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
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
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.
CONTENTS

1. Introduction................................................................. 2
2. Full-decompositions and their sorts................................. 4
3. Partitions, partition pairs and partition trinities............14
4. Parallel full-decomposition...........................................17
5. Serial full-decomposition of type PS................................19
6. Serial full-decomposition of type NS.......................22
7. Serial full-decomposition of type PO............................26
8. Serial full-decomposition of type NO.................29
9. General full-decomposition of type PS.................33
10. General full-decomposition of type PO................35
11. Conclusion.................................................................37

References.................................................................40
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 following 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 submachines, 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 today's 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).

![Diagram](image)

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

The control units of large digital systems can engage 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 needed for implementing a sequential machine is a very complicated process which can be carried 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-flops,
- 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 following 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 optimizing of the separate submachines, 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,
- \( \delta \) - the next state function, \( \delta : S \times I \rightarrow S \),
- \( \lambda \) - the output function, \( \lambda : S \times I \rightarrow O \) (a Mealy machine),
  or \( \lambda : S \rightarrow O \) (a Moore machine).

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

The machine functions \( \delta \) and \( \lambda \) can be considered to be sets of functions created for each input:

\[ \delta = \{ \delta_x | \delta_x : S \rightarrow S \text{ and } x \in I \} \]

and

\[ \lambda = \{ \lambda_x | \lambda_x : S \rightarrow O \text{ and } x \in I \} \]

where \( \delta_x : S \rightarrow S \) and \( \lambda_x : S \rightarrow O \) are defined by:

\[ \forall x \in I \ \forall s \in S \ \delta_x(s) = \delta(s, x) \]

\[ \lambda_x(s) = \lambda(s, x). \]

\( \delta_x \) and \( \lambda_x \), 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:

\[ \overline{\delta}_x : 2^S \rightarrow 2^S \text{ and } \overline{\lambda}_x : 2^S \rightarrow 2^O \]

are defined, where:

\[ \forall x \in I \ \forall Q \subseteq S \ \overline{\delta}_x = \{ s \delta_x | s \in Q \}, \overline{\lambda}_x = \{ s \lambda_x | s \in Q \}. \]
For \( X \in I \) and \( Q \subseteq S \) the following two partial functions are also defined:

\[ \bar{t}_X : 2^S \rightarrow 2^I \text{ and } \bar{\lambda}_X : 2^S \rightarrow 2^O, \]

where:

\[ Q_{\bar{t}_X} = \{ s_{\bar{t}_X} \mid s \in Q \land x \in X \}, \]
\[ Q_{\bar{\lambda}_X} = \{ s_{\bar{\lambda}_X} \mid s \in Q \land x \in X \}. \]

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.

**Definition 2.2** 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:

\( \phi : I \rightarrow I' \) (a function),
\( \phi : S \rightarrow 2^I \) (a function into nonvoid subsets of \( S' \)),
\( \theta : O' \rightarrow O \) (a surjective partial function),

and these relations satisfy the following conditions:

\[ \phi(s \delta' \phi(x)) \subseteq \phi(s_{\bar{t}_X}) \]

and

\[ s_{\bar{\lambda}_X} = \theta(s \lambda' \phi(x)) \] (for a Mealy machine)

or

\[ s_{\bar{\lambda}} = \theta(s' \lambda') \] (for a Moore machine)

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

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

- \( \hat{s} : S \times I^* \to S, \quad \hat{s}(s, x) = \hat{s}(s, x'), x \)
- \( \hat{l} : S \times I^* \to O, \quad \hat{l}(s, x) = l(\hat{s}(s, x'), x) \) (Mealy case),
- \( \hat{l}(s, x) = l(\hat{s}(s, x)) \) (Moore case).

It can be proved that if \( M' \) is a realization of \( M \) in the sense of definition 2.1 then \( \forall s \in S \forall x \in I^* : \hat{l}(s, x) = \hat{l}(\hat{s}(s, x'), x) \), 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:

**DEFINITION 2.3** Machine \( M' = (I', S', O', \delta', \lambda') \), realizes the state and output behaviour of machine \( M = (I, S, O, \delta, \lambda) \) if and only if the following relations exist:

- \( \phi : I \to I' \) (a function),
- \( \phi : S' \to S \) (a surjective partial function),
- \( \theta : O' \to O \) (a surjective partial function)

such that:

\[ \phi(s') \delta_x = \phi(s' \delta_x \phi(x)) \]

and

\[ \phi(s') \lambda_x = \theta(s' \lambda_x \phi(x)) \] (for a Mealy machine)

or

\[ \phi(s') \lambda = \theta(s' \lambda') \] (for a Moore machine).

The realization of state and output behaviour is a special case of the realization of output behaviour. If function \( \phi \) in definition 2.2 maps each state of \( M \) onto a single state of \( M' \) and \( \phi \) 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 \( \phi \) maps each state of \( M \) onto a single state of \( M' \), i.e. \( \phi : S \to S' \).
DEFINITION 2.4 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:

- \( \psi: I \rightarrow I' \) (a function),
- \( \phi: S \rightarrow S' \) (a function),
- \( \theta: O' \rightarrow O \) (a surjective partial function),

and this relations satisfy the following conditions:

\[
\psi(s)\delta'(x) = \phi(s\delta_x)
\]

and

\[
s\lambda_x = \theta(\phi(s)\lambda'(x)) \quad \text{(for a Mealy machine)}
\]

or

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

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:

- \( \psi: I \rightarrow I_1 \times I_2 \),
- \( \phi: S \rightarrow S_1 \times S_2 \),
- \( \theta: O_1 \times O_2 \rightarrow O \),

that the machines \( M_1 \) and \( M_2 \) together with the mappings \( \psi, \phi, \theta \) realize the behaviour of a machine \( M \).

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

\[
|I_1| < |I| \land |I_2| < |I| \lor |S_1| < |S| \land |S_2| < |S| \lor |O_1| < |O| \land |O_2| < |O|, \text{ where } |Z| \text{ - number of elements in the set } Z.
\]

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: $PS, PO$.

![Diagram](image)

**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 $P^S$ and $P^O$ 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 $s \in S_1$, $t \in S_2$, $x_1 \in I_1$, $x_2 \in I_2$.

**Definition 2.5** 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:
\[ M_1 \| M_2 = (I_1xI_2, S_1xS_2, O_1xO_2, \delta^*, \lambda^*) \]
where:
\[ \delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1), \delta^2(t,x_2)) \]
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)

**Definition 2.6** A serial connection of type $P^S$ 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 $I_2' = S_1xI_2$,
is the machine
\[ M_1 \rightarrow^S M_2 = (I_1xI_2, S_1xS_2, O_1xO_2, \delta^*, \lambda^*) \]
where:
\[ \delta^*((s,t),(x_1,x_2)) = (\delta^1(s,x_1), \delta^2(t,(s,x_2))) \]
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 \( M_1 \rightarrow M_2 = (I_1 x I_2', S_1 x S_2, O_1 x O_2, \delta^*, \lambda^*) \),
where:
\[ \delta^*((s, t), (x_1, x_2)) = (\delta^1(s, x_1), \delta^2(t, (\delta^1(s, x_1), x_2))) \]
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 \( M_1 \rightarrow M_2 = (I_1 x I_2', S_1 x S_2, O_1 x O_2, \delta^*, \lambda^*) \),
where:
\[ \delta^*((s, t), (x_1, x_2)) = (\delta^1(s, x_1), \delta^2(t, (y_1, x_2))) \]
and
\[ \lambda^*((s, t), (x_1, x_2)) = (\lambda^1(s, x_1), \lambda^2(t, (y_1, x_2))) \]
and \( y_1 \in O_1 : y_1 \) 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 \( I_1' = O_1 \times I_2 \)
is the machine \( M_1 \xrightarrow{\delta^0} M_2 = (I_1' \times I_2, S_1 \times S_2, O_1 \times O_2, \delta^*, \lambda^*), \)
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:
\[ I_1' = S_2 \times I_1 \]
\[ I_2' = S_1 \times I_2 \]
is the machine:
\[ M_1 \xrightarrow{\delta} M_2 = (I_1' \times I_2, S_1 \times S_2, O_1 \times O_2, \delta^*, \lambda^*), \]
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:
\[ I_1' = O_2 \times I_1 \]
\[ I_2' = O_1 \times I_2 \]
is the machine:
\[ M_1 \xrightarrow{\delta} M_2 = (I_1' \times I_2, S_1 \times S_2, O_1 \times O_2, \delta^*, \lambda^*), \]
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)

**DEFINITION 2.12** The machine \( M_1 \bowtie M_2 \) is a full decomposition of type \( \bowtie \) of machine \( M \) if and only if the connection of a given type \( \bowtie \) of machines \( M_1 \) and \( M_2 \) realizes \( M \), where:
\[
\bowtie = | |, \rightarrow , \rightarrow , \rightarrow , \rightarrow , \leftrightarrow , \leftrightarrow .
\]

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.

3. Partitions, partition pairs and partition trinities.

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.

**DEFINITION 3.1** Partition \( \pi \) on \( S \) is defined as follows:
\[
\pi = \{ B_i | B_i \subseteq S \text{ and } B_i \cap B_j = 0 \text{ for } i \neq j \text{ and } \bigcup_i B_i = S \},
\]
i.e. a partition \( \pi \) 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 \( \pi \) 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 \( \pi \). Similarly, the block of a partition \( \pi \) containing \( S' \), where \( S' \subset S \), 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 \( \pi_s(1) \).

Let \( \pi_1 \) and \( \pi_2 \) 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 \([s]\pi_2 = [t]\pi_2 \).

**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 \( i = 1, \ldots, n \), exists for which either \([s_i]\pi_1 = [s_i+1]\pi_1 \) and either \([s_i]\pi_2 = [s_i+1]\pi_2 \), \( 0 \leq i \leq n-1 \).

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

**DEFINITION 3.4** \( \pi_2 \) is greater than or equal to \( \pi_1 \): \( \pi_1 \leq \pi_2 \) if and only if each block of \( \pi_1 \) is included in a block of \( \pi_2 \).

Thus \( \pi_1 \leq \pi_2 \) if and only if \( \pi_1 \cdot \pi_2 = \pi_1 \) if and only if \( \pi_1 + \pi_2 = \pi_2 \).

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

Let \( (Z, \preceq) \) 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) \( \forall t \in T : z \geq t \),
(ii) \( \forall t \in T : \text{if } z' \geq t \text{ then } z' \geq z \).

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

(i) \( \forall t \in T : z \leq t \),
(ii) \( \forall t \in T : \text{if } z' \leq t \text{ then } z' \leq z \).

**DEFINITION 3.6** A partially ordered set \( L = (Z, \preceq) \), 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 $\leq$ form a lattice with 
$\text{GLB}(\pi_1, \pi_2) = \pi_1 \cap \pi_2$ and $\text{LUB}(\pi_1, \pi_2) = \pi_1 + \pi_2$.

Let $\pi_s, \tau_s, \pi_I, \pi_0$ be the partitions on $M=(I, S, O, \delta, \lambda)$, in 
particular: $\pi_s, \tau_s$ on $S$, $\pi_I$ on $I$, $\pi_0$ on $O$.

**DEFINITION 3.7**

(i) $(\pi_s, \tau_s)$ is an $S-S$ partition pair if and only if 
$\forall B \in \pi_s \ \forall x \in I : \overline{B} \subseteq B', \ B' \in \tau_s$.

(ii) $(\pi_I, \pi_s)$ is an $I-S$ partition pair if and only if 
$\forall A \in \pi_I \ \forall x \in S : \overline{s} \subseteq B, \ B \in \pi_s$.

(iii) $(\pi_s, \pi_0)$ is an $S-O$ partition pair if and only if 
$\forall B \in \pi_s \ \forall x \in I : B \subseteq C, \ C \in \pi_0$ (Mealy case) 
or
$\forall B \in \pi_s : B \subseteq C, \ C \in \pi_0$ (Moore case).

(iv) $(\pi_I, \pi_0)$ is an $I-O$ partition pair if and only if 
$\forall A \in \pi_I \ \forall s \in S : \overline{s} \subseteq C, \ C \in \pi_0$ (Mealy case) 
or
$\forall A \in \pi_I \ \forall s \in S : s \subseteq C, \ C \in \pi_0$ (Moore case).

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

$(\pi_s, \tau_s)$ is an $S-S$ partition pair if and only if the blocks of $\pi_s$ 
are mapped by $M$ into the blocks of $\tau_s$. Thus, if the block of $\pi_s$ 
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 $\tau_s$ which contains the next state of $M$ for the states 
from a given block of $\pi_s$ 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 $(\pi_I, \pi_s)$ satisfies it (the output of $M$ is 
defined by the state of $M$ unambiguously).

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

**DEFINITION 3.9** Partition trinity $T=(\pi_I, \pi_s, \pi_0)$ 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:

$\forall A \in \pi_I \ \forall B \in \pi_s : \overline{B} \subseteq B', \ B' \in \pi_s$ and $\overline{B} \subseteq C, \ C \in \pi_0$. 
Thus, if \((\pi_I, \pi_S, \pi_0)\) is a partition trinity on \(M\) and the block \(B\) of \(\pi_s\) which contains the present state of \(M\) is known and the block \(A\) of \(\pi_I\) which contains the present input of \(M\) is known too, it is possible to compute unambiguously block \(B'\) of \(\pi_s\) that contains the next state of \(M\) and block \(C\) of \(\pi_0\) that contains the output of \(M\) for the states from block \(B\) and inputs from block \(A\).

For completely specifed machines, it has been proved that \((\pi_I, \pi_S, \pi_0)\) is a partition trinity on \(M\) if and only if \((\pi_s, \pi_s)\) is an \(S-S\) pair, \((\pi_I, \pi_s)\) is an \(I-S\) pair, \((\pi_s, \pi_0)\) is an \(S-O\) pair and \((\pi_I, \pi_0)\) is an \(I-O\) pair on \(M\) [14][15].

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

\[
\text{GLB}(T_1, T_2) = T_1 \circ T_2 \quad \text{and} \quad \text{LUB}(T_1, T_2) = T_1 \circ T_2,
\]

where \(\circ\) and \(\circ\) are defined as a collection of pairwise operations "\(*\)" and "\(\circ\)" for partitions of the same type (input, state, output) of trinities of \(T_1\) and \(T_2\).

4. Parallel full-decomposition.

**Theorem 4.1** 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 \(M\): \((\pi_I, \pi_S, \pi_0)\) and \((\tau_I, \tau_S, \tau_0)\) exist and they satisfy the following conditions:

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

**Proof** (for the case of a Mealy machine)

Let \(M_1 = (\pi_I, \pi_S, \pi_0, \delta^1, \lambda^1)\) and \(M_2 = (\tau_I, \tau_S, \tau_0, \delta^2, \lambda^2)\) be two sequential machines satisfying the following conditions:

(1) \((\pi_I, \pi_S, \pi_0)\) and \((\tau_I, \tau_S, \tau_0)\) satisfy the conditions of theorem 4.1,
(2) \(\forall B I \in \pi_S \forall A I \in \pi_I: B \delta^1 A_1 = [B \delta^1 A_1] \pi_S\),
\(B \lambda^1 A_1 = [B \lambda^1 A_1] \pi_I\),
(3) \(\forall B I \in \tau_S \forall A I \in \tau_I: B \delta^2 A_2 = [B \delta^2 A_2] \tau_S\),
\(B \lambda^2 A_2 = [B \lambda^2 A_2] \tau_I\).

Since \((\pi_I, \pi_S, \pi_0)\) is a partition trinity (1), \(B \delta^1 A_1\) is placed in just one block of \(\pi_S\) and \(B \lambda^1 A_1\) in only one block of \(\pi_0\). This means, that \(B \delta^1 A_1\) and \(B \lambda^1 A_1\) are defined unambiguously. Similarly, since \((\tau_I, \tau_S, \tau_0)\) is a partition trinity (1), \(B \delta^2 A_2\) and \(B \lambda^2 A_2\)
are defined unambiguously. So, each of the partial machines $M_1$ and $M_2$ can calculate its next states and outputs unambiguously.

Let $\psi: I \rightarrow \pi_I \times \tau_I$ be an injective function,
\[ \phi: S \rightarrow \pi_S \times \tau_S \] be an injective function,
\[ \theta: \pi_0 \times \tau_0 \rightarrow O \] be a surjective partial function

and

\[ \psi(x) = ([x] \pi_I, [x] \tau_I) \] \hspace{1cm} (4)
\[ \phi(s) = ([s] \pi_S, [s] \tau_S) \] \hspace{1cm} (5)
\[ \theta(C_1, C_2) = C_1 \cap C_2 \text{ if } C_1 \cap C_2 \neq \emptyset . \] \hspace{1cm} (6)

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), $\theta$ is a one-to-one function and for $C_1 \cap C_2 \neq \emptyset$:

\[ (C_1, C_2) \in O . \] \hspace{1cm} (7)

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

\[ \phi(s) \delta^x \psi(x) = \]
\[ = ([s] \pi_S, [s] \tau_S) \delta^x ([x] \pi_I, [x] \tau_I) \] \hspace{1cm} (1), (2), (3)
\[ = ([s] \pi_S \delta^x [x] \pi_I, [s] \tau_S \delta^x [x] \tau_I) \] \hspace{1cm} (4), (5)
\[ = ([[[s] \pi_S \delta^x [x] \pi_I] \pi_S, [s] \tau_S \delta^x [x] \tau_I] \tau_S) \] \hspace{1cm} (2), (3)
\[ = ([s\delta_x] \pi_S, [s\delta_x] \tau_S) \] \hspace{1cm} (6)
\[ = \phi(s\delta_x) \] \hspace{1cm} (5)

and similarly:

\[ \theta(\phi(s) \delta^x \psi(x)) = \]
\[ = \theta(([s] \pi_S, [s] \tau_S) \delta^x ([x] \pi_I, [x] \tau_I)) \] \hspace{1cm} (4), (5)
\[ = \theta([s] \pi_S \delta^x [x] \pi_I, [s] \tau_S \delta^x [x] \tau_I) \] \hspace{1cm} (4), (5)
\[ = ([s] \pi_S \delta^x [x] \pi_I \cap [s] \tau_S \delta^x [x] \tau_I) \] \hspace{1cm} (6)
\[ = ([s] \pi_S \delta^x [x] \pi_I \pi_0 \cap ([s] \pi_S \delta^x [x] \pi_I \tau_0 \tau_0) \] \hspace{1cm} (2), (3)
\[ = [s\delta_x] \pi_0 \cap [s\delta_x] \tau_0 \] \hspace{1cm} (1)
\[ = s\delta_x \] \hspace{1cm} ($\pi_0 \cdot \tau_0 = \pi_0(0)$)

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 \((\pi_I, \pi_S, \pi_O)\) is a partition trinity, based only on the information about the block of \(\pi_I\) containing the input of \(M\) and the block of \(\pi_S\) 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 \(\pi_S\) in which the next state of \(M\) is contained, as well as, the block of \(\pi_O\) that contains the output of \(M\) for the input from a given block of \(\pi_I\) and the present state from a given block of \(\pi_S\) (i.e. \(M_1\) can calculate its next state and output). Similarly, since \((\tau_I, \tau_S, \tau_O)\) 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 \(\tau_I\) and block of \(\tau_S\)), can calculate its next state and output (i.e. the adequate blocks of \(\tau_S\) and \(\tau_O\)).

Since \(\tau_O \cdot \tau_I = \pi_O(0)\), the knowledge of of the block of \(\pi_O\) and the block of \(\tau_I\) 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:

\[
\begin{align*}
|\pi_I| &< |I| \land |\pi_S| = |S| \land |\pi_O| = |O| \lor |\tau_S| = |S| \land |\tau_O| = |O|
\end{align*}
\]

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 \(\tau_I, \tau_S, \tau_O\) be partitions on a machine \(M\) on \(I, S\) and \(O\) respectively.

**DEFINITION 5.1** \((\tau_I, \tau_S, \tau_O)\) is a present-state-dependent trinity for an independent state partition \(\xi\) if and only if \(\tau_I, \tau_S\) and \(\tau_O\) satisfy the following conditions:

(i) \((\tau_I, \tau_S)\) 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

\((\tau_I, \tau_O)\) is an I-O partition pair (for a Mealy machine),
or
\((\tau_S, \tau_O)\) is a S-O partition pair (for a Moore machine)
In other words, \((T_1, T_\delta, T_0)\) is a present-state-dependent trinity if and only if, based only on the knowledge of the block of a partition \(T_1\) containing the input of \(M\) and the knowledge of the blocks of partitions \(T_\delta\) and \(T_0\) containing the present state of \(M\), it is possible to calculate the block of \(T_\delta\) 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 \(T_0\) 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 \(T_1\) in which the state of \(M\) is contained, it is possible to calculate the block of \(T_0\) in which the output of \(M\) will be contained for the state from a given block of \(T_\delta\).

**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 \((\pi_1, \pi_\delta, \pi_0)\) and a present-state-dependent partition trinity \((T_1, T_\delta, T_0)\) for \(\pi_\delta = \pi_\delta\) exist and they satisfy the following conditions:

\(\begin{align*}
(i) & \quad \pi_0 \cdot T_0 = \pi_0(0), \\
(ii) & \quad |T_1| < |I| \wedge |T_\delta| < |S| \vee |T_0| < |O| \wedge \\
& \quad \wedge |T_0| < |O|.
\end{align*}\)

**Proof** (for the case of a Mealy machine)

Let \(M_1 = (\pi_1, \pi_\delta, \pi_0, \delta^1, \lambda^1)\) and \(M_2 = (\pi_\delta \times T_1, T_\delta, T_0, \delta^2, \lambda^2)\) be two machines that satisfy the following conditions:

(1) \((\pi_1, \pi_\delta, \pi_0)\) and \((T_1, T_\delta, T_0)\) satisfy the conditions of the theorem 6.1,

(2) \(\forall B \in \pi_\delta \forall A \in \pi_1 : B \delta^1 A_1 = [B \lambda^1 A_1] \pi_\delta\), \(B \lambda^1 A_1 = [B \lambda^1 A_1] \pi_0\),

(3) \(\forall B \in \pi_\delta \forall B_2 \in T_\delta \forall A_2 \in T_1 : \\
B \delta^2 (B_1, A_2) = [(B_1 \pi_\delta B_2) \lambda A_2] T_\delta, \ B \lambda^2 (B_1, A_2) = [(B_1 \pi_\delta B_2) \lambda A_2] T_0\).

Since \((\pi_1, \pi_\delta, \pi_0)\) is a partition trinity (1), \(B \lambda A_1\) is placed in just one block of \(\pi_\delta\) and \(B \lambda A_1\) in only one block of \(\pi_0\). This means, that \(B \delta A_1\) and \(B \lambda A_1\) are defined unambiguously.

Since \((T_1, T_\delta, T_0)\) is a present-state-dependent trinity (1), \(B \lambda A_2\) is placed in just one block of \(T_\delta\) and \(B \lambda A_2\) is placed in only one block of \(T_0\). This means, that \(B \delta A_2\) and \(B \lambda A_2\) are defined unambiguously.
Let \( \psi : I \rightarrow \pi_I \times \tau_I \) be an injective function, \\
\( \phi : S \rightarrow \pi_S \times \tau_S \) be an injective function, \\
\( \theta : \pi_0 \times \tau_0 \rightarrow O \) be a surjective partial function \\
and \\
(4) \quad \psi(x) = ([x]\pi_I, [x]\tau_I), \\
(5) \quad \phi(s) = ([s]\pi_S, [s]\tau_S), \\
(6) \quad \theta(C_1, C_2) = \pi_1 \cap \pi_2 \text{ if } C_1 \cap C_2 \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), \( \theta \) is a one-to-one function and for 
\( C_1 \cap C_2 \neq 0 \):

(7) \quad (C_1, C_2) \in O.

Therefore, \( \forall s \in S \forall x \in I \):

\[
\phi(s) \delta^* \psi(x) = \\
= ([s]\pi_S, [s]\tau_S) \delta^* ([x] \pi_I, [x] \tau_I) \quad \text{ (4), (5)} \\
= ([s]\pi_S \delta^* [x] \pi_I, [s]\tau_S \delta^* [x] \tau_I) \quad \text{ (definition 2.6)} \\
= ([s] \pi_S \delta^* [x] \pi_I, [s] \tau_S \delta^* [x] \tau_I) \quad \text{ (2), (3)} \\
= ([s] \pi_S \delta^* [x] \pi_I, [s] \tau_S \delta^* [x] \tau_I) \quad \text{ (4), (5)}
\]

and similarly:

\[
\theta(\phi(s) \lambda^* \psi(x)) = \\
= \theta(([s]\pi_S, [s]\tau_S) \lambda^* ([x] \pi_I, [x] \tau_I)) \quad \text{ (4), (5)} \\
= \theta([s]\pi_S \lambda^* [x] \pi_I, [s]\tau_S \lambda^* [x] \tau_I) \quad \text{ (definition 2.6)} \\
= [s]\pi_S \lambda^* [x] \pi_I \cap [s]\tau_S \lambda^* [x] \tau_I \quad \text{ (6)} \\
= ([s]\pi_S \lambda^* [x] \pi_I) \pi_0 \cap ([s]\tau_S \lambda^* [x] \tau_I) \tau_0 \quad \text{ (2), (3)} \\
= [s] \lambda_x \pi_0 \cap [s] \lambda_x \tau_0 \quad \text{ (1)} \\
= s \lambda_x \pi_0 \quad (\pi_0 \cdot \tau_0 = \pi_0(0))
\]

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. \( \square \)
Theorem 5.1 has a straightforward interpretation.

Since \((\pi_I, \pi_s, \pi_0)\) is a partition trinity, based only on the information about the block of a partition \(\pi_I\) containing the input and the block of a partition \(\pi_s\) 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 \(\pi_s\) in which the next state of \(M\) is contained and the block of \(\pi_0\) 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 \((\tau_I, \tau_s, \tau_0)\) is a present-state-dependent trinity, based only on the information about the block of a partition \(\tau_I\) containing the input and the blocks of partitions \(\tau_s\) and \(\tau_0\) 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 \(\tau_s\) in which the next state of \(M\) is contained and, in the case of a Mealy machine, the block of \(\tau_0\) 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 \(\tau_0\) in which the output of \(M\) is contained, based only on information about the block of \(\tau_s\) in which the state of \(M\) is contained.

Since \(\pi_0 \cdot \tau_0 = \pi_0(0)\), with information about the blocks of \(\pi_0\) calculated by \(M_1\) and the blocks of \(\tau_0\) 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 \(\tau_I\), \(\tau_s\), \(\tau_0\) be partitions on machine \(M\), on \(I\), \(S\) and \(O\) respectively, and \(\xi_s\) be another partition on \(S\).

**DEFINITION 6.1** \((\tau_I, \tau_s, \tau_0)\) is a next-state-dependent trinity for an independent state partition \(\xi_s\) if and only if \(\tau_I\), \(\tau_s\), \(\tau_0\) satisfy one of the following conditions for a given \(\xi_s\):

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

If \([s]_{\tau_s} = [t]_{\tau_s} \land [x_1]_{\tau_I} = [x_2]_{\tau_I} \land [s \delta_{x_1}]_{\xi_s} = [t \delta_{x_2}]_{\xi_s}\)
then \([s \delta_{x_1}]_{\tau_s} = [t \delta_{x_2}]_{\tau_s} \land [s \lambda_{x_1}]_{\tau_0} = [t \lambda_{x_2}]_{\tau_0}\)
(for a Mealy machine),
(ii) \( \forall s,t \in S \ \forall x_1, x_2 \in I: \)
if \( [s] \tau_s = [t] \tau_s \land [x_1] \tau_I = [x_2] \tau_I \land [s \delta x_1] \xi_s = [t \delta x_2] \xi_s \)
then \( [s \delta x_1] \tau_s = [t \delta x_2] \tau_s \land [(s \delta x_1) \lambda] \tau_0 = [(t \delta x_2) \lambda] \tau_0 \)
(for a Moore machine).

In other words, \((\tau_I, \tau_s, \tau_0)\) is a next-state-dependent trinity for an independent state partition \(\xi_s\) if and only if, based only on the knowledge of the block of a partition \(\tau_I\) containing the input of machine \(M\), knowledge of the block of a partition \(\tau_s\) containing the present state of \(M\) and knowledge of the block of a partition \(\xi_s\) in which the next state of \(M\) is contained for a given input and state, it is possible to calculate the block of \(\tau_s\) in which the next state of \(M\) will be contained and the block of \(\tau_0\) in which the output of \(M\) will be contained.

**THEOREM 6.1** A machine \(M\) has a nontrivial serial full-decomposition of type NS with the realization of the output behaviour if such a partition trinity \((\pi_I, \pi_s, \pi_0)\) and such a next-state-dependent trinity \((\tau_I, \tau_s, \tau_0)\) for \(\xi_s = \pi_s\) exist that the following conditions are satisfied:
(i) \(\pi_s \cdot \tau_s = \pi_s(0)\) and \(\pi_0 \cdot \tau_0 = \pi_0(0)\),
(ii) \(|\pi_I| < |I|\), \(|\pi_s| < |S|\), \(|\pi_0| < |O|\), \(|\pi_s| \cdot |\tau_I| < |I|\), \(|\tau_s| < |S|\), \(|\tau_0| < |O|\).

**Proof** (for the case of a Mealy machine)

Let \(M_1 = (\pi_I, \pi_s, \pi_0, \delta^1, \lambda^1)\) and \(M_2 = (\pi_s x \tau_I, \tau_s, \tau_0, \delta^2, \lambda^2)\) be two machines for which the following conditions are satisfied:
(1) \((\pi_I, \pi_s, \pi_0)\) and \((\tau_I, \tau_s, \tau_0)\) satisfy the conditions of the theorem 6.1,
(2) \(\forall B1 \in \pi_s \ \forall A1 \in \pi_I: B\lambda^1_{A1} = [B\delta^1_{A1}] \pi_s, B\lambda^1_{A1} = [B\lambda^1_{A1}] \pi_0\),
(3) \(\forall B2 \in \pi_s \ \forall A2 \in \tau_I \ \forall B1' \in \pi_s:
\quad B2\lambda^2_{(B1'), \lambda^2_2} = [(S \delta x) s e B2, x e A2, s \delta x e B1'] \tau_s,\)
\quad B2\lambda^2_{(B1'), \lambda^2_2} = [(S \delta x) s e B2, x e A2, s \delta x e B1'] \tau_0 .

Since \((\pi_I, \pi_s, \pi_0)\) is a partition trinity (1), \(B\delta^1_{A1}\) is placed in just one block of \(\pi_s\) and \(B\lambda^1_{A1}\) is placed in only one block of \(\pi_0\). This means that \(B\lambda^1_{A1}\) and \(B\lambda^1_{A1}\) are defined unambiguously.
Since $(\tau_I, \tau_S, \tau_0)$ is a next-state-dependent trinity for $\xi=\pi$, (1), the following condition is satisfied:

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

if $[s]\tau_S=[t]\tau_S \land [x_1]\tau_I=[x_2]\tau_I \land [s\delta x_1]\pi_S=[t\delta x_2]\pi_S$

then $[s\delta x_1]\tau_S=[t\delta x_2]\tau_S \land [s\lambda x_1]\tau_0=[t\lambda x_2]\tau_0$.

From (4), it follows that $B2\delta^2_{(B_1', A_2)}$ and $B2\lambda^2_{(B_1', A_2)}$ are defined unambiguously because $(s\delta x | s \in B_2, x \in A_2, s\delta x \in B_1')$ is located in only one block of $\tau_S$ and $(s\lambda x | s \in B_2, x \in A_2, s\lambda x \in B_1')$ is in just one block of $\tau_0$.

Let $\psi: I \rightarrow \pi_I x \tau_I$ be an injective function,
$\phi: S \rightarrow \pi_S x \tau_S$ be an injective function,
$\theta: \pi_0 x \tau_0 \rightarrow 0$ 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) $\theta(C_1, C_2) = C_1 \cap C_2$ if $C_1 \cap C_2 \neq 0$.

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 $\pi_0 \cdot \tau_0 = \pi_0 (0)$ (1), $\theta$ is a one-to-one function and for $C_1 \cap C_2 \neq 0$:

(8) $(C_1, C_2) \in 0$.

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

$\phi(s) \delta^* \psi(x) =
= ([s]\pi_S, [s]\tau_S) \delta^* ([x]\pi_I, [x]\tau_I) \quad (5), (6))$

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

$= ([s\delta x] \pi_S, [s\delta x] \tau_S) \quad (2), (3))$

$= \phi(s\delta x) \quad (6))$

and similarly:

$\theta(\phi(s) \lambda^* \psi(x)) =
= \theta([s\lambda x] \pi_S, [s\lambda x] \tau_S) \lambda^* ([x]\pi_I, [x]\tau_I) \quad (5), (6))$

$= \theta([s\lambda x] \lambda^1 (x]\pi_I), [s] \tau_S \lambda^2 ([s\delta x] \pi_S, [x]\tau_I)) \quad \text{(definition 2.7)}$
\[ = \left[ \mathcal{S} \right] \pi_1 \lambda_1 \{ x \} \pi_1 \cap \left[ \mathcal{S} \right] \tau_0 \lambda_1 \{ x \} \pi_2 \{ x \} \tau_1 \]  \hspace{1cm} ((7))
\[ = \left[ \left[ \left[ \mathcal{S} \right] \pi_1 \lambda_1 \{ x \} \pi_1 \right] \right] \pi_0 \cap \left[ \left[ \mathcal{S} \lambda_1 \right] \left[ \mathcal{S} \pi_1 \lambda_1 \pi_2 \pi_1 \lambda_1 \pi_1 \pi_2 \right] \right] \tau_0 \]  \hspace{1cm} ((2), (3))
\[ = \left[ \mathcal{S} \lambda_1 \right] \pi_0 \cap \left[ \left[ \mathcal{S} \lambda_1 \right] \right] \tau_0 \]  \hspace{1cm} ((1))
\[ = \mathcal{S} \lambda_1 \]  \hspace{1cm} (\pi_0 \cdot \tau_0 = \pi_0(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 \( (\pi_1, \pi_2, \pi_0) \) is a partition trinity, based only on the information about its own input and present state (i.e. knowledge of the adequate block of \( \pi_1 \) and block of \( \pi_2 \)), machine \( M_1 \) is able to calculate its next state and output (i.e. the adequate blocks of \( \pi_1 \) and \( \pi_0 \)).

Since \( (\tau_1, \tau_2, \tau_0) \) is a next-state-dependent partition trinity for \( \xi_2 = \pi_2 \), based only on information about the block of \( \tau_1 \) containing the input, the block of \( \tau_2 \) containing the present state of \( M \) and the block of \( \pi_2 \) 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 \( \tau_2 \) in which the next state of \( M \) is contained and the block of \( \tau_0 \) 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 \( \tau_0 \cdot \pi_0 = \pi_0(0) \), with information about blocks of \( \pi_0 \) calculated by \( M_1 \) and blocks of \( \tau_0 \) calculated by \( M_2 \), it is possible to calculate unambiguously the outputs of machine \( M \).
7. **Serial full-decomposition of type PO.**

Let $\pi_1'$ and $\xi_0$ be partitions on $M$ on $S$ and $O$ respectively.

**DEFINITION 7.1** $\pi_1'$ is a state partition induced by an output partition $\xi_0$ if and only if one of the following conditions is satisfied:

(i) $\forall s,t \in S \forall x,y \in I : [s \delta x] \xi_0 = [t \delta y] \xi_0$
then $[s \delta x] \pi_1' = [t \delta y] \pi_1'$
(for a Mealy machine),

(ii) $\forall s,t \in S : [s] \pi_1' = [t] \pi_1'$ if and only if
$[s \delta x] \xi_0 = [t \delta y] \xi_0$
(for a Moore machine).

In other words, if $\pi_1'$ is a state partition induced by an output partition $\xi_0$ and, if it is known that the present output $y$ of $M$ is contained in a block $C: C \epsilon \xi_0$, then, it is known that the present state $s$ of $M$ is contained in a block $B: B \epsilon \pi_1'$, where block $B$ is indicated unambiguously by block $C$. It can be said, that block $B$ of $\pi_1'$ is induced by block $C$ of $\xi_0$ and denoted by: $B = \text{ind}(C)$.

Let $\tau_I$, $\tau_s$, $\tau_0$ be partitions on a machine $M$, on $I$, $S$ and $O$ respectively, and $\xi_0$ be the other partition on $O$.

**DEFINITION 7.2** $(\tau_I, \tau_s, \tau_0)$ is a partition trinity induced by an output partition $\xi_0$ if and only if such a state partition $\pi_1'$ induced by $\xi_0$ exists, that $\tau_I$, $\tau_s$, and $\tau_0$ satisfy the following conditions for this $\pi_1'$:

(i) $(\tau_I, \tau_s)$ is an $I$-$S$ partition pair,
(ii) $(\tau_s, \tau_s')$, $\tau_s$) is a $S$-$S$ partition pair,
(iii) $(\tau_s, \tau_s')$, $\tau_0$) is a $S$-$O$ partition pair,
and
$(\tau_I, \tau_0)$ is an $I$-$O$ partition pair (for a Mealy machine),
or
$(\tau_s, \tau_0)$ is a $S$-$O$ partition pair (for a Moore machine).

In other words, $(\tau_I, \tau_s, \tau_0)$ is a trinity induced by an output partition $\xi_0$ if and only if, based on the knowledge of the block of a partition $\tau_I$ containing the input of $M$ and the knowledge of the block of a partition $\tau_s$ and the block of an induced partition $\pi_1'$ containing the present state of $M$, it is possible to calculate the
block of \( \tau_s \) 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 \( \tau_0 \) 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 \( \tau_s \) and \( \pi_s' \) containing the state of \( M \), it is possible to calculate the block of \( \tau_0 \) containing the output of \( M \) for the given state.

**Theorem 7.1** A machine \( M \) has a nontrivial serial full-decomposition of type \( \Phi_0 \) with the realization of the output behaviour if such a partition trinity \( (\pi_1, \pi_s, \pi_0) \) and such a partition trinity \( (\tau_1, \tau_s, \tau_0) \) induced by \( \xi_0 = \pi_0 \) exist that the following conditions are satisfied:

(i) \( \pi_0 \cdot \tau_0 = \pi_0(0) \),
(ii) \( |\pi_1| < I \land \pi_0 \cdot |\tau_1| < I \lor |\pi_s| < S \land |\tau_s| < S \lor |\pi_0| < 0 \land \land |\tau_0| < 0 \).

**Proof** (for the case of a Mealy machine)

Let \( M_1 = (\pi_1, \pi_s, \pi_0, \delta_1, \lambda_1) \) and \( M_2 = (\pi_0 \cdot \tau_1, \tau_s, \tau_0, \delta_2, \lambda_2) \) be the two machines for which the following conditions are satisfied:

(1) \( (\pi_1, \pi_s, \pi_0) \) and \( (\tau_1, \tau_s, \tau_0) \) satisfy the conditions of the theorem 7.1,
(2) \( \forall B \in \pi_s \ \forall A \in \pi_1 : B \delta_1 A_1 = [B \delta A_1] \pi_s \), \( B \lambda_1 A_1 = [B \lambda A_1] \pi_0 \),
(3) \( \forall C \in \pi_0 \ \forall B \in \pi_s \ \forall A \in \pi_1 : B \delta^2 (C_1, A_2) = [(s \delta x | s \in B \land s \in \text{ind}(C_1) \land x \in A_2)] \tau_s \),
\( B \lambda^2 (C_1, A_2) = [(s \lambda x | s \in B \land s \in \text{ind}(C_1) \land x \in A_2)] \tau_0 \).

Since \( (\pi_1, \pi_s, \pi_0) \) is a partition trinity (1), \( B \delta^1 A_1 \) and \( B \lambda^1 A_1 \) are defined unambiguously.

Since \( (\tau_1, \tau_s, \tau_0) \) 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) \( (\tau_1, \tau_s) \) is an I-S pair,
(7) \( (\tau_1, \tau_0) \) is an I-O pair.

From (4) and (6), it follows that \( (s x | s \in B \land s \in \text{ind}(C_1) \land x \in A_2) \) is located in just one block of \( \tau_s \). From (5) and (7), it follows that \( (s x | s \in B \land s \in \text{ind}(C_1) \land x \in A_2) \) is located in just one block of \( \tau_0 \). This means, that \( B \delta^2 (C_1, A_2) \) and \( B \lambda^2 (C_1, A_2) \) are defined unambiguously.
Let \( \psi : I \rightarrow \pi_1 \times \pi_1 \) be an injective function, 
\( \phi : S \rightarrow \pi_s \times \pi_s \) be an injective function, 
and
\( \theta : \pi_0 \times \pi_0 \rightarrow O \) be a surjective partial function

and

\[
(8) \quad \psi(x) = ([x] \pi_1, [x] \pi_1),
\]
\[
(9) \quad \phi(s) = ([s] \pi_s, [s] \pi_s),
\]
\[
(10) \quad \theta(C_1, C_2) = C_1 \cap C_2 \text{ if } C_1 \cap C_2 \neq 0.
\]

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 \( \pi_0 \cdot \tau_0 = \pi_0(0) \quad (1) \), \( \theta \) is a one-to-one function and for
\( C_1 \cap C_2 \neq 0 \) :
\[
(11) \quad (C_1, C_2) \circ 0.
\]

Therefore, \( \forall s \in S \forall x \in I : \)
\[
\psi(s) \delta^* \psi(x) =
\]
\[
= ([s] \pi_s, [s] \pi_s) \delta^* ([x] \pi_1, [x] \pi_1) \quad ((8), (9))
\]
\[
= ([s] \pi_s \delta^1 [x] \pi_1, [s] \pi_s \delta^2 ([s] \pi_s \delta^1 [x] \pi_1)) \quad (\text{definition 2.8})
\]
\[
= ([s] \pi_s \delta^1 [x] \pi_1 \pi_s, ([s] \pi_s \delta^1 [x] \pi_1 \pi_s) \delta^1 [x] \pi_1) \tau_s \quad ((2), (3))
\]
\[
= ([s] \pi_s, [s] \pi_s) \tau_s \quad ((1))
\]
\[
= \phi(s \delta^1 x) \quad ((5))
\]
and similarly:
\[
\theta(\phi(s) \delta^* \psi(x)) =
\]
\[
= \theta(([s] \pi_s, [s] \pi_s) \delta^* ([x] \pi_1, [x] \pi_1)) \quad ((8), (9))
\]
\[
= \theta([s] \pi_s \delta^1 [x] \pi_1 \pi_s, [s] \pi_s \delta^2 ([s] \pi_s \delta^1 [x] \pi_1 \pi_s)) \quad (\text{definition 2.8})
\]
\[
= [s] \pi_s \delta^1 [x] \pi_1 \pi_s \cap [s] \pi_s \delta^2 ([s] \pi_s \delta^1 [x] \pi_1 \pi_s) \quad ((10))
\]
\[
= ([s] \pi_s \delta^1 [x] \pi_1 \pi_s) \pi_0 \cap ([s] \pi_s \delta^1 [x] \pi_1 \pi_s) \pi_0 \quad ((2), (3))
\]
\[
= [s \delta^1 x] \pi_0 \cap [s \delta^1 x] \pi_0 \quad ((1))
\]
\[
= s \delta^1 x \quad (\pi_0 \cdot \tau_0 = \pi_0(0))
\]

From the above calculations and definitions 2.4, 2.8 and 2.12,
It follows immediately that the serial connection of type PO of
the 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. \( \Box \)
The interpretation of theorem 7.1 is as follows:
Since \((\pi_I, \pi_S, \pi_0)\) is a partition trinity, based only on the information about its own input and present state (i.e. knowledge of the adequate block of \(\pi_I\) and block of \(\pi_S\)), machine \(M_1\) is able to calculate its next state and output (i.e. the appropriate blocks of \(\pi_S\) and \(\pi_0\)).

Since \((\tau_I, \tau_S, \tau_0)\) is a partition trinity induced by \(\tau_0\), based only on the information about the block of a partition \(\tau_I\) containing the input, the block of a partition \(\tau_S\) containing the present state and the block of a partition \(\tau_0\) 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 \(\tau_S\) 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 \(\tau_0\) 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 \(\tau_0\) in which the output of \(M\) will be contained using only information about the block of \(\tau_S\) in which the state of \(M\) is contained. So, \(M_2\) is able to calculate its next state and output.

Since \(\pi_0 \cdot \tau_0 = \pi_0(0)\), with information about blocks of \(\tau_0\) calculated by \(M_1\) and blocks of \(\tau_0\) calculated by \(M_2\), it is possible to calculate unambiguously the outputs of machine \(M\).

8. Serial full-decomposition of type NO.

Let \(\tau_I, \tau_S, \tau_0\) be partitions on a machine \(M\), on \(I, S, 0\) respectively, and \(\xi_0\) be the other partition on \(O\).

**DEFINITION 8.1** \((\tau_I, \tau_S, \tau_0)\) is a (next) output-dependent trinity for the independent output partition \(\xi_0\) if and only if \(\tau_I, \tau_S\) and \(\tau_0\) satisfy one of the following conditions for a given \(\xi_0\):

(i) \(\forall s, t \in S \ \forall x_1, x_2 \in I:\)
    \[ [s] \tau_S = [t] \tau_S \land [x_1] \tau_I = [x_2] \tau_I \land [s \downarrow x_1] \xi_0 = [t \downarrow x_2] \xi_0 \]
    \[ \text{then } [s \delta x_1] \tau_S = [t \delta x_2] \tau_S \land [s \downarrow x_1] \tau_0 = [t \downarrow x_2] \tau_0 \]
    (for a Mealy machine),
(ii) \( \forall s, t \in S \forall x_1, x_2 \in I: \)
if \( [s] \tau_s = [t] \tau_s \land [x_1] \tau_I = [x_2] \tau_I \land [(s \delta x_1)] \xi_0 = [(t \delta x_2)] \xi_0 \)
then \( [s \delta x_1] \tau_s = [t \delta x_2] \tau_s \land [(s \delta x_1)] \tau_0 = [(t \delta x_2)] \tau_0 \)
(for a Moore machine).

In other words, \((\tau_I, \tau_s, \tau_0)\) is an output-dependent trinity for the independent output partition \(\xi_0\) if and only if, based on the knowledge of the block of a partition \(\tau_I\) in which the input of a machine \(M\) is contained, the block of a partition \(\tau_s\) in which the present state of \(M\) is contained and the block of a partition \(\xi_0\) in which the outputs of \(M\) are contained for inputs from a given block of \(\tau_I\) and states from a given block of \(\tau_s\), it is possible to calculate the block of \(\tau_s\) in which the next state of \(M\) is contained and the block of \(\xi_0\) in which the output of \(M\) is contained for the present state from a given block of \(\tau_I\) and input from a given block of \(\tau_I\).

**THEOREM 8.1** A machine \(M\) has a nontrivial serial full-decomposition of type NO with the realization of the output behaviour if such a partition trinity \((\pi_I, \pi_s, \pi_0)\) and such an output-dependent trinity \((\tau_I, \tau_s, \tau_0)\) for \(\xi_0 = \pi_0\) exist that the following conditions are satisfied:

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

(ii) \( |\pi_I| < |I| \land |\tau_0| > |I| \lor |\pi_s| < |S| \lor |\tau_s| < |S| \lor |\pi_0| < |O| \land |\tau_0| < |O| \).

**Proof** (for the case of Mealy machine)

Let \(M_1 = (\pi_I, \pi_s, \pi_0, \delta^1, \lambda^1)\) and \(M_2 = (\pi_0 \times \pi_I, \pi_s, \pi_0, \delta^2, \lambda^2)\) be two machines for which the following conditions are satisfied:

(1) \((\pi_I, \pi_s, \pi_0)\) and \((\tau_I, \tau_s, \tau_0)\) satisfy the conditions of theorem 9.1,

(2) \(\forall B_1 \in \pi_s \forall A_1 \in \pi_I: B_1 \delta^1 A_1 = [B_1 \delta^1 A_1] \pi_s \land B_1 \lambda^1 A_1 = [B_1 \lambda^1 A_1] \pi_0\),

(3) \(\forall B_2 \in \pi_s \forall A_2 \in \pi_I \forall C_1 \in \pi_0:

\[
B_2 \delta^2 (c_1, A_2) = [(S \delta x_1 \mid x \in B_2, x \in A_2, S_1 \in C_1)] \tau_s, \\
B_2 \lambda^2 (c_1, A_2) = [(S_1 \mid x \in B_2, x \in A_2, S_1 \in C_1)] \tau_0 .
\]

Since \((\pi_I, \pi_s, \pi_0)\) is a partition trinity (1), \(B_1 \delta^1 A_1\) is placed in just one block of \(\pi_s\) and \(B_1 \lambda^1 A_1\) is placed in just one block of \(\pi_0\).
This means that $B_{\lambda^1 \lambda^1}$ and $B_{\lambda^1 \lambda^1}$ are unambiguously defined.

Since $(\tau_I, \tau_0, \tau_0)$ is an output dependent trinity for $\xi_0 = \pi_0$ (1), the following condition is satisfied:

4. $\forall s, t \in S \forall x_1, x_2 \in I$:
   If $[s] \tau_0 = [t] \tau_0 \land [x_1] \tau_I = [x_2] \tau_I \land [s] \pi_0 = [t] \pi_0$.
   Then $[s] \delta_{x_1} \tau_0 = [t] \delta_{x_2} \tau_0 \land [s] \pi_0 = [t] \pi_0$.

From (4), it follows that $B_2 \delta^2 \lambda^2 \delta^2 (C_1, \lambda_2)$ and $B_2 \delta^2 \lambda^2 \delta^2 (C_1, \lambda_2)$ are defined unambiguously, because $\{s \delta_{x_1} \mid s \in B_2, x \in \lambda_2, s \lambda_{x_1} \in C_1\}$ is located in just one block of $\tau_0$ and $\{s \lambda_{x_1} \mid s \in B_2, x \in \lambda_2, s \lambda_{x_1} \in C_1\}$ is in just one block of $\tau_0$.

Let $\psi : I \rightarrow \pi_I \times \tau_I$ be an injective function,
$\phi : S \rightarrow \pi_S \times \tau_S$ be an injective function,
$\theta : \pi_0 \times \tau_0 \rightarrow 0$ 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. $\theta(C_1, C_2) = C_1 \cap C_2$ if $C_1 \cap C_2 \neq 0$.

It will be proved below that the serial connection of type $\lambda^2 \delta^2 \lambda^2 \delta^2 (C_1, \lambda_2)$ 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), $\theta$ is a one-to-one function and for $C_1 \cap C_2 \neq 0$:

8. $(C_1, C_2) \in O$.

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

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

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

$= ([s] \pi_S \delta^1 (x), [s] \tau_S \delta^2 ([s] \lambda_{x_1} \pi_0, [x] \tau_I))$ (definition 2.9)

$= ([s] \pi_S \delta_{x_1}, [s] \tau_S \delta^2 ([s] \lambda_{x_1} \pi_0 \land [x] \tau_I)) (2), (3))$

$= ([s] \delta_{x_1} \pi_S, [s] \delta_{x_1} \tau_S) \quad (1))$

$= \phi(s) \delta_{x_1} \quad (6))$

and similarly:

$\theta(\phi(s) \delta^* \psi(x)) =$

$= \theta([s] \pi_S, [s] \tau_S) \delta^* ([x] \pi_I, [x] \tau_I) \quad (5), (6))$

$= \theta([s] \pi_S \delta^1 (x), [s] \tau_S \delta^2 ([s] \lambda_{x_1} \pi_0, [x] \tau_I))$ (definition 2.9)
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 $(\pi_I, \pi_s, \pi_0)$ 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 $\pi_I$ and block of $\pi_s$), is able to calculate its next state and output (i.e. the appropriate blocks of $\pi_s$ and $\pi_0$).

Since $(\tau_I, \tau_s, \tau_0)$ is an output-dependent partition trinity for $\xi_0 = \pi_0$, based only on information about the block of $\tau_I$ containing the input, the block of $\tau_s$ containing the present state of $M$ and the block of $\pi_0$ 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 $\tau_s$ in which the next state of $M$ is contained and the block of $\tau_0$ 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 $\tau_0 \cdot \pi_0 = \pi_0(0)$, with information about blocks of $\pi_0$ calculated by $M_1$ and blocks of $\tau_0$ 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: $(\pi_I, \pi_S, \pi_0)$ and $(\tau_I, \tau_S, \tau_0)$ exist and they satisfy the following conditions:

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

(ii) $|\tau_S| \cdot |\pi_I| < |I| \land |\pi_S| \cdot |\tau_I| < |S| \lor |\tau_0| < |O| \land \forall |\tau_0| < |O|$

Proof (for the case of a Mealy machine)

Let $M_1 = (\pi_S \times \pi_I, \pi_S, \pi_0, \delta_1, \lambda_1)$ and $M_2 = (\pi_S \times \pi_I, \pi_S, \pi_0, \delta_2, \lambda_2)$ be the two machines for which the following conditions are satisfied:

(1) $(\pi_I, \pi_S, \pi_0)$ and $(\tau_I, \tau_S, \tau_0)$ satisfy the conditions of theorem 9.1,

(2) $\forall B1 \in \pi_S \lor B2 \in \tau_S \land \forall A1 \in \pi_I$:

\[ B1 \delta_1(B2, A1) = ([B1 \cap B2] \delta A1) \pi_S, \quad B1 \lambda_1(B2, A1) = ([B1 \cap B2] \lambda A1) \pi_0, \]

(3) $\forall B1 \in \pi_S \lor B2 \in \tau_S \land \forall A2 \in \pi_I$:

\[ B2 \delta_2(B1, A2) = ([B1 \cap B2] \delta A2) \tau_S, \quad B2 \lambda_2(B1, A2) = ([B1 \cap B2] \lambda A2) \tau_0. \]

Since $(\pi_I, \pi_S, \pi_0)$ and $(\tau_I, \tau_S, \tau_0)$ are the present-state-dependent trinities (1), $(B1 \cap B2) \delta A1$ is placed in just one block of $\pi_S$, $(B1 \cap B2)$ is placed in just one block of $\pi_0$, $(B1 \cap B2) \lambda A2$ is placed in only one block of $\tau_S$ and $(B1 \cap B2) \lambda A2$ is placed in only one block of $\tau_0$. 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 \( \psi : I \rightarrow \pi_I \times \pi_I \) be an injective function,

\( \phi : S \rightarrow \pi_S \times \pi_S \) be an injective function,

\( \theta : \pi_0 \times \pi_0 \rightarrow O \) be a surjective partial function

and

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

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

(6) $\theta(C_1, C_2) = C_1 \cap C_2$ if $C_1 \cap C_2 \neq 0$.

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 $\pi_0 \cdot \tau_0 = \pi_0(0)$ \((1)\), $\theta$ is a one-to-one function and for $C_1 \cap C_2 \neq 0$:
\[(7) \quad (C_1, C_2) \in S .\]
Therefore, $\forall s \in S \forall x \in I$:
\[
\phi(s) \delta^* \psi(x) = \]
\[
= ([s] \pi_s, [s] \tau_s) \delta^* ([x] \pi_I, [x] \tau_I) \quad \text{((4), (5))}
\]
\[
= ([s] \pi_s \delta^1 ([s] \tau_s, [x] \pi_I), [s] \tau_s \delta^2 ([s] \pi_s, [x] \tau_I))
\]
\[
= ([([s] \pi_s \cap [s] \tau_s) \delta^1 ([x] \pi_I), [([s] \tau_s \cap [s] \pi_s) \delta^2 ([x] \tau_I)] \tau_s)
\]
\[
= ([s_1 x] \pi_s, [s_1 x] \tau_s)
\]
\[
= \phi(s_1 x) \quad \text{((5))}
\]
and similarly:
\[
\theta(\phi(s) \lambda^* \psi(x)) = \]
\[
= \theta(([s] \pi_s, [s] \tau_s) \lambda^* ([x] \pi_I, [x] \tau_I)) \quad \text{((4), (5))}
\]
\[
= \theta([s] \pi_s \lambda^1 ([s] \tau_s, [x] \pi_I), [s] \tau_s \lambda^2 ([s] \pi_s, [x] \tau_I))
\]
\[
= ([s] \pi_s \lambda^1 ([s] \tau_s, [x] \pi_I) \cap [s] \tau_s \lambda^2 ([s] \pi_s, [x] \tau_I)) \quad \text{((6))}
\]
\[
= ([([s] \pi_s \cap [s] \tau_s) \lambda^1 ([x] \pi_I)] \pi_0 \cap [(s \tau_s \cap [s] \pi_s) \lambda^2 ([x] \tau_I)] \tau_0)
\]
\[
= [s_1 x] \pi_0 \cap [s_1 x] \tau_0 \quad \text{((1))}
\]
\[
= s_1 x \quad (\pi_0 \cdot \tau_0 = \pi_0(0))
\]

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 \((\pi_1, \pi_s, \pi_0)\) induced by \(\xi_{02} = \tau_0\) and \((\tau_1, \tau_s, \tau_0)\) induced by \(\xi_{01} = \pi_0\) exist and they satisfy the following conditions:

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

(ii) \(|\tau_0| \cdot |\pi_1| < |I| \cdot |\pi_0| \cdot |\tau_1| < |I| \cdot |\pi_s| < |S| \cdot |\tau_s| < |S| \cdot |\pi_0| < |O| \cdot \\
\quad \cdot |\tau_0| < |O|\).

Proof (for the case of a Mealy machine)

Let \(M_1 = (\tau_0 \times \pi_1, \pi_s, \pi_0, \delta^1, \lambda^1)\) and \(M_2 = (\pi_0 \times \tau_1, \tau_s, \tau_0, \delta^2, \lambda^2)\) be the two machines for which the following conditions are satisfied:

(1) \((\pi_1, \pi_s, \pi_0)\) and \((\tau_1, \tau_s, \tau_0)\) satisfy the conditions of theorem 10.1,

(2) \(\forall C_2 \in \tau_0 \ \forall B_1 \in \pi_s \ \forall A_1 \in \pi_1\) :
    \[ B_1 \delta^1_{(C_2, A_1)} = \left[ \{ s \delta \in C_2, s \in B_1 \land \text{find}(C_2) \wedge x \in A_1 \} \pi_s \right], \]
    \[ B_1 \lambda^1_{(C_2, A_1)} = \left[ \{ s \lambda \in C_2, s \in B_1 \land \text{find}(C_2) \wedge x \in A_1 \} \pi_0 \right], \]

(3) \(\forall C_1 \in \pi_0 \ \forall B_2 \in \pi_s \ \forall A_2 \in \pi_1\) :
    \[ B_2 \delta^2_{(C_1, A_2)} = \left[ \{ s \delta \in C_1, s \in B_2 \land \text{find}(C_1) \wedge x \in A_2 \} \tau_s \right], \]
    \[ B_2 \lambda^2_{(C_1, A_2)} = \left[ \{ s \lambda \in C_1, s \in B_2 \land \text{find}(C_1) \wedge x \in A_2 \} \tau_0 \right]. \]

Since \((\pi_1, \pi_s, \pi_0)\) is a partition trinity induced by \(\xi_{02} = \tau_0\) and \((\tau_1, \tau_s, \tau_0)\) is a partition trinity induced by \(\xi_{01} = \pi_0\) (1), the following conditions are satisfied:

(4) \((\pi_s', \tau_s, \tau_0)\) is a S-S pair,

(5) \((\pi_s', \tau_s, \pi_0)\) is a S-S pair,

(6) \((\pi_s', \tau_s, \tau_0)\) is a S-O pair,

(7) \((\pi_s', \tau_s', \pi_0)\) is a S-O pair,

(8) \((\pi_1, \pi_s)\) is an I-S pair,

(9) \((\pi_1, \pi_0)\) is an I-O pair,

(10) \((\tau_1, \tau_s)\) is an I-S pair,

(11) \((\tau_1, \tau_0)\) is an I-O pair.

From (5) and (8), it follows that \(\{ s \delta \in C_2, s \in B_1 \land \text{find}(C_2) \wedge x \in A_1 \} \pi_s\) is located in just one block of \(\pi_s\). From (7) and (9), it follows
that \((s \setminus x \mid s \in \mathcal{B}_1 \wedge s \in \text{ind}(C_2) \wedge x \in A_1)\) is located in only one block of \(\pi_0\). This means, that \(\mathcal{B}_1^1(C_2, A_1)\) and \(\mathcal{B}_1^2(C_2, A_1)\) are unambiguously defined.

Similarly, from (4) and (10), it follows that \((s \setminus x \mid s \in \mathcal{B}_2 \wedge s \in \text{ind}(C_1) \wedge x \in A_2)\) is located in just one block of \(\tau_s\) and, from (6) and (11), it follows that 
\((s \setminus x \mid s \in \mathcal{B}_2 \wedge s \in \text{ind}(C_1) \wedge x \in A_2)\) is located in just one block of \(\tau_0\). So, \(\mathcal{B}_2^2(C_1, A_2)\) and \(\mathcal{B}_2^2(C_1, A_2)\) are unambiguously defined.

Let \(\psi: I \rightarrow \pi_1 \times \tau_1\) be an injective function, 
\(\phi: S \rightarrow \pi_5 \times \tau_5\) be an injective function, 
\(\theta: \pi_0 \times \tau_0 \rightarrow 0\) be a surjective partial function
and

\[(12) \quad \psi(x) = ([x] \pi_1, [x] \tau_1),\]
\[(13) \quad \phi(s) = ([s] \pi_5, [s] \tau_5),\]
\[(14) \quad \theta(C_1, C_2) = C_1 \cap C_2 \text{ if } C_1 \cap C_2 \neq 0.\]

It will be proved below that the general connection of type \(P_0\) of the machines \(M_1\) and \(M_2\) defined above realizes the output behaviour of machine \(M\).

Since \(\pi_0 \times \tau_0 = \pi_0(0) (1)\), \(\theta\) is a one-to-one function and for \(C_1 \cap C_2 \neq 0:\)

\[(11) \quad (C_1, C_2) \in 0.\]

Therefore, \(\forall s \in S \forall x \in I:\)

\[
\phi(s) \downarrow \psi(x) =
\]
\[
= ([s] \pi_5, [s] \tau_5) \delta^\ast([x] \pi_1, [x] \tau_1)
\]
\[(\text{definition 2.11})\]
\[
= ([s] \pi_5 \delta^1([s] \tau_5, [x] \pi_1), [s] \tau_5 \delta^2([s] \pi_5, [x] \tau_1))
\]
\[(\text{definition 2.11})\]
\[
= ([s] \pi_5 \cap [s] \tau_5)' \delta([x] \pi_1) \pi_5, ([s] \tau_5 \cap [s] \tau_5)' \delta([x] \tau_1) \tau_5)
\]
\[(\text{definition 2.11})\]
\[
= ([s] \pi_5, [s] \pi_5') \tau_5
\]
\[(\text{definition 2.11})\]
\[
= \phi(s) \downarrow \psi(x)
\]
\[(\text{definition 2.11})\]

and similarly:

\[
\theta(\phi(s) \downarrow \psi(x)) =
\]
\[
= \theta(([s] \pi_5, [s] \pi_5') \downarrow \pi_1, [x] \pi_1, [x] \tau_1)
\]
\[(\text{definition 2.11})\]
\[
= \theta([s] \pi_5 \downarrow ([s] \pi_5', [x] \pi_1), [s] \tau_5 \downarrow ([s] \pi_5', [x] \tau_1))
\]
\[(\text{definition 2.11})\]
\[
= [s] \pi_5 \downarrow ([s] \pi_5', [x] \pi_1) \cap [s] \tau_5 \downarrow ([s] \pi_5', [x] \tau_1)
\]
\[(\text{definition 2.11})\]
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: $\pi_1 \cdot \tau = \pi_0 (0)$ 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 appropriate theorems proved here for 
\[ \pi_1 \cdot \tau_1 = \pi_1(0) \].

Similarly, considering a state machine \( M = (I, S, \delta) \) to be a Moore machine \( M' = (I, S, 0, \delta, \lambda) \) where \( 0 = S \) and \( \lambda \) is an identity function or a Mealy machine \( M'' = (I, S, 0, \delta, \lambda) \) where \( 0 = S \) and \( \lambda = \delta \), the appropriate 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 \( \delta \) and the output function \( \lambda \) rather than to consider them simultaneously. It is possible to abstract from the output function \( \lambda \) and to decompose first the state machine defined by the next-state function \( \delta \). Then, it is possible to realize the output function \( \lambda \), where \( \lambda \) 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 \( \delta \).

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


(171) Monnee, P. and M.H.A.J. Herben
MULTIPLE-BEAM GROUNDSTATION REFLECTOR ANTENNA SYSTEM: A preliminary study.

(172) Bastiaans, M.J. and A.M.M. Akkermans
ERROR REDUCTION IN TWO-DIMENSIONAL PULSE-AREA MODULATION, WITH APPLICATION TO COMPUTER-GENERATED TRANSPARENCIES.

(173) Zhu Yu-Cai
ON A BOUND OF THE MODELLING ERRORS OF BLACK-BOX TRANSFER FUNCTION ESTIMATES.

(174) Berkelaar, M.R.C.M. and J.F.M. Themmen
TECHNOLOGY MAPPING FROM BOOLEAN EXPRESSIONS TO STANDARD CELLS.

(175) Janssen, P.H.M.
FURTHER RESULTS ON THE McMILLAN DEGREE AND THE KRONECKER INDICES OF ARMA MODELS.

(176) Janssen, P.H.M. and P. Stoica, T. Söderström, P. Eykhoff
MODEL STRUCTURE SELECTION FOR MULTIVARIABLE SYSTEMS BY CROSS-VALIDATION METHODS.

(177) Stefanov, B. and A. Veekind, L. Zarkova
ARCS IN CESIUM SEEDED NOBLE GASES RESULTING FROM A MAGNETICALLY INDUCED ELECTRIC FIELD.

(178) Janssen, P.H.M. and P. Stoica
ON THE EXPECTATION OF THE PRODUCT OF FOUR MATRIX-VALUED GAUSSIAN RANDOM VARIABLES.

(179) Lieshout, G.J.P. van and L.P.P.P. van Ginneken
GM: A gate matrix layout generator.

(180) Ginneken, L.P.P.P. van
GRIDLESS ROUTING FOR GENERALIZED CELL ASSEMBLIES: Report and user manual.

(181) Bollen, M.H.J. and P.T.M. Vaessen
FREQUENCY SPECTRA FOR ADMITTANCE AND VOLTAGE TRANSFERS MEASURED ON A THREE-PHASE POWER TRANSFORMER.

(182) Zhu Yu-Cai
BLACK-BOX IDENTIFICATION OF MIMO TRANSFER FUNCTIONS: Asymptotic properties of prediction error models.

(183) Zhu Yu-Cai
ON THE BOUNDS OF THE MODELLING ERRORS OF BLACK-BOX MIMO TRANSFER FUNCTION ESTIMATES.

(184) Kadete, H.
ENHANCEMENT OF HEAT TRANSFER BY CORONA WIND.

(185) Hermans, P.A.M. and A.M.J. Knaks, I.V. Bruza, J. Dijk
THE IMPACT OF TELECOMMUNICATION ON RURAL AREAS IN DEVELOPING COUNTRIES.

(186) Fu Yanhong
THE INFLUENCE OF CONTACT SURFACE MICROSTRUCTURE ON VACUUM ARC STABILITY AND ARC VOLTAGE.

(187) Keiser, F. and L. Stok, R. van den Born
DESIGN AND IMPLEMENTATION OF A MODULE LIBRARY TO SUPPORT THE STRUCTURAL SYNTHESIS.
THE FULL DECOMPOSITION OF SEQUENTIAL MACHINES WITH THE STATE AND OUTPUT BEHAVIOUR REALIZATION.

ALWAYS: A system for wafer yield analysis.

OPTICAL COUPLERS FOR COHERENT OPTICAL PHASE DIVERSITY SYSTEMS.

LOCAL-FREQUENCY DESCRIPTION OF OPTICAL SIGNALS AND SYSTEMS.

A MULTI-FREQUENCY ANTENNA SYSTEM FOR PROPAGATION EXPERIMENTS WITH THE OLYMPUS SATELLITE.

ANALOG AND DIGITAL SIMULATION OF LINE-ENERGIZING OVERVOLTAGES AND COMPARISON WITH MEASUREMENTS IN A 400 kV NETWORK.

MARTINUS VAN MARUM: A Dutch scientist in a revolutionary time.

ON SYSTEM IDENTIFICATION USING PULSE-FREQUENCY MODULATED SIGNALS.

MODEL BUILDING FOR AN INGOT HEATING PROCESS: Physical modelling approach and identification approach.