"Tutorial Input"—Standardizing the Computer/Human Interface

A. I. Zygielbaum
Communications Systems Research Section

This article describes a new technique for implementing a human/computer interface for computer-based subsystems for the DSN. Known as "tutorial input," this technique provides convenient short input procedures for the experienced operator and a helping hand for the novice. From the programmer's viewpoint, the technique is implemented in a compact, modular, easily modified table-driven structure. The technique has been successfully used through two generations of R&D ranging machines.

I. Motivation

Though much time is spent by programmers in producing efficient, clean, and ego-pleasing code, very little time is spent in developing an efficient, reliable human/computer interface. From an operational standpoint, this interface is the most important and least understood in DSN subsystem programming. With a view toward minimizing operator error and increasing subsystem efficiency, this article will present the technique for human/computer communication successfully used in two generations of R&D ranging systems.

In a typical subsystem program, the operator must provide operating parameters, critical times and perhaps limits to the software. Two interface techniques are generally used. With the first technique the operator may be queried on an input-by-input basis. For instance,[1]

ENTER T1: ↵
26
ENTER T2: ↵
15
ENTER T3: ↵
15

A second technique is the preset format wherein the operator must enter numbers in accordance with some specified template. An analogous entry to the one given above could be

*/26/15/15$

The first technique has the advantage that a format need not be memorized or followed. Given the parameters, an operator just follows directions. The disadvantage is that at the 10 characters per second of the usual tele-

[1]Computer typeouts are underlined.

typewriter, the questions sometimes waste an unacceptable amount of time. The second technique is clearly faster but requires a thorough a priori knowledge of the format. This leads to increased training time and a greater chance of operator error from misplaced fields.

Overriding the problems and tradeoffs inherent in these techniques is the multitude of programs involved in a typical DSN operation. An operator may have to communicate with a variety of programs, each designed with a different input philosophy and format. This situation naturally leads to an increased probability of human error.

In a real-time mission environment, an incorrect program entry can be just as disastrous as an equipment malfunction. A specific example occurred to ranging during the Mariner Mars 1971 (MM'71) mission. The software for command and telemetry running on the XDS 920 Telemetry and Command Processors (TCPs) used a dollar sign for a line terminator, whereas the software for the Mu ranging system used a carriage return. On at least two occasions, range data were lost because the operator typed a dollar sign at the end of an input line and walked away thinking his task complete. The ranging software waited the time-out period and then cancelled the necessary input. A simple modification to the software to allow it to recognize both a dollar sign and carriage return as a terminator saved a significant amount of data.

II. Proposal

Usually the subsystem operator is treated as a button-pusher who is taught to run specific software and devices. This view was believed incorrect, and software to support both the Mu-I and Mu-II ranging systems was developed with the direct cooperation and interaction of the operators. The input routine devised in this effort has been shown to be easy to learn as well as easy to operate. It uses a table-driven structure which makes it highly visible to the programmer who must implement it and relatively easy to modify and enlarge. This algorithm, known as "tutorial input," is presented here.

III. Tutorial Input as Viewed by an Operator

There are basically two parts to an input using the "tutorial" technique. First, a command is typed to designate the type of entry; second, the parameter or parameters are entered. For clarity, consider the commands used with the Mu-II system given in Table 1. (Parameter definitions are given in Table 2 for completeness.)

To input commands and data, the operator first notifies the PDP 11/20 that command input is desired by pressing an interrupt button (this could be a breakpoint on the XDS 9-series machines). When the machine responds with a pound sign (#) he types an input line terminated with a carriage return. The first field is a command followed by a slash (/). If the command is acceptable, the software looks to see if anything else is in the input line. If more characters have been typed, they are used. If not, the operator is queried by a specific message for the appropriate parameter of the now-active command. This is true every time a field containing a command or parameter has been used. If more characters are in the input line, they are processed; if not, and more parameters are required, the operator is queried.

The field delimiters used are a slash between a command and parameters and commas between multiple parameters. Note that the whole line is used. This allows commands to be entered contiguously.

The following are equivalent ways to initialize all ranging parameters:

Example 1:

#A/ ↵
TOF: ↵
#1200 ↵
SYN FREQ: ↵
44.01234 ↵
T1,T2,T3,TC: ↵
20,20,20,12 ↵
C1,C2,CN: ↵
4,19,3 ↵
MODE: ↵
- ↵

Example 2:

#A/1200,44.01234,20,20,20,12,4,19,3,- ↵

Example 3:

#A/1200,44.01234,20,20,20 ↵
TC: ↵
#12,4,19 ↵
CN: ↵
#3,- ↵

Example 1 shows how the operator can be led through the input parameters. In Example 2, an experienced operator has entered all parameters without software prompting. The operator in example 3, after losing his place, has let the computer request the remaining required numbers.

Multiple commands may be entered on a single line. In order to change the mode and number of components, for instance, one could enter

#M/124,C/3,10,50 ↵

or

#M/124,C/ ↵
C1,C2,CN: ↵
#3,*,50 ↵

The asterisk (*) entered for C2 causes the previous value of C2 to remain unchanged.

A further provision to ease the operator's task is error correction. Input cancellation (control R—R^c) results in immediately exiting the input routine. Character deletion (C^c) causes the program to type ← and results in the deletion of the last character entered into the string. Character deletion may be used more than once, e.g., to delete the last three characters so that they can be retyped. Line deletion (E^c) deletes the line, upspaces, types →, and allows the whole line to be retyped. Error correction can also be accomplished by use of the command to change a particular parameter.

IV. Tutorial Input as Viewed by a Programmer

Although written in machine language for a PDP-11/20, via the SAPDP Xerox Sigma 5 cross assembler (Ref. 1) the routine is amenable to coding in another machine language or in a higher level language such as BASIC or FORTRAN. In this discussion the interaction of the input routine with other real-time processors will not be covered. The Mu-II software required interfaces to a teletype output routine and to a real-time scheduler. These topics will be documented in a forthcoming article on the Mu-II system software.

Though perhaps not meeting the letter of structured programming, which is difficult, if not impossible, in a real-time environment, tutorial input does realize all advantages normally claimed for structured programming. Through the use of a table-driven technique, the input sequencing and programming is straightforward, easily modified, and simply documented.

The technique involves two tables. The first table is the command list. Each entry in the list takes three words in the Mu-II realization. There is one entry for each command. The first word has the two characters of the command, e.g., "A/" or "TF," the second a pointer to a word in the second table, and the third word a number equal to the number of parameters to be entered with the particular command.

The second table is the parameter list. Each entry in this list takes four words. There is one entry for each parameter. The first word of the entry contains the number of characters in the message associated with each parameter, while the second contains the message location. (The messages are the queries shown earlier, e.g., TOF: ↵.) The third word gives the location of the decoding subroutine (integer, floating point, etc.) to convert the parameter's ASCII character string to binary. The fourth and final word contains the location of the destination for the binary number. Figure 1 contains a listing of the two tables in the Mu-II software.

The algorithm which interprets these tables is described structurally in Fig. 2 and in a detailed flow chart in Fig. 3.

As an example of the algorithm, let us consider that a command "T/" has been input and follow the algorithm operation. Scanning the command list (Fig. 1), a match is found (line 334) giving the parameter table location as SCT1 and the number of parameters to be input as four. The program looks to see if there are any more characters in the input string. If not, the message TMSG, containing 14 characters, is typed as shown by the SCT1 entry in the parameter list. If input already exists, the message is skipped. In either case, the input parameter is processed by INGR which is the integer decoding subroutine and the result stored in IT1.

Thus far, one parameter has been processed and three more are left. The program proceeds directly to the SCT2 entry in the parameter table and repeats the input process. This continues until all parameters are entered.

Once again the algorithm checks to see if there are more characters in the input stack. If there are none, the routine exits. If there are more, the first field is used to search the command stack as the program assumes that another command is in the input string. The process goes on until an error occurs or until the input line is exhausted.

The input logic flow is readily observable by following the driving tables. The tables are also a simple documentation of the input sequences. The ease with which commands and entries can be modified or deleted is shown by a program change that occurred during the Mariner Venus/Mercury 1973 (MVM'73) mission.

As originally written, an operator could modify any parameter or set of parameters individually except for time of flight (TOF). The only way to change this parameter was through the "enter all parameters" A/ command. As the mission proceeded, the TOF changed by several seconds each day. It was obvious that going through the entire initialization sequence to change TOF was a waste of time. The decision was made to add the TF/ command so that TOF could be changed individually.

Because of the table-driven structure, only three cells were actually needed. These were, from Fig. 1, lines 346–348:

"TF"	Command
SCTOF	Parameter Table Pointer
1	Number of parameters to be entered.

This simple modification changed a tedious operation into a trivial one.

V. Summation

Data will continue to be lost due to incompatibilities, indeed contradictions, between input formats in various DSN software systems. This coupled with the trend toward smaller station operational crews, automation, and the continued proliferation of minicomputers in the DSN makes standardization increasingly important. The algorithm presented herein is a candidate to aid in that task.

A final comment, tutorial input has been used in ranging software throughout the MVM'73 mission. It has shown itself to be an easy-to-use as well as an easy-to-learn input technique. The extra time taken to interact with experienced ranging operators during its development has been paid back many fold through simplified training and the low probability of human error. This programming effort clearly shows the value of bringing the system operators into the software design process at a very early stage. Many times the operators were able to point out features which were not helpful and could be discarded as well as request features which would simplify their task. The success of the Mu-II system programming is largely due to this cooperation between the programmer and the system operators.

Reference

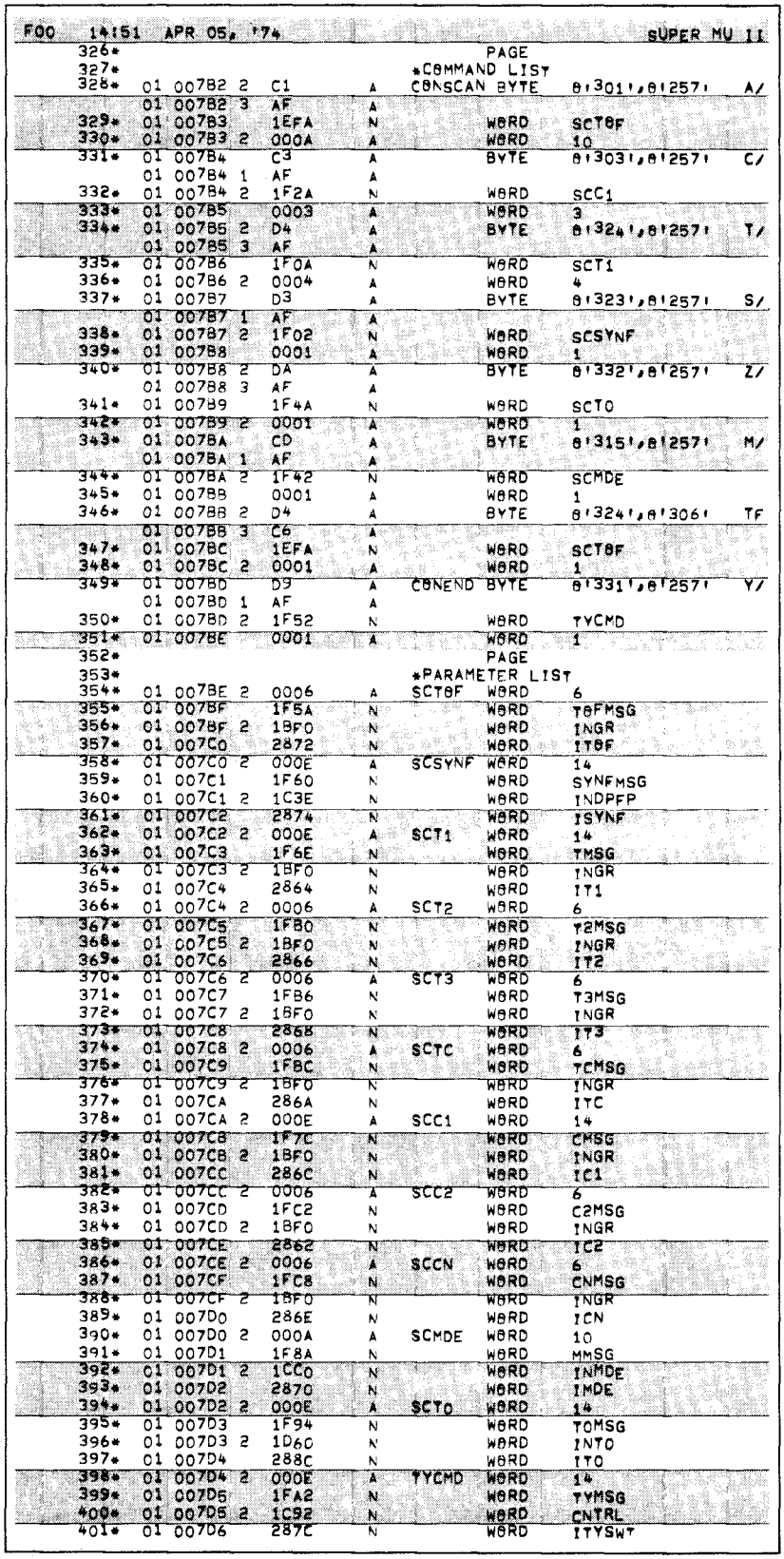
1. Erickson, D. E., "The SAPDP Program Set for Sigma 5 Assembly," in *The Deep Space Network Progress Report*, Technical Report 32-1526, Vol. VII, pp. 91–96, Jet Propulsion Laboratory, Pasadena, Calif., Feb. 15, 1972.

Table 1. Mu-II commands

Command	Parameter to be entered
A/	All operational parameters: TOF, SYN FREQ, T1,T2,T3,TC,C1,C2,CN,MODE
TF/	TOF
C/	C1,C2,CN
T/	T1,T2,T3,TC
S/	SYN FREQ
M/	MODE
Z/	Requested TØ time
Y/	Typewriter printout ON/OFF: i.e., Y/ON Y/OFF

Table 2. Parameter definitions

Abbreviation	Meaning
TOF	Round-trip light time
SYN FREQ	Exciter synthesizer frequency
T1	First component integration time
T2	Lower-frequency components integration time
T3	Post-acquisition DRVID integration time
TC	10-MHz calibration integration time
C1	Highest-frequency component
C2	Lowest-frequency component
CN	Number of post-acquisition DRVID points before automatic reacquisition
MODE	Select configuration (i.e., Block III or IV receiver phasing, reacquisition with or without coder sync, etc.) "-" results in standard configuration
TØ	Coder synchronization time



```
FOO   14:51  APR 05, '74                                        SUPER MU II
     326*                                    PAGE
     327*                          *COMMAND LIST
     328*  01 007B2 2  C1        A  COMSCAN BYTE   8'301',8'257'   A/
           01 007B2 3  AF        A
     329*  01 007B3    1EFA      N          WORD   SCT0F
     330*  01 007B3 2  000A      A          WORD   10
     331*  01 007B4    C3        A          BYTE   8'303',8'257'   C/
           01 007B4 1  AF        A
     332*  01 007B4 2  1F2A      N          WORD   SCC1
     333*  01 007B5    0003      A          WORD   3
     334*  01 007B5 2  D4        A          BYTE   8'324',8'257'   T/
           01 007B5 3  AF        A
     335*  01 007B6    1F0A      N          WORD   SCT1
     336*  01 007B6 2  0004      A          WORD   4
     337*  01 007B7    D3        A          BYTE   8'323',8'257'   S/
           01 007B7 1  AF        A
     338*  01 007B7 2  1F02      N          WORD   SCSYNF
     339*  01 007B8    0001      A          WORD   1
     340*  01 007B8 2  DA        A          BYTE   8'332',8'257'   Z/
           01 007B8 3  AF        A
     341*  01 007B9    1F4A      N          WORD   SCT0
     342*  01 007B9 2  0001      A          WORD   1
     343*  01 007BA    CD        A          BYTE   8'315',8'257'   M/
           01 007BA 1  AF        A
     344*  01 007BA 2  1F42      N          WORD   SCMDE
     345*  01 007BB    0001      A          WORD   1
     346*  01 007BB 2  D4        A          BYTE   8'324',8'306'   TF
           01 007BB 3  C6        A
     347*  01 007BC    1EFA      N          WORD   SCT0F
     348*  01 007BC 2  0001      A          WORD   1
     349*  01 007BD    D9        A  COMEND  BYTE   8'331',8'257'   Y/
           01 007BD 1  AF        A
     350*  01 007BD 2  1F52      N          WORD   TYCMD
     351*  01 007BE    0001      A          WORD   1
     352*                                    PAGE
     353*                          *PARAMETER LIST
     354*  01 007BE 2  0006      A  SCT0F   WORD   6
     355*  01 007BF    1F5A      N          WORD   T0FMSG
     356*  01 007BF 2  1BF0      N          WORD   INGR
     357*  01 007C0    2872      N          WORD   IT0F
     358*  01 007C0 2  000E      A  SCSYNF  WORD   14
     359*  01 007C1    1F60      N          WORD   SYNFMSG
     360*  01 007C1 2  1C3E      N          WORD   INDPFP
     361*  01 007C2    2874      N          WORD   ISYNF
     362*  01 007C2 2  000E      A  SCT1    WORD   14
     363*  01 007C3    1F6E      N          WORD   TMSG
     364*  01 007C3 2  1BF0      N          WORD   INGR
     365*  01 007C4    2864      N          WORD   IT1
     366*  01 007C4 2  0006      A  SCT2    WORD   6
     367*  01 007C5    1FB0      N          WORD   T2MSG
     368*  01 007C5 2  1BF0      N          WORD   INGR
     369*  01 007C6    2866      N          WORD   IT2
     370*  01 007C6 2  0006      A  SCT3    WORD   6
     371*  01 007C7    1FB6      N          WORD   T3MSG
     372*  01 007C7 2  1BF0      N          WORD   INGR
     373*  01 007C8    2868      N          WORD   IT3
     374*  01 007C8 2  0006      A  SCTC    WORD   6
     375*  01 007C9    1FBC      N          WORD   TCMSG
     376*  01 007C9 2  1BF0      N          WORD   INGR
     377*  01 007CA    286A      N          WORD   ITC
     378*  01 007CA 2  000E      A  SCC1    WORD   14
     379*  01 007CB    1F7C      N          WORD   CMSG
     380*  01 007CB 2  1BF0      N          WORD   INGR
     381*  01 007CC    286C      N          WORD   IC1
     382*  01 007CC 2  0006      A  SCC2    WORD   6
     383*  01 007CD    1FC2      N          WORD   C2MSG
     384*  01 007CD 2  1BF0      N          WORD   INGR
     385*  01 007CE    2862      N          WORD   IC2
     386*  01 007CE 2  0006      A  SCCN    WORD   6
     387*  01 007CF    1FC8      N          WORD   CNMSG
     388*  01 007CF 2  1BF0      N          WORD   INGR
     389*  01 007D0    286E      N          WORD   ICN
     390*  01 007D0 2  000A      A  SCMDE   WORD   10
     391*  01 007D1    1F8A      N          WORD   MMSG
     392*  01 007D1 2  1CC0      N          WORD   INMDE
     393*  01 007D2    2870      N          WORD   IMDE
     394*  01 007D2 2  000E      A  SCT0    WORD   14
     395*  01 007D3    1F94      N          WORD   T0MSG
     396*  01 007D3 2  1D60      N          WORD   INT0
     397*  01 007D4    288C      N          WORD   IT0
     398*  01 007D4 2  000E      A  TYCMD   WORD   14
     399*  01 007D5    1FA2      N          WORD   TYMSG
     400*  01 007D5 2  1C92      N          WORD   CNTRL
     401*  01 007D6    287C      N          WORD   ITYSWT
```

Fig. 1. Mu-II software listing

```
Command, IF command request DO.
      Save current parameters.
      Input line.
      DO UNTIL input stack empty.
            Separate next input field.
            Search command table for match.
            IF NOT match.
                  Notify operator.
                  EXIT command.
            ELSE.
                  Location ← command table (match +1).
                  Count ← command table (match +2).
         Parameter, DO UNTIL count = 0.
            If input stack NOT EMPTY.
                  Separate next input field.
            ELSE
                  Print parameter table (location) characters from
                    string addressed by parameter table (location +1).
                  Input line.
            Location ← location +2.
            DO SUBROUTINE addressed by parameter table (location).
            Location ← location +1.
            Store result in parameter addressed by parameter table (location)
            Location ← location +1.
            Count ← count -1
         END
      END
END
```

Fig. 2. Structured representation

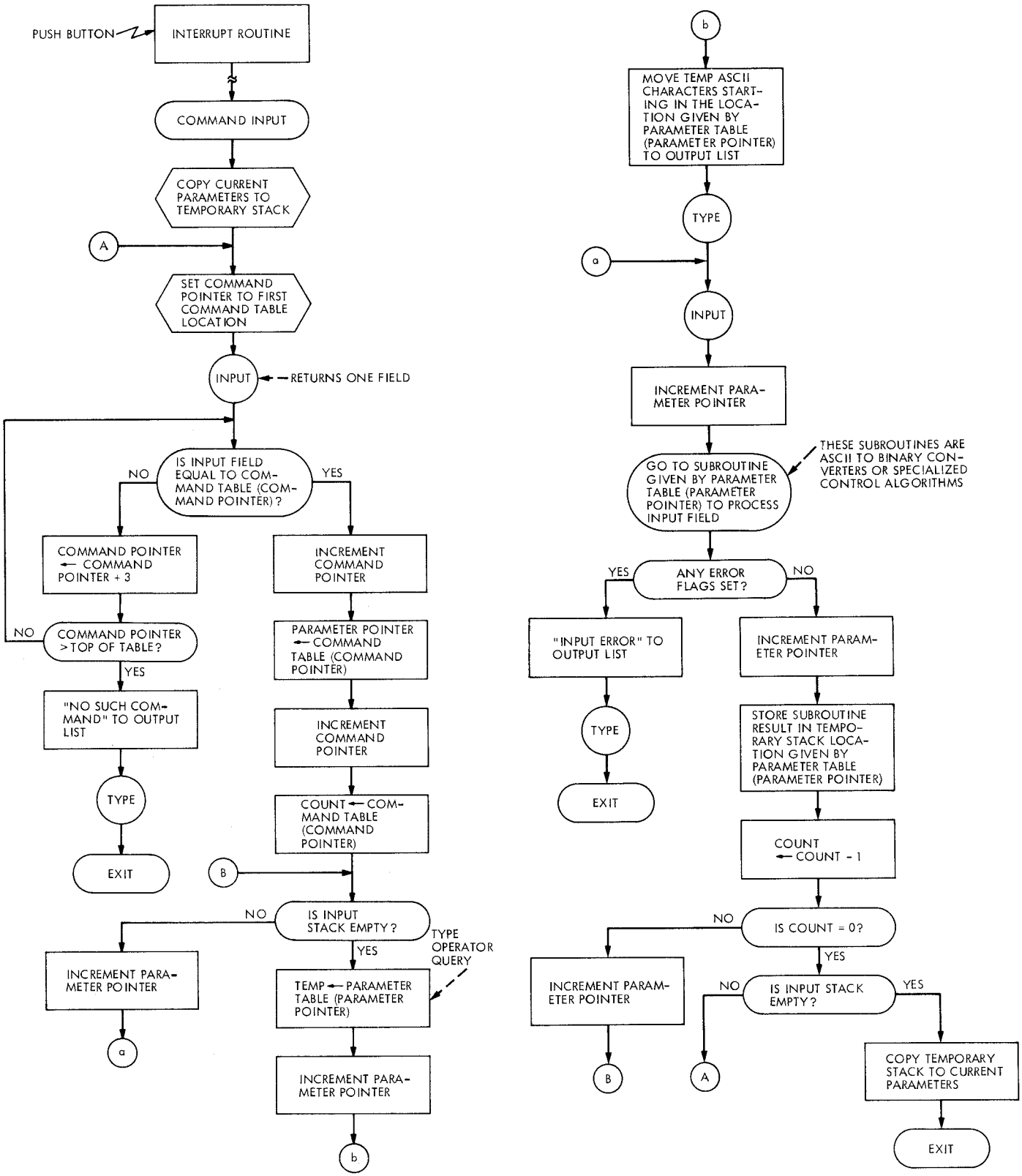


Fig. 3. Tutorial input schematic flowchart (Mu-II realization)

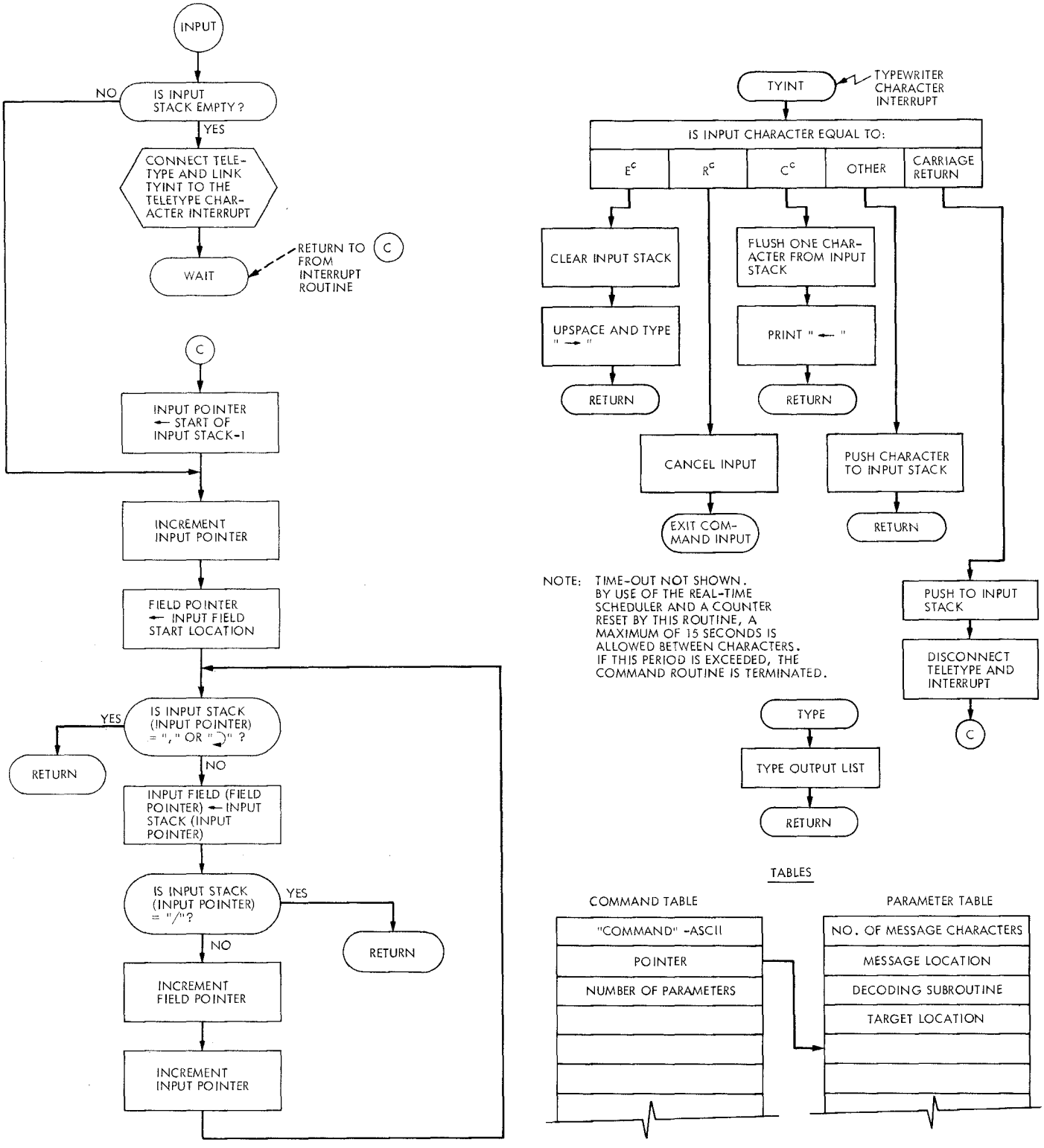

INPUT
NO
IS INPUT STACK EMPTY?
YES
CONNECT TELETYPE AND LINK TYINT TO THE TELETYPE CHARACTER INTERRUPT
WAIT
RETURN TO FROM INTERRUPT ROUTINE
C
INPUT POINTER ← START OF INPUT STACK-1
INCREMENT INPUT POINTER
FIELD POINTER ← INPUT FIELD START LOCATION
IS INPUT STACK (INPUT POINTER) = "," OR "⏎" ?
YES
RETURN
NO
INPUT FIELD (FIELD POINTER) ← INPUT STACK (INPUT POINTER)
IS INPUT STACK (INPUT POINTER) = "/"?
YES
RETURN
NO
INCREMENT FIELD POINTER
INCREMENT INPUT POINTER
TYINT
TYPEWRITER CHARACTER INTERRUPT
IS INPUT CHARACTER EQUAL TO:
E^C
R^C
C^C
OTHER
CARRIAGE RETURN
CLEAR INPUT STACK
UPSPACE AND TYPE " → "
RETURN
FLUSH ONE CHARACTER FROM INPUT STACK
PRINT " ← "
RETURN
CANCEL INPUT
EXIT COMMAND INPUT
PUSH CHARACTER TO INPUT STACK
RETURN
PUSH TO INPUT STACK
DISCONNECT TELETYPE AND INTERRUPT
C
NOTE: TIME-OUT NOT SHOWN. BY USE OF THE REAL-TIME SCHEDULER AND A COUNTER RESET BY THIS ROUTINE, A MAXIMUM OF 15 SECONDS IS ALLOWED BETWEEN CHARACTERS. IF THIS PERIOD IS EXCEEDED, THE COMMAND ROUTINE IS TERMINATED.
TYPE
TYPE OUTPUT LIST
RETURN
TABLES
COMMAND TABLE
"COMMAND" -ASCII
POINTER
NUMBER OF PARAMETERS
PARAMETER TABLE
NO. OF MESSAGE CHARACTERS
MESSAGE LOCATION
DECODING SUBROUTINE
TARGET LOCATION

Fig. 3 (contd)