[redacted]

N66-17082
(ACCESSION NUMBER) (THRU)
27 1
(PAGES) (CODE)
CR 70340 05
(NASA CR OR TMX OR AD NUMBER) (CATEGORY)

FACILITY FORM 602

GPO PRICE $ ______

CFSTI PRICE(S) $ ______

Hard copy (HC) 2.00

Microfiche (MF) .50

ff 653 July 65

MELPAR INC

A SUBSIDIARY OF WESTINGHOUSE AIR BRAKE COMPANY

3000 ARLINGTON BOULEVARD, FALLS CHURCH, VIRGINIA

Human Performance Control

Monitoring Systems

Interim Report No. 2

Contract No. NASW-1085

15 January 1966

Prepared for

NASA Headquarters
Washington, D.C.

Prepared by

J. M. Gervinski and R. E. Mirabelli

Melpar, Inc.
3000 Arlington Boulevard
Falls Church, Virginia

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

1. INTRODUCTION

This Interim Report describes the basic features of the computer program developed under Contract No. NASW-1085.

During the first phase of the Human Performance Control and Monitoring System contract, theoretical studies were performed, and a mathematical model of a performance control and monitoring system was developed. To illustrate the use of adaptive logic in such a system, three possible problems were posed for simulation on a digital computer. The program selected by NASA for simulation was the following:

> "Given that failures and/or changes in plant characteristics have occurred in an automatic control system, can trainable logic be designed to take over the control function by monitoring of human performance?"

In this, the second phase of the contract, a computer program has been written which simulates a second-order servo plant and a controller has been made from adaptive logic elements. This program provides the option of being in the automatic or manual control mode at any time. Preliminary results indicate the system dynamics are working correctly and the controller is trainable.

2. PROBLEM FORMULATION

2.1 Review of Problem Motivation

Space flights to date indicate that man is one of the most reliable components in the complex man-machine system during space flights. In the future it is logical to expect that his capability to monitor space vehicle systems, perform control functions, and troubleshoot will make him a utility backup for many existing subsystems in the spacecraft. With this in mind, it seems reasonable to assume that his workload will vary greatly, depending upon equipment performance. Further, we may assume that man's performance on tasks deteriorates if he becomes overloaded with work. It is possible for trainable logic to relieve some of the burden.

Let us suppose that one of the many automatic control or regulator systems fails and must be controlled manually. It is possible that, while the system is being controlled manually, trainable logic monitors both the astronaut's control and the data upon which the astronaut is basing his control decisions. In this way, automatic control can be re-established by the trainable logic reorganization.

2.2 Plant

The plant chosen for simulation is a servomotor that is adjusting to command inputs, which are step functions. The differential equation governing its motion is:

$$\ddot{y} + a\dot{y} = Ku \ .$$

It is desired to control the plant so that y(t) approaches a desired sequence of values. The desired output is y_{in} and the actual output is y_{out}. The difference is, of course, the error. See figure 2-1.

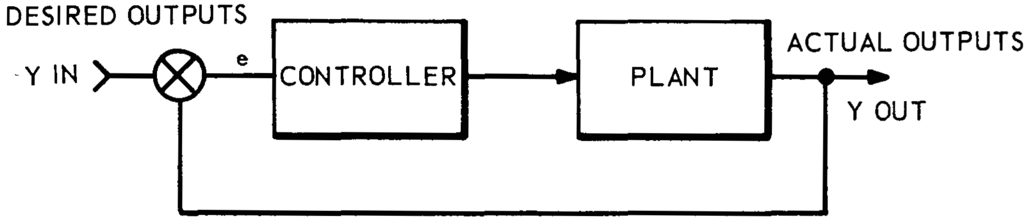

Figure 2-1. Basic Feedback Control Loop

The control variable, u, is constrained to take on one of three possible values:

$$u = (1, 0, -1) .$$

2.3 Performance

Thinking of u as a torque-producing parameter and $|u|$ as a rate of fuel consumption, we consider a system which attempts to null its error while minimizing a combination of fuel and time. For a single step input the functional

$$P(u,t) = \int_{t' = t_o}^{t' = t_f} (C|u| + 1)\, dt'$$

is minimized. Where $t_f - t_o$ is the time required to bring the system to the desired output value. By letting the time intervals between step changes be much greater than the system time constant, the steps can be considered independent in time. This being the case, performance may be judged on nulling the error for individual steps. To accomplish this, the function P is treated as a cost function and its value is compared with an expected value, E(P). The expected value is:

$$E(P) = \text{min cost} + \text{tolerance}$$

$$= \min \int_{t_o}^{t_f} (C|u| + 1)\, d\tau + \gamma$$

where the minimization is over control policy $u\ (e,\dot{e})$.

A warning of performance deterioration is given to the human when

$$E(P) - P \geq 0.$$

2.4 Control Policy

A control policy, $u(e,\dot{e})$, is a specification of control values (1,0,1) for all points in the error, error rate plane. A convenient method is dividing the phase plane into regions and specifying control values for each of the regions. The proper choice of regions is arrived at by laborious computation of switching boundaries for the control variable u, which minimizes the performance criterion. These boundaries are dependent on both plant parameters and the choice of performance criterion.

A change in the control policy may be brought about either by a change of switching boundaries or by a change of the control values used within the regions defined by the boundaries. It was decided to take the latter approach. The phase plane has been divided into more regions than an optimal control policy demands. In addition, the boundaries can be adjusted by input data. The extra regions permit a selection from a larger class of control policies, while the adjustable boundaries permit experiments to be conducted with various values of plant parameters.

When the performance of the automatic control system is judged to be inadequate, the control may be transferred to manual mode.

In the manual mode, adaptive logic monitors the manual control and adapts to an available control policy which most closely resembles that of the human. A block diagram indicating the flow of information is shown in figure 2-2. Section 3 explains the computer implementation of the problem.

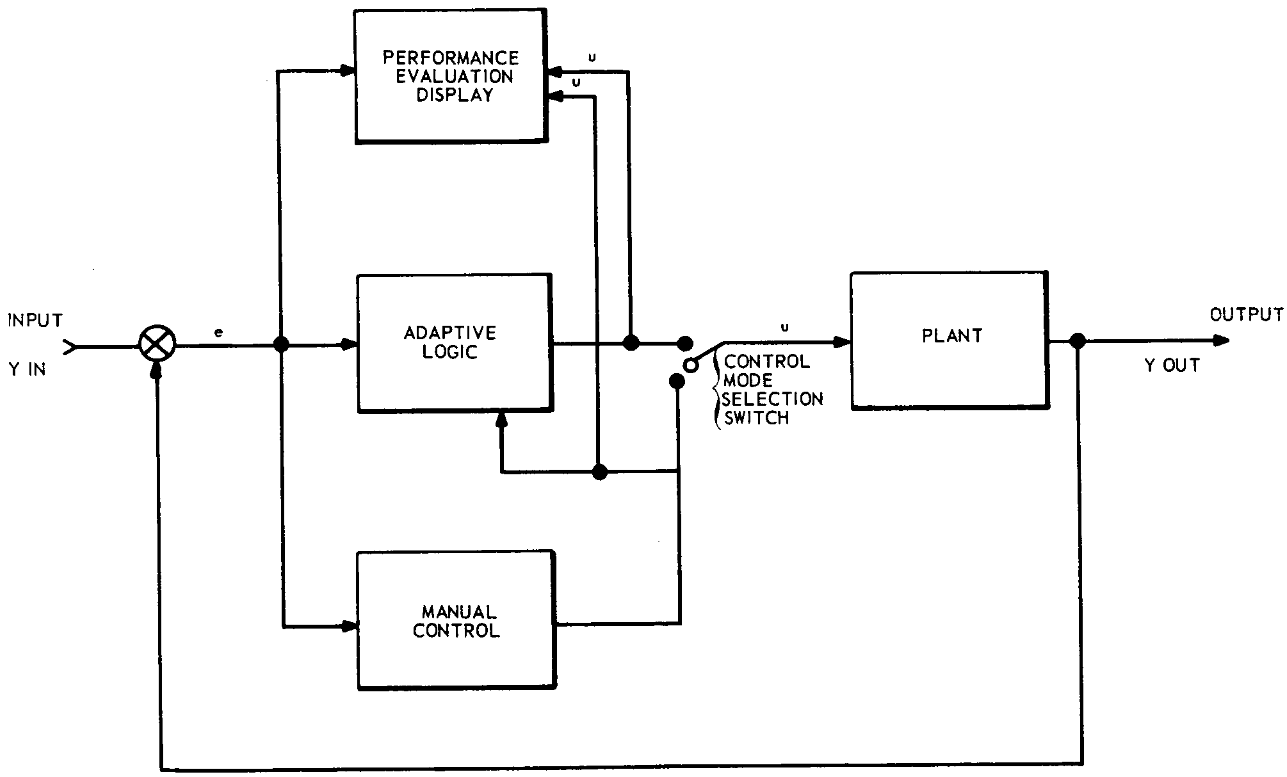

Figure 2-2. Control Policy Change with Adaptive Logic

3. DIGITAL IMPLEMENTATION

The digital program for the Human Performance Control and Monitoring System was written for the SDS 910 computer. The main program is in FORTRAN, and the random number subroutine is in Meta Symbol. A flow diagram of the program is presented in figure 3-1. A complete list of symbols and the program listings are presented in appendixes A and B.

The program begins by reading in the data for the experiment and setting the system parameters equal to their initial values. The expected performance is computed for the value of y_{in} (desired output) corresponding to TIME = 0, as explained in section 2.

The main loop of the program (figure 3-2) is then completed for each increment of time. The state variables are evaluated as to their position in the phase space, which is presently divided by four straight lines with variable slopes and intercepts and one curve through the origin. This quantizes the space into 32 possible regions. Associated with each region is a control value and a counter that is used when monitoring manual operation. The training takes place by rewarding the counter when the manual control and the control value associated with the region agree, and punishing the counter otherwise. The maximum number of steps in the counter is a variable and is input at the beginning of the experiment. If the counter is decreased to zero, a new random control is generated and is now associated with that region. A specific example of the above procedure is given below where the number of steps needed for training is set at 3.

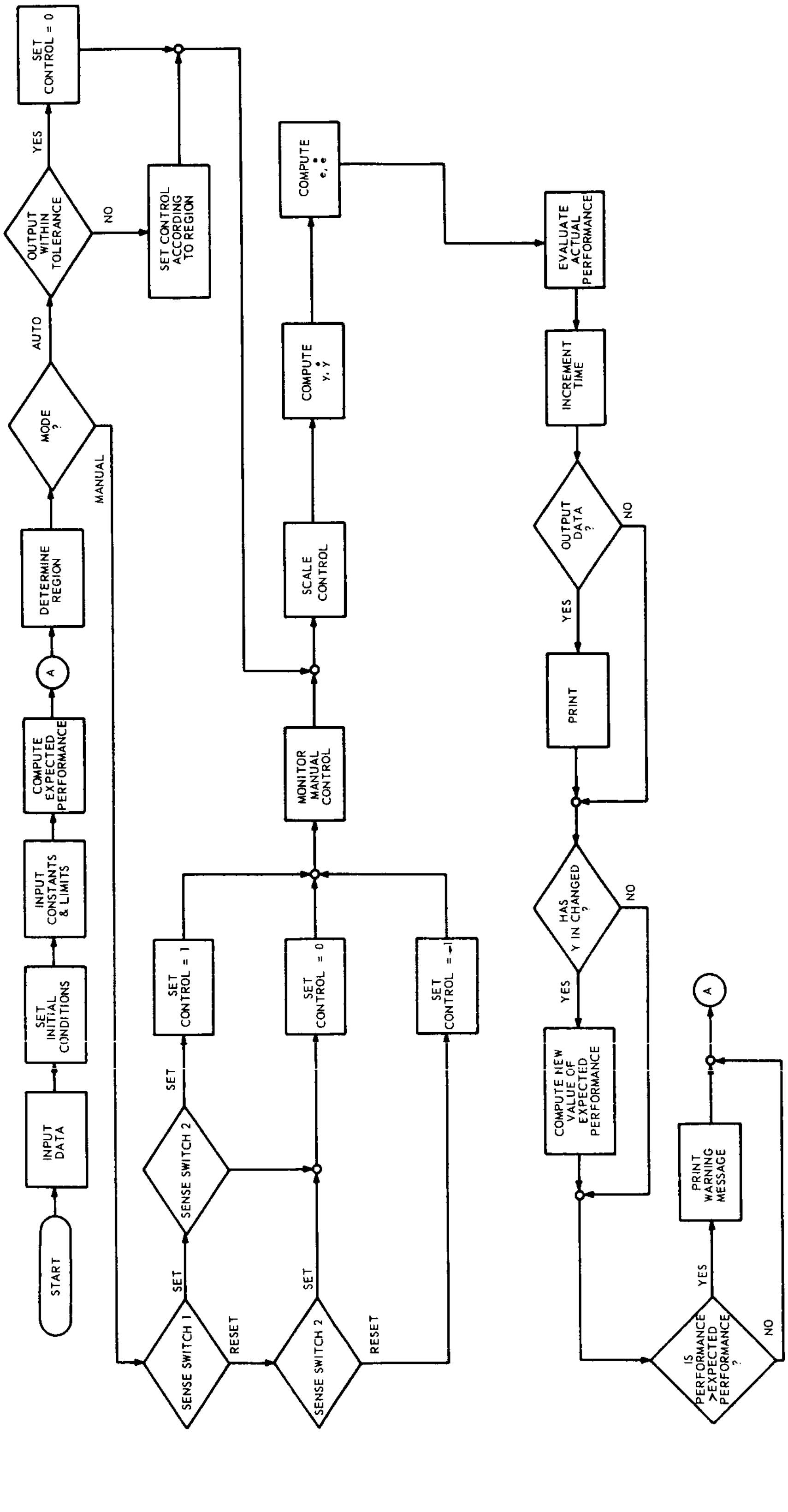

Figure 3-1. Flow Diagram of Digital Program

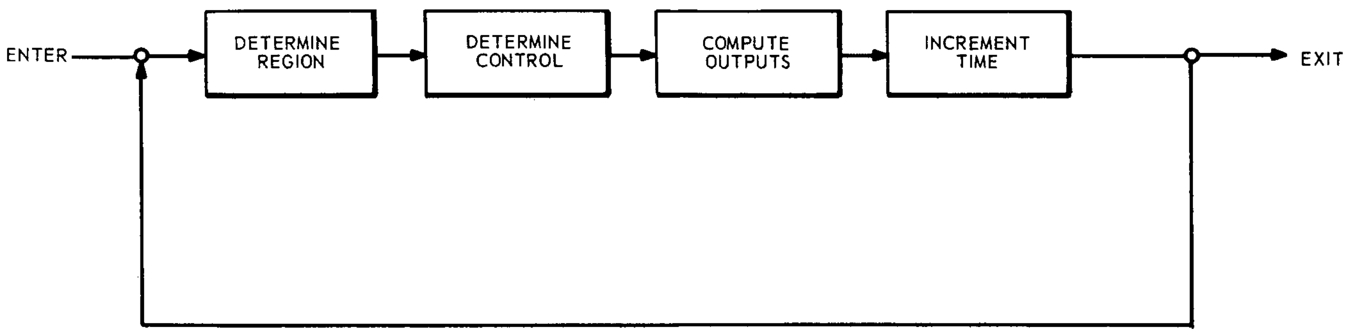

Figure 3-2. Main Loop of Digital Program

Time	Manual Control	Trained Control	Counter
t_k	1	1	2
t_{k+1}	1	1	3
t_{k+2}	1	1	3
t_{k+3}	-1	1	2
t_{k+4}	-1	1	1
t_{k+5}	-1	1	0
t_{k+6}	-1	(Random) 0	0
t_{k+7}	-1	(Random)-1	1
t_{k+8}	-1	-1	2
t_{k+9}	-1	-1	3
t_{k+10}	-1	-1	3

A fairly simple method of generating pseudo random numbers in a binary digital machine was found;* for our purpose, the series appears to be generated by random processes. While adequate random numbers were available on punched cards or magnetic tape, they were impractical for our use because of insufficient quantity and slow access. The deterministic method employed is given by the equation

$$R_{n+1} = KR_n \bmod 2^N$$

* Ralston, Anthony and Wilf, Herbert S., Mathematical Methods for Digital Computers, John Wiley & Sons, 1964, p. 253.

where

R_n is the nth random number

R_{n+1} is the (n+1) st random number

K is a constant multiplier - the largest odd power of 5 that a 24-bit word will hold

N is the number of binary digits per word = 24.

The mod 2^N operation is done by simply taking K times R_n and then setting R_{n+1} equal to the least significant half of the result. It can be shown that starting with an odd R_o, one will run through 2^{N-2} numbers before repeating a number. Since our random decisions could only take on three values, -1, 1, and 0, only 2 bits of the generated 24 random bits were used per decision, according to the following tabulation.

Random Bits	Decision
0 0	0
0 1	1
1 1	-1
1 0	Not used

This then increases our repeatability factor by 6.

Since four sense switches are available on the SDS 910 computer, it was decided to have SS 4 determine the mode of operation and a combination of SS 1 and SS 2 the control value when in the manual mode. When in the automatic mode, the trained control is used.

$$SS\ 4 \begin{cases} \text{Set - Manual mode} \\ \text{Reset - Automatic mode} \end{cases}$$

SS 1	SS 2	Manual Control
Set	Set	1
Set	Reset	0
Reset	Set	0
Reset	Reset	-1

This control value is then altered by the system gain constant, which is input with the initial data.

Straightforward computations follow which evaluate the plant equations and the error equations.

$$y(t_{k+1}) = \frac{u}{a}\left\{\tau - \frac{1}{a}\right\} + \frac{\dot{y}(t_k) + a \cdot y(t_k)}{a} + \left\{\frac{u}{a^2} - \frac{\dot{y}(t_k)}{a}\right\} e^{-a\tau}$$

$$\dot{y}(t_{k+1}) = \frac{u}{a} - \left\{\frac{u}{a} - \dot{y}(t_k)\right\} e^{-a\tau}$$

$$e(t_{k+1}) = y_{in} - y(t_{k+1})$$

$$\dot{e}(t_{k+1}) = \dot{y}_{in} - \dot{y}(t_{k+1})$$

where

τ = time increment

a = input constant

u = control value.

The actual performance is then evaluated where $P = \int_{t_o}^{t_k} (Cp|u| + 1)\,dt$ and checked against the expected performance. Time is incremented, and the data for this loop is output if sense switch 3 is reset. Before repeating the main loop, a check is made to see if the value of y_{in} has changed. If it has, a new value for expected performance is computed. This process continues until the upper limit of the performance integral is found, which occurs when

$$\dot{e}^2 + e^2 \leq C_e \text{ where } C_e \text{ is a specified constant.}$$

4. CONCLUSIONS

During the second phase of this contract, a computer program was written which simulates the dynamics of a second-order servosystem under both automatic and manual control. The operating mode of the system is determined by an external sense switch on the computer. External sense switches also control the torque value when in the manual mode. The monitoring of this manual mode results in the training of the adaptive logic. Preliminary results indicate that the system is performing as expected and that the controller is trainable.

Experimental plans for next period include determining how the human operator responds to:

a. Changes in plant parameters.

b. Changes in control objectives related to performance criteria.

These experiments will provide a basis for investigating the organization of adaptive logic controllers.

APPENDIX A

LIST OF SYMBOLS

FORTRAN Name	Meaning
A	Constant used in y, $\dot{y}$, and performance equations
ALPH(J)	Slope of line J
BETA(J)	Intercept of line J
CE	Tolerance for desired output region
CP	Constant used in performance evaluation
ER	e = error
ERDAB	Absolute value of $\dot{e}$
ERDOT	$\dot{e}$
EXPR	Expected performance
ICTR(M)	Counter for region M
IFLAG	Flag to denote change in y_{in}
IRU(M)	Monitored control for region M
L	Number of steps in time function for y_{in}
NUMST	Number of steps needed for training
OFLAG	Flag to denote actual output within tolerance region
PERF	Performance evaluation
PFLAG	Flag to denote poor performance
RAND	Random number subroutine
SIER	Sign function of $\dot{e}=\frac{\lvert\dot{e}\rvert}{\dot{e}}$
STP	Distance from desired output
TAU	Increment of time
TIME	Time
TYIN(J),J=1,L	TIME for values of y_{in}
YINP(J),J=1,L	Values of y_{in} as a function of time
TOL	Performance tolerance
UU	Control value before incorporating gain
V(J)	Value of regional function J at some point
Y	y = actual output
YDOT	$\dot{y}$
YIN	Value of y_{in} (desired output)
ZK	System gain constant

APPENDIX B

PROGRAM LISTINGS FOR APPENDIX A

Random Number Subroutine

```
  00000  0 00 00000     $RAND  PZE
* 00001  0 43 0 00000          BRM   201SYS
  00002  1 00 0 00041          XSD   TEM
* 00003  0 43 0 00000          BRM   202SYS
                        *
  00004  0 71 0 00033          LDX   N12
  00005  0 41 0 00016   A      BRX   BB
  00006  0 71 0 00034          LDX   N121
  00007  0 76 0 00036   AA     LDA   RANDM
  00010  0 72 0 00026          SKA   N1
  00011  0 01 0 00013          BRU   $+2
  00012  0 76 0 00035          LDA   RN1
  00013  1 40 0 00037          XMP   K
  00014  0 35 0 00036          STA   RANDM
  00015  0 35 0 00043          STA   R2BIT
  00016  0 75 0 00043   BB     LDB   R2BIT
  00017  0 76 0 00023          LDA   ZERO
  00020  0 6700 002            LSH   2
  00021  0 36 0 00043          STB   R2BIT
  00022  0 73 0 00024          SKG   ONE
  00023  0 01 0 00030          BRU   OUT
  00024  0 75 0 00040          LDB   THRE
  00025  0 70 0 00040          SKM   THRE
  00026  0 01 0 00005          BRU   A
  00027  0 76 0 00026          LDA   N1
  00030  0 35 1 00041   OUT    STA   *TEM
  00031  0 37 0 00033          STX   N12
  00032  0 51 0 00000          BRR   RAND
                        *
  00033  77777777       N12    DATA  -1
  00034  77777764       N121   DATA  -12
  00035  37145213       RN1    DATA  037145213
  00036  37145213       RANDM  DATA  037145213
  00037  07346545       K      DATA  07346545
  00040  00000003       THRE   DATA  3
  00041                 TEM    RES   2
  00043  0 00 00000     R2BIT  PZE
         00000026       N1     EQU   026
         00000023       ZERO   EQU   023
         00000024       ONE    EQU   024
                        XMP    OPD   014000000
                        XSD    OPD   010000000
                               END
  00001                 201SYS
  00003                 202SYS
```

```
*     MELPAR, INC.
C
C     HUMAN PERFORMANCE CONTROL AND MONITORING SYSTEM
C
C     CONTRACT NO. NASW 1085
C
      DIMENSION TYIN[75], YINP[75], ALPH[5], BETA[5], V[5], IRU[0/31],
     1ICTR[0/31]
      COMMON TYIN,YINP,ALPH,BETA,V,IRU,ICTR
C
C
C     INPUT INITIAL DATA
C
C     L=NO. OF VALUES YIN CAN ASSUME
C     TYIN,YINP=TIME,F[TIME] FOR YIN
C
  125 READ 126,L
  126 FORMAT [I3]
  130 READ 131,[TYIN[J],YINP[J],J=1,L]
  131 FORMAT[2F10.2]
      TYPE 134
  134 FORMAT[//,7X,4HTIME,7X,3HYIN]
  135 TYPE 136,[J,TYIN[J],YINP[J],J=1,L]
  136 FORMAT [I3,2F10.2]
C
C     READ SLOPES AND INTERCEPTS OF LINES DETERMINING REGIONS
  140 READ 131,[ALPH[J],BETA[J],J=1,4]
      TYPE 141
  141 FORMAT[//,7X,4HALPH,7X,4HBETA]
  145 TYPE 136,[J,ALPH[J],BETA[J],J=1,4]
C
C     READ NO. OF STEPS NEEDED FOR TRAINING
      READ 126, NUMST
  146 TYPE 147,NUMST
  147 FORMAT[//,5X,6HNUMST=,I3]
C
C     IRU[M]=INITIAL CONTROL FOR REGION M
C
      READ 126,[IRU[M],M=0,31]
      TYPE 127
  127 FORMAT[//,2X,$REGION  CONTROL$]
  148 TYPE 149,[M,IRU[M],M=0,31]
  149 FORMAT[5X,I3,7X,I3]
C
C
C
  150 TIME=0.0
  151 DO 154 M=0,31
      ICTR[M]=0
  154 CONTINUE
      Y=0.0
      YDOT=0.0
      PERF=0.0
      IFLAG=0
      I=1
```

```
 155 YIN=YINP[I]
     ER= YIN-Y
     ERDOT = 0.0
C
C
 100 READ 110,A
     READ 110,ZK
     READ 110,CP
     READ 110,TAU
     READ 110,CE
     READ 110, TOL
     TYPE 115, A,ZK,CP,TAU,CE,TOL
     TYPE 120
 110 FORMAT[F13.3]
 115 FORMAT[/,2HA=,F7.3,6H---ZK=,F7.3,6H---CP=,F7.3,7H---TAU=,F7.3,
    16H---CE=,F7.3,7H---TOL=,F7.3]
 120 FORMAT[//,3X,4HTIME,8X,1HU,9X,1HY,7X,5HY DOT,6X,1HE,8X,5HE DOT,
    16X,1HP,8X,4HY IN]
C
C     COMPUTE 1ST EXPECTED PERFORMANCE
     EXPR=EXP[YIN]
     EXPR=ELOG[CP*[1.0-CP/[1.0+CP]]*EXPR]
    1+[CP+1]*[2.0*ELOG[1.0+1.0/CP]+YIN]
     PFLAG=0
     TYPE 160, EXPR
 160 FORMAT[///,$EXPECTED PERFORMANCE=$,F7.3]
C
C
 499 IF[ERDOT] 500,505,510
 500 SIER = -1
     GO TO 520
 505 SIER = 0
     GO TO 520
 510 SIER = 1
 520 ERDAB = ABS[ERDOT]
C
C      DETERMINE REGION M
C
 600 M=0
     DO 620 J=1,5
     IF[J-5] 602,601,602
 601 V[J]=A**2*ER - SIER *ELOG[1.0+A*ERDAB]+A*ERDOT
     GO TO 603
 602 V[J]=ERDOT-ALPH[J]*ER-BETA[J]
 603 IF[V[J]] 605,605,610
 605 V[J]=0
     GO TO 615
 610 V[J]=1
 615 M=M+2**[J-1]*V[J]
 620 CONTINUE
C
C
C
C
C     DETERMINE AUTO OR MANUAL CONTROL
```

```
      C
        170 IF(SENSE SWITCH 4) 200,250
      C
      C          UNDER MANUAL CONTROL
        200 IF(SENSE SWITCH 1) 205,215
        205 IF(SENSE SWITCH 2) 210,220
        210 UU=1.0
            GO TO 700
        215 IF(SENSE SWITCH 2) 220,230
        220 UU=0.0
            GO TO 700
        230 UU= -1.0
      C      MONITOR MANUAL CONTROL
      C
        700 IF(UU-IRU(M)) 730,710,730
        710 IF(ICTR(M)-NUMST) 720,300,300
        720 ICTR(M)=ICTR(M)+1
            GO TO 300
        730 IF(ICTR(M)) 760,740,760
        740 CALL RAND(NEWU)
            IRU(M)=NEWU
            IF(UU-NEWU) 300,750,300
        750 ICTR(M)=2
        760 ICTR(M)=ICTR(M)-1
            GO TO 300
      C
      C         UNDER AUTOMATIC CONTROL
      C
        250 IF(OFLAG) 270,270,260
        260 U=0.0
            GO TO 310
        270 UU=IRU(M)
        300 U=UU*ZK
      C
      C
      C         COMPUTE Y AND YDOT
        310 Y=U/A*(TAU-1.0/A)+(YDOT+A*Y)/A+(U/A**2-YDOT/A)*EXP(-A*TAU)
            YDOT= U/A-(U/A-YDOT/A)*EXP(-A*TAU)
      C
        350 ER=YIN-Y
            ERDOT=-YDOT
      C
            IF(OFLAG) 355,355,375
        355 IF(IFLAG) 360,360,370
        360 PERF=PERF+(CP*ABS(U)+1.0)*TAU
            GO TO 375
        370 PERF=(CP*ABS(U)+1.0)*TAU
      C
      C      OUTPUT DATA FOR THIS LOOP
      C
        375 TIME =TIME+TAU
            IF(SENSE SWITCH 3) 401,400
        400 TYPE 380, TIME,U,Y,YDOT,ER,ERDOT,PERF,YIN
        380 FORMAT(8F10.4)
        401 IF(TIME-TYIN(I+1)) 420,410,410
```

```
  410 YIN=YINP[I+1]
      I=I+1
      IFLAG=1
C
C     COMPUTE EXPECTED PERFORMANCE
      EXPR=EXP[YIN]
      EXPR=ELOG[CP*[ 1.0-CP/[ 1.0+CP]]*EXPR]
     1+[ CP+1]*[ 2.0*ELOG[ 1.0+1.0/CP]+YIN]
      PFLAG=0
      TYPE 160, EXPR
C
      GO TO 430
  420 IFLAG=0
C
  430 STP=ERDOT**2+ER**2
      IF[STP-CE] 450,455,455
  450 OFLAG=1
      GO TO 460
  455 OFLAG=0
  460 IF[[PERF-EXPR]-TOL] 499,470,470
  470 PFLAG=PFLAG+1
      IF[PFLAG-1] 499,480,499
  480 TYPE 485
  485 FORMAT[//,$-PERFORMANCE IS LOUSY$]
      GO TO 499
      STOP
*END
```

COMMON ALLOCATION

```
77552 TYIN      77324 YINP      77312 ALPH      77300 BETA
77266 V         77226 IRU       77166 ICTR
```

PROGRAM ALLOCATION

```
00022 L         00023 J         00024 NUMST     00025 M
00026 IFLAG     00027 I         00030 NEWU      00031 TIME
00033 Y         00035 YDOT      00037 PERF      00041 YIN
00043 ER        00045 ERDOT     00047 A         00051 ZK
00053 CP        00055 TAU       00057 CE        00061 TOL
00063 EXPR      00065 PFLAG     00067 SIER      00071 ERDAB
00073 UU        00075 OFLAG     00077 U         00101 STP
```

SUBPROGRAMS REQUIRED

```
EXP      ELOG      ABS      RAND
```