MPS/LIN - Note 71-11
5.8.1971. GJJ/fm

INTERACTIVE SIMULATION

G.J. Jennings

SYNOPSIS

The purpose of this note is to remind the reader of a potential application of digital computers in an area of interest to control engineers. In the Appendix, a working example of an interactive simulation is given which can be used to demonstrate the advantages of working on-line in an interactive manner with the computer. The example may also be used as a cook-book recipe for incorporation of graphics into MIMIC.

ON SIMULATION AND COMPUTERS

The engineer is often interested in the time solution of a set of differential equations, and frequently would like to examine the effect of changes in system parameters on this solution. In the past the analog computer was widely used for this sort of analysis, there being no other alternative. Recent advances in digital computer technology have however opened up new possibilities, and the digital machine can now be profitably used to solve particular classes of simulation problems. To be more specific, the real-time capability and high speed of the third generation computer coupled with modern display systems have improved the on-line 'conversational' ability of the machine and possible to solve differential equations in such a manner that the user might consider himself as operating a pseudo-analog computer.

The digital computer has some distinct advantages over the analog. There is no need to be concerned with scaling of the model variables - which can be quite time-consuming and which may necessitate a digital computer time solution anyway -, and the digital machine can handle function generation and non-linearities in a much more flexible and straight-forward fashion. However the analog computer can claim significant speed advantages over the digital computer, making it attractive for the 're-op' (iterative) type of problem (providing the model has only a few well defined non-linearities). For problems where the user wishes to interact with the model after each time solution the slower relative speed of the digital machine may be an insignificant disadvantage. A delay of ten seconds awaiting the display of a new solution is usually acceptable and can enable the computer to obtain time solutions for a wide range of small to medium size models, particularly if an efficient integration algorithm with self adjusting time steps is used.

It need hardly be stressed that on-line interactive simulation soon loses its appeal to engineers if vast amounts of time have to be spent on programming each application. In essence, if on-line digital simulation is to be acceptable, a mechanism must be provided enabling the simulation model to be easily programmed (i.e. a simulation language), and software must exist to enable easy on-line user interaction with the model.

RECENT DEVELOPMENT

At this stage it is appropriate to mention advances made over the past few years which partly fulfill the above requirements and to comment on the facilities currently available at CERN.

The desire to rapidly program a set of dynamic equations for digital computer solution has led to the development of a number of Simulation Languages. One of the most well-known of these, MIMIC[1,2], has been implemented by CDC on the 6600 series computer, and is available at CERN. Unfortunately the CDC version needs 33K core storage, and with SJOB status, turn-around time is long. Since XJOB status for quick turn-around is highly desirable a modified version of MIMIC is available[3] which uses overlays to reduce the execution space below the 30K limit. This modified version retains the full flexibility and capability of the original version, and with turn-around times of 2-5 minutes at Remote Input Stations, may be used in a semi-interactive manner.

The technology of graphical displays is advancing rapidly. CERN has a 3200 Processor/Display facility and an ARGUS computer with display which may be linked to the 6600 computer for on-line Interactive Graphics. Control of the latter may be established in Fortran by using the GD3 software package[4]. More recently the introduction of the 'MINING' software[5] has provided an extremely simple-to-use set of subroutine calls which enable the Fortran user to rapidly gain experience with graphical display.

With regard to the requirements of an on-line interactive simulation facility, which is a logical step from the foregoing developments, it is understood[6] that already CERN has contacted American laboratories on the subject of adding to MIMIC an interactive graphics capability. In addition a method of coupling MIMIC and MINING has been derived (see Appendix), which may be used to demonstrate interactive simulations. The method of coupling may also be used (with GD3 calls) to set up picture frames for later display on the Tektronix storage scope.

CONCLUSION

Interactive simulation has been outlined as a possible application of the digital computer. With the days of the 7600 computer rapidly approaching, with no analog computer on site, and with the DD Division thinking about future developments, it is suggested that this application of the digital computer should merit some consideration.

REFERENCES

1. MIMIC. Digital Simulation Language Reference Manual (1968). (CDC Doc. 44610400).

2. Digital Simulation of Continuous Systems. Chu.(1969). McGraw-Hill.

3. Memorandum. Modifications to MIMIC. G.J. Jennings.

4. Graphic Display Systems (GD3). Program Library Long Write-Up J510.

5. MINING. Users' Manual. Lyle Smith. Data Handling Division Report. DD/DH/70/23.

6. Tel. conversation with J. Altaber. DD Division.

Distribution

DD, MPS and SI Group Leaders
Data Handling Policy Group
G. Brianti
G.R. MacLeod
P.H. Standley

APPENDIX

DEMONSTRATION EXAMPLE

The simulation model is specified in MIMIC language, and MINING is used for the interactive graphics. The example may be used as a cook-book recipe for incorporating GD3 calls into MIMIC. In this fashion results may be displayed, under the control of FOCUS on the storage scope.

DESCRIPTION

The MIMIC language describes a control system with a conventional 3 term controller and a second order system in the forward loop (time constants 1 sec. and .5 sec.), and a delay of .25 seconds in the feed-back loop (see Fig. 1.).

Interactive graphical display is achieved by calls to the link function subprogram SR1, which is coded in Fortran and which, in turn, makes calls on the MINING software package (see Figs. 2,3,4).

NOTES ON THE SR1 CALLS

MIMIC was originally intended for batch operation and to ensure proper sequential execution of the interactive graphical display requests, some tricks have to be incorporated.

1. MIMIC has to be deceived into thinking that it is going to be making a series of runs - hence the PAR card.

2. After reading the first PAR data card however, no others are read and new parameter values are fed back into MIMIC by the third call to SR1. Further reads of PAR data cards are stopped by setting the logical control variable READ to false (+1).

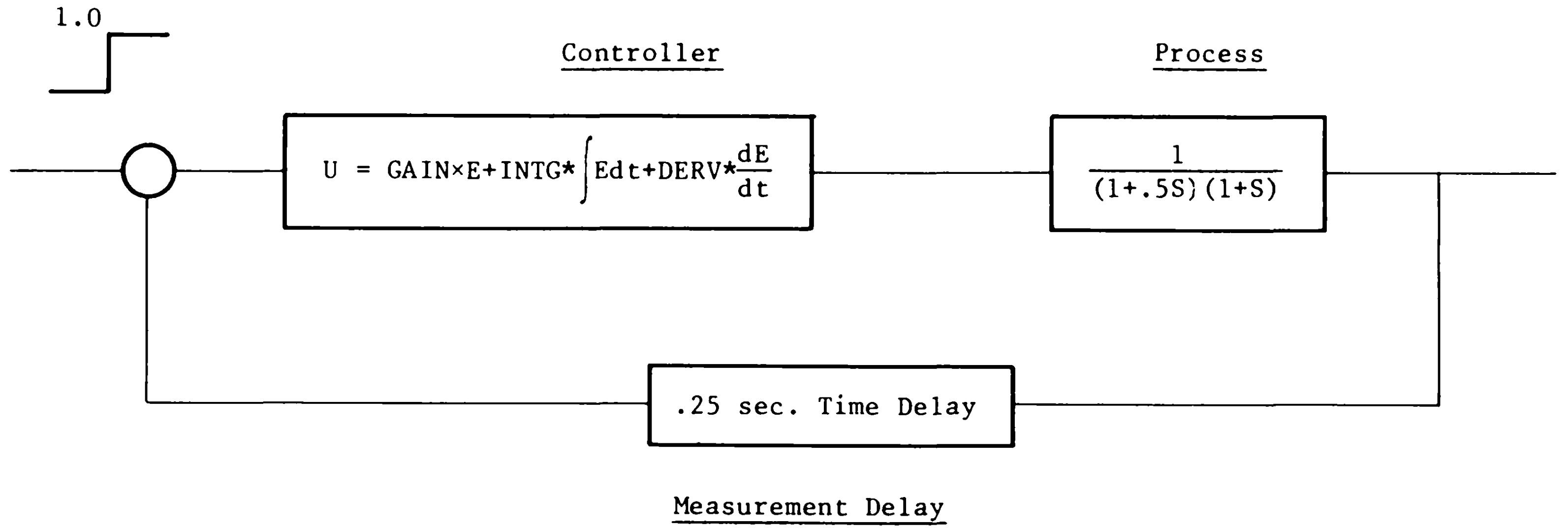

Fig. 1. Demonstration Example

```
                    CON(PARAD1,PARAD2,PARAD3,DUM)
                    CON(TC1,TC2,ONE,TWO,THREE,READ)
 READ               PAR(G,RESET,RATE)
C                                               INITIALISE DISPLAY
 READ               SP1(ONE,DUM,DUM,G,RESET,RATE)
          DT        .1
          DTMIN     .001
C                                               SYSTEM EQUATIONS
          X1        INT((INP-X1)/TC1,0.)
          X2        INT((X1-X2)/TC2,0.)
          Y         TDL(X2,.25,100.)
          ERR       1.-Y
C                                               SYSTEM CONTROL
          INP       G*ERR+RESET*INT(ERR,0.)+RATE*DER(T,ERR,0.)
C                                               BUILD-UP OF DISPLAY ARRAY
          READ      SP1(TWO,T,X2,DUM,DUM,DUM)
C                                               DISPLAY CONTROL
          DISP      FIN(T,10.0)
 DISP     RAD1      SP1(THREE,DUM,DUM,PARAD1,PARAD2,PARAD3)
 DISP     G         PARAD1+RAD1-RAD1
 DISP     RESET     PARAD2+RAD1-RAD1
 DISP     RATE      PARAD3+RAD1-RAD1
 DISP     READ      RAD1
                    END
```

Fig. 2. MIMIC Source Program

3. Since MIMIC only allows function subprograms, strictly speaking only one argument can be returned to the MIMIC calling program at a time. This restriction however can be remedied by the manner shown. Multiple calls could also be used.

4. MIMIC features parallelism. The statements defining a program may be written in any order. Within MIMIC there is a sorting routine which decides on the appropriate final execution of the commands. The statements ... + PAD1 - PAD1 are a safety insertion to ensure that these commands are executed after PAD1 is evaluated, i.e. that new values for G, RESET and RATE are established after the call to SR1.

5. MIMIC has a self adjusting time-step and the variables T and X2 go through a series of updates between each print-out interval DT. To set up an array of values at intervals DT, the subprogram SR1 examines the switch IOUT, which becomes non-zero at print-out intervals.

6. The flag MASK ensures that no array building commences until the Display System and the array argument pointer I have been initialised.

7. The process of argument transfer between MIMIC program and subprogram SR1 can also be augmented by reading and writing into the MIMIC common array R

R(1)	contains	T
R(2)	"	DT
R(3)	"	DTMAX
R(5)	"	DTMIN

The abridged version of MIMIC has blank common laid out in the following manner

```
COMMON P(95), R(2,500), S(2,500), FF(4100).
```

8. The Function Language Program generated by MIMIC is of use in determining the order in which statements are executed, and also gives information on where the independent variables are stored in the S and R arrays.

9, The demonstration example uses calls to MINING. One may, of course, make calls direct to the GD3 package instead, and, if desired, generate picture frames for later display on the storage scope. Alternatively one may use GD3 to set up interaction with ARGUS in a manner different from MINING. However this will take significant software effort.

10. Cautionary Note. The demonstration program has excessive requirement (38656 words) and with the current computer facilities at CERN is not recommended of general use. The heavy consumption of core is unavoidable. The original version of MIMIC takes approx. 33K words. The version used in the demonstration takes 27,382 words, which was achieved by trimming back on array sizes and removing unwanted line-printer plotting routines. A further reduction of 4K words may be achieved by simple overlaying, and possible another K may be saved by further cut-backs on array sizes.

CONTROL CARDS

```
QJOB. ________,________,_____
MACHINE (6600)
ASSIGN AR, XXXX.
FUN(S)
SPACE(38660)
LØAD(INPUT)
LGO.
E of R card
SR1 Fortran Deck
E of R card
Mining Relocatable Object Deck
Mimic Relocatable Object Deck
E of R card
Mimic Source Language Deck
Data Deck
E of F card
```

```
                                  ***FUNCTION-LANGUAGE PROGRAM GENERATED***

IFN     LCV         RESULT      FTN     A         B         C         D         E         F


 1                              CON     PARAD1    PARAD2    PARAD3    DUM
 2                              CON     TC1       TC2       ONE       TWO       THREE     READ
 3      READ                    PAR     G         RESET     RATE
 4                  DT          EQL     .1
 5                  DTMIN       EQL     .001
 6                  (010)       SUB     X1        X2
 7                  (011)       DIV     (010)     TC2
 8                  X2          INT     (011)     0.
 9                  Y           TDL     X2        .25       100.
10                  ERR         SUB     1.        Y
11                  (015)       INT     ERR       0.
12                  (016)       DER     T         ERR       0.
13                  (017)       MPY     G         ERR
14                  (018)       MAD     RESET     (015)     (017)
15                  INP         MAD     RATE      (016)     (018)
16                  READ        SR1     TWO       T         X2        DUM       DUM       DUM
17                  DISP        FIN     T         10.0
18      DISP        PAD1        SR1     THREE     DUM       DUM       PARAD1    PARAD2    PARAD3
19      DISP        (023)       ADD     PARAD1    PAD1
20      DISP        G           SUB     (023)     PAD1
21      DISP        (025)       ADD     PARAD2    PAD1
22      DISP        RESET       SUB     (025)     PAD1
23      DISP        (027)       ADD     PARAD3    PAD1
24      DISP        RATE        SUB     (027)     PAD1
25      DISP        READ        EQL     PAD1
26      READ                    SR1     ONE       DUM       DUM       G         RESET     RATE
27                  (007)       SUB     INP       X1
28                  (008)       DIV     (007)     TC1
29                  X1          INT     (008)     0.
30                              END
```

Fig. 3. MIMIC Function Language Program

```
      FUNCTION SP1(FLAG,T,Y,PARA1,PARA2,PARA3)
      DIMENSION XX(250),FX(250),VAL(3)
      COMMON /SWITCH/ IOUT,DUMMY(8)
      EXTERNAL TVARGS
      IFLAG = FLAG+.1
      GO TO (1,2,3), IFLAG
      FIRST ENTRY-INITIALISE
    1 I=0
      MASK = 1
      SP1 = 0
      CALL TVBGN(6,TVARGS)
      CALL INITLZ
      VAL(1)=PARA1
      VAL(2)=PARA2
      VAL(3)=PARA3
      RETURN
      CONSTRUCTION OF ARRAY
    2 IF (MASK.NE.1) GO TO 6
      SP1 = 1
      IF (IOUT.EQ.0) RETURN
      I = I+1
      XX(I)=T
      FX(I)=Y
      RETURN
      GENERATION OF PICTURE AND DISPLAY
    3 CALL PLOT(XX,FX,DUM,I,1)
      CALL TUTOR(17,17HMIMIC-MINING TEST)
      CALL PARAMS(3,4,3,12HPROPINTGDERV,VAL)
      CALL DSPLAY(ISW,ICHOIC,VAL,DUM,DUM)
      GO TO (4,4,5), ISW
      CONTINUE,REINITIALISE,AND RETURN ARGUMENTS
    4 PARA1 = VAL(1)
      PARA2 = VAL(2)
      PARA3 = VAL(3)
      I = 0
      CALL INITLZ
      SP1 = 1
      RETURN
      TERMINATE PROGRAM
    5 CALL TVEND
      STOP
    6 SP1 = -1
      RETURN
      END
```

Fig. 4. Fortran Link Routine