A semiautomatic method for qualitative failure mode analysis

Taylor, J.R.

Publication date:
1974

Document Version
Publisher's PDF, also known as Version of record

Risö-M-1707

Risö-M-1707

Danish Atomic Energy Commission

Research Establishment Risö

ELECTRONICS DEPARTMENT

A semiautomatic method for qualitative failure mode analysis

by

J.R. Taylor

August 1974

R-4-74

Paper presented at the CSNI Specialist Meeting on: The Development and Application of Reliability Techniques to Nuclear Plant Liverpool, 8th-10th April 1974

A. E. K. Risø

Risø - M - 1707

Risø - M - 1207

Title and author(s)

A semiautomatic method for qualitative failure mode analysis.

J.R. Taylor

Date August 1974

Department or group

Electronics

Group's own registration number(s)

R-4-74

pages + tables + illustrations

Abstract

Modern control systems are becoming increasingly complex. If failure mechanisms and effect are known, it is often simple to make small design changes, which lead to fail safe design. The problem is to discover the large number of different failure possibilities.

The cause/consequence diagram provides a good way of describing the sequential effects in a failure, important in control systems. Production of cause/consequence diagrams is at present a skilled manual task.

A method is presented for producing cause/consequence diagrams automatically, starting with a block diagram of the system to be analysed, and equations describing the operation of each component, under normal and failure conditions. Theorem proving techniques are used to deduce the sequence of events occurring after a failure.

The method makes it possible to define what is meant by a "complete" failure analysis. In practice, it seems that such "complete" analyses require a large amount of computer time. Human interaction helps in avoiding unnecessarily detailed analysis, and in increasing efficiency of the analyses.

Copies to

Available on request from the Library of the Danish Atomic Energy Commission (Atomenergikommissionens Bibliotek), Risø, DK-4000 Roskilde, Denmark
Telephone: (03) 35 51 01, ext. 334, telex: 43116

F. 25-204

A semiautomatic method for qualitative failure mode analysis

Failure mode and effects analysis, and fault tree analysis, are two established techniques for studying the reliability of large systems. The method to be described here is a formalized version of "cause-consequence analysis" which is a diagrammatic technique for presenting the sequence of events leading to a failure, and the conditions under which these events can take place (D.S. Nielsen 1971).

The reasons for formalizing failure analysis, are the difficulty and cost of analysing the reliability of large systems, with many, unlikely, failure modes. The needs for more automatic methods of qualitative failure analysis have been studied in depth by Powers (1973) and methods for building fault trees automatically have been described by Fussel (1973). The methods described here are directed especially to those cases where sequence is important, particularly sequential controllers, computer control, and the checking of written operating procedures.

Cause consequence analysis

Cause consequence diagrams present events in a flow chart form. Alternative event sequences may depend on conditions within a system, and decision boxes describe this dependency. Broken lines are used to represent conditions, and conditions are combined using and/or gates. A cause consequence diagram can be regarded as a combination of fault trees and flow charts. (See example, Figure 1).

Cause consequence diagrams have certain advantage when compared with fault trees. They give a more concise expression of the sequential dependence of events, than do fault trees, and do not need to make use of time labelling conventions. This is an advantage when sequential control, operator procedures, or computer operation is to be described. (However, a cause consequence diagram can be translated to a fault tree, by using time labelling conventions).

As will be seen, the cause consequence diagram of a failure is also more directly related to the physical structure of the system which is to be analysed, than the corresponding fault tree, and its construction requires smaller steps in reasoning.

Systematic development of cause/consequence diagrams

The starting point for a systematic development of a cause/consequence diagram, is a block diagram showing the physical structure of a plant. Each block represents a piece of equipment. Each line represents an interconnection or relation between pieces of equipment, or plant components.

Each block requires a mathematical description of how it works, in input/output terms. These descriptions can be given by means of

equations, transfer functions, or logical statements.

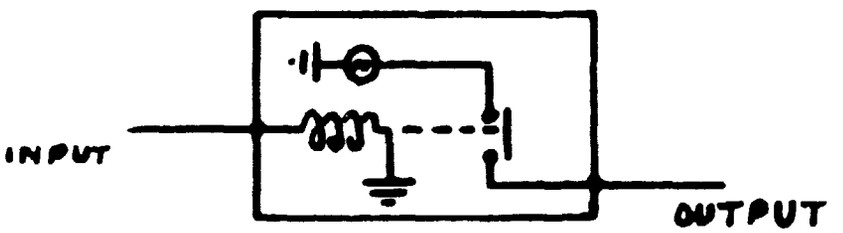
Each line on the diagram represents a variable - for example a voltage, and the blocks are regarded as recieving input variables and producing the values of output variables. Arrows are drawn on the diagram, representing the direction of cause and effect (Taylor 1973). For example the coil current of a relay may be regarded as an input variable, and the contact state of the relay as an output variable.

A "condition description" is a logical statement which describes the values of (some) system variables over a period of time. An "event" is a logical statement which describes a change in conditions.

Given a condition at the input to a component, and a mathematical description of the component, an output condition can be deduced.

Example:

A relay can be described by -
if power at input, then power at output



an input condition: power at input
yields an output condition: power at output

Similarly, if an input event description is given, an output event description can be deduced. For describing events, it is convenient to use the condition description which becomes true after the event has occured. It is also useful to associate with the event description, a record of the time at which the event occurred.

Fig. 5 and 6 shows one way of representing the process of deducing events within a system. If there are several output lines flowing from a component, then there may well be several event chains also flowing from the component. If there are several inputs to a component, then the consequences of an input event on one input, will depend on the conditions which are present at the other inputs. This gives rise to the "condition boxes" in the cause and consequence diagram.

Some components have "memory", and as a result, may produce output events at some time after an input event. In some cases, several delayed event chains will be produced. An example is the case of a timer relay. Components with memory introduce delay into the cause consequence diagram, and also the possibility for logical conflict between event chains.

Automation of diagram construction

The process of deducing output events, given an input event, can be automated, using techniques developed for automatic theorem proving (see e.g. Nilsson 1971). What is more, provided that the component and input event descriptions are not too complicated, the deduction process is "complete". This means that a standard, or "canonical" form is chosen for event descriptions. Then given an input event, a canonical form description of all the possible output events is produced, and the conditions under which these output events can occur.

What remains, is to automate the process by which the effect of a single "spontaneous" or "initial" event leads to following chains of events. The rules for tracing event chains are given in table 1.

Each time a component is reached which has several inputs, the ensuing chains of events will depend on the conditions at these inputs. This means that it is also necessary to trace backward through the block diagram, building up a "tree" of conditions, to check if the necessary input conditions can be fulfilled. This process is described in table 2.

An example

An example was given by Haasl of a chemical dosing system, and safety system. This example was used by Fussell to demonstrate a technique for building fault trees automatically (Fussel 1973). The example is used here to demonstrate the construction of cause consequence diagrams.

The system consists of a pump and reservoir. When a button is pressed the pump operates filling the reservoir with gas up to a certain pressure. When that pressure is reached, the required amount of gas is ready for delivery. A timer relay exists to switch off the pump, in the case of failure within the remainder of the control system.

A diagram of the cause-consequence model of the plant is given in Fig. 7. A complete listing of the fault trees for each component would take up too much space, but those for the relay and pressure switch are as follows.

relay coil

if (no failure) then if (power at input 1 and sink at input 2)

then (contact closed)

else if (coil failed) then (contact open)

contact

if (no failure) then if (contact closed and power at contact input)

then (power at contact output and sink at contact input)

else if (contact failed closed) then (contact closed)

pressure switch

if (no failure) then if (pressure > switching pressure)

then (contact open) else (contact closed)

else if (pressure switch failed or pressure line plugged)

then (contact closed)

A cause-consequence diagram for the example was developed by hand, following the rules given in tables 1 and 2, and it shows in Fig. 8 to 10.

Significanse for failure analysis

In order to make a complete cause consequence analysis in the way described here, the event chains leading from both "normal operation" events and failure events must be traced. This results in diagrams with a very large amount of information. To be useful, the cause-consequence diagrams must be edited. Events occurring during normal operation may be eliminated, and sequences of directly connected events condensed to a single event. If sequence is not significant, the cause consequence diagram can be condensed to a failure tree, as in Fig. 11.

Automation of some of the steps in constructing cause consequence diagrams has begun. It is felt that interactive analysis is better than complete automation, with an engineer controlling the development of the diagrams. In this way, it is possible for the significance of the diagrams to be presented more clearly, and the engineer is more involved with the working of the system. It also means that any deficiencies in plant models are more likely to be recognized.

The fact that the procedure described here is, in a sense "complete", is an advantage. At the moment engineers must check designs, both for "wear out" and design errors, on the basis of their experience and intuition. They carry a large responsibility, and their studies must be thorough and detailed. The technique described here depends on the degree of detail in individual component models. If the models are adequate to describe forms of component failure observed in the past, then the consequences of those component failures repeating themselves, within a different system, can be predicted completely.

Table 1

Method for obtaining basic cause consequense diagram.

1) Start with initial set of independent events and initial conditions.

2) Select an initial event (failure or normal operation) and
3) deduce the output event(s) for the related component.

4) Check the input conditions for the component, to discover which of the output events are feasible (see table 2).

5) If there are any conditions which conflict with conditions earlier in the event chain, or with the initial conditions, delete the related output event.

6) Add the new events, and the related condition trees to the diagram. Record the new conditions established for the component.

7) Select the earliest (in time) of the output events for the component, and trace the block diagram, so that the output events become input events to the next component. Repeat the procedure from step 3.

8) If there are no output events for a component, back track in the block diagram, to the first "undeveloped" output event, and develop that.

9) Iterate, to ensure that all possible system states are treated.

Table 2

Method for checking prior conditions for an event.

1) Begin with an output event for a particular component - the "main" component.

2) Deduce the input conditions for the component, which will allow the event to occur.

3) Trace one of the input lines to the component backwards, to the previous component in the block diagram, to determine whether the associated condition is feasible.

4) Deduce whether the output condition of the new component is feasible. The condition may be feasible because
   a) an earlier event established the condition, in which case this dependency should be recorded on the cause consequence diagram,
   b) the condition is an initial condition for the system.

5) If the condition is feasible, combine it in "and" form with conditions associated with other input lines to the main component. If the condition is not feasible, record it as "false".

6) Repeat from step 3, but using another input line for the main component.

7) When all the input lines for the main component have been checked, simplify the associated condition tree. If it simplifies to false - then ignore the output event.

Table 3

Method for editing cause/consequence diagram to give a fault tree.

1) Delete all events which depend on normal input conditions, and normal operation events alone.

2) If there is a series of "event boxes" in the diagram with no delay or decision boxes intervening, shrink the series to a single event.

3) Replace decision boxes by two "and" gates, as in figure 2.

4) Replace "decision to event" boxes by a single line, but record on subsequent "and" gates the relative timings of events as in figure 2.

5) Indicate times on all initial events and conditions. Trace through the fault tree recording the times on each event box, and updating the time value, whenever a delay box is reached. Delete the delay boxes.

References

D.S. Nielsen, 1971, The cause consequence method as a basis for quantitative accident analysis, Report Risø-M-1374.

Available from Library of the Danish Atomic Energy Commission
Risö DK 4000 Roskilde
Denmark

Powers and Tomkins, 1973, Fault tree synthesis, Nato Conference or reliability, Liverpool, England, July 1973.

Fussel, 1973, A formal methodology for fault tree construction, Nuclear Science and Engineering, 52:421 - 432, 1973.

Nillson, 1971, Problem solving methods in artificial intelligence, McGraw-Hill Book Company, New York.

Taylor, 1973, A formalisation of failure mode analysis, Report Risö-M-1654.

Available from Library of the Danish Atomic Energy Commission
Risö DK 4000 Roskilde
Denmark

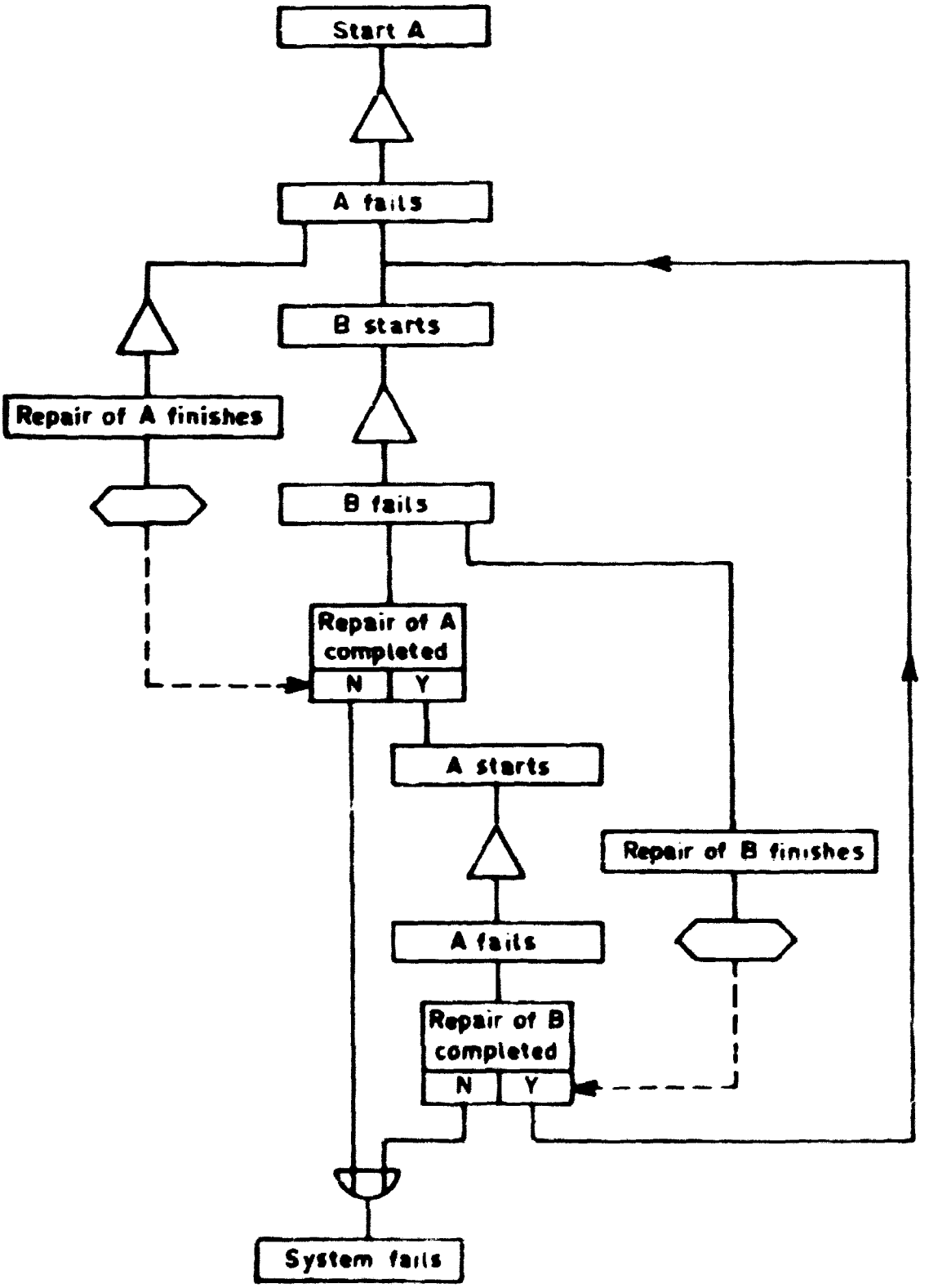
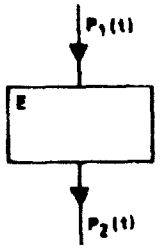


Fig. 1. Cause concequence diagram for system with standby and repair

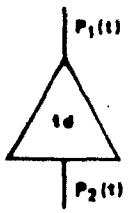
1. Simple chain of events, without delay



P(t) is probability distribution function for an event

S(t) is cumulative distribution function for an event

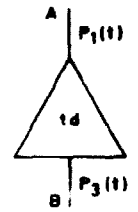
2. Chain of events with delay



$\forall t: P_2(t) = P_1(t - t_d)$

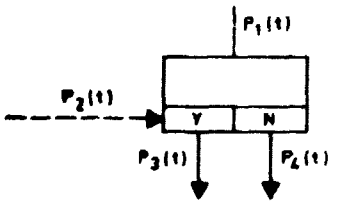
3. Chain of events with non deterministic delay

$P(td = t) = P_2(t)$



$\forall t: P_3(t) = \int_0^t P_1(t-x)\, P_2(x)\, dx$

4 Event depends on a prior condition

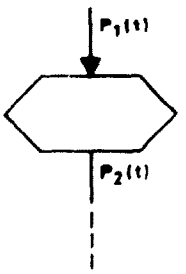


$\forall t\ P_3(t) = P_1(t).P_2(t)$

$\forall t\ P_4(t) = P_1(t).(1 - P_2(t))$

Decision box

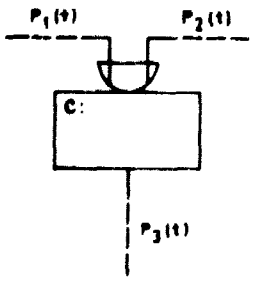
5. Condition that an event has already accured



$\forall t: P_2(t) = \int_0^t P_1(x)\, dx$

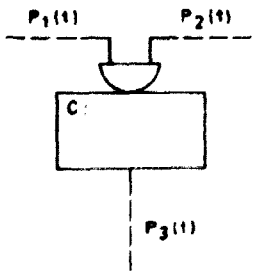
Event to condition box

6. Fault tree combination of conditions



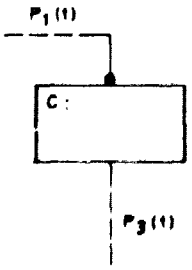
Or box

$\forall t: P_3(t) = P_1(t) + P_2(t) - P_1(t).P_2(t)$



And box

$\forall t: P_3(t) = P_1(t).P_2(t)$



Not box

$\forall t\ P_3(t) = 1 - P_1(t)$

Fig. 2. Symbols used in systematically generated cause/consequence diagrams

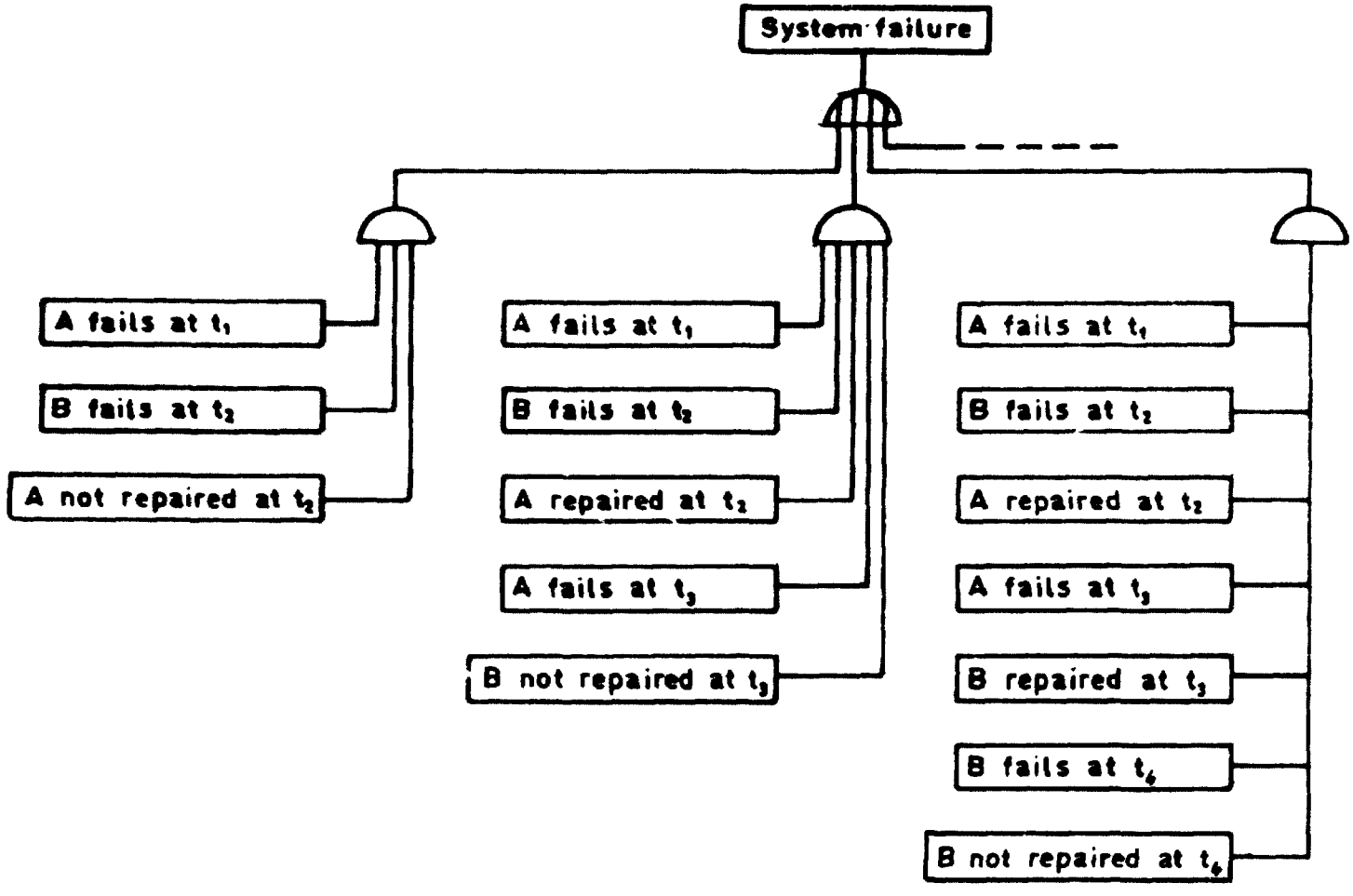


Fig. 3. Fault tree for system with standby and repair.

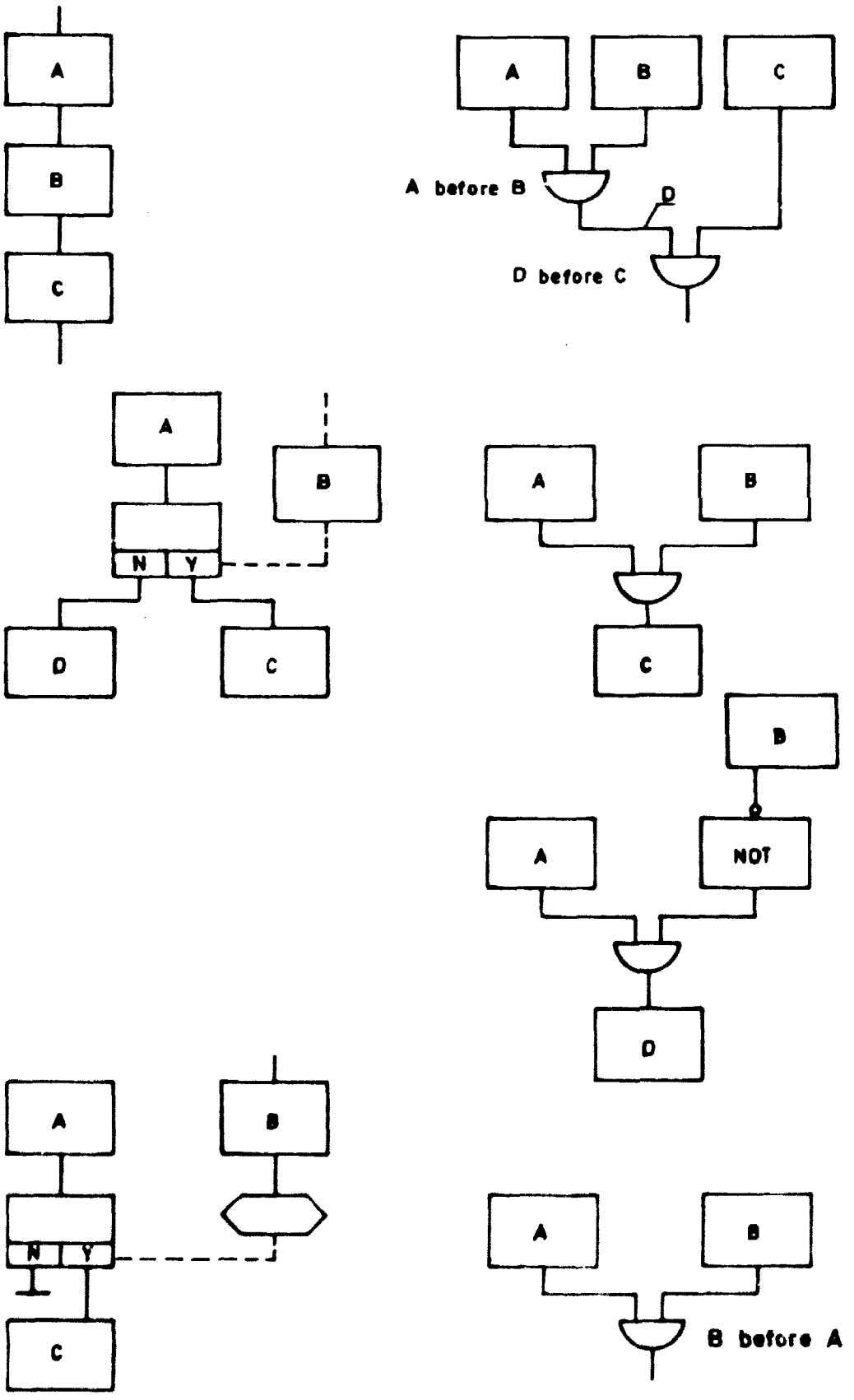


Fig. 4. Conversion between cause consequence diagrams and fault tree.

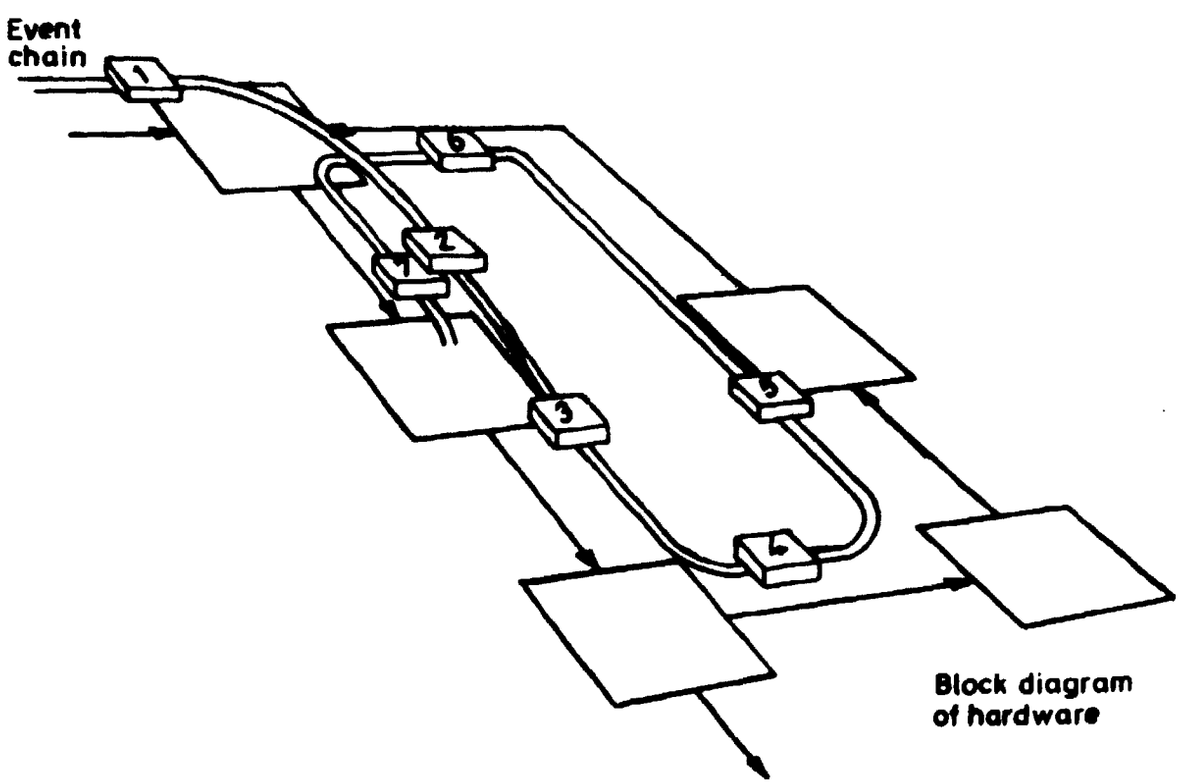


Fig. 5. 'Unwinding' an event sequence chain from a hardware block diagram

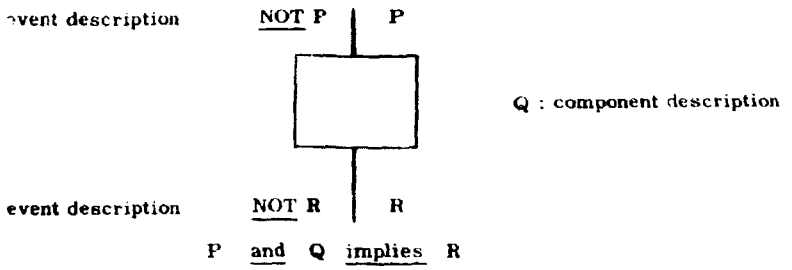
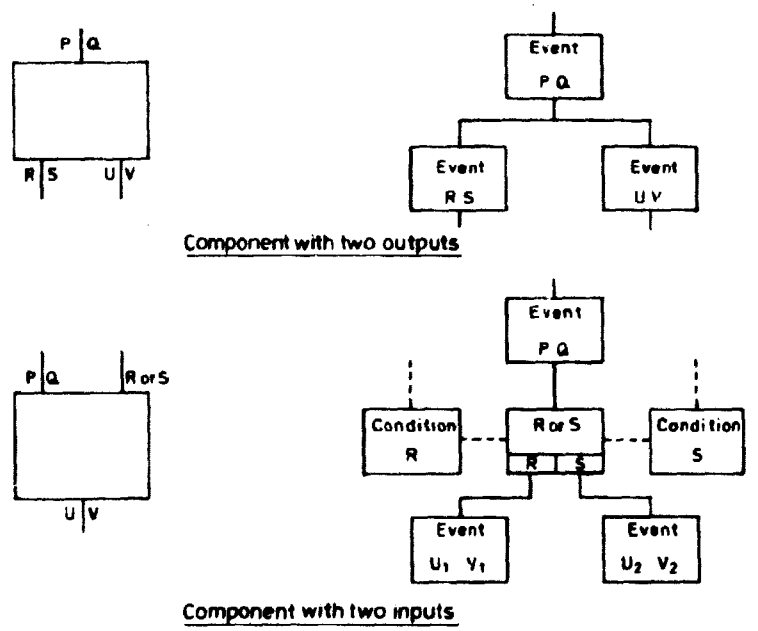


Fig. 6a. Simple event deduction across a component



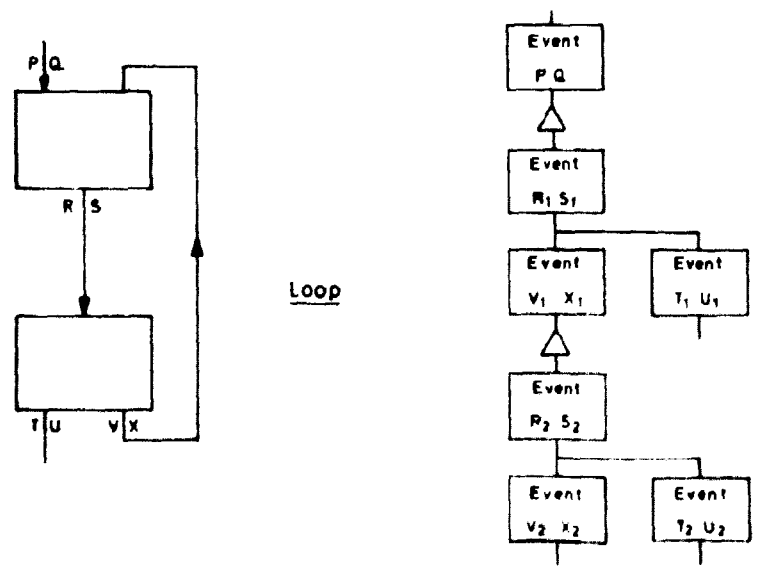


Fig 6 Block diagrams and event sequence diagrams

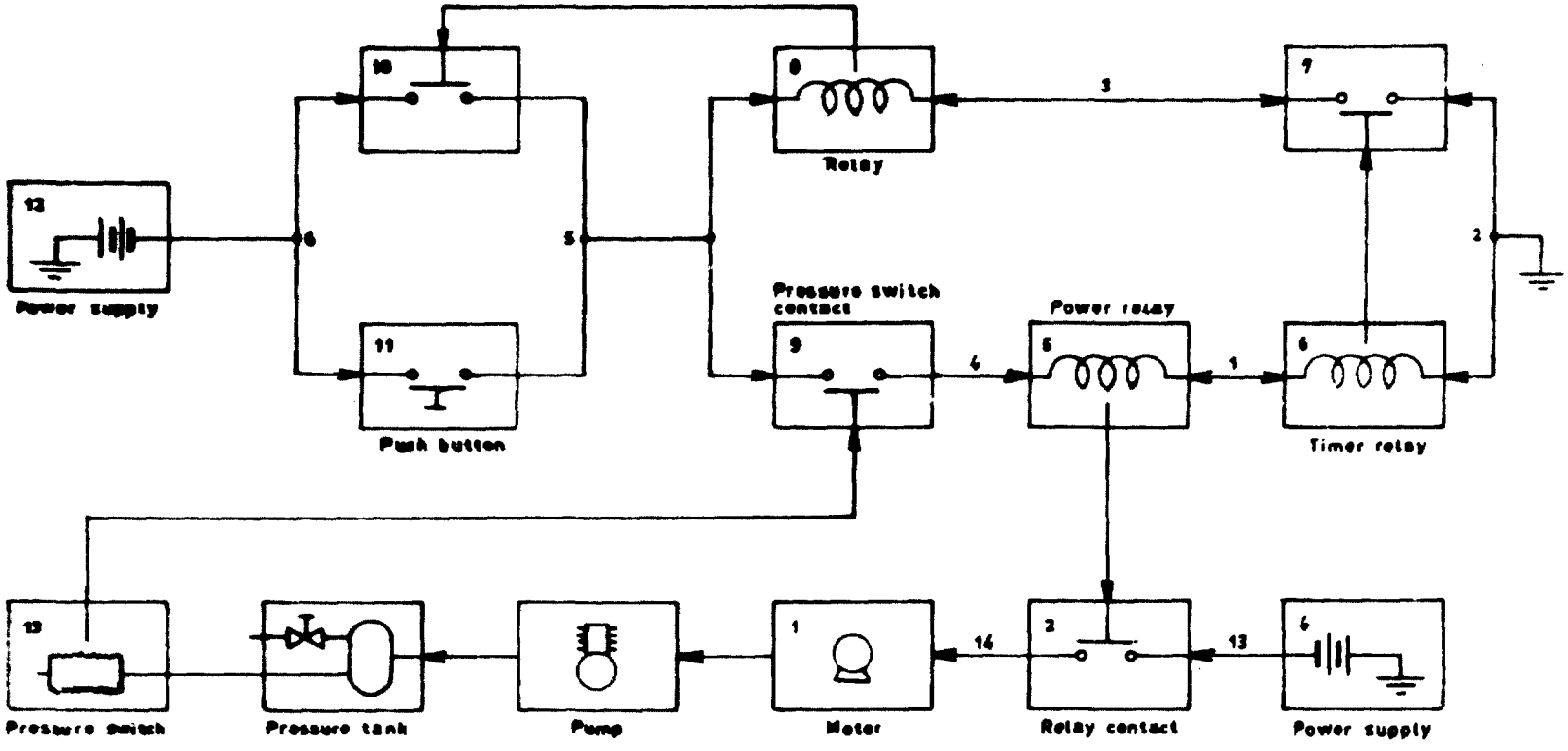


Fig. 7. Block Diagram of Pressure Tank Dosing System, With Cause - Effect Directions Marked.

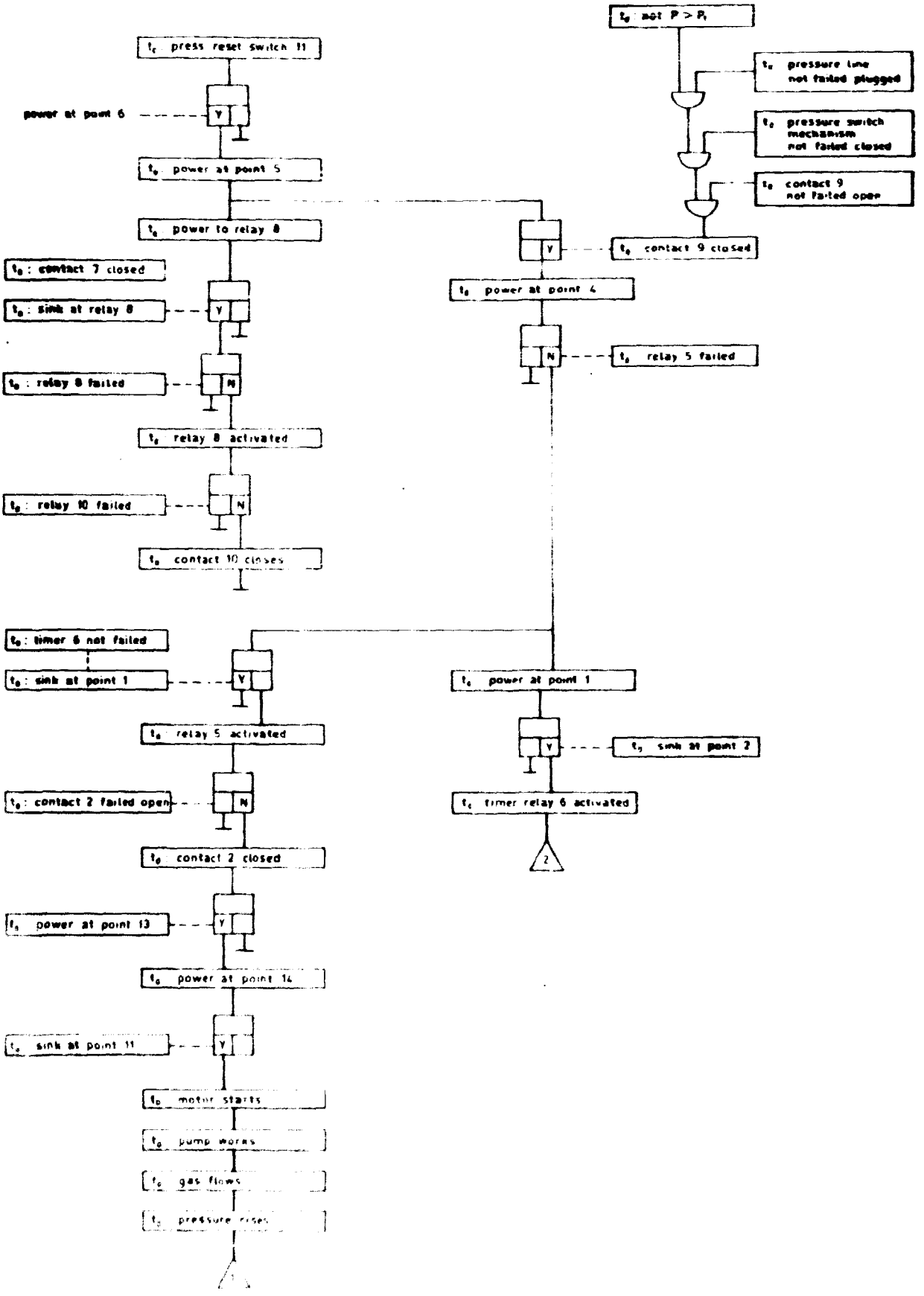


Fig. 8 Normal operation up to pump start

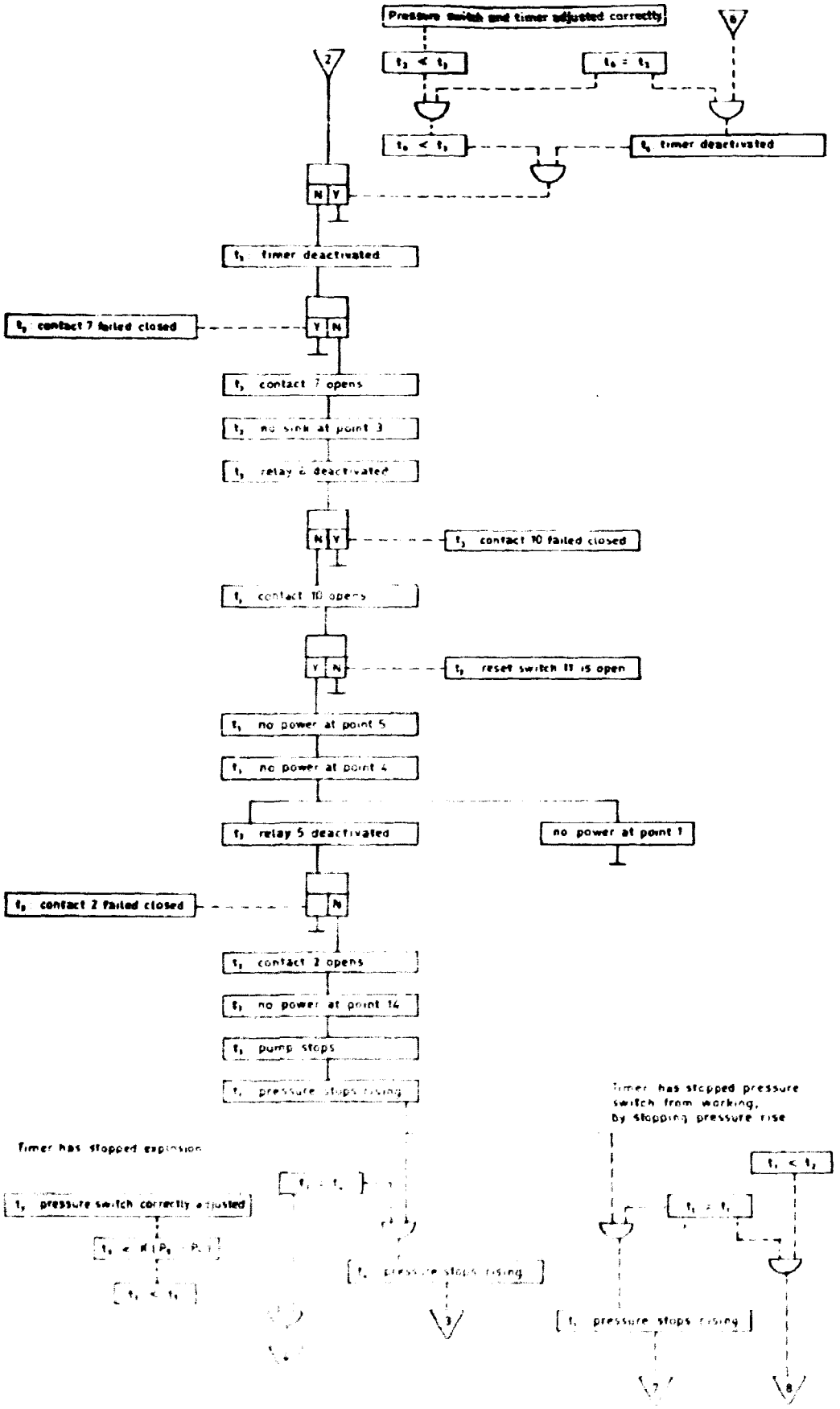


Fig. 1 Timer operation

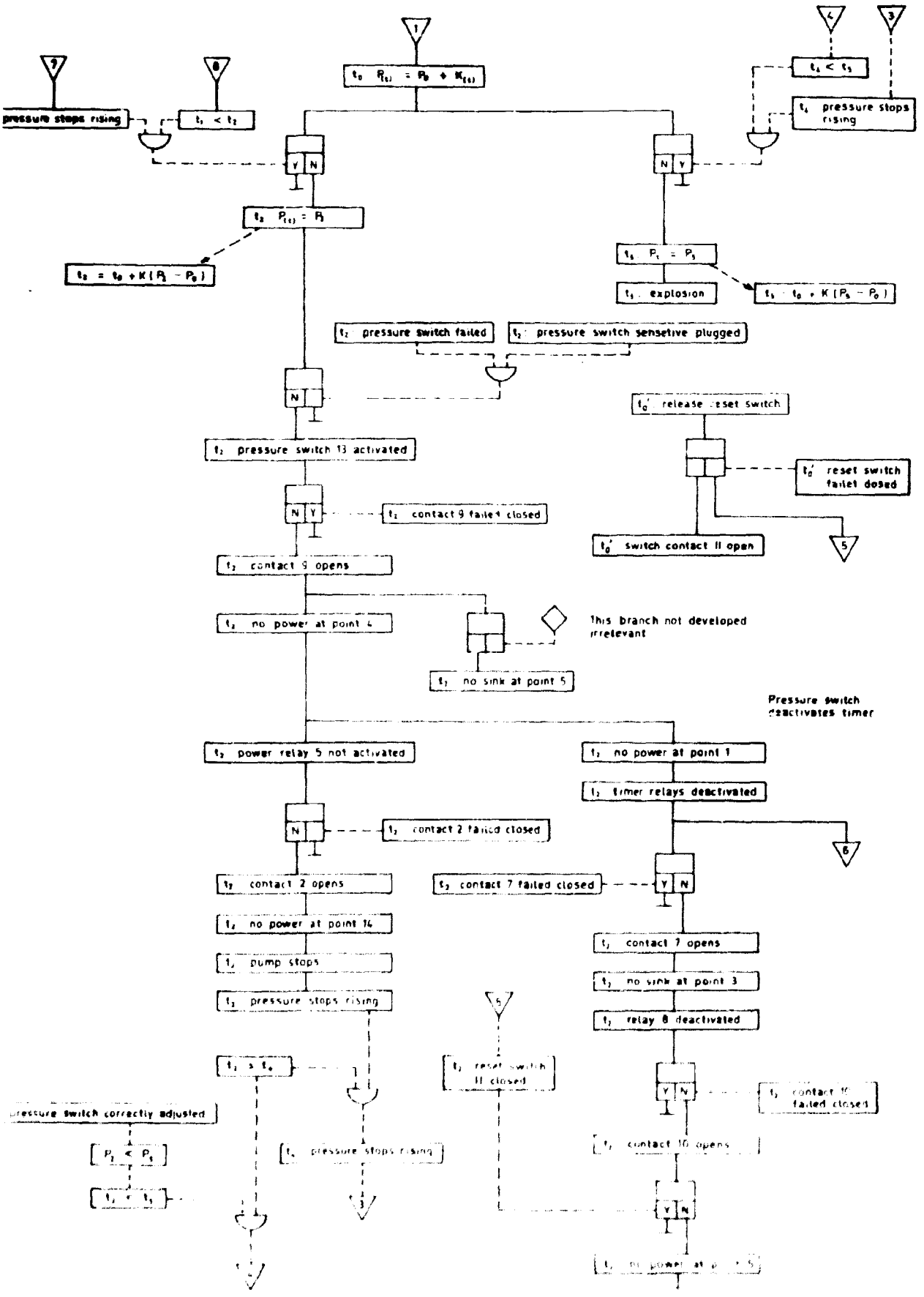


Fig. 19. Pressure switch operation and system failure

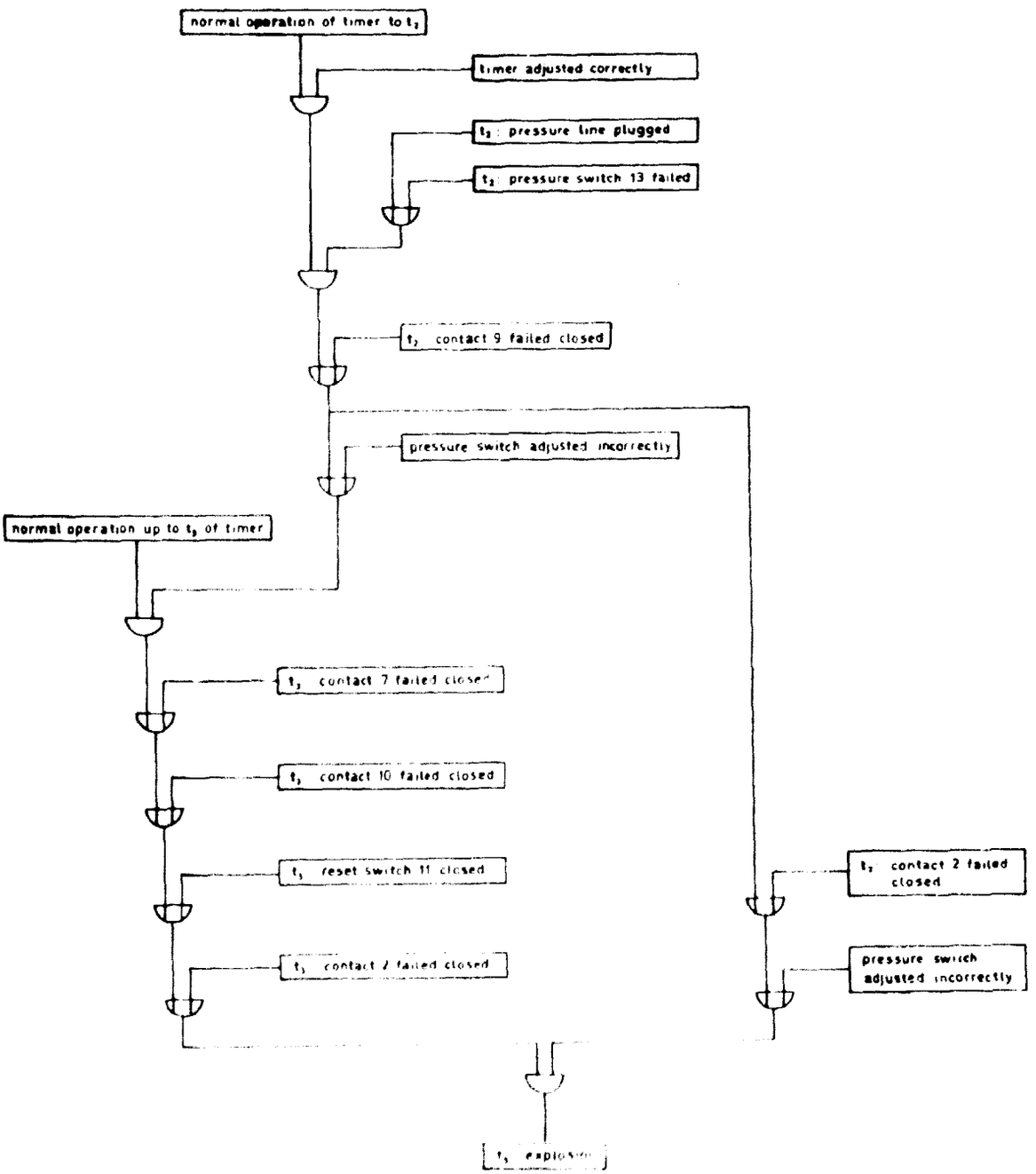


Fig. 11 Failure tree