LOCUS, Simulator for logic circuit education

Nobuo Yoshioka* and Susumu Matoba*

1. INTRODUCTION

LOCUS (Logic Circuit Simulator) allows each student to activate, test and modify a small synchronous logic circuit which has been designed by himself, interactively.

Each student puts his circuit into operation in the course of the conversation with LOCUS as follows. First, he designs his logic circuit with standard logic modules and puts the circuit description into LOCUS. Then he specifies input value sequences, and starts the simulation. By investigating the output value sequences, he verifies whether it operates as he intended or not. If necessary, he modifies the circuit description and tries again. All these operations are processed promptly and interactively. Furthermore LOCUS can be concurrently used by several terminals, so that at the same time many students can exercise their circuits individually.

LOCUS is implemented on the minicomputer PDP-11/20, presently supporting three terminals.

2. LOGIC CIRCUIT SIMULATION WITH LOCUS

We have two types of conversation to LOCUS. One is a logic circuit description in LOCUS notation. The other is a command operation on this circuit.

In the following section, these features and usages are shown with an example circuit, ternary counter in Fig.1 and its simulation list in Fig.2.

2.1 Circuit Description (Fig.2②)

In order to describe logic circuit, we use the nodal representation method in LOCUS.

At first, a unique name is given to each node within two alphanumeric characters and a unique number is given to each element within two decimal

This paper first appeared in Japanese in Joho－Shori (Journal of the Information Processing Society of Japan), Vol. 14, No. 8 (1973), pp. 561～568.

* Faculty of Engineering Science, Osaka University Toyonaka, Osaka, Japan

digits in order to describe the logic circuit (Fig.1) . There are nine types of logic elements in LOCUS and their element symbols are defined in Table 1. (Notice that each element symbol may be abbreviated to the first two characters.) Using these element numbers, element symbols and node names, the circuit description is put into LOCUS in the format of Table 1.

2.2 Commands

By putting commands into LOCUS, we can execute various operations which are necessary for the logic circuit simulation. In general fomat of commands, an abbreviated command symbol is placed at the first field and parameters are placed in the next field if necessary. At the last field DELETE (DE) switch may be placed in some commands. Detailed formats are shown in Table 2 and their functions are shown below.

(1) ED Command (Fig.2 ③)

It makes LOCUS to accept circuit description. When it has a DE switch, it deletes the element with specified element number from the circuit description. If we have some errors or need some modifications in the circuit description, we can delete those elements by this command and put in correct or modified circuit descriptions instead of them.

(2) NI Command (Fig.2 ④)

This specifies a node as a external input node. When it has a DE switch, it cancells the specification.

(3) NO Command (Fig.2 ⑤, ⑫)

This specifies a node as a output observation node. The value of the node specified by this command is printed out at the execution time of the simulation. When it has a DE switch, it cancells the specification.

(4) EC Command (Fig.2 ⑥)

This indicates the termination of circuit description. LOCUS, then, determines the order in which it simulates the logic function of each element in the circuit, sets the output node value of FF to the initial value, and accepts the commands associated with the execution of the simulation.

(5) IN Command (Fig.2 ⑦)

This specifies the sequence of logical values (1 or 0) which is put into the external input node at the execution time.

(6) PR Command (Fig.2 ⑧)

This executes the simulation of the logic circuit during given clock times. At first LOCUS prints out the node names in this format:

external input node/ Q node of FF // output observation node.

Then LOCUS executes the simulation and prints out the values of each node in the above format in every clock time. Notice that the output node values of FF shown left and right side fields of // represent the value of one-clock time before and the value of one-clock time later, respectively.

(7) MO Command (Fig.2 ⑨)

This prints out the current value of the specified node. It is used to observe the node value at any time in the course of the simulation.

(8) IZ Command (Fig.2 ⑩)

This resets the state of FF (the output node value of FF) to the initial value which was specified in the circuit description and deletes the sequence of input values put into the external input node.

(9) SE Command (Fig.2 ⑪)

This sets the Q node of FF to the specified value. ($\bar{Q}$ node is set to the complement value of Q node.) By using this command the state of FF can be set to any state.

(10) AU Command (Fig.2 ⑬)

This automatically constructs the input-output-state table for the logic circuit. LOCUS simulates the circuit for all permutations of external input values and states of FF, and prints out the results in the same format as that of PR command. When the Q node of FF is specified by NO command as an output observation node, this table becomes an input-output-state transition table. Notice that this table becomes a truth table for a combinational circuit.

(11) SA Command (Fig.2 ⑭)

This prints out the circuit description, the specification of the external input node and the specification of the output observation node which have been put into LOCUS up to this time. This format is the same as the one used at the time of input to LOCUS, so putting out to paper tape enables the circuit to be preserved.

(12) RE Command (Fig.2 ⑮)

This deletes all the circuit descriptions, internal tables and so forth that have been put into LOCUS. LOCUS returns to the initial state and accepts a new circuit again.

3. CONFIGURATION OF LOCUS SYSTEM

The PDP-11/20 system is used here. The parts of its hardware directly related to LOCUS are composed of CPU, 24KW core memory (16 bits/word), three teletype terminals and 1.2MW disk unit.

LOCUS is composed of several modules which are written in assembly language PAL-11R. The size of the program area which consists of the area re-entrantly used by each terminal and common data area, is about 5K words in total. The size of data area allocated to each terminal is 2K words. In this case, we can handle the logic circuit with about 60 elements and 120 nodes. At present LOCUS has only three terminals and it total size is 11K words. Notice that the number of available terminals can be easily increased if the terminal units are added.

4. CONCLUSION

As a simulator for logic circuit education, LOCUS has many features and almost sufficient functions listed below. So it will be very useful tool in the education.

1) Interactive execution of circut simulation.
2) Extensive error checkings are done. Errors are pointed by plain diagnostic messages.
3) Multi-terminal usage is supported. As many students as many terminals can simulate their own circuits.
4) (i) Truth table or input-output -state table can be made by a single command.
   (ii) The state of a sequential circuit can be set to any state.
   (iii) Circuit descriptions can be put out to printer to verify them or to paper tape to preserve them.

ACKNOWLEDGEMENT

The authors wish to thank to the menbers of Kasami Lab., Faculty of Engineering Science, Osaka University for their assistances in developing

this system.

REFERENCES

1. Boyce, A. H., "Simulation of binary logic circuits by digital computers", The Marconi Review, Second quater, 1971.

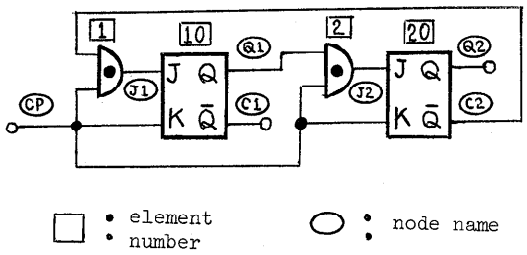

Fig. 1 Ternary counter

Table 1 Element symbol and circuit description format

(1) Logic elements (except FF)
  (i) AND, NAND, OR, NOR, XOR (exclusive OR), INV (inverter)
  (ii) element number/element symbol/input node 1, input node 2,···· /output node

(2) Flip Flop (FF)
  (i) FRS(R-S FF), JK(J-K FF), D(D FF)
  (ii) element number/element symbol/R(or J or D) input node, S(or K) input node/Q output node, $\bar{Q}$ output node /Q node initial value

(i) represents element symbol,
(ii) circuit description format.

Table 2 Command list

Command name	Format
EDIT	ED {/element number,···/DE}
NODE FOR INPUT	NI {external input node name,···/DE}
NODE FOR OUTPUT	NO {/node name,···/DE}
END OF CIRCUIT	EC
INPUT VALUE	IN /external input node name; value sequence/···
PROCESS	PR /clock number
MONITOR	MO /node name,···
INITIALIZE	IZ
SET FF	SE /Q node name; value /···
AUTOMATIC	AU
SAVE	SA
RESTART	RE

{} represents optional field

```
①LOCUS V001A

  *1/AND/CP,C2/J1
  *2/AND/CP,Q1/J2
②*10/FJK/J1,CP/Q1,C1/0
  *20/FJK/J2,C1/Q2,C2/0
③*ED/20/DE
  *20/FJK/J2,CP/Q2,C2/0
④*NI/CP
⑤*NO/J1,J2,Q1,Q2,C1,C2
⑥*EC
⑦#IN/CP;1111
⑧#PR/4
  CP/Q1,Q2//J1,J2,Q1,Q2,C1,C2
    1 /0 ,0 //1 ,0 ,1 ,0 ,0 ,1
    1 /1 ,0 //1 ,1 ,0 ,1 ,1 ,0
    1 /0 ,1 //0 ,0 ,0 ,0 ,1 ,1
    1 /0 ,0 //1 ,0 ,1 ,0 ,0 ,1
⑨#MO/J1,Q1
  J1,Q1
    1 ,1
⑩#IZ
⑪#SE/Q1;1
  #IN/CP;1
  #PR/1
  CP/Q1,Q2//J1,J2,Q1,Q2,C1,C2
    1 /1 ,0 //1 ,1 ,0 ,1 ,1 ,0
⑫#NO/C1,C2/DE
⑬#AU
  CP/Q1,Q2//J1,J2,Q1,Q2
    0 /0 ,0 // 0 ,0 ,0 ,0
    1 /0 ,0 //1 ,0 ,1 ,0
    0 /1 ,0 //0 ,0 ,1 ,0
    1 /1 ,0 //1 ,1 ,0 ,1
    0 /0 ,1 //0 ,0 ,0 ,1
    1 /0 ,1 //0 ,0 ,0 ,0
    0 /1 ,1 //0 ,0 ,1 ,1
    1 /1 ,1 //0 ,1 ,0 ,0
⑭#SA
  TYPE "P" OR SPACE
  1 /AN/CP,C2/J1
  2 /AN/CP,Q1/J2
  10/FJ/J1,CP/Q1,C1/0
  20/FJ/J2,CP/Q2,C2/0
  NI/CP
  NO/J1,J2,Q1,Q2
⑮#RE
  LOCUS V001A

  *
——— represents input to LOCUS
```

Fig. 2 Simulation list of ternary counter