GENERAL MICROPROGRAMMED MODELS AND THEIR INFLUENCE ON THE COMPUTATIONAL SPEED

P. Ciompi - L. Simoncini
Researchers at
Istituto di Elaborazione dell'Informazione
Consiglio Nazionale delle Ricerche
Pisa - Italy

AB-34

ABSTRACT

In this paper the general models of microprogrammed systems are compared. It is shown how the choice of the models affects the computational speed of the microprograms which can be implemented on the various models.

INTRODUCTION

In the literature (1) the problem of the equivalence of microprogrammed systems with centralized control part is studied. The systems are outlined by the interconnection of two sequential machines, a "Control Part" (C.P.) and an "Operating Part" (O.P.), and the C.P. and the O.P. are transformed in different model equivalent machines by the Cadden's procedure (2).

Aim of this paper is to show the influence that the choice of the models of the C.P. and of the O.P. has on the computational speed of the system, quite apart from the procedures which can be used to transform the machines, which compose the system, in different model machines.

We shall use some results from the theory of sequential machines and some remarks about microprogramming languages.

1. GENERAL REMARKS ON MICROPROGRAMMED SYSTEMS

It is well known that a digital system S with centralized C.P. can be outlined as shown in Fig. 1, by the interconnection of two sequential machines: a C.P. and an O.P.

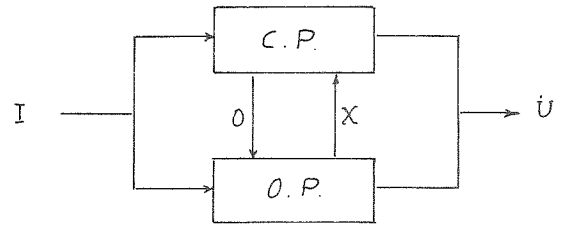

Fig. 1

I and U are respectively the inputs and the outputs of the system S; X are the "conditions" which control the sequencing of the C.P.; O are the "operations" which determine the execution of the elementary steps of the O.P.

The system S outlined in Fig. 1, as it is general, may represent a microprogrammed system too. In such systems the C.P. has particular structures which use read-only memories and/or read-write memories. In these memories the microprograms, which implement the machine instructions set, are written.

Let us consider the microprogramming languages studied in (3), and which are defined "phrase structured" (p.s.) and "microinstruction structured" (m.s.). These are two of the several languages studied for the description of a microprogrammed system, but they are general, as with them it is possible to describe any system.

Moreover it has been shown (4) that many microprogramming languages are particular cases of them.

The general structures of the p.s. and m.s. microinstructions are the following:

p.s. $\gamma_i \mid (X_1)O_1, \gamma_k ; (X_2)O_2, \gamma_j ; \ldots . (X_n)O_n, \gamma_m$

m.s. $\gamma_i \mid O_i ; (X_1)\gamma_k, (X_2)\gamma_j, \ldots \ldots (X_n)\gamma_m$

γ is the symbol corresponding to the generic microinstruction; $O_1, O_2, \ldots . O_n$ are the operations which are executed in the O.P. of the system; $X_1, X_2, \ldots . X_n$ are the mutually exclusive conditions which, in the p.s. language, determine the execution of a particular operation and the sequencing of the C.P.; in the m.s. language, they determine only the sequencing of the C.P., choosing a particular state among the several states of the C.P.

It is trivial that the m.s. language is a particular case of the p.s. language, that is, when $O_1 = O_2 = \ldots O_n$ in the microinstruction .

It is shown in (3) that the microprograms written with these two languages can be converted in flow tables, as in Fig. 2; these flow tables describe in an alternative way the behaviour of the C.P.

Generally the C.P. of a system is structured as a Mealy machine, if the microprograms are written in a p.s. language, and is structured as a Moore machine, if the microprograms are

written in a m.s. language.

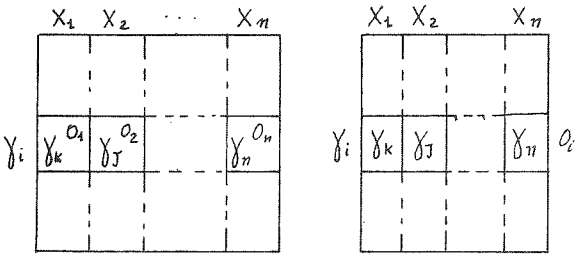

Fig. 2

Two general models of the C.P. obtained directly from the structure of the p.s. and m.s. languages (or from the two types of the flow tables of Fig. 2) are shown in Fig. 3.

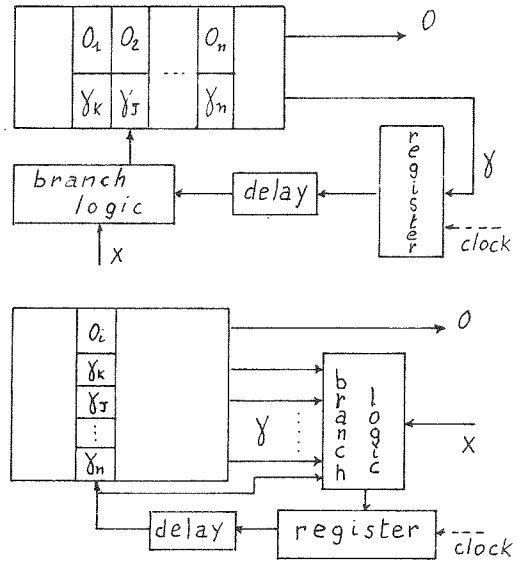

Fig. 3

In the following, we will refer to these structures as those which allow the implementation of the microprograms written in p.s. and m.s. language respectively. We will call these structures ME.CP and MO.CP respectively.

As the C.P. of a microprogrammed system can be realized with one of these two structures and the O.P. can be a Mealy or a Moore machine, any microprogrammed system can belong to one of the following three classes:

a) ME.CP - MO.OP

b) MO.CP - ME.OP

c) MO.CP - MO.OP

We will compare these three classes of systems as regards the computational speed, that is the number of elementary steps which are necessary to execute the microprograms which are implemented on them.

As the microprograms describe the machine instructions set of the system, and as during the execution of an instruction the system can be outlined as an autonomous machine (we will suppose that the external inputs affect the behaviour of the system only in particular moments, but non during the execution of an instruction), we can outline the system as an autonomous machine, as in Fig. 4, if we consider that the initial states are obtained by the combination of the state of the C.P. corresponding to the initial microinstruction γ_0 of the microprogram, and all the possible states of the O.P., corresponding to all the possible contents of the registers and memory elements for which the particular instruction is defined.

The results of the execution of the instruction is the state in which is the O.P. when the C.P. goes to the state corresponding to the final microinstruction γ_F .

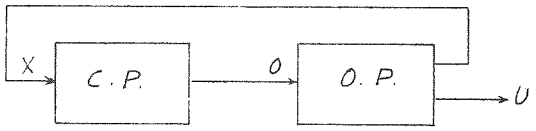

Fig. 4

2. COMPARISON BETWEEN A ME.CP - MO.CP SYSTEM AND A MO.CP - MO.OP SYSTEM

Now we shall make the comparison between two systems of the following classes:

ME.CP - MO.OP and MO.CP - MO.OP.

In order that this comparison may be meaningful, it is necessary that the microprograms, which are implemented on the two models, have the same input and output (conditions and operations) dictionary.

Therefore we will suppose that the O.P.s are the same.

Any microprogram which is implementable on a MO.CP - MO.OP model, and therefore written using a m.s. language, can be implemented on a ME.CP - MO.OP model with the same O.P., as the m.s. language is a particular case of the p.s. language. On the contrary, if a microprogram is implementable on a ME.CP - MO.OP model, and therefore written using a p.s. language, it may not be implemented on a MO.CP - MO.OP model if it exists at least a p.s. microinstruction where $O_1 \neq O_2 \neq \ldots \neq O_n$. Therefore it is necessary to write a new microprogram using the m.s. language.

It is easy to show that, given a microprogram implemented on a ME.CP - MO.OP model, and therefore written using the p.s. language, it is not always possible to write another microprogram in a m.s. language which describes the same instruction on a MO.CP - MO.OP model with

a greater or equal computational speed.

Let us consider the microprogram in Fig. 5.

$\gamma_0 \mid (X_1)O_1, \gamma_1 \; ; \; (X_2)O_3, \gamma_3$
$\gamma_1 \mid (X_1)O_2, \gamma_F \; ; \; (X_2)O_1, \gamma_2$
$\gamma_2 \mid (X_1)O_1, \gamma_F \; ; \; (X_2)O_3, \gamma_F$
$\gamma_3 \mid (X_1)O_3, \gamma_F \; ; \; (X_2)O_1, \gamma_2$
$\gamma_F \mid --$

Fig. 5

It describes an instruction f. The flow table of the O.P. which executes f is shown in Fig. 6.

	O_0	O_1	O_2	O_3	
1	1	2	4	1	X_1 U_1
2	2	3	2	6	X_2 U_2
3	3	6	4	5	X_2 U_3
4	4	6	4	2	X_1 U_4
5	5	1	6	2	X_1 U_5
6	6	5	6	4	X_2 U_6

Fig. 6

The instruction f is defined by the following table of the transitions from all the initial states of the O.P. to the corresponding final states:

$1 \rightarrow 5 \; ; \; 2 \rightarrow 1 \; ; \; 3 \rightarrow 2$

$4 \rightarrow 1 \; ; \; 5 \rightarrow 4 \; ; \; 6 \rightarrow 2$

It is easy to see that it is not possible to write any microprogram which describes the function f with a m.s. language, in such a way to obtain a MO.CP - MO.OP system with equal or greater computational speed than that of a ME.CP - MO.OP system which implements the microprogram of Fig. 5.

It is possible to write the following statement:

> A ME.CP - MO.OP system has generally a greater computational speed than that of a MO.CP - MO.OP system, when the O.P. are the same, for the microprograms which can be implemented on the first system have greater computational speed than those which can be implemented on the second system.

In the literature (1) it exists a condition which allows the transformation by the Cadden's procedure of a microprogram written with the p.s. language, in a microprogram written in the m.s. language. This condition is such that the transformation mantains an equal computational speed for the two microprograms. Moreover, in that paper a procedure is shown, with which any microprogram, for which that condition is not valid, can be written in a form which verifies that condition, but that procedure lessens the computational speed of the microprogram, for it uses the "no-operation" (O_0) which does not change the state of the O.P.

With that procedure applied to the microprogram of Fig. 5, it is possible to write the microprogram of Fig. 7 which is slower than the other and it is still written in the p.s. language.

$\gamma_0 \mid O_0, \gamma_0'$
$\gamma_0' \mid (X_1)O_1, \gamma_1 \; ; \; (X_2)O_3, \gamma_3$
$\gamma_1 \mid O_0, \gamma_1'$
$\gamma_1' \mid (X_1)O_2, \gamma_F \; ; \; (X_2)O_1, \gamma_2$
$\gamma_3 \mid O_0, \gamma_3'$
$\gamma_2 \mid O_0, \gamma_2'$
$\gamma_2' \mid (X_1)O_1, \gamma_F \; ; \; (X_2)O_3, \gamma_F$
$\gamma_3' \mid (X_1)O_3, \gamma_F \; ; \; (X_2)O_1, \gamma_2$
$\gamma_F \mid - -$

Fig. 7

Now if we use the Cadden's procedure to transform the flow table of the Mealy machine, which describes the behaviour of the C.P. which implements the microprogram of Fig. 7, we obtain the microprogram, written in the m.s. language, of Fig. 8 which has the same computational speed of the microprogram of Fig. 7. Therefore it is slower than the one of Fig. 5.

$\gamma_0 \mid O_0 \; ; \; (X_1)\gamma_1 \; ; \; (X_2)\gamma_2$
$\gamma_1 \mid O_1, \gamma_3$
$\gamma_2 \mid O_3, \gamma_5$
$\gamma_3 \mid O_0 \; ; \; (X_1)\gamma_{F1} \; ; \; (X_2)\gamma_4$
$\gamma_4 \mid O_1, \gamma_6$
$\gamma_5 \mid O_0 \; ; \; (X_1)\gamma_{F1} \; ; \; (X_2)\gamma_4$
$\gamma_6 \mid O_0 \; ; \; (X_1)\gamma_{F1} \; ; \; (X_2)\gamma_{F3}$
$\gamma_{F1} \mid O_1, \ldots$
$\gamma_{F2} \mid O_2, \ldots$
$\gamma_{F3} \mid O_3, \ldots$

Fig. 8

The microprogram of Fig. 8, obtained using the previous procedure is not the faster obtainable by writing it with the m.s. language. Moreover the use of the operation O_0 (no-operation) to lessen the computational speed of the microprogram of Fig. 5 determines a particular characteristic on the structure of the microinstructions of the microprogram. They are of the following two types:

1) O_J, γ_K

2) $O_0 \; ; \; (X_1)\gamma_J \; ; \; \ldots \; ; \; (X_K)\gamma_K$

In other words, the microinstructions, composing the microprogram obtained from the one of Fig. 5, describe separately the action on the O.P. (type 1)) and the tests on the results (type 2)). This type of microprogramming is found in some computers.

This procedure allows the transformation from a microprogram, written using the p.s. language, to a microprogram, written using the m.s. language, with the same input and output

dictionaries, by the introduction of the operation O_o , and therefore they are implementable on the same O.P.

Now we will show under what conditions it is possible to obtain from a ME.CP - MO.OP system, a MO.CP - MO.OP system with equal computational speed, modifying the input dictionary (conditions).

We have shown (5) that, given the machine $\bar{M}'$ of Fig. 9 it is always possible to obtain the machine $\bar{M}$ such that the following relation holds:

$$\bar{M} \supset \bar{M}' \Big|_{\substack{1,1' \\ 2,2'}}$$

This relation means that, given any state of $\bar{M}'$ it exists at least one state of $\bar{M}$ such that they are equivalent. The numbers on the right of the bar indicate the outputs for which is valid the relation on the left.

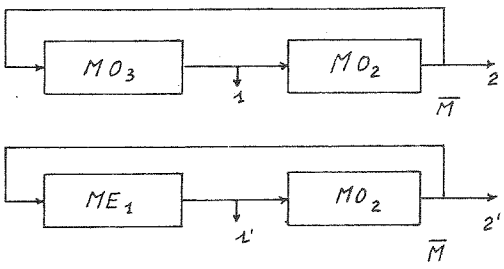

Fig. 9

In (5) it is shown that the correspondence among the equivalent states is the following:

$$S_i T_j \equiv S_{ij} T_j$$

where S_i , T_j and S_{ij} indicate respectively the states of the ME_1 , MO_2 and MO_3 machines.

Moreover in (5) it is shown that MO_3 is a sequential machine whose flow table has, in each row, only one specified entry, that one corresponding to the condition X_j associated to the state T_j of MO_2 .

Let ME_1 and MO_2 of $\bar{M}'$ be respectively the C.P. and the O.P. of a ME.CP - MO.OP system which implements a microprogram. Let S_o be the state of ME_1 corresponding to the initial microinstruction of this microprogram.

If MO_3 of $\bar{M}$ must be the C.P. implementing a microprogram which executes the same function with equal computational speed on a MO.CP-MO.OP system, it must exist a state S_{oo} , corresponding to tha initial microinstruction of this new microprogram, such that the relation $S_o T_j \equiv S_{oo} T_j$ holds for j = 1...n, where n is the number of the states of MO_2. This condition can be expressed in another way: it is necessary that the initial states S_{o1} , S_{o2} , .., S_{on} of MO_3 , obtained by the procedure given in (5) are compatible among them and compatible with S_{oo} .

From the definition of compatible states (6) it is necessary that the states S_{oj} of MO_3 have the same output and this implies that the microinstruction γ_o of the microprogram implemented on the ME.CP - MO.OP model is of the m.s. type.

It is easy to show that this condition is sufficient when the O.P. has different conditions associated to all the states.

Therefore, given a microprogram implementable on a ME.CP - MO.OP model, it is always possible to write a microprogram which executes the same function with equal computational speed on an O.P. obtained from the first one, associating to different states different conditions.

It is now possible to write the following statement:

> Given a microprogram implemented on a ME.CP - MO.OP model, it exists a microprogram implementable on a MO.CP - MC.OP model, which describes the same instruction with equal computational speed, and with the same output dictionary (operations), if the initial microinstruction γ_o is of m.s. type.

This is a sufficient condition. As an example let us consider the microprogram in Fig.10.

$\gamma_o \mid O_o \ , \gamma_o'$
$\gamma_o' \mid (X_1)O_1, \gamma_1 \ ; \ (X_2)O_3, \gamma_3$
$\gamma_1 \mid (X_1)O_2, \gamma_F \ ; \ (X_2)O_1, \gamma_2$
$\gamma_2 \mid (X_1)O_1, \gamma_F \ ; \ (X_2)O_3, \gamma_F$
$\gamma_3 \mid (X_1)O_3, \gamma_F \ ; \ (X_2)O_1, \gamma_2$
$\gamma_F \mid \ldots$

Fig. 10

This microprogram is obtained from the one of Fig. 5, adding the microinstruction:

$$\gamma_o \mid O_o \ , \gamma_o'$$

which does not change the function; this microprogram verify the given condition.

The microprogram of Fig. 10 can also be written as in Fig. 11 if we suppose to associate to the different states of the O.P. different conditions:

1 : X_1 ; 2 : X_2 ; 3 : X_3 ; 4 : X_4 ; 5 : X_5 ;
6 : X_6 .

$\gamma_o \mid O_o \ , \gamma_o'$
$\gamma_o' \mid (X_1,X_4,X_5)O_1, \gamma_1 \ ; \ (X_2,X_3,X_6)O_3 \ , \gamma_3$
$\gamma_1 \mid (X_1,X_4,X_5)O_2, \gamma_F \ ; \ (X_2,X_3,X_6)O_1 \ , \gamma_2$
$\gamma_2 \mid (X_1,X_4,X_5)O_1, \gamma_F \ ; \ (X_2,X_3,X_6)O_3 \ , \gamma_F$
$\gamma_3 \mid (X_1,X_4,X_5)O_3, \gamma_F \ ; \ (X_2,X_3,X_6)O_1 \ , \gamma_2$
$\gamma_F \mid$

Fig. 11

Transforming this program by the procedure given in (5) it is possible to write the microprogram of Fig. 12 which has the same computational speed as the one of Fig. 10, if we associate to the states of the O.P. the following conditions:
1 , 4 : X_1 ; 2 : X_2 ; 3 , 6 : X_3 ; 5 : X_4 .

$\gamma_0 | O_0 \; ; \; (X_1, X_4)\gamma_1 \; ; \; (X_2, X_3)\gamma_2$
$\gamma_1 | O_1 \; ; \; (X_1)\gamma_1 \; ; \; (X_2)\gamma_3 \; ; \; (X_3)\gamma_5 ; \; (X_4)\gamma_4$
$\gamma_2 | O_3 \; ; \; (X_1, X_4)\gamma_F \; ; \; (X_2)\gamma_1 \; ; \; (X_3)\gamma_2$
$\gamma_3 | O_3 \; ,$
$\gamma_4 | O_2 \; , \gamma_F$
$\gamma_5 | O_1 \; , \gamma_F$
$\gamma_F |$

Fig. 12

This microprogram is clearly written in a m.s. language.

3. COMPARISON BETWEEN A ME.CP-MO.OP SYSTEM AND A MO.CP-ME.OP SYSTEM

In this section the computational speeds of a ME.CP - MO.OP system and of a MO.CP - ME.OP system are compared.

As we suppose that the microprograms implemented by the two systems are written with a language which use the same input and output dictionaries, we will suppose that the two O.P. are represented by flow tables obtained by the Cadden's procedure.

We will indicate this relation with:

$$ME.OP \simeq MO.OP$$

We have shown (5) that if the machines shown in Fig. 13 are such that: $ME_1 \simeq MO_1$ and $ME_2 \simeq MO_2$, then the relation:

$$\bar{M}' \supset \bar{M}\,|_{1,1'}$$

holds and the correspondence among the equivalent states of $\bar{M}$ and $\bar{M}'$ is:

$$S_i' \; T_J \equiv S_{i+1} \; T_J'$$

where T_J is a state of MO_2 and T_J' is the similar state (7) of T_J in ME_2 , S_i' is a state of ME_1 and S_{i+1} is the similar state in MO_2 of the state S'_{i+1} which is the successor of S_i' under the input associated to T_J . The outputs in 2 and 2' are shifted of a single step.

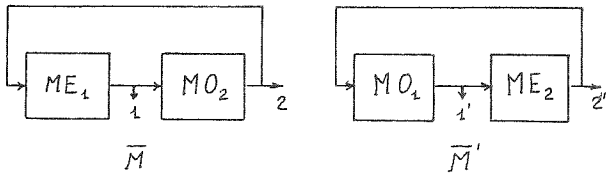

Fig. 13

Let us now suppose that on the MO.CP - ME.OP system is implemented a microprogram and let be the initial microinstruction:

$$\gamma_0 | O_i \; ; \; (X_1)\gamma_1 \; , \; \ldots \; , \; (X_n)\gamma_n$$

It defines the behaviour of the system starting from the states $S_0 T_J'$ (j=1...n), where S_0 is the state of the C.P. corresponding to γ_0 .

Let us now modify this microprogram by adding as initial microinstruction the following one:

$$\gamma_{00} | O_0 ,$$

Let also $\bar{M}'$ be the machine which represents the system implementing this microprogram.

Let us transform $\bar{M}'$ in $\bar{M}$; the relation among the equivalent states is: $S'_{00} \; T \equiv S_0 T'_J$.

Therefore from the flow table of ME_1 of $\bar{M}$ is then possible to obtain a microprogram which has, as initial microinstruction, the one associated to S'_{00} and which is implementable on a ME.CP - MO.OP model which has the same computational speed as the original microprogram which is implemented on the MO.CP - ME.OP model. Moreover the initial microinstruction of this microprogram is:

$$\gamma_0 | O_i \; , \gamma_{0+1}$$

that is it is not conditioned as regards the operation and the successive microinstruction.

Therefore:

> Given any microprogram which describes a machine instruction with a m.s. language on a MO.CP - ME.OP model, it is always possible to write a microprogram which, written with a p.s. language, with the same input and output dictionary, describes the same instruction with equal computational speed on a ME.CP - MO.OP model and its initial microinstruction is: $\gamma_0 | O_i , \gamma_{0+1}$.

Relating to Fig. 13 and to the relation among the equivalent states of $\bar{M}$ and $\bar{M}'$ it is possible to write:

> If a microprogram which implements on a ME.CP - MO.OP model a machine instruction using a p.s. language is given, and its initial microinstruction is not conditioned, it is then possible to write a microprogram which, with the same input and output dictionary, implements the same instruction with the same computational speed on a MO.CP - ME.OP model using a m.s. language.

From the two previous statements it follows that, given a microprogram which is implementable on a ME.CP - MO.OP model and whose initial microinstruction is not of the type: $\gamma_0 | O_i , \gamma_{0+1}$, if a microprogram exists which is implementable on a MO.CP - ME.OP model with equal computational speed, then it must exist another microprogram implementable on a ME.CP-MO.OP model and which has equal computational speed and its initial microinstruction is not conditioned.

It is easy to verify, if we impose that a microprogram implementable on a ME.CP - MO.OP model has an initial microinstruction of the type $\gamma_0 | O_i , \gamma_{0+1}$, that generally we obtain microprograms with less computational speed than those obtainable without the condition on the initial microinstruction.

As an example you can consider the microprogram of Fig. 5 and you can try to write another microprogram which implements the same function

f, in such a way that the initial microinstruction is not conditioned.

It is possible to write:

> A ME.CP - MO.OP system, has generally a greater computational speed than the one of a MO.CP - ME.OP system, where ME.OP ≠ MO.OP, because the microprograms which are implemented on the first system have generally greater computational speed than the one of the microprograms which can be implemented on the second system.

However the computational speeds of these models are comparable because it is possible, given any microprogram implementing a function f on the ME.CP - MO.OP system to write a microprogram implementing the same function on a MO.CP - ME.OP system such that its computational speed is at most one step slower than the computational speed of the other.

In fact any microprogram written using a p.s. language can be modified in such a way that its initial microinstruction is not conditioned by lessening its computational speed, using an initial microinstruction of the type:

$\gamma_0 | O_0 , \gamma_{0+1}$.

As an example let us consider the microprogram of Fig. 14 obtained from the one of Fig.5, adding $\gamma_0 | O_0, \gamma_1$ as initial microinstruction.

$\gamma_0 | O_0, \gamma_1$
$\gamma_1 | (X_1) O_1, \gamma_2 \; ; \; (X_2) O_3, \gamma_4$
$\gamma_2 | (X_1) O_2, \gamma_F \; ; \; (X_2) O_1, \gamma_3$
$\gamma_3 | (X_1) O_1, \gamma_F \; ; \; (X_2) O_3, \gamma_F$
$\gamma_4 | (X_1) O_3, \gamma_F \; ; \; (X_2) O_1, \gamma_3$
$\gamma_F | \ldots$

Fig. 14

It is possible to obtain the microprogram of Fig. 15 which implements the same function on an O.P., whose flow table is shown in Fig.16.

$\gamma_0 | O_0 \; ; \; (X_1)\gamma_1 \; , \; (X_2)\gamma_3$
$\gamma_1 | O_1 \; ; \; (X_1)\gamma_{F2} \; , \; (X_2)\gamma_2$
$\gamma_2 | O_1 \; ; \; (X_1)\gamma_{F1} \; , \; (X_2)\gamma_{F3}$
$\gamma_3 | O_3 \; ; \; (X_1)\gamma_{F3} \; , \; (X_2)\gamma_2$
$\gamma_{F1} | O_1 \; ; \; \ldots$
$\gamma_{F2} | O_2 \; ; \; \ldots$
$\gamma_{F3} | O_3 \; ; \; \ldots$

Fig. 15

	O_0	O_1	O_2	O_3
1	1 X_1	2 X_2	4 X_1	1 X_1
2	2 X_2	3 X_2	2 X_2	6 X_2
3	3 X_2	6 X_2	4 X_1	5 X_2
4	4 X_1	6 X_2	4 X_1	2 X_1
5	5 X_1	1 X_1	6 X_2	2 X_1
6	6 X_2	5 X_1	6 X_2	4 X_2

Fig. 16

4. CONCLUSIONS

In this paper we have compared the general models of microprogrammed systems with centralized C.P., as regards the computational speed.

It has been shown that a ME.CP - MO.OP model has a greater computational speed than the two other general models; moreover the MO.CP-ME.OP and the ME.CP - MO.OP models have comparable computational speed.

This is valid when we do not change the input and output dictionary (conditions and operations) of the C.P.s of the models.

Modifying the input dictionary (conditions), we have shown that the computational speeds of the ME.CP - MO.OP and MO.CP - MO.OP models are comparable.

All the arguments are valid leaving out of consideration the intrinsic speed of the structures of the systems, that is, the minimal time between a pulse clock and another (this time varies according to the different structures of the models).

Therefore a comprehensive comparison among the speeds of the various models can be made only considering the computational speeds and the intrinsic speeds of the models.

BIBLIOGRAPHY

(1) G.F.Casaglia, G.B.Gerace, M.Vanneschi: "Equivalent Models and Comparison of Microprogrammed Systems" Int. Advanced Summer School on Microp.-St.Raphael, France 30 Aug.-10 Sept. 1971.

(2) W.I.Cadden: "Equivalent Sequential Circuits" IRE Trans. on Circuit Theory - Vol. CT-6, pp. 30 - 34, March 1969.

(3) G.F.Casaglia,M.Vanneschi: "Modelli e Strutture per il Progetto di Sistemi Microprogrammati" in Italian, Int.Rep. n.2, Conv. CNR-ENI, Aug. 71.

(4) G.Conti: "Sui Linguaggi e le Strutture di Microprogrammazione" in Italian, Int.Rep. n. 5, Conv. CNR-ENI, Dec. 71.

(5) P.Ciompi, L.Simoncini: "On the Equivalence and Similitude of Systems of Interconnected Sequential Machines" 8^ Yugoslav Int. Symp. on Information Proc., Bled, 1 - 5 Oct.1973.

(6) E.J.McCluskey: "Introduction to the Switching Theory" McGraw-Hill, 1968.

(7) J.Hartmanis, R.E.Stearns: "Algebraic Structure Theory of Sequential Machines" Prentice Hall, 1966.