Formal Verification of Safety I&C System Designs: Two Nuclear Power Plant Related Applications

Janne Valkonen[1], Matti Koskimies[2], Ville Pettersson[1], Keijo Heljanko[2], Jan-Erik Holmberg[1], Ilkka Niemelä[2], Jari J. Hämäläinen[1]

[1] *Technical Research Centre of Finland (VTT), P.O. Box 1000, FI-02044 VTT, Finland (firstname.lastname@vtt.fi)*

[2] *Helsinki University of Technology (TKK), Department of Information and Computer Science, P.O. Box 5400, FI-02015 TKK, Finland (firstname.lastname@tkk.fi)*

Abstract

Instrumentation and control (I&C) systems play a crucial role in the operation of nuclear power plants (NPP) and other safety critical processes. An important change is the replacement of the old analogue I&C systems with new digitalised ones. The programmable digital logic controllers enable more complicated control tasks than the old analogue systems and thus the validation of the control logic designs against safety requirements has become more important. In order to diminish the subjective component of the evaluation there is a need to develop new formal verification methods. A promising approach is a method called model checking, which enables the complete verification of requirements when a finite state machine model of the system is available. The use of model checking to verify two nuclear power plant related systems is described: an arc protection system and a reactor emergency cooling system. For the verification, it was also necessary to model the operation environment of the device and the larger system it is part of. The environment models could be kept relatively simple, but it is important that the essential behaviour of the environment is covered. The reactor emergency cooling system is in use in an operating nuclear power plant and the arc protection system model included a typical realistic operation environment. The results showed that it was possible to reliably verify the presence of desired behaviour as well as the absence of an undesired behaviour of the system. The possibility for complete verification makes model checking different from simulation-based testing where only a number of selected scenarios can be simulated and one can never be sure that all the possible behaviour is covered. The challenges for future research are to develop more dedicated methods for the verification of safety critical automation and safety critical embedded software.

1. INTRODUCTION

The verification of safety I&C designs still relies heavily on subjective evaluation. Formal methods are available, but often they are only used for certain tasks as indicators of possible problems. Model checking is a promising approach that at least theoretically enables complete verification of the requirements, which is not possible with traditional simulation methods. A detailed dynamic model of the process and automation can be utilised in simulation-based analysis. However, model checking is based on a so-called state machine model of the control logic and the essential surrounding systems. Certain provably correct computational algorithms that analyse the state machine model are applied for evaluating the safety requirements one by one. Model checking can also handle delays and other time-related operations, which are crucial in many safety I&C systems and challenging to design and verify.

This paper describes the preliminary results of the research project MODSAFE (Model-based safety evaluation of automation systems) in the Finnish Research Programme on Nuclear Power Plant Safety 2007–2010 (SAFIR2010) [1]. The principle of the model checking methodology and two applications are described. An electric arc protection system with selective multi-zone protection is a typical representative example. Another example is a reactor emergency cooling system in an operating nuclear power plant.

2. MODEL CHECKING METHODOLOGY

The traditional way of validating safety critical systems has relied heavily on two standard techniques of testing and simulation. However, these techniques frequently do not scale at the rate of the system's size growth. The use of computer aided verification methods has been suggested as an aid to supplement these methods.

Model checking [2] is a set of methods for analysing whether a model of a system fulfils its specification by examining all of its possible behaviours. Model checking was introduced in the early 1980s simultaneously by two different groups [3,4] and for this invention Clarke, Emerson, and Sifakis were jointly given the prestigious ACM Turing award for 2007. Good introductory books on model checking are [2,5]. The employed models are usually finite state models such as finite state machines, but can contain very large state spaces that the model checking tools are tailored to efficiently explore with efficient algorithms. In this paper, state machine models contain state variables holding the current state of the modelled system, as well as holding any state needed for modelling the environment the analysed system is interacting with. In verifying such models, we employ much of the technology currently being applied for circuit validation applications, such as microprocessor validation tools. For more background on model checking in the context of NPP I&C, see [10].

In model checking, at least in principle, the model analysis can be made fully automatic with computer-aided tools. In our case, the models were written in the input language of the NuSMV model checker [6]. The specifications the models are required to fulfil were expressed in linear temporal logic that is also directly supported by the NuSMV tool. In the model of the emergency cooling system, also fairness properties [2] were employed to make the state space of the model smaller while retaining all interesting behaviours of the environment. Now, given a model and its specification as input, a model checker decides whether the system violates its specification or not. If none of the behaviours of the system violate the given specification, the (model of the) system is correct. Otherwise, the model checker will automatically give a counterexample execution of the system model demonstrating why the property has been violated.

In symbolic model checking employed in this paper, the main idea is to represent the behaviour of the system very compactly in a symbolic form rather than explicit one. There are several variations of symbolic methods. The most well-known is the use of a data structure called ordered binary decision diagrams (OBDDs), which are a canonical representation of Boolean functions [7,8]. This allows for a much more efficient memory and often also faster examination of the reachable states of the system than with methods representing each one of the reachable states of the system explicitly.

3. ARC PROTECTION SYSTEM

The first case we studied concerned how model checking can be used to verify the design of an electric arc protection system. For the case study, we considered a rather involved setting

where selective multi-zone protection is designed for a typical power distribution set-up. The case study was not directly based on any existing installation, but it was reviewed by experts and considered to be realistic. In this section, we first introduce the design of the system to be verified, discuss the assumptions needed for modelling the system, consider the properties of the design to be verified, and then represent the verification results.

3.1 The System Design under Verification

In order to make the case study concrete we considered a real electric arc protection system, the Falcon system developed by the Urho Tuominen (UTU) company, when designing the arc protection functionality. Falcon can be used to protect switchgear and electrical instrumentation from electric arcs. The system consists of a master unit, overcurrent sensor units, and light sensor units. Sensors are installed into the protected system and connected to the master unit via optical cables. The master unit collects the alarm signals from sensors, and when necessary, launches circuit breakers which close the power feed from the protected device leading to termination of the electric arc. The master unit is based on a Programmable Logic Controller (PLC) so that one can freely design and program the tripping logic according to the protected system and the protection required for it. This provides the possibility for selective tripping: the protected system can be divided into several protection zones with different tripping conditions. Falcon also provides a possibility for controlling backup breakers which can be launched in the case of a malfunction in a primary breaker. Backup breakers introduce an indirect closed control loop between Falcon and its environment: a continuous alarm of a certain length indicates a malfunction in a primary breaker.

Figure 1 shows the switching diagram of the system design that we are considering. The protected system consists of the following components:

- main power feeds pf1 and pf2
- transformers tr1, tr2, tr3, and tr4
- primary circuit breakers A, B, C, and D
- backup circuit breakers E, F, H, and G
- protection zones 1, 2, and 3

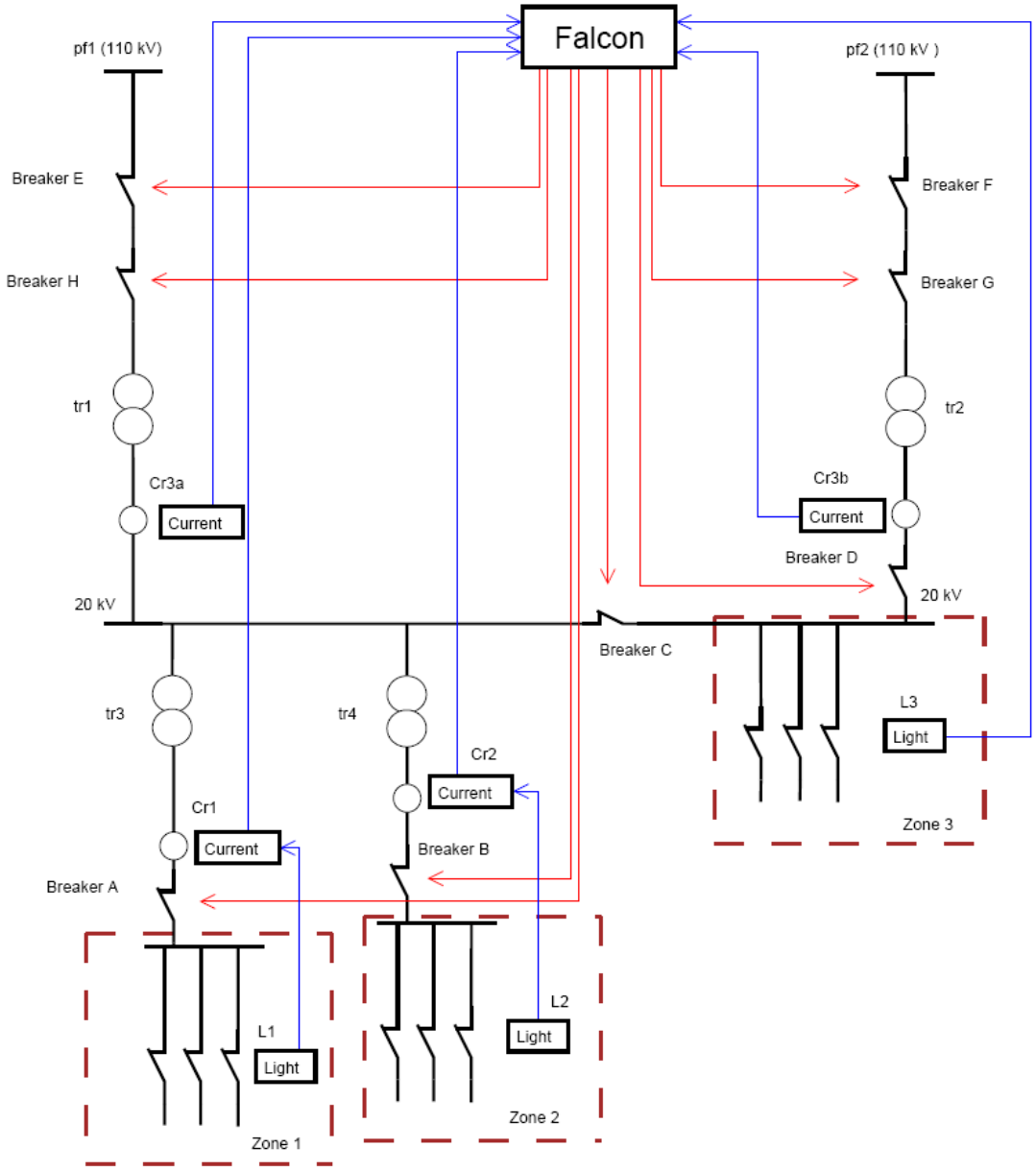

Figure 1. The switching diagram of the system design in the arc protection case study

The Falcon system for providing arc protection introduces the following elements into the whole system:

- the Falcon master unit
- over current sensors Cr1, Cr2, Cr3a, and Cr3b
- light sensors L1, L2, and L3

The main power feeds pf1 and pf2 distribute electricity to the protected system. They are connected to each other by the switch operated with the circuit breaker C, and therefore, they act as backup systems for each other. Hence, both pf1 and pf2 can deliver power to the whole protected system alone if a malfunction occurs in one of them.

The protected system is divided into three distinct protection zones. For all of these there is a zone-specific tripping condition, which causes tripping of the circuit breakers, which in turn leads to the isolation of the protection zone from the power feed. The protection system is

designed to operate with each protection zone so that there are two levels of backup breakers. That is, if the primary breakers are broken, the protection system tries first to cut down the power feed only from the main power feed that is closest to the alarming zone (the first level). If the alarm is still on (which might result, for example, if the connecting breaker C was broken), then the power feed will also be cut in the other main power feed, which will lead to cutting down the power feed in the whole system (it is presumed that backup breakers are working correctly).

The tripping conditions and related actions are listed in Table 1. Figure 2 shows a tripping logic which is created based on this table. The delay parameters D1–D4 introduced in Table 1 specify how long an alarm has to last non-stop before an action is taken. The delays D1 and D3 are related to the backup breakers of the first level and delays D2 and D4 are related to the second level. Therefore, it should be that D1 < D2 and D3 < D4.

The goal of the design is to guarantee that the following requirements are satisfied:

- p1: The installation of the sensors and the tripping logic should conform to the specified tripping conditions.
- p2: The backup breakers should not be tripped unless necessary.
- p3: Existence of an electric arc in the protected system leads eventually to shutting down the power feed to the protected system.

In order to verify these properties, we need to model the essential features of the arc protection system and the protected system and then formalize unambiguously the requirements p1-p3. We discuss first the assumptions needed for modelling the system and then the formalization of the properties to be verified.

Table 1: Tripping conditions and related actions caused by alarms in different protection zones

Alarm	First action	Second action	Third action
Cr1 AND L1 (alarm on zone 1)	Breakers A and C launched	Breaker E launched (after delay D1)	Breaker F launched (after delay D2)
Cr2 AND L2 (alarm on zone 2)	Breakers B and C launched	Breaker E launched (after delay D1)	Breaker F launched (after delay D2)
(Cr3a OR Cr3b) AND L3 (alarm on zone 3)	Breakers C and D launched	Breaker G launched (after delay D3)	Breaker H launched (after delay D4)

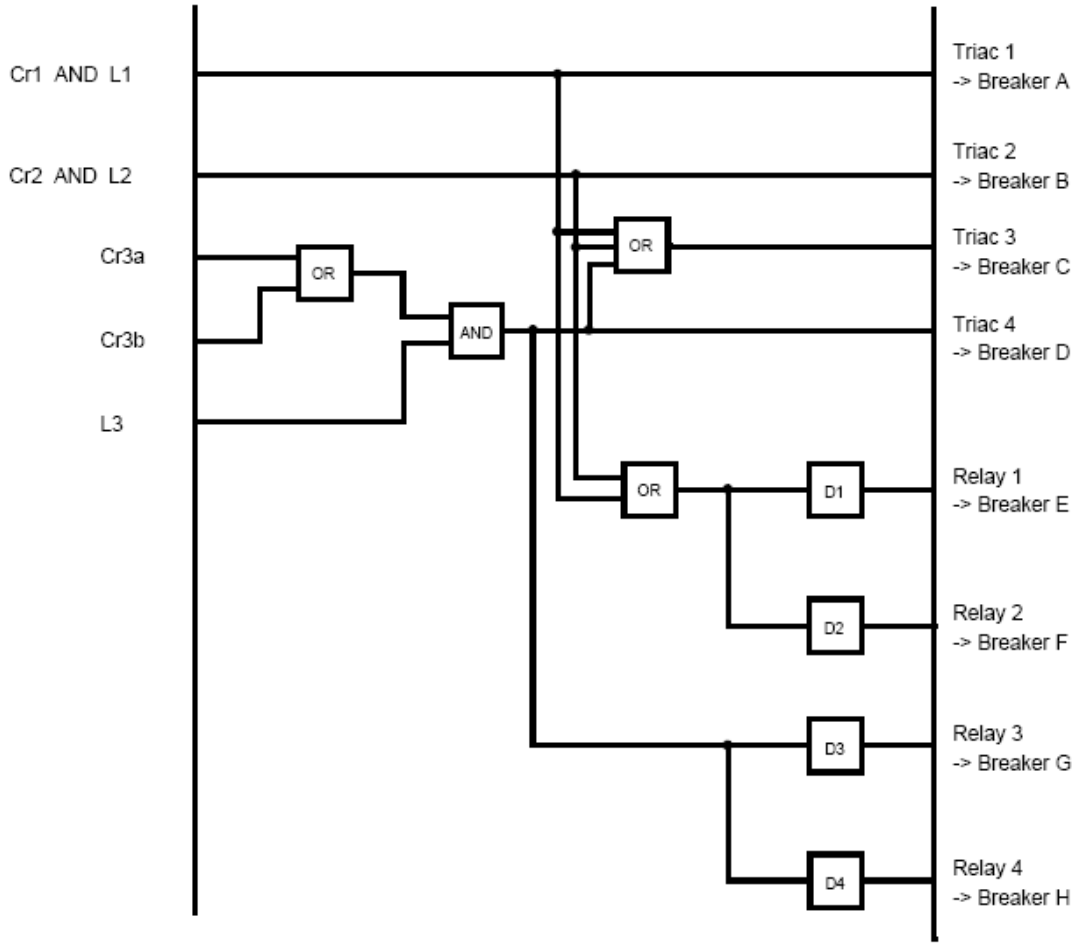

Figure 2. Tripping logic in the arc protection case

3.2 Modelling of the System

In order to be able to carry out the modelling process we need to explicate a number of assumptions of the functional and behavioural properties of the system. The main assumptions are described in the following.

The duration of one operation cycle of the controller of the Falcon master unit, that is, the time during which the Falcon system detects an alarm signal through a sensor and passes a launch signal to a circuit breaker is 1 millisecond. This time period will correspond to a single time step in the model of the system. Failures of the physical devices other than the primary circuit breakers are not considered. Overcurrent peaks detected by the overcurrent sensors are caused by short circuits. Short circuits can arise only in the parts of the protected system which are defined as protection zones. Over current peaks cannot move through the transformers. An overcurrent sensor can raise an alarm signal non-deterministically anytime as long as it is connected to the protection zone it is overseeing and the protection zone is still connected to a power feed. If these conditions are not met, the overcurrent sensor cannot raise an alarm. A light sensor can raise an alarm signal non-deterministically at any given time instant, that is, light alarms are independent of the rest of the system. Once a circuit breaker has been activated, it opens the electric circuit and prevents the flowing of the current. An activated circuit breaker will remain activated forever. There is an activation delay associated with each circuit breaker, which is the period between the moment when a breaker is launched

and the moment when it has opened the circuit preventing the electric current flowing. A non-activated primary circuit breaker can break down at any given time. A broken circuit breaker cannot open a circuit and will stay broken forever.

In order to verify the specified requirements p1–p3 both the protection system and the protected system need to be modelled because of the (indirect) closed control loop between the Falcon system and its environment.

Using the assumptions above, a NuSMV model capturing the essential behaviour of the protected system and the arc protection system can be built in a systematic way. The parts of the whole system that are modelled include the controller and the sensors of the Falcon system, the circuit breakers of the protected system, and the operational logic of the electricity distribution network. In addition, the time delays related to the delay gates of the controller and the physical opening delay of the circuit breakers are covered. The modelling of the whole system is done by describing the physical parts of the system with NuSMV module declarations and interchaining the modules according to their physical counterparts in the system description. The delays are modelled with modules acting like counters and by using these as building blocks for the physical parts where needed. For more details about the resulting NuSMV model, see [9].

3.3 Properties to Be Verified

Given the model of the system, requirements p1–p3 can be formulated precisely as follows:

- For primary breakers p1 states: If a primary circuit breaker is launched in a certain time step, then the tripping condition of this breaker was realized in the previous time step.
- For backup breakers p1 requires: If a backup breaker is launched in a given time step, then at the same time step one of the primary breakers covered by the backup breaker is receiving a launch signal.
- Property p2 can be formalised as a requirement for backup breakers: If a backup breaker receives a launch signal, then at least one of the primary breakers covered by it has broken down.
- Property p3 states: If the protection system receives an alarm from a protection zone in a given instant of time, there will be an instant of time in the future, when the alarm has either disappeared from the protection zone or the protection zone is disconnected from the power feed.

These requirements can be stated directly using linear temporal logic, which is one of the requirement specification languages supported by NuSMV (for details see [9]).

3.4 Verification Results

The model contained five adjustable parameters, which were the physical activation delay A of the circuit breakers, and the delay parameters D1, D2, D3, and D4 of the four different delay gates of the tripping logic (see Figure 2). The parameters were defined as positive integer values and they corresponded to milliseconds in real life.

Table 2 presents a few experimental results of the running times of the model checking process with parameter values that satisfy the properties p1, p2, and p3 described above. For the experiments, a standard PC and NuSMV ver. 2.4.2 with BDD-based LTL model checking was used. For each value of A, valid values for the parameters D1–D4 have to be determined from the system design. For a given parameter A it is of interest to find as small values of

parameters D1–D4 as possible for which the properties p1–p3 are still valid. For a complex system design, this is a difficult and error-prone task. This is reflected by the fact that with the parameter values of Table 2, the state space of the system model of this case study varies between $3{,}0*10^{21}$ and $2{,}4*10^{29}$ states (size of the state space can be calculated as the product of the value ranges of each state variable of the model). Consequently, the benefit of the automatic verification with model checking increases with the complexity of the verified system.

If the delay parameters for a certain activation delay A are chosen to be too small, all the properties are not valid anymore. In this case, the NuSMV model checker returns a counter example for each property that is violated. For example, if the value of the parameter D1 on the row A=3 in Table 2 is decreased from value 8 to value 7, a property of type p2 no longer holds and NuSMV returns a counter example, that is, an execution of the system along which the property is violated. In this case, the running time was increased to 10min 9sec.

Table 2: Running times for different parameter values

A (activation delay)	D1	D2	D3	D4	Running time
2	6	9	3	6	1min
3	8	12	4	8	4min
4	10	15	5	10	21min
5	12	18	6	12	39min
6	14	21	7	14	1h 48min
7	16	24	8	16	3h 33min
8	18	27	9	18	26min
9	20	30	10	20	11h

4. EMERGENCY COOLING SYSTEM

In the second application, the main focus was on verifying the control logic design of an existing emergency cooling system.

4.1 Functional Description

The purpose of the emergency cooling system is to ensure the water cooling of the reactor core if the ordinary cooling systems are out of order. The cooling system is controlled by an electronic control system which regulates the water level in the reactor containment by controlling the pumps and valves. The operational principle is analogous to a thermostat: When the water level gets too low (due to boiling and evaporation) in the reactor container, more water is pumped in until the water level reaches the upper level. This cycle is repeated as long as necessary, that is, until normal cooling systems are in use again. The system has four redundant and identical units. Figure 1 below demonstrates the system's behaviour and relationship between the physical parts, sensors, and control logic (for details see [9]).

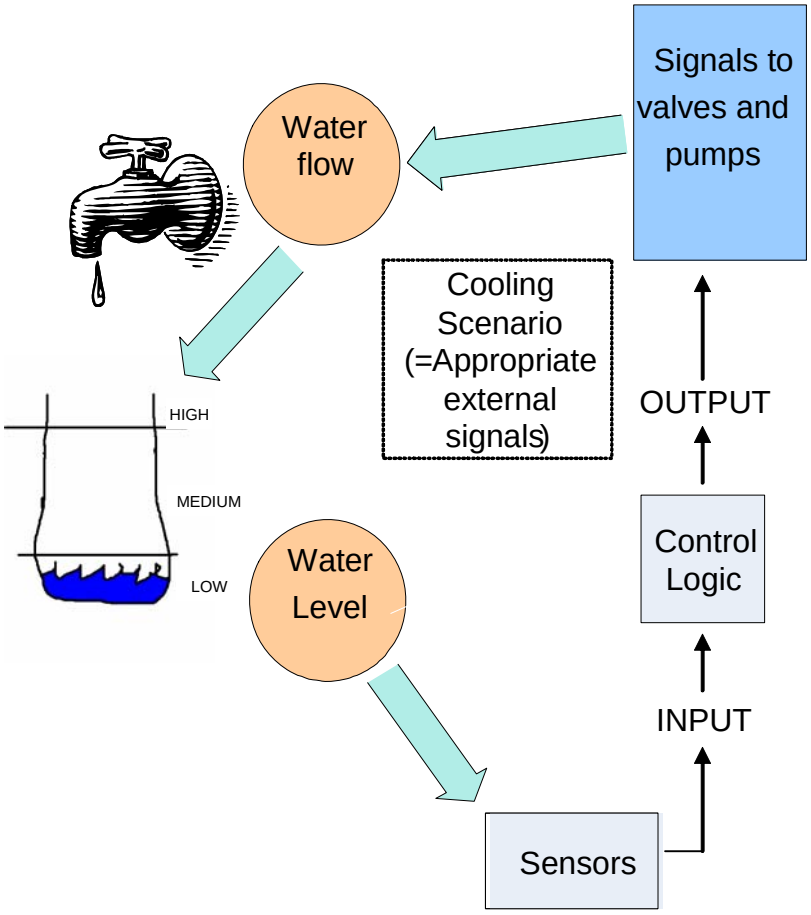

Figure 1. Functional principle of the emergency cooling system

4.2 Model of the System and Specifications

The model was created mainly based on safety assessment reports and flow charts, and the model concentrated on a detailed description of the control logic. The model covers all the automatic functions and delays of the system excluding only a couple of parts that were considered irrelevant to the analysis. All four redundant units of the system were included in the model.

Besides the control logic, the most relevant functions connected to it were modelled. A function was considered relevant if it directly or indirectly affected the pumping process, that is, if it could change the input signals of the control logic. The model included several pumps and valves, and the water level.

The water level of the reactor container was described by a state variable having three possible states: “high”, “medium” and “low”. The sensors connected to the control logic only measure water level from two positions and thus three states were sufficient. The model was based on the following principles: (i) If no water is flowing into the reactor container, the water level can remain the same, or decrease. (ii) If water is flowing in, then the water level can remain the same, or rise.

The specifications analysed included the behaviour of the control logic in certain emergency scenarios. The specifications were formalised with linear temporal logic.

4.3 Results of Model Checking

The number of different states of the model is about $1.3*10^{18}$ which is the product of the number of different states of the variables in the model (number_of_states_of_variable_X1* number_of_states_of_variable_X2*number_of_states_of_variable_X3…). For the experiments, a standard PC and NuSMV ver. 2.4.2 with BDD-based LTL model checking was used. The computing time for model checking of a single requirement was a few seconds with a standard workstation.

Some initial violations of the specifications suggested by the model checker were due to the simplifications made in the model and were removed by revising the model. The invalid behaviour could be handled by inserting additional conditions which prevented the model from entering unrealistic states. The behaviour of the model was also restricted by so-called fairness constraints to avoid situations where some component could remain in a transition state. For instance, a valve cannot remain in a state "opening" forever.

No violations of the specifications were found. Some specifications were checked in order to evaluate the method and some to analyse how well the model described the actual system. The results showed that the model described the behaviour of a real system accurately enough and that model checking could be utilised in the verification of the specifications.

Certain properties were checked for the verification of the basic operating principles. For instance, it was checked that in a normal operating state, a low water level in the reactor container always leads to water inflow to the reactor container (in some future state). Normal operating state means that the physical conditions, like pressure in the container, are favourable and that the control signals allow the operation of the system. It was also noticed that the pumps might occasionally remain pumping against closed valves. After more specific investigations, the action of the pumps proved to be planned behaviour and was due to a signal which had the pumps operating even though the valves were closed. With that same signal forced to 0, the pumps never remained pumping against closed valves.

5. CONCLUSIONS

Model checking tools typically offer a finite state machine-based modelling language for modelling the system to be verified, a specification language based on temporal logic for expressing the properties to be verified and a set of analysis tools to check that the system satisfies the given properties. We employed a state-of-the-art open-source model checking system, NuSMV, and with reasonable effort we were able to (i) model both systems on an adequate level, (ii) formulate required safety properties in the specification language and (iii) perform a full verification of the properties using the NuSMV system. Thus, the current model checking techniques are applicable in the analysis of safety I&C systems in NPPs.

These results show the potential and power of model checking and give a good basis for future research. After making the model, it is rather easy to check different scenarios and see how small changes in the signals and conditions change the behaviour of the model.

Model checking seems to be directly usable for verifying designs of safety I&C systems. An advantage of this approach to more traditional testing and simulation work is that it can provide full coverage of the verification. When model checking system properties, it is often necessary to model the system environment to some degree. Fortunately, modelling languages supported by model checking tools are quite usable for capturing the environment and it is possible to create simple models covering the essential behaviour of the environment.

Both modelled systems included timing aspects, especially delays, which seem to be crucial in many safety I&C systems and which are also very challenging to design and verify. This is an area where more work is needed for developing robust design and verification techniques for safety systems where delays are exploited. The challenges for future research also include the development of more dedicated methods for the verification of safety critical embedded software.

6. REFERENCES

[1] SAFIR2010, The Finnish Research Programme on Nuclear Power Plant Safety 2007 – 2010, http://www.vtt.fi/safir2010, 2008

[2] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. Model Checking. The MIT Press, 1999.

[3] J.P. Quielle and J. Sifakis. Specification and verification of concurrent systems in CESAR. In Proceedings of the 5th International Symposium on Programming, pages 337-350, 1981.

[4] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization of skeletons using branching time temporal logic. In Proceedings of the IBM Workshop on Logics of Programs, volume 131 of LNCS, pages 52-71. Springer, 1981.

[5] B. Berard, M. Bidoit, A. Finkel, F. Laroussinie, A. Petit, L. Petrucci, and Ph. Schnoebelen. Systems and Software Verification. Model-Checking Techniques and Tools. Springer, 2001.

[6] A. Cimatti, E. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, Marco Roveri, R. Sebastiani, and Armando Tacchella. NuSMV 2: An opensource tool for symbolic model checking. In CAV'02, volume 2404 of LNCS, pages 359-364. Springer, 2002.

[7] Randal E. Bryant. Symbolic Boolean manipulation with ordered binary decision diagrams. ACM Computing surveys, 24(3):293-318, 1992.

[8] Jerry R. Burch, Edmund M. Clarke,Kenneth L. McMillan, David L. Dill, and L. J. Hwang. Symbolic model checking: 10^20 states and beyond. Information and Computation, 98(2):142-170, 1992.

[9] Janne Valkonen, Ville Pettersson, Kim Björkman, Jan-Erik Holmberg, Matti Koskimies, Keijo Heljanko, and Ilkka Niemelä. Model-Based Analysis of an Arc Protection and an Emergency Cooling System – MODSAFE 2007 Work Report. VTT Working Papers 93, 2008.

[10] Janne Valkonen, Ilkka Karanta, Matti Koskimies, Keijo Heljanko, Ilkka Niemelä, Dan Sheridan, and Robin E. Bloomfield. NPP Safety Automation Systems Analysis - State of the Art. VTT Working Papers 94, 2008.