

Function-directed Electrical Design Analysis

C. J. Price

Department of Computer Science
University of Wales, Aberystwyth
Ceredigion, SY23 3DB, United Kingdom
cjp@aber.ac.uk

Abstract

Functional labels provide a simple but very reusable way for defining the functionality of a system and for making use of that knowledge. Unlike more complex functional representation schemes, these labels can be efficiently linked to a behavioral simulator to interpret the simulation in a way that is meaningful to the user. They are also simple to specify, and highly reusable with different behavioral implementations of the system's functions. This claim has been substantiated by the development of the FLAME application, a practical automated design analysis tool in regular use at several automotive manufacturers.

The combination of functional labels and behavioral simulator can be employed for a variety of tasks – simulation, failure mode and effects analysis (FMEA), sneak circuit analysis, design verification, diagnostic candidate generation – producing results that are very valuable to engineers and presented in terms that are easily understood by them. The utility of functional labels is illustrated in this paper for the domain of car electrical systems, with links to a qualitative circuit simulator. In this domain, functional labels provide a powerful way of interpreting the behavior of the circuit simulator in terms an engineer can understand.

Keywords: functional reasoning; qualitative reasoning; automotive applications; FMEA; sneak circuit analysis; design verification;

1. Introduction

Qualitative simulation based on the structure of a device or system has been a promising technology for many years¹⁹. A model of a system can be composed by describing the behavior of individual components, and then describing the structure that links the components together. Such models can be used to simulate the behavior of the complete system. Very few successful industrial applications of the technology have emerged, and one of the main problems in using qualitative reasoning for applications has been interpretation of the results of qualitative reasoning at a level which is relevant to potential users.

Several researchers have used a separate representation of overall system behavior, sometimes referred to as function, to interpret the simulation results. Schemes joining functional reasoning and qualitative reasoning^{15,4,3} have tended to be too complex for use in industry because it has proven impractical to expect engineers to construct such model descriptions in their daily work.

This paper describes a practical method for using functional knowledge to interpret qualitative simulation of electrical circuits at an appropriate level. A number of researchers have developed qualitative electrical simulators^{8,9,18} capable of identifying electrical activity in a circuit, but the simulation results provide too much information to be of direct use to engineers. Functional labels can be employed to guide the use of component-based qualitative simulation. This method has proven very effective for both design analysis and diagnosis tasks. Instead of specifying the overall behavior of a device or system, the

significant functions of the overall behavior are identified and named. These functional labels can then be linked to the state of one or more specific components in the device, in order to identify when a function is being achieved.

The linking of functional labels to qualitative simulation has been used in the latest version of the FLAME system^{13,11} to provide practical automated assistance for design analysis to automotive engineers. Design analysis tools described in the paper are in daily use in industry for automotive design analysis.

The structure of the paper is as follows:

Section two describes the features of the qualitative structural simulator used in the FLAME system, and explains how the results of such simulators are far too detailed to be used directly for diagnosis or design analysis.

Section three introduces functional reasoning, and shows how functional labels can be linked to the results of qualitative structural simulation and employed to abstract the results.

Section four details the use of the simulation results for a number of design analysis tasks and for diagnosis.

Section five examines the limitations inherent in the described linkage of functional labels and qualitative structural simulation. It also considers to what extent the described scheme can be used for similar tasks in other domains.

2. Simulating Electrical Circuits

This section describes the component-based qualitative reasoning system used in the FLAME system to simulate electrical circuits, and shows how it works for the simple headlight system illustrated in figure 1, where the driver can select either of two sets of headlights (main or dipped beam).

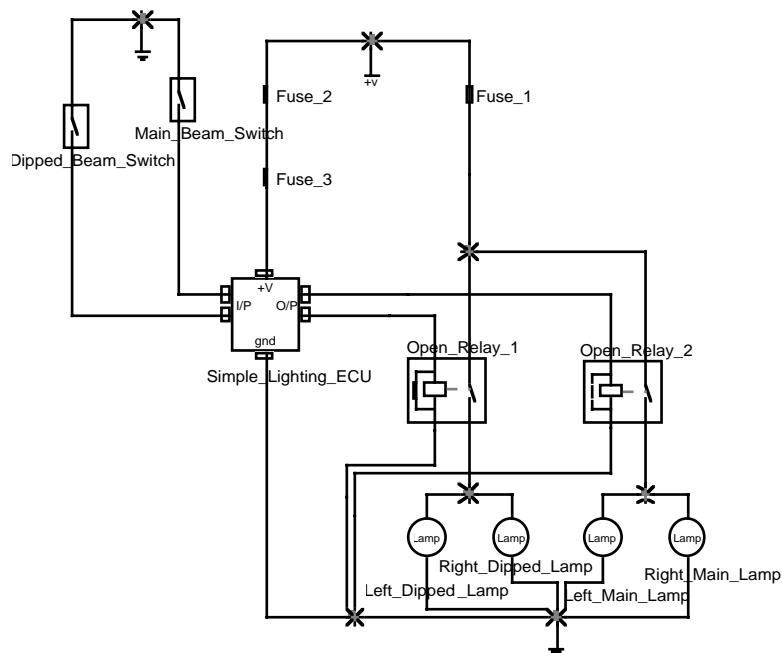


Figure 1: Simple headlight circuit

In order to persuade automotive engineers to use qualitative reasoning, it must be painless for them to build qualitative models of the circuits as they are being designed. This means that the qualitative models must use the same types of components that the engineers employ during design, and that component descriptions must be reusable.

The underlying algorithm performing reasoning about electrical flow only uses connectivity information, and the circuit topology used by this underlying algorithm will change as the state of components change. This has led to two levels of reasoning being employed.

Simulation controller (higher level reasoning): From the description of each component in the circuit, and knowledge of that component's state, generate a static network of connected electrical resistors representing the electrical connectivity of the circuit at this point in time. Change the topology of the network and resimulate whenever the state of a component changes.

Network analyzer (lower level reasoning): Evaluate the static resistive network, and assign new states to each component.

2.1 Simulation controller

The higher level reasoning employs component descriptions that describe the internal topology of a component in different states and under different failure modes. More detailed examples of component descriptions are given in Pugh and Snooke¹⁴. A component description consists of:

Terminals. Terminals are the inputs and outputs for the component. So, for example, an open relay has four terminals, two to the coil, and two to the relay switch. 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 internal resistors of a component change as the state of the other parts of the component change. If the open relay has two resistors, one for its coil and one for its switch, then one dependency might be that when the state of the coil was "Active" (current is flowing through it) then the value of the switch resistor is zero (the switch is closed).

External states. There are cases where the function of a component cannot be completely encapsulated, and the "no function in structure" rule has to be violated to obtain the desired behavior. External states make information about the component's internal state available to other components. Engineers are warned to use this feature only under certain conditions in order to ensure that their results are meaningful.

Failure modes. Topology and dependencies can be redefined for different failure modes of the component, so that the component acts appropriately when failures are being simulated. For example, for a "stuck open" failure of the open relay, the dependency between the coil and the switch might be redefined so that the switch resistor has infinite resistance no matter whether current is flowing in the coil or not.

The definition of components that has been described is powerful enough to represent most electrical components in a car, from a simple wire to the complicated behavior of a electronic control unit (ECU) within a circuit. Many of the components, such as wires, relays, and lamps, are reusable in many different circuits. ECU behavior tends to be unique to a particular circuit, and is defined by the electrical engineer when importing the circuit from a CAD tool.

This way of defining new components has proved to be accessible to engineers in industry, and a fairly intuitive way of representing many components. This contrasts with many other qualitative simulation schemes where new components would have to be constructed by the system developers.

Given a circuit to simulate, and the initial state of each component in the circuit, the simulation controller will perform the following steps:

1. Build a resistive network from knowledge of the component states and the connections between components.
2. Pass the resistive network to the network analyzer, and get back details of where current is flowing in the network.
3. Use the details of the current flow to identify components whose internal state has changed.
4. If any components have changed state, repeat from step 1, else terminate.

For example, to find out the state of the example circuit when the main headlight switch has been closed, the simulation controller does the following:

- Find out the electrical state of the circuit using the network analyzer. Both relay switch resistors are set to infinity (representing an open switch). There is current to the ECU and so it is active. The main headlight switch is closed, so there is also current to the main headlight input signal of the ECU.
- When the ECU is active, the main headlight input line is connected to the main headlight relay coil input with zero resistance (if the ECU was not active, then it would be connected with infinite resistance).
- The main headlight relay can then be calculated to be active, and so the state of the relay will change (the switch will close).
- The resistance through the relay switch will then be zero, and current will flow through the switch and through the main headlights. No further state changes take place.

In general, the processing will continue until the circuit reaches equilibrium or a feedback loop is identified.

2.2. Network analyzer

The network analyzer, based on CIRQ⁸, takes as input a graph made up of qualitative electrical resistances which represent either a component or part of a component; the resistances can take the values of zero, load or infinity. The output generated by CIRQ consists of a qualitative description of the electrical state of each resistance. This description will indicate active and inactive paths, as well as any shorted paths in the circuit. The CIRQ algorithm is based upon Dijkstra's shortest distance algorithm¹.

The analysis of a circuit network is split into two stages. The first stage labels the terminals of each resistance in the graph with a forward / reverse (F/R) value; this specifies the number of loads which will be traversed to reach the negative and positive terminals respectively. In some instances, it will not be possible to reach one of the terminals, in which case "infinity" is used as the load needed to reach a terminal in that direction. Figure 2 shows the CIRQ graph for the headlight labelled with F/R values. To aid readability, nodes for wires, fuses, and the ECU power supply have been omitted from the example network.

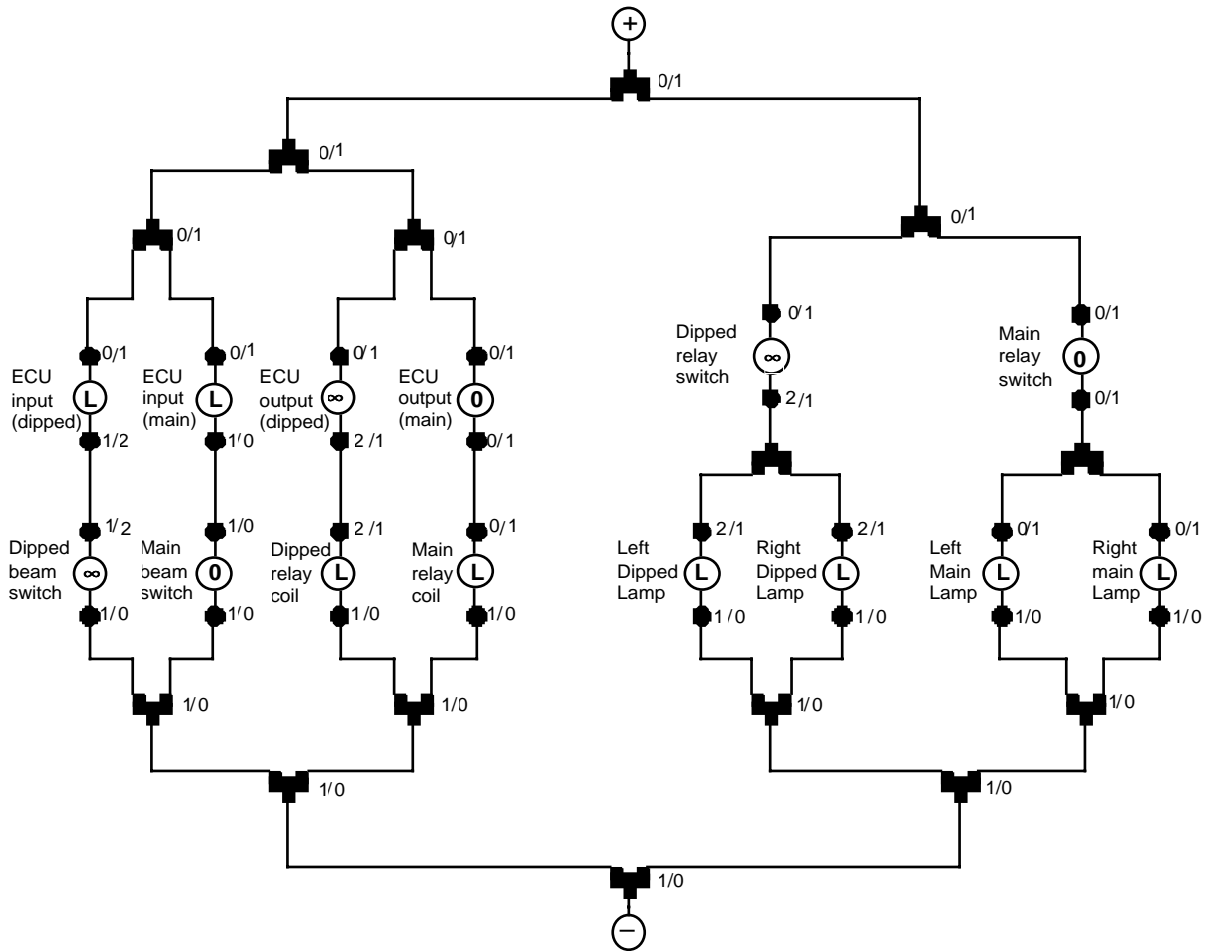


Figure 2: Resistance graph for headlight circuit

The circuit is in the state where the main headlight switch is closed, the dipped headlight switch is open, and the circuit has become stable. Present resistive values are shown for each node, and F/R values for each side of each node.

In the second stage, deciding which paths are active, shorted, and inactive, the network is traversed using a form of depth first traversal. All components whose terminal F/R values include a value of infinity are immediately marked inactive. Shorted paths are identified by a branch of the circuit having the same (non-infinite) F/R value at both ends; this implies that no load is being drawn by this part of the circuit. Components on other branches between these two points will be marked inactive (assuming that these branches are not shorted paths also) as the zero-resistance branch will draw all current. All other paths in the circuit are marked active i.e. they have not been shorted out, and are not inactive.

2.3 Level of usefulness of the qualitative simulation

For understanding how a circuit works, the level of information provided by the qualitative simulator is very valuable. The results of simulation can be conveyed to an engineer visually, by coloring the circuit diagram to illustrate which parts of the circuit are active under any set of conditions.

However, for the tasks being discussed in this paper, design analysis and diagnosis, the amount of information provided is far too detailed, even for the headlight example given here, and the qualitative simulation provides no way of abstracting the information. For a

typical large automotive subsystem which might have hundreds of components, the amount of information produced can be overwhelming. The next section describes how knowledge of the functions of the overall system can be used to focus the results of behavioral simulation for design analysis and diagnosis.

3. Interpreting Behavioral Simulation with Functional Labels

3.1. Relevant work in functional reasoning

Sembugamoorthy and Chandrasekaran¹⁷ characterize the significant overall behavior of a system in terms of the functions that the system performs. In that paper, a functional label is linked to an external stimulus (e.g. pressing a button) and to a modular set of descriptions of the behavior of the overall system. The behavioral descriptions explain how the stimulus achieves the function for a correctly working version of the device. By including in the behavioral descriptions the state of particular components and assumptions about structure, they can be used to perform predictions of loss of behavior due to component or connection failures. However, the behavioral descriptions are cumbersome to build for large systems, and no evidence has been provided of such descriptions being practical for large real-world diagnostic systems. Where available, qualitative simulation is a much more efficient (in engineer's time) way of predicting system behavior under failure conditions (what is needed for diagnosis or failure mode and effects analysis). Indeed, where the simulation produces novel behavior or unexpected structural interactions (as in sneak circuit analysis), then the correct behavior cannot be deduced without qualitative simulation.

On the other hand, functional labels such as those used in the scheme described above are an important way of focusing attention when performing diagnosis on a device. They identify the important attributes of a system or device. These are likely to be related to user goals: they might be an abstraction either of states that the user wishes to attain or of states that the user wishes to avoid.

Examples of functional labels for particular car subsystems might be:

Central locking system:

Doors locked
Doors open
Doors locking
Doors opening

Wash/wipe system:

Screen wash
Slow wipe
Intermittent wipe
Fast wipe

External lighting system:

Main beam
Dipped beam
Sidelights
Stop lights
Right indicators
Left indicators
Fog lights
Reversing lights

Cruise control system:

Accelerating
Cruising
Decelerating

The significant overall behavior of many systems can be characterized by a few such labels. Unlike the descriptions of how the overall system behavior is achieved, these labels are highly reusable. Linking the functional labels to the results of qualitative simulation can overcome the problems with qualitative simulation of providing too much detail, as described at the end of section 2. Functional labels can enable interpretation of the behavior of the system in terms of the overall goals of that system.

Other schemes linking function to qualitative simulation^{15,4,3} have chosen to employ both the functional labels and descriptions of overall behavior similar to those described in Sembugamoorthy and Chandrasekaran¹⁷. Applying such schemes to practical tasks demands a great deal of effort from the users. For some tasks, such as design verification⁵, that level of detail is necessary. For diagnosis, for failure mode and effects analysis, and for sneak circuit analysis, it is possible to link functional labels directly to structure in such a way as to provide the necessary information. The next section will illustrate this for the simple headlight system discussed earlier.

3.2. Linking function to behavioral simulation

In order to use functional labels to simplify and interpret the behavior of the qualitative simulation, it is necessary to be able to identify when the functions are being carried out in the simulation. The presence of each function can be identified from the states of key components.

Functional labels for the headlight system are: *Main beam* and *Dipped beam*. An arbitrarily complex expression combining the state of many components can be used to link a functional label to component states. In the headlight example, the links are fairly simple, with the *Main Beam* function happening when *left_main_lamp = ACTIVE* and *right_main_lamp = ACTIVE*, and the *Dipped Beam* function happening when *left_dipped_lamp = ACTIVE* and *right_dipped_lamp = ACTIVE*.

A functional interpretation of each state in the qualitative simulation can then be automatically obtained by matching each possible function against the state of the relevant components. By changing the states of each of the input stimuli (switches and sensors) in the headlight system, a state graph like that shown in figure 3 can be generated, summarising possible states for the headlight system and indicating how each state can be attained from any other state.

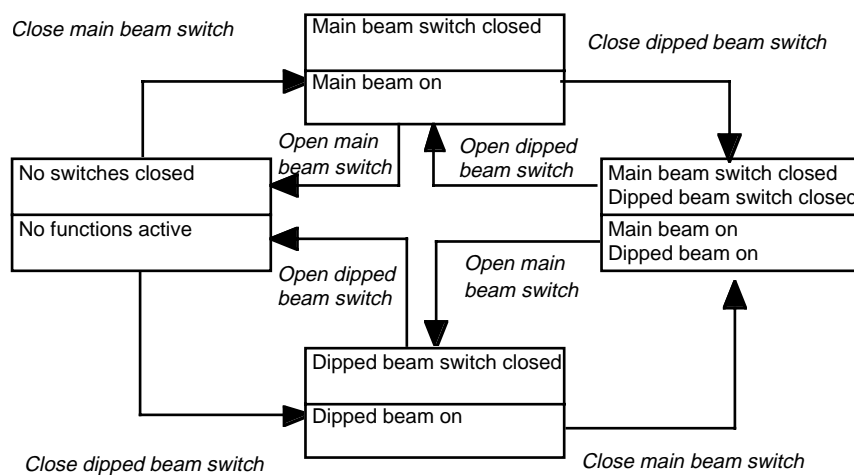


Figure 3: State summary produced by using functional labels

This is a dramatic reduction in the amount of information produced by the simulation, and the remaining information is focused on problem-solving in the domain. For a more complex circuit, the reduction from state information produced by the behavioral simulation to functional state information is even more significant.

3.3. Usefulness of functional labels

One of the major advantages of using functional labels linked to a qualitative simulation is the high level of reusability that can be obtained. A simple example of this advantage can be shown by a change to the structure of the circuit. If an extra switch was added to the circuit in series with one of the other switches, or if both sets of headlights were operated by a single switch, then the new state graph for the system can be generated automatically, without even changing the links between function and structure. This is because the links to the functional level only involve the components that are needed to interpret the behavior: all generation of behavior is done by qualitative circuit simulation. Where explicit descriptions of overall circuit behaviors are part of the functional representation, they need to be altered in order to deal with such situations.

Where the task being carried out necessitates reasoning about the effects of component failures (tasks such as diagnosis or FMEA), then automatic generation of state graphs summarising function from the results of a qualitative simulation is even more significant. It becomes possible to make a change to the structure of the system (to change the circuit, in the case of electrical systems) and then to determine which functions of the system will be affected.

For example, assume the headlight circuit contained a relay with a “stuck open” failure, connected to the main headlight beams. The qualitative simulation could be run for a version of the circuit containing a faulty relay, and the state graph shown in figure 4 would be automatically generated for the headlight circuit. By comparing this state graph with the one shown in figure 3 for the correctly working system, it is possible to infer that the effect of the faulty relay is that the function *main beam on* will not be achieved when expected.

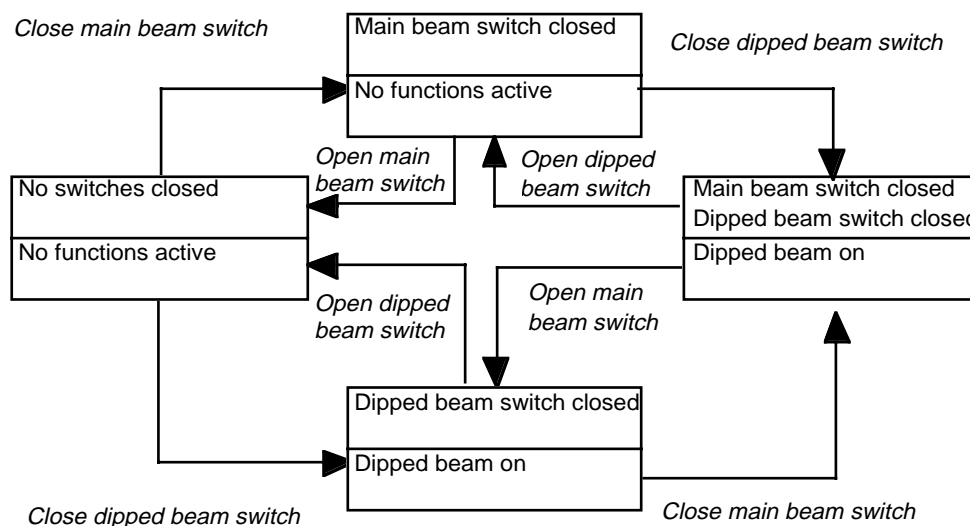


Figure 4: State summary with faulty relay to main beams

The use of functional labels along with structure has advantages over reasoning only at a structural level. At a first glance, it would seem that similar results could be obtained for simulation and envisionment merely by flagging interesting states of the system. The advantage of explicitly expressing the functional labels is that the behavior of the system is expressed in terms of the goals of the system, as is the case with other kinds of functional reasoning. This means that significant behavior can be identified and meaningful design analysis reports can be produced. The functional labels are reusable for new versions of the same system, and are applicable to the automation of a number of different design disciplines.

The next section shows how this works out in practice for a variety of design disciplines practiced on electrical circuit designs.

4. Using Function for Design Analysis Applications

The combination of structure-based qualitative simulation with functional labels is effective for automation of several design analysis tasks. This section of the paper will consider its use for four types of application:

- failure mode and effects analysis
- sneak circuit analysis
- design verification
- design-time generation of diagnostic candidates

The FLAME system is used by automotive engineers in industry to perform the first two of these tasks. Software to perform the third and fourth task is still being developed, but is producing encouraging results in research settings.

4.1. Failure mode and effects analysis (FMEA)

Failure mode and effects analysis⁶ is used extensively in the aerospace and automotive industry to verify the safety of proposed electrical and mechanical designs, and to highlight any problems with the design. There are a number of steps that an engineer needs to go through in order to perform an FMEA of an electrical design:

1. For all components in the system, identify the ways in which they can fail. These are the failure modes for the component. In the context of the whole system, they are failure causes. For example, the main failure modes of a wire are shorted to ground, shorted to battery, and open circuit.
2. For each of the component failure modes, identify the effect that failure has on the system (the failure effect). For example, a wire shorting to battery may cause the headlights to stay on, when they should be off.
3. For each of the failure effects, rankings within the three classes of severity, detectability, and likelihood of occurrence need to be assigned. For example, brake failure would have a severity of 10, on a scale of 1 to 10, as this may cause death, serious injury, or damage to the car. Once values are defined for the three classes, the values are usually multiplied to generate a Risk Priority Number (RPN). If any of the three indices have high values, or the overall RPN is greater than a certain pre-set value (often 100), then ways of improving the design will be sought.

Item / Function	Potential Failure Mode	Potential Failure Effect	Sev	Potential Failure Cause	Occ	Current Design Controls	Det	RPN
(89)	Firstly, a short circuit possible caused FUSE_1 to blow. When DIPPED_BEAM_SWITCH was set to Switch_Closed (2) the "lights of function was achieved when the "main beam on" function was expected. Also, when MAIN_BEAM_SWITCH was set to Switch_Open (3) the "lights off" function was achieved	The dipped beam lights have failed. Some of the lights failed to come on when expected. The main beam lights have failed.	9	The component \$1N98 has failure short_to_ground.	1		10	90
(90)	Regardless of any event change, the "lights off" function was never achieved and the "dipped beam on" function was always achieved.	The dipped beam was on unexpectedly. The lights failed to go off when they should have.	1	The component \$1N98 has failure short_to_battery.	1		2	2
(91)	When MAIN_BEAM_SWITCH was set to Switch_Open (3) the "lights off" function was achieved unexpectedly. Finally, regardless of any event change, the "dipped beam on" function was never achieved.	The dipped beam lights have failed. Some of the lights failed to come on when expected.	9	The component DIPPED_BEAM_SWITCH has failure switch_broken.	1		10	90
(92)	Regardless of any event change, the "lights off" function was never achieved and the "dipped beam on" function was always achieved.	The dipped beam was on unexpectedly. The lights failed to go off when they should have.	1	The component DIPPED_BEAM_SWITCH has failure stuck_closed.	1		2	2
(93)	Regardless of any event change, the "dipped beam on" function was never achieved.	The dipped beam lights have failed.	9	The component DIPPED_HEADLIGHT_L has failure blown.	1		10	90

File: /dcs/rcs/powerview/testing/fmea/headlamps.ef Subsystem: testing:headlamps Order: simulation.

Figure 5: FMEA report generated by FLAME

The FLAME application automates the whole of the process described above. Having designed a circuit using a standard CAD tool, the design analysis tools are available to the engineer within the CAD tool's environment. Qualitative definitions of the behavior of standard components such as wires, relays and lamps already exist. The internal qualitative behavior of novel components such as ECUs must be described. Possible failure modes for those new components must also be defined.

The engineer can then run qualitative simulations of the circuit to ensure that its correct behavior is as expected. The functions for the circuit are taken from a database of functions for each subsystem of the car, and can be linked to the circuit state using a simple graphical tool.

The FLAME system then carries out the following actions to generate an FMEA report:

Obtain the correct behavior. Use the electrical qualitative simulator to simulate the behavior of the circuit through its possible changes, by operating switches and changing sensor states. The resultant behavior of the circuit is abstracted by recognizing the activity of significant components, and attributing functional labels to each state, as illustrated in section 3.3.

Make a list of all of the possible failures that can occur in the circuit. The possible failures modes of each type of component are defined. The complete list of possible failures for the circuit can be compiled from all of the possible failures of each component in the circuit.

Obtain faulty behavior of the circuit. For each possible single point failure, impose that failure upon the circuit and repeat the simulation and abstraction work done for the correct circuit.

Compare the faulty and correct abstracted behaviors. Functions that occur when they should not, or which do not occur when they should, describe the significant incorrect

behavior of the circuit under the given failure. Discrepancies of this kind form the content of the textual report for the effects of the failure, and are transformed into a more engineer-friendly style, as shown in figure 5.

Assign severity, detectability, and likelihood values. RPNs are generated by assigning severity and detection values to each functional label, and combining this with a likelihood of occurrence value which depends on structural complexity and component type.

The simple headlight circuit illustrated in figure 1 has 92 possible failures that must be examined. In more complex circuits, many hundreds of failures need to be examined, and manual generation of an FMEA report can take several months of effort by an engineer. An example of the output generated automatically by the FMEA generation system is shown in figure 5.

Execution time for generating a complete FMEA report with the latest version of the FMEA automation software varies from a few seconds for the circuit given as an example, to several minutes for the most complex subsystem in a car, containing hundreds of components. Empirical tests show execution to be $O(n^3)$ with number of components.

Ford Motor Company and Jaguar Cars have each evaluated the FLAME software by attempting to generate FMEA reports for all of the electrical subsystems in a new car design, and comparing the results with those generated 'by hand'. This evaluation has highlighted both the benefits and the limitations of the work. The benefits will be highlighted here. The limitations will be addressed separately as they are limitations with the modeling capabilities, and so apply to all of the design tasks discussed here.

The main benefit of the method is that it has enabled the automatic production of FMEA reports of a similar quality to those produced by engineers, only much more quickly and with a much greater degree of consistency.

The consistency of the produced output gives benefits that had not been anticipated when building the system. Because the FMEA report is given in terms of reusable functions, incremental FMEA becomes possible: when the design of a circuit is changed, a new FMEA report can be generated, with the engineer being shown only the FMEA results that differ from the report for the previous version of the circuit. This means that a discipline which previously took several weeks of an engineer's time has been turned into an exercise which can be performed on a "what if" basis to see if a change in design produces a circuit with improved reliability.

The simplicity of the functional label idea linked with use of the standard structural representation for the domain means that FLAME's FMEA system can be used in industry by automotive engineers with no help from computer scientists. It can deal adequately with almost all of the circuits. The limitations of the system and the modeling compromises that need to be made are discussed in section 5.1.

4.2. Sneak circuit analysis

Sneak circuit analysis is the process of identifying and eliminating unexpected interactions between electrical subsystems. Typically, a problem might be caused when a wire which was expected to provide current in one direction is used in the opposite direction, causing a sneak path. A good example problem is given by Savakoor et al.¹⁶, and pictured in figure 6, for the cargo bay doors of an aircraft.

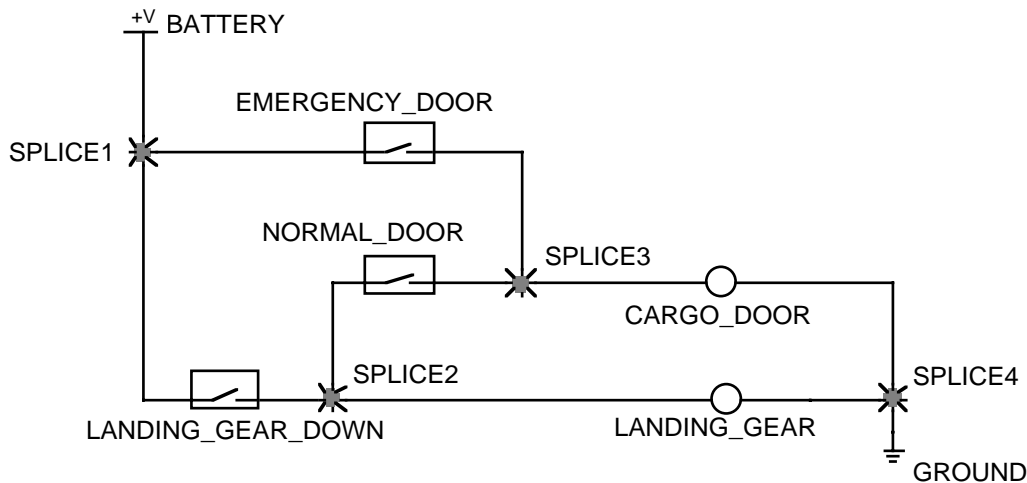


Figure 6: Cargo door circuit with sneak path

The cargo door switch should only make the cargo door open when the landing gear is down (i.e. when the aircraft is on the ground). Operating the emergency switch for the cargo doors can cause the landing gear to lower unintentionally while the plane is in flight if the normal door switch is also closed. Similar published examples also exist in car electrical systems.

Eliminating a sneak path once it is discovered is often much simpler than detecting it in the first place: if a wire is allowing current to flow in an unexpected direction, this can often be prevented by the addition of a diode in the design. However, the problem cannot be finessed away by adding diodes to all lines in the circuit, as cost considerations mean that diodes should not be added to the circuit unless they are really needed.

FLAME can identify sneak paths automatically. In order to do this, as well as the information that already exists for FMEA, the user must declare which inputs (switches or sensors) would be expected to cause which functions to occur. The FLAME system then carries out the following algorithm on the circuit in order to detect sneaks:

- Activate each combination of switches in the circuit and run the electrical simulator.
- Use function-to-circuit links to identify which functions are active in each state.
- Compare the active functions with the functions expected to be active.
- If any additional functions occur, then there is a significant sneak between the subsystems, and a report to that effect should be generated, giving the switches that were closed at the time, and the additional effects that happened.

The described method correctly identifies classic example sneaks such as the cargo door sneak, and example output is shown in figures 7 and 8. Figure 7 shows a one line report for each of the sneak paths discovered – only one in this case. Clicking on the sneak report starts up the circuit simulator in the state which contains the sneak, as shown by the state settings in figure 8. FLAME’s sneak circuit facility has also found actual sneaks in circuits where the engineer did not know that the sneak existed. It is capable of finding all significant sneak paths in a circuit (this assumes that sneak paths which do not activate any extra functions are not significant). This is useful to engineers, but it relies on them identifying a set of subsystems where sneak paths might exist. The ultimate challenge for a sneak circuit system is to consider the whole circuitry of a car (or airplane) at once and find all possible sneaks. Our present research is working towards this by reasoning about circuits more

hierarchically, and using well-proven domain heuristics to identify subsystems which might interact in this way.

Sneak circuit analysis as described in this section is a task practiced only in the electrical domain. However, the question that is answered by the work outlined above is of wider interest: *Given two (or more) functions fulfilled by linked structures, is there any interaction between the structures which fulfil the functions that can cause other functions to be fulfilled?*

This question is of interest in domains other than the electrical, such as the hydraulic or the mechanical domain.

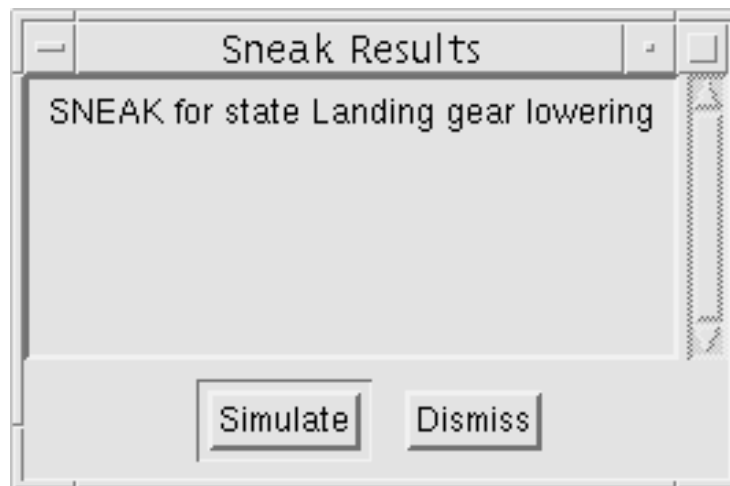


Figure 7: Output for cargo door sneak

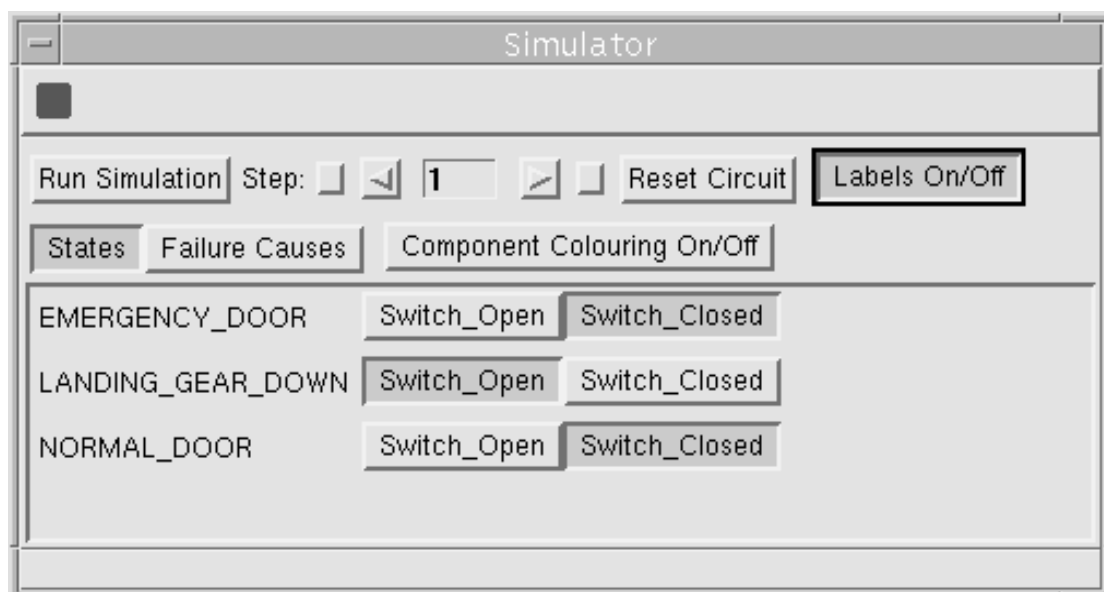


Figure 8: Circuit state where sneak occurs

4.3. Design verification

Significant previous work in this area has been done by Iwasaki and Chandrasekaran⁵. They use a “standard” functional representation of the type in Sembugamoorthy and Chandrasekaran¹⁷ to describe both the function of a system and how it is to be achieved. They use the description of overall behavior to verify that a qualitative simulation of a system achieves the expected function, and achieves it in the expected way.

One problem with this approach is that the method used to describe overall system behavior does not match the kind of descriptions produced by engineers to describe overall system behavior. This means that such models are not easily available in industry. In the automotive industry, engineers are moving towards using techniques such as Statecharts² to produce a state-based specification of the operation of a system. These specifications are very similar to the state map shown in figure 3. Statecharts can be used to produce a requirements definition for the system being designed.

The Statechart requirement specification can be compared with an envisionment generated from qualitative simulation and functional labels, in order to identify unexpected or missing states. Because the envisionment looks at all possible combinations of inputs, some of the unexpected states can be ruled out by stating that certain combinations of inputs are physically impossible (for example, main and dipped beam lights being switched on at the same time might be physically impossible if there is one switch for the two functions, and the switch can only switch one of them on at any time).

We are currently investigating the generation of proposed changes to structure when there are discrepancies between the Statechart description and the qualitative envisionment, but automated identification of discrepancies is a useful first step.

4.4. Diagnostic candidate generation

The combination of structural simulation and functional labels is capable of generating candidate faults with the right level of symptom for performing diagnosis – descriptions of functional level changes to expected behavior. The FMEA output can be rearranged by overall functional effect rather than by cause of failure, and the result would be an ordered tree giving the possible single component failures that could cause any symptom. This can be extended. The qualitative simulator is able to deal with multiple failures, and so can overcome the limitation of only dealing with single faults. It has been used to generate all pairs of component failures for circuits. For an example circuit with 220 single failures, there are 24090 possible pairs of failures.

Most of the pairs of failures produced in this way do not have interesting effects: they only give results that might be inferred from the single failure results. The following test can be used to decide whether a pair of component failures has an interesting effect.

Let the failure symptoms for a component failure 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 an expected function.

For two component failure X and Y , the multiple failure $X\wedge Y$ is not interesting if:

$$D_{X\wedge Y} = D_X \quad \text{or} \quad D_{X\wedge Y} = D_Y \quad \text{or} \quad D_{X\wedge Y} = D_X \cup D_Y$$

For example, if the consequence of component failure 1 is that the main headlights do not work when expected, and the consequence of component failure 2 is that the dipped headlights do not work when expected, then the combination of the two failures is not

interesting if the consequence is that neither the main nor the dipped headlights work when expected.

For the circuit with 220 single failures, less than 200 of the 24090 pairs of failures are interesting by these criteria. This reduces a large number of results down to a size where the design engineer can study both the single results and interesting pairs of results to verify their correctness. This means that candidate faults verified by the circuit designer can be produced for use in diagnosis for almost no extra effort. Function also provides a way of ordering the faults so that the faults that match the symptoms can be easily discovered.

This work has been extended to cover all combination of failures up to a certain level of likelihood of occurrence¹², and we are developing better algorithms for using the results in diagnosis.

5. Limitations of the approach and further work

5.1 Coping with complex behavior

The FLAME system has been used to simulate the electrical systems of modern automobiles, and is able to produce reasonable results for 85% of the circuitry. For many of the circuits, some modeling compromises are necessary, and this section gives examples of the kinds of modeling strategies that have been employed in order to produce useful results.

Encapsulate complex behavior within a component. This is the most common way of dealing with complex behavior. For example, where modern cars use ECUs (computerised control) to switch circuits, it is not necessary, or in many cases even desirable, to model the software that is performing the switching. The relevant behavior of the ECU can be modelled as connections between ECU terminals with resistances that switch between zero and infinity. This strategy fits in well with the two levels of reasoning outlined in the previous section. It can also work for non-electrical components with switching behavior.

Distribute complex behavior among several components. For example, because of the qualitative nature of the simulation, effects which depend on quantitative values cannot be directly modelled. This happens in some windscreen wiper systems, where a multi-way switch changes the value of a resistance, and an ECU reads an analogue value on a line to decide on the speed of the windscreen wiper. This can be achieved by a mild violation of the “no function in structure” principle, where you have two linked components, the switch and the ECU. The switch sets an external state giving its analogue value, and the ECU reads that analogue value from the switch. It only does this if the wire on its terminal coming from the switch is ACTIVE. This allows it to deal correctly with failure conditions such as the wire failing open.

Simplify the complex behavior so that it is manageable. Electronic cruise control systems contain fairly complex algorithms for deciding what to do, but for examining electrical failures, they can be simplified to three conditions: above desired speed, below desired speed and at desired speed. This allows the basic behavior of the cruise control to be exercised, without consideration of much unnecessary detail.

Ignore the complex behavior altogether. When performing failure mode and effects analysis (FMEA) on the electrical circuitry of a car, it is possible to ignore some phenomena altogether, and still produce a reasonable FMEA report. Take an indicator light for example. The light will flash on and off when the circuit is working

correctly. However, the overall behavior of the circuit can be obtained by treating the lamp as a “normal” lamp rather than a flashing one.

The first two types of problem are solved at the structural level, by building components with the right kinds of behavior. Some of the components being built are very complex, for example, central locking ECUs which change the state of the locks to open after a certain delay if all of the locks do not succeed in closing properly. Such components could be built more succinctly as statechart descriptions of behavior. Early experiments indicate that such encapsulated statecharts can be included in the FLAME system as component descriptions.

Some of the challenges of complex behavior identified in the third and fourth types of modeling compromise could be more fully addressed by having more complex descriptions of function. For example, the “flashing” function in the flashing lamp example could be defined as the bulb being repeatedly active and not active without any outside state being changed. Keuneke and Allemang⁷ discuss the representation of function in terms of sets of states, possibly moving iteratively between the states. We have chosen to avoid such descriptions at present. They would make providing a description of function into a much more complex process for the engineer, and it is not clear that they would provide significant benefits.

5.2 Use of functional labels in other domains

This paper has shown that the use of functional labels to interpret the results of a qualitative circuit simulator provides a practical tool for building design applications in the domain of electrical circuits. Functional labels also show promise for use with other types of simulator and in other domains.

Work has been done by our collaborators at Ford Research Centre, Dearborn, Michigan into using SABER, a quantitative circuit simulator, for generating the effects of failures¹⁰. The same functional labels can be used to interpret the meaning of the quantitative simulation. The links are more complex, involving ranges of values rather than just whether components are active, but they enable the extraction of meaningful information from a complex simulation. Even better, they could be used to link the quantitative results to situations where the qualitative results are ambiguous. This has not yet been done.

Functional labels should also be useful in other domains. Experiments with qualitative simulation of hydraulic systems have shown promising results. The main requirements in order to use functional labels seem to be the ability to perform a component-centred simulation and to identify function from the state of significant components in a system.

6. Conclusions

Qualitative simulation from structure can be used to answer questions about the behavior of a system. Practical applications of the technology have been limited because of the problem of highlighting significant details from the results of a good qualitative simulator. Functional labels give a simple and effective way of interpreting the results of the simulation.

For the types of task described in this paper, the use of functional labels has several advantages over more complex descriptions of functionality:

Simplicity. More complex functional schemes can be difficult for engineers to specify. Functional labels require only that the main purposes of the device are identified, and that the component structure for qualitative simulation can be imported from the design for the device.

Reusability. Where functional representations specify how a function is achieved, that specification is tied to a particular implementation of the function. Functional labels only specify what is done, not how, and so are much more reusable.

Capability. More complex specifications of functionality cannot identify the unexpected achievement of functions or decide on functionality under fault conditions. Functional labels identify functionality from the results of the structural simulator, and so can detect unexpected achievement of functions and deal with fault conditions.

These advantages have been illustrated in FLAME, a design analysis system in current use by automotive engineers and able to produce quality FMEA and sneak circuit reports efficiently and accurately.

Acknowledgements

This work has been carried out on the UK EPSRC funded projects “Flame” and “Aquavit”, with the cooperation of Jaguar Cars Ltd, Ford Motor Company Ltd, the Motor Industry Research Association, Integral Solutions Ltd and Viewlogic Ltd.

References

1. Dijkstra, E. W. A Note on Two Problems in Connection with Graphs. *Numerische Math.*, 1959, 1, 269-271.
2. Harel, D.; Pnueli, A.; Schmidt, J. P.; Sherman, R. On the formal semantics of Statecharts. in *Proceedings 2nd IEEE Symposium on Logic in Computer Science*, 1987, pp. 54-64.
3. Hunt, J.; Price, C.; Lee, M. Automating the FMEA Process. *Intelligent Systems Engineering*, 1993, 2(2), 119-132.
4. Iwasaki, Y.; Vescovi, M.; Fikes, R.; Chandrasekaran, B. Causal Functional Representation Language with Behavior-based Semantics. *Applied AI*, 1995, 9(1), pp. 5-31.
5. Iwasaki, Y.; Chandrasekaran, B. Design verification through function- and behavior-oriented representations. in *Proceedings Artificial Intelligence in Design Conference*, eds. Gero and Sudweeks, 1992, pp. 597-616.
6. Jordan, W. Failure Modes, Effects and Criticality Analyses. in *Proceedings of Annual Reliability and Maintainability Symposium*, IEEE Press, 1972, pp. 30-37.
7. Keuneke, A. & Allemang, D. Understanding Devices: Representing Dynamic States. Ohio State University, Laboratory for AI Research Technical Report 88-AK-DYNSTATES, 1988.
8. Lee, M. & Ormsby, A. Qualitative Modelling of the Effects of Electrical Circuit Faults. *Artificial Intelligence in Engineering*, 1993, 8, 293-300.
9. Mauss, J. and Neumann, B. 1996. Qualitative Reasoning about Electrical Circuits using Series-parallel-star Trees. in *Proceedings of 10th International Workshop on Qualitative Reasoning*, 1996, pp.147-153.
10. Montgomery, T.; Pugh, D.; Leedham, S.; Twitchett, S. FMEA Automation for the Complete Design Process. in *Proceedings of Annual Reliability and Maintainability Symposium*, IEEE Press. 1996. pp. 30-36.
11. Price, C. Effortless Incremental FMEA, In *Proceedings of Annual Reliability and Maintainability Symposium*, IEEE Press. 1996. pp. 43-47.
12. Price, C. & Taylor, N. Multiple fault diagnosis from FMEA. to appear in *Proceedings of the Innovative Applications of Artificial Intelligence Conference*, Providence, Rhode Island, 1997.
13. Pugh, D.; Price, C.; Snooke, N. Practical Applications of Multiple Models - the need for simplicity and reusability. in *Applications and Innovations in Expert Systems III*, 1995, pp. 277-291.
14. Pugh, D. & Snooke, N. Dynamic Analysis of Qualitative Circuits. in *Proceedings of*

- Annual Reliability and Maintainability Symposium, IEEE Press, 1996, pp. 37-42.
15. Sasajima, M.; Kitamura, Y.; Ikeda, M.; Mizoguchi, R. FBRL: A Function and Behavior Representation Language. in Proceedings IJCAI-95, 1995, pp. 1830-1836.
 16. Savakoor, D. S.; Bowles, J. B.; Bonnell, R. D. Combining sneak circuit analysis and failure modes and effects analysis, in Proceedings of Annual Reliability and Maintainability Symposium, , IEEE Press, 1993, pp. 199-205.
 17. Sembugamoorthy, V. & Chandrasekaran, B. Functional Representation of Devices and Compilation of Diagnostic Problem-solving Systems. in Experience, Memory and Reasoning. eds. Kolodner and Riesbeck, Lawrence Erlbaum, 1986, pp. 47-73.
 18. Struss, P.; Malik, A.; Sachenbacher, M. Qualitative Modeling is the Key. in Workshop Notes of 6th International Workshop on Principles of Diagnosis (DX-95), 1995, pp. 99-106.
 19. Weld, D., & de Kleer, J. eds. Readings in Qualitative Reasoning about the Physical World, Morgan Kaufmann, 1990.