Application of the Cause-Consequence Diagram Method to Static Systems

L.M.Ridley and J.D.Andrews
Department of Mathematical Sciences
Loughborough University
Loughborough
Leicestershire
LE11 3TU

Keywords: Cause-Consequence Analysis, Fault Tree Analysis, Binary Decision Diagram, Dependencies.

Summary

In the last 30 years various mathematical models have been used to identify the effect of component failures on the performance of a system. The most frequently used technique for system reliability assessment is Fault Tree Analysis (FTA) and a large proportion of its popularity can be attributed to the fact that it provides a very good documentation of the way that the system failure logic was developed. Exact quantification of the fault tree, however, can be problematic for very large systems and in such situations approximations can be used. Alternatively an exact result can be obtained via the conversion of the fault tree into a binary decision diagram. The binary decision diagram, however, loses all failure logic documentation during the conversion process.

This paper outlines the use of the Cause-Consequence Diagram method as a tool for system risk and reliability analysis. As with the fault tree analysis method, the Cause-Consequence Diagram documents the failure logic of the system. In addition to this the Cause-Consequence Diagram produces the exact failure probability in a very efficient calculation procedure. The Cause-Consequence Diagram technique has been applied to a static system and shown to yield the same result as those produced by the solution of the equivalent fault tree and binary decision diagram. On the basis of this, general rules have been devised for the correct construction of the Cause-Consequence Diagram given a static system. The use of the cause-consequence method in this manner has significant implications in terms of efficiency of the reliability analysis and can be shown to have benefits for static systems.

1. Introduction

Analysis of industrial systems is carried out to aid in the protection of facilities and to help reduce the risk of adverse events such as loss of profit, injury or death by reducing the frequency or consequences of such accidents. Since the early 1960's various mathematical models have been used to perform reliability analysis in order to predict the likelihood that a system will function given a demand. Each analysis model has different features which make it more appropriate to some system types than others and to achieve the most efficient analysis the simplest technique should be utilised.

The most commonly employed technique used to assess the probability of failure of industrial systems is the Fault Tree Analysis (FTA) method (1). For systems containing independent failure events it has been shown that the FTA technique produces a logical description of the failure process and can also yield, among other things, the systems unreliability. It has been highlighted, however, that this technique has limitations even when it is applied to systems containing independent failure events. Qualitatively, if the fault tree is complex then finding the minimal cut sets can be CPU intensive. In addition to this the exact top event probability, found via the inclusion-exclusion formula, may also be computationally expensive if the system contains even a moderate number of minimal cut sets. In the past this problem has been solved by using approximations for the top event probability. These approximations, however, can be inaccurate if the likelihood of component failure is not small. The problem of inaccuracies due to approximation techniques has been alleviated recently by the development of the Binary Decision Diagram (BDD) approach (2).

BDDS are based on Bryant's trees (3) and obtain the exact top event probability efficiently by expressing the system failure modes as disjoint paths. The calculation of the top event probability is achieved by summing the probabilities of these disjoint paths. This analysis procedure makes the BDD technique more efficient than the traditional FTA technique. The BDD however cannot be constructed from the system description and is developed from the fault tree representation of the system. During the conversion process the BDD loses all the causality information that is represented in the fault tree structure. In addition to this an inefficient ordering of the basic events can result in an excessively large diagram, which can prove difficult to analyse reducing the efficiency of the method.

A technique, however, has been developed that represents all system outcomes, given an initial event, on a diagram which contains a full textual description of the systems behaviour and produces an exact quantification of the system failure probability. The technique is based on the Cause-Consequence Diagram method which was developed at RISO laboratories in the 1970's to aid in the reliability analysis of nuclear power plants in Scandinavian countries (4). The method involves the identification of the potential modes of failure of individual components and then relates the causes to the ultimate consequences for the system (5). The consequences evaluated include those

that represent system failure as well as those that represent other system behaviour. As all consequence sequences are investigated the method can assist in identifying system outcomes which may not have been envisaged at the design stage.

Cause-Consequence analysis is most frequently applied to systems where the system state changes with time (6,7). No literature exists which documents the application of the Cause-Consequence Diagram method to a static system and this is the topic of this paper. Rules for construction and quantification of a Cause-Consequence Diagram representing a static system have been developed and applied to an example system.

2. The Cause-Consequence Diagram Method

The Cause-Consequence Diagram is developed from some initiating event, i.e. an event that starts a particular operational sequence or an event which activates certain safety systems. The Cause-Consequence Diagram comprises two conventional reliability analysis methods the FTA method and the Event Tree Analysis method. The Event Tree method is used to identify the various paths that the system could take, following the initiating event, depending on whether certain subsystems/components function correctly or not. The fault tree method is used to describe the failure causes of the subsystems considered in the event tree part of the diagram. This relationship is shown in figure 1.

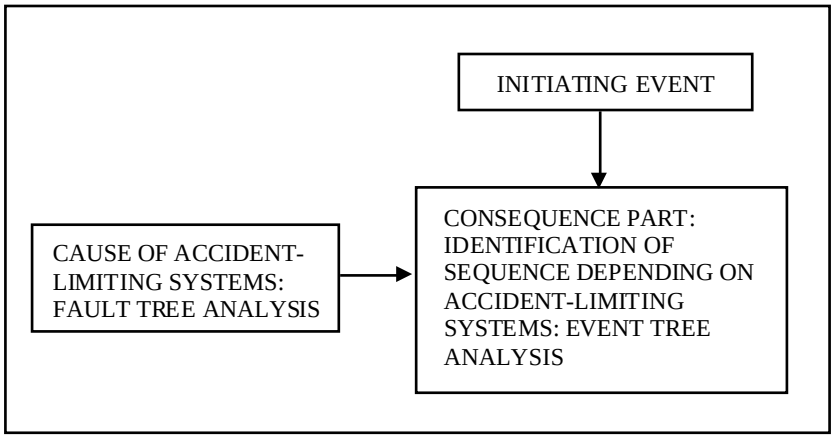


Figure 1 Structure of the Cause-Consequence Diagram

2.1 Symbols used for a Cause-Consequence Diagram

The symbols used for construction of a Cause-Consequence Diagram are illustrated in table 1.

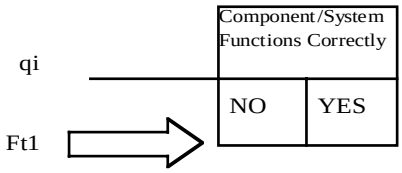
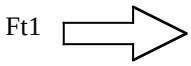
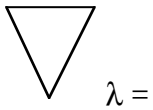
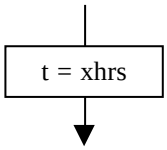
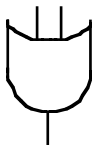
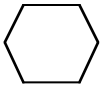
SYMBOL	FUNCTION
	The Decision Box represents the functionality of a component/system. The NO box represents failure to perform correctly, the probability of which is obtained via a fault tree or single component failure probability q_i
	Fault Tree Arrow represents the number of the fault tree structure which corresponds to the decision box
	The initiator triangle represents the initiating event for a sequence where λ indicates the rate of occurrence
	Time delay 1 indicates that the time starts from the time at which the delay symbol is entered and continues up to the end of the time interval in the delay symbol
	OR gate symbol: Used to simplify the Cause-Consequence Diagram when more than one decision box enters the same decision box or consequence box
	Consequence Box represents the outcome event due to a particular sequence of events.

Table 1 Cause-Consequence Diagram Symbols and Functions

2.2 Rules for Construction and Quantification

The Cause-Consequence Diagram technique has been applied to a static system and shown to yield the same results as those produced by the solution of the equivalent fault tree. On the basis of this study general rules have been devised for the correct construction of the Cause-Consequence Diagram given a static system. The use of the cause-consequence method in this manner has significant implications in terms of efficiency of the reliability analysis and can be shown to have benefits for static systems. The algorithm for static system analysis is as follows:

Step 1 Component Failure Event Ordering

If order of failure is irrelevant, which is the case in a static system, then the Cause-Consequence Diagram can be initiated by considering any of the components in the system. The analysis of the Cause-Consequence Diagram should yield identical results regardless of the component or variable ordering, however the actual diagrams may vary in size. The first step of the Cause-Consequence Diagram construction is therefore deciding on the order in which component failure events are to be taken. To

ensure a logical development of the causes of the system failure mode it was decided that the ordering should follow the temporal action of the system, for example the systems activation for the function required.

Step 2 Cause-Consequence Diagram Construction

The second stage involves the actual construction of the diagram. Starting from the initiating component the functionality of each component or sub-system is investigated and the consequences of these sequences determined. If the decision box is governed by a sub-system then the probability of failure will be obtained via a fault tree diagram.

Step 3 Reduction

If any decision boxes are deemed irrelevant, for example the boxes attached to the NO and YES branches are identical and their outcomes and consequences are the same, then these should be removed and the diagram reduced to a minimal form. Removal of these boxes will in no way affect the end result. This is illustrated in figure 2 where failure (F) occurs due to either of the two paths that terminate in the failure consequence. On one path the component A works, on the other it fails proving that the state of component A represented by the decision box is irrelevant.

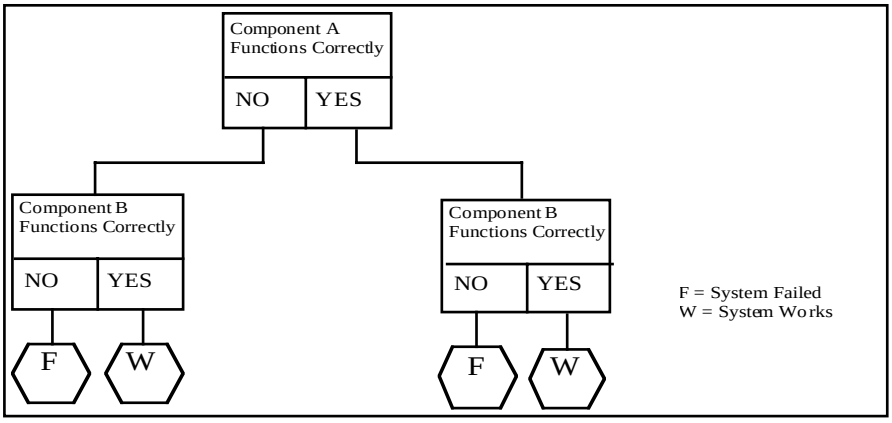


Figure 2 Redundant Decision Box

When a redundant decision box is identified, reduction is achieved by removing the box and entering the next decision/consequence box encountered in its place. Each decision box is inspected and when no further redundancies exist the Cause-Consequence Diagram is deemed minimal.

Step 4 System Failure Quantification

The probability of each consequence for a static system is determined by summing the probability of each set of events which lead to this particular outcome. Each sequence probability is obtained by simply multiplying the probabilities of the component events represented by the branch, as illustrated by Nielsen (8). This is possible as

each sequence of events is mutually exclusive and the probability of component failure events are assumed independent. The 4-step procedure can be represented in a flowchart as shown in figure 3.

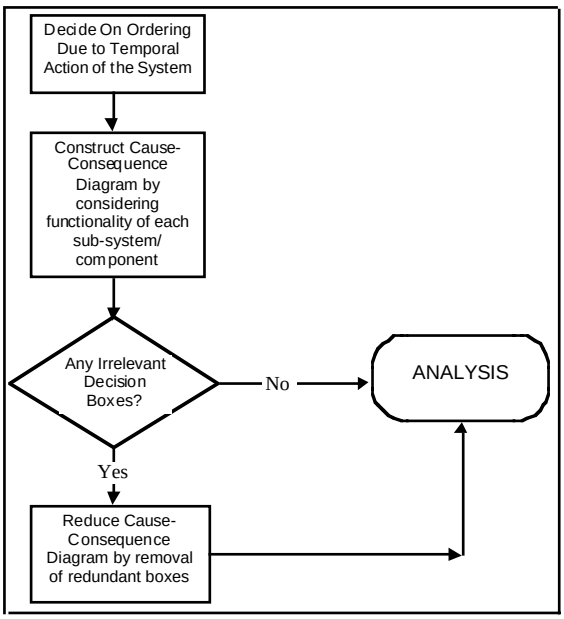


Figure 3 Flowchart for Cause-Consequence Diagram Construction

3. Example 1: Three Component System

The Cause-Consequence Diagram approach for static systems can be demonstrated by application to a very simple system example. In the approach it is shown why the method has potential advantages in comparison to a conventional fault tree study for larger systems. The example system contains three components A, B and C and system failure is caused by either A and B failing together or C failing alone. The system failure causes are illustrated as a fault tree structure in figure 4.

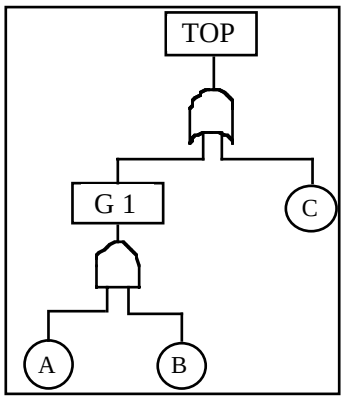


Figure 4 Example Fault Tree

The Cause-Consequence Diagram was constructed and analysed using the algorithm developed.

Steps 1 and 2 Component Failure Event Ordering and Cause-Consequence Diagram Construction

The ordering chosen was that of A, B, C and the Cause-Consequence Diagram was constructed by inspecting the failures of those components in that order (Figure 5).

Step 3 Reduction

Boxes 3 and 4 are both irrelevant and were therefore removed. This process reduced the Cause-Consequence Diagram, the final form being illustrated in figure 6, and as no further redundancies existed the diagram was minimal.

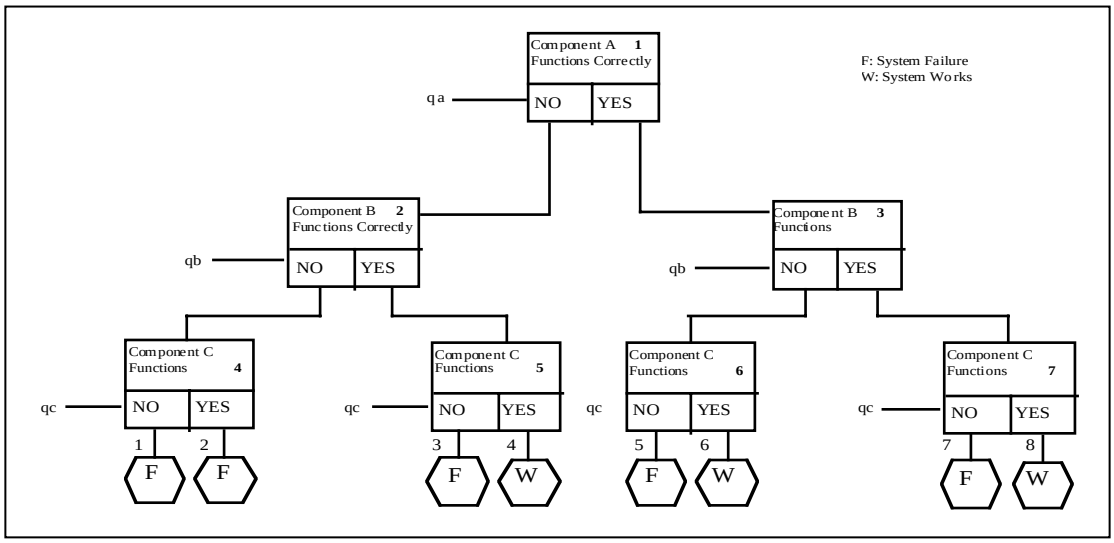


Figure 5 Cause-Consequence Diagram for three component system

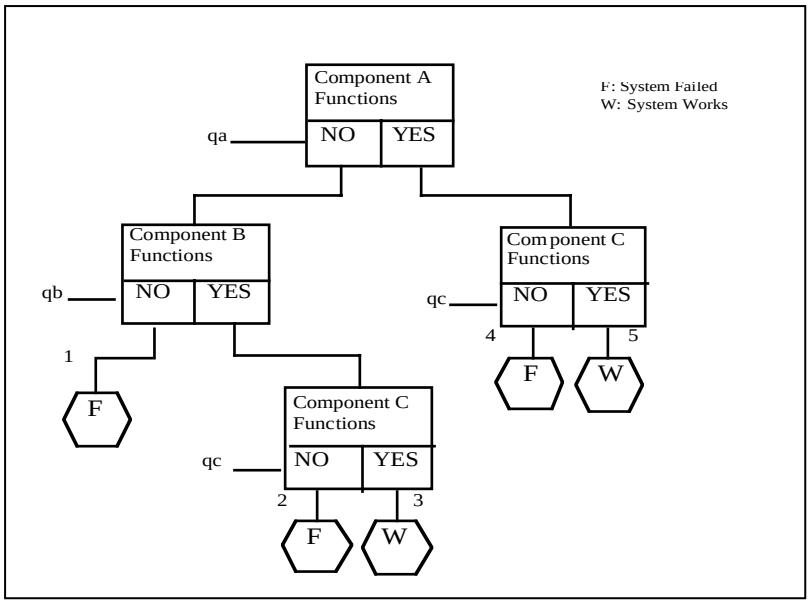


Figure 6 Reduced Cause-Consequence Diagram

Step 4 System Failure Quantification

The probability of system failure is equal to the sum of the probability of the 3 sequence paths that lead to the consequence '**F**'. Therefore since the paths are mutually exclusive:

Probability of Failure = P(Path 1) + P(Path 2) + P(Path 4)

$$= q_A.q_B + q_A.(1-q_B).q_C + (1-q_A).q_C$$
$$= q_A.q_B + q_A.q_C - q_A.q_B.q_C + q_C - q_A.q_C$$
$$= q_A.q_B + q_C - q_A.q_B.q_C$$

The fault tree quantification, using the exact method, calculates the top event probability to be identical to that obtained by the Cause-Consequence Diagram approach. By studying the reduced form of the Cause-Consequence Diagram it can be noted that it is equivalent to the Binary Decision Diagram (BDD) for the fault tree in figure 4, with the variable ordering A<B<C (Figure 7). The top event probability can also be obtained directly from the BDD by multiplying the probabilities down the paths that lead to the terminal 1 node (9).

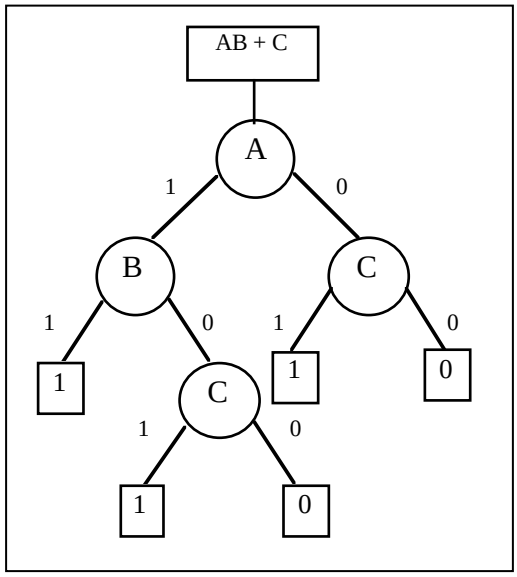


Figure 7 BDD with variable ordering A<B<C

4. Repeated Events

If the four stage procedure developed to construct and analyse a Cause-Consequence Diagram is to be considered as a generally applicable approach it must be capable of dealing with the events which occur in more than one fault tree structure attached to the decision boxes in any sequence path. It can be shown that the Cause-Consequence Diagram method can deal with repeated events in a more efficient way to that used for FTA. Using the Cause-Consequence Diagram method there is no need to obtain the Boolean expression of the top event and then manipulate it to produce a minimal form prior to analysis. The cause-consequence method deals with sequences of events which either occur (fail) or not occur (work). The probability of a particular outcome

is obtained by summation of the probability of all paths that lead to the outcome. Summation of the probabilities of the mutually exclusive paths results in the development of the reduced form which would be obtained from the fault tree following Boolean reduction. An algorithm has been developed that can trace through a Cause-Consequence diagram, identify and extract any repeated basic events in more than one fault tree structure on the same sequence path. The procedural steps used in the extraction algorithm are:

1) Identify the fault tree structures in the path under inspection.

2) Each fault tree identified in a path undergoes a modularisation process (10) and the independent subtrees identified are stored.

3) Each independent subtree for each fault tree diagram is compared to one another and following the identification of any common subtrees or individual basic events the Cause-Consequence Diagram is modified.

4) The Cause-Consequence Diagram is modified using the following rules:
    i) Following the identification of a common subtree or basic event the common element is extracted and set as a new decision box at the highest point in the Cause-Consequence diagram which has all dependencies below it.
    ii) The Cause-Consequence diagram is then duplicated on each branch starting from the new decision box.
    iii) Having developed a single decision box for the common subtree or basic event, the decision boxes that contained the common event prior to extraction require modification. The common event/s are set to 1 (TRUE) in the fault trees following the NO outlet branch from the new decision box, as this indicates failure, and 0 (FALSE) in the fault trees following the YES outlet branch to signify that the common event/s works.
    iv) After extraction of the common subtree or basic event each fault tree which has been modified requires reorganisation. Each fault tree containing the extracted Boolean variable is inspected and the fault trees modified by setting the Boolean variable to represent the path taken in the cause-consequence diagram.
    v) The Cause-Consequence Diagram is then reduced to a minimal form by removing any redundant decision boxes identified.

This procedure is repeated until all sequence paths have been inspected and no repeated subtrees or basic events discovered.

5. Industrial Example

As an example the technique has been applied to the simple high pressure protection system depicted in figure 8. The basic functions of the components present in the high pressure protection system are shown in table 2. The function of the system is to

prevent the passage of a high-pressure surge. The high pressure originates from a production well and the equipment to be protected are vessels located downstream on the processing platform.

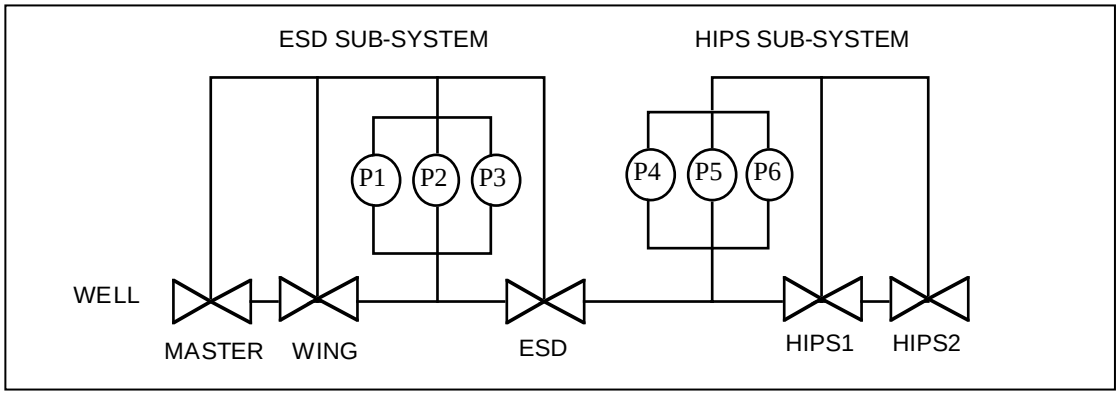


Figure 8 High-integrity protection system (HIPS)

The first level of protection is the emergency shutdown (ESD) sub-system. This comprises of 3 pressure sensors, for which 2 out of 3 must indicate a high pressure to cause a trip. Three shutdown valves, a Master, a Wing and an ESD valve activate to trip. If a high pressure surge is detected then the ESD system acts to close the Master valve, the Wing valve and the ESD valve. To provide an additional level of protection a second sub-system is included, the high-integrity protection sub-system (HIPS). This sub-system also comprises of 3 pressure sensors, 2 to trip, and 2 isolation valves labelled HIPS1 and HIPS2. The HIPS works in an identical manner to the ESD but has independent pressure sensors. The pressure sensors for each sub-system feed information into a common computer.

The Cause-Consequence Diagram was constructed following the rules given in section 2.

Component	Function	Failure Modes	λ	Mean Repair Time	Maintenance Test Interval Time
Master Valve	To stop high pressure surge passing through system	Valve fails open: VM	$1.14x10^{-5}$	36.0	4360
Wing Valve	To stop high pressure surge passing through system	Valve fails open: VW	$1.14x10^{-5}$	36.0	4360
ESD Valve	To stop high pressure surge passing through system	Valve fails open: VE	$5.44x10^{-6}$	36.0	4360
HIPS1 Valve	To stop high pressure surge passing through system	Valve fails open: VH1	$5.44x10^{-6}$	36.0	4360
HIPS2 Valve	To stop high pressure surge passing through system	Valve fails open: VH2	$5.44x10^{-6}$	36.0	4360
Solenoid	To supply power to valves	Fails Energized: SM,SW,SE,SH1 ,SH2	$5.0x10^{-6}$	36.0	4360
Relay Contacts	To supply power to solenoids (2 per solenoid)	Fails Closed R1-R10	$0.23x10^{-6}$	36.0	4360
Pressure Sensors	Indicates the level of pressure to the computer	Fails to record actual pressure: P1-P6	$1.5x10^{-6}$	36.0	4360
Computer	Reads information sent from pressure sensors and acts to close appropriate values	Fails to read or act on information: C	$1x10^{-5}$	36.0	4360

Table 2 Component Functions for HIPS System

Steps 1 and 2 Event Ordering and Cause-Consequence Diagram Construction

The ordering was based on the action of components which could perform the task required by the system i.e. Master Valve, Wing Valve, ESD Valve, HIPS1 Valve, HIPS2 Valve. The Cause-Consequence Diagram was constructed by considering the functionality of each valve and their effect on the system . Following the removal of all redundant decision boxes the minimal cause-consequence structure was created (Figure 9). The fault trees developed for each decision box are illustrated in figure 10a and 10b.

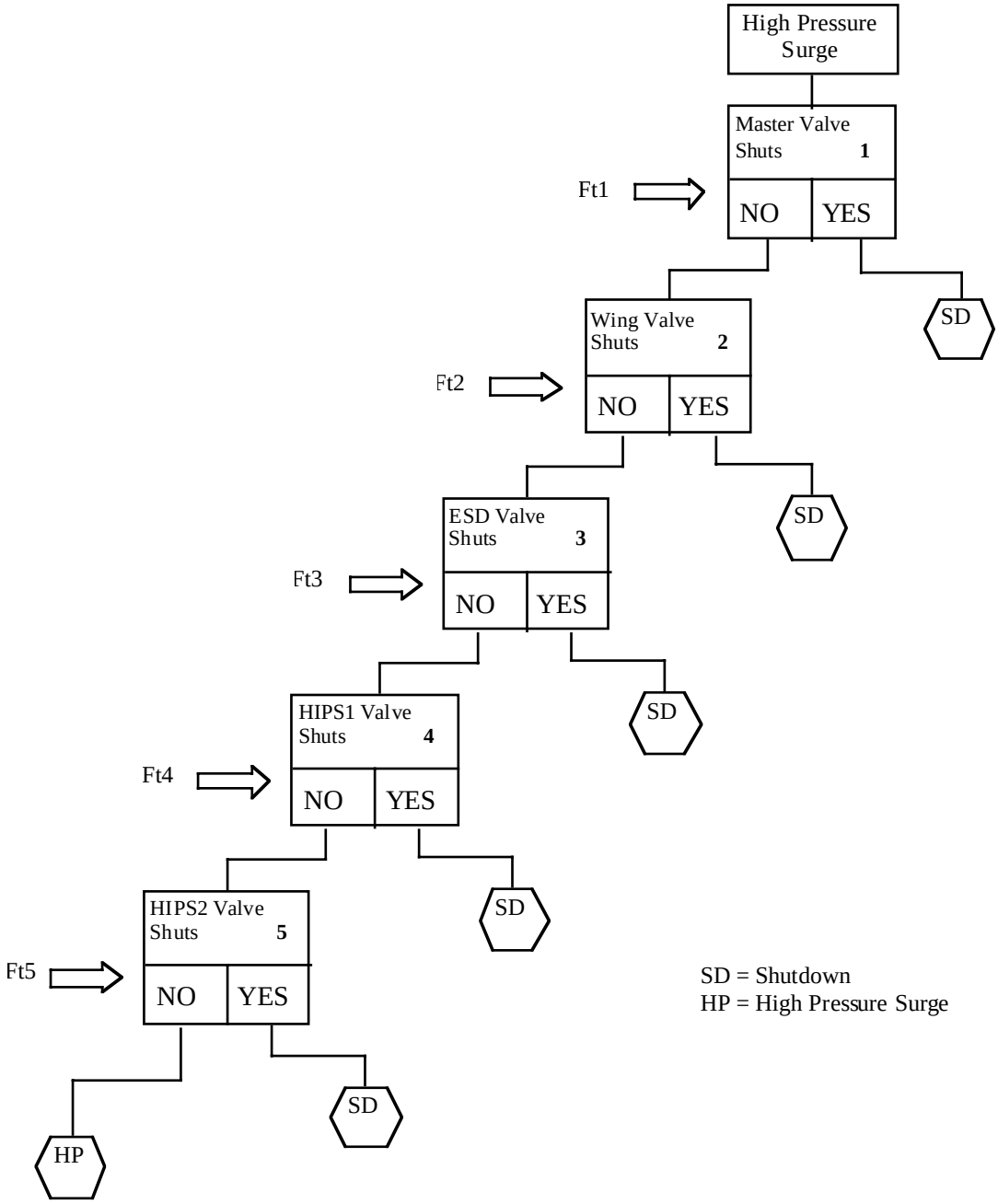


Figure 9 Cause-Consequence Diagram for HIPS System

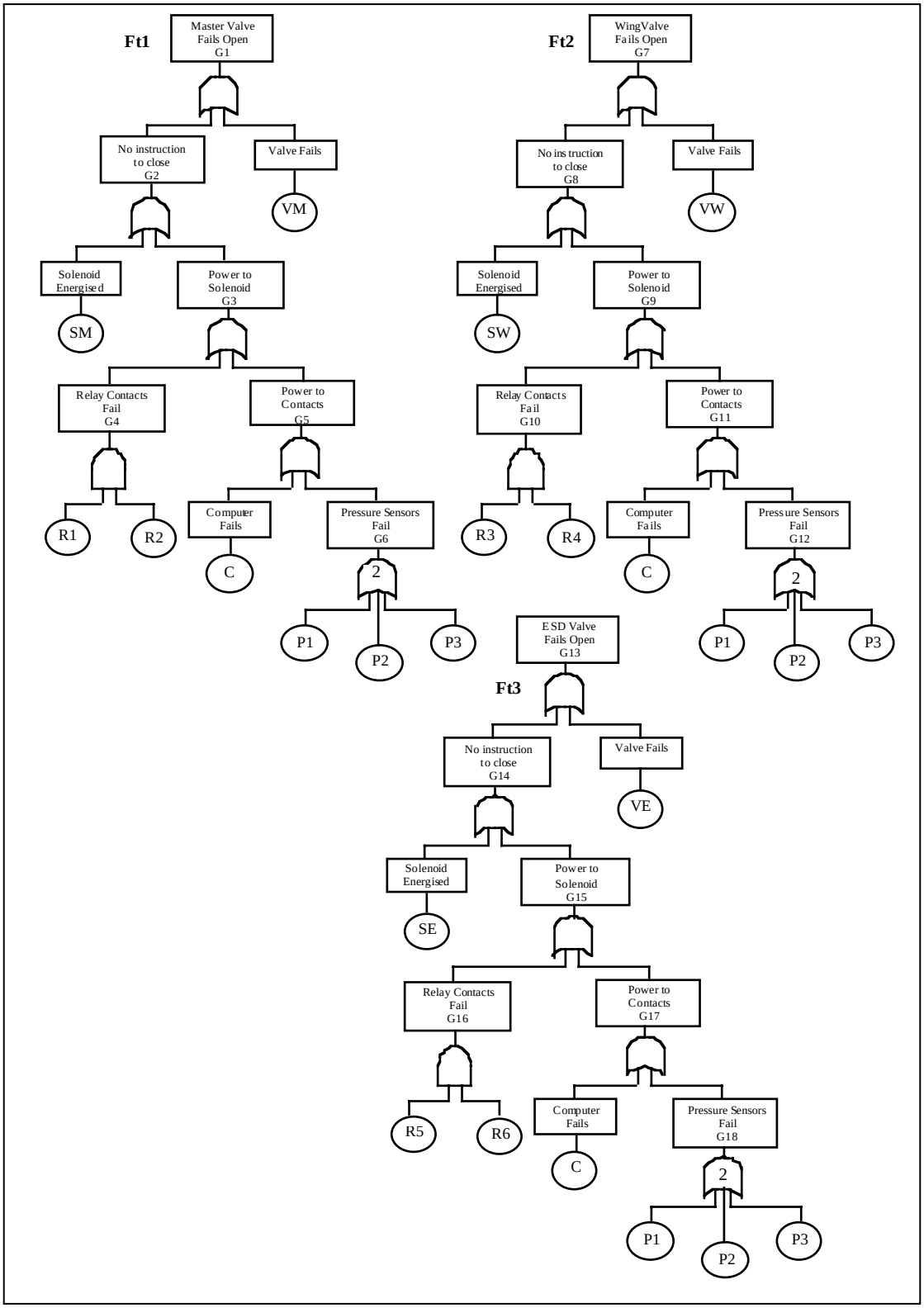


Figure 10a Fault Trees for Cause-Consequence Diagram for ESD sub-system

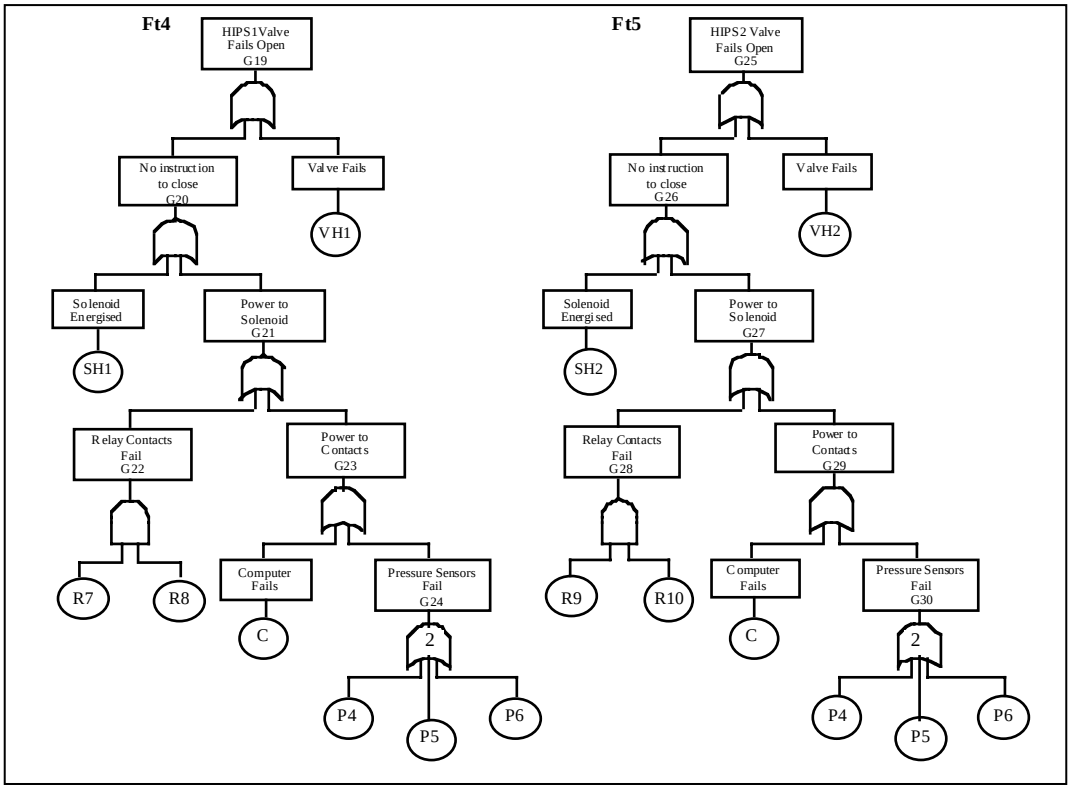


Figure 10b Fault Trees for Cause-Consequence Diagram for HIPS sub-system

Following the construction of the Cause-Consequence Diagram, each sequence path is inspected and any common independent subtrees or basic events are identified. The first sequence path inspected in the HIPS system identified that a common submodule was present in Ft1, Ft2 and Ft3, namely G6, G12 and G18 respectively. Extraction of this common submodule results in the Cause-Consequence Diagram depicted in figure 11 with corresponding fault trees shown in figure 12.

From this new version of the Cause-Consequence Diagram for the HIPS system, all sequence paths were investigated and modified accordingly using the rules outlined in section 4.

The final Cause-Consequence Diagram is illustrated in figure 13, with corresponding fault trees shown in figure 14. This is now in a form where each path contains independent events in the decision boxes and can be quantified with ease. The probability of a high pressure surge could now be obtained by summing the probabilities of ending in the consequence **HP**, which was reached via 5 mutually exclusive paths. Therefore

$$\text{Probability (High Pressure)} = \sum_{i=1}^{n} P(\text{Path i})$$

Component failures on the safety system are unrevealed and tested and repaired on scheduled maintenance. Their failure probabilities are given by equation (1).

$$Q_i = \lambda_i\left(\tau + \frac{\theta}{2}\right) \qquad (1)$$

The system unavailability was calculated as 2.216×10^{-2}. The figure is identical to that produce by the FTA and BDD methods. This result does not reflect poorly on the Cause-Consequence Diagram method, in comparison to the FTA method, it merely emphasizes the fact that this particular system can be failed by a single component, the computer. The remaining minimal cut sets are of order 4 or more and therefore have little effect on the overall system unavailability. For a system that contained a large number of small order minimal cut sets it can be stated that the Cause-Consequence Diagram method would yield a more accurate result than that obtained via FTA. The Cause-Consequence Diagram produced is of a similar form to that of the BDD for the system, however the Cause-Consequence Diagram is more concise due to extract of common submodules rather than extraction of each basic events present in the submodule.

6. Conclusion

An algorithm has been developed that will produce the correct Cause-Consequence Diagram and calculate the exact system failure probability for static systems with binary success or failure responses to the trigger event. This is achieved without having to construct the fault tree of the system and retains the documented failure logic of the system.

The Cause-Consequence Diagram is reduced to a minimal form by firstly removing any redundant decision boxes and secondly by manipulating any common failure events which exist on the same path. The common failure events can be extracted as common submodules or individual events. This process is equivalent to constructing the Fault Tree, converting it to a BDD and identifying and extracting independent submodules. The minimised Cause-Consequence Diagram is then analysed using a BDD analysis procedure. Thus exact rather than approximate calculations are performed.

The advantages of the Cause-Consequence Diagram are:

- The diagram can be constructed directly from the system description
- Dependencies in the system can be incorporated in the analysis
- The system is modularised to increase efficiency
- Exact calculation procedures are adopted

7. References

1) J.D.Andrews, T.R. Moss, "Reliability and Risk Assessment";1993:Longmans
2) S.B. Akers, “ Binary decision diagrams”, IEEE Transactions on Computers, Vol. 27, No. 6, June 1978
3) R.E. Bryant, “Graph-Based Algorithms for Boolean Function Manipulation”, IEEE Transactions on Computers, Vol. 35, No. 8, 1986
4) A Villemeur, " Reliability, Availability, Maintainability and Safety Assessment", Wiley, Chichester, 1991
5) D.S. Nielsen, “The Cause/Consequence Diagram Method as a Basis for Quantitative Accident Analysis”, Danish Atomic Energy Commission, RISO-M-1374, May 1971
6) D.S Nielsen, B. Runge, “Unreliability of a Standby System with Repair and Imperfect Switching” IEEE Transaction on Reliability, Vol. 23, pp. 17-24, April 1974.
7) D.S. Nielsen, O. Platz, B. Runge, “A Cause-Consequence Chart of a Redundant Protection System”, IEEE Transactions on Reliability, Vol. 24, No. 1, April 1975.
8) Nielsen D.S, “Use of Cause-Consequence Charts in Practical Systems Analysis”, Reliability and Fault Tree Analysis, SIAM, 1975, p849-880
9) R.M Sinnamon, J.D Andrews, "Improved Efficiency in Qualitative Fault Trees", Proceedings of Esrel'95 Conference, June 1995.
10) Y. Dutuit, A. Rauzy, "A Linear-Time Algorithm to Find Modules of Fault Trees", IEEE Transactions on Reliability, Vol. 45, No. 3, 1996 September

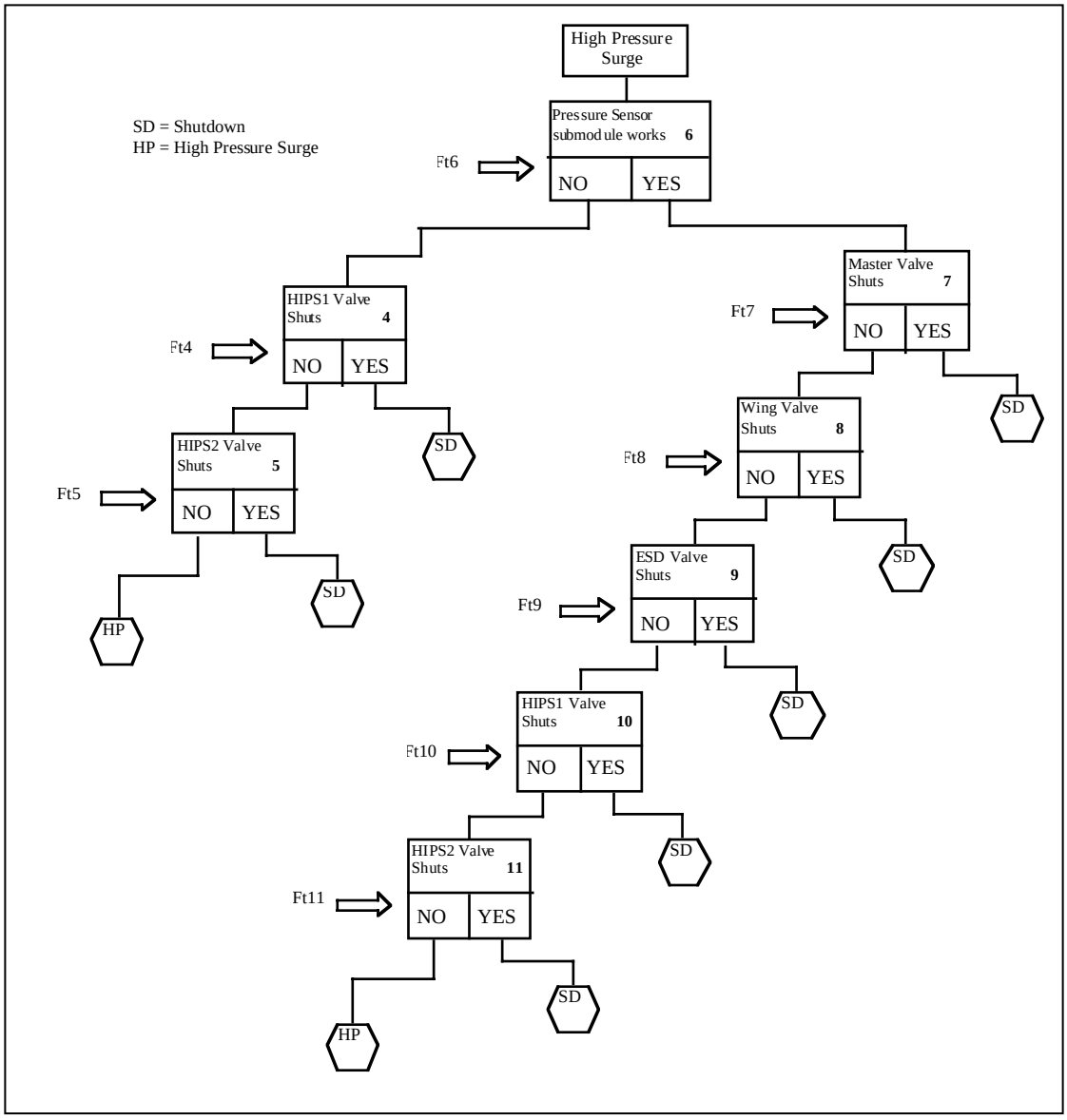


Figure 11 Reduced Cause-Consequence Diagram for HIPS System

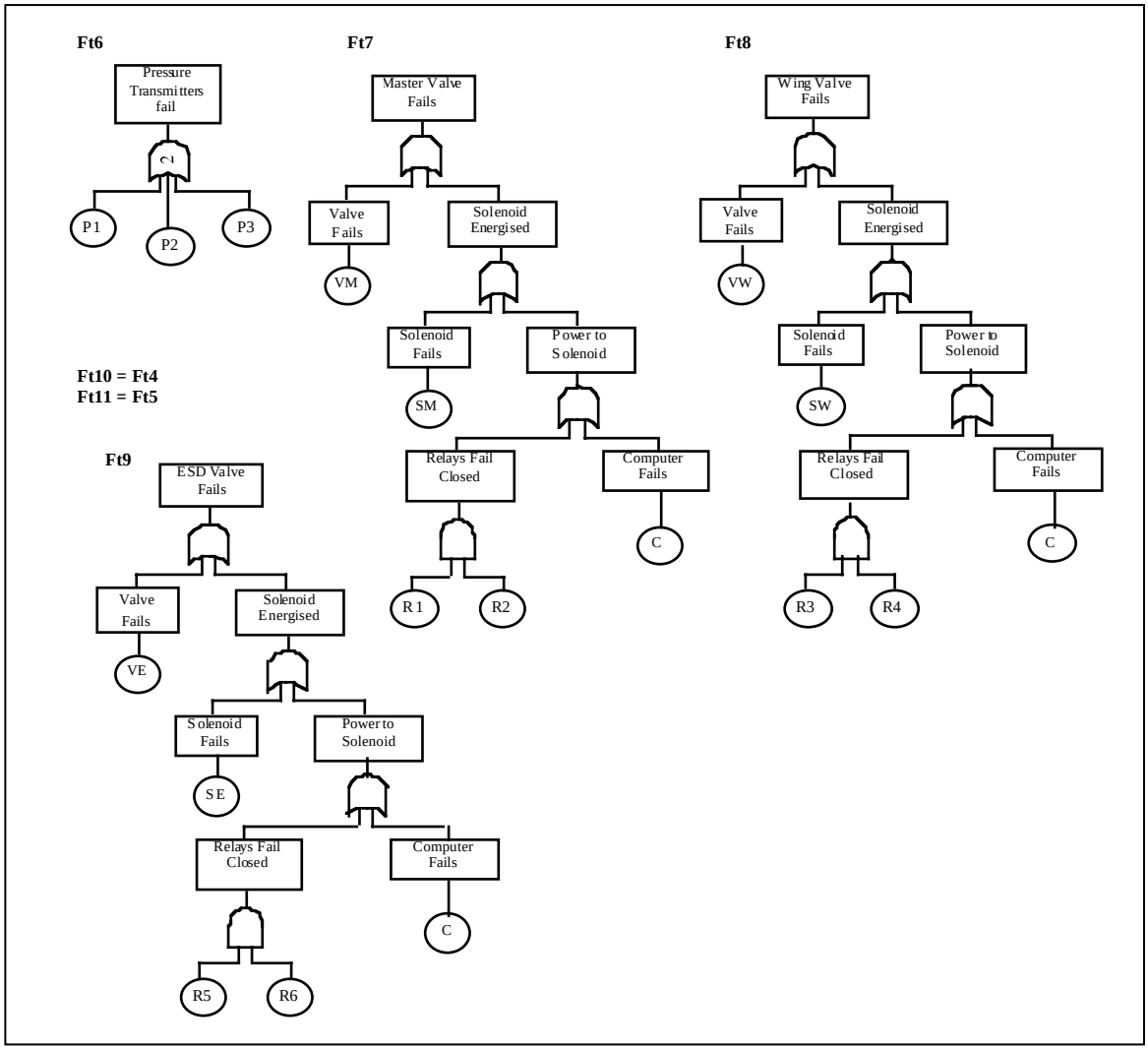


Figure 12 Fault Trees Ft6-Ft11 for figure 11

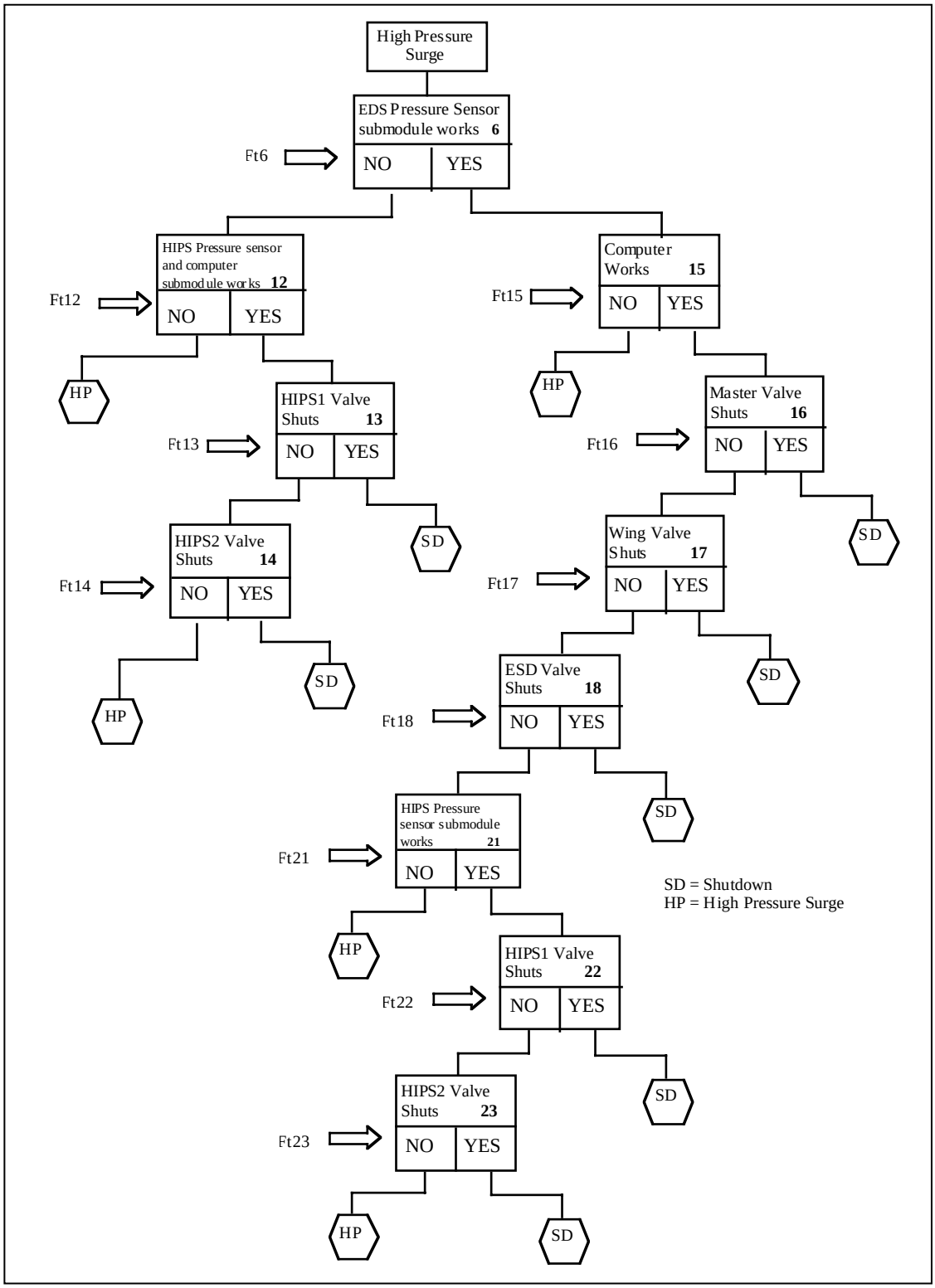


Figure 13 Final Cause-Consequence Diagram for the HIPS System

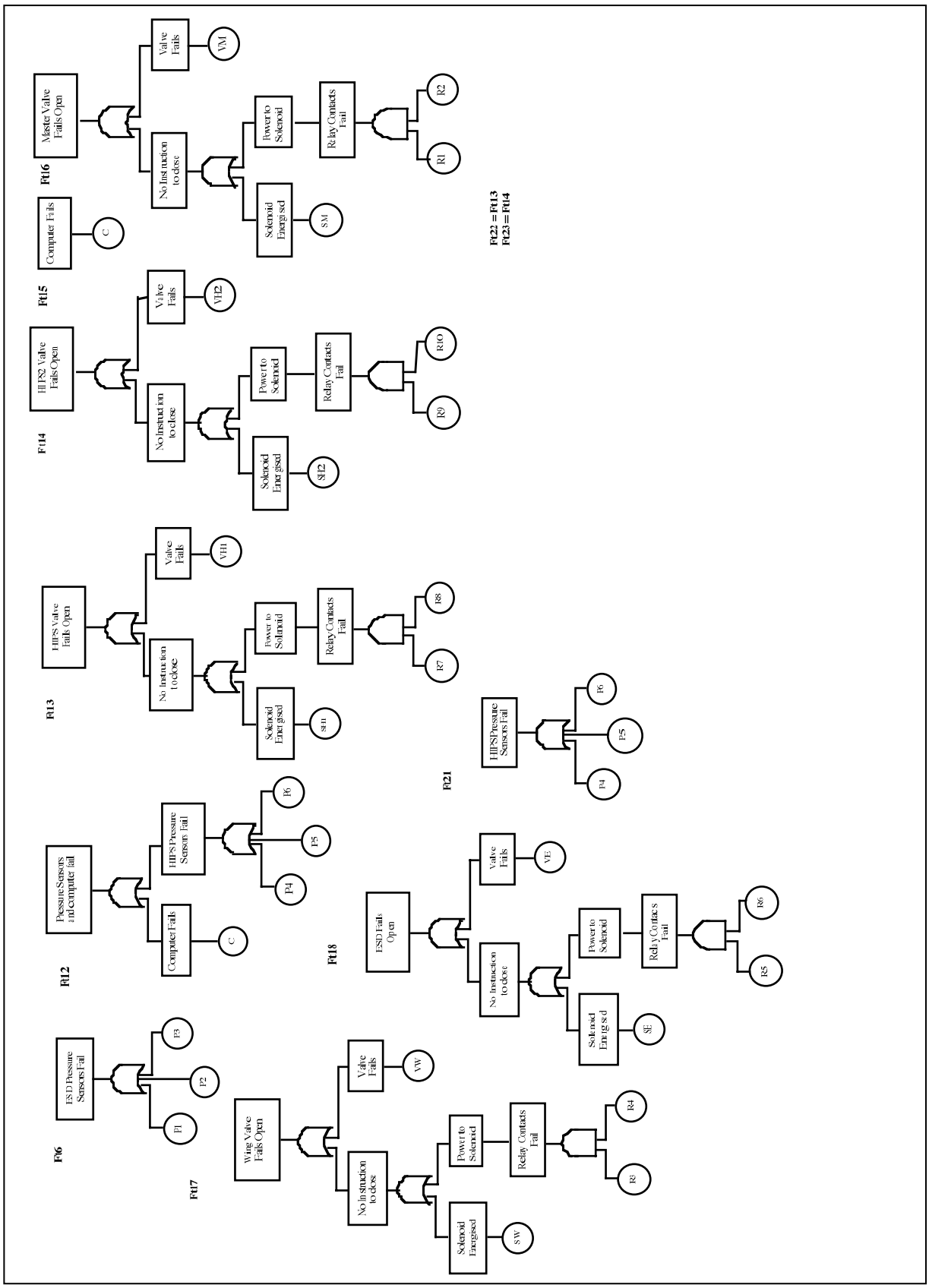


Figure 14 Fault Trees for figure 13