEFINITION: This column gives a mathematical equation that represents the computation. (It is not meant to represent the way the subprogram is called.) An alternative equation is given when there is another way of representing the computation in mathematical notation. For example, the square root can be represented as either

$\sqrt{x}$ or $x^{1/2}$.

ARGUMENT(S): These columns describe the values(s) for which the function value is to be computed.

- Argument Number -- The number of arguments (one or two) that the user must supply.

- Argument Type -- The type and length of the argument. Integer, real, and complex represent the type of number; the notations *4, *8, and *16 represent the length, in bytes, of the argument.

Note: In FORTRAN IV, a real argument is a REAL*4 argument, and a double-precision argument is a REAL*8 argument. A single-precision complex argument is a COMPLEX*8 argument, and a double-precision complex argument is a COMPLEX*16 argument.

- Argument Range -- The valid range for an argument. (See the Error Condition and Error Message column descriptions below.)

FUNCTION VALUE RETURNED: This column describes the function value returned by the subprogram; the notation is the same as that used for the argument type.

ERROR CONDITION: This column shows the argument range not allowed when using the subprogram. If the argument is within this range, the subprogram will call CHCBZ. (See the description of CHCBZ, at the end of this section.)

STORAGE ESTIMATES: This column shows the approximate number of bytes required for each mathematical routine: the approximate total size of each routine's CSECT and PSECT. (FORTRAN IV mathematical routines each contain one public, read-only, reenterable CSECT and one PSECT. The length of each of the control sections is less than 4096 bytes. The routines are link edited, and their CSECTs are combined.)

OTHER SUBPROGRAMS REQUIRED: Many mathematical subprograms require other mathematical subprograms to perform their function. The entry names of the other subprograms are listed in this column. (This column does not include CHCBZA, which is called by all mathematical subprograms where error exit is possible.)

ROUTINE NAME: The name of the routine containing the subprogram.

ACCURACY FIGURES (TABLE 2 ONLY): These columns give accuracy figures for one or more representative segments within the valid argument range. The accuracy figures are based upon the assumption that the arguments are perfect (that is, without error and, therefore, having no error-propagation effect on the answers). The only errors in the answers are those introduced by the subprograms. Information given in the accuracy-figures columns is:

- Argument Range -- This column gives the argument range used to obtain the accuracy figures. For each function, accuracy figures are given more representative segments within the valid argument range. These figures are the most meaningful to the function and range under consideration.

  The maximum relative error and standard deviation of the relative error are generally useful and revealing statistics. However, they are useless for the range of a function where its value becomes 0, because the slightest error in the argument can cause an unpredictable fluctuation in the magnitude of the answer. When a small argument error would have this effect, the maximum absolute error and standard deviation of the absolute error are given for the range. For example, absolute error is given for sin (x) for values of x near π.

- Sample -- This column indicates the type of sample used for the accuracy figures; the type depends upon the function and range under consideration. The statistics may be based either upon an exponentially (E) distributed argument sample or a uniformly (U) distributed argument sample.

- Statistical results:

$M(\epsilon)=\text{Max}\left\lvert\frac{f(x)-g(x)}{f(x)}\right\rvert$	Maximum relative error produced during testing
$\sigma(\epsilon)=\sqrt{\frac{1}{N}\sum_i\left\lvert\frac{f(x_i)-g(x_i)}{f(x_i)}\right\rvert^2}$	Standard deviation (root-mean-square) of the relative error
$M(E)=\text{Max}\lvert f(x)-g(x)\rvert$	Maximum absolute error produced during testing
$\sigma(E)=\sqrt{\frac{1}{N}\sum_i\lvert f(x_i)-g(x_i)\rvert^2}$	Standard deviation (root-mean-square) of the absolute error.

In the formulas for the standard deviation, N represents the total number of arguments in the sample; i is a subscript that varies from 1 to N.

Test ranges, where they do not cover the entire legal range of a subroutine, were selected so that users may infer from the accuracy figures presented the trend of errors as an argument moves away from the principal range. The accuracy of the answer deteriorates substantially as the argument approaches the limit of the permitted range in several of the subroutines. This is particularly true for trigonometric functions. However, an error generated by any of these subroutines is, at worst, comparable in order of magnitude to the effect of the inherent rounding error of the argument.

Tables 4, 5, and 6 show how the values of the base and exponent affect the exponentiation subprograms.

Table 7 shows the system entry names and compiler-generated entry names of indirect-reference mathematical routines.

Table 4. Exponentiation with integer base and exponent

Base (I)	Exponent (J)		
	J > 0	J = 0	J < 0
I > 1	Compute the function value	Function value = 1	Function value = 0
I = 1	Compute the function value	Function value = 1	Function value = 1
I = 0	Function value = 0	Error Message 241	Error Message
I = -1	Compute the function value	Function value = 1	If J is an odd number, function value = -1 If J is an even number, function value = 1
I < -1	Compute the function value	Function value = 1	Function value = 0

Table 5. Exponentiation with real or double-precision base and integer exponent

Base (A)	Exponent (J)		
	J > 0	J = 0	J < 0
A > 0	Compute function value	Function value = 1	Compute function value
A = 0	Function value = 0	Error message	Error message
A < 0	Compute function value	Function value = 1	Compute function value

Table 6. Exponentiation with real or double-precision base and exponent

Base (A)	Exponent (B)		
	B > 0	B = 0	B < 0
A > 0	Compute function value	Function value = 1	Compute function value
A = 0	Function value = 0	Error message	Error message
A < 0	Error Message	Function value = 1	Error message

Table 7. Entry names of indirect-reference mathematical routines

Compiler-Generated Entry Name	Routine Name	System Entry Name
FCXPJ	CHCBC	CHCBCA
FCXPI	CHCBC	CHCBCB
FIXPI	CHCBG	CHCBGA
FJXPJ	CHCBG	CHCBGB
FJXPI	CHCBG	CHCBGC
FIXPJ	CHCBG	CHCBGD
FRXPI	CHCBH	CHCBHA
FRXPJ	CHCBH	CHCBHB
FDXPI	CHCBI	CHCBIA
FDXPJ	CHCBI	CHCBIB
FRXPR	CHCBJ	CHCBJA
FJXPR	CHCBJ	CHCBJB
FIXPR	CHCBJ	CHCBJC
FDXPD	CHCBK	CHCBKA
FJXPD	CHCBK	CHCBKB
FIXPD	CHCBK	CHCBKC
FRXPD	CHCBK	CHCBKD
FDXPR	CHCBK	CHCBKE
FCDXI	CHCBM	CHCBMA
FCDXJ	CHCBM	CHCBMB
	CHCBZ	CHCBZA*

*Called by mathematical library routines

Table 8 describes the macro instructions used by mathematical library routines. The mathematical library makes extensive use of macro instructions for code similarity, programming efficiency, and ease of interface modification.

CHCBZ -- Error Processor

Entry Parameters: CHCBZ expects general register 1 to point to a two-word parameter list. The first word is the address of a one-byte code indicating the error condition; codes and meanings are:

0 - zero
1 - not positive
2 - exceeds limit
3 - negative
4 - absolute value exceeds limit
5 - zero or negative
6 - close to singularity

The second word is a pointer to the name of the mathematical routine in which the error occurred, with a one-byte length preceding the name. CHCBZ does a trace back to obtain the address of the call to the mathematical routine in the user's program. CHCBZ issues a standard message, inserting the supplied values and issues the EXIT macro instruction. The message is:

```
CHCBZ100  TERMINATED: ARGUMENT $1 FOR
                      $2 AT $3

       where $1 represents the code
             $2 represents the name
             $3 represents the trace-
                back address
```

Entry Point: CHCBZA

Operation: CHCBZ adds the error code to the standard system message and issues the EXIT macro instruction. The standard system message is:

OUT OF RANGE PARAMETER GIVEN A FORTRAN IV SUPPLIED SUBPROGRAM. ERROR CODE IS nnn.

Table 8. Mathematical library macro instruction summary

Macro Name	Purpose
CEKZA	Generates the V-Con/R-Con pair required when other TSS/360 subroutines must be called.
CEKZD	Generates the code following a SAVE macro instruction, to add to the backward and forward chains and load register 13 with the current PSECT location.
CEKT1	Generates the CSECT label, register EQUs, ENTRY statement, and other code duplicated at the beginning of every subroutine.
CEKT2	Used for error exits.
CEKT3	Generates PSECT label and save area.
CEKT4	Generates code similar to that of CEKT1, for additional entry points.
CEKT5	Similar to CEKT3, but used for PSECT items required for additional entry points.
CEKT6	Used to reload register 13, stop the forward chain, and return.

This section gives the following information on each service subprogram:

- Name
- Name of containing routine
- Function
- Attributes
- Entry
- Routines called
- Error checks
- Data references
- Operation

GENERAL INFORMATION

The three service routines are:

- CHCBE -- interrupt and machine indicator subprograms
- CHCIV -- DUMP and PDUMP subprograms
- CHCIW -- EXIT, STOP, and PAUSE subprograms

CHCBE resides in the mathematical library; CHCIV and CHCIW reside in the I/O library.

Like other FORTRAN IV library routines, the service routines use type-I linkage and are nonprivileged, reenterable, and closed.

Service subprograms that are user-referenced have user entry names. No service subprograms have compiler-generated entry names.

Information on entry parameters is contained in the individual routine summaries. Service subprograms do not pass exit parameters, except for the calls to CHCBZ (mathematical library error-handling routine) by the SLITE and SLITET subprograms. (See subprogram summaries.)

SUBPROGRAM SUMMARIES

Table 9 briefly describes all service subprograms except the Exit Routine subprograms CHCIW4 and CHCIW5, which are referenced by the I/O library routines.

The following summaries should be read in conjunction with flowcharts BD, BE, IV, and IW.

CHCBE -- Interrupt and Machine Indicator Routine

Subprograms:

CHCBE2 -- Specification Exception Handler

CHCBE3 -- Exponent Overflow Handler

CHCBE4 -- Exponent Underflow Handler

CHCBE5 -- Divide Check Handler

OVERFL -- Exponent Overflow and Underflow Tester

DVCHK -- Divide Check Tester

SLITE -- Sense Light Handler

SLITET -- Sense Light Tester

Function: To simulate certain machine indicators, and to mask off or process interruptions caused by these exceptions:

- Fixed point overflow
- Fixed point divide
- Exponent overflow
- Exponent underflow
- Significance
- Floating point divide

(See Chart BE.)

Entry Parameters: CHCBE2 through CHCBE5 have no entry parameters. OVERFL, DVCHK, and SLITE each receive one parameter, the address of an integer variable. A pointer to this address is passed in register 1. For SLITET, register 1 points to two addresses, the address of the integer variable indicating the sense light number and the address of the integer variable to receive the result code.

Routines Called: CHCBZ -- Error Processor (CHCBZ)

Error Checks: If entered at either SLITE or SLITET, this routine tests the integer variable and notes an error condition if it is any value but 0 to 4 for SLITE, or 1 to

Table 9. Summary of service subprogram characteristics

Function (1)		Entry Name (2)	Error Condition (3)	Storage Estimates (4) HEX	DEC	Module Name (5)
Pseudo sense light subprograms	Turn all sense lights off or one sense light on	SLITE	Argument other than 0, 1, 2, 3, 4	324	804	CHCBE
	Test a sense light or record its status	SLITET	Argument other than 1, 2, 3, 4			CHCBE
Overflow and underflow subprogram	Test and record status of exponent overflow and underflow indicators	OVERFL				CHCBD
Divide check subprogram	Test and record status of divide check indicator	DVCHK				CHCBD
Exception processing subprograms	Process arithmetic exceptions	CHCBE3 (EXPONENT OVERFLOW) CHCBE4 (EXPONENT UNDERFLOW) CHCBE5 (DIVIDE CHECK)				CHCBE
	Process specification exceptions	CHCBE2 (SPECIFICATION)				CHCBE
Exit subprogram	Terminate execution	EXIT CHCIW2 (STOP) CHCIW3 (PAUSE)		1AC	428	CHCIW
Dump subprogram	Dump specified storage area with or without termination	DUMP, PDUMP		48	168	CHCIV

4 for SLITET. If there is an error condition, CHCBE calls CHCBZA, passing the appropriate error code as a parameter. CHCBZA terminates the task after printing the error code. If entered at CHCBE2, CHCBE checks for an interruption that should not occur and allows the standard system action for such an interruption.

Operation: At the beginning of all FORTRAN main programs, the compiler generates code that calls CHCBD1, the exception processing enabler and I/O initialization routine included in the I/O library.

The system interruption handler, CHCBD1, calls one of the CHCBE2 through CHCBE5 subprograms whenever an exception occurs.

CHCBE2 passes control to CHCBE1 to control fixup of a specification exception.

CHCBE2 checks the interruption address and the instruction that caused the interruption to determine whether the interruption was due to a condition that should not occur. If so, the standard system action is invoked. If not, CHCBE1 interprets the instruction causing the interruption and executes it as though the alignment restriction was removed. The save area referred to by register 0 at entry is changed to reflect instruction execution.

CHCBE3 sets to one the flag which CHCBD6 tests if an "OVERFL (j)" statement occurs in the user program.

CHCBE4 does the same as CHCBD3, except that the flag is set to three.

CHCBE5 sets to one the flag which CHCBD7 tests if a "DVCHK (j)" statement occurs in the user program.

OVERFL puts the value of the overflow-underflow indicator into the user-specified variable, resets the indicator, and returns control to the user.

DVCHK puts the value of the divide check indicator into the user-specified variable, resets the indicator, and returns control to the user.

SLITE tests the user-specified variable for a value of zero. If it is zero, all four indicators are set to zero and control is returned to the user. If the variable is not zero, it is tested for values of one, two, three, and four. If it is one of these values, the corresponding indicator is set to one and control is returned to the user. If the variable is not one of these values, SLITE calls CHCBZA and then returns control to the user.

SLITET tests the first user-specified variable to see that it is within the proper range. If the variable is in error, CHCBD9 calls CHCBZA and returns control to the user. If the variable is from one to four, the corresponding indicator is tested. If the indicator is zero, SLITET sets the second user-specified variable to two and returns control to the user. If the indicator is one, SLITET sets the second variable to one, resets the indicator to zero, and returns control to the user.

CHCIV -- Dump Routine

This routine causes the user-indicated limits of storage to be dumped in the format desired, with or without program termination. The user calls this module by specifying either of the statements CALL DUMP (A_1, B_1, F_1, ..., A_n, B_n, F_n) or CALL PDUMP (A_1, B_1, F_1, ..., A_n, B_n, F_n) in his source program. The variable data names, A and B, indicate the limits of storage to be dumped and the integer, F, indicates the dump format desired. (See Chart IV.)

For sample printouts by the DUMP and PDUMP subprograms, see FORTRAN IV Library Subprograms, GC28-2026, "Appendix D: DUMP and PDUMP Sample Storage Printouts."

Entry: There are two entry points: CHCIV1 and CHCIV2, for DUMP and PDUMP, respectively. Standard type-I linkage is used with register 1 pointing to the first address constant in the parameter list. The parameter list is variable-length and has the following format:

Word 1		Word size of this parameter list, minus one.
Word 2	First Group	Address constant pointing to the starting location symbol for the dump.
Word 3		Address constant pointing to the end location symbol for the dump.
Word 4		Address constant pointing to the FORMAT code word. When FORMAT code is zero, the FORMAT code word is divided into two parts: first half-length of end location symbol minus one second half - 0.
Word 5	Second Group	Same as words 2, 3, and 4 respectively.
Word 6		
Word 7		
. . .		. . .

The entries in the parameter list (excluding the first word) are groups of three words. There may be more than one group depending on the number of different areas of virtual storage the user wants dumped. The first two words of a group may be in either order; but the dump is always taken in ascending order. The third word of a group, the format code word, represents the type of data conversion to be performed, as follows:

0	hexadecimal	5	real*4
1	logical*1	6	real*8
2	logical*4	7	complex*8
3	integer*2	8	complex*16
4	integer*4	9	literal

The format code 0 is the default value.

Routines Called:

- Integer Output Conversion (CHCIH)
- Complex Output Conversion (CHCIN)
- Logical Output Conversion (CHCIR)
- General Output Conversion (CHCIT)
- Exit (CHCIW)
- GATE facilities (GATWR macro instruction)

Error Checks: CHCIV makes no error checks; all error checking is done by the called data conversion routines.

Operation: CHCIV scans the parameter list to locate a parameter group. The specified dump area is then formatted and written on SYSOUT using the GATE macro instruction facility. After the parameter list has been completely processed, control returns to the user program, if PDUMP was specified. If DUMP was specified, control passes to the exit routine (CHCIW) for program termination.

CHCIV determines the presence of the third word in a parameter group by testing the value of the word in storage to which it refers. If the value is 9 or less, it is assumed to be the format code. If the value is greater than 9 (or if it is negative), it is assumed that the third word was omitted and a format code of zero (hexadecimal) is taken as a default. If the format code is either 0 or 9 (hexadecimal or literal), the conversion formatting is done within this routine; otherwise, the appropriate output conversion routine is called.

CHCIW -- Exit Routine

This routine performs user-program exit functions whenever any of the following FORTRAN statements are executed:

CALL EXIT

STOP n

PAUSE n

PAUSE 'message'

In the case of a STOP or PAUSE statement, this module causes a one- through five-digit integer (n) or a message ('message') to be produced at the user's SYSOUT. With the exception of the PAUSE statement, CHCIW causes all DCBs in use in the DCB table to be closed. (See Chart IW.)

Entry: The five entry points and their associated causes of entry are as follows:

CHCIW1	Execution of a CALL EXIT statement
CHCIW2	Execution of a STOP statement
CHCIW3	Execution of a PAUSE statement
CHCIW4	Exit requested by other I/O library routines
CHCIW5	Exit requested by a user subprogram error

Standard type-1 linkage is used with all entries except CHCIW4, which uses type-4 restricted linkage. In the case of STOP and PAUSE statements, the address of a parameter list is passed in register 1. The parameter list is fixed-length and has the following format:

Word 1	Address of byte containing the length of the integer or message (n or 'message'). A length of zero indicates that no integer or message was defined by the user.
Word 2	Address of the integer or message, if any.

There is no parameter list for entry to CHCIW1, CHCIW4, or CHCIW5.

Routines Called:

- DCB Maintenance (CHCIB)
- I/O Control (CHCIC)
- Task Supervisor Routines (CZAFQ1), entered by use of a supervisor call (SVC)
- I/O Communication (CHCIY)
- PRMPT (CZATJ1)

Operation: If this routine is entered via CHCIW4 or CHCIW5, a message is issued to the user indicating the reason it was called. If this routine is entered via CHCIW1, CHCIW2, or CHCIW3, a message is issued (including a user-supplied message, if any) to the user by the PRMPT macro instruction facility (CZATJ1) indicating the reason it was called. In the case of PAUSE, return is then made to the user program. If entry to this routine was not caused by a PAUSE statement, the DCB table, maintained by CHCIB, is searched and each open DCB is closed. The DCB table pointers are then cleared, and exit is made to the system exit routine.

This section gives the following information on each I/O library routine, excluding the service routines, which are described in Section 4:

* Name
* Function
* Attributes
* Entry
* Routines called (where applicable)
* Error checks (where applicable)
* Data references (where applicable)
* Operation

The routines are described in the alphabetical order of their names.

GENERAL INFORMATION

Certain information is common to most I/O routines. This information includes:

* Names
* Attributes
* Format of parameters passed to data conversion routines

Attributes

All FORTRAN I/O routines are nonprivileged, reenterable, and closed. CHCIA, CHCIE, CHCIU, CHCIW1, CHCIW2, CHCIW3, CHCIW5, CHCIV1, CHCIV2, and CHCBD1 are entered by standard type-1 linkage with the address of a parameter list (where applicable) in register 1; exit is a return to the calling routine. All of the other I/O routines are entered by restricted type-4 linkage.

Work Areas and Register Save Areas

The FORTRAN I/O routine has one common PSECT -- CHCRWW -- which contains all necessary work and storage areas and a 19-word register save area. All parameters used by the data conversion routines are stored in this PSECT.

ROUTINE SUMMARIES

CHCIA -- I/O Initialization

This routine is the initial FORTRAN I/O library interface with the user. It manages the disposition of each I/O request by setting switches about formatted and unformatted I/O for the information of other I/O routines, by allocating a buffer area for output requests, and by obtaining a logical record for input requests.

Every I/O request in the FORTRAN user program causes the compiler to generate one call to CHCIA. On this call, if there is no list, CHCIA supervises the complete execution of an I/O request. If the I/O request is a READ, WRITE, PRINT, or PUNCH with list, CHCIA simply prepares the I/O library for compiler-generated calls to List Item Processor (CHCIE) and List Termination (CHCIU).

Entry: The entry point is CHCIA1. The parameter list is variable-length and has this format:

Word 1	Address of a fullword containing the user-specified data set reference number.
Word 2	Address of a control byte in I/O Control (CHCIC) indicating type of operation: READ, WRITE, PRINT, PUNCH, REWIND, BACKSPACE, ENDFILE.
Word 3	Address of a control byte in I/O Control (CHCIC) indicating whether a list was present in the I/O statement and whether any of the following parameters in this list are present.
Word 4 (Optional)	Address of a FORMAT string or NAMELIST table. This parameter is included only if the I/O source statement has an associated FORMAT or NAMELIST source statement.
Word 5 (Optional)	Address of an error exit. Included only if the I/O source statement has an ERR operand.
Word 6 (Optional)	Address of an end-of-file exit. Included only if the I/O source statement has an END operand.

If any optional parameter is missing, any parameters following it are moved up in the list and the list is shortened. For example, if there is no FORMAT or NAMELIST address and no error exit address, word 4 of the parameter list would be the end-of-file exit address.

Routines Called:

- DCB Maintenance (CHCIB)
- I/O Control (CHCIC)
- FORMAT Processor (CHCIF)
- NAMELIST Processor (CHCID)
- Exit (CHCIW)
- PRMPT (CZATJ1)

Error Checks: If the user-specified data set reference number is negative, an error message is issued by the PRMPT macro instruction facility (CZATJ1), and CHCIW is entered to terminate the user program.

Operation: Upon entry, the FORTRAN I/O PSECT switches are cleared, control parameters are set, and CHCIB is called to find a DCB corresponding to a user-specified data set reference number (or to create a DCB if one is not found). CHCIC is then called to complete the I/O request.

If the operation is a REWIND, BACKSPACE, or ENDFILE, return is made to the user. If the operation is READ or WRITE according to a NAMELIST, CHCID is called before returning to the user.

If the operation is a WRITE with no list and no format, a dummy record containing two bytes of zeros is written with a call to CHCIC before returning to the user. If a FORMAT statement is involved, CHCIF is called. If the operation is a WRITE with no list and CHCIF is called for construction of a record anyway, CHCIC is called to write the record before returning to the user; otherwise, return is made immediately after calling CHCIF. If a FORMAT statement is not involved, return is made to the user.

CHCIB -- DCB Maintenance

This routine finds or initializes the data control block (DCB) that contains a description of the data to be transmitted by a user-specified I/O operation. If an appropriate DCB is not found, this routine allocates the necessary space in the DCB table and constructs a new DCB, including within it information about the data to be transmitted that the user defined in his DDEF command. (See Chart IB.)

Entry: The entry point is CHCIB1. CHCIA stores the address of the user-supplied data set reference number and the data set reference number itself if present in the I/O common PSECT.

Routines Called:

- Data management routines used to search for and read JFCBs (CZAEB)
- Data management routines used to allocate storage for DCB construction (CZCGA)
- Exit (CHCIW)
- PRMPT (CZATJ1)

Error Checks: If the user-specified data set reference number exceeds 99, an error message is issued by the PRMPT macro instruction facility (CZATJ1), and CHCIW is entered to terminate the user program. No alternate data set reference number is established.

If a discrepancy exists in the user DDEF command between permissible RECFM, KEYLEN, and DSORG values, an error message is issued by the PRMPT macro instruction facility (CZATJ1), and CHCIN is called to terminate processing. A description of the assumptions FORTRAN I/O makes an initializing associated DCBs is contained in Appendix A.

Format and Content of the DCB Prefix: The DCB Prefix is used by the FORTRAN I/O routines, in conjunction with the DCB, when performing any type of I/O operation. The DCB Prefix, created by the DCB Maintenance routine (CHCIB), is eight words long and always immediately precedes the DCB itself. (See Table 10.)

Operation: This routine is called to perform one of three functions: to supply the address of an initialized DCB corresponding to a specified data set reference number; to locate the first open DCB in the DCB table used by the I/O library routines; or to obtain a GATE DCB. The first function is performed by searching the table of DCB entries. If the appropriate DCB is found, the address of that DCB is returned. If the search fails, the data management routine CZAEB is called to determine whether the user has issued a DDEF command (which causes the creation of a Job File Control Block -- JFCB) for the specified data set reference number. If CZAEB finds a JFCB that corresponds, space is reserved in the DCB table, a DCB is constructed, and the new DCB is "chained" into the DCB table. If CZAEB cannot locate a corresponding JFCB (meaning that the user did not supply a

Table 10. Format and content of DCB prefix

Word 1	Address of the starting location in the buffer area for the current logical record.
Word 2	Address of the current location in the buffer area for the current logical record.
Word 3	Address of the end location in the buffer area for the current logical record.
Word 4	Byte 1: Current operation (READ, WRITE, etc.) Byte 2: Control flags (FORMAT, NAMELIST, List, ERR exit, END exit) Byte 3: Control flags (Span, GATE, recent READ, END or ERR encountered) Byte 4: Previous operation (byte 1 from last call on CHCIC with this DCB)
Word 5	Address of current DECB, if required (BSAM).
Word 6	User-specified data set reference number, plus one.
Word 7	Address of the next DCB.
Word 8	Address of last DCB associated with this data set reference number.
Word 9	DCB begins here.

DDEF command), a GATE DCB is constructed and initialized.

The GATE DCB causes SYSIN and SYSOUT to be treated as the input origin and output destination, respectively.

CHCIB performs its second function, locating the first open DCB in the DCB table, when a negative data set reference number has been specified by CHCIW at final close of the file. If an open DCB is not found in the DCB table, normal return is made to the calling routine.

CHCIB performs its third function, obtaining a GATE DCB, when the address of the data set reference number is zero. In this case, the I/O request is assumed to be for the first data set reference number that this routine defines by a GATE DCB. Therefore, as before, SYSIN and SYSOUT are treated as the input origin and output destination, respectively. Note that the GATE DCB is a dummy DCB (with a prefix), used to allow other I/O routines to treat the case of GATE I/O the same as the case of data sets defined by a DDEF command.

CHCIC -- I/O Control

This routine fulfills I/O requests made through other I/O library routines by using the data management macro instruction facilities of TSS/360. The particular data management facilities to be used are determined both by the type of I/O statement issued in the user program, and by related DDEF commands, if any, defining such things as the type of records being transferred and the manner in which they should be processed.

The following list identifies the more significant macro instructions used by CHCIC for each of the FORTRAN I/O statements. Other macros used in conjunction with those listed below (e.g., OPEN and CLOSE) are identified in the "Operation" section of this routine description. (See Chart IC.)

FORTRAN I/O Statement	CHCIC Function
READ	Obtains a logical record from a user-specified input source by using the READ, GATRD, or GET macro instruction.
WRITE	Initializes the writing of a logical record by establishing pointers to the output buffer area. Subsequent output processing is performed by using the WRITE, GATWR, or PUT macro instruction.
REWIND	Repositions the user-specified volume of one or more data sets to the first record of the first data set by using the POINT or SETL macro instruction.
BACKSPACE	Repositions the user-specified data set to the previous logical record by using the NOTE, POINT, SETL, and BSP macro instructions.
ENDFILE	Defines the end of the user-specified data set by using the WRITE and STOW macro instructions.

Entry: The entry point is CHCIC1.

Routines Called:

- DCB Maintenance (CHCIB)
- Exit (CHCIW)
- Data management routines to perform I/O functions as determined by the macro instruction issued.
- PRMPT (CZATJ1)

Error Checks: If the I/O operations performed by data management cause either a SYNAD or EODAD exit, and if the user provides an ERR or END return point, CHCIC locates the adcons for these return points in the work area CHCRWW, and locates the register save area for the user's program registers. Return is then made to the ERR or END return point rather than to the calling I/O routine.

If the user does not provide return points (or if the operation is other than a READ statement), an error message is issued and the program is terminated.

If an invalid character is encountered in hexadecimal input from a GATRD operation performed for an unformatted READ statement, an error message is issued and the erroneous character is treated as the termination of the hexadecimal input. Processing then continues.

In addition to the above error checks, error messages are issued (PRMPT macro instruction) and the user program is terminated by CHCIW for any of the following reasons:

1. The record is not format-V for unformatted READ statement.
2. Error return code received from the use of the FIND or STOW macro instruction for a member in a VPAM data set.
3. Invalid sequence of I/O operations for a user-specified data set reference number. The invalid sequences are: READ preceded by END FILE; END FILE preceded by READ; and READ preceded by WRITE (except when using GATE I/O).

Operation: In fulfilling the I/O request, this module ensures that the DCB associated with the DDEF command is properly opened by examining the access qualifier (for example, read-only) of the data set. If the data set is read-only, the OPEN macro instruction is issued for INPUT. Otherwise, the OPEN macro instruction is issued for UPDAT, when the Virtual Access Methods (VAM) are specified, or INOUT, when the Basic Sequential Access Method (BSAM) is specified. When a data set is opened for UPDAT or INOUT, both reading and writing may be performed. However, if an attempt is made to write on a read-only data set that is opened for INPUT, the user program is abnormally terminated. When the user does not issue a DDEF command for a given I/O request, CHCIC uses the GATE macro instruction facility to perform reading and writing. This facility does not require issuance of an associated OPEN macro instruction. CHCIC first checks whether the I/O operation requested is compatible with the previous operation. If it is not, an error message is issued and the user program is terminated. If it is compatible, processing continues according to the particular type of I/O request involved (that is, READ, WRITE, WRITE Initialize, REWIND, BACKSPACE, and ENDFILE).

If a READ is issued, the data set is opened, if necessary, and the pointers in the DCB prefix are set to the beginning and end of the data. CHCIC then checks which access method is being used. If VAM is used, a GET macro instruction is issued. If the GATE macro instruction facility is used, a GATRD macro instruction is issued. If BSAM is used, the record format is examined to determine whether the records are blocked. If they are not blocked or if the previous record was the last record of the block, a READ is issued; otherwise, the next logical record within the current block is used. In any case, after the data is located, the beginning and end are determined and the pointers are saved in the DCB prefix. Return is then made to the calling routine.

If a WRITE Initialize is issued for an open data set for which the last operation was an END FILE, the data set is closed and an end file lockout indicator is set on in the DCB prefix, indicating to DCB Maintenance (CHCIB) that this DCB should be ignored when looking for a DCB for this data set reference number. As a result of this indicator being set, any subsequent WRITE statements referring to the same data set reference number causes the creation of a new data set. A call is then made to CHCIB to obtain a new DCB describing the new data set.

Whether or not the previous operation was an END FILE, the DCB is opened, if necessary. If VAM is used, a SETL macro instruction is issued to point to the beginning of the specified data set. In all cases, the pointers to the output buffer area are then set in the DCB prefix and return is made to the calling routine.

If a WRITE is issued, CHCIC checks the record format type. For non-format-V records, fill characters (padding) are inserted, if necessary, to the end of the record. Blanks are used for NAMELIST or formatted WRITE; binary zeros are used otherwise. For format-V records, the record length and any needed spanning bits are set. The spanning bits are used to define the relationship between the FORTRAN logical record and the data management logical record when the FORTRAN logical record exceeds the length of the maximum record supported by the access methods being employed. (See Appendix A for more details on spanning bits.)

CHCIC next checks the access method being used for the write request. For VAM, a PUT macro instruction is issued. For the GATE macro instruction facility, a GTWRC is issued. For BSAM, the record format is examined to determine whether the records are blocked. If the records are not blocked or the block is full, a WRITE macro instruction is issued. For blocked records, where the block is not full, the output area pointers in the DCB prefix are updated to the next record area in the block. In all cases, the DCB prefix is re-initialized and return is made to the calling routine.

If a REWIND is issued, CHCIC searches the DCB table for all DCBs relating to the user-specified data set reference number. For each such DCB found, the end file lockout indicator within it is turned off. The POINT macro instruction is then used to reposition the volume to the first record of the first data set, if the data set is BSAM; the SETL macro instruction is used for VAM. Return is then made to the calling routine.

If a BACKSPACE is issued, CHCIC checks the access method being employed. For VAM, CHCIC checks to see whether the last operation was an ENDFILE. If so, a CLOSE macro instruction is issued followed by an OPEN macro instruction and a SETL macro instruction pointing to the end of the data set. In either case, a SETL macro instruction is then issued to point to the previous record.

If BSAM is used, CHCIC examines the record format to determine whether the records are blocked. If unblocked and if the last operation was an END FILE, a NOTE macro instruction is issued to establish the current location, CLOSE and OPEN macro instructions are issued, and a POINT macro instruction is issued to reposition the data set to the position prior to closing. A BSP macro instruction is then issued to backspace. If double buffering is used, a second BSP is issued. If the last operation was not an END FILE, only the BSP macro instruction is issued. If the records are blocked, a scan is made backwards through the block to locate the beginning of the previous record. In some cases it may be necessary to issue a BSP macro instruction and then read the previous block in order to position to the previous record. In all cases, it is not possible to position to a record in a previous data set if a READ or WRITE was issued after an END FILE operation. Upon completion of processing, return is made to the calling routine.

If an END FILE is issued, it is noted but otherwise ignored; the actual END FILE operation is performed only when the next operation is a WRITE Initialize or a BACKSPACE.

CHCID -- NAMELIST Processor

This routine interacts with CHCIC to control the I/O for each NAMELIST record and interacts with the appropriate data conversion routines to effect the desired item-by-item conversion. (See Chart ID.)

Entry: The entry point is CHCID1.

Routines Called:

- I/O Control (CHCIC)
- Complex Input Conversion (CHCIM)
- Complex Output Conversion (CHCIN)
- General Input Conversion (CHCIS)
- General Output Conversion (CHCIT)
- Exit (CHCIW)
- PRMPT (CZATJ1)

Error Checks: There are no error checks for output. For input, if errors are detected in the NAMELIST table, a message is issued via PRMPT, and CHCIW is called to terminate the user program. Other error messages are generated for any of the conditions listed below. In these cases, processing continues with the next entry of the input record.

- Name exceeds six characters
- First character of each input record is not blank
- Subscripts appear on a name that is not an array name

- Incorrect number or range of subscripts
- Subscripting causes array size to be exceeded
- Multiple constants or repeated constants appear with a name that is not an array or a subscripted array name
- Multiple constants or repeated constants exceed the size of an array or the size of an array portion beginning at a specified element
- An equal sign or left parenthesis is not preceded by the variable or array name for that item
- An invalid character appears in a repeat constant
- End of a logical record caused an item to be logically incomplete
- The NAMELIST name is not in the NAMELIST table

NAMELIST Table: The address of the NAMELIST table generated by the FORTRAN compiler is communicated in the call to I/O Initialization (CHCIA) and then passed to this routine. The table is made up of two-word entries, each of which contains an identifier in the first halfword.

NAMELIST Name Entry:

```
Bytes 0-1   Identifier (X'0100')
      2-7   Name (left-justified)
```

Variable Name Entry:

```
Bytes 0-1   Identifier (X'0200')
      2-7   Name (left-justified)
```

Variable Type and Location Entry:

```
Bytes 0-1   Identifier (X'300')
        2   Length and Type (4 bits each)

            Length:  Number of bytes
                     minus 1

            Type:    X'01' Logical
                     X'02' Integer
                     X'03' Real
                     X'04' Complex

        3   Class:   Letter A for array;
                     otherwise, an S

      4-7   Storage Location
```

Array Size Entry:

```
Bytes 0-1   Identifier (X'0400')
      2-3   Not used
      4-7   Number of bytes in array
```

Dimension Product Entry:

```
Bytes 0-1   Identifier (X'0500')
      2-3   Not used
      4-7   Dimension Product (see
            explanation below)
```

Terminal Entry:

```
Bytes 0-3   Zero
      4-7   Not used
```

A dimension is a level of subdivision, or level of subscripting, within an array. For example, an array could be a string of seven thirty-word elements (first dimension), each subdivided into six five-word elements (second dimension), each subdivided into five one-word elements (third dimension). An array may have as many as seven dimensions.

For each dimension there is a corresponding dimension product, which is the product of: (1) the byte-size of the array's smallest element, (2) the number of elements within all lower dimensions except the first dimension, and (3) the number of elements within that dimension. In the example just given, the dimension product for the third dimension would be 4x6x5, or 120. This dimension product would be seven times greater if there were another dimension before the seven-element dimension. The dimension product for the first dimension is always the byte-size of the array's smallest element -- this dimension product is never entered. If there is only one level of subdivision, there is no Dimension Product Entry.

Operation: For a WRITE, CHCID calls CHCIT to write, in the following format, the data for each variable or array in the NAMELIST:

```
For Variables:
   variable name = constant

For Arrays:
   array name or array element name =
   constant1, constant2, ...
```

In the second case, each constant may have a repetition notation appended in the form that is also usable for input.

CHCID also produces header and trailer records so that the information can be read with the same NAMELIST.

For a READ, CHCID reads records until a correct header record is located. It then uses a scanning procedure to locate the input entries. As each is found, the routine locates the variable or array name in the NAMELIST Table, determines its size and type, and converts the input.

Error checking functions performed by CHCID for a READ operation have already been described under "Error Checks".

CHCIE -- List Item Processor

Every I/O statement in the user's source program generates one or more calls to this routine if there is a list associated with a READ, WRITE, PRINT, or PUNCH. A list item may be a simple variable, an array element (a subscripted variable), or an entire array. If a FORMAT statement is specified, this routine calls on FORMAT Processor (CHCIF) to control any necessary conversion. If there is no FORMAT statement, CHCIE is directly responsible for filling or emptying the output or input buffer area. (See Chart IE.)

Entry: The entry point is CHCIE1. Register 0 contains either zeros, if the list item is a single element, or a number expressing the array length, in bytes, if the list item is an entire array. The parameter list is fixed-length and has the following format:

Word 1	Address of a control byte. The first four bits of the control byte contain the size of the element, minus one. The second four bits contain a flag indicating the type of item as follows: Flag / Type of Item 01 logical 02 integer 03 real 04 complex
Word 2	Address of a first (or only) element of the list item.

Routines Called:

- Format Processor (CHCIF)
- I/O Control (CHCIC)
- Exit (CHCIW)
- PRMPT (CZATJ1)

Error Check: With unformatted input, if a list item is requested after the logical record is exhausted, an error message is transmitted to the user via PRMPT, and CHCIW is called to terminate the user's program.

Operation: At entry, register 0 either contains zeros if the list item is a single element, or a number expressing the array-length, in bytes, if the list item is an entire array. Another parameter indicates the length of the list item (in the case of an array, the length of each individual item in the array). The transfer of data between I/O buffers and item storage locations is performed by CHCIE if there is no FORMAT statement; if there is a FORMAT statement, the transfer of data is accomplished by a call to CHCIF. After the transfer of information between the I/O buffer and the storage location specified for the item, the item length is subtracted from the array length. A negative or zero result indicates either that the item was a single element or that the last element in the array has been processed; the routine returns to the caller. If the result is positive, the storage location of the list item and the location in the buffer are each incremented by the item length and transfer is made to the beginning of the loop to process the next element of the array. If, after incrementing the buffer location, the end of the buffer has been reached, further processing depends on whether the operation is for input or output.

For output, the current buffer is written and flagged to indicate that this logical record continues on the next physical record. A new output buffer is then initialized.

For input, a check is made to see if the current physical record is flagged in this manner, and if it is, the next physical record is read. If the current physical record is the last one written as part of this logical record, an error message is transmitted to the user and the task is terminated. The reading and writing of additional physical records as part of this logical record is accomplished by a call to I/O control (CHCIC).

CHCIF -- Format Processor

This routine scans the FORMAT string, interacts with CHCIC to control the I/O for each FORMAT-referenced record, and interacts with the appropriate data conversion routines to effect the item-by-item conversion specified by the referencing FORMAT statement. CHCIF is entered once for each list item in the I/O request. (See Chart IF.)

Entry: The entry point is CHCIF1. Before the first entry to CHCIF to process a reference to a FORMAT statement, CHCIA (I/O Initialization) stores the address of the FORMAT string in CHCRWW.

Routines Called:

- I/O Control (CHCIC)
- Exit (CHCIW)
- One of the eleven conversion routines (CHCIH through CHCIT)
- The FORTRAN compiler routine for translating FORMAT statements (CEKBF), at entry point SYSPFMT
- PRMPT (CZATJ1)

Error Checks: Since FORMAT statements may be dynamically modified, certain error conditions may arise due to the syntax of the FORMAT string. If there are no syntax errors, errors could arise due to conversion of the data. In such cases the conversion routines issue messages describing the errors before returning. All syntax error checks produce messages describing the error.

Processing is terminated upon encountering invalid control characters in the string, strings that exceed the maximum, or too many levels of parentheses. When it is possible to assume values other than those specified (as in the case of invalid size of w or d fields after a control character), processing continues on the current item after the error message is issued. Otherwise, the erroneous FORMAT item is skipped, and processing continues with the next control character.

Operation: At the first entry to CHCIF for a given I/O request, CHCIF tests whether the FORMAT statement has been translated. If not, CHCIF calls CEKBF at entry point SYSPFMT. CEKBF enters translations of FORMAT codes into a FORMAT table, as shown in Table 11.

CHCIF scans the translated FORMAT string, tests each entry for validity, and performs these actions:

- Initialization of repetition factors both inside and outside the FORMAT parentheses
- Immediate input or output of character strings or spaces (wH or w or Tw or '..')
- Initialization of scale factors before FORMAT control units
- Immediate input or output of logical records
- Initialization of return points to be used when closing parentheses are reached
- Immediate conversion of input or output data in accordance with the FORMAT conversion code

In the last case, if an item is known to be available, the appropriate conversion routine is called and return is made to the calling routine. If no item is available, the scan continues.

On entries for subsequent list items in the I/O request, CHCIF keeps track of its position in the FORMAT table.

The scan ends when a closing parenthesis is reached.

CHCIH -- Integer Output Conversion

This routine converts a two-byte or four-byte binary list item to an integer field in the output buffer, according to the format In, where n is the integer field's size. (See Chart IH.)

Entry: The entry point is CHCIH1. The parameter list is described under "Data Conversion Routine Parameter Lists," in this section.

Routine Called: PRMPT (CZATJ1)

Error Checks: If the output buffer area is too small to contain the integer field, the field is filled with asterisks and a message is issued by PRMPT.

Operation: The contents of the list item are converted in a work area to packed decimal, then into zoned decimal. The sign, if negative, is set in the conventional FORTRAN position. The work area is moved into the output buffer field from right to left, to check whether sufficient space is supplied by the field width. Blanks are inserted as padding if the buffer size is larger than the converted field size.

CHCII -- Real and Integer Input Conversion

This routine converts a data field in an input buffer to the appropriate type of list item. An integer field is converted to a binary list item. A real field is converted to a single- or double-precision, floating-point list item. The integer field has the format In, where n is the field width. The real field has the format Fw.d, Ew.d, or Dw.d, where w is the field

Table 11. Translation of FORMAT codes

FORMAT CODE	FORMAT TABLE ENTRY						
	SIZE (BYTES)	BYTE 0	BYTE 1	BYTE 2	BYTE 3	BYTE 4	BYTE 5
H	STRING LENGTH+2	X'0'	LENGTH	CHARACTER STRING AS MANY BYTES AS NEEDED (MAX 255)			
/	1	X'1'					
X	2	X'2'	REPEAT COUNT				
T	2	X'3'	W				
P	2	X'4'	SCALE FACTOR				
)	2	X'5'	NEST LEVEL				
(	3	X'6'	NEST LEVEL	REPEAT COUNT			
A	3	X'7'	REPEAT COUNT	W-1			
Z	3	X'8'	REPEAT COUNT	W-1			
L	3	X'9'	REPEAT COUNT	W-1			
I	3	X'A'	REPEAT COUNT	W-1			
G	4	X'B'	REPEAT COUNT	W-1	D		
F	4	X'C'	REPEAT COUNT	W-1	D		
D	4	X'D'	REPEAT COUNT	W-1	D		
E	4	X'E'	REPEAT COUNT	W-1	D		
SPECIAL H	6	X'F'	LENGTH	ADDRESS OF CHARACTER STRING			

width and d is the width of the decimal fraction. (See Chart II.)

Entry: There are three entry points: CHCII, CHCIK, and CHCIG.

Routine Called: PRMPT (CZATJ1)

Error Checks: If the FORMAT specification (F, E, D, or I) is improperly specified or the data field is greater than the permissible range, PRMPT is called.

Operation: The input data field is scanned from left to right to determine:

1. The value of the sign, if any.
2. Position of the base number's first digit.
3. Number of digits before the decimal point, if real.
4. Position of first digit after decimal point, if real.
5. Number of digits after decimal point, if real.
6. The letter E or D, if present.
7. Sign preceding exponent, if any.
8. Position of exponent's first digit.
9. Number of exponent digits.

During the scan a test is made for invalid characters in the numeric fields. In addition, a test is made to determine if the integer or real data exceeds the permissible maximum. If such errors are encountered, the scan is terminated and control is passed to PRMPT to issue an error message. Otherwise, conversion continues and a normal return is made.

CHCIJ -- Real Output Conversion

This routine converts a single- or double-precision floating-point list item to a real field in the output buffer. The real field has a format of either Ew.d, Dw.d, or Fw.d, where w is the field width and d is the size, in digit positions, of the fractional position. (See Chart IJ.)

Entry: There are two entry points: CHCIJ1 and CHCIL1.

Routines Called: Return is made to the calling routine.

Error Checks: If the output buffer is too small to contain the real field, the real field is filled with asterisks.

Operation: A test is made against a table of powers of ten to determine the required output exponent, if any, and the item is divided by the appropriate power of ten. If there is an exponent, it is moved into the output buffer from right to left, followed by its sign and the letter E or D. If there is no exponent, the data field, after processing is complete, is moved into the output buffer and control is passed to the calling routine.

CHCIM -- Complex Input Conversion

This routine converts a complex data field in the input buffer to a complex list item consisting of two real data fields. Each real field is converted to a single- or double-precision floating-point list item according to the format Fw.d, Ew.d, or Dw.d, where w is the real field width and d is the width of the decimal fraction. (See Chart IM.)

Entry: The entry point is CHCIM1.

Routines Called:

- Real and Integer Input Conversion (CHCII)
- PRMPT (CZATJ1)

Error Checks: If the complex data field in the input buffer contains no real field or only one real field, or if there is a missing parenthesis or central comma, CHCIM issues an error message via PRMPT. No further action is taken and the list items remain unchanged. If either or both real fields contain invalid characters or exceed the permissible magnitude, CHCII issues an error message.

Operation: The input buffer is scanned to find the size of each real data field. Once it is determined that the user properly specified these data fields and included the correct delimiters (parentheses and central comma), the first real data field is passed as a parameter to CHCII for conversion to a single- or double-precision floating-point list item. When CHCII returns control to CHCIM, CHCIM passes the second real data field to CHCII for conversion, completing the complex input conversion.

CHCIN -- Complex Output Conversion

This routine converts a complex list item consisting of two single- or double-precision floating-point items to a complex data field in an output buffer. Each floating-point list item is converted to a real data field according to the format code Fw.d, Ew.d, Dw.d, or Gw.s, where w is the real field width, d is the width of the decimal fraction, and s is the number of significant digits. (See Chart IN.)

Entry: The entry point is CHCIN1.

Routine Called: General Output Conversion (CHCIT)

Error Check: If the FORMAT specification (F, E, D, or G) is improperly given or the real data field is greater than the permissible range, CHCIT issues an error message.

Operation: CHCIN passes the first list item as a parameter to CHCIT for conversion to a real data field. When CHCIT completes the conversion process and returns to CHCIN, CHCIN passes the second list item to CHCIT for conversion, completing the complex output conversion.

CHCIO -- Alphameric and Hexadecimal Input Conversion

This routine transfers a specified number of bytes (alphameric or hexadecimal characters) from an input buffer area to a list item. The format is Aw (alphameric) or Zw (hexadecimal), where w, field width, is the number of characters being transferred. (See Chart IO.)

Entry: The entry points are CHCIO1 (alphameric data) and CHCIO2 (hexadecimal data).

Operation: When the conversion format is alphameric, this occurs:

- If the byte size of the list item equals the number of alphameric characters in the buffer, a simple move is executed, with no data test made.
- If the byte size of the list item is less than the number of alphameric characters in the buffer, the excess, left-most characters are truncated in the list item field.
- If the byte size of the list item is larger than the number of alphameric characters in the buffer, the characters are transferred into the list item field, left-justified.

When the conversion format is hexadecimal, the field is scanned for valid hexadecimal characters. If invalid, a call is made to PRMPT. If valid, the characters are converted to binary and moved to the input item. If the input field was preceded by a minus sign, the result is complemented. In either case, a return is made to the calling routine.

CHCIP -- Alphameric and Hexadecimal Output Conversion

This routine transfers a list item containing a specified number of bytes (alphameric or hexadecimal characters) to an output buffer area. The format is A*w* (alphameric) or Z*w* (hexadecimal), where *w*, field width, is the number of characters being transferred. (See Chart IP.)

Entry: The entry points are CHCIP1 (alphameric data) and CHCIP2 (hexadecimal data).

Operation: When the list item contains alphameric data, this occurs:

- If the byte size of the list item equals the size of the buffer area, a simple move is executed, with no data test made.
- If the byte size of the list item is less than the size of the buffer area, the list item is entered right-justified and padded to the left with blanks.
- If the byte size of the list item is greater than the size of the buffer, the list item is transferred to the buffer area left-justified with excess characters truncated.

When the list item contains hexadecimal data, it is unpacked and translated to hexadecimal digits in the output buffer.

CHCIQ -- Logical Input Conversion

This routine converts a logical field in the input buffer area. The field has the format L*w*, where *w* is the field width. (See Chart IQ.)

Entry: The entry point is CHCIQ1.

Operation: The list item is set to an internal value of .FALSE. (binary zero). The logical data field is then scanned until either an F or a T is encountered. In the first case, immediate return occurs. In the second case, the list item is set to an internal value of .TRUE. (binary one) before returning.

CHCIR -- Logical Output Conversion

This routine converts a list item to a logical field in the output buffer area. The field has the format L*w*, where *w* is the field width. (See Chart IR.)

Entry: The entry point is CHCIR1.

Operation: The output buffer area is first filled with blanks. If the list item contains a value of zero, the letter F is placed in the last buffer area position; otherwise, the letter T is inserted.

CHCIS -- General Input Conversion

This routine converts a data field in the input buffer to a list item according to the format G*w*.*s*, where *w* is the field width and *s* is an optional specification of the number of significant digits. (See Chart IS.)

Entry: The entry point is CHCIS1.

Routines Called:

- Real and Integer Input Conversion (CHCII)
- Logical Input Conversion (CHCIQ)
- Alphameric and Hexadecimal Input Conversion (CHCIO)

Error Checks: CHCIS does no error checking. Error checks, if any, are made by the called data conversion routines.

Operation: If the conversion type is 'N', indicating a call from the NAMELIST Processor (CHCID), this happens:

* If the data field begins with a quote, a literal character string is moved to the list item.

* If the data field begins with a 'Z', hexadecimal conversion is performed.

If the conversion type is not 'N', the list item is checked to determine which conversion routine should be called:

Type of Data	Routine Called
Integer	CHCII (Entry point CHCIG)
Real	CHCII (Entry point CHCIK)
Logical	CHCIQ (Entry point CHCIQ1)
Alphameric	CHCIO (Entry point CHCIO1)
Hexadecimal	CHCIO (Entry point CHCIO2)

CHCIT -- General Output Conversion

This routine converts a list item to a data field in the output buffer, according to the format Gw.s; where w is the field width and s is an optional specification of the number of significant digits. (See Chart IT.)

Entry: The entry point is CHCIT1.

Routines Called:

* Integer Output Conversion (CHCIH)

* Real Output Conversion (CHCIJ)

* Logical Output Conversion (CHCIR)

Error Checks: CHCIT performs no error checks. Discrepancies between the size and type specification of the list item and the data field are detected by the called conversion routine.

Operation: CHCIT examines the list item type. If logical, a call is made to CHCIR. If integer, a call is made to CHCIH. If real, the magnitude of the item is tested. Should it fall in the range 0.1 to 10**s, a call is made to CHCIJ using F format specification. If real, and outside this range, a call is made to CHCIJ using E or D format specification.

CHCIU -- List Termination

This routine terminates list processing for a READ, WRITE, PRINT, or PUNCH statement, and completes any I/O operation that is pending. (See Chart IU.)

Entry: The entry point is CHCIU1.

Routines Called:

* Format Processor (CHCIF)

* I/O Control (CHCIC)

The final return is made with registers unchanged, except that register 13 is set to the address of the calling module's PSECT and register 15 is set to zero.

Operation: If the previous list processing was an unformatted READ, no further action is necessary, and control returns to the caller. If the previous action was a formatted READ, CHCIF is entered with a zero list item to indicate termination of the FORMAT scan. This permits scanning for Hollerith format controls that have still not been processed, or skipping records until the end of the format character string.

If the previous action was an unformatted WRITE, a final call is made to CHCIC to write the complete logical record.

If the previous action was a formatted WRITE, CHCIF is entered as above, followed by CHCIC.

CHCIW -- Exit

CHCIW is described in "Section 4: Service Subprogram Descriptions."

CHCBD -- I/O Interruption and Machine Indicator Routine

At the beginning of all FORTRAN main programs, the compiler generates code that calls this routine at entry point CHCBD1.

Entry: The entry point is CHCBD1.

Operation: This routine sets bits in the PSW such that the fixed-point overflow and significance exceptions will be ignored, and directs the system interruption handler as to where to pass control if any of the following exceptions occur:

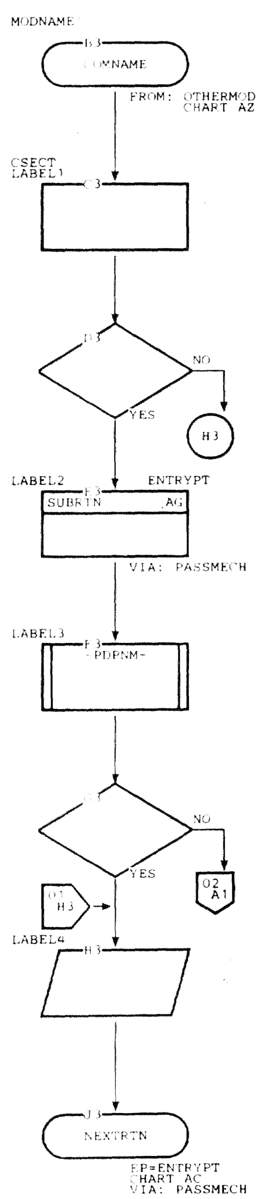
Exception	Subprogram
Specification	CHCBE2
Exponent Overflow	CHCBE3
Exponent Underflow	CHCBE4
Divide Check	CHCBE5

In addition, this routine initializes the machine indicator flags and the sense light simulators, and clears any pointers to entries in the DCB table. It then returns control to the calling program.

The flowcharts in this manual have been produced by an IBM program, using ANSI symbols. The symbols are defined in the left column below, and examples of their use are shown at the right.

SYMBOL	DEFINITION
B1 TERMINAL BLOCK	INDICATES AN ENTRY OR TERMINAL POINT IN A FLOWCHART; SHOWS START, STOP, HALT, DELAY, OR INTERRUPTION. MAY ALSO INDICATE RETURN TO THE CALLING PROGRAM.
C1 PROCESS BLOCK	INDICATES A PROCESSING FUNCTION OR A DEFINED OPERATION CAUSING CHANGE IN VALUE, FORM OR LOCATION OF INFORMATION.
D1 DECISION BLOCK	INDICATES A DECISION OR SWITCHING-TYPE OPERATION THAT DETERMINES WHICH OF A NUMBER OF ALTERNATE PATHS SHOULD BE FOLLOWED.
E1 SUBROUTINE BLOCK	INDICATES A SUBROUTINE OR MODULE THAT IS DESCRIBED IN THIS MANUAL.
F1 PREDEFINED PROCESS BLOCK	INDICATES A SUBROUTINE OR MODULE THAT IS INCLUDED IN THE FLOWCHARTS OF ANOTHER MANUAL.
G1 INPUT/OUTPUT BLOCK	INDICATES GENERAL I/O FUNCTIONS, SUCH AS GET, PUT, READ, WRITE, SIO, AND DEVICE-CONTROL MACRO INSTRUCTIONS.
H1 PREPARATION BLOCK	INDICATES A PROCESS THAT CHANGES SYSTEM OPERATION, FOR EXAMPLE, SETS A SWITCH, MODIFIES AN INDEX REGISTER, OR INITIALIZES A ROUTINE.
ONPAGE CONNECTOR	INDICATES ENTRY TO OR EXIT FROM ANOTHER BLOCK ON THE SAME FLOWCHART PAGE.
OFFPAGE CONNECTOR	INDICATES ENTRY TO OR EXIT FROM A BLOCK ON ANOTHER PAGE OF THE SAME SET OF FLOWCHARTS.

EXAMPLE



COMMENTS

B3: MODNAME IS THE LOAD MODULE OR LIBRARY NAME OF THE ROUTINE DESCRIBED BY THIS FLOWCHART.

COMNAME IS THE COMMON NAME OF THE ROUTINE.

OTHERMOD INDICATES THE MODULES PASSING CONTROL TO THIS MODULE AND THEIR FLOWCHARTS.

C3: CSECT IS THE CSECT NAME OR OTHER ENTRY POINT AT WHICH PROCESSING BEGINS.

LABEL1 IS THE LABEL OF THE FIRST INSTRUCTION.

D3: PROGRAM EXECUTION CONTINUES WITH BLOCK H3 WHEN THE DECISION IS NO, OR BLOCK E3 WHEN THE DECISON IS YES.

E3: LABEL2 IS THE LABEL OF THE SECTION OF CODE IN THIS ROUTINE FROM WHICH CONTROL IS PASSED TO THE SUBROUTINE. CONTROL RETURNS TO THE NEXT INSTRUCTION FOLLOWING THE SUBROUTINE CALL.

ENTRYPT IS THE ENTRY POINT.

SUBRTN IS THE COMMON NAME OF THE SUBROUTINE IN FLOWCHART AG.

VIA: PASSMECH INDICATES HOW CONTROL PASSES FROM COMNAME TO SUBRTN.

F3: LABEL3 IS THE LABEL OF THE SECTION OF CODE FROM WHICH CONTROL IS PASSED TO THE PREDEFINED PROCESS PDPNM, WHICH IS DOCUMENTED IN ANOTHER PUBLICATION (-PDPNM- MAY ALSO BE USED IN A PROCESSING BLOCK).

G3: EXECUTION CONTINUES WITH BLOCK H3 WHEN THE DECISION IS YES, OR WITH BLOCK A1 ON PAGE 2 OF THIS SET OF FLOWCHARTS WHEN THE DECISION IS NO.

THE OFFPAGE CONNECTOR MARKED 01H3 INDICATES THAT EXECUTION CONTINUES WITH BLOCK H3 FROM ANOTHER PAGE OF THIS SET OF FLOWCHARTS. THIS CONNECTOR IS ALSO PAIRED WITH THE ONPAGE CONNECTOR FROM BLOCK D3.

H3: LABEL4 IS THE LABEL OF A SECTION OF CODE OF THIS ROUTINE THAT INITIATES I/O.

J3: NEXTRTN IS THE COMMON NAME OF THE ROUTINE THAT EXECUTES AFTER THIS ROUTINE.

ENTRYPT IS THE ENTRY POINT OF NEXTRTN, WHICH IS DESCRIBED IN CHART AC.

VIA: PASSMECH INDICATES HOW CONTROL PASSES FROM COMNAME TO NEXTRTN.

Program Logic Manual

GY28-2020-2

FORTRAN IV Library Subprograms

Flowcharts on pages 43-88 were not scanned.

This appendix describes the assumptions that the FORTRAN I/O library makes in initializing DCBs with information concerning record format (RECFM) and data set organization (DSORG). These assumptions are described in this appendix to help reduce a frequent source of error encountered by the user when performing I/O.

Introductory material is presented on the DCB describing its general use, contents, and sources of initialization, before discussing the permissible record formats and data set organizations. (For a description of the DCB in terms of its size, format, and use by the rest of TSS/360, see System Control Blocks PLM.)

DCB Use

The Data Control Block (DCB) is created by DCB Management (CHCIB) and is used by certain data management routines which are invoked by macro instruction references in I/O Control (CHCIC). The DCB is required for all I/O performed using either BSAM or VAM. However, the DCB is not required for I/O when using the GATE macro instructions, even though CHCIB reserves space for one. In this case, the DDNAME field is used to save the name that the data set would have had if GATE were not used. The principal reason for constructing this dummy DCB for GATE is to retain consistency for various routines when handling data set reference numbers with no corresponding DDEF commands.

DCB Content

The DCB contains information such as the DDNAME, type of data set organization, the type and size of records, block size for blocked data sets, number of buffer areas, exits for SYNAD and EODAD, and various control flags used by data management.

DCB Initialization

The FORTRAN I/O routines, when processing an input data set, take advantage of information in the DCB to adapt to the characteristics of the data set and read it correctly. Characteristics are based on the parameters for a DCB that can be supplied from:

- The user program -- type of I/O used and associated data format.
- User-supplied DDEF commands -- some of the information in the DCB can be changed in this manner; however, the extent of change is limited.
- Input data set labels -- these override both of the above sources of information, within limits set by data management.

Combinations of DSORG and RECFM

Table 12 gives the permissible combinations of record formats and data set organizations that may be specified when using the FORTRAN I/O library.

Table 12. Combinations of DSORG and RECFM values

RECFM	DSORG VALUES				
	VS	PS	VSP	VI	VIP
V	A	A	A	A	A
VB	N	A	N	N	N
VT	N	A	N	N	N
F	A	A	A	A	A
FB	N	A	N	N	N
FS	N	A	N	N	N
FT	N	A	N	N	N
FBS	N	A	N	N	N
FBT	N	A	N	N	N
FBST	N	A	N	N	N
FST	N	A	N	N	N
U	L	A	L	N	N

Codes mean:
A - Acceptable
L - Limited acceptable
N - Not acceptable

DSORG abbreviations mean:
VS - Virtual sequential (direct access only)
PS - Physical sequential (any device except terminals)
VSP - Virtual sequential partitioned (direct access only)
VI - Virtual index sequential (direct access only)
VIP - Virtual index sequential partitioned (direct access only)

RECFM abbreviations mean:
V - Variable-length unblocked records
VB - Variable-length blocked records
VT - Variable-length unblocked with track overflow
F - Fixed-length unblocked records
FB - Fixed-length blocked records
FS - Same as F, no truncated blocks or unfilled tracks
FT - Same as F, track overflow
FBS - Same as FB, no truncated blocks or unfilled tracks
FBT - Same as FB, track overflow
FBST- Same as FBS, track overflow
FST - Same as F, no truncated blocks, track overflow
U - Undefined record length

Any of the RECFM codes shown can be followed by the letter A or the letter M. A indicates that the first character of every logical record is a FORTRAN carriage or punch control character. M indicates that the first character of every record is a TSS/360 machine control byte. In general, the M option cannot be used by FORTRAN output data sets, since the control codes are unprintable and do not conform to FORTRAN conventions.

Unformatted FORTRAN Logical Records

Under any of the organization techniques used, an unformatted WRITE statement may lead to a logical record that exceeds the length of the maximum record supported by the access method. Furthermore, it is not possible to enter the byte size of the entire FORTRAN logical record into the beginning of the I/O physical record without the possibility of tying up an indefinite amount of virtual storage. Therefore, unformatted FORTRAN logical records may span over data management physical records. In the management of unformatted FORTRAN data, the first two bits of every VS physical record or the third byte of every PS physical record is a control byte defined as follows:

X'00'	A FORTRAN logical record does not span into or out of the data management physical record.
X'01'	This data management physical record is the first of a span.
X'02'	This data management physical record is the last of a span.
X'03'	This data management physical record is within the range of a span.

No data management logical record will be written containing more than one unformatted FORTRAN logical record.

All FORTRAN library routines have five-letter names beginning with the letters 'CHC'. The names of mathematical library routines begin with 'CHCA' or 'CHCB', and the names of I/O library routines begin with 'CHCI' except one -- CHCBD. All other external names consist of the routine names with suffixes.

Table 13. External names of FORTRAN IV library subprograms

	Mathematical Subprograms	Service Subprograms	I/O Subprograms
Entry Name	See Tables 2, 3, and 7.	See Tables 7 and 9, and Section 4.	See Section 5.
Routine Name	See Tables 2, 3, and 7.	See Tables 7 and 9, and Section 4.	See Section 5.
CSECT Name	Routine name suffixed by 'W'.	CHCBE: Routine name suffixed by 'W'. CHCIV and CHCIW: Routine name suffixed by 'C'.	Routine name suffixed by 'C'.
PSECT Name	Routine name suffixed by 'R'.	CHCBE: Routine name suffixed by 'R'. CHCIV and CHCIW: Routine name suffixed by 'W'.	Routine name suffixed by 'W'.

INDEX

Where more than one page reference is given, the major reference is first.