ELSEVIER

Reliability Engineering and System Safety 76 (2002) 1–10

RELIABILITY ENGINEERING & SYSTEM SAFETY

www.elsevier.com/locate/ress

Automated multiple failure FMEA

C.J. Price*, N.S. Taylor

Department of Computer Science, University of Wales, Aberystwyth, Ceredigion, SY23 3DB, UK

Received 18 April 2001; accepted 18 October 2001

Abstract

Failure mode and effects analysis (FMEA) is typically performed by a team of engineers working together. In general, they will only consider single point failures in a system. Consideration of all possible combinations of failures is impractical for all but the simplest example systems. Even if the task of producing the FMEA report for the full multiple failure scenario were automated, it would still be impractical for the engineers to read, understand and act on all of the results.

This paper shows how approximate failure rates for components can be used to select the most likely combinations of failures for automated investigation using simulation. The important information can be automatically identified from the resulting report, making it practical for engineers to study and act on the results. The strategy described in the paper has been applied to a range of electrical subsystems, and the results have confirmed that the strategy described here works well for realistically complex systems.

Keywords: Failure mode and effects analysis; Qualitative simulation; Multiple failures

1. Introduction

Automotive engineers are under increasing pressure to produce correct, safe designs for electrical systems in shorter time frames. The electronic complexity of vehicles is also increasing, making it ever more difficult to ensure that electrical systems meet their design requirements. Whereas 30 years ago a vehicle might have contained a few fuses and a collection of simple wiring, modern cars contain many subsystems with complex functionality, based around electronic control units (ECUs). They can contain so much electrical and electronic equipment that two batteries are needed to power the vehicle electrics. Advanced features such as car area networks and drive-by-wire make the task of simulating electrical systems progressively more challenging.

As complexity of electrical systems has increased, it has become more difficult for designers to comprehend all the possible implications of component failures on a design. Failure mode and effects analysis (FMEA) [5,15] is a design analysis discipline that considers the effects of any failure in a design, and identifies the more serious problems as areas where the design may need to be improved. This might be done either by adding redundancy to the design, or by reducing the likelihood of failure through using more reliable components. FMEA is conventionally carried out by a team of engineers. They consider each possible failure in turn, and decide what the effects of that failure would be. It is a labour-intensive, time-consuming, tedious, error-prone activity. For the most part, only the effect of single failures is considered, as there is not enough time to consider a meaningful number of multiple failures. Typically, the engineer will produce an FMEA report covering all single point failures and a few significant combinations that are heuristically identified.

Previous papers by the authors and their colleagues [9,12,13] have described techniques for the automated generation of design FMEA reports for electrical systems. The emphasis of the previous papers has been on automating and shortening the FMEA process as automotive engineers already practise it. That has meant concentrating on single failure FMEA. The resulting software, AutoSteve [11], is now a commercial product, and has been adopted as part of their design process by major automotive companies.

This paper takes automated FMEA a step further, automating the work of producing an FMEA report containing an analysis of the effects of significant multiple failures as well as all single failures. This is impractical for an engineer to achieve by performing FMEA without automated help. Automatically generating FMEA results means that a great many more failure combinations can be explored, but even

* Corresponding author. Fax: +44-1970-622-455.
E-mail address: cjp@aber.ac.uk (C.J. Price).

0951-8320/02/$ - see front matter
PII: S0951-8320(01)00136-3

with an automated tool, there are two significant problems to overcome:

- Generating all combinations of failures is not a feasible option for large circuits. For n possible failures, the single failure case is linear with n, but the multiple case gives approximately 2^n possibilities. How can the best combinations to explore be selected?
- Engineers need to understand and act on the results produced by an automated FMEA system, but they cannot be expected to examine a report containing effects details for many thousands of multiple failures. Even for cases where there are few enough combinations of failures to be able to simulate them all, there are still too many to expect the engineers to be able to consider all of the results to decide whether any action needed to be taken because of them. How can the important results of multiple failure FMEA be selected?

This paper presents solutions to both of these problems. It describes how to identify which failures should be considered, and shows how the information presented to the engineers can be reduced to manageable proportions.

2. Automated FMEA

AutoSteve is an automated electrical design FMEA report generation system. It performs single failure FMEA based on simulation of a circuit design, producing a textual report with appropriate content for an engineer to understand. It achieves this by performing qualitative simulation [2,18] of good and faulty versions of a circuit design, then it abstracts the results, and generates an output report from the difference between the effects of the good and faulty cases. This section of the paper outlines the process of producing a textual FMEA report in AutoSteve.

2.1. Qualitative simulation

Qualitative simulation reasons about circuit activity without any detailed knowledge of values such as the voltage drop along a wire. The main intuition behind qualitative simulation is that much of the reasoning done by engineers is done at a qualitative level. Deciding the behaviour of vehicle circuits can mostly be done at the level of presence of current flow, rather than needing to calculate the exact current to several decimal places. The qualitative simulator does this in much the same way that an engineer might, for example: *This switch is connected to ground, therefore that input to the ECU is active. Therefore this output from the ECU will be active, and so the relay will close and the lamp will be connected to supply and to ground and will light.*

The main advantages of qualitative simulation over quantitative simulators such as SABER [7] are that qualitative simulation does not need the detailed test-based parameter information for components that a numerical simulation would demand. This is especially useful early in the design life cycle, where exact values for resistors are not known. Where exact calculations are necessary, that need can be highlighted by a qualitative simulation, and explored in more detail later in the design life cycle. A second advantage is that qualitative simulation is very efficient—this becomes significant when the system is performing millions of repeated simulations in order to carry out multiple failure FMEA.

Qualitative simulation enables FMEA to be performed very early in the design lifecycle, as soon as a circuit representing the system being analysed can be drawn, with significant reduction in the amount of effort needed. This can mean very early detection of possible problems, while they can still be corrected at low cost.

A single qualitative description can cover many real components (for example, only one switch description might cover many similar types of switches), and so qualitative descriptions are highly reusable.

The description of component behaviour that is needed for each type of component will have three separate aspects:

- *Terminals*: Terminals are the inputs and outputs for the component. They are the points where other components can connect to this component.
- *Internal topology of component*: The functionality of the component is determined in terms of links between terminals. These links can include logical resistors whose resistance value can change depending on the state of other parts of the component.
- *Dependencies*: Dependencies define how the values of the internal resistors of a component change as the state of the other parts of the component change.

Example behaviour for a switch: A simple switch would have two terminals. The terminals can be regarded as joined by a resistor whose value depends on the state of the switch. When the switch is open, then the resistor has infinite resistance. When it is closed, the resistor has zero resistance.

Example behaviour for an open relay: An open relay is composed of a coil and a switch, where the state of the switch depends on the state of the coil. When current flows through the coil, the switch is closed otherwise it is open. Such a relay has four terminals, two to the coil, and two to the relay switch. The coil will be a resistor with a fixed load, and the value of the switch resistor will depend on whether current is flowing through the coil. When the state of the coil is Active, i.e. current is flowing through it, then the value of the switch resistor is zero (because the switch is closed). When the state of the coil is Inactive, i.e. no current is flowing through it, then the value of the switch resistor is infinite (because the switch is open).

When the structure of a circuit is drawn within an electrical CAD tool, a netlist can be extracted and used with the component descriptions to simulate the circuit. The underlying ability of the simulator is to calculate where current is

flowing through a network of resistors [6]. Given a circuit to simulate and an initial state for each component in the circuit, the simulation controller will perform the following steps:

1. Build a network of resistors from knowledge of the components, their states, and the connections between components.
2. Pass the network of resistors to the network analyser, and receive back details of where current is flowing in the network.
3. Use the details of the current flow to identify any component whose internal state has changed.
4. If any components have changed state, repeat from Step 1, otherwise terminate.

A much more detailed description of this process is given in [16]. Instead of using dependency descriptions, the behaviour of a complex component can be provided as a state-chart. This facility makes it much easier to describe the behaviour of complex components. A good example of such a component is an ECU within a central door-locking circuit, where the ECU might be required to detect that the circuit was locking the doors, and reset all the doors as unlocked if the locking process was not completed within a few seconds. To describe the behaviour of this component as a set of dependencies between resistors takes several hundred lines of dependency expressions, whereas it can be described much more compactly as a state-chart containing a small number of linked boxes.

The result of qualitatively simulating a circuit is a changing set of values for each component in the circuit as the inputs to the circuit (switches, sensors, ECU states) are changed. This facility provides answers to questions such as: '*What happens when I turn the key clockwise in the lock on the driver's door?*'

As well as correct behaviour for a component, the user can describe behaviour under failure conditions. For example, failures for an open relay might be 'stuck at open' (the relay switch does not close when the coil is powered), 'stuck at closed' (the relay switch is closed whether the coil is powered or not) and 'burnt out' (the coil is shorted and no longer works). The component behaviour description can be enhanced with an extra set of information that describes the behaviour of the component under failure conditions.

For each failure, the different dependencies that operate under fault conditions must be described. For the relay failed *stuck at open* and *burnt out*, the value of the switch resistor is infinite, irrespective of the value of the coil resistor. For the relay failed *stuck at closed*, the value of the switch resistor is zero, irrespective of the value of the coil resistor.

The correct version of a component can be replaced in a simulated circuit by a faulty version, and so it is also possible to simulate the behaviour of the circuit when failures exist in the circuit.

2.2. Interpreting simulation with functions

The simulation is a qualitative DC simulation. It calculates the state of each component in the circuit after each change in inputs to the circuit. If a circuit has several hundred components, a report of all component states for each state change would overwhelm the engineer with details. For a typical design, the significant information is not whether each component of the circuit is in its correct state, but whether the circuit is achieving its intended functions. Most circuits have a small number of intended functions. For example, in a car central locking system, the functions might be:

- doors locking;
- doors locked;
- doors deadlocked;
- doors unlocking;
- doors unlocked.

Whether a particular function is occurring can be recognised from the state of very few of the components in the circuit. The functions of a circuit are very reusable between different versions of the same vehicle subsystem, and just need to be linked to the state of relevant components in the implemented circuit for this vehicle. So, for example, the *doors locking* function can be recognised as occurring when one or more of the door motors are ACTIVE, with current flowing in a FORWARD direction. The *doors locked* function can be recognised from the state of the door lock sensors.

The use of functions in the AutoSteve system is significant, because it enables the production of a summary of the state of a circuit in very few details, but in a principled manner. It also provides a method for generating consistent, concise textual descriptions of potential failures effects (see Table 1, where all text has been generated by AutoSteve from the results of simulation), and occurrence and detection values for each possible system failure. As well as automated FMEA, it provides a basis for other kinds of design analysis such as sneak circuit analysis and design verification [8,14].

2.3. Generating an FMEA report

Qualitative simulation has provided a method of simulating circuit behaviour without needing to know exact values for resistors. The results of that simulation can be summarised succinctly by interpreting the results of qualitative simulation with functions. This can be performed for the correctly working circuit, and for versions of the circuit containing components with faulty behaviour.

Table 1
Extract from FMEA report with single failures (The results shown are an extract from the 271 rows of results shown for the door-lock schematic. The meaning of the items are: Item/Fn: this is an identifying number generated for each of the 271 results. Potential failure cause: the component failure, which has caused the failure mode. Potential failure mode: the effect of the component failure on the operation of the system. Potential failure effect: the effect of the component failure on the user of the system. Severity: the significance of the effect on the operation of the car, on a scale of 1–10, where 10 is most severe. Occurrence: the likelihood of the component failure occurring during the lifetime of the system, on a scale of 1–10, where 1 is 1:1,500,000, and 10 is 1:1. Detectability: the likelihood that the failure will be detected, on a scale of 1–10, where 1 means it is immediately detectable)

Item/Fn	Potential failure cause	Potential failure mode	Potential failure effect	Sev	Occ	Det
(23)	The component UNLOCK_RELAY has failure switch stuck at contact2	For the first time, the 'doors unlocking' function was achieved. Finally, regardless of any event change, the 'doors locked' function was never achieved, and the 'doors unlocked' function was always achieved	Doors started unlocking unexpectedly. Doors unlocked unexpectedly. Doors failed to lock	6	3	2
(24)	The component DEADLOCK_RELAY has failure coil blown	When DRIVER_KEY_SWITCH was set to lock (3) the 'doors locked' function was achieved unexpectedly. Also, when DRIVER_KEY_SWITCH was set to neutral (4) the 'doors locked' function was achieved unexpectedly	Doors locked unexpectedly	6	2	4
(25)	The component DEADLOCK_RELAY has failure switch stuck at contact1	When DRIVER_KEY_SWITCH was set to lock (3) the 'doors locked' function was achieved unexpectedly. Also, when DRIVER_KEY_SWITCH was set to neutral (4) the 'doors locked' function was achieved unexpectedly	Doors locked unexpectedly	6	3	4

These facilities can be used to generate an automated FMEA report in the following manner:

The computer generates a simulation for the correctly working circuit, stepping through the different possible states of the circuit (e.g. turn the key clockwise in the driver's door, turn the key anti-clockwise in the driver's door, turn the key clockwise in the passenger's door, etc.).

The results of the simulation are abstracted, giving a summary of the functions occurring in the different states (e.g. when the key is turned clockwise in the driver's door, the *doors locking* state is entered, and after 0.5 s the *doors locked* state is entered).

Next, each possible failure for each component is applied in turn to the circuit. The simulation is repeated but the correct version of the component is replaced with a version of the component exhibiting the chosen failure. For example, if a relay supplying power to one of the motors failed open, then when the key is turned clockwise in the driver's door, the *doors locking* state is entered. However, not all of the doors become locked and so, after 2 s, the *doors unlocking* state is entered and after a further 0.5 s, the *doors unlocked* state is entered.

Following the production of each simulation with a failure applied, the functions that occurred in the correct version of the working circuit are compared with those that occurred when the failure was applied. This comparison leads to a short effects report. Some example results generated by AutoSteve are shown in Table 1.

The single failure automated FMEA tool described in this section has been used by engineers in the automotive industry for several years, and has proved to be a very efficient way of generating a single failure FMEA report. The engineers examine the results of the automated FMEA, and can have them depicted graphically on the schematic displayed within the CAD tool they normally use. In this way, the tool meets the main aim of design FMEA by helping the engineers understand their circuit designs and the implications of component failures. However, the increased complexity of car circuitry means that many of the more interesting effects occur as a result of multiple failures. Section 3 describes how a practical automated multiple failure FMEA report can be produced.

3. Handling multiple failures

The circuit shown in Fig. 1 gives a schematic for a typical central door locking subsystem. This example is used in the remainder of this paper. It shows the circuitry for a four-door vehicle together with associated actuators and security ECU.

3.1. Illustrating the problem

The example central door locking schematic has 271 single point failures on 139 components, and so the standard single failure FMEA report would have 271 entries. This takes 2 min to generate on a Sun UltraSparc 2 computer. It should be noted that this report on the implications of single failures will include consideration of dependent or consequent failures, e.g. where a short circuit on a wire causes a fuse to blow. This means that some very common multiple

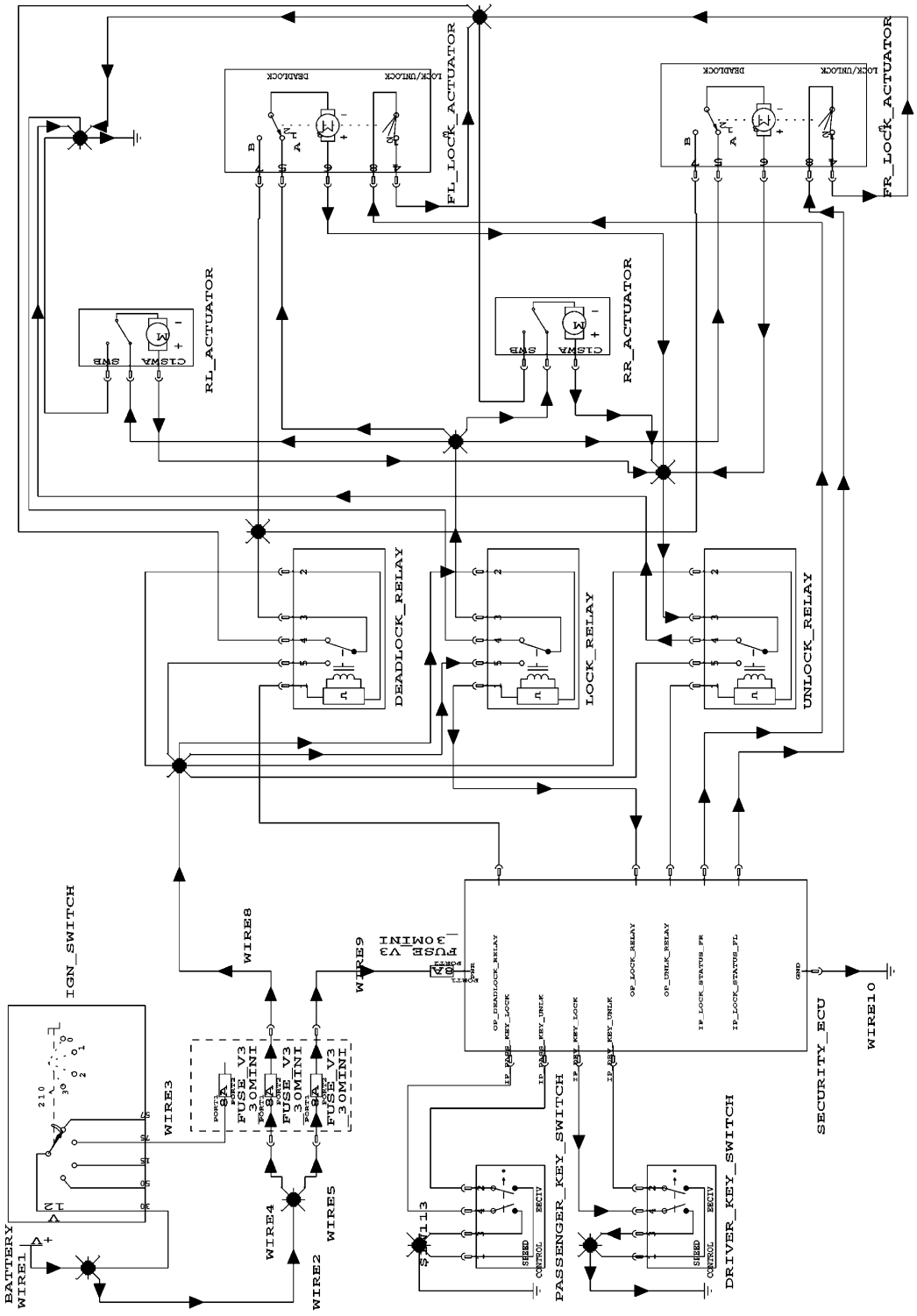

Fig. 1. Central door-locking schematic.

Table 2
Extract from FMEA report including multiple failures

Item/Fn	Potential failure cause	Potential failure mode	Potential failure effect	Sev	Occ	Det
(1628)	The component IGN-SWITCH has failure switch stuck at start and component FR_LOCK_ACTUATOR has failure switch stuck at position b	Regardless of any event change, the 'doors locked' function was never achieved	Doors failed to lock	6	1	4
(1629)	The component IGN-SWITCH has failure switch stuck at start and component FR_LOCK_ACTUATOR has failure lock switch stuck open,…	For the first time, the 'doors unlocking' function was achieved. Finally, regardless of any event change, the 'doors locked' function was never achieved, and the 'doors unlocked' function was always achieved	Doors started unlocking unexpectedly. Doors unlocked unexpectedly. Doors failed to lock	6	1	2
(1630)	The component IGN-SWITCH has failure switch stuck at start and component FR_LOCK_ACTUATOR has failure lock switch stuck closed.	Regardless of any event change, the 'doors locked' function was never achieved.	Doors failed to lock.	6	1	4
(1631)	The component IGN-SWITCH has failure switch stuck at start and component FR_LOCK_ACTUATOR has failure Motor Blown	Regardless of any event change, the 'doors locked' function was never achieved	Doors failed to lock	6	1	4

failures will be dealt with automatically when considering the single point failures.

Some of the challenges of multiple failures can be illustrated by extending the single failure system to handle pairs of failures. The 271 single failures can be combined to form 36,585 unique pairs of failures; this is $f(f-1)/2$ where f is the total number of single point failures in the circuit. However, some of those pairs are mutually exclusive. For example, a single wire cannot simultaneously be shorted to battery and shorted to ground. But nevertheless, excluding mutually exclusive pairs of failures has little effect on the number of failures to be explored, leaving 36,370 pairs of failures to be examined for the central door locking schematic. Calculating the consequences of each pair of failures would be impractical for an engineer working without automated assistance, but is perfectly reasonable for an automated system.

Table 2 shows examples of multiple failure reports.

Generating the FMEA report for all pairs of failures takes AutoSteve approximately 21 h, but presents a new problem. The reason for producing an FMEA report is for engineers to understand the implications of failures on the circuit, thus identifying improvements that can be made to increase the circuit's safety and reliability. However, it is impractical for engineers to examine each of 36,370 entries in an FMEA report, and to ensure that all of the important details have been understood. Even if they did attempt to inspect every single entry in the report, they would be likely to miss any important information in the mass of details.

The problem is more serious for the case of higher order multiple failures. The total number of possible failure combinations for a circuit are of the order of 2^n, where n is the number of single-point failures that could occur in the circuit. For the central door locking circuit, with 271 possible failures, there are around 10^{81} ways in which those failures might be combined. However, as with the double failures, a number of these combinations can be discarded because they are combinations of contradictory faults on the same component. The number of combinations that can be discarded depends upon the number of possible failures for each component. Even if each of the 139 components had only one possible failure, there would be approximately 10^{42} combinations to simulate and examine.

The following two subsections describe the approach we have developed to focus the multiple FMEA report on the significant failures. The first stage, described in Section 3.2, is to simulate only those multiple failures that are below a specified failure threshold. The second stage, described in Section 3.3, is to analyse the resulting FMEA report and remove all multiple failure results that could have been inferred by combining the single failure reports.

3.2. Limiting the FMEA generation to likely combinations

Let C be the set of components in the circuit being analysed. $|C|$ is the number of components in C.

$$C = \{c_1, c_2, \ldots, c_{|C|}\}$$

Let $\mathrm{Op}(c)$ be the set of operational modes of component c, comprised of the correct working behaviour of the component, plus each possible failure mode for the component.

All possible combinations of failures for the circuit can be characterised by the Cartesian product of the sets of operational modes:

$$Op(c_1) \times Op(c_2) \times \cdots \times Op(c_{|C|})$$

Section 3.1 observed that for the central door-locking circuit with 139 possible single point failures, there are at least 10^{42} multiple failures to be simulated. It is not computationally feasible to examine all failure combinations.

A sensible strategy is to examine the most likely combi-

nations of failures. The number of failure combinations explored can be reduced by introducing a threshold T for the probability of a failure combination occurring, and not exploring faults with probability less than T.

If $p(\text{Mode}(c_i, m))$ is the probability that component c_i will be in operating mode m during its planned lifetime, and $M(c_i)$ is the selected operating mode of component c_i then an overall probability for any failure combination can be generated.

$$\prod_{i=1,\ldots,|C|} (p(\text{Mode}(c_i, M(c_i))))$$

The probability of the failure combination can be compared against a specified threshold, for example 10^{-9}. Any failure combination with a failure probability above the threshold is simulated. Because the probabilities are only being used to select which failure combinations are explored, not to compute reliability values, these approximations are reasonable as long as they do not significantly distort the probability for a combination of failures. One place where this might happen is for dependent failures (where one failure causes another failure to occur). Such failures should be predicted by simulation of the more likely simpler failure, and so will not fall out of consideration where the probability of the failure combination is below the threshold.

On a practical note, failure probabilities are not generated for all combinations, for the same reason that simulation is not done for all combinations. Single failures are generated first, then pairs of failures. Triples and higher failure combinations are only generated from pairs of failures which are still above the threshold.

For the central door-locking circuit, a threshold of 10^{-9} generates 8506 combinations from the 271 possible failures on 139 components. It takes AutoSteve about 1 h to simulate and produce an FMEA report for each of these combinations. This is less than the number of pairs of possible failures, as many of the failures have an occurrence value of 1 (a probability of 1.5×10^{-6}), and so pairs of such failures fall below the threshold.

3.3. Pruning the results by interest

While there are large numbers of multiple failure cases, many of them provide the engineer with little extra information. Most of the results could have been inferred by combining the single failure reports. For example, if the Lock Relay fails open, then the *doors locking* and *doors locked* functions will fail to occur when expected. Similarly, if the wire between the Security ECU and the Lock Relay fails open circuit, then the *doors locking* and *doors locked* functions will fail to occur when expected. When these failures occur together and the *doors locking* and *doors locked* functions fail to occur, then the multiple failure result is of little interest to the engineer.

For pairs of failures, the FMEA report can be pruned without reducing the significant information that is presented to the engineer, in the following way.

Let the fault symptoms for a set of component failures F be a set of function differences $D[F]$, where a function difference is either the unexpected operation of a function, or the absence of expected operation of a function. For two failures x and y, the multiple failure $\{x, y\}$ should not be reported if:

$$(D(\{x, y\}) = D(\{x\}))$$

$$\text{or } (D(\{x, y\}) = D(\{y\}))$$

$$\text{or } D(\{x, y\}) = D(\{x\}) \cup D(\{y\})$$

Discarding the uninteresting cases in this manner for the central door locking schematic reduces the FMEA report for all pairs of failures to a size where it is practical for engineers to study and understand the results. For the given example, it prunes the 36,370 possible combinations of pairs of faults, leaving 3056 interesting pairs of failures. This leaves few enough interesting combinations that the engineer can consider the implications of each of them. The software has done the work of generating and simulating the results of all pairs of failures, but the results have been reduced to a degree where they can be assimilated by human beings.

The pruning criteria described above can be extended to deal with the case of combinations of three or more failures, and can then be applied to prune the results of the multiple failure FMEA generated up to a threshold described in Section 3.2.

Let FC be the set of failures being examined. FC should not be reported if

$$\exists\{\text{fc}_1 \ldots \text{fc}_n\} \wedge (\{\text{fc}_1 \ldots \text{fc}_n\} \subset \text{FC})$$

$$\wedge((D(\text{FC}) = D(\{\text{fc}_1 \ldots \text{fc}_n\}))$$

$$\vee(D(\text{FC}) = \underset{i=1\ldots n}{U} (D\{\text{fc}_i\})))$$

This pruning reduces the number of FMEA report entries for the door locking example with a threshold of 10^{-9} from 8506 reports to just 734. This figure is composed of 463 interesting multiple failures plus the 271 single failures. This is certainly a small enough number of failure reports to expect an engineer to be able to examine them all. Choosing a higher threshold increases the number of combinations considered.

3.4. Effectiveness of these strategies

Table 3 shows how the numbers of multiple failures within the threshold increases as the threshold is raised for the central door locking schematic. The door locking

Table 3
Effect of threshold level on multiple failures generated for the central door-locking circuit

Threshold	Number of multiple failures within threshold	Multiple failures left after pruning	Percentage of failures pruned
10^{-9}	8506	734	91%
10^{-10}	26,196	2166	92%
10^{-11}	35,820	2997	92%
10^{-12}	36,364	2999	92%
10^{-13}	401,633	59,883	85%

schematic is representative of subsystems in the vehicle, and so is a good test of the effectiveness of the strategy. Comparable results have been obtained for other subsystems.

Table 3 also shows the effect of the pruning exercise. It can be seen that the application of thresholds and the selection of failures to report by interest are both needed to reduce the size of the produced report to a level that the engineers can absorb.

4. Implications of these results

4.1. Usefulness of multiple failure FMEA

The multiple failure FMEA strategy described in this paper has been implemented and used on a number of circuit designs. It is efficient enough for the largest subsystems within modern vehicles.

The pruning of the multiple failure results is heuristic in nature, and so it is sensible to ask whether it is also effective. The intention of the pruning strategy was to report on those multiple failures where the outcome for the multiple failures was different from what would have been predicted by examining the single failure results. In a well-designed circuit, there are not many such effects, and so it is not surprising that the pruning strategy is successful in concentrating the engineer's attention on the interesting multiple failures.

The most debatable aspect of the pruning is discarding multiple faults where

$$(D(\mathrm{FC}) = \underset{i=1\ldots n}{U} (D\{\mathrm{fc}_i\}))$$

In some cases, the combined effect is more severe than the effect of either single fault. In a windscreen wiping system, for example, losing all wiper functions is more severe than losing any one of the intermittent wipe, slow wipe and fast wipe functions. Such combinations of effects are not reported in the given strategy. It is true that the engineer can infer them from the report on the single failures, but perhaps a better strategy would be to explicitly include them in the multiple failures FMEA report. Another possibility is that they could be fed into an automated fault tree analysis (FTA) for the circuit, concentrating on identifying important failure conditions.

4.2. Applicability of multiple failure FMEA

This paper has described the automated generation of a multiple failure FMEA report based on a simulation of the underlying structure of the system. This has been shown for an automotive electrical system, but the technique is of wider applicability.

The technique should work for any system where there are compositional models for components of the system. Typically, such models will be independent of the device being simulated. This condition has been expressed as 'no function in structure' [3]. A simple example of violating this condition would be to model a lamp in a car as being lit whenever a specific switch is closed. This would make it impossible to use simulation to predict the effect of an open circuit in the wires between the switch and the lamp. Simulation of such failures would erroneously predict that the lamp would still light in the failure situation.

Digraph models [17] do not strictly meet the condition of compositionality. They enable the prediction of loss of functionality by tracing what functionality is dependent on the failed component. They have a good deal of information about the causality of the overall system—causality which can be violated by failures, especially complex multiple failures. The addition of more information about failure causality, as Vaidhyanathan and Venkatasubramanian do for use in HAZOP, might make it possible to use such models to generate a selective multiple failure FMEA and prune it in the way described in this paper.

As well as the qualitative simulator described earlier, a version of the FMEA generation software has been built which uses the commercially available SABER numerical simulator. SABER is typically used for modelling mechanical and electrical systems, and uses MAST models [7] to represent component behaviour. The results using the SABER simulator match the results using the qualitative simulator, and allow application of the system to a wide range of electro-mechanical devices.

4.3. Practical considerations of multiple failure FMEA

The work limiting multiple failure FMEA generation to likely combinations described in Section 3.2 depends on the availability of the probability that a component will be in a specific operating mode during its lifetime. It must be considered whether the necessary probabilities are available and whether they should be used in this way. If good reliability data is available for components, then the probabilities of occurrence of single failures can be provided quite accurately, and can be used directly.

In automotive applications, the reliability data (from warranty reports) is notoriously inaccurate because of the uncontrolled environment in which warranty reports are compiled. However, where specific component failure

rates are not available, they can be approximated by employing the Occurrence value used for that failure on that type of component in the FMEA generation process. Several standard schemes for performing FMEA, such as the QS-9000 standard promoted by the three largest U.S. motor manufacturers [1,4] give a failure rate for each value of 1–10 on the occurrence scale, and each type of component failure will have an occurrence value assigned to it. For example, a wire shorting open might have an occurrence of 1, and an occurrence of 1 might correspond to a 1.5×10^{-6} probability of failure during the planned vehicle lifetime.

These approximate failure probabilities can then be used to compute an approximate probability of failure for each combination. The resultant failure probabilities are not accurate, but do not need to be. As long as they provide a reasonable approximation, then they are adequate for the intended purpose of focusing the analysis on the more important multiple failure combinations.

4.4. Using the information for diagnosis

One motivation of this research has been to use design information as the basis for creating diagnostic systems. The automated FMEA describes the failures and effects precisely and consistently for the entire report. This consistency makes it easy to rearrange the information in the FMEA report so that all failures that cause the same effect can be brought together.

Using the results of the analysis on each electrical subsystem in a vehicle, a rudimentary diagnostic tree can be constructed. By identifying subsystems and their deviating functions, a set of candidate components can be generated; the list can be ordered according to occurrence value.

This paper describes a two-stage approach to generating multiple failure FMEA reports. The generation stage provides a practical solution to managing the number of multiple failure combinations to simulate. Whilst pruning is an appropriate strategy for FMEA reports, it does not provide a practical benefit for diagnosis.

Two advantages of using design information to construct a diagnostic system are

- the models used for design are reused, reducing the time and effort involved in constructing diagnostic systems,
- it leads to efficient runtime systems, because the detailed model analysis is already complete.

The implications of using multiple failure FMEA information for diagnosis are discussed further in [10].

4.5. Relationship between multiple failure FMEA and fault tree analysis

FTA can be used to calculate detailed reliability values such as mean time before failure. However, some companies use the early steps of FTA to assess the effect of multiple failures. They identify an overall condition to be avoided, such as the car doors deadlocking when they should not. The results of the single failure FMEA can be used to help build a fault tree, which describes the dependencies between the failures.

As vehicles become more complex, and more reliant on software, this can be a dangerous practice. Software within ECUs is often used to mitigate failures. Such software can interact with a further failure in multiple failure situations in ways, which were not foreseen by the designer.

Multiple failure FMEA can give the correct answer for such situations by performing simulation which includes the failure mitigating behaviour of the ECU. This means that the fault tree built from those results is likely to be more accurate because it correctly analyses the interaction of the failures and the failure mitigating behaviour of the system. The correct results provided by multiple failure FMEA also give a more realistic basis if overall reliability values are desired.

5. Conclusions

The multiple failure FMEA makes possible the analysis of large numbers of failure combinations, while presenting the engineer with only those combinations which have ‘interesting’ results. This facility provides a much wider safety analysis for electrical systems than was previously feasible.

In order to achieve this, it was necessary to be able to automate the generation of results, to be able to rank failure combinations so that the more likely combinations were explored, and to be able to prune the results produced so that the engineer can assimilate the important information. Answers to each of these problems have been described. They have been tested on realistic subsystem designs and found to be both practical and useful.

Acknowledgements

This work has been carried out by the University of Wales Aberystwyth and by FirstEarth Limited. The University have been supported in this work by Ford Motor Company, and by the UK Engineering and Physical Sciences Research Council (grants numbered GR/L20542 and GR/N06052).

References

[1] Automotive Industry Action Group, Potential Failure Mode and Effects Analysis (FMEA), 3rd ed., available from http://www.aiag.org.
[2] Davis R. Diagnostic reasoning based on structure and behavior. Artif Intell 1984;24:347–410.
[3] de Kleer J, Brown JS. A qualitative physics based on confluences. Artif Intell 1984;24:7–83.
[4] Ford Automotive Safety and Engineering Standards Office. Failure Mode and Effects Analysis Handbook. Ford Motor Co, 1995.
[5] Jordan W. Failure modes, effects and criticality analyses. In:

Proceedings of the Annual Reliability and Maintainability Symposium. IEEE Press, 1972. p. 30–37.

[6] Lee MH. Qualitative circuit models in failure analysis reasoning. Artif Intell 1999;111:239–76.

[7] MAST. Guide to Writing MAST Templates, Book 1, Release 5.1. Analogy Inc, 1999.

[8] McManus AG, Price CJ, Snooke N, Joseph R. Design verification of automotive electrical circuits. In: Proceedings of the 13th International Workshop on Qualitative Reasoning QR'99. Loch Awe, 1999.

[9] Price CJ. Function directed electrical design analysis. Artif Intell Engng 1998;12(4):445–56.

[10] Price CJ. Computer Based Diagnostic Systems. Berlin: Springer, 1999 chapters 5 and 6.

[11] Price CJ. AutoSteve: automated electrical design analysis. In: Proceedings of the ECAI-2000. Berlin, August 2000. p. 721–725.

[12] Price CJ, Hunt JE, Lee NH, Ormsby ART. A model-based approach to the automation of failure mode effects analysis. Proc Instn Mech Engrs, Part D: J Automobile Engng 1992;206:285–91.

[13] Price CJ, Pugh DR, Wilson MS, Snooke N. The flame system: automating electrical failure modes and effects analysis. In: Proceedings of the Annual Reliability and Maintainability Symposium. 1995. p. 90–95.

[14] Price CJ, Snooke N, Ellis D. Identifying design glitches through automated design analysis. In: Proceedings of the Annual Reliability and Maintainability Symposium. Washington, DC, January 1999. p. 277–282.

[15] Savakoor S, Bowles JB, Bonnell RD. Combining sneak circuit analysis and failure modes and effects analysis. In: Proceedings of the Annual Reliability and Maintainability Symposium. 1993. p. 199–205.

[16] Snooke N. Simulating electrical devices with complex behaviour. AI Commun 1999;12(1,2):45–59.

[17] Vaidhyanathan R, Venkatasubramanian V. Digraph-based models for automated HAZOP analysis. Reliab Engng Syst Saf 1995; 50:33–49.

[18] Readings in Qualitative Reasoning about Physical Systems. In: Weld D, de Kleer J, editors. Los Altos, CA: Morgan Kaufmann, 1990.