IBM

Application Program

GH20-0240-3

System/360 Continuous System Modeling Program (360A-CX-16X) Application Description

This is an IBM System/360 program for the simulation of continuous systems. It provides an application-oriented input language that accepts problems expressed in the form of either an analog block diagram or a system of ordinary differential equations. Data input and output are facilitated by means of application-oriented control statements.

This manual contains a general description of the application, the machine configuration, a general systems chart, and a sample problem. More detailed information on the simulation language and its use is provided in the User's Manual (GH20-0367).

CONTENTS

Fourth Edition (March 1972)

This is a major revision of, and supersedes, GH20-0240-2.

Changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

This edition applies to Version 1, Modification Level 3 of the System/360 Continuous System Modeling Program and to all subsequent versions and modifications until otherwise indicated in new editions or Technical Newsletters. Changes are continually made to the specifications herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/360 and System/370 Bibliography, Order No. GA22-6822, and its associated SRL Newsletter, Order No. GN20-0360, for the editions that are applicable and current.

Copies of this and other IBM publications can be obtained through IBM branch offices. Address comments concerning the contents of this publication to IBM Corporation, Technical Publications Department, 1133 Westchester Avenue, White Plains, New York 10604. Comments and suggestions become the property of IBM.

INTRODUCTION

Simulation is a well established tool for investigating phenomena ranging from information flow in business organizations to the dynamic behavior of complex mechanical systems. The former have often been treated as discrete processes on digital computers through the use of such discrete system simulation programs as the General Purpose Simulation System (GPSS). By contrast, those continuous dynamic systems that are the usual concern of engineers and scientists have traditionally been simulated on analog computers. However, as the systems under investigation have become more and more complex, and the need for accuracy and flexibility has increased, interest has grown in the application of digital computers to continuous system simulation.

One typical application might be a control engineer's study of the effectiveness of various control system designs by simulation of both the process to be controlled and the several control mechanisms. Other examples might be a physiologist's simulation of a model of the cardiovascular system, and a mechanical engineer's investigation of the effects of damping and backlash in a proposed mechanical device.

It has been recognized, however, that many engineers and scientists working in these areas had no desire to learn digital computer programming. The need for a problem-oriented program designed to help prepare problems for solution on large-scale digital machines was clearly indicated.

The System/360 Continuous System Modeling Program (S/360 CSMP) is intended to help satisfy that need by allowing problems to be prepared directly and simply from either a block-diagram representation or a set of ordinary differential equations. The program provides a basic set of functional blocks with which the components of a continuous system may be represented, and it accepts application-oriented statements for defining the connections between these functional blocks. S/360 CSMP also accepts S/360 FORTRAN IV statements, thereby allowing the user to readily handle nonlinear and time-variant problems of considerable complexity. Input and output are facilitated by means of user-oriented control statements. A fixed format is provided for printing (tabular format) and print-plotting (graphic format) at selected increments of the independent variable. Convenient means are available for terminating a simulation run with a sequence of computations and logical tests, and thereby accomplishing iterative simulations of the type required for parameter optimization studies. Through these features S/360 CSMP permits the user to concentrate upon the phenomenon being simulated, rather than the mechanism for implementing the simulation.

This program is based on the Digital Simulation Language (DSL/90).

GENERAL DESCRIPTION OF THE APPLICATION

S/360 CSMP is a "continuous system simulator" which combines the functional block modeling feature of such "digital analog simulators" as 1130 CSMP II with a powerful algebraic and logical modeling capability. The input language enables a user to prepare structure statements describing a physical system, starting from either a block diagram or a differential equation representation of that system. The program provides a basic set of functional blocks plus means for the user to define functions specially suited to his particular simulation requirements. Application-oriented input statements are used to describe the connections between these functional blocks. S/360 CSMP also accepts FORTRAN statements, thereby allowing the user to readily handle complex nonlinear and time-variant problems. A translator converts these structure statements into a FORTRAN subroutine, which is then compiled and executed alternately with a selected integration routine to accomplish the simulation. Figure 1 shows the general form and application of the S/360 CSMP functions.

Input and output are simplified by means of a free format for data entry and user-oriented input and output control statements. Data and control statements may be entered in any order and may be intermixed with structure statements. Output options include printing of variables in standard tabular format, print-plotting in graphic form, and preparation of a data set for user-prepared XY plotting programs.

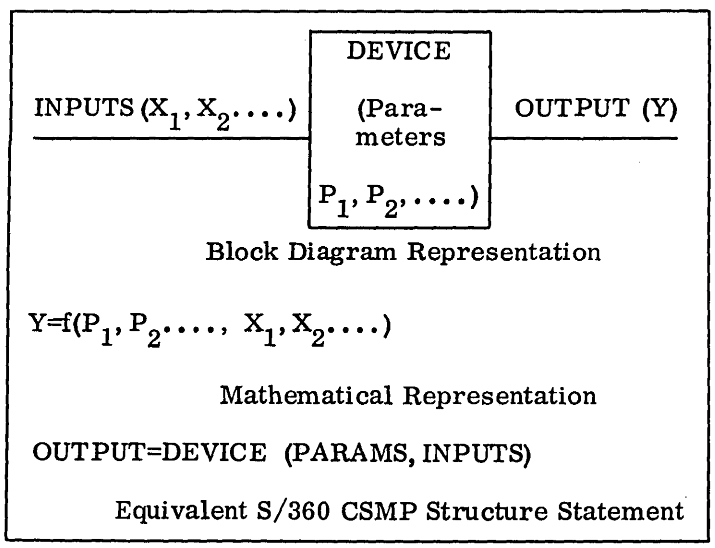


Figure 1. Illustration of S/360 CSMP functional blocks

Two important features of S/360 CSMP are statement sequencing and a choice of integration methods. With few exceptions, structure statements may be written in any order and, at the user's option, may be automatically sorted by the system to establish the correct information flow. Centralized integration is used to ensure that all integrator outputs are computed simultaneously at the end of the iteration cycle. A choice may be made between the fifth-order Milne predictor-corrector, fourth-order Runge-Kutta, Simpson's, second-order Adam's, trapezoidal, and rectangular integration methods. The first two methods allow the integration interval to be adjusted by the system to meet a specified error criterion.

Several simple methods are available for automatically obtaining a prescribed sequence of runs.

The entire S/360 CSMP simulation can also be conveniently controlled by a user-supplied sequence of FORTRAN statements, which are performed only at the termination of a simulation run. This provides a simple and efficient method for handling the type of iterative computation involved, for example, in automatic search procedures for parameter optimization, or solution of two-point boundary value problems.

S/360 CSMP permits the user to concentrate on the phenomenon being studied rather than the mechanisms of digital computer programming.

Extent of Coverage

S/360 CSMP provides a complement of 34 functional blocks (also called functions) for modeling a continuous system. These functions include such conventional analog computer components as integrators and relays plus many special purpose functions such as delay time, zero-order hold, dead space, and limiter functions. This complement is augmented by the FORTRAN library functions such as cosine, tangent, and absolute value. In addition, the user can define functional blocks specially suited to his own application area. This definition can be accomplished either through FORTRAN programming or, more simply, through a macro capability that permits individual existing functions to be combined into a larger functional block. By combining these functional blocks with FORTRAN algebraic and logical statements, the user may handle very complex non-linear and time-variant problems.

When the S/360 CSMP simulation is treated as a subroutine, or when the termination option is used, a user-written FORTRAN program can test run responses, define run control conditions, and supervise both input and output of information.

A fixed format for data output at selected increments of the independent variable is provided for all output options, thereby freeing the user from the details of formatting.

Advantages

The System/360 Continuous System Modeling Program provides the following advantages:

- The input language is nonprocedural, application-oriented, and free-form.
- A problem can be prepared directly from either a functional block diagram or a system of ordinary differential equations.
- With few exceptions FORTRAN statements can be intermixed with the S/360 CSMP simulation statements.
- The method of integration can be chosen from several standard options provided in the program.
- The output is provided automatically in a fixed format for all output options.
- Thirty-four standard functional blocks are provided; in addition, the user can add his own functions to the library.
- Simulation runs are completely repeatable; problem decks may be conveniently stored for future reruns.
- The entire S/360 CSMP simulation may be controlled by a sequence of conventional FORTRAN statements.

Application-Oriented Concepts

Designed for use specifically by the engineer or scientist, S/360 CSMP requires only a minimum knowledge of computer programming and operation. Simplicity and flexibility are salient characteristics of the input language. A knowledge of basic FORTRAN is helpful but not necessary.

The user is not burdened with operational details, because a translator automatically sorts the input statements into correct sequence, converts them into a FORTRAN subroutine, and then automatically compiles and executes the program. The translation and execution functions operate independently under standard Operating System control — in effect, as one continuous, single-pass system.

A simulation problem is programmed for solution by preparing the following three types of statements:

1. Structure statements. These describe the functional relationships between the variables of the model; taken together, they define the network to be simulated.
2. Data statements. These assign numerical values to the parameters, initial conditions, and table entries associated with the problem.

3. Control statements. These specify options relating to the translation, execution, and output phases of the S/360 CSMP program, such as run time, integration interval, and output variables to be printed.

Four elements constitute these S/360 CSMP language statements:

1. Constants. These are unchanging quantities used in their numerical form in the source statements.

2. Variable names (also called variables). These are symbolic representations of quantities that may either change during a run or be changed, under program control, between successive runs of the same model structure. Some examples are DIST, RATE2, and MASS.

3. Operators. These are used, instead of functional blocks, to indicate the basic arithmetic functions or relationships. As in FORTRAN, these operators are:

Symbol	Function
+	addition
-	subtraction
*	multiplication
/	division
**	exponentiation
( )	grouping of variables and/or constants
=	replacement

Some illustrations of the use of operators to construct structure statements are:

```
DIST=RATE*TIME
    Y=A*X**2+B
    A=(B*C)+(D*E)
```

4. Functional blocks (functions). These perform the more complex mathematical operations such as integration, time delay, quantization, and limiting. An example of their use is:

```
Y=INTGRL(10.0,X)
```

which states that the variable Y is equal to the integral of the variable X, with the initial condition that Y at time zero is equal to the constant quantity 10.0.

The basic S/360 CSMP library includes all the standard functions found in analog computers plus a complement of special purpose functions often encountered in simulation problems. Table 1 contains the library of functional blocks. Note again that the user can add any desired function to this library. In addition, all the functions available in the standard FORTRAN library in the user's system can be treated as blocks in the S/360 CSMP complement. Illustrations of the most useful functions are shown in Table 2 in their block notation.

MATHEMATICAL FUNCTIONS

GENERAL FORM	FUNCTION
Y = INTGRL (IC, X) Y (0) = IC INTEGRATOR	$Y = \int_0^t X\,dt + IC$ EQUIVALENT LAPLACE TRANSFORM: $\frac{1}{S}$
Y = DERIV (IC, X) $\dot{X}$ (t = 0) = IC DERIVATIVE	$Y = \frac{dX}{dt}$ EQUIVALENT LAPLACE TRANSFORM: S
Y = DELAY (N, P, X) P = DELAY TIME N = NUMBER OF POINTS SAMPLED IN INTERVAL P (INTEGER CONSTANT) DEAD TIME (DELAY)	$Y(t) = X(t - P) \quad t \geq P$ $Y = 0 \quad t < P$ EQUIVALENT LAPLACE TRANSFORM: e^{-PS}
Y = ZHOLD (X_1, X_2) ZERO-ORDER HOLD	$Y = X_2 \quad X_1 > 0$ Y = LAST OUTPUT $\quad X_1 \leq 0$ Y (0) = 0 EQUIVALENT LAPLACE TRANSFORM: $\frac{1}{S}(1 - e^{St})$
Y = IMPL (IC, P, FOFY) IC = FIRST GUESS P = ERROR BOUND FOFY = OUTPUT NAME OF LAST STATEMENT IN ALGEBRAIC LOOP DEFINITION IMPLICIT FUNCTION	Y = FUNCT (Y) $\lvert Y - FUNCT(Y) \rvert \leq P \lvert Y \rvert$

Table 1. Library of S/360 CSMP functional blocks

SYSTEM MACROS

GENERAL FORM	FUNCTION
Y = MODINT (IC, X_1, X_2, X_3) MODE-CONTROLLED INTEGRATOR	$Y = \int_0^t X_3\,dt + IC$ $\quad X_1 > 0$, any X_2 Y = IC $\quad X_1 \le 0,\ X_2 > 0$ Y = LAST OUTPUT $\quad X_1 \le 0,\ X_2 \le 0$
Y = REALPL (IC, P, X) Y (0) = IC 1ST ORDER LAG (REAL POLE)	$P\dot{Y} + Y = X$ EQUIVALENT LAPLACE TRANSFORM: $\frac{1}{PS + 1}$
Y = LEDLAG (P_1, P_2, X) LEAD-LAG	$P_2 \dot{Y} + Y = P_1 \dot{X} + X$ EQUIVALENT LAPLACE TRANSFORM: $\frac{P_1 S + 1}{P_2 S + 1}$
Y = CMPXPL (IC_1, IC_2, P_1, P_2, X) Y (0) = IC_1 $\dot{Y}$ (0) = IC_2 2ND ORDER LAG (COMPLEX POLE)	$\ddot{Y} + 2P_1P_2\dot{Y} + P_2^2 Y = X$ EQUIVALENT LAPLACE TRANSFORM: $\frac{1}{S^2 + 2P_1P_2S + P_2^2}$

Table 1. (continued)

SWITCHING FUNCTIONS

GENERAL FORM	FUNCTION		
$Y = FCNSW(X_1, X_2, X_3, X_4)$	$Y = X_2$		$X_1 < 0$
	$Y = X_3$		$X_1 = 0$
FUNCTION SWITCH	$Y = X_4$		$X_1 > 0$
$Y = INSW(X_1, X_2, X_3)$	$Y = X_2$		$X_1 < 0$
INPUT SWITCH (RELAY)	$Y = X_3$		$X_1 \geq 0$
$Y_1, Y_2 = OUTSW(X_1, X_2)$	$Y_1 = X_2, Y_2 = 0$		$X_1 < 0$
OUTPUT SWITCH	$Y_1 = 0, Y_2 = X_2$		$X_1 \geq 0$
$Y = COMPAR(X_1, X_2)$	$Y = 0$		$X_1 < X_2$
COMPARATOR	$Y = 1$		$X_1 \geq X_2$
$Y = RST(X_1, X_2, X_3)$	$Y = 0$		$X_1 > 0$
	$Y = 1$		$X_2 > 0, X_1 \leq 0$
	$Y = 0$		$X_3 > 0, Y_{n-1} = 1$
	$Y = 1$	$X_1 \leq 0,$	$X_3 > 0, Y_{n-1} = 0$
	$Y = 0$	$X_2 \leq 0,$	$X_3 \leq 0, Y_{n-1} = 0$
RESETTABLE FLIP-FLOP	$Y = 1$		$X_3 \leq 0, Y_{n-1} = 1$

Table 1. (continued)

FUNCTION GENERATORS

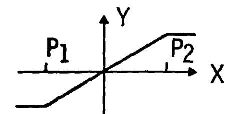
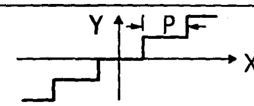
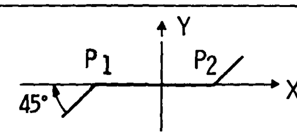
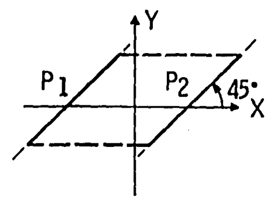
GENERAL FORM	FUNCTION
Y = AFGEN (FUNCT, X) ARBITRARY FUNCTION GENERATOR (LINEAR INTERPOLATION)	Y = FUNCT (X)
Y = NLFGEN (FUNCT, X) ARBITRARY FUNCTION GENERATOR (QUADRATIC INTERPOLATION)	Y = FUNCT (X) $X_0 \le X \le X_n$
Y = LIMIT (P_1, P_2, X) LIMITER	$Y = P_1$ $X<P_1$ $Y = P_2$ $X>P_2$ $Y = X$ $P_1 \le X \le P_2$ 
Y = QNTZR (P, X) QUANTIZER	Y = kP (k - 1/2)P<X≤(k + 1/2)P k = 0, ±1, ±2, ±3... 
Y = DEADSP (P_1, P_2, X) DEAD SPACE	$Y = 0$ $P_1 \le X \le P_2$ $Y = X - P_2$ $X>P_2$ $Y = X - P_1$ $X<P_1$ 
Y = HSTRSS (IC, P_1, P_2, X) Y (0) = IC HYSTERESIS LOOP	$Y = X - P_2$ $(X - X_{n-1})>0$ AND $Y_{n-1} \le (X - P_2)$ $Y = X - P_1$ $(X - X_{n-1})<0$ AND $Y_{n-1} \ge (X - P_1)$ OTHERWISE Y = LAST OUTPUT 

Table 1. (continued)

SIGNAL SOURCES

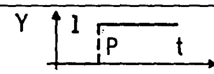
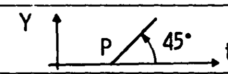
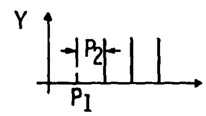
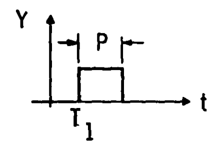
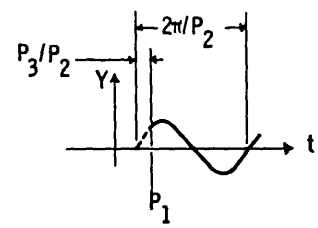
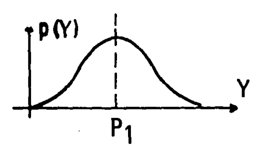
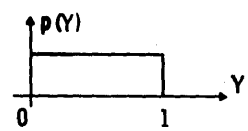
GENERAL FORM	FUNCTION		
Y = STEP (P) STEP FUNCTION	Y = 0 Y = 1	$t<P$ $t \geq P$	
Y = RAMP (P) RAMP FUNCTION	Y = 0 Y = t - P	$t<P$ $t \geq P$	
Y = IMPULS (P_1, P_2) IMPULSE GENERATOR	Y = 0 Y = 1 Y = 0	$t<P_1$ $(t-P_1) = kP_2$ $(t-P_1) \neq kP_2$ $k = 0, 1, 2, 3....$	
Y = PULSE (P, X) P = MINIMUM PULSE WIDTH PULSE GENERATOR (WITH X>0 AS TRIGGER)	Y = 1 Y = 0 T_k = TIME OF TRIGGER	$T_k \leq t < (T_k + P)$ or X>0 OTHERWISE	
Y = SINE (P_1, P_2, P_3) P_1 = DELAY P_2 = FREQUENCY (RADIANS PER UNIT TIME) P_3 = PHASE SHIFT IN RADIANS TRIGONOMETRIC SINE WAVE WITH DELAY, FREQUENCY AND PHASE PARAMETERS	Y = 0 $Y = SIN(P_2(t-P_1)+P_3)$	$t<P_1$ $t \geq P_1$	
Y = GAUSS (N, P_1, P_2) N = ANY ODD INTEGER P_1 = MEAN P_2 = STANDARD DEVIATION NOISE (RANDOM NUMBER) GENERATOR WITH NORMAL DISTRIBUTION	NORMAL DISTRIBUTION OF VARIABLE Y p(Y) = PROBABILITY DENSITY FUNCTION		
Y = RNDGEN (N) N = ANY ODD INTEGER NOISE (RANDOM NUMBER) GENERATOR WITH UNIFORM DISTRIBUTION	UNIFORM DISTRIBUTION OF VARIABLE Y p(Y) = PROBABILITY DENSITY FUNCTION		

Table 1. (continued)

LOGIC FUNCTIONS

GENERAL FORM	FUNCTION	
$Y = AND (X_1, X_2)$	$Y = 1$	$X_1>0,\ X_2>0$
AND	$Y = 0$	OTHERWISE
$Y = NAND (X_1, X_2)$	$Y = 0$	$X_1>0,\ X_2>0$
NOT AND	$Y = 1$	OTHERWISE
$Y = IOR (X_1, X_2)$	$Y = 0$	$X_1\leq 0,\ X_2\leq 0$
INCLUSIVE OR	$Y = 1$	OTHERWISE
$Y = NOR (X_1, X_2)$	$Y = 1$	$X_1\leq 0,\ X_2\leq 0$
NOT OR	$Y = 0$	OTHERWISE
$Y = EOR (X_1, X_2)$	$Y = 1$	$X_1\leq 0,\ X_2>0$
	$Y = 1$	$X_1>0,\ X_2\leq 0$
EXCLUSIVE OR	$Y = 0$	OTHERWISE
$Y = NOT (X)$	$Y = 1$	$X\leq 0$
NOT	$Y = 0$	$X>0$
$Y = EQUIV (X_1, X_2)$	$Y = 1$	$X_1\leq 0,\ X_2\leq 0$
	$Y = 1$	$X_1>0,\ X_2>0$
EQUIVALENT	$Y = 0$	OTHERWISE

Table 1. (continued)

GENERAL FORM	FUNCTION
Y = EXP (X) EXPONENTIAL	$Y = e^X$
Y = ALOG (X) NATURAL LOGORITHM	Y = LN (X)
Y = ALOG10 (X) COMMON LOGORITHM	$Y = LOG_{10}(X)$
Y = ATAN (X) ARCTANGENT	Y = ARCTAN (X)
Y = SIN (X) TRIGONOMETRIC SINE	Y = SIN (X)
Y = COS (X) TRIGONOMETRIC COSINE	Y = COS (X)
Y = SQRT (X) SQUARE ROOT	$Y = X^{1/2}$
Y = TANH (X) HYPERBOLIC TANGENT	Y = TANH (X)
Y = ABS (X) ABSOLUTE VALUE (REAL ARGUMENT AND OUTPUT)	$Y = \|X\|$
Y = IABS (X) ABSOLUTE VALUE (INTEGER ARGUMENT AND OUTPUT)	$Y = \|X\|$

Table 2. FORTRAN functions

GENERAL FORM	FUNCTION
$Y = AMAX0 (X_1, X_2 \ldots X_n)$ LARGEST VALUE (INTEGER ARGUMENTS AND REAL OUTPUT)	$Y = MAX (X_1, X_2 \ldots X_n)$
$Y = AMAX1 (X_1, X_2 \ldots X_n)$ LARGEST VALUE (REAL ARGUMENTS AND OUTPUT)	$Y = MAX (X_1, X_2 \ldots X_n)$
$Y = MAX0 (X_1, X_2 \ldots X_n)$ LARGEST VALUE (INTEGER ARGUMENTS AND OUTPUT)	$Y = MAX (X_1, X_2 \ldots X_n)$
$Y = MAX1 (X_1, X_2 \ldots X_n)$ LARGEST VALUE (REAL ARGUMENTS AND INTEGER OUTPUT)	$Y = MAX (X_1, X_2 \ldots X_n)$
$Y = AMIN0 (X_1, X_2 \ldots X_n)$ SMALLEST VALUE (INTEGER ARGUMENTS AND REAL OUTPUT)	$Y = MIN (X_1, X_2 \ldots X_n)$
$Y = AMIN1 (X_1, X_2 \ldots X_n)$ SMALLEST VALUE (REAL ARGUMENTS AND OUTPUT)	$Y = MIN (X_1, X_2 \ldots X_n)$
$Y = MIN0 (X_1, X_2 \ldots X_n)$ SMALLEST VALUE (INTEGER ARGUMENTS AND OUTPUT)	$Y = MIN (X_1, X_2 \ldots X_n)$
$Y = MIN1 (X_1, X_2 \ldots X_n)$ SMALLEST VALUE (REAL ARGUMENTS AND INTEGER OUTPUT)	$Y = MIN (X_1, X_2 \ldots X_n)$

Table 2. (continued)

The structure statements which use these functional blocks to describe the relationships between variables can be prepared immediately from a block diagram representation or a set of differential equations. This procedure is illustrated by the sample problem discussed later in this manual. The statements can be written in any order and will be automatically sorted for proper sequencing of computations.

Generally, the user would logically first prepare the structure statements and follow these by the data and control statements, in that order. Data statements can be used to assign numeric values to those variables that are to be fixed during a given run. The advantage of assigning variable names and using data statements to specify numeric values is that the latter can be changed, automatically, between successive runs of the same model structure. An example of a data statement is:

PARAMETER RATE=550.0, DIST=1000.0

where PARAMETER is the label identifying the card as a parameter card, RATE and DIST are the variables to be assigned numeric values, and 550.0 and 1000.0 are, respectively, the values assigned. Different types of data can be specified by the following labels:

Label	Type of Data
PARAMETER	Parameters
CONSTANT	Constants
INCON	Initial conditions
FUNCTION	Coordinates of an arbitrary function
TABLE	Entries in a stored array

Lastly, the user prepares control statements to specify certain operations associated with the translation, execution, and output phases of the program. Control statements, like data statements, can be changed between runs under control of the simulation program. An example of a control statement is:

PRINT X, XDOT, X2DOT

where PRINT is a card label specifying that lists of the variables X, XDOT and X2DOT are to be printed.

Examples of the other control labels are:

Label	Control Operation
NOSORT	Specify areas of the structure statements that are not to be sorted

Label	Control Operation
INITIAL	Define a block of computations that is to be executed only at the beginning of
DYNAMIC	the run
TIMER	Specify the print interval, print-plot interval, finish time, and integration interval
FINISH	Specify a condition for termination of the run
RELERR	Specify relative error for the integration routine
ABSERR	Specify absolute error for the integration routine
METHOD	Specify the integration method
PRINT	Identify variables to be printed
PRTPLT	Identify variables to be print-plotted
TITLE	Print page headings for printed output
LABEL	Print page headings for print-plot output
RANGE	Obtain minimum and maximum values of specified variables

Integration Methods

S/360 CSMP uses centralized integration to ensure that all integrator outputs are computed simultaneously. Integration statements are placed at the end of the structure coding by the sorting algorithm, so that all current inputs to integration functions are defined before integration. A single routine is then used to update each of the integrator output variables used in the simulation.

Several different types of routines are available to perform the integration operation. They include both fixed step-size routines and variable integration step-size routines. Five fixed step-size routines are available: fixed Runge-Kutta, Simpson's, trapezoidal, rectangular, and second-order Adams. Two variable step-size routines are available: fifth-order Milne predictor-corrector and fourth-order Runge-Kutta. In these latter routines, the integration interval Δt is automatically varied during problem execution to satisfy the user-specified error criterion.

If none of the above methods satisfies the user's requirement, a dummy integration routine named CENTRL can be used to specify a different integration method. The desired routine is entered into S/360 CSMP by giving it the name CENTRL.

Special Techniques

SORTING

A correct computational sequence is essential to proper solution of a continuous system simulation problem. Proper implementation of numeric methods requires that the output of each statement at time t be computed on the basis of input values at time t. An incorrect computational sequence would update some statement outputs at time t with input values from time $t - \Delta t$. This incorrect sequence would introduce phase lags that could seriously affect stability and accuracy of the solution. S/360 CSMP automatically sorts the user's input statements and thereby relieves him of the sequencing task.

NOSORT OPTION

In some simulations it may be desirable not to sort certain sections of the problem configuration. To answer this requirement, S/360 CSMP provides a NOSORT option, which bypasses the sorting phase for sections of coding identified by a NOSORT label. Thus, the user may include any type of procedural statement capability, such as branching on conditions and logical tests, within a sequence of either S/360 CSMP or FORTRAN statements.

USER-DEFINED FUNCTIONS

In some simulation problems, the mix of functional blocks available from the S/360 CSMP library may not be sufficient to describe conveniently the user's problem. The user, therefore, has been given means for building his own special purpose functional blocks. These functions may range from a few nonlinear statements to an extremely complex model of a complete plant in a process control problem. To define special purpose functional blocks, either S/360 CSMP statements, or FORTRAN, or a combination of both, may be used. Three different types of functions, identified as MACRO, PROCEDURE, and subprogram, may be prepared by the user. These functions differ somewhat in their use and the way in which they are handled by the S/360 CSMP program.

These different methods of building special functions give the user a high degree of flexibility, and enable him to restructure S/360 CSMP into a problem-oriented language for such application areas as chemical kinetics, control system analysis, and biochemistry. In effect, S/360 CSMP does not have to operate within the framework of a digital analog simulator language, but can take on the characteristics of a language oriented to any particular special purpose field of continuous system simulation.

The MACRO type of function-defining capability of S/360 CSMP is a particularly powerful feature of the language. It allows the user to build larger functional blocks from the basic functions available in the library. Once defined, the MACRO may be used any number of times within the simulation structure statements.

The PROCEDURE type of user-defined function allows simple application of the logic capabilities of FORTRAN. During sorting, the statements that define the PROCEDURE are treated as a single functional group, and the entire set is moved around as an entity in order to satisfy the input/output sequencing requirements of the sorting algorithm. There is no internal sorting of statements within a PROCEDURE. Generally, a particular PROCEDURE can be used only once within a simulation. However, the PROCEDURE block can be embedded within a MACRO block and thereby used repeatedly.

The FORTRAN subprogram approach permits the user to go offline from the rest of the S/360 CSMP simulation and prepare a separate subprogram representing the functional characteristics of the phenomenon to be modeled. This approach actually adds little to the algebraic and logical capabilities available through use of the PROCEDURE technique. However, use of the subprogram permits the new block to be permanently added to the system library.

IMPLICIT FUNCTION

The library includes an implicit functional block for solving loops defined by algebraic equations containing no memory element. A memory element is one in which the output depends only on past values of the input and output. This feature, in effect, directs the system to perform a subiteration within the implicit loop at each instant of time until the algebraic relationship has been satisfied. A standard convergence formula is provided for which the user can specify the error criterion. If there is no convergence after 100 iterations, the run is terminated and a diagnostic message is provided.

Precision

The calculations for the S/360 CSMP are done in single-precision, floating-point arithmetic.

MACHINE CONFIGURATION

The program requires a minimum of 102K bytes of storage (excluding that required by OS/360), the Standard Instruction Set, and the Floating-Point Option. In addition to the I/O units needed by the Operating System/360 for FORTRAN IV (Level G) compiling, the program requires three logical utility units. One of these must be a direct access storage device (DASD); the other two may be portions of the required DASD, or may be portions of other DASD's or magnetic tape drives.

SOURCE LANGUAGE

FORTRAN IV (Level G) is used as the source language for approximately 95% of the application package; those operations not readily performed in FORTRAN IV (Level G) are coded as subroutines in System/360 Assembler Language. All routines operate under Operating System/360.

GENERAL SYSTEMS CHART

To the user and operator of the system, the entire run will appear as a single job, even though it is a multiple-step program. Figure 2 shows the I/O configuration.

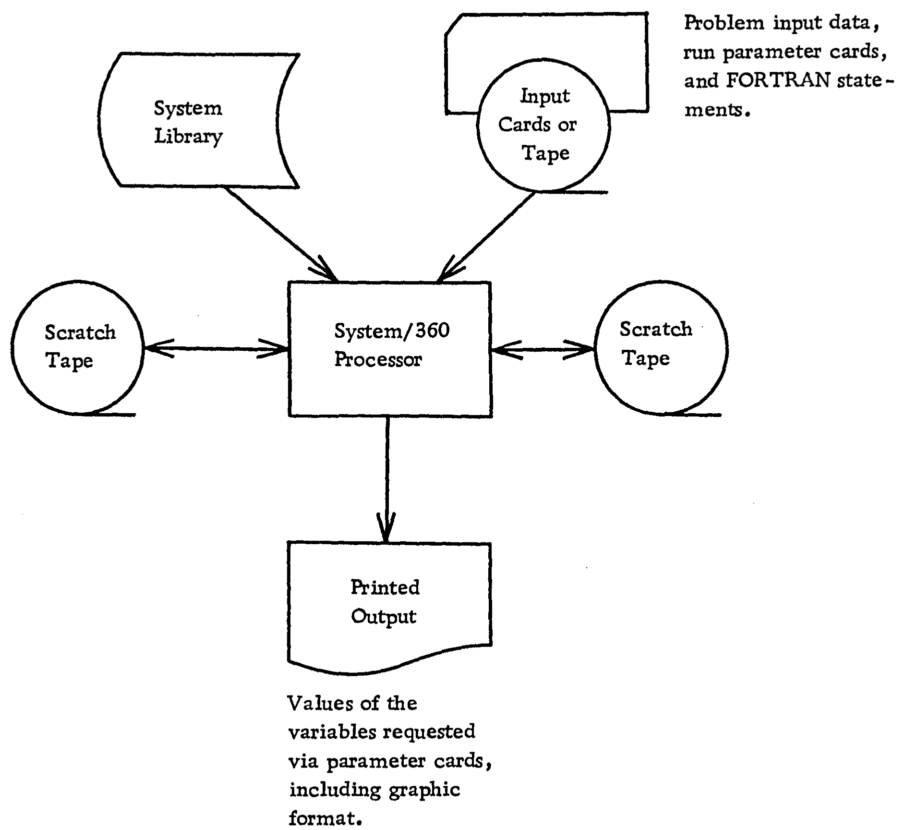


Figure 2. General systems chart

SAMPLE PROBLEM

An illustration of the basic modeling capabilities of S/360 CSMP is provided by a design study of a cable reel system. The objective is to devise an effective controller for a large motor-driven reel that will maintain a constant linear cable velocity as the cable unwinds from the reel. Figure 3 shows a sketch of the physical system.

Control is to be established by measuring the cable velocity, comparing it with a desired or reference signal, and using the error to generate a motor control signal. This is the classical feedback method of control. A tachometer is used to sense the cable velocity and convert that measurement into a corresponding voltage that can be compared with a reference. The operational characteristic of the tachometer is represented by a simple first-order transfer function.

To maintain a constant linear cable velocity, the angular reel velocity θ must increase as the cable unwinds — that is, as the effective radius of the reel decreases. The situation is complicated by the fact that the moment of inertia of the reel decreases as the cable unwinds, thereby reducing the torque required to maintain the constant cable velocity. Since the moment of inertia of the reel is proportional to the fourth power of its effective radius, this phenomenon is quite nonlinear. Common analytic control system techniques would, therefore, be inadequate for solving the design problem, and simulation seems the most suitable approach.

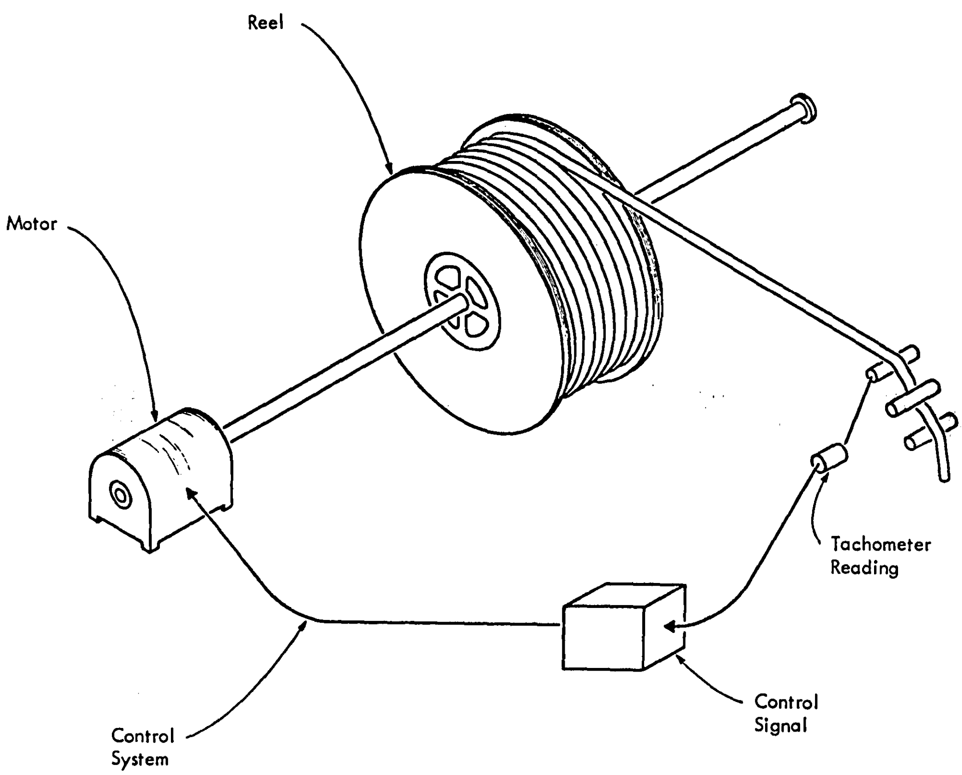


Figure 3. Cable reel control system

Table 3 presents the equations and specific physical data for the system. For a reel of width W and cable diameter D, there will be W/D windings per layer. Thus, the rate of change of effective radius R will be $D^2/2\pi W$ times the angular velocity of the reel. The motor output/input relationship is represented as a simple first-order transfer function. The cable speed and reel acceleration equations describe the basic dynamics of the problem. It is indicated that the desired linear cable velocity is 50 feet per second.

<u>Effective Radius of Reel</u>

$$R\ (\text{full}) = 4.0\ \text{ft.}$$

$$\dot{R} = -(D^2/2\pi W)\dot{\theta} = -K1\,\dot{\theta}$$

$$R\ (\text{empty}) = 2.0\ \text{ft.}$$

<u>Moment of Inertia</u>

$$I = 18.5\,R^4 - 221.0$$

<u>Tachometer Transfer Function</u>

$$\frac{V\ \text{measured (volts)}}{V\ \text{actual (fps)}} = \frac{2.0}{S + 2.0} = \frac{1}{0.5S + 1}$$

where S is the Laplace transform operator

<u>Torque Motor Transfer Function</u>

$$\frac{\text{Torque (ft lbs)}}{\text{Control Signal (volts)}} = \frac{500.0}{S + 1.0} = 500.0\left(\frac{1}{S + 1}\right)$$

<u>Cable Speed</u>

$$V_{actual} = R\dot{\theta}$$

<u>Desired Cable Speed</u>

50 ft/sec (represented by 50 volts at set point)

<u>Reel Velocity</u>

$$\dot{\theta} = \frac{1}{I}\int_0^t \text{Torque}\ dt$$

Table 3. Equations for cable reel control system

In approaching this problem, the engineer could conceivably work directly from either the system equations or a very detailed block diagram representation. Most likely, however, he would sketch the basic operational units in the type of block diagram shown in Figure 4. The S/360 CSMP statements would then be developed from a composite block diagram-differential equation representation of the dynamics.

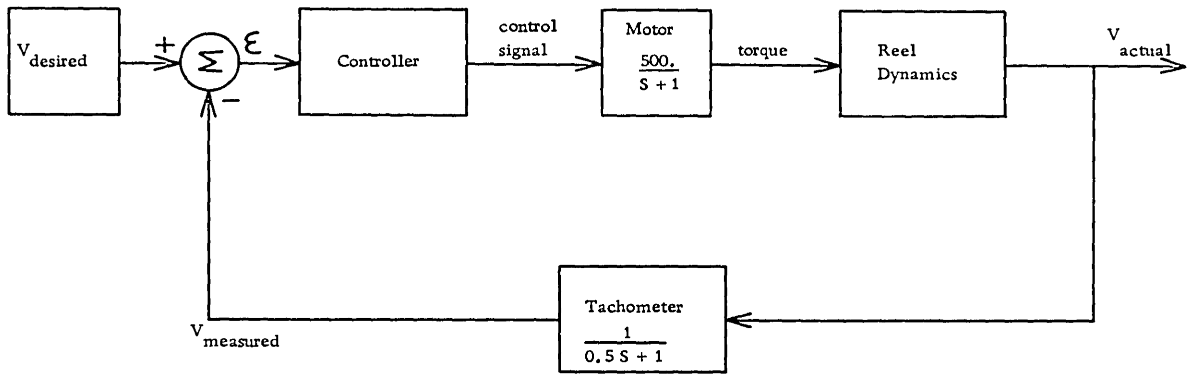


Figure 4. Block diagram for cable reel control system

```
                    ****CONTINUOUS SYSTEM MODELING PROGRAM****
                              *** VERSION 1.3 ***
TITLE                       CABLE REEL CONTROL DESIGN
INITIAL
                         K1 = (D**2)/(2.0*PI*W)
DYNAMIC
                     VDESIR = 50. * ( 1. -STEP(25.) -STEP(40.) )
                      ERROR = VDESIR - VM
                      CONTL = GAIN * ERROR
*         (REMEMBER TO ADD CONTROL LIMITING ON NEXT TRY)
                     TORQUE = REALPL(0.0, 1.0, 500.*CONTL)
                          I = 18.5 * (R**4)    - 221.0
                     THIDOT = ( 1. / I ) * INTGRL( 0.0, TORQUE )
                       RDOT = -K1 * THIDOT
                          R = INTGRL( RZERO, RDOT )
                       VACT = R * THIDOT
                         VM = REALPL(0.0, 0.5, VACT)
CONSTANT                 PI = 3.14159
PARAMETER             RZERO = 3.5,   D = 0.1,   W = 2.0
PARAMETER              GAIN = ( 0.5, 1.0, 1.5 )
FINISH                    R = 2.0       ,       R = 4.0
TIMER               FINTIM = 50.       ,   CUTDEL = 1.
PRTPLOT  VACT(-75., 75., ERROR, TORQUE, R)
LABEL SYSTEM RESPONSE (BRAKE AT 25 SECONDS, REWIND AT 40 SECONDS)
END
STOP
```

Figure 5. Listing of cable reel control system input

Figure 5 shows a complete listing of the statements that might be prepared for this problem. It must be emphasized that this is simply one possibility and that there is no "one best way" to describe the system. Some programs might be more direct or efficient, but any complete S/360 CSMP statement of the equations should produce equivalent results. Note again that S/360 CSMP is a nonprocedural language; thus the structure and parameter statements may be in any order in the card deck.

In this case, the engineer decided to provide flexibility in the program by entering D and W as parameters and directing the computer to determine the composite coefficient K1. This coefficient is computed only once — that is, during the initialization phase of the run. The necessary structure and data statements are identified as initializing operations by means of the translator control cards INITIAL and DYNAMIC, which respectively precede and follow that portion of the statements. The DYNAMIC card indicates the end of the initialization statements and the beginning of the dynamic portion of the simulation.

To prepare the dynamic simulation, the engineer began, in this instance, with the reel dynamics and developed the appropriate structure statements directly from the differential equations. He then proceeded around the block diagram, starting at the comparison point. Note that both the motor and tachometer blocks were modeled by the library function REALPL, which represents a first-order transfer function. Data statements specifying parameter values were inserted wherever the engineer found it convenient.

For the first attempt at a design, a simple proportional controller with a gain of 0.5 was simulated. The engineer added a comment card, which was included in the documentation, to remind himself that a modification should be considered in the next series of trial runs.

The programming was completed with a group of control statements specifying the "Finish" run termination conditions (R=2 or R=4); integration and data output intervals; and variables to be print-plotted; and tabulated. The engineer has requested a series of three runs with a controller gains of 0.5, 1.0 and 1.5. This illustrates how easily the user can prepare, in advance, the desired parameters, execution options, and output options for a series of runs. Figure 6 illustrates the printer output for the run with a gain of 0.5; Figure 7 shows the same for a gain of 1.5. These illustrations show that the system response is overdamped for a gain of 0.5, and is underdamped (that is, oscillating) for a gain of 1.5. The lower gain provides a smoother but slower transient behavior.

SYSTEM RESPONSE (BRAKE AT 25 SECONDS, REWIND AT 40 SECONDS) PAGE 1

MINIMUM -7.5000E 01 VACT VERSUS TIME GAIN = 5.0000E-01 MAXIMUM 7.5000E 01

TIME	VACT	ERROR	TORQUE	R
0.0	0.0	5.0000E 01	0.0	3.5000E 00
1.0000E 00	6.2500E 00	4.7136E 01	7.7342E 03	3.4995E 00
2.0000E 00	1.8555E 01	3.7374E 01	9.4610E 03	3.4967E 00
3.0000E 00	3.1109E 01	2.5014E 01	8.2537E 03	3.4910E 00
4.0000E 00	4.1287E 01	1.3736E 01	5.9229E 03	3.4827E 00
5.0000E 00	4.8313E 01	5.1905E 00	3.5241E 03	3.4724E 00
6.0000E 00	5.2416E 01	-3.4467E-01	1.5706E 03	3.4608E 00
7.0000E 00	5.4266E 01	-3.3105E 00	2.2182E 02	3.4485E 00
8.0000E 00	5.4631E 01	-4.4182E 00	-5.6252E 02	3.4358E 00
9.0000E 00	5.4182E 01	-4.3820E 00	-9.1236E 02	3.4232E 00
1.0000E 01	5.3420E 01	-3.7845E 00	-9.7733E 02	3.4107E 00
1.1000E 01	5.266CE 01	-3.0306E 00	-8.8838E 02	3.3983E 00
1.2000E 01	5.2061E 01	-2.3563E 00	-7.4177E 02	3.3860E 00
1.3000E 01	5.1669E 01	-1.8645E 00	-5.9741E 02	3.3738E 00
1.4000E 01	5.1466E 01	-1.5691E 00	-4.8487E 02	3.3616E 00
1.5000E 01	5.1401E 01	-1.4359E 00	-4.1233E 02	3.3494E 00
1.6000E 01	5.1420E 01	-1.4125E 00	-3.7539E 02	3.3372E 00
1.7000E 01	5.1474E 01	-1.4486E 00	-3.6416E 02	3.3249E 00
1.8000E 01	5.1534E 01	-1.5053E 00	-3.6804E 02	3.3125E 00
1.9000E 01	5.1581E 01	-1.5580E 00	-3.7828E 02	3.3001E 00
2.0000E 01	5.16C9E 01	-1.5951E 00	-3.8903E 02	3.2877E 00
2.1000E 01	5.1619E 01	-1.6140E 00	-3.9715E 02	3.2751E 00
2.20C0E C1	5.1616E 01	-1.6176E 00	-4.0167E 02	3.2626E 00
2.3000E 01	5.1606E 01	-1.6111E 00	-4.0290E 02	3.2500E 00
2.40C0E 01	5.1593E 01	-1.5995E 00	-4.0180E 02	3.2373E 00
2.5000E 01	5.1581E 01	-5.1587E 01	-3.9942E 02	3.2246E 00
2.6000E 01	4.3177E 01	-4.7738E 01	-8.0743E 03	3.2126E 00
2.7000E 01	2.6692E 01	-3.4608E 01	-9.3923E 03	3.2039E 00
2.8000E 01	1.083CE 01	-1.8515E 01	-7.4367E 03	3.1993E 00
2.9000E 01	-2.17C8E-01	-5.2120E 00	-4.3786E 03	3.1981E 00
3.0000E 01	-5.6885E 00	2.9362E 00	-1.6154E 03	3.1989E 00
3.1000E 01	-6.8134E 00	6.1824E 00	2.2305E 02	3.2005E 00
3.2000E 01	-5.4544E 00	6.0596E 00	1.0794E 03	3.2021E 00
3.3000E 01	-3.2431E 00	4.2930E 00	1.2033E 03	3.2032E 00
3.4000E 01	-1.2286E 00	2.2051E 00	9.2690E 02	3.2037E 00
3.5000E 01	1.2293E-01	5.4310E-01	5.2842E 02	3.2038E 00
3.6000E 01	7.646CE-01	-4.4035E-01	1.8089E 02	3.2037E 00
3.7C00E 01	8.7140E-01	-8.0897E-01	-4.3696E 01	3.2035E 00
3.8C00E 01	6.8051E-C1	-7.6663E-01	-1.4382E 02	3.2033E 00
3.9000E 01	3.9410E-01	-5.3044E-01	-1.5359E 02	3.2031E 00
4.00C0E 01	1.4020E-01	-5.0263E 01	-1.1540E 02	3.2031E 00
4.1000E 01	-8.456CE 00	-4.6186E 01	-7.7392E 03	3.2038E 00
4.2000E 01	-2.4585E 01	-3.3138E 01	-9.0240E 03	3.2079E 00
4.30C0E 01	-3.944CE 01	-1.7753E 01	-7.1257E 03	3.2159E 00
4.4000E 01	-4.9269E 01	-5.5726E 00	-4.2446E 03	3.2270E 00
4.50C0E 01	-5.38C2E 01	1.5034E 00	-1.7253E 03	3.2397E 00
4.6000E 01	-5.4510E 01	4.0842E 00	-1.1152E 02	3.2531E 00
4.7000E 01	-5.3210E 01	3.7917E 00	6.0316E 02	3.2662E 00
4.80C0E 01	-5.1331E 01	2.2240E 00	6.8463E 02	3.2789E 00
4.9000E 01	-4.9697E 01	4.9133E-01	4.4113E 02	3.2912E 00
5.0000E 01	-4.8625E 01	-8.4412E-01	1.1019E 02	3.3030E 00

Figure 6. Output of cable reel control system for gain = 0.5

SYSTEM RESPONSE (BRAKE AT 25 SECONDS, REWIND AT 40 SECONDS) PAGE 1

MINIMUM -7.5000E 01 — VACT VERSUS TIME, GAIN = 1.5000E 00 — MAXIMUM 7.5000E 01

TIME	VACT	ERROR	TORQUE	R
0.0	0.0	5.0000E 01	0.0	3.5000E 00
1.0000E 00	1.8457E 01	4.1488E 01	2.2207E 04	3.4985E 00
2.0000E 00	5.0488E 01	1.4551E 01	2.0789E 04	3.4906E 00
3.0000E 00	7.1259E 01	-1.1532E 01	6.9177E 03	3.4763E 00
4.0000E 00	7.2058E 01	-2.1669E 01	-6.4549E 03	3.4595E 00
5.0000E 00	5.9082E 01	-1.5122E 01	-1.1338E 04	3.4442E 00
6.0000E 00	4.5112E 01	-1.6073E 00	-7.6370E 03	3.4323E 00
7.0000E 00	3.9424E 01	7.9349E 00	-6.3055E 02	3.4227E 00
8.0000E 00	4.2885E 01	8.7238E 00	4.0799E 03	3.4132E 00
9.0000E 00	5.0173E 01	3.2116E 00	4.2503E 03	3.4024E 00
1.0000E 01	5.5256E 01	-2.8999E 00	1.3149E 03	3.3899E 00
1.1000E 01	5.5489E 01	-5.3880E 00	-1.7557E 03	3.3768E 00
1.2000E 01	5.2273E 01	-3.7687E 00	-2.8770E 03	3.3640E 00
1.3000E 01	4.8895E 01	-4.5766E-01	-1.9342E 03	3.3521E 00
1.4000E 01	4.7733E 01	1.7355E 00	-2.4097E 02	3.3406E 00
1.5000E 01	4.8901E 01	1.6448E 00	7.9321E 02	3.3292E 00
1.6000E 01	5.0848E 01	5.2902E-02	6.5888E 02	3.3172E 00
1.7000E 01	5.1946E 01	-1.4432E 00	-1.7766E 02	3.3048E 00
1.8000E 01	5.1659E 01	-1.7971E 00	-8.9769E 02	3.2923E 00
1.9000E 01	5.0610E 01	-1.0958E 00	-1.0154E 03	3.2799E 00
2.0000E 01	4.9800E 01	-1.7107E-01	-6.2812E 02	3.2677E 00
2.1000E 01	4.9767E 01	2.2163E-01	-1.7532E 02	3.2556E 00
2.2000E 01	5.0310E 01	-5.7892E-02	-1.6153E 01	3.2433E 00
2.3000E 01	5.0845E 01	-6.0005E-01	-1.8399E 02	3.2309E 00
2.4000E 01	5.0956E 01	-9.0799E-01	-4.5286E 02	3.2183E 00
2.5000E 01	5.0671E 01	-5.0804E 01	-5.8246E 02	3.2057E 00
2.6000E 01	2.5254E 01	-3.8904E 01	-2.2176E 04	3.1954E 00
2.7000E 01	-1.4995E 01	-3.5089E 00	-1.7192E 04	3.1944E 00
2.8000E 01	-3.0866E 01	2.3805E 01	4.7234E 02	3.2008E 00
2.9000E 01	-1.6824E 01	2.3381E 01	1.2410E 04	3.2071E 00
3.0000E 01	6.4508E 00	4.2099E 00	1.0664E 04	3.2084E 00
3.1000E 01	1.7537E 01	-1.2557E 01	8.3178E 02	3.2050E 00
3.2000E 01	1.1130E 01	-1.4165E 01	-6.7212E 03	3.2011E 00
3.3000E 01	-2.4260E 00	-3.7908E 00	-6.5657E 03	3.2001E 00
3.4000E 01	-9.9256E 00	6.5525E 00	-1.1257E 03	3.2018E 00
3.5000E 01	-7.0982E 00	8.4483E 00	3.5877E 03	3.2041E 00
3.6000E 01	6.1302E-01	2.9286E 00	3.9688E 03	3.2049E 00
3.7000E 01	5.5049E 00	-3.2939E 00	1.0214E 03	3.2040E 00
3.8000E 01	4.4714E 00	-4.9800E 00	-1.8660E 03	3.2027E 00
3.9000E 01	1.1983E-01	-2.1224E 00	-2.3735E 03	3.2021E 00
4.0000E 01	-3.0198E 00	-4.8402E 01	-8.0368E 02	3.2025E 00
4.1000E 01	-2.7432E 01	-3.5661E 01	-2.0741E 04	3.2055E 00
4.2000E 01	-6.3467E 01	-3.0662E 00	-1.5789E 04	3.2170E 00
4.3000E 01	-7.6526E 01	2.0685E 01	1.5272E 02	3.2349E 00
4.4000E 01	-6.4045E 01	1.9815E 01	1.0531E 04	3.2525E 00
4.5000E 01	-4.4650E 01	3.5280E 00	9.0017E 03	3.2657E 00
4.6000E 01	-3.5506E 01	-1.0343E 01	7.7410E 02	3.2752E 00
4.7000E 01	-4.0003E 01	-1.2094E 01	-5.5925E 03	3.2841E 00
4.8000E 01	-5.0059E 01	-4.5575E 00	-5.9016E 03	3.2950E 00
4.9000E 01	-5.6214E 01	3.4030E 00	-1.9867E 03	3.3079E 00
5.0000E 01	-5.5316E 01	5.7383E 00	1.7816E 03	3.3214E 00

Figure 7. Output of cable reel control system for gain = 1.5

GLOSSARY

Block diagram. A diagrammatic representation of the interconnection of functional blocks constituting the simulation model.

Continuous system. A system that can be adequeately modeled by a set of differential equations in which time is the independent variable.

Continuous system simulator. A simulation language that provides the block modeling capability of the digital analog simulators plus a powerful algebraic and logical modeling capability.

Digital analog simulator. A simulation language that provides a complement of functional block elements and a block-oriented language for specifying their interconnection.

Discrete system. A system that can be adequately modeled by a sequence of events that occur at discrete points in time.

Functional block (function). The basic structural unit of the simulation configuration.

Nonlinear element. An element in which the output is not directly proportional to the input but is some complex function of it — for example, square, exponential, sine.

Parameter. A value associated with a specific functional block to particularize the desired operation. For example, the parameters associated with a limiter define its upper and lower limits.

Procedural program. A program in which the order of presentation of language statements determines the order of execution of the corresponding computer operations.

Simulation (modeling). The act of representing some aspect of the real world by numbers or symbols that may be easily manipulated to facilitate their study.

Time-variant system. A system in which the parameters defining the functional relationships between variables are not constant but vary with time.

BIBLIOGRAPHY

Introduction to 1130 Continuous System Modeling Program II (GH20-0848). IBM Corporation, Data Processing Division, 1133 Westchester Avenue, White Plains, 1966.

Brennan, R.D., "Continuous System Modeling Programs: State-of-the-Art and Prospectus for Development", Proceedings of the IFIP Working Conference on Simulation Programming Languages, Oslo 1967.

Brennan, R.D., and Linebarger, R.N., "A Survey of Digital Simulation: Digital Analog Simulator Programs", Simulation, vol. 3, no. 6, December 1964, pp. 22-36.

Brennan, R.D., and Silberberg, M.Y., "Continuous System Modeling Programs", IBM Systems Journal, Vol. 6, No. 4, December 1967.

Clancy, J.J., and Fineberg, M.S., "Digital Simulation Languages: A Critique and Guide", 1965 Fall Joint Computer Conference, vol. 27, pp. 23-36.

Fahidy, T.Z., and Luke, C.A., "Digital Simulation of Chemical Reactor Dynamics", Instruments and Control Systems, Vol. 41, March 1968.

Syn, W.M., and Linebarger, R.N., "DSL/90 - A Digital Simulation Program for Continuous System Modeling", 1966 Spring Joint Computer Conference, April 26-28. 1966.

System/360 Continuous System Modeling Program (360A-CX-16X) User's Manual (GH20-0367). IBM Corporation, Data Processing Division, 1133 Westchester Avenue, White Plains, New York.

System/360 Continuous System Modeling Program (360A-CX-16X) System Manual (GY20-0111). IBM Corporation, Data Processing Division, 1133 Westchester Avenue, White Plains, New York.

READER'S COMMENT FORM

S/360 Continuous System Modeling Program

Application Description Manual

GH20-0240-3

Please comment on the usefulness and readability of this publication, suggest additions and deletions, and list specific errors and omissions (give page numbers). All comments and suggestions become the property of IBM. If you wish a reply, be sure to include your name and address.

COMMENTS

fold

fold

fold

fold

- Thank you for your cooperation. No postage necessary if mailed in the U.S.A.
FOLD ON TWO LINES, STAPLE AND MAIL.

YOUR COMMENTS PLEASE...

Your comments on the other side of this form will help us improve future editions of this publication. Each reply will be carefully reviewed by the persons responsible for writing and publishing this material.

Please note that requests for copies of publications and for assistance in utilizing your IBM system should be directed to your IBM representative or the IBM branch office serving your locality.

fold fold

FIRST CLASS
PERMIT NO. 1359
WHITE PLAINS, N.Y.

BUSINESS REPLY MAIL
NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

POSTAGE WILL BE PAID BY...

IBM Corporation
1133 Westchester Avenue
White Plains, N.Y. 10604

Attention: Technical Publications

fold fold

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
[U.S.A. only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

S/360 Continuous System Modeling Program ADM Printed in U.S.A. GH20-0240-3

GH20-0240-3

S/360 Continuous System Modeling Program ADM Printed in U.S.A. GH20-0240-3

IBM

International Business Machines Corporation
Data Processing Division
1133 Westchester Avenue, White Plains, New York 10604
(U.S.A. only)

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
(International)