Title	DARISS : a program system of block-oriented language for digital-analog simulation
Sub Title	
Author	横山, 光男(Yokoyama, Teruo) 北川, 節(Kitagawa, Misao)
Publisher	慶応義塾大学藤原記念工学部
Publication year	1967
Jtitle	Proceedings of the Fujihara Memorial Faculty of Engineering Keio University (慶応義塾大学藤原記念工学部研究報告). Vol.20, No.80 (1967.) ,p.151(7)- 170(26)
JaLC DOI	
Abstract	In this paper the programming system to provide a solution to problems involving analog and/or hybrid simulation is described. This system has features applicable to the problems from the simulation of analog computation to that of hybrid computation. Problem solving method is initiated with the preparation of a block diagram for the system to be simulated. The input language is block-oriented with the computer blocks corresponding one to one with the system blocks, and is easily learned, even by those without any analog and digital programming experience. The system has been named "DARISS" after "Digital Analog hybRId Simulation System".
Notes	
Genre	Departmental Bulletin Paper
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO50001004-00200080-0007

DARISS—A Program System of Block-Oriented Language for Digital-Analog Simulation

(Received February 21, 1968)

Teruo YOKOYAMA*
Misao KITAGAWA**

Abstract

In this paper the programming system to provide a solution to problems involving analog and/or hybrid simulation is described. This system has features applicable to the problems from the simulation of analog computation to that of hybrid computation.

Problem solving method is initiated with the preparation of a block diagram for the system to be simulated. The input language is block-oriented with the computer blocks corresponding one to one with the system blocks, and is easily learned, even by those without any analog and digital programming experience. The system has been named "DARISS" after "Digital Analog hybRId Simulation System".

I. Introduction

This programming system has been prepared for the simulation and analysis in various engineering problems which were constructed with linear and simple nonlinear analog and/or digital elements. These problems are so rarely solved analytically that those are often calculated with the aid of digital computer. In this case, however, the troubles due to the complexity of programming for them is always introduced. It is desirable to use a computer, keeping the physical meanings unchanged at each block of the block diagram, in process of system analysis, which is widely used in order to understand the engineering problems directly. For these purposes, DARISS has the capability to be applied easily to the computer by arranging the program written by a block-oriented input language which may be most easily converted from the block diagram.

Such attempts have been already published[2-7], and there exist many program systems concerned with the digital simulator for the sake of analog computer having similar thoughts. We also made such a simulator[9] for the digital computer K-1[8] several years ago.

This new programming system which involves perfectly the function of digital-analog simulation has been further developed to extend its function and to increase

* 横山光男, Instructor, Faculty of Engineering, Keio University.
** 北川 節, Associate Professor, Faculty of Engineering, Keio University.

its flexibility. DARISS has been written for the digital computer TOSBAC 3400 which is installed in our Central Laboratory.

The block elements having various functions and the specific notations for input language are prepared. Each block is capable of performing just one distinct operation, like integration, multiplication-addition, delay line, limiting, switching function, logical function, etc. Writing an input program corresponds to make the interconnection of these elements which are prepared as fundamental block in the block diagram used frequently in our studies. And then, the input program is punched on the specified cards together with a few additional elements demanded for this program. It should be recognized that in DARISS various sorts of elements provided make possible to describe directly an input program from any problems, i. e., t (time)-function expression in ordinary differential equation, Laplace transformation in a transfer function, and Z-transformation in a sampled-data control system. In any case, the suitable elements for the problems may be used. But a few attentions should be paid in the case of making the input program from Z-notation block diagram, because of the nature of a digital computer which is discontinuous computation tool. These are expressed in details later.

DARISS is written in symbolic language and an object program is compiled to the machine language.

II. Description of the Program System

2.1 Classification and Representation of the Block Elements.

Since this program system deals with the programs consisting of the digital and analog elements, and requires various kinds of natures in each element mentioned above, the numbers and kinds of input signals for it are different. But as the representation of an input language is desirable to standardize, it is decided that any element which was once designated becomes the unit of an input language and is punched on a card with the fixed format as a medium for the computer inputs.

Column allocation for a card is as follows:

Cols.	1~5	7~12		14~72	
	△△△△△	○○○	□□□	$(X_1), (X_2), \cdots, (X_m)$:	$P_1, P_2, \cdots, P_n$
Part	Label	Element symbol (3 cols.)	Element number (1~3 cols.)	Variable inputs	Parameters

where (X_i) $(i = 1, 2, \cdots, m)$ represents the i-th input signal to the element from any other element or itself. But the maximum number, m, is fixed each element. X_i is not always one input. When X_i is more than one input, it may take the form of product-sum which is put in parentheses having no restriction in length.

When X_i is only one input, it can not be put in parentheses. Namely, the parentheses mean the product-sum input, which is permissible in the specific elements. $P_i (i = 1, 2, \cdots, n)$ represents the i-th parameter and the maximum number, n, is fixed to each element. As a rule, this parameter is a constant number or output signal from any other element. In the latter case, however, the change of its value after received as an input cannot be permitted. When the element has more than one input signal and parameter, respectively, they are separated into two parts by colon (:), both parts are separated by comma each other. At element symbol part (cols. 7~9), alphabetic characters are placed which are named for each element as in Table 1 and 1~3 digits of decimal number are placed to element number part (cols. 10~12) to distinguish one from the other in case of using two or more identical elements. One element which is fixed through a certain program is designated together with both its element symbol and number. The label has a function which specifies the position of a program similar to other programs. Its name consists of five or less alphanumeric characters, the first of which should be alphabetic. One element (statement) is punched on a card, but it may be continued to the next cards by placing a slant symbol (/) on any column when the statement is to be placed beyond 72 columns. The blank column in the variable input or parameter part of the card neglects the following columns except for PRT element (Refer to Table 1).

Some examples of input language for basic element connections are described below:

INT 1 (INT 3×MAD 2, DIV 2): 1.0

In the block diagram representation in Fig. 1, INT 1 is the name of integrator No. 1. It has three inputs, INT 3, MAD 2, and DIV 2 with which the product-sum operation is performed before executing the integration, and an initial value of 1.0 is in the parameter part.

INT 1 (−13×INT 3×MAD 2, 2×DIV 2×MAD 1, LIM 3): −2.0

As an above-mentioned example shows, the product parts of the product-sum inputs can be multiplied by only one positive or negative integer, if necessary. The commas in parentheses mean the addition of the product-sum inputs.

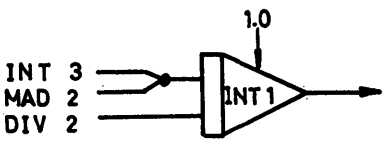


Fig. 1. Integrator example.

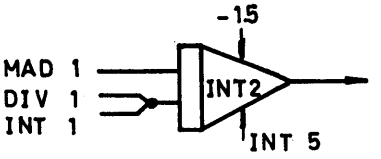


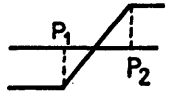
Fig. 2. Integrator example having independent variable designation.

Table 1. Functional description of DARISS elements.

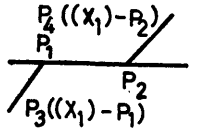
In the following, ( ) : Variable of product-sum form may be optionally permitted in this place.
If it has only one input, it must not be put in parentheses.
$\underline{\quad}$: Optional variable or parameter.
Z^{-1} : One clock time delay ($= e^{-ST}$)

Name	Symbol & general form	Function	Remarks
1. Mathematical element			
Integrator	INT (X_1), $\underline{X_2}$: P_1	$Y = \int (X_1)dt + P_1$	If X_2 is not specified, IDV 1 is automatically designated to the independent variable for INT or DIF.
Differentiator	DIF (X_1), $\underline{X_2}$: P_1	$Y = d(X_1)/dt$	P_1: Initial value
Multiplication-Adder	MAD (X_1),	$Y = (X_1)$	This element is applied to the subtractor, sign changer, amplifier, potentiometer, etc.
Divider	DIV (X_1), (X_2)	$Y = (X_1)/(X_2)$	If $(X_2) = 0$, error message is printed out on the console typewritter.
Linear function generator	LFC (X_1): P_1	$Y = P_1((X_1))$	P_1 is the function name. Fnuction name list is as follows: ABS (Absolute) SQR (Square root) SIN (Sine) COS (Cosine) EXP (Exponent) NLG (Natural logarithm) LOG (Common logarithm) ATN (Arc tangent) ASN (Arc sine) ACN (Arc cosine) TNH (Hyperbolic tangent) From the above list, only one function is designated to P_1 by using function symbol.
Maximum circuit	MAX (X_1), (X_2), $(\underline{X_3})$, $(\underline{X_4})$	$Y = \max_{i=1}^{4} (X_i)(i \leqq 4)$	

Minimum circuit	MIN (X_1), (X_2), $(\underline{X_3})$, $(\underline{X_4})$	$Y = \min_{i=1}^{4}(X_i)(i \leqq 4)$
2. Nonlinear element		
Function switch	FSW (X_1), (X_2), (X_3), (X_4)	$Y = \begin{cases}(X_2), \text{ if } (X_1) > 0,\\(X_3), \text{ if } (X_1) = 0, \text{ or}\\(X_4), \text{ if } (X_1) < 0.\end{cases}$
Plus switch	PSW (X_1), (X_2)	$Y = \begin{cases}(X_2), \text{ if } (X_1) > 0, \text{ or}\\0, \text{ if } (X_1) \leqq 0.\end{cases}$
Zero switch	ZSW (X_1), (X_2)	$Y = \begin{cases}(X_2), \text{ if } (X_1) = 0, \text{ or}\\0, \text{ if } (X_1) \neq 0.\end{cases}$
Non-zero switch	NSW (X_1), (X_2)	$Y = \begin{cases}(X_2), \text{ if } (X_1) \neq 0, \text{ or}\\0, \text{ if } (X_1) = 0.\end{cases}$
Limiter	LMT (X_1): P_1, P_2 $(P_1 < P_2)$	$Y = \begin{cases}P_2, \text{ if } (X_1) > P_2,\\(X_1), \text{ if } P_2 \geqq (X_1) \geqq P_1, \text{ or}\\P_1, \text{ if } (X_1) < P_1.\end{cases}$
Dead zone	DZN (X_1): P_1, P_2, P_3, P_4	$Y = \begin{cases}P_4((X_1)-P_2), \text{ if } (X_1) > P_2,\\0, \text{ if } P_2 \geqq (X_1) \geqq P_1, \text{ or}\\P_3((X_1)-P_1), \text{ if } (X_1) < P_1.\end{cases}$
Hysteresis	HYS (X_1): P_1, P_2, P_3, P_4	$Y = \begin{cases}\text{Stage 1}\\-P_2, \text{ if } (X_1) < P_1 + P_2/P_3 \text{ (point a), or}\\P_2, \text{ if } (X_1) \geqq P_1 + P_2/P_3 \text{ (point a).}\\\text{Stage 2}\\P_2, \text{ if } (X_1) > -P_1 - P_2/P_3 \text{ (point b), or}\\-P_2, \text{ if } (X_1) \leqq -P_1 - P_2/P_3 \text{ (point b).}\end{cases}$



P_3, P_4: Gradient.



P_4: Initial state, 1 or 0.

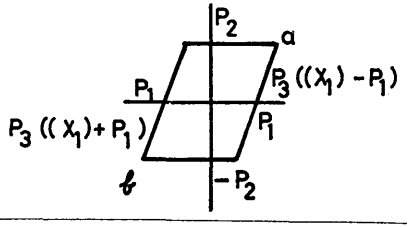
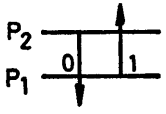


Table 1. Continued.

Name	Symbol & general form	Function	Remarks
3. Digital element			
AND circuit	AND $X_1, X_2, \underline{X_3}, \underline{X_4}$	$Y = X_1 \cdot X_2 \cdot X_3 \cdot X_4$	
OR circuit	IOR $X_1, X_2, \underline{X_3}, \underline{X_4}$	$Y = X_1 + X_2 + X_3 + X_4$	
Exclusive OR circuit	EOR X_1, X_2	$Y = \bar{X}_1 \cdot X_2 + X_1 \cdot \bar{X}_2$	
NOT circuit	NOT X_1	$Y = \bar{X}_1$	
Coincidence circuit	COD X_1, X_2	$Y = \bar{X}_1 \cdot \bar{X}_2 + X_1 \cdot X_2$	
Flip-flop	FFP (X_1): P_1, P_2, P_3	$Y = \begin{cases} 0, & \text{if } (X_1) < P_2, \text{ or} \\ 1, & \text{if } (X_1) > P_1. \end{cases}$	 P_1: Lower threshold. P_2: Upper threshold. P_3: Initial state, 1 or 0. FFP has the function to convert a non-logical signal into a logical one, therefore this output is 1 or 0.
Monostable multivibrator	MSM (X_1): P_1, P_2	$Y = \begin{cases} 1 \text{ for } P_2 \text{ steps, if } (X_1) > P_1, \text{ or} \\ 0, \text{ otherwise.} \end{cases}$	P_1: Triggering level. P_2: Step number defining pulse length.
Astable multivibrator	ASM (X_1): P_1, P_2	$Y = 1$ or 0, that is, output is alternately switched 1 or 0 for every P_2 steps.	When (X_1) signal greater than P_1 is applied, an astable oscillation is started.
Counter	CNT X_1: P_1, P_2, P_3	$Y = \begin{cases} Z^{-1}Y + P_1 \cdot P_2, & \text{if } X_1 = 1, \text{ or} \\ Z^{-1}Y, & \text{if } X_1 = 0. \end{cases}$	P_2: 1 or -1. P_3: Initial value. When $P_2 = 1$, CNT acts as an addition counter and is added by P_1, if $X_1 = 1$. When $P_2 = -1$, CNT is subtraction counter and is subtracted by P_1, if $X_1 = 1$. When $X_1 = 0$, both are holding to the previous state.
Trigger circuit	TRG X_1	$Y = \begin{cases} 1, & \text{if } Z^{-1}X_1 = 0 \text{ and } X_1 = 1, \\ 0, & \text{if } Z^{-1}X_1 = X_1 = 0, \text{ or} \\ -1, & \text{if } Z^{-1}X_1 = 1 \text{ and } X_1 = 0. \end{cases}$	

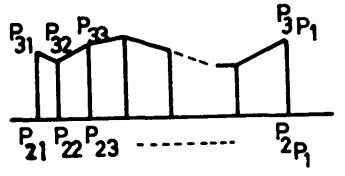
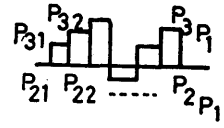
Sampler	SMP (X_1): P_1, P_2	$Y = \begin{cases} (X_1), \text{ at every } P_2 \text{ steps, or} \\ 0, \text{ otherwise.} \end{cases}$	P_1: Threshold. P_2: Sampling period. When (X_1) signal greater than P_1 is applied, SMP is started.
Holder	HLD (X_1), X_2	$Y = \begin{cases} X_1, \text{ if } X_2 \neq 0, \text{ or} \\ Z^{-1}Y, \text{ if } X_2 = 0 \end{cases}$	Zero order hold Equivalent Laplace transform, $(1-e^{-ST})/S$
Comparator	CMP (X_1), (X_2)	$Y = \begin{cases} 1, \text{ if } (X_1) \geqq (X_2), \text{ or} \\ 0, \text{ if } (X_1) < (X_2). \end{cases}$	
Modular	MOD X_1: P_1, P_2	$Y = \begin{cases} 1, \text{ if } Y_t = P_2, \text{ or} \\ 0, \text{ otherwise.} \end{cases}$	If $X_1 > P_1$, X_1 is added to $Z^{-1}Y_t$. If $Y_t = P_2$, then Y becomes to 1 and Y_t to 0. Y_t: Internal counter output.
4. Delay element			
Unit delay	UDL (X_1)	$Y = Z^{-1}(X_1)$	
Variable delay	DEL (X_1): P_1	$Y = Z^{-P_1}(X_1)$	P_1: Steps of delay.
5. Function element			
Non-linear function	NFC (X_1): P_1, (P_2), (P_3)	$Y = P_{3i} + ((X_1) - P_{2i}) \dfrac{P_{3(i+1)} - P_{3i}}{P_{2(i+1)} - P_{2i}}$ for $P_{2i} \leqq (X_1) \leqq P_{2(i+1)}$	P_1: Point number (up to 50). (P_2): X-coordinate set, $(P_{21}, P_{22}, \cdots, P_{2P_1})$ (P_3): Y-coordinate set, $(P_{31}, P_{32}, \cdots, P_{3P_1})$ Up to 50 sets of X-Y coordinates can be used. Coordinate points in P_2 and P_3 are separated by comma, respectively. The slope of the function is zero above and below the set of specified points. 

Table 1. Continued.

Name	Symbol & general form	Function	Remarks
Quantizer	QTZ (X_1): P_1, (P_2), (P_3)	$Y = P_{3i}$ for $P_{2i} \leqq (X_1) \leqq P_{2(i+1)}$	P_1, P_2, P_3: Same to NFC element.
Linear quantizer	LQZ (X_1): P_1	$Y = \left[\frac{(X_1)}{P_1}\right] \cdot P_1$	[ ]: Gauss symbol
			
6. Transfer function element*			
First order system	FST (X_1), $\underline{X_2}$: P_1, P_2, P_3	$Y = \frac{P_1}{1+SP_2} \cdot (X_1)$	P_1: Gain P_2: Time constant P_3: Initial value X_2: Designation of independent variable, if necessary.
Second order system	SND (X_1), $\underline{X_2}$: P_1, P_2, P_3, P_4	$Y = \frac{1}{S^2+2P_1P_2S+P_2^2} \cdot (X_1)$	P_3: Initial value of $\dot{Y}(0)$. P_4: Initial value of $Y(0)$. X_2: Same to FST element.
7. Special element			
Constant	CON : P_1	$Y = P_1$	
Independent variable	IDV : P_1, P_2, P_3 or IDV (X_1), (X_2), (X_3)		P_1, (X_1): Initial value. P_2, (X_2): Increment value. P_3, (X_3): Final value. If (X_1), (X_2), and (X_3) are given instead of P_1, P_2, and P_3, respectively, the specified output may be designated to the independent variable.
Printer	PRT** X_1, $\underline{X_2}$, $\underline{X_3}$, $\cdots$, $\underline{X_7}$: P_1, $\underline{P_2}$		X_1, X_2, $\cdots$, X_7: All the elements up to seven given as X's are printed out on the lineprinter. One element has 14

			digits length, in which 7 digits are the significant figure and the others are a sign, decimal point, and exponential part. P_1: Step number of print interval. P_2: Significant digit number, if necessary. ($P_2 \leq 7$)
	or PRT**: P_1, (P_2)		PRT has another representation and operation with one described above, in order to print headings on the paper of lineprinter. Only two parameters, P_1 and P_2, are given. When $P_1 = 1$, the paper is set to the first line of the next page and then characters indicated at P_2 are printed out. When $P_1 = 2$, the characters specified at P_2 are printed to the next line. When $P_1 = 3$, the character sets for several elements which are specified at (P_2) are printed to the output positions of each element of the next line keeping the regular intervals. The character sets are separated by comma, and put in parentheses.
Data read	DTR		One decimal data is read from one card on the card reader. This data may be used as a CON element after reading.
Loop	LOP X_1: P_1, P_2		X_1: Designation of the IDV element. P_1: A loop is formed with the element having the label indicated at this parameter part. P_2: When the current value of IDV becomes grater than the final value, a control is branched to the element having the label specified at this part.
Branch	BCH (X_1): P_1, P_2, P_3		Branch to the label of $\begin{cases} P_1, \text{ if } (X_1) > 0, \\ P_2, \text{ if } (X_1) = 0, \\ P_3, \text{ if } (X_1) < 0. \end{cases}$

Table 1. Continued.

Name	Symbol & general form	Function	Remarks
Declaration	DCL**		DCL is used to the declaration of macro-element before the execution of program.
Macro	MAC** #1, #2, ···: #·, ···		Label specified at MAC becomes the name of macro-element.
Macro end	MED**		The last element of macro-element declaration.
Start	STA**		Start element of execution program (source program).
Stop	STP**: $\underline{P_1}$		The last element of execution program. Control is transfered to P_1 (Label), if any.
End	END** $\underline{P_1}$		This shows the end of compilation to the compiler. Program run is started from STA element having the label of P_1, if any. If END has no parameter, the first STA is selected.

* These elements exist as built-in macro-element within DARISS.

** The element having no element number.

In Fig. 2, the output from INT 5 is designated to the second input, X_2, of independent variable for INT 2. This means that INT 2 performs the integration by using the output from INT 5 as an independent variable. In DARISS, any number of independent variable elements (IDV) can be available and perform the multi-time-base operation in general problems, and further the output from any other element can be designated to an independent variable as in the above example. If there is no designation for X_2 as shown in Fig. 1, IDV 1 becomes automatically the independent variable and performs a single time-base operation. Table 1 should be referred to for the element to be defined as independent variable. An example of the divider is shown in Fig. 3.

DIV 5 (INT 2×DIV 2, −1×MAD 1), (INT 5, −6×INT 2×MAD 3)

In the divider operation, the inputs of numerator and denominator are defined as (X_1) and (X_2), respectively, which are permitted to take the product-sum forms. This element has no parameter.

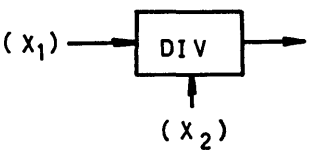


Fig. 3. Divider example.

The representation and symbols for all elements in DARISS are shown in Table 1.

2.2 Special Element of DARISS

The present simulator program using a digital computer necessitates the special control elements. In DARISS, such elements are prepared which decide the sequence of arithmetic operation among the elements, define the beginning or end of the sequence, or indicate the end of source program to the compiler.

STA is an element which defines the starting point of execution at one program section. If there are more than one program section in a program, the selection of the first running section among them may be possible by defining a label in the parameter part of END element.

STP is an element which indicates the end of one program section and may be available to the optional selection of the next element by the label in parameter part.

END is an element which indicates the end to the compiler, and is put on the last of source program.

LOP is an element which makes the connection from one element to the other. LOP also takes one independent variable (IDV) by which an arithmetic operation in a loop is performed. The flow of control leaves away from the loop when current value of the independent variable exceeds the final value.

BCH is used to control the flow of elements without the specified independent variable.

About the method to use these elements, refer to the later examples.

2.3 Designation of Macro-element

In DARISS, when the identical combinations of elements appear several times into the problem, an ability to define the macro-element as an optional element may be used. These are available like the other elements, by declarating it once in the program and giving an arbitrary name of three alphabetic characters to the label part of MAC. Macro-element also may be used for lack of the kind of elements prepared for DARISS by making up the elements with the combination of further fundamental elements.

The example defining the macro-element of Laplace transform $1/(S+a)$ is shown as follows.

Laplace transform representation of output is expressed in the form

$$Y(S) = \frac{1}{S+a} \cdot X(S) .$$

We assume that the input is a step function of magnitude X:

$$X(S) = \frac{X}{S} ,$$

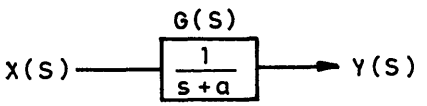


Fig. 4. Transfer function.

and hence

$$Y(S) = \frac{1}{a}\left(\frac{1}{S} - \frac{1}{S+a}\right)X .$$

The Z-transform of $Y(S)$ is expressed in the form

$$\begin{aligned} Y(Z) &= \frac{X}{a}\left(\frac{1}{1-Z^{-1}} - \frac{1}{1-e^{-aT}Z^{-1}}\right) \\ &= \frac{X}{a} \cdot \frac{Z^{-1}(1-e^{-aT})}{(1-Z^{-1})(1-e^{-aT}Z^{-1})} , \end{aligned}$$

where T is a sampling period. The Z-transform for a step function, $\frac{X}{S}$, becomes $\frac{X}{1-Z^{-1}}$. Thus the Z-transform of the transfer function is

$$G(Z) = \frac{Y(Z)}{X(Z)} = \frac{Z^{-1}(1-e^{-aT})}{a(1-e^{-aT}Z^{-1})} ,$$

and remember that $e^{-aT} = 1-aT+\frac{(aT)^2}{2} \cdots$.

As a result, it becomes

$$G(Z) = \frac{TZ^{-1}}{1-e^{-aT}Z^{-1}} .$$

Fig. 5 shows the sampled data block of $G(Z)$ and Fig. 6 also shows the block diagram by DARISS elements.

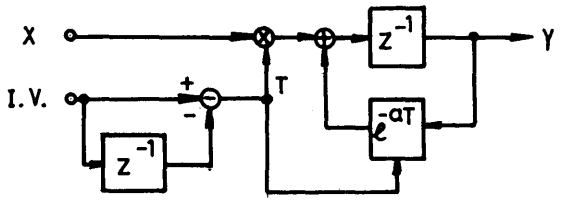


Fig. 5. Block diagram of $G(Z)$.

The macro-element to be written from Fig. 6 becomes as follows.

```
cols.   1~5   7~12        14~
              DCL
        TRN   MAC         #1, #2: #3
              UDL#2       #2
              MAD#1       (#2, -1×UDL#2)
              UDL#1       (#1×MAD #1, LFC #1)
              LFC#1       (-1×UDL #1×#3×MAD #1): EXP
              MED         UDL #1
```

This macro-element is named TRN and is available like the other prepared elements.

The declaration is performed with DCL, and both variable and parameter parts in MAC receive the inputs from the outside of macro-block as a formal variables and parameters with the sequence number (in this case: 1~3), respectively, in the left side column of which the sharp symbol (#) are needed to place. The element within macro-body also needs the sharp symbol to designate the internal element in any parts as the above example. When a number of TRN elements are used, the element with different element number are automatically reserved by the sharp symbol in each TRN. The output from the macro-element is indicated at the variable part of MED element. In the above example, this output becomes the output from UDL #1.

To use the macro-element, the following representation is taken:

```
TRN 3  INT 2, IDV 1: CON 3.
```

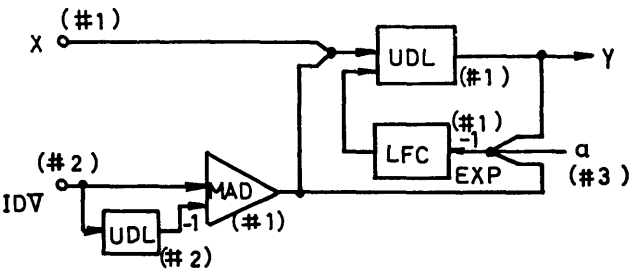


Fig. 6. Block diagram in DARISS elements.

In this example, we assume that input magnitude, X, is the output of INT 2, independent variable is IDV 1, and constant number, a, is CON 3. The element orderring of actual data (variables and parameters) in TRN 3 must correspond to that of formal data specified at MAC. The name of macro-elements must not be identical with the name of elements prepared in DARISS.

III. Examples

To illustrate the method of formulating a DARISS program, the following simple example is shown. Let us assume to solve the second order nonlinear differential equation

$$\frac{d^2y}{dt^2} + 2t\frac{dy}{dt} + t^2y = \sin y \ ,$$

for the conditions of $\dot{y}(0) = 1$, $y(0) = 0$ and also we are interested in observing y, $\dot{y}$, and $\sin y$ for $0 \leqq t \leqq 10$ at intervals of 0.01 units of time.

A block diagram that represents this equation is shown in Fig. 7.

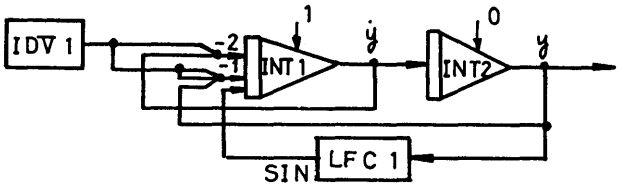


Fig. 7. Block diagram of the example.

The following program shows a complete listing for this problem.

```
      STA
      IDV   : 0, 0.01, 10
LOOP  INT 1 (−2×IDV 1×INT 1, −1×IDV 1×IDV 1×INT 2, LFC 1): 1
      INT 2 INT 1: 0
      LFC 1 INT 2: SIN
      PRT   : 3, (T, Y, Y DOT, SIN Y)
      PRT   IDV 1, INT 2, INT 1, LFC 1: 10
      LOP 1 IDV 1: LOOP, NEXT
NEXT  STP
      END
```

In the internal operation of DARISS, all the elements are processed sequentially, not in parallel as in an analog computer. Therefore it must be cut the loop of the block diagram at proper position and the elements are arranged in adequate

order. A calculation is performed in accordance with this order. In the above mentioned example, INT 1, INT 2, and LFC 1 are successively arranged and the elements from INT 1 having the label name of LOOP to LOP 1 will make a loop by containing two PRT elements for output operation. The output of INT 1 which is given to input of INT 2 represents $\dot{y}$ and has the initial value of 1. The output of INT 2 represents y and has the initial value of 0. LFC 1 is the element to convert the input signal i. e., y, to sine function, $\sin y$. PRT elements will cause the title printing of T, Y, Y DOT, and SIN Y to the first line of each page and, in each fixed position, print the current output of IDV 1, INT 2, INT 1, and LFC 1, respectively, at intervals of 0.1 units of time.

The result of actual computation in this problem is shown below.

T	Y	Y DOT	SIN Y
0.	0.	0.1000000E 001	0.
0.1000000E 000	0.9973504E −001	0.9929994E 000	0.9956978E −001
0.2000000E 000	0.1982678E 000	0.9759293E 000	0.1969713E 000
0.3000000E 000	0.2945814E 000	0.9485673E 000	0.2903393E 000
0.4000000E 000	0.3876313E 000	0.9105941E 000	0.3779965E 000
0.5000000E 000	0.4763401E 000	0.8616974E 000	0.4585298E 000
0.5999999E 000	0.5596050E 000	0.8016966E 000	0.5308515E 000
0.6999999E 000	0.6363170E 000	0.7306718E 000	0.5942373E 000
0.7999999E 000	0.7053931E 000	0.6490776E 000	0.6483332E 000
0.8999999E 000	0.7658182E 000	0.5578261E 000	0.6931270E 000
0.9999999E 000	0.8166930E 000	0.4583242E 000	0.7288857E 000
0.1100000E 001	0.8572835E 000	0.3524600E 000	0.7560674E 000
0.1200000E 001	0.8870660E 000	0.2425361E 000	0.7752217E 000
0.1300000E 001	0.9057637E 000	0.1311579E 000	0.7868966E 000
0.1400000E 001	0.9133697E 000	0.2108932E −001	0.7915673E 000
0.1500000E 001	0.9101552E 000	−0.8490800E −001	0.7895990E 000
0.1600000E 001	0.8966616E 000	−0.1842342E 000	0.7812474E 000
0.1700000E 001	0.8736773E 000	−0.2745941E 000	0.7666950E 000
0.1800000E 001	0.8422017E 000	−0.3541124E 000	0.7461108E 000
0.1900000E 001	0.8034002E 000	−0.4214137E 000	0.7197209E 000
0.2000000E 001	0.7585537E 000	−0.4756696E 000	0.6878723E 000
0.2099999E 001	0.7090052E 000	−0.5166154E 000	0.6510790E 000
0.2199999E 001	0.6561097E 000	−0.5445507E 000	0.6100389E 000
0.2299999E 001	0.6011863E 000	−0.5603391E 000	0.5656211E 000
0.2399999E 001	0.5454780E 000	−0.5654371E 000	0.5188268E 000
0.2499999E 001	0.4901181E 000	−0.5619958E 000	0.4707301E 000
0.2599999E 001	0.4361042E 000	−0.5531172E 000	0.4224115E 000
0.2699999E 001	0.3842814E 000	−0.5434187E 000	0.3748930E 000
0.2799999E 001	0.3353317E 000	−0.5402091E 000	0.3290824E 000

Another example which solves the problem of the above example with a different method is shown as follows.

The previous equation may be transformed into the forms

$$\begin{cases} \dfrac{d^2y}{dt^2} + 2t\dfrac{dy}{dt} + t^2y = z \\ \dfrac{d^2z}{dy^2} + z = 0 \end{cases}$$

which satisfies the initial conditions:

$$\dot{y}(0) = 1,\ y(0) = 0, \left(\frac{dz}{dy}\right)_{y=0} = 1,\ (z)_{y=0} = 0\ .$$

The auxiliary variable z which is solved from the second equation is the function of y and hence

$$z = \sin y\ .$$

By substituting it to the first equation, this equation which has two independent variables may be solved. In this case, the block diagram is shown in Fig. 8.

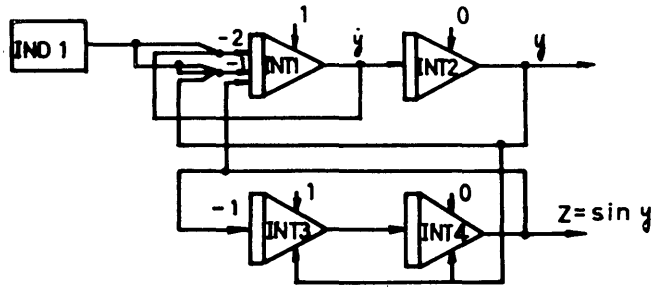


Fig. 8. Block diagram in DARISS elements.

The source program written in Fig. 8 and the result of it are shown as follows.

```
        STA
        IDV 1 : 0, 0.01, 10.0
LP 1    INT 1 (−2×IDV 1×INT 1, −1×IDV 1×IDV 1×INT 2, INT 4): 1
        INT 2 INT 1: 0
        INT 3 (−1×INT 4), INT 2: 1
        INT 4 INT 3, INT 2: 0
        PRT   : 3, (T, Y, Y DOT, SIN Y)
        PRT   IDV 1, INT 2, INT 1, INT 4: 10
        LOP 1 IDV 1: LP 1, STOP
STOP    STP
        END
```

T	Y	Y DOT	SIN Y
0.	0.	0.1000000E 001	0.
0.1000000E 000	0.9973504E −001	0.9929995E 000	0.9957148E −001
0.2000000E 000	0.1982678E 000	0.9759296E 000	0.1969754E 000
0.3000000E 000	0.2945815E 000	0.9485682E 000	0.2903477E 000
0.4000000E 000	0.3876315E 000	0.9105962E 000	0.3780134E 000
0.5000000E 000	0.4763407E 000	0.8617015E 000	0.4585620E 000
0.5999999E 000	0.5596061E 000	0.8017044E 000	0.5309084E 000
0.6999999E 000	0.6363192E 000	0.7306855E 000	0.5943301E 000
0.7999999E 000	0.7053971E 000	0.6491001E 000	0.6484740E 000
0.8999999E 000	0.7658249E 000	0.5578603E 000	0.6933267E 000
0.9999999E 000	0.8167038E 000	0.4583729E 000	0.7291522E 000
0.1100000E 001	0.8573000E 000	0.3525252E 000	0.7564030E 000
0.1200000E 001	0.8870899E 000	0.2426183E 000	0.7756216E 000
0.1300000E 001	0.9057966E 000	0.1312555E 000	0.7873471E 000
0.1400000E 001	0.9134130E 000	0.2119829E −001	0.7920454E 000
0.1500000E 001	0.9102097E 000	−0.8479427E −001	0.7900731E 000
0.1600000E 001	0.8967274E 000	−0.1841246E 000	0.7816787E 000
0.1700000E 001	0.8737533E 000	−0.2744992E 000	0.7670399E 000
0.1800000E 001	0.8422860E 000	−0.3540433E 000	0.7463237E 000
0.1900000E 001	0.8034897E 000	−0.4213812E 000	0.7197571E 000
0.2000000E 001	0.7586442E 000	−0.4756827E 000	0.6876914E 000
0.2099999E 001	0.7090919E 000	−0.5166807E 000	0.6506468E 000
0.2199999E 001	0.6561871E 000	−0.5446711E 000	0.6093294E 000
0.2299999E 001	0.6012489E 000	−0.5605140E 000	0.5646176E 000
0.2399999E 001	0.5455206E 000	−0.5656621E 000	0.5175215E 000
0.2499999E 001	0.4901359E 000	−0.5622631E 000	0.4691240E 000
0.2599999E 001	0.4360936E 000	−0.5534164E 000	0.4205133E 000
0.2699999E 001	0.3842397E 000	−0.5437376E 000	0.3727172E 000
0.2799999E 001	0.3352576E 000	−0.5405350E 000	0.3266484E 000

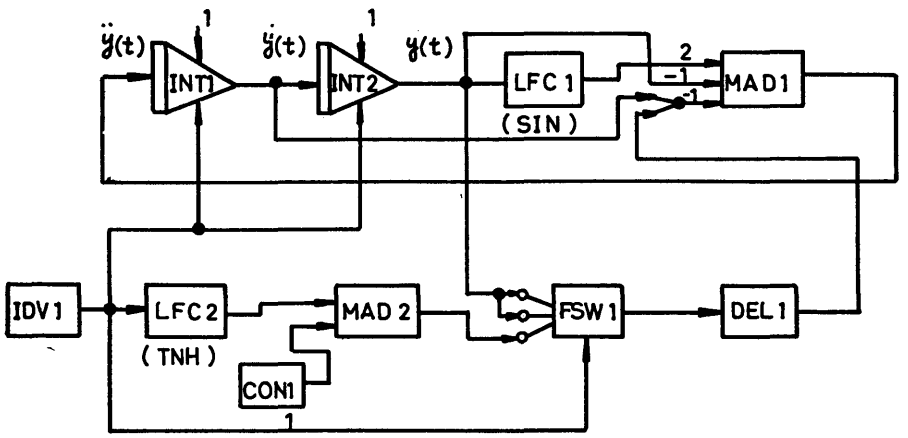


Fig. 9. Block diagram in DARISS elements.

It may be permitted to designate the identical element to the different passes of calculation in a program, only when the element has the same variable and parameter inputs. To show the example of these case, we assume that the ordinary differential equation is given in the form

$$\frac{d^2y(t)}{dt^2}+y(t-0.3)\frac{dy(t)}{dt}+y(t)=2\sin y(t)\ ,$$

for $0\leq t\leq 20$, where the conditions $y(0)=\dot{y}(0)=1$ are satisfied, and that, however, $y(t)=\tanh t+1$ for $-0.3\leq t\leq 0$.

The block diagram of this problem is shown in Fig. 9 and the program list is as follows:

```
        STA
        CON 1 : 1
        IDV 1 : −0.3, 0.01, 20
LOOP    BCH 1 IDV 1: PLUS, PLUS, MINUS
MINUS   LFC 2 IDV 1: TNH
        MAD 2 (LFC 2, CON 1)
        FSW 1 IDV 1, INT 2, INT 2, MAD 2
        DEL 1 FSW 1: 31
        BCH 2 :COM
PLUS    INT 1 MAD 1: 1
        INT 2 INT 1: 1
        LFC 1 INT 2: SIN
        FSW 1 IDV 1, INT 2, INT 2, MAD 2
        DEL 1 FSW 1: 31
        MAD 1 (2×LFC 1, −1×INT 2, −1×INT 1×DEL 1)
COM     LOP 1 IDV 1: LOOP, STOP
STOP    STP
        END
```

In this problem, at first, y is calculated with the equation $y=\tanh t+1$ for $-0.3\leq t\leq 0$, and this value is send to DEL 1, and then the differential equation is solved when the output of IDV 1 becomes 0. Therefore, the flow of calculation is divided into two parts, but FSW 1 and DEL 1 are commonly used in both parts having similar variable and parameter inputs as the above example listing.

BCH 2 shows that the flow of control branches unconditionally to the element having the label name of COM. In such a case, we can take the label name branched in the parameter part, P_1, after colon symbol.

We showed the simple examples of the solution of ordinary differential equation, to illustrate the method of formulating a DARISS program. Similar to these, we

shall be able to analyze more complex problems containing both analog and digital elements by using the fixed representation for each element shown in Table 1.

In DARISS, the reservation of used elements becomes relocatable in the main memory, and so the total numbers of each element have no restriction as far as all the used elements do not exceed the memory capacities. We are available about one thousand elements for the analysis of a given problem.

IV. Conclusions

DARISS is a developed programming system in order to simulate the hybrid system containing both digital and analog elements as mentioned above. It will be available for the people with little or no digital computer experience and with the problems to be calculated with the aid of digital computer in many application areas including process analysis and control, circuit analysis and design, mechanical dynamics, plant modeling, etc. As a matter of course, digital simulation methods are used, only when realtime solutions are unnecessary and when physical hardware is not to be included in the simulation. But this is especially attractive to the people expecting to use the excellent digital computing facilities and having poor analog computing capability. The inclusion of logical simulation capability is extended the usefulness of digital simulation to new problem areas.

The advantages of digital simulation are as follows: elimination of scaling problem (the number treated with DARISS is between -10^{+153} and 10^{+153}.), reliable and various elements (DARISS has more than 40 types of elements), accurate operation, simple problem coding, reduced problem preparation time, and so on.

As digital computer becomes faster and more versatile than at present, it will be possible to realize the digital simulation and the realtime simulation of very large systems, which become still more important in the fields of engineering.

Acknowledgement

We wish to thank Professor Tomoyuki Somiya and Professor Jiro Yamanouchi for their kind introduction and encouragement, and also Mr. Takehiko Tanaka and Mr. Hisayoshi Toda for their excellent contribution in programming. The discussions made by Mr. Yasuyuki Matsuo for our study are particularly acknowledged.

References

1. F. Lesh: "Methods of Simulating a Differential Analyzer on a Digital Computer", ACM Journal, Vol. 5, No. 3, 1958.
2. J. R. Hurley and J. J. Skiles: "DYSAC: A Digitally Simulated Analog Computer", AFIPS Proceedings, Spring Joint Computer Conference, Vol. 23, 1963.
3. R.A. Gaskill, J.W. Harris, and A.L. McKnight: "DAS: A Digital Analog Simulator",

AFIPS Proceedings, Spring Joint Computer Conference, Vol. 23, 1963.
4. M. Palevsky and J.V. Howell: "The DES-1—A Real-Time Digital Simulation Computer", AFIPS Proceedings, Fall Joint Computer Conference, Vol. 24, 1963.
5. R.T. Harnett and F.J. Sansom: "MIDAS Programming Guide", Report No. SEG-TDR-64-1, Wright-Patterson AFB, Ohio, Jan. 1964.
6. B. J. Karafin: "The New Block Diagram Compiler for Simulation of Sampled-Data Systems", AFIPS Proceedings, Fall Joint Computer Conference, Vol. 27, 1965.
7. R. N. Linebarger: "DSL/90—A Digital Simulation Program for Continuous System Modeling", AFIPS Proceedings, Spring Joint Computer Conference, Vol. 28, 1966.
8. M. Kitagawa and T. Tsuzuki: "Transistorized Digital Automatic Computer Keio Mark 1", Proceedings of Fujihara Memorial Faculty of Engineering, Keio University, Vol. 12, No. 45, pp. 45-59, 1959.
9. T. Yokoyama: "Automatic Programming for Digital-Analog Computer", Master Thesis, Faculty of Engineering, Keio University, 1962.