Full decomposition of sequential machines with the output behaviour realization

Research Report
ISSN 0167-9708
Coden: TEUEDE

Eindhoven
University of Technology
Netherlands

Faculty of Electrical Engineering

The Full Decomposition of Sequential Machines with the Output Behaviour Realization

by
L. Jóźwiak

EUT-Report 88-E-199
ISBN 90-6144-199-4

March 1988

Eindhoven University of Technology Research Reports

EINDHOVEN UNIVERSITY OF TECHNOLOGY

Faculty of Electrical Engineering
Eindhoven The Netherlands

ISSN 0167- 9708 Coden: TEUEDE

THE FULL DECOMPOSITION OF SEQUENTIAL MACHINES WITH THE OUTPUT BEHAVIOUR REALIZATION

by

L. Jóźwiak

EUT Report 88-E-199
ISBN 90-6144-199-4

Eindhoven
March 1988

CIP-GEGEVENS KONINKLIJKE BIBLIOTHEEK, DEN HAAG

Jóźwiak, L.

The full decomposition of sequential machines with the output behaviour realization / by L. Jóźwiak. - Eindhoven: University of Technology, Faculty of Electrical Engineering. - Fig. - (EUT report, ISSN 0167-9708; 88-E-199)
Met lit. opg., reg.
ISBN 90-6144-199-4
SISO 664 UDC 681.325.65:519.6 NUGI 832
Trefw.: automatentheorie.

C O N T E N T S

THE FULL DECOMPOSITION OF SEQUENTIAL MACHINES WITH THE OUTPUT BEHAVIOUR REALIZATION

Lech Jóźwiak

Group Digital Systems, Faculty of Electrical Engineering,
Eindhoven University of Technology (The Netherlands)

Abstract-The control units of large digital systems can use up to 80% of the entire hardware implementing the system . Therefore, it is very important to reduce the amount of hardware taken by the control unit and to simplify the design, implementation and verification process. In most cases, the control unit can be constructed as a sequential machine. So, the design of control units for digital systems leads to the fololowing practical problem:

How to decompose a complex sequential machine into a number of simpler submachines in order to : simplify the design, implementation and verification process; make it possible to optimize separate sumachines, whereas it may be impossible to optimize directly the whole machine; make possible to implement the machine with existing building blocks.

The decomposition theory of sequential machines aims to find answers to this question. For many years, decomposition of internal states of sequential machines was considered. However, together with the progress in LSI technology and the introduction of array logic into the design of sequential circuits, a real need arose for decomposition of not only the states of sequential machines but of inputs and outputs too, i.e. for full-decomposition.

In this work, a general and unified classification of full-decompositions and formal definitions of different sorts of full-decompositions for Mealy and Moore machines are presented and some theorems about the existence of full-decompositions with the output behaviour realization are formulated and proved. This theorems constitute a theoretical basis for the practical decomposition algorithms and for the software system calculating different sorts of decomposition for sequential machines. Similar theorems for the case of full-decompositions with the state and output behaviour realization are available in [16].

Index Terms-**Automata theory, decomposition, logic system design, sequential machines.**

Acknowledgements-**The author is indebted to Prof. ir. A. Heetman and Prof. ir. M. P. J. Stevens for making it possible to perform this work and to Dr. P. R. Attwood for making corrections to the English text.**

1. Introduction.

Most of the architectures of todays composed digital systems implement Glushkov's model of the information processing system. In these architectures, it is possible to distinguish two basic parts:

- an ***operative unit,*** implementing tools for performing operations with the data,
- a ***control unit***, implementing control algorithms of a given information processing system.

A control unit, based on the status of the operative part and certain external signals, generates and sends the control signals to the operative unit in order to perform the given sequences of operations with the data in the operative part (Fig. 1.1).

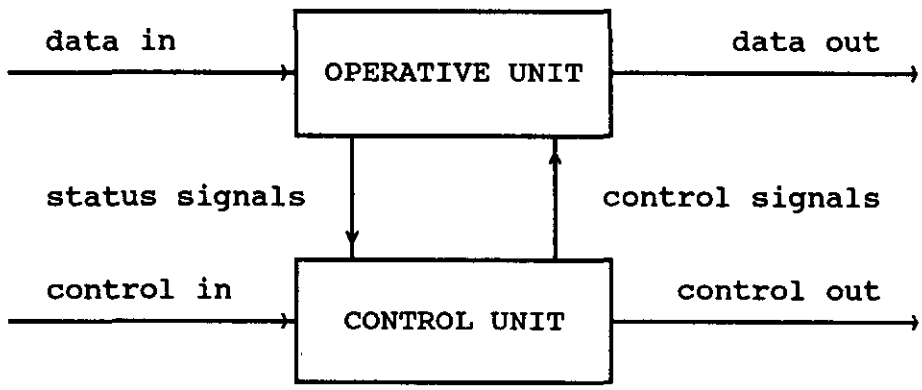

Fig. 1.1 The basic architecture of a composed digital system.

The control units of large digital systems can angage up to 80% of the entire hardware implementing the system and, therefore, it is very important to reduce the amount of hardware used by the control unit and to simplify the design, implementation and verification process.

In most cases, the control unit can be constructed as a sequential machine (a finite automaton).

Reducing the amount of hardware neded for implementing a sequential machine is a very complicated process which can be carry into effect in a number of steps implementing some optimization algorithms.This steps include:

- the optimal state reduction,
- the optimal state assignment,
- the optimal choice of flip-fops,
- minimization of the Boolean functions representing the next-state and output functions of a sequential machine.

However, the efficiency of these optimization algorithms (understood to be a function of such parameters as: the quality of the result, the computation time, the memory space used) decreases rapidly with the dimensions of a sequential machine.

So, the design of control units for large digital systems can lead to the fololowing practical problem:

How to decompose a complex sequential machine into a number of simpler submachines in order to obtain:

- *the* better *organization of the system and of the design, implementation and verification process,*
- *the possibility of optimiz*ing *of the separate sub*machines, *whereas it* may *be impossible to optimize the whole machine directly,*
- *the possibility of implementing the machine with existing building blocks.*

The decomposition theory of sequential machines aims to find answers to this question.

Research in the above mentioned field was started in the early Sixties [8][9][10][20][21]. For many years, decomposition on internal states of sequential machines has been considered [4][12][17][18][19][20][21]. However, together with the progress in LSI technology and the introduction of array logic (PAL, PGA, PLA, PLS) into the design of sequential circuits, a real need has arisen for decompositions not only of states of sequential machines, but of inputs and outputs too, i.e. for full-decompositions.

An approach to the full-decomposition of sequential machines has been presented in [14] and [15]. Among other things, the definitions and theorems concerning one parallel and two serial types of full-decompositions for Mealy machines were introduced.

In [16], a general and unified classification of full-decompositions is presented, formal definitions of different sorts of full-decompositions for Mealy and Moore machines were introduced and theorems about the existence of full-decompositions with the *state and output behaviour* realization were formulated and proved.

In this work, theorems about the existence of full-decompositions with the *output behaviour* realization will be formulated and proved. These theorems constitute the theoretical basis of the practical decomposition algorithms and the software system for calculating different sorts of decompositions of sequential machines.

2. Full-decompositions and their sorts.

DEFINITION 2.1 A ***sequential machine*** M is an algebraic system defined as follows:

$$M = (I,\ S,\ O,\ \delta,\ \lambda)\ ,$$

where:

I - a finite nonempty set of inputs,

S - a finite nonempty set of internal states,

O - a finite set of outputs,

δ - the next state function, δ: SxI $\longrightarrow$ S,

λ - the output function, λ: SxI $\longrightarrow$ O (a *Mealy machine*),
or λ: S $\longrightarrow$ O (a *Moore machine*).

If the output set O and the output function λ are not defined, the sequential machine M = (I, S, δ) is called a ***state machine.***

The machine functions δ and λ can be considered to be sets of functions created for each input:

$\delta = \{\delta_x |\ \delta_x: S \longrightarrow S \text{ and } x \in I\}$

and

$\lambda = \{\lambda_x |\ \lambda_x: S \longrightarrow O \text{ and } x \in I\}$,

where $\delta_x: S \longrightarrow S$ and $\lambda_x: S \longrightarrow O$ are defined by:

$\forall x \in I\ \forall s \in S\quad \delta_x(s) = \delta(s,x)$,

$\lambda_x(s) = \lambda(s,x)$.

δ_x and λ_x , respectively, are called the next-state function and the output function with respect to the input x.

In the next sections for $\delta_x(s)$ and $\lambda_x(s)$ the notations $s\delta_x$ and $s\lambda_x$ will be used.

For $x \in I$ and $Q \subseteq S$ two partial functions:

$\overline{\delta}_x: 2^S \longrightarrow 2^S$ and $\overline{\lambda}_x: 2^S \longrightarrow 2^O$ are defined,

where:

$\forall x \in I\ \forall Q \subseteq S\quad Q\overline{\delta}_x = \{s\delta_x |\ s \in Q\},\ Q\overline{\lambda}_x = \{s\lambda_x |\ s \in Q\}$.

For $X \subseteq I$ and $Q \subseteq S$ the following two partial functions are also defined:

$\overline{\delta}_X$: $2^S \longrightarrow 2^S$ and $\overline{\lambda}_X$: $2^S \longrightarrow 2^O$,

where:

$Q\overline{\delta}_X = \{s\overline{\delta}_x | \ s \in Q \wedge x \in X\}$,

$Q\overline{\lambda}_X = \{s\overline{\lambda}_x | \ s \in Q \wedge x \in X\}$.

In this work, only simple decompositions (i.e. decompositions with two component machines) will be taken into account and, therefore, the term "decomposition" is assumed to mean "simple decomposition".

Let $M = \{I, S, O, \delta, \lambda\}$ be the machine to be decomposed and $M_1 = \{I_1, S_1, O_1, \delta_1, \lambda_1\}$ and $M_2 = \{I_2, S_2, O_2, \delta_2, \lambda_2\}$ be two partial machines.

In a full-decomposition, it is necessary to find the partial machines M_1 and M_2 each having fewer states and/or outputs than machine M and/or each calculating its next states and outputs using only the part of information about the input of machine M and, in combination, forming a machine M'which imitate M from the input-output point of view.

In a state-decomposition, it was necessary to find the machines M_1 and M_2 having only fewer internal states. Inputs and outputs needed not be decomposed.

Before considering the different sorts of full-decomposition, the definition of realization from [12] will be presented.

<u>**DEFINITION**</u> <u>2.2</u> Machine $M' = (I', S', O', \delta', \lambda')$ *realizes (is realization of)* machine $M = (I, S, O, \delta, \lambda)$ *if and only if* the following relations exist:

ψ: $I \longrightarrow I'$ (a function),

ϕ: $S \longrightarrow 2^{S'}$ (a function into nonvoid subsets of S'),

θ: $O' \longrightarrow O$ (a surjective partial function) ,

and this relations satisfy the following conditions:

$\phi(s)\delta'_{\psi(x)} \subseteq \phi(s\delta_x)$

and

$s\lambda_x = \theta(s'\lambda'_{\psi(x)})$ (for a Mealy machine)

or

$s\lambda = \theta(s'\lambda')$ (for a Moore machine)

for all $s \in S$, $s' \in \phi(s)$ and $x \in I$.

Let I^* be a set of all the input sequences $x_1 x_2 \ldots x_n$ $(n=0,1,\ldots)$, let $\vec{x} = \vec{x}'x$ for $\vec{x}' \in I^*$ and $x \in I$ and let $\vec{\lambda}$ and $\vec{\delta}$ be the two functions calculating the final output and the final state reached by a machine from the state s under the input sequence $\vec{x}$:

$$\vec{\delta}: SxI^* \longrightarrow S,\ \vec{\delta}(s,\vec{x}) = \delta(\vec{\delta}(s,\vec{x}'),x),$$

$$\vec{\lambda}: SxI^* \longrightarrow O,\ \vec{\lambda}(s,\vec{x}) = \lambda(\vec{\delta}(s,\vec{x}'),x) \quad \text{(Mealy case)},$$

$$\vec{\lambda}(s,\vec{x}) = \lambda(\vec{\delta}(s,x)) \quad \text{(Moore case)}.$$

It can be proved that if M' is a realization of M in the sense of definition 2.1 then $\forall s \epsilon S\ \forall s' \epsilon \phi(s)$ and $\forall \vec{x} \epsilon I^*$: $\vec{\lambda}(s,\vec{x}) = \theta(\vec{\lambda}'(s',\psi(x))$, i.e. for all possible input sequences outputs reached by machine M and its imitation M' are, after a renaming, identical. Due to this fact, a realization in the sense of definition 2.1 will be called by us: ***realization of the output behaviour.***

In some cases, not only the output changes of the machine are concerned but also the state changes. The full-decompositions with the realization of the state and output behaviour of sequential machines has been considered in [16] and their definition is only presented below:

<u>DEFINITION</u> <u>2.3</u> Machine M' = (I', S', O', δ', λ') , ***realizes the state and output behaviour*** of machine M = (I, S, O, δ, λ) *if and only if* the following relations exist:

ψ: I $\longrightarrow$ I' (a function),

ϕ: S' $\longrightarrow$ S (a surjective partial function)

θ: O' $\longrightarrow$ O (a surjective partial function)

such that:

$$\phi(s')\delta_x = \phi(s'\delta'_{\psi(x)})$$

and

$$\phi(s')\lambda_x = \theta(s'\lambda'_{\psi(x)}) \quad \text{(for a Mealy machine)}$$

or

$$\phi(s')\lambda = \theta(s'\lambda') \quad \text{(for a Moore machine)}.$$

The realization of state and output behaviour is a special case of the realization of output behaviour. If function ϕ in definition 2.2 maps each state of M onto a single state of M' and ϕ is a one-to-one function then definition 2.2 is equivalent to definition 2.3.

Since, the partition concept has to be used for analyzing the information streams in a machine, a special case of realization will be considered for which function ϕ maps each state of M onto a single state of M',i.e. ϕ:S $\longrightarrow$ S'.

<u>DEFINITION</u> <u>2.4</u> Machine $M' = (I',S',O',\delta',\lambda')$ is a ***single-state output behaviour realization*** of machine $M = (I,S,O,\delta,\lambda)$ *if and only if* the following relations exist:

ψ: $I \longrightarrow I'$ (a function),

ϕ: $S \longrightarrow S'$ (a function),

θ: $O' \longrightarrow O$ (a surjective partial function) ,

and this relations satisfy the following conditions:

$$\phi(s)\delta'_{\psi(x)} = \phi(s\delta_x)$$

and

$$s\lambda_x = \theta(\phi(s)\lambda'_{\psi(x)}) \qquad \text{(for a Mealy machine)}$$

or

$$s\lambda = \theta(\phi(s)\lambda') \qquad \text{(for a Moore machine)}$$

for all $s \in S$ and $x \in I$.

Since in this work only the single-state output behaviour realizations are considered, they will be called simply output behaviour realizations.

In a ***full-decomposition with the output behaviour realization*** of sequential machine M, we have to find the partial machines M_1 and M_2 as well as the mappings:

ψ: $I \longrightarrow I_1 x I_2$,

ϕ: $S \longrightarrow S_1 x S_2$,

θ: $O_1 x O_2 \longrightarrow O$,

that the machines M_1 and M_2 together with the mappings ψ, ϕ, θ realize the behaviour of a machine M.

We will say that a full-decomposition is ***nontrivial*** *if and only if*:

$|I_1|<|I| \wedge |I_2|<|I| \vee |S_1|<|S| \wedge |S_2|<|S| \vee |O_1|<|O| \wedge |O_2|<|O|$, where $|Z|$ – number of elements in the set Z.

Decompositions can be classified according to the kind of connections between the component machines.

In general, each of the component machines can use the information about the state or output of the other component machine in order to compute its own next state and output (Fig.3.1).

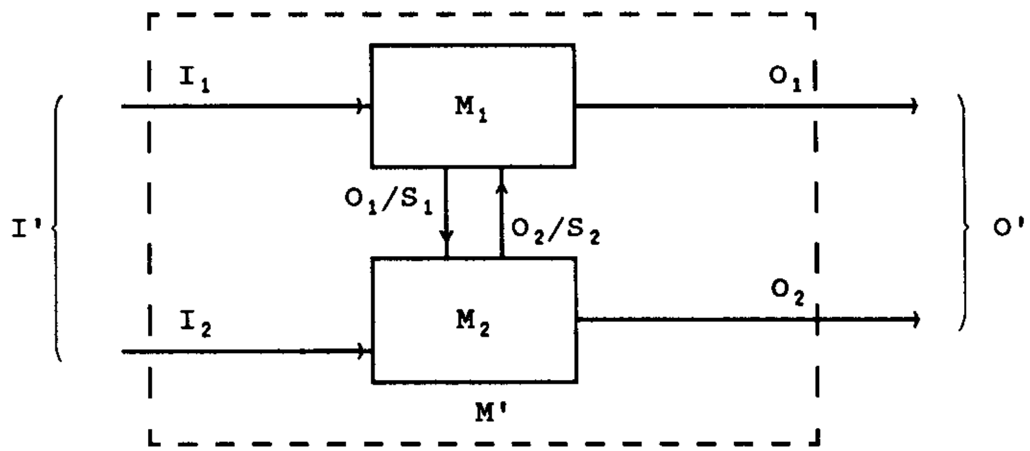

Fig 3.1 The information flow between the component machines in full-decomposition.

From the point of view of the strength of the connections between the component machines, the following sorts of full-decompositions can be distinguished:

(i) ***a parallel full-decomposition*** - each of the component machines can calculate its own next states and outputs independently of the other component machine, based only on information about its own internal state and partial information about the inputs (Fig.3.2),

(ii) ***a serial full-decomposition*** - one of the component machines, called the *tail* or *dependent machine* (M_2), uses the information about the outputs or states of the second machine, called the *head* or *independent machine* (M_1), plus partial information about the inputs in order to calculate its own next states and outputs (Fig.3.3),

(iii) ***a general full-decomposition*** - each of the component machines uses information about the outputs or states of the other component machine and partial information about the inputs in order to calculate its own next states and outputs (Fig.3.4).

The parallel full-decomposition and the serial full-decomposition can be treated as special cases of a general full-decomposition with zero information about one submachine used by another submachine.

From the point of view of the sort of information about a given submachine used by another submachine in order to calculate its next states and outputs, the following two types of full-decomposition can be distinguished:

(i) a decomposition with **information about the outputs**, called **type *O***,

(ii) a decomposition with **information about the internal states**, called ***type S***.

A given submachine can use the information about the "present" or the "next" state or output of the other submachine. So, the following two classes of full-decomposition occur:

(i) ***class P*** - a decomposition with information about the **present** state or output,

(ii) ***class N*** - a decomposition with information about the **next** state or output.

From the classification above, it immediately follows that the following cases of full-decomposition are feasible:

- one sort of parallel full-decomposition;
- four sorts of serial full decomposition: **PS**, **NS**, **PO**, and **NO** ;
- two sorts of general full-decomposition: **PS**, **PO**.

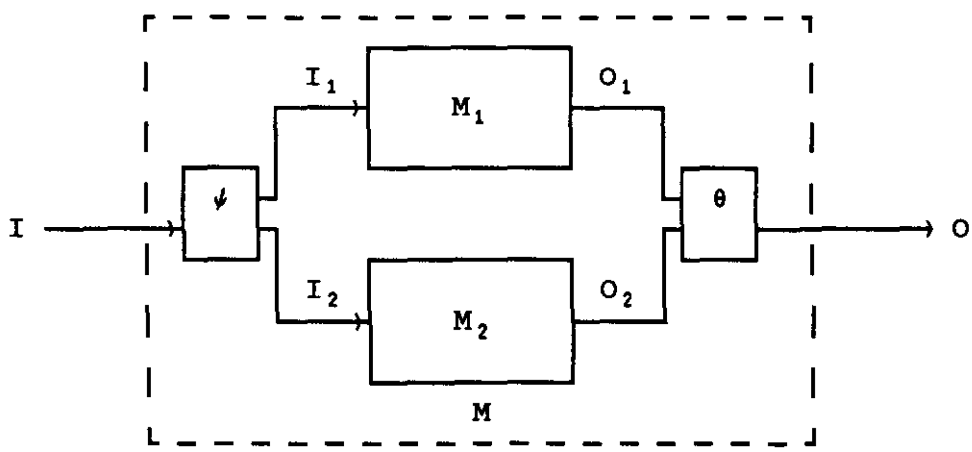

Fig 3.2 The parallel full-decomposition of a machine M into component machines M_1 and M_2.

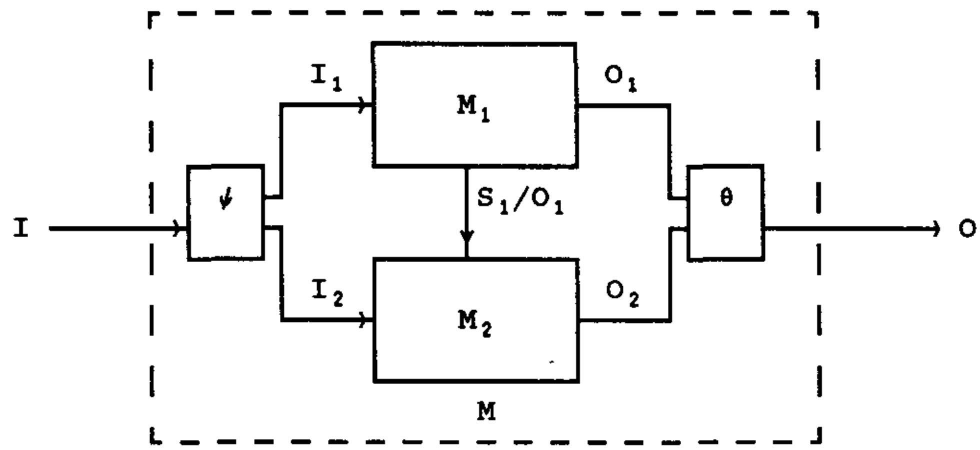

Fig 3.3 The serial full-decomposition of a machine M into component machines M_1 and M_2.

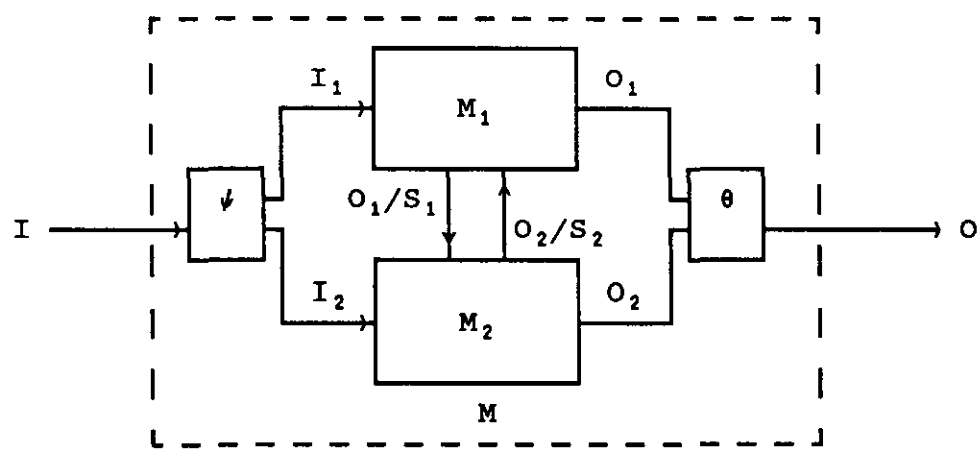

Fig 3.4 The general full-decomposition of a machine M into component machines M_1 and M_2 .

For a general full-decomposition, it is possible to have both the "pure" cases PS and PO and the "mixture" of types S and O and classes P and N (the first submachine can use the information about the state of the second and the second about the output of the first and vice versa ; the first submachine can use the information about the present state/output of the second submachine and the second can use the information about the next state/output of the first). In this report, "mixed" types are not considered because the definitions and theorems for them can be formulated easily as "mixtures" of the adequate definitions and theorems for the "pure" cases considered here.

From the considerations above, it follows that full-decomposition can be characterized by the type of connection between the component machines. The formal definitions of all connection types considered in this work are given below.

Let $s \in S_1$, $t \in S_2$, $x_1 \in I_1$, $x_2 \in I_2$.

<u>DEFINITION</u> <u>2.5</u> A *parallel connection* of two machines:

$$M_1 = (I_1, S_1, O_1, \delta^1, \lambda^1)$$

and

$$M_2 = (I_2, S_2, O_2, \delta^2, \lambda^2)$$

is the machine:

$$M_1 || M_2 = (I_1 x I_2, S_1 x S_2, O_1 x O_2, \delta^*, \lambda^*)$$

where:

$$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1), \delta^2(t,x_2))$$

and

$$\lambda^*((s,t),(x_1,x_2)) = (\lambda^1(s,x_1), \lambda^2(t,x_2))$$

(for Mealy machine)

or

$$\lambda^*((s,t)) = (\lambda^1(s), \lambda^2(t))$$

(for Moore machine)

<u>DEFINITION</u> <u>2.6</u> A *serial connection of type PS* of two machines:

$$M_1 = (I_1, S_1, O_1, \delta^1, \lambda^1)$$

and

$$M_2 = (I_2', S_2, O_2, \delta^2, \lambda^2) ,$$

for which $I_2' = S_1 x I_2$,

is the machine $M_1 \xrightarrow{PS} M_2 = (I_1 x I_2, S_1 x S_2, O_1 x O_2, \delta^*, \lambda^*)$,

where:

$$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1), \delta^2(t,(s,x_2)))$$

and

$\lambda^*((s,t),(x_1,x_2)) = (\lambda^1(s,x_1),\lambda^2(t,(s,x_2)))$

(for a Mealy machine)

or

$\lambda^*((s,t)) = (\lambda^1(s),\lambda^2(t))$

(for a Moore machine).

DEFINITION 2.7 A *serial connection of type NS* of two machines:

$$M_1 = (I_1,\ S_1,\ O_1,\ \delta^1,\ \lambda^1)$$

and

$$M_2 = (I_2',\ S_2,\ O_2,\ \delta^2,\ \lambda^2),$$

for which $I_2' = S_1 x I_2$,

is the machine $M_1 \xrightarrow{NS} M_2 = (I_1xI_2, S_1xS_2, O_1xO_2, \delta^*, \lambda^*)$,

where:

$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1),\delta^2(t,(\delta^1(s,x_1),x_2))$

and

$\lambda^*((s,t),(x_1,x_2)) = (\lambda^1(s,x_1),\lambda^2(t,(\delta^1(s,x_1),x_2))$

(for a Mealy machine)

or

$\lambda^*((s,t)) = (\lambda^1(s),\lambda^2(t))$

(for a Moore machine)

DEFINITION 2.8 A *serial connection of type PO* of two machines:

$$M_1 = (I_1,\ S_1,\ O_1,\ \delta^1,\ \lambda^1)$$

and

$$M_2 = (I_2',\ S_2,\ O_2,\ \delta^2,\ \lambda^2) ,$$

for which $I_2' = O_1 x I_2$,

is the machine $M_1 \xrightarrow{PO} M_2 = (I_1xI_2, S_1xS_2, O_1xO_2, \delta^*, \lambda^*)$,

where:

$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1),\delta^2(t,(y_1,x_2)))$

$\lambda^*((s,t),(x_1,x_2)) = (\lambda^1(s,x_1),\lambda^2(t,(y_1,x_2)))$

and $y_1 \epsilon O_1$: y_1 is the present output of M_1

(the output of M_1 contemporary with the state s of M_1)

(for a Mealy machine)

or

$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1),\delta^2(t,(\lambda^1(s),x_2))))$

$\lambda^*((s,t)) = (\lambda^1(s),\lambda^2(t))$

(for a Moore machine)

DEFINITION 2.9 A *serial connection of type NO* of two machines:

$$M_1 = (I_1, S_1, O_1, \delta^1, \lambda^1)$$

and

$$M_2 = (I_2', S_2, O_2, \delta^2, \lambda^2),$$

for which $I_2' = O_1 x I_2$

is the machine $M_1 \xrightarrow{NO} M_2 = (I_1xI_2, S_1xS_2, O_1xO_2, \delta^*, \lambda^*)$,

where:

$$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1), \delta^2(t,(\lambda^1(s,x_1),x_2)))$$
$$\lambda^*((s,t),(x_1,x_2)) = (\lambda^1(s,x_1), \lambda^2(t,(\lambda^1(s,x_1),x_2)))$$

(for a Mealy machine)

or

$$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1), \delta^2(t,(\lambda^1(\delta^1(s,x_1)),x_2)))$$
$$\lambda^*((s,t)) = (\lambda^1(s), \lambda^2(t))$$

(for a Moore machine)

DEFINITION 2.10 A *general connection of type PS* of two machines :

$$M_1 = (I_1', S_1, O_1, \delta^1, \lambda^1)$$

and

$$M_2 = (I_2', S_2, O_2, \delta^2, \lambda^2)$$

where:

$$I_1' = S_2xI_1 \ , \ I_2' = S_1xI_2 \ ,$$

is the machine:

$$M_1 \overset{PS}{\longleftrightarrow} M_2 = (I_1xI_2, S_1xS_2, O_1xO_2, \delta^*, \lambda^*) \ ,$$

where:

$$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,(t,x_1)), \delta^2(t,(s,x_2))$$

and

$$\lambda^*((s,t),(x_1,x_2)) = (\lambda^1(s,(t,x_1)), \lambda^2(t,(s,x_2))$$

(for a Mealy machine)

or

$$\lambda^*((s,t)) = (\lambda^1(s), \lambda^2(t))$$

(for a Moore machine)

DEFINITION 2.11 A *general connection of type PO* of two machines:

$$M_1 = (I_1', S_1, O_1, \delta^1, \lambda^1)$$

and

$$M_2 = (I_2', S_2, O_2, \delta^2, \lambda^2) \ ,$$

where:

$$I_1' = O_2xI_1 \ , \ I_2' = O_1xI_2$$

is the machine:

$$M_1 \overset{PO}{\longleftrightarrow} M_2 = (I_1xI_2, S_1xS_2, O_1xO_2, \delta^*, \lambda^*) \ ,$$

where:

$$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,(y_2,x_1)), \delta^2(t,(y_1,x_2)))$$
$$\lambda^*((s,t),(x_1,x_2)) = (\lambda^1(s,(y_2,x_1)), \lambda^2(t,(y_1,x_2)))$$

and $y_1 \in O_1$, $y_2 \in O_2$ (present outputs of M_1 and M_2)
(for a Mealy machine)

or

$$\delta^*((s,t),(x_1,x_2)) = (\delta^1(s,(\lambda^2(t),x_1)),\delta^2(t,(\lambda^1(s),x_2)))$$
$$\lambda^*((s,t)) = (\lambda^1(s),\lambda^2(t))$$

(for a Moore machine)

<u>**DEFINITION**</u> <u>2.12</u> The machine $M_1 \Diamond M_2$ is a ***full decomposition of type*** $\Diamond$ of machine M *if and only if* the connection of a given type $\Diamond$ of machines M_1 and M_2 realizes M,
where:

$$\Diamond = \; || \; , \stackrel{PS}{\longrightarrow} , \stackrel{NS}{\longrightarrow} , \stackrel{PO}{\longrightarrow} , \stackrel{NO}{\longrightarrow} , \stackrel{PS}{\longleftrightarrow} , \stackrel{PO}{\longleftrightarrow} .$$

Each of the types of a full-decomposition defined above can be considered to be a full-decomposition with the realization of the output behaviour or a full-decomposition with the realization of the state and output behaviour. Full-decompositions with the state and output behaviour realization have been considered in [16]. In the next paragraphs, the theorems concerning the existence of different types of a full- decomposition with the output behaviour realization will be formulated and proved. Only the proves for a Mealy machine are presented in the report, because those for a Moore machine are analogous.

<u>3. Partitions, partition pairs and partition trinities.</u>

The concepts of partitions and partition pairs introduced by Hartmanis [11][12] and partition trinities introduced by Hou [14][15] are useful tools for analyzing the information flow in machines or between machines; therefore they were used in this work.
Let S be any set of elements.

<u>**DEFINITION**</u> <u>3.1</u> ***Partition*** π on S is defined as follows:

$$\pi = \{B_i \mid B_i \subseteq S \text{ and } B_i \cap B_j = 0 \text{ for } i \neq j \text{ and } \bigcup_i B_i = S\},$$

i.e. a partition π on S is a set of disjoint subsets of S whose set union is S.

For a given $s \in S$, the block of a partition π containing s is denoted as $[s]\pi$ and $[s]\pi = [t]\pi$ is written to denote that s and t are in the same block of π. Similarly, the block of a partition π containing S',where $S' \subseteq S$, is denoted by $[S']\pi$.

A partition containing only one element of S in each block is called a *zero partition* and denoted by $\pi_S(0)$. A partition containing all the elements of S in one block is called an *identity* or *one partition* and is denoted by $\pi_S(I)$.

Let π_1 and π_2 be two partitions on S.

DEFINITION 3.2 *Partition product* $\pi_1 \cdot \pi_2$ is the partition on S such that $[s]\pi_1 \cdot \pi_2 = [t]\pi_1 \cdot \pi_2$ *if and only if* $[s]\pi_1 = [t]\pi_1$ and $[s]\pi' = [t]\pi_2$.

DEFINITION 3.3 *Partition sum* $\pi_1+\pi_2$ is the partition on S such that $[s]\pi_1+\pi_2 = [t]\pi_1+\pi_2$ *if and only if* a sequence: $s=s_0, s_1, \ldots, s_n=t$, $s_i \in S$ for i=1..n , exists for which either $[s_i]\pi_1 = [s_{i+1}]\pi_1$ either $[s_i]\pi_2 = [s_{i+1}]\pi_2$, $0 \le i \le n-1$.

From the above definitions, it follows that the blocks of $\pi_1 \cdot \pi_2$ are obtained by intersecting the blocks of π_1 and π_2, while the blocks of $\pi_1+\pi_2$ are obtained by uniting all the blocks of π_1 and π_2 which contain common elements.

DEFINITION 3.4 π_2 is *greater than or equal to* π_1: $\pi_1 \le \pi_2$ *if and only if* each block of π_1 is included in a block of π_2.

Thus $\pi_1 \le \pi_2$ *if and only if* $\pi_1 \cdot \pi_2 = \pi_1$ *if and only if* $\pi_1+\pi_2 = \pi_2$.

Let S_π be the set of all partitions on S.Since the relation $\le$ is a relation of *partial ordering* (i.e. it is reflexive, antisymmetric and transitive), $(S_\pi, \le)$ is a *partially ordered set*.

Let $(Z, \le)$ be a partially ordered set and T be a subset of Z.

DEFINITION 3.5 z, $z \in Z$, is the *least upper bound (LUB)* of T *if and only if* :

(i) $\forall t \in T$: $z \ge t$,

(ii) $\forall t \in T$: *if* $z' \ge t$ *then* $z' \ge z$.

z, $z \in Z$, is the *greatest lower bound (GLB)* of T *if and only if*:

(i) $\forall t \in T$: $z \le t$,

(ii) $\forall t \in T$: *if* $z' \le t$ *then* $z' \le z$.

DEFINITION 3.6 A partially ordered set $L = (Z, \le)$, which has a LUB and a GLB for every pair of elements, is called a *lattice*.

It is evident that the set of all partitions on S together with the relation of a partial ordering $\le$ form a lattice with $GLB(\pi_1,\pi_2) = \pi_1 \cdot \pi_2$ and $LUB(\pi_1,\pi_2) = \pi_1 + \pi_2$.

Let π_S, τ_S, π_I, π_O be the partitions on $M=(I, S, O, \delta, \lambda)$, in particular: π_S, τ_S on S, π_I on I, π_O on O.

DEFINITION 3.7

(i) (π_S,τ_S) is an ***S-S partition pair*** *if and only if*
$\forall B\in\pi_S\ \forall x\in I : B\bar{\delta}_x \subseteq B',\ B'\in\tau_S$.

(ii) (π_I,π_S) is an ***I-S partition pair*** *if and only if*
$\forall A\in\pi_I\ \forall s\in S : s\bar{\delta}_A \subseteq B\ ,\ B\in\pi_S$.

(iii) (π_S,π_O) is an ***S-O partition pair*** *if and only if*
$\forall B\in\pi_S\ \forall x\in I : B\bar{\lambda}_x \subseteq C\ ,\ C\in\pi_O$ (Mealy case)
or
$\forall B\in\pi_S : B\bar{\lambda} \subseteq C\ ,\ C\in\pi_O$ (Moore case).

(iv) (π_I,π_O) is an ***I-O partition pair*** *if and only if*
$\forall A\in\pi_I\ \forall s\in S : s\bar{\lambda}_A \subseteq C\ ,\ C\in\pi_O$ (Mealy case)
or
$\forall A\in\pi_I\ \forall s\in S : s\lambda \subseteq C\ ,\ C\in\pi_O$ (Moore case).

The practical meaning of the notions introduced above is as follows:

(π_S,τ_S) is an S-S partition pair *if and only if* the blocks of π_S are mapped by M into the blocks of τ_S. Thus, if the block of π_S which contains the present state of the machine M is known and the present input of M too, it is possible to compute unambiguously the block of τ_S which contains the next state of M for the states from a given block of π_S and a given input. The interpretation of the notions of I-S, S-O and I-O partition pairs is similar.

In the case of a Moore machine, the definition of an I-O pair is trivial, because each (π_I,π_S) satisfies it (the output of M is defined by the state of M unambiguously).

DEFINITION 3.8 Partition π_S has a *substitution property* (it is an *SP-partition*) *if and only if* (π_S,π_S) is an S-S pair.

DEFINITION 3.9 *Partition trinity* $T = (\pi_I,\pi_S,\pi_O)$ on the machine $M = (I, S, O, \delta, \lambda)$ is an ordered triple of partitions on sets I, S and O, respectively, which satisfies the following conditions:

$\forall A\in\pi_I\ \forall B\in\pi_S : B\bar{\delta}_A \subseteq B',\ B'\in\pi_S$ *and* $B\bar{\lambda}_A \subseteq C\ ,\ C\in\pi_O$.

Thus, if (π_I, π_S, π_O) is a partition trinity on M and the block B of π_S which contains the present state of M is known and the block A of π_I which contains the present input of M is known too, it is possible to compute unambiguously block B' of π_S that contains the next state of M and block C of π_O that contains the output of M for the states from block B and inputs from block A.

For completely spacified machines, it has been proved that (π_I, π_S, π_O) is a <u>partition trinity</u> on M *if and only if* (π_S, π_S) is an <u>S-S pair</u>, (π_I, π_S) is an <u>I-S pair</u> , (π_S, π_O) is an <u>S-O pair</u> and (π_I, π_O) is an <u>I-O pair</u> on M [14][15].

It was shown in [14] that the set of trinities on a machine M forms a finite trinity lattice with

$$GLB(T_1,T_2) = T_1 \odot T_2 \quad \text{and} \quad LUB(T_1,T_2) = T_1 \oplus T_2 ,$$

where $\odot$ and $\oplus$ are defined as a collection of pairwise operations "$\cdot$" and "+" for partitions of the same type (input,state,output) of trinities of T_1 and T_2 .

<u>4. Parallel full-decomposition.</u>

<u>THEOREM 4.1</u> A machine $M = (I,S,O,\delta,\lambda)$ has a nontrivial parallel full-decomposition with the realization of the output behaviour *if* two partition trinities on M: (π_I, π_S, π_O) and (τ_I, τ_S, τ_O) exist and they satisfy the following conditions:

(i) $\pi_O \cdot \tau_O = \pi_O(0)$,

(ii) $|\pi_I|<|I| \wedge |\tau_I|<|I| \vee |\pi_S|<|S| \wedge |\tau_S|<|S| \vee |\pi_O|<|O| \wedge |\tau_O|<|O|$.

<u>Proof</u> (for the case of a Mealy machine)

Let $M_1 = (\pi_I, \pi_S, \pi_O, \delta^1, \lambda^1)$ and $M_2 = (\tau_I, \tau_S, \tau_O, \delta^2, \lambda^2)$ be two sequential machines satisfying the following conditions:

(1) (π_I, π_S, π_O) and (τ_I, τ_S, τ_O) satisfy the conditions of theorem 4.1,

(2) $\forall B1 \in \pi_S \ \forall A1 \in \pi_I$: $\quad B1\delta^1{}_{A1} = [B1\overline{\delta}_{A1}]\pi_S$,

$\quad B1\lambda^1{}_{A1} = [B1\overline{\lambda}_{A1}]\pi_I$,

(3) $\forall B2 \in \tau_S \ \forall A2 \in \tau_I$: $\quad B2\delta^2{}_{A2} = [B2\overline{\delta}_{A2}]\tau_S$,

$\quad B2\lambda^2{}_{A2} = [B2\overline{\lambda}_{A2}]\tau_I$.

Since (π_I, π_S, π_O) is a partition trinity (1), $B1\overline{\delta}_{A1}$ is placed in just one block of π_S and $B1\overline{\lambda}_{A1}$ in only one block of π_O . This means, that $B1\delta^1{}_{A1}$ and $B1\lambda^1{}_{A1}$ are defined unambiguously. Similarly, since (τ_I, τ_S, τ_O) is a partition trinity (1), $B2\delta^2{}_{A2}$ and $B2\lambda^2{}_{A2}$

are defined unambiguously. So, each of the partial machines M_1 and M_2 can calculate its next states and outputs unambiguously.

Let ψ: $I \longrightarrow \pi_I x \tau_I$ be an injective function,

ϕ: $S \longrightarrow \pi_s x \tau_s$ be an injective function,

θ: $\pi_0 x \tau_0 \longrightarrow O$ be a surjective partial function

and

(4) $\psi(x) = ([x]\pi_I, [x]\tau_I)$,

(5) $\phi(s) = ([s]\pi_s, [s]\tau_s)$,

(6) $\theta(C1, C2) = C1 \cap C2$ if $C1 \cap C2 \neq 0$.

It is proved below that the parallel connection of the machines M_1 and M_2 defined above realizes a machine M.

Since $\pi_0 \cdot \tau_0 = \pi_0(0)$ (1) , θ is a one-to-one function and for $C1 \cap C2 \neq 0$:

(7) $(C1, C2) \in O$.

Therefore, $\forall s \in S \; \forall x \in I$:

$$\phi(s)\delta^*_{\psi(x)} =$$

$$= ([s]\pi_s, [s]\tau_s)\delta^*_{([x]\pi_I, [x]\tau_I)} \qquad ((4),\ (5))$$

$$= ([s]\pi_s \delta^1_{[x]\pi_I}, [s]\tau_s \delta^2_{[x]\tau_I}) \qquad \text{(definition 2.5)}$$

$$= ([[s]\pi_s \bar{\delta}_{[x]\pi_I}]\pi_s, [[s]\tau_s \bar{\delta}_{[x]\tau_I}]\tau_s) \qquad ((2),\ (3))$$

$$= ([s\delta_x]\pi_s, [s\delta_x]\tau_s) \qquad ((1))$$

$$= \phi(s\delta_x) \qquad ((5))$$

and similary:

$$\theta(\phi(s)\lambda^*_{\psi(x)}) =$$

$$= \theta(([s]\pi_s, [s]\tau_s)\lambda^*_{([x]\pi_I, [x]\tau_I)}) \qquad ((4),\ (5))$$

$$= \theta([s]\pi_s \lambda^1_{[x]\pi_I}, [s]\tau_s \lambda^2_{[x]\tau_I}) \qquad \text{(definition 2.5)}$$

$$= [s]\pi_s \lambda^1_{[x]\pi_I} \cap [s]\tau_s \lambda^2_{[x]\tau_I} \qquad ((6))$$

$$= [[s]\pi_s \bar{\lambda}_{[x]\pi_I}]\pi_0 \cap [[s]\pi_s \bar{\lambda}_{[x]\tau_I}]\tau_0 \qquad ((2),\ (3))$$

$$= [s\lambda_x]\pi_0 \cap [s\lambda_x]\tau_0 \qquad ((1))$$

$$= s\lambda_x \qquad (\ \pi_0 \cdot \tau_0 = \pi_0(0)\ )$$

From the above calculations and definitions 2.4, 2.5 and 2.12, it follows immediately that the parallel connection of machines M_1 and M_2 realizes M, i.e. M has a parallel full-decomposition with the output behaviour realization. If condition (ii) of theorem 4.1 is satisfied, then the decomposition is nontrivial. □

Theorem 4.1 has the following interpretation:

Since (π_I,π_S,π_O) is a partition trinity, based only on the information about the block of π_I containing the input of M and the block of π_S containing the present state of M (i.e information about the input and present state of M_1) machine M_1 can calculate unambiguously the block of π_S in which the next state of M is contained , as well as, the block of π_O that contains the output of M for the input from a given block of π_I and the present state from a given block of π_S (i.e. M_1 can calculate its next state and output). Similarly, since (τ_I,τ_S,τ_O) is a partition trinity, machine M_2, based only on the information about its input and present state (i.e. knowledge of the adequate block of τ_I and block of τ_S), can calculate its next state and output (i.e. the adequate blocks of τ_S and τ_O).

Since $\pi_O \cdot \tau_O = \pi_O(0)$, the knowledge of of the block of π_O and the block of τ_O in which the output of M is contained makes it possible to calculate this output. So, the machines M_1 and M_2 together can calculate the output of M unambiguously.

A special case of theorem 4.1 for:
$|\pi_I|<|I| \wedge |\tau_I|<|I| \wedge (|\pi_S|=|S| \wedge |\pi_O|=|O| \vee |\tau_S|=|S| \wedge |\tau_O|=|O|)$
expresses, in fact, the input redundancy. In this case, machine M should be replaced with machine M_1 or M_2, having fewer inputs and realizing M, instead of being decomposed. Similar special cases exist for all the other theorems presented in this report.

5. Serial full-decomposition of type PS.

Let τ_I, τ_S, τ_O be partitions on a machine M on I, S and O respectively.

DEFINITION 5.1 (τ_I,τ_S,τ_O) is a *present-state-dependent trinity* for an independent state partition ξ_S *if and only if* τ_I, τ_S and τ_O satisfy the following conditions:

(i) (τ_I,τ_S) is an I-S partition pair,

(ii) $(\tau_S \cdot \xi_S,\tau_S)$ is a S-S partition pair,

(iii) $(\tau_S \cdot \xi_S,\tau_O)$ is a S-O partition pair

and

(τ_I,τ_O) is an I-O partition pair (for a Mealy machine),

or

(τ_S,τ_O) is a S-O partition pair (for a Moore machine)

In other words, (τ_I,τ_S,τ_O) is a present-state-dependent trinity if and only if, based only on the knowledge of the block of a partition τ_I containing the input of M and the knowledge of the blocks of partitions τ_S and ξ_S containing the present state of M, it is possible to calculate the block of τ_S in which the next state of M will be contained. In the case of a Mealy machine, based on the same information, it is possible to calculate the block of τ_O in which the output of M will be contained for the given input and state. While, in the case of Moore machine, based on the knowledge of the block of a partition τ_S in which the state of M is contained, it is possible to calculate the block of τ_O in which the output of M will be contained for the state from a given block of τ_S.

THEOREM 5.1 A machine M has a nontrivial serial full-decomposition of type PS with the realization of the output behaviour *if* a partition trinity (π_I,π_S,π_O) and a present-state-dependent partition trinity (τ_I,τ_S,τ_O) for $\xi_S = \pi_S$ exist and they satisfy the following conditions:

(i) $\pi_O \cdot \tau_O = \pi_O(0)$,

(ii) $|\pi_I|<|I| \wedge |\pi_S|\cdot|\tau_I|<|I| \vee |\pi_S|<|S| \wedge |\tau_S|<|S| \vee |\pi_O|<|O| \wedge$
$\wedge |\tau_O|<|O|$.

Proof (for the case of a Mealy machine)

Let $M_1 = (\pi_I,\pi_S,\pi_O,\delta^1,\lambda^1)$ and $M_2 = (\pi_S x \tau_I,\tau_S,\tau_O,\delta^2,\lambda^2)$ be two machines that satisfy the following conditions:

(1) (π_I,π_S,π_O) and (τ_I,τ_S,τ_O) satisfy the conditions of the theorem 6.1 ,

(2) $\forall B1 \epsilon \pi_S\ \forall A1 \epsilon \pi_I$: $B1\delta^1{}_{A1} = [B1\bar{\delta}_{A1}]\pi_S$, $B1\lambda^1{}_{A1} = [B1\bar{\lambda}_{A1}]\pi_O$,

(3) $\forall B1 \epsilon \pi_S\ \forall B2 \epsilon \tau_S\ \forall A2 \epsilon \tau_I$:

$B2\delta^2{}_{(B1,A2)} = [(B1 \cap B2)\bar{\delta}_{A2}]\tau_S$, $B2\lambda^2{}_{(B1,A2)} = [(B1 \cap B2)\bar{\lambda}_{A2}]\tau_O$.

Since (π_I,π_S,π_O) is a partition trinity (1), $B1\bar{\delta}_{A1}$ is placed in just one block of π_S and $B1\bar{\lambda}_{A1}$ in only one block of π_O . This means, that $B1\delta^1{}_{A1}$ and $B1\lambda^1{}_{A1}$ are defined unambiguously.

Since (τ_I,τ_S,τ_O) is a present-state-dependent trinity (1), $(B1 \cap B2)\bar{\delta}_{A2}$ is placed in just one block of τ_S and $(B1 \cap B2)\bar{\lambda}_{A2}$ is placed in only one block of τ_O . This means, that $B2\delta^2{}_{(B1,A2)}$ and $B2\lambda^2{}_{(B1,A2)}$ are defined unambiguously.

Let ψ: $I \longrightarrow \pi_I x \tau_I$ be an injective function,

ϕ: $S \longrightarrow \pi_S x \tau_S$ be an injective function,

θ: $\pi_0 x \tau_0 \longrightarrow O$ be a surjective partial function

and

(4) $\psi(x) = ([x]\pi_I, [x]\tau_I)$,

(5) $\phi(s) = ([s]\pi_S, [s]\tau_S)$,

(6) $\theta(C1,C2) = C1 \cap C2$ if $C1 \cap C2 \neq 0$.

It is proved below that the serial connection of type PS of the machines M_1 and M_2 defined above realizes the output behaviour of machine M.

Since $\pi_0 \cdot \tau_0 = \pi_0(0)$ (1) , θ is a one-to-one function and for $C1 \cap C2 \neq 0$:

(7) $(C1,C2) \in O$.

Therefore, $\forall s \in S$ $\forall x \in I$:

$$\phi(s)\delta^*_{\psi(x)} =$$

$$= ([s]\pi_S,[s]\tau_S)\delta^*_{([x]\pi_I,[x]\tau_I)} \qquad ((4),\ (5))$$

$$= ([s]\pi_S\delta^1_{[x]\pi_I},[s]\tau_S\delta^2_{([s]\pi_S,[x]\tau_I)}) \qquad \text{(definition 2.6)}$$

$$= ([[s]\pi_S\overline{\delta}_{[x]\pi_I}]\pi_S,[([s]\tau_S \cap [s]\pi_S)\overline{\delta}_{[x]\tau_I}]\tau_S) \qquad ((2),\ (3))$$

$$= ([s\delta_x]\pi_S,[s\delta_x]\tau_S) \qquad ((1))$$

$$= \phi(s\delta_x) \qquad ((5))$$

and similary:

$$\theta(\phi(s)\lambda^*_{\psi(x)}) =$$

$$= \theta(([s]\pi_S,[s]\tau_S)\lambda^*_{([x]\pi_I,[x]\tau_I)}) \qquad ((4),\ (5))$$

$$= \theta([s]\pi_S\lambda^1_{[x]\pi_I},[s]\tau_S\lambda^2_{([s]\pi_S,[x]\tau_I)}) \qquad \text{(definition 2.6)}$$

$$= [s]\pi_S\lambda^1_{[x]\pi_I} \cap [s]\tau_S\lambda^2_{([s]\pi_S,[x]\tau_I)} \qquad ((6))$$

$$= [[s]\pi_S\overline{\lambda}_{[x]\pi_I}]\pi_0 \cap [([s]\tau_S \cap [s]\pi_S)\overline{\lambda}_{[x]\tau_I}]\tau_0 \qquad ((2),\ (3))$$

$$= [s\lambda_x]\pi_0 \cap [s\lambda_x]\tau_0 \qquad ((1))$$

$$= s\lambda_x \qquad (\ \pi_0 \cdot \tau_0 = \pi_0(0)\ )$$

From the above calculations and definitions 2.4, 2.6 and 2.12, it follows immediately that the serial connection of type PS of machines M_1 and M_2 realizes M, i.e. M has a serial full-decomposition of type PS with the output behaviour realization. If condition (ii) of theorem 5.1 is satisfied, the decomposition is nontrivial. □

Theorem 5.1 has a straightforward interpretation.

Since (π_I, π_S, π_O) is a partition trinity, based only on the information about the block of a partition π_I containing the input and the block of a partition π_S containing the present state of machine M (i.e. information about the input and present state of M_1), machine M_1 can calculate unambiguously the block of π_S in which the next state of M is contained and the block of π_O in which the output of M is contained for the given input and present state (i.e M_1 is able to calculate its next state and output).

Since (τ_I, τ_S, τ_O) is a present-state-dependent trinity, based only on the information about the block of a partition τ_I containing the input and the blocks of partitions τ_S and π_S containing the present state of the machine M (i.e. information about the primary input and the present state of M_2 and about the present state of M_1 being a part if the input to M_2), machine M_2 is able to calculate unambiguously the block of τ_S in which the next state of M is contained and, in the case of a Mealy machine, the block of τ_O in which the output of M is contained for the given input and present state (i.e. M_2 can calculate its next state and output). In the case of a Moore machine, M_2 is able to calculate the block of τ_O in which the output of M is contained, based only on information about the block of τ_S in which the state of M is contained.

Since $\pi_O \cdot \tau_O = \pi_O(0)$, with information about the blocks of π_O calculated by M_1 and the blocks of τ_O calculated by M_2 (i.e. information about the outputs of M_1 and M_2), it is possible to calculate unambiguously the outputs of machine M.

6. Serial full-decomposition of type NS.

Let τ_I, τ_S, τ_O be partitions on machine M, on I, S and O respectiviely, and ξ_S be another partition on S.

DEFINITION 6.1 (τ_I, τ_S, τ_O) is a ***next-state-dependent trinity*** for *an independent state partition* ξ_S *if and only if* τ_I, τ_S, τ_O satisfy one of the following conditions for a given ξ_S:

(i) $\forall s,t \in S \ \forall x_1,x_2 \in I$:

if $[s]\tau_S=[t]\tau_S \wedge [x_1]\tau_I=[x_2]\tau_I \wedge [s\delta_{x_1}]\xi_S=[t\delta_{x_2}]\xi_S$

then $[s\delta_{x_1}]\tau_S=[t\delta_{x_2}]\tau_S \wedge [s\lambda_{x_1}]\tau_O=[t\lambda_{x_2}]\tau_O$

(for a Mealy machine),

(ii) $\forall s,t \in S\ \forall x_1,x_2 \in I$:

if $[s]\tau_S=[t]\tau_S \wedge [x_1]\tau_I=[x_2]\tau_I \wedge [s\delta_{x_1}]\xi_S=[t\delta_{x_2}]\xi_S$

then $[s\delta_{x_1}]\tau_S=[t\delta_{x_2}]\tau_S \wedge [(s\delta_{x_1})\lambda]\tau_O=[(t\delta_{x_2})\lambda]\tau_O$

(for a Moore machine).

In other words, (τ_I,τ_S,τ_O) is a next-state-dependent trinity for an independent state partition ξ_S if and only if, based only on the knowledge of the block of a partition τ_I containing the input of machine M , knowledge of the block of a partition τ_S containing the present state of M and knowledge of the block of a partition ξ_S in which the next state of M is contained for a given input and state, it is possible to calculate the block of τ_S in which the next state of M will be contained and the block of τ_O in which the output of M will be contained.

<u>THEOREM</u> <u>6.1</u> A machine M has a nontrivial serial full-decomposition of type NS with the realization of the output behaviour *if* such a partition trinity (π_I,π_S,π_O) and such a next-state-dependent trinity (τ_I,τ_S,τ_O) for $\xi_S=\pi_S$ exist that the following conditions are satisfied:

(i) $\pi_S\cdot\tau_S = \pi_S(0)$ *and* $\pi_O\cdot\tau_O = \pi_O(0)$,

(ii) $|\pi_I|<|I|$, $|\pi_S|<|S|$, $|\pi_O|<|O|$, $|\pi_S|\cdot|\tau_I|<|I|$, $|\tau_S|<|S|$, $|\tau_O|<|O|$.

<u>Proof</u> (for the case of a Mealy machine)

Let $M_1 = (\pi_I,\pi_S,\pi_O,\delta^1,\lambda^1)$ and $M_2 = (\pi_S x \tau_I,\tau_S,\tau_O,\delta^2,\lambda^2)$ be two machines for which the following conditions are satisfied:

(1) (π_I,π_S,π_O) and (τ_I,τ_S,τ_O) satisfy the conditions of the theorem 6.1 ,

(2) $\forall B1\in\pi_S\ \forall A1\in\pi_I$: $B1\delta^1_{A1} = [B1\overline{\delta}_{A1}]\pi_S$, $B1\lambda^1_{A1} = [B1\lambda_{A1}]\pi_O$,

(3) $\forall B2\in\tau_S\ \forall A2\in\tau_I\ \forall B1'\in\pi_S$:

$$B2\delta^2_{(B1',A2)} = [\{s\delta_x|\ s\in B2,\ x\in A2,\ s\delta_x\in B1'\}]\tau_S \ ,$$

$$B2\lambda^2_{(B1',A2)} = [\{s\lambda_x|\ s\in B2,\ x\in A2,\ s\delta_x\in B1'\}]\tau_O \ .$$

Since (π_I,π_S,π_O) is a partition trinity (1), $B1\overline{\delta}_{A1}$ is placed in just one block of π_S and $B1\overline{\lambda}_{A1}$ is placed in only one block of π_O. This means that $B1\delta^1_{A1}$ and $B1\lambda^1_{A1}$ are defined unambiguously.

Since (τ_I, τ_S, τ_O) is a next-state-dependent trinity for $\xi_S = \pi_S$ (1), the following condition is satisfied:

(4) $\forall s,t \in S\ \forall x_1, x_2 \in I$:

if $[s]\tau_S = [t]\tau_S \wedge [x_1]\tau_I = [x_2]\tau_I \wedge [s\delta_{x_1}]\pi_S = [t\delta_{x_2}]\pi_S$

then $[s\delta_{x_1}]\tau_S = [t\delta_{x_2}]\tau_S \wedge [s\lambda_{x_1}]\tau_O = [t\lambda_{x_2}]\tau_O$.

From (4), it follows that $B2\delta^2_{(B1',A2)}$ and $B2\lambda^2_{(B1',A2)}$ are defined unambiguously because $\{s\delta_x \mid s \in B2,\ x \in A2,\ s\delta_x \in B1'\}$ is located in only one block of τ_S and $\{s\lambda_x \mid s \in B2,\ x \in A2,\ s\delta_x \in B1'\}$ is in just one block of τ_O.

Let $\psi: I \longrightarrow \pi_I x \tau_I$ be an injective function,

$\phi: S \longrightarrow \pi_S x \tau_S$ be an injective function,

$\theta: \pi_O x \tau_O \longrightarrow O$ be a surjective partial function

and

(5) $\psi(x) = ([x]\pi_I, [x]\tau_I)$,

(6) $\phi(s) = ([s]\pi_S, [s]\tau_S)$,

(7) $\theta(C1,C2) = C1 \cap C2$ if $C1 \cap C2 \neq 0$.

It will be proved below that the serial connection of type NS of defined above machines M_1 and M_2 realizes the output behaviour of machine M.

Since $\pi_O \cdot \tau_O = \pi_O(0)$ (1) , θ is a one-to-one function and for $C1 \cap C2 \neq 0$:

(8) $(C1,C2) \in O$.

So, $\forall s \in S\ \forall x \in I$:

$\phi(s)\delta^*_{\psi(x)}) =$

$= ([s]\pi_S, [s]\tau_S)\delta^*_{([x]\pi_I, [x]\tau_I)}$ ((5), (6))

$= ([s]\pi_S\delta^1_{[x]\pi_I}, [s]\tau_S\delta^2_{([s\delta_x]\pi_S, [x]\tau_I)})$ (definition 2.7)

$= ([[s]\pi_S\bar{\delta}_{[x]\pi_I}]\pi_S, [\{s\delta_x \mid [s]\tau_S \wedge [s\delta_x]\pi_S \wedge [x]\tau_I\}]\tau_S)$ ((2), (3))

$= ([s\delta_x]\pi_S, [s\delta_x]\tau_S)$ ((1))

$= \phi(s\delta_x)$ ((6))

and similary:

$\theta(\phi(s)\lambda^*_{\psi(x)}) =$

$= \theta(([s]\pi_S, [s]\tau_S)\lambda^*_{([x]\pi_I, [x]\tau_I)})$ ((5), (6))

$= \theta([s]\pi_S\lambda^1_{[x]\pi_I}, [s]\tau_S\lambda^2_{([s\delta_x]\pi_S, [x]\tau_I)})$ (definition 2.7)

$$= [s]\pi_S \lambda^1{}_{[x]\pi_I} \cap [s]\tau_S \lambda^2{}_{([s\delta_x]\pi_S,[x]\tau_I)} \qquad ((7))$$

$$= [[s]\pi_S \overline{\lambda}_{[x]\pi_I}]\pi_O \cap [\{s\lambda_x | [s]\tau_S \wedge [s\delta_x]\pi_S \wedge [x]\tau_I\}]\tau_O \qquad ((2),(3))$$

$$= [s\lambda_x]\pi_O \cap [s\lambda_x]\tau_O \qquad ((1))$$

$$= s\lambda_x \qquad (\pi_O \cdot \tau_O = \pi_O(0))$$

From the above calculations and definitions 2.4, 2.7 and 2.12, it follows that the serial connection of type NS of machines M_1 and M_2 realizes M, i.e. M has a serial full-decomposition of type NS with the output behaviour realization. If condition (ii) of theorem 6.1 is satisfied, the decomposition is nontrivial. □

Theorem 6.1 has a straightforward interpretation.

Since (π_I, π_S, π_O) is a partition trinity, based only on the information about its own input and present state (i.e. knowledge of the adequate block of π_I and block of π_S), machine M_1 is able to calculate its next state and output (i.e. the adequate blocks of π_S and π_O).

Since (τ_I, τ_S, τ_O) is a next-state-dependent partition trinity for $\xi_S = \pi_S$, based only on information about the block of τ_I containing the input, the block of τ_S containing the present state of M and the block of π_S containing the next state of M for the given input and present state (i.e. information about the primary input and present state of M_2 and the next state of M_1 which is part of the input of M_2), machine M_2 is able to calculate unambiguously the block of τ_S in which the next state of M is contained and the block of τ_O in which the output of M is contained for the given input and present state (i.e. M_2 is able to calculate its next state and output).

Since $\tau_O \cdot \pi_O = \pi_O(0)$, with information about blocks of π_O calculated by M_1 and blocks of τ_O calculated by M_2, it is possible to calculate unambiguously the outputs of machine M.

7. Serial full-decomposition of type PO.

Let $\pi_s^!$ and ξ_O be partitions on M on S and O respectively.

DEFINITION 7.1 $\pi_s^!$ is ***a state partition induced by an output partition*** ξ_O *if and only if* one of the following conditions is satisfied:

(i) $\forall s,t \in S\ \forall x,y \in I$: *if* $[s\lambda_x]\xi_O = [t\lambda_y]\xi_O$
then $[s\delta_x]\pi_s^! = [t\delta_y]\pi_s^!$
(for a Mealy machine),

(ii) $\forall s,t \in S$: $[s]\pi_s^! = [t]\pi_s^!$ *if and only if*
$[s\lambda]\xi_O = [t\lambda]\xi_O$
(for a Moore machine) .

In other words, if $\pi_s^!$ is a state partition induced by an output partition ξ_O and, if it is known that the present output y of M is contained in a block C: $C \in \xi_O$, then, it is known that the present state s of M is contained in a block B: $B \in \pi_s^!$, where block B is indicated unambiguously by block C. It can be said, that *block* B of $\pi_s^!$ *is induced by block* C of ξ_O and denoted by: B = ind(C).

Let τ_I, τ_S, τ_O be partitions on a machine M, on I, S and O respectively, and ξ_O be the other partition on O.

DEFINITION 7.2 (τ_I, τ_S, τ_O) is a ***partition trinity induced by an output partition*** ξ_O *if and only if* such a state partition $\pi_s^!$ induced by ξ_O exists, that τ_I, τ_S and τ_O satisfy the following conditions for this $\pi_s^!$:

(i) (τ_I, τ_S) is an I-S partition pair,

(ii) $(\tau_S \cdot \pi_S', \tau_S)$ is a S-S partition pair,

(iii) $(\tau_S \cdot \pi_S', \tau_O)$ is a S-O partition pair,
and
(τ_I, τ_O) is an I-O partition pair (for a Mealy machine),
or
(τ_S, τ_O) is a S-O partition pair (for a Moore machine).

In other words, (τ_I, τ_S, τ_O) is a trinity induced by an output partition ξ_O if and only if, based on the knowledge of the block of a partition τ_I containing the input of M and the knowledge of the block of a partition τ_S and the block of an induced partition $\pi_s^!$ containing the present state of M, it is possible to calculate the block of τ_S in which the next state of M will be contained. In the case of a Mealy machine, based on the same information it is possible to calculate the block of τ_0 in which the output of M will be contained for the given input and state. While, in the case of a Moore machine, based on the knowledge of the blocks of partitions τ_S and π_S' containing the state of M, it is possible to calculate the block of τ_0 containing the output of M for the given state.

<u>THEOREM</u> <u>7.1</u> A machine M has a nontrivial serial full-decomposition of type PO with the realization of the output behaviour *if* such a partition trinity (π_I,π_S,π_0) and such a partition trinity (τ_I,τ_S,τ_0) induced by $\xi_0 = \pi_0$ exist that the following conditions are satisfied:

(i) $\pi_0 \cdot \tau_0 = \pi_0(0)$,

(ii) $|\pi_I|<|I| \wedge |\pi_0|\cdot|\tau_I|<|I| \vee |\pi_S|<|S| \wedge |\tau_S|<|S| \vee |\pi_0|<|0| \wedge$
$\wedge |\tau_0|<|0|$.

<u>Proof</u> (for the case of a Mealy machine)

Let $M_1 = (\pi_I,\pi_S,\pi_0,\delta^1,\lambda^1)$ and $M_2 = (\pi_0 x \tau_I,\tau_S,\tau_0,\delta^2,\lambda^2)$ be the two machines for which the following conditions are satisfied:

(1) (π_I,π_S,π_0) and (τ_I,τ_S,τ_0) satisfy the conditions of the theorem 7.1 ,

(2) $\forall B1 \in \pi_S\ \forall A_1 \in \pi_I$: $B1\delta^1{}_{A1} = [B1\overline{\delta}_{A1}]\pi_S$, $B1\lambda^1{}_{A1} = [B1\overline{\lambda}_{A1}]\pi_0$,

(3) $\forall C1 \in \pi_0\ \forall B2 \in \tau_S\ \forall A2 \in \tau_I$:

$$B2\delta^2{}_{(C1,A2)}=[\{s\delta_x|\ s\in B2 \wedge s\in ind(C1) \wedge x\in A2\}]\tau_S,$$
$$B2\lambda^2{}_{(C1,A2)}=[\{s\lambda_x|\ s\in B2 \wedge s\in ind(C1) \wedge x\in A2\}]\tau_0.$$

Since (π_I,π_S,π_0) is a partition trinity (1), $B1\delta^1{}_{A1}$ and $B1\lambda^1{}_{A1}$ are defined unambiguously.

Since (τ_I,τ_S,τ_0) is a trinity induced by $\xi_0=\pi_0$ (1), the following conditions are satisfied:

(4) $(\tau_S\cdot\pi_S',\tau_S)$ is a S-S pair,

(5) $(\tau_S\cdot\pi_S',\tau_0)$ is a S-O pair,

(6) (τ_I,τ_S) is an I-S pair,

(7) (τ_I,τ_0) is an I-O pair.

From (4) and (6), it follows that $\{s\delta_x|\ s\in B2 \wedge s\in ind(C1) \wedge x\in A2\}$ is located in just one block of τ_S. From (5) and (7), it follows that $\{s\lambda_x|\ s\in B2 \wedge s\in ind(C1) \wedge x\in A2\}$ is located in just one block of τ_0. This means, that $B2\delta^2{}_{(C1,A2)}$ and $B2\lambda^2{}_{(C1,A2)}$ are defined unambigously

Let ψ: $I \longrightarrow \pi_I x \tau_I$ be an injective function,
ϕ: $S \longrightarrow \pi_S x \tau_S$ be an injective function,
θ: $\pi_O x \tau_O \longrightarrow O$ be a surjective partial function

and

(8) $\psi(x) = ([x]\pi_I, [x]\tau_I)$,

(9) $\phi(s) = ([s]\pi_S, [s]\tau_S)$,

(10) $\theta(C1, C2) = C1 \cap C2$ if $C1 \cap C2 \neq 0$.

It will be proved below that the serial connection of type PO of the machines M_1 and M_2 defined above realizes the output behaviour of machine M.

Since $\pi_O \cdot \tau_O = \pi_O(0)$ (1) , θ is a one-to-one function and for $C1 \cap C2 \neq 0$:

(11) $(C1, C2) \in O$.

Therefore, $\forall s \in S$ $\forall x \in I$:

$$\phi(s)\delta^*_{\psi(x)} =$$
$$= ([s]\pi_S, [s]\tau_S)\delta^*_{([x]\pi_I, [x]\tau_I)} \qquad ((8),\ (9))$$
$$= ([s]\pi_S\delta^1_{[x]\pi_I}, [s]\tau_S\delta^2_{([s]\pi_S', [x]\tau_I)}) \qquad \text{(definition 2.8)}$$
$$= ([[s]\pi_S\bar{\delta}_{[x]\pi_I}]\pi_S, [([s]\tau_S \cap [s]\pi_S')\bar{\delta}_{[x]\tau_I}]\tau_S) \qquad ((2),\ (3))$$
$$= ([s\delta_x]\pi_S, [s\delta_x]\tau_S) \qquad ((1))$$
$$= \phi(s\delta_x) \qquad ((5))$$

and similary:

$$\theta(\phi(s)\lambda^*_{\psi(x)}) =$$
$$= \theta(([s]\pi_S, [s]\tau_S)\lambda^*_{([x]\pi_I, [x]\tau_I)}) \qquad ((8),\ (9))$$
$$= \theta([s]\pi_S\lambda^1_{[x]\pi_I}, [s]\tau_S\lambda^2_{([s]\pi_S', [x]\tau_I)}) \qquad \text{(definition 2.8)}$$
$$= [s]\pi_S\lambda^1_{[x]\pi_I} \cap [s]\tau_S\lambda^2_{([s]\pi_S', [x]\tau_I)} \qquad ((10))$$
$$= [[s]\pi_S\bar{\lambda}_{[x]\pi_I}]\pi_O \cap [([s]\tau_S \cap [s]\pi_S')\bar{\lambda}_{[x]\tau_I}]\tau_O \qquad ((2),\ (3))$$
$$= [s\lambda_x]\pi_O \cap [s\lambda_x]\tau_O \qquad ((1))$$
$$= s\lambda_x \qquad (\ \pi_O \cdot \tau_O = \pi_O(0)\ )$$

From the above calculations and definitions 2.4, 2.8 and 2.12, it follows immediately that the serial connection of type PO of machines M_1 and M_2 realizes M, i.e. M has a serial full-decomposition of type PO with the output behaviour realization. If condition (ii) of theorem 5.1 is satisfied, the decomposition is nontrivial. □

The interpretation of theorem 7.1 is as follows:

Since (π_I, π_S, π_O) is a partition trinity, based only on the information about its own input and present state (i.e. knowledge of the adequate block of π_I and block of π_S), machine M_1 is able to calculate its next state and output (i.e. the appropriate blocks of π_S and π_O).

Since (τ_I, τ_S, τ_O) is a partition trinity induced by π_O, based only on the information about the block of a partition τ_I containing the input, the block of a partition τ_S containing the present state and the block of a partition π_O containing the output of machine M (i.e. information about the primary input and the present state of M_2 and about the present output of M_1 which is a part of the input of M_2), machine M_2 is able to calculate unambiguously the block of τ_S in which the next state of M will be contained. In the case of Mealy machine, based on the same information M_2 is able to calculate the block of τ_O in which the output of M will be contained for the given input and present state In the case of Moore machine, M_2 is able to calculate the block of τ_O in which the output of M will be contained using only information about the block of τ_S in which the state of M is contained. So, M_2 is able to calculate its next state and output.

Since $\pi_O \cdot \tau_O = \pi_O(0)$, with information about blocks of π_O calculated by M_1 and blocks of τ_O calculated by M_2 , it is possible to calculate unambiguously the outputs of machine M.

8. Serial full-decomposition of type NO.

Let τ_I, τ_S, τ_O be partitions on a machine M, on I, S, O respectiviely, and ξ_O be the other partition on O.

DEFINITION 8.1 (τ_I, τ_S, τ_O) is a (next) *output-dependent trinity* for the *independent output partition* ξ_O *if and only if* τ_I, τ_S and τ_O satisfy one of the following conditions for a given ξ_O:

(i) $\forall s,t \in S\ \forall x_1,x_2 \in I$:

if $[s]\tau_S=[t]\tau_S \wedge [x_1]\tau_I=[x_2]\tau_I \wedge [s\lambda_{x_1}]\xi_O=[t\lambda_{x_2}]\xi_O$

then $[s\delta_{x_1}]\tau_S=[t\delta_{x_2}]\tau_S \wedge [s\lambda_{x_1}]\tau_O=[t\lambda_{x_2}]\tau_O$

(for a Mealy machine),

(ii) $\forall s,t \in S\ \forall x_1,x_2 \in I$:

if $[s]\tau_S=[t]\tau_S \wedge [x_1]\tau_I=[x_2]\tau_I \wedge [(s\delta_{x_1})\lambda]\xi_O=[(t\delta_{x_2})\lambda]\xi_O$

then $[s\delta_{x_1}]\tau_S=[t\delta_{x_2}]\tau_S \wedge [(s\delta_{x_1})\lambda]\tau_O=[(t\delta_{x_2})\lambda]\tau_O$

(for a Moore machine).

In other words, (τ_I,τ_S,τ_O) is an output-dependent trinity for the independent output partition ξ_O if and only if, based on the knowledge of the block of a partition τ_I in which the input of a machine M is contained, the block of a partition τ_S in which the present state of M is contained and the block of a partition ξ_O in which the outputs of M are contained for inputs from a given block of τ_I and states from a given block of τ_S, it is possible to calculate the block of τ_S in which the next state of M is contained and the block of τ_O in which the output of M is contained for the present state from a given block of τ_S and input from a given block of τ_I.

<u>THEOREM</u> <u>8.1</u> A machine M has a nontrivial serial full-decomposition of type NO with the realization of the output behaviour *if* such a partition trinity (π_I,π_S,π_O) and such an output-dependent trinity (τ_I,τ_S,τ_O) for $\xi_O=\pi_O$ exist that the following conditions are satisfied:

(i) $\pi_O \cdot \tau_O = \pi_O(0)$,

(ii) $|\pi_I|<|I| \wedge |\pi_O|\cdot|\tau_I|<|I| \vee |\pi_S|<|S| \wedge |\tau_S|<|S| \vee |\pi_O|<|O| \wedge$
$\wedge |\tau_O|<|O|$.

<u>Proof</u> (for the case of Mealy machine)

Let $M_1 = (\pi_I,\pi_S,\pi_O,\delta^1,\lambda^1)$ and $M_2 = (\pi_O x \tau_I,\tau_S,\tau_O,\delta^2,\lambda^2)$ be two machines for which the following conditions are satisfied:

(1) (π_I,π_S,π_O) and (τ_I,τ_S,τ_O) satisfy the conditions of theorem 9.1 ,

(2) $\forall B1 \in \pi_S\ \forall A1 \in \pi_I$: $B1\delta^1{}_{A1} = [B1\bar{\delta}_{A1}]\pi_S \wedge B1\lambda^1{}_{A1} = [B1\bar{\lambda}_{A1}]\pi_O$,

(3) $\forall B2 \in \tau_S\ \forall A2 \in \tau_I\ \forall C1 \in \pi_O$:

$$B2\delta^2{}_{(C1,A2)} = [\{s\delta_x |\ s \in B2,\ x \in A2,\ s\lambda_x \in C1\ \}]\tau_S ,$$
$$B2\lambda^2{}_{(C1,A2)} = [\{s\lambda_x |\ s \in B2,\ x \in A2,\ s\lambda_x \in C1\ \}]\tau_O .$$

Since (π_I,π_S,π_O) is a partition trinity (1), $B1\bar{\delta}_{A1}$ is placed in just one block of π_S and $B1\bar{\lambda}_{A1}$ is placed in just one block of π_O.

This means that $B1\delta^1_{A1}$ and $B1\lambda^1_{A1}$ are unambiguously defined.

Since (τ_I, τ_S, τ_O) is an output dependent trinity for $\xi_O = \pi_O$ (1), the following condition is satisfied:

(4) $\forall s,t \in S\ \forall x_1, x_2 \in I$:

if $[s]\tau_S = [t]\tau_S \wedge [x_1]\tau_I = [x_2]\tau_I \wedge [s\lambda_{x_1}]\pi_O = [t\lambda_{x_2}]\pi_O$

then $[s\delta_{x_1}]\tau_S = [t\delta_{x_2}]\tau_S \wedge [s\lambda_{x_1}]\tau_O = [t\lambda_{x_2}]\tau_O$.

From (4), it follows that $B2\delta^2_{(C1,A2)}$ and $B2\lambda^2_{(C1,A2)}$ are defined unambiguously, because $\{s\delta_x \mid s \in B2,\ x \in A2,\ s\lambda_x \in C1\}$ is located in just one block of τ_S and $\{s\lambda_x \mid s \in B2,\ x \in A2,\ s\lambda_x \in C1\}$ is in just one block of τ_O.

Let $\psi: I \longrightarrow \pi_I x \tau_I$ be an injective function,

$\phi: S \longrightarrow \pi_S x \tau_S$ be an injective function,

$\theta: \pi_O x \tau_O \longrightarrow O$ be a surjective partial function

and

(5) $\psi(x) = ([x]\pi_I, [x]\tau_I)$,

(6) $\phi(s) = ([s]\pi_S, [s]\tau_S)$,

(7) $\theta(C1,C2) = C1 \cap C2$ if $C1 \cap C2 \neq 0$.

It will be proved below that the serial connection of type NS of the machines M_1 and M_2 defined above realizes the output behaviour of machine M.

Since $\pi_O \cdot \tau_O = \pi_O(0)$ (1) , θ is a one-to-one function and for $C1 \cap C2 \neq 0$:

(8) $(C1,C2) \in O$.

So, $\forall s \in S\ \forall x \in I$:

$\phi(s)\delta^*_{\psi(x)} =$

$= ([s]\pi_S, [s]\tau_S)\delta^*_{([x]\pi_I, [x]\tau_I)}$ ((5), (6))

$= ([s]\pi_S\delta^1_{[x]\pi_I}, [s]\tau_S\delta^2_{([s\lambda_x]\pi_O, [x]\tau_I)})$ (definition 2.9)

$= ([[s]\pi_S\overline{\delta}_{[x]\pi_I}]\pi_S, [\{s\delta_x \mid [s]\tau_S \wedge [s_{\lambda x}]\pi_O \wedge [x]\tau_I\}]\tau_S)$ ((2), (3))

$= ([s\delta_x]\pi_S, [s\delta_x]\tau_S)$ ((1))

$= \phi(s\delta_x)$ ((6))

and similary:

$\theta(\phi(s)\lambda^*_{\psi(x)}) =$

$= \theta(([s]\pi_S, [s]\tau_S)\lambda^*_{([x]\pi_I, [x]\tau_I)})$ ((5), (6))

$= \theta([s]\pi_S\lambda^1_{[x]\pi_I}, [s]\tau_S\lambda^2_{([s\lambda_x]\pi_O, [x]\tau_I)})$ (definition 2.9)

$$= [s]\pi_S \lambda^1{}_{[x]\pi_I} \cap [s]\tau_S \lambda^2{}_{([s\lambda_x]\pi_O,[x]\tau_I)} \qquad ((7))$$

$$= [[s]\pi_S \bar{\lambda}_{[x]\pi_I}]\pi_O \cap [\{s\lambda_x | \ [s]\tau_S \wedge [s\lambda_x]\pi_O \wedge [x]\tau_I\}]\tau_O \qquad ((2),\ (3))$$

$$= [s\lambda_x]\pi_O \cap [s\lambda_x]\tau_O \qquad ((1))$$

$$= s\lambda_x \qquad (\ \pi_O \cdot \tau_O = \pi_O(0)\ )$$

From the above calculations and definitions 2.4, 2.9 and 2.12, it follows that the serial connection of type NO of machines M_1 and M_2 realizes M, i.e. M has a serial full-decomposition of type NO with the output behaviour realization. If condition (ii) of theorem 8.1 is satisfied, the decomposition is nontrivial. □

Theorem 8.1 has the following interpretation:

Since (π_I, π_S, π_O) is a partition trinity, machine M_1, based only on the information about its input and present state (i.e. knowledge of the adequate block of π_I and block of π_S), is able to calculate its next state and output (i.e. the appropriate blocks of π_S and π_O).

Since (τ_I, τ_S, τ_O) is an output-dependent partition trinity for $\xi_O = \pi_O$, based only on information about the block of τ_I containing the input, the block of τ_S containing the present state of M and the block of π_O containing the output of M for the given input and present state (i.e. information about the primary input and present state of M_2 and the output of M_1 which is a part of the input of M_2), machine M_2 is able to calculate unambiguously the block of τ_S in which the next state of M is contained and the block of τ_O in which the output of M is contained for the given input and present state (i.e. M_2 is able to calculate its next state and output).

Since $\tau_O \cdot \pi_O = \pi_O(0)$, with information about blocks of π_O calculated by M_1 and blocks of τ_O calculated by M_2 , it is possible to calculate unambiguously the next states and outputs of machine M.

9. General full-decomposition of type PS

THEOREM 9.1 A machine M has a nontrivial general full-decomposition of type PS with the realization of the output behaviour *if* two present-state-dependent partition trinities: (π_I,π_S,π_O) and (τ_I,τ_S,τ_O) exist and they satisfy the following conditions:

(i) $\pi_O \cdot \tau_O = \pi_O(0)$,

(ii) $|\tau_S|\cdot|\pi_I|<|I| \wedge |\pi_S|\cdot|\tau_I|<|I| \vee |\pi_S|<|S| \wedge |\tau_S|<|S| \vee |\pi_O|<|O| \wedge$
$\wedge |\tau_O|<|O|$.

Proof (for the case of a Mealy machine)

Let $M_1 = (\tau_S x \pi_I, \pi_S, \pi_O, \delta^1, \lambda^1)$ and $M_2 = (\pi_S x \tau_I, \tau_S, \tau_O, \delta^2, \lambda^2)$ be the two machines for which the following conditions are satisfied:

(1) (π_I,π_S,π_O) and (τ_I,τ_S,τ_O) satisfy the conditions of theorem 9.1 ,

(2) $\forall B1\in\pi_S\ \forall B2\in\tau_S\ \forall A_1\in\pi_I$:
$B1\delta^1{}_{(B2,A1)}=[(B1\cap B2)\overline{\delta}_{A1}]\pi_S$, $B1\lambda^1{}_{(B2,A1)}=[(B1\cap B2)\overline{\lambda}_{A1}]\pi_O$,

(3) $\forall B1\in\pi_S\ \forall B2\in\tau_S\ \forall A2\in\tau_I$:
$B2\delta^2{}_{(B1,A2)}=[(B1\cap B2)\overline{\delta}_{A2}]\tau_S$, $B2\lambda^2{}_{(B1,A2)}=[(B1\cap B2)\overline{\lambda}_{A2}]\tau_O$.

Since (π_I,π_S,π_O) and (τ_I,τ_S,τ_O) are the present-state-dependent trinities (1), $(B1\cap B2)\overline{\delta}_{A1}$ is placed in just one block of π_S, $(B1\cap B2)$ is placed in just one block of π_O, $(B1\cap B2)\overline{\delta}_{A2}$ is placed in only one block of τ_S and $(B1\cap B2)\overline{\lambda}_{A2}$ is placed in only one block of τ_O . This means, that $B1\delta^1{}_{(B2,A1)}$, $B1\lambda^1{}_{(B2,A1)}$, $B2\delta^2{}_{(B1,A2)}$ and $B2\lambda^2{}_{(B1,A2)}$ are defined unambiguously.

Let ψ: $I \longrightarrow \pi_I x \tau_I$ be an injective function,
ϕ: $S \longrightarrow \pi_S x \tau_S$ be an injective function,
θ: $\pi_O x \tau_O \longrightarrow O$ be a surjective partial function
and

(4) $\psi(x) = ([x]\pi_I,[x]\tau_I)$,

(5) $\phi(s) = ([s]\pi_S,[s]\tau_S)$,

(6) $\theta(C1,C2) = C1\cap C2$ if $C1\cap C2 \neq 0$.

It will be proved below that the general connection of type PS of the machines M_1 and M_2 defined above realizes the output behaviour of machine M.

Since $\pi_0 \cdot \tau_0 = \pi_0(0)$ (1) , θ is a one-to-one function and for $C1 \cap C2 \neq 0$:

(7) $(C1,C2) \in O$.

Therefore, $\forall s \in S\ \forall x \in I$:

$$\phi(s)\delta^*_{\psi(x)} =$$

$$= ([s]\pi_S,[s]\tau_S)\delta^*_{([x]\pi_I,[x]\tau_I)} \qquad ((4),\ (5))$$

$$= ([s]\pi_S\delta^1_{([s]\tau_S,[x]\pi_I)},[s]\tau_S\delta^2_{([s]\pi_S,[x]\tau_I)}) \qquad \text{(definition 2.10)}$$

$$= ([([s]\pi_S \cap [s]\tau_S)\overline{\delta}_{[x]\pi_I}]\pi_S,[([s]\tau_S \cap [s]\pi_S)\overline{\delta}_{[x]\tau_I}]\tau_S) \qquad ((2),\ (3))$$

$$= ([s\delta_x]\pi_S,[s\delta_x]\tau_S) \qquad ((1))$$

$$= \phi(s\delta_x) \qquad ((5))$$

and similary:

$$\theta(\phi(s)\lambda^*_{\psi(x)}) =$$

$$= \theta(([s]\pi_S,[s]\tau_S)\lambda^*_{([x]\pi_I,[x]\tau_I)}) \qquad ((4),\ (5))$$

$$= \theta([s]\pi_S\lambda^1_{([s]\tau_S,[x]\pi_I)},[s]\tau_S\lambda^2_{([s]\pi_S,[x]\tau_I)}) \qquad \text{(definition 2.10)}$$

$$= [s]\pi_S\lambda^1_{([s]\tau_S,[x]\pi_I)} \cap [s]\tau_S\lambda^2_{([s]\pi_S,[x]\tau_I)} \qquad ((6))$$

$$= [([s]\pi_S \cap [s]\tau_S)\overline{\lambda}_{[x]\pi_I}]\pi_0 \cap [([s]\tau_S \cap [s]\pi_S)\overline{\lambda}_{[x]\tau_I}]\tau_0 \qquad ((2),\ (3))$$

$$= [s\lambda_x]\pi_0 \cap [s\lambda_x]\tau_0 \qquad ((1))$$

$$= s\lambda_x \qquad (\ \pi_0 \cdot \tau_0 = \pi_0(0)\ )$$

From the above calculations and definitions 2.4, 2.10 and 2.12, it follows that the general connection of type PS of machines M_1 and M_2 realizes M, i.e. M has a general full-decomposition of type PS with the output behaviour realization. If condition (ii) of theorem 9.1 is satisfied, the decomposition is nontrivial. □

The interpretation of theorem 9.1 is similar to the interpretation of theorem 5.1.

10. General full-decomposition of type PO

THEOREM 10.1 A machine M has a nontrivial general full-decomposition of type PO with the realization of the output behaviour *if* two partition trinities (π_I,π_S,π_O) induced by $\xi_{O2} = \tau_O$ and (τ_I,τ_S,τ_O) induced by $\xi_{O1} = \pi_O$ exist and they satisfy the following conditions:

(i) $\pi_O \cdot \tau_O = \pi_O(0)$,

(ii) $|\tau_O|\cdot|\pi_I|<|I| \; |\pi_O|\cdot|\tau_I|<|I| \vee |\pi_S|<|S| \wedge |\tau_S|<|S| \vee |\pi_O|<|O| \wedge$
$\wedge |\tau_O|<|O|$.

Proof (for the case of a Mealy machine)

Let $M_1 = (\tau_O x\pi_I,\pi_S,\pi_O,\delta^1,\lambda^1)$ and $M_2 = (\pi_O x\tau_I,\tau_S,\tau_O,\delta^2,\lambda^2)$ be the two machines for which the following conditions are satisfied:

(1) (π_I,π_S,π_O) and (τ_I,τ_S,τ_O) satisfy the conditions of theorem 10.1 ,

(2) $\forall C2\epsilon\tau_O \; \forall B1\epsilon\pi_S \; \forall A_1\epsilon\pi_I$:

$$B1\delta^1_{(C2,A1)} = [\{s\delta_x| \; s\epsilon B1 \wedge s\epsilon ind(C2) \wedge x\epsilon A1]\pi_S ,$$
$$B1\lambda^1_{(C2,A1)} = [\{s\lambda_x| \; s\epsilon B1 \wedge s\epsilon ind(C2) \wedge x\epsilon A1]\pi_O ,$$

(3) $\forall C1\epsilon\pi_O \; \forall B2\epsilon\tau_S \; \forall A2\epsilon\tau_I$:

$$B2\delta^2_{(C1,A2)} = [\{s\delta_x| \; s\epsilon B2 \wedge s\epsilon ind(C1) \wedge x\epsilon A2\}]\tau_S,$$
$$B2\lambda^2_{(C1,A2)} = [\{s\lambda_x| \; s\epsilon B2 \wedge s\epsilon ind(C1) \wedge x\epsilon A2\}]\tau_O.$$

Since (π_I,π_S,π_O) is a partition trinity induced by $\xi_{O2}=\tau_O$ and (τ_I,τ_S,τ_O) is a partition trinity induced by $\xi_{O1}=\pi_O$ (1), the following conditions are satisfied:

(4) $(\pi_S'\cdot\tau_S,\tau_S)$ is a S-S pair,

(5) $(\pi_S\cdot\tau_S',\pi_S)$ is a S-S pair,

(6) $(\pi_S'\cdot\tau_S,\tau_O)$ is a S-O pair,

(7) $(\pi_S\cdot\tau_S',\pi_O)$ is a S-O pair,

(8) (π_I,π_S) is an I-S pair,

(9) (π_I,π_O) is an I-O pair,

(10) (τ_I,τ_S) is an I-S pair,

(11) (τ_I,τ_O) is an I-O pair.

From (5) and (8), it follows that $\{s\delta_x| \; s\epsilon B1 \wedge s\epsilon ind(C2) \wedge x\epsilon A1\}$ is located in just one block of π_S. From (7) and (9), it follows that $\{s\lambda_x | s\in B1 \wedge s\in ind(C2) \wedge x\in A1\}$ is located in only one block of π_0. This means, that $B1\delta^1_{(C2,A1)}$ and $B1\lambda^1_{(C2,A1)}$ are unambiguously defined.

Similarly, from (4) and (10), it follows that $\{s\delta_x | s\in B2 \wedge s\in ind(C1) \wedge x\in A2\}$ is located in just one block of τ_S and, from (6) and (11), it follows that $\{s\lambda_x | s\in B2 \wedge s\in ind(C1) \wedge x\in A2\}$ is located in just one block of τ_0. So, $B2\delta^2_{(C1,A2)}$ and $B2\lambda^2_{(C1,A2)}$ are unambigously defined.

Let $\psi\colon I \longrightarrow \pi_I x \tau_I$ be an injective function,
$\phi\colon S \longrightarrow \pi_S x \tau_S$ be an injective function,
$\theta\colon \pi_0 x \tau_0 \longrightarrow O$ be a surjective partial function
and

(12) $\psi(x) = ([x]\pi_I, [x]\tau_I)$,

(13) $\phi(s) = ([s]\pi_S, [s]\tau_S)$,

(14) $\theta(C1,C2) = C1 \cap C2$ if $C1 \cap C2 \neq 0$.

It will be proved below that the general connection of type PO of the machines M_1 and M_2 defined above realizes the output behaviour of machine M.

Since $\pi_0 \cdot \tau_0 = \pi_0(0)$ (1) , θ is a one-to-one function and for $C1 \cap C2 \neq 0$:

(11) $(C1,C2) \in O$.

Therefore, $\forall s\in S\ \forall x\in I$:

$$\phi(s)\delta^*_{\psi(x)} =$$

$$= ([s]\pi_S,[s]\tau_S)\delta^*_{([x]\pi_I,[x]\tau_I)} \qquad ((12),\ (13))$$

$$= ([s]\pi_S\delta^1_{([s]\tau_S^{:},[x]\pi_I)}, [s]\tau_S\delta^2_{([s]\pi_S^{:},[x]\tau_I)})$$

(definition 2.11)

$$= ([([s]\pi_S \cap [s]\tau_S')\overline{\delta}_{[x]\pi_I}]\pi_S, [([s]\tau_S \cap [s]\pi_S')\overline{\delta}_{[x]\tau_I}]\tau_S)$$

((2), (3))

$$= ([s\delta_x]\pi_S, [s\delta_x]\tau_S) \qquad ((1))$$

$$= \phi(s\delta_x) \qquad ((13))$$

and similary:

$$\theta(\phi(s)\lambda^*_{\psi(x)}) =$$

$$= \theta(([s]\pi_S,[s]\tau_S)\lambda^*_{([x]\pi_I,[x]\tau_I)}) \qquad ((12),\ (13))$$

$$= \theta([s]\pi_S\lambda^1_{([s]\tau_S^{:},[x]\pi_I)}, [s]\tau_S\lambda^2_{([s]\pi_S^{:},[x]\tau_I)})$$

(definition 2.11)

$$= [s]\pi_S\lambda^1_{([s]\tau_S^{:},[x]\pi_I)} \cap [s]\tau_S\lambda^2_{([s]\pi_S^{:},[x]\tau_I)} \qquad ((14))$$

$= [([s]\pi_s \cap [s]\tau_s')\overline{\lambda}_{[x]\pi_I}]\pi_0 \cap [([s]\tau_s \cap [s]\pi_s')\overline{\lambda}_{[x]\tau_I}]\tau_0$ ((2), (3))

$= [s\lambda_x]\pi_0 \cap [s\lambda_x]\tau_0$ ((1))

$= s\lambda_x$ ($\pi_0 \cdot \tau_0 = \pi_0(0)$)

From the above calculations and definitions 2.4, 2.11 and 2.12, it follows immediately that the general connection of type PO of machines M_1 and M_2 realizes M, i.e. M has a general full-decomposition of type PO with the output behaviour realization. If condition (ii) of theorem 10.1 is satisfied, the decomposition is nontrivial. □

The interpretation of theorem 10.1 is similar to the interpretation of theorem 7.1.

11. Conclusion.

The notions and theorems presented in the previous sections have straightforward practical interpretations and they constitute the theoretical basis for practical algorithms and for a system of programs for computing the different sorts of decompositions. These algorithms and some practical conclusions will be presented in a separate report.

The results presented in this report can be extended easily in order to cover the case of incompletely specified sequential machines. This can be done by using the concepts of the weak partition pairs or extended partition pairs introduced by Hartmanis [12].

From Chapter 2, it follows that a full-decomposition with the state and output behaviour realization is such a special case of the full-decomposition with the output behaviour realization that the partial machines M_1 and M_2 imitate a given machine M not only from the input-output point of view but also from the input-state point of view. It is easy to observe that if the condition: $\pi_s \cdot \tau_s = \pi_s(0)$ is added to the assumptions of the theorems formulated in this work, the theorems proved in [16] are obtained concerning the existence of full-decompositions with the state and output behaviour realization. So, the theorems proved in [16] are special cases of the appriopriate theorems proved here for $\pi_S \cdot \tau_S = \pi_S(0)$.

Similarly, considering a state machine M = (I, S, δ) to be a Moore machine M'= (I, S, O, δ, λ) where O = S and λ is an identity function or a Mealy machine M" = (I, S, O, δ, λ) where O = S and λ = δ, the appriopriate theorems 12.1 - 12.4 from [16] concerning the existence of full-decompositions for state machines can be obtained directly from the theorems 4.1, 5.1, 6.1 and 10.1 proved in this work.

In some practical cases, it is more economical to consider separately the realization of the next-state function δ and the output function λ rather than to consider them simultaneously. It is possible to abstract from the output function λ and to decompose first the state machine defined by the next-state function δ. Then, it is passible to realize the output function λ, where λ is treated as a function of inputs (in the Mealy case) and states of the partial state machines obtained in a full-decomposition of the state machine defined by δ.

From the practical point of view, full-decompositions of type N are not so attractive as decompositions of type P, because decompositions of type N introduce timing problems. In decompositions of type N, one of the component machines has to be able to compute its next state or output, before the second component machine, using the information about the computed next state or output of the first submachine, can compute its own next state or output. If it is assumed that computing the next-state and output for one component machine requires one time interval, a valid next-state and output for the whole machine will appear after two such time intervals. In this situation, the frequency of input signals need to be limited and a two-phase clock is required.

Solving the practical cases starts with trying to find a parallel full-decomposition which satisfies the given requirements and, only in the case of failure, is there need to look for a serial decomposition or, in the case of failure, for a general decomposition. In the case of the serial and general decompositions, the connections between the partial machines have to be implemented and the functional dependences between the input, state and output variables of the partial machines are in most cases decrising from a parallel through serial to a general decomposition, i.e. the complexity of the combinational logic of each of the component machines is usually least for parallel decompositions and greatest for general decompositions.

The practical decomposition algorithms should implement some optimization criteria. The full-decomposition of sequential machines can be a tool for making it possible to implement the machine with existing building blocks , to design, implement and verify the machine more easily or to optimize the separate submachines, whereas, it may be impossible or very difficult to optimize the whole machine directly. However, it may be a suitable optimization tool itself.

REFERENCES

[1] M.A. Arbib : Theories of abstract automata, Englewood Cliffs,N.J.: Prentice - Hall, 1969.

[2] G. Cioffi, E. Constantini, S. de Julio : A new approach to the decomposition of sequential systems, Digital Processes, vol.3, p. 35-48, 1977.

[3] G. Cioffi, S. de Julio, M. Lucertini : Optimal decomposition of sequential machines via integer nonlinear programming: A computational algorithm, Digital Processes, vol.5, p. 27-41, 1979.

[4] A.D. Friedman, P.R. Menon : Theory and design of switching circuits, Woodland Hills, Cal.: Computer Science Press, 1975.

[5] A. Ginzburg : Algebraic theory of automata, N.Y.: Academic Press, 1968.

[6] J. Hartmanis : On the state assignment problem for sequential machines I, IRE Trans. Electron. Comput., vol.EC-10, p.157-165, 1961.

[7] J. Hartmanis, R.E. Stearns : On the state assignment problem for sequential machines II, IRE Trans. Electron. Comput., vol.EC-10, p.593-603, 1961.

[8] J. Hartmanis : Loop-free structure of sequential machines, Inf. & Control, vol.5, p.25-43, 1962.

[9] J. Hartmanis : Further results on the structure of sequential machines, J. Assoc. Comput. Mach., vol.10,p.78-88, 1963.

[10] J. Hartmanis, R.E. Stearns : Some danger in state reduction of sequential machines, Inf. & Control, vol.5, p.252-260, 1962.

[11] J. Hartmanis, R.E. Stearns : Pair algebra and its application to automata theory, Inf. & Control, vol.7, p.485-507, 1964.

[12] J. Hartmanis, R.E. Stearns : Algebraic structure theory of sequential machines, Englewood Cliffs, N.J.: Prentice-Hall, 1966.

[13] W.M.L. Holcombe : Algebraic Automata Theory, Cambridge University Press, 1982. (Cambridge studies in advanced mathematics, vol.1).

[14] Y. Hou : Trinity algebra and full-decompositions of sequential machines, Ph.D. thesis, Eindhoven University of Technology, The Netherlands, 1986.

[15] Y. Hou : Trinity algebra and its application to machine decompositions, Information Processing Letters, vol.26, p.127-134, 1987.

[16] L. Jóźwiak : The full decomposition of sequential machines with the state and output behaviour realization, Eindhoven University of Technology Research Reports, Eindhoven University of Technology, The Netherlands, January 1988. EUT Report 88-E-188.

[17] Yu.V. Pottosin, E.A. Shestakov : Approximate algorithms for parallel decomposition of automata, Autom. Contr. & Comput. Sci., vol.15, No 2, p.24-31, 1981. (Translation of: Avtom. & Vytchisl. Techn.).

[18] Yu.V. Pottosin, E.A. Shestakov E.A. : Decomposition of an automaton into a two-component network with constraints on internal connections, Autom. Contr. & Comput. Sci., vol.16, No 6, p.24-31, 1982.

[19] Yu.V. Pottosin : Decompositional method for coding the states of a parallel automaton, Autom. Contr. & Comput. Sci., vol.21, No 1, p.78-84, 1987.

[20] M. Yoeli : The cascade decomposition of sequential machines, IRE Trans. Electron. Comput., vol.EC-10, p.587-592, 1961.

[21] M. Yoeli : Cascade-parallel decompositions of sequential machines, IEEE Trans. Electron. Comput., vol.EC-12, p.322-324, 1963.

ISSN 0167-9708
Coden: TEUEDE

(171) Monnee, P. and M.H.A.J. Herben
MULTIPLE-BEAM GROUNDSTATION REFLECTOR ANTENNA SYSTEM: A preliminary study.
EUT Report 87-E-171. 1987. ISBN 90-6144-171-4

(172) Bastiaans, M.J. and A.H.M. Akkermans
ERROR REDUCTION IN TWO-DIMENSIONAL PULSE-AREA MODULATION, WITH APPLICATION TO COMPUTER-GENERATED TRANSPARENCIES.
EUT Report 87-E-172. 1987. ISBN 90-6144-172-2

(173) Zhu Yu-Cai
ON A BOUND OF THE MODELLING ERRORS OF BLACK-BOX TRANSFER FUNCTION ESTIMATES.
EUT Report 87-E-173. 1987. ISBN 90-6144-173-0

(174) Berkelaar, M.R.C.M. and J.F.M. Theeuwen
TECHNOLOGY MAPPING FROM BOOLEAN EXPRESSIONS TO STANDARD CELLS.
EUT Report 87-E-174. 1987. ISBN 90-6144-174-9

(175) Janssen, P.H.M.
FURTHER RESULTS ON THE McMILLAN DEGREE AND THE KRONECKER INDICES OF ARMA MODELS.
EUT Report 87-E-175. 1987. ISBN 90-6144-175-7

(176) Janssen, P.H.M. and P. Stoica, T. Söderström, P. Eykhoff
MODEL STRUCTURE SELECTION FOR MULTIVARIABLE SYSTEMS BY CROSS-VALIDATION METHODS.
EUT Report 87-E-176. 1987. ISBN 90-6144-176-5

(177) Stefanov, B. and A. Veefkind, L. Zarkova
ARCS IN CESIUM SEEDED NOBLE GASES RESULTING FROM A MAGNETICALLY INDUCED ELECTRIC FIELD.
EUT Report 87-E-177. 1987. ISBN 90-6144-177-3

(178) Janssen, P.H.M. and P. Stoica
ON THE EXPECTATION OF THE PRODUCT OF FOUR MATRIX-VALUED GAUSSIAN RANDOM VARIABLES.
EUT Report 87-E-178. 1987. ISBN 90-6144-178-1

(179) Lieshout, G.J.P. van and L.P.P.P. van Ginneken
GM: A gate matrix layout generator.
EUT Report 87-E-179. 1987. ISBN 90-6144-179-X

(180) Ginneken, L.P.P.P. van
GRIDLESS ROUTING FOR GENERALIZED CELL ASSEMBLIES: Report and user manual.
EUT Report 87-E-180. 1987. ISBN 90-6144-180-3

(181) Bollen, M.H.J. and P.T.M. Vaessen
FREQUENCY SPECTRA FOR ADMITTANCE AND VOLTAGE TRANSFERS MEASURED ON A THREE-PHASE POWER TRANSFORMER.
EUT Report 87-E-181. 1987. ISBN 90-6144-181-1

(182) Zhu Yu-Cai
BLACK-BOX IDENTIFICATION OF MIMO TRANSFER FUNCTIONS: Asymptotic properties of prediction error models.
EUT Report 87-E-182. 1987. ISBN 90-6144-182-X

(183) Zhu Yu-Cai
ON THE BOUNDS OF THE MODELLING ERRORS OF BLACK-BOX MIMO TRANSFER FUNCTION ESTIMATES.
EUT Report 87-E-183. 1987. ISBN 90-6144-183-8

(184) Kadete, H.
ENHANCEMENT OF HEAT TRANSFER BY CORONA WIND.
EUT Report 87-E-184. 1987. ISBN 90-6144-6

(185) Hermans, P.A.M. and A.M.J. Kwaks, I.V. Bruza, J. Dijk
THE IMPACT OF TELECOMMUNICATION ON RURAL AREAS IN DEVELOPING COUNTRIES.
EUT Report 87-E-185. 1987. ISBN 90-6144-185-4

(186) Fu Yanhong
THE INFLUENECE OF CONTACT SURFACE MICROSTRUCTURE ON VACUUM ARC STABILITY AND ARC VOLTAGE.
EUT Report 87-E-186. 1987. ISBN 90-6144-186-2

(187) Kaiser, F. and L. Stok, R. van den Born
DESIGN AND IMPLEMENTATION OF A MODULE LIBRARY TO SUPPORT THE STRUCTURAL SYNTHESIS.
EUT Report 87-E-187. 1987. ISBN 90-6144-187-0

ISSN 0167-9708
Coden: TEUEDE

(188) Jóźwiak, J.
THE FULL DECOMPOSITION OF SEQUENTIAL MACHINES WITH THE STATE AND OUTPUT BEHAVIOUR REALIZATION.
EUT Report 88-E-188. 1988. ISBN 90-6144-188-9

(189) Pineda de Gyvez, J.
ALWAYS: A system for wafer yield analysis.
EUT Report 88-E-189. 1988. ISBN 90-6144-189-7

(190) Siuzdak, J.
OPTICAL COUPLERS FOR COHERENT OPTICAL PHASE DIVERSITY SYSTEMS.
EUT Report 88-E-190. 1988. ISBN 90-6144-190-0

(191) Bastiaans, M.J.
LOCAL-FREQUENCY DESCRIPTION OF OPTICAL SIGNALS AND SYSTEMS.
EUT Report 88-E-191. 1988. ISBN 90-6144-191-9

(192) Worm, S.C.J.
A MULTI-FREQUENCY ANTENNA SYSTEM FOR PROPAGATION EXPERIMENTS WITH THE OLYMPUS SATELLITE.
EUT Report 88-E-192. 1988. ISBN 90-6144-192-7

(193) Kersten, W.F.J. and G.A.P. Jacobs
ANALOG AND DIGITAL SIMULATION OF LINE-ENERGIZING OVERVOLTAGES AND COMPARISON WITH MEASUREMENTS IN A 400 kV NETWORK.
EUT Report 88-E-193. 1988. ISBN 90-6144-193-5

(194) Hosselet, L.M.L.F.
MARTINUS VAN MARUM: A Dutch scientist in a revolutionary time.
EUT Report 88-E-194. 1988. ISBN 90-6144-194-3

(195) Bondarev, V.N.
ON SYSTEM IDENTIFICATION USING PULSE-FREQUENCY MODULATED SIGNALS.
EUT Report 88-E-195. 1988. ISBN 90-6144-195-1

(196) Liu Wen-Jiang, Zhu Yu-Cai and Cai Da-Wei
MODEL BUILDING FOR AN INGOT HEATING PROCESS: Physical modelling approach and identification approach.
EUT Report 88-E-196. 1988. ISBN 90-6144-196-X