VIRTUALIZED HARDWARE ENVIRONMENTS FOR SUPPORTING DIGITAL I&C VERIFICATION

Frederick E. Derenthal IV, Carl R. Elks, Tim Bakker, Mohammadbagher Fotouhi
Department of Electrical and Computer Engineering, Virginia Commonwealth University (VCU)
601 West Main Street, Richmond, Virginia 23681
derenthalfe@vcu.edu; crelks@vcu.edu; bakkert@vcu.edu; foutouhimb@vcu.edu

ABSTRACT

The technology of virtualized hardware testing platforms within the nuclear context in order to provide a framework for evaluating virtualized hardware/software approaches is discussed. To understand the applicability of virtualized hardware approaches, we have developed a virtual hardware model of an actual smart sensor using the Imperas product OVPsim development environment, as is seen in [1]. An evaluation of the modeling effort required to create the digital I&C smart sensor is provided, followed by an analysis and findings on the various tools in the OVPsim environment for supporting testing, evidence collecting, and analysis. The analyzed tools include traditional input/output testing, model-based testing, mutation testing, and fault injection techniques. Finally, the goal of this research effort is to analyze virtual hardware platforms and present findings to the nuclear community on the efficacy and viability of this technology as a complementary means to actual hardware-in-the-loop testing Factory Acceptance Testing for Nuclear Power Plant (NPP) digital I&C systems.

Key Words: Virtual Hardware, Virtual Modeling, Verification, Validation

1 INTRODUCTION

Much of the instrumentation and control (I&C) equipment in operating U.S. NPPs is based on very mature, primarily analog technology that is steadily trending toward obsolescence. The use of digital I&C systems for plant upgrades, particularly in NPP safety systems, has been slow primarily due to the fact that uncertainties, such as (e.g. failure modes, complexity, and unknown interactions) with digital I&C systems are not well characterized by historical data. On the other hand, model-based engineering and verification methods and practices for embedded safety systems have transformed other safety critical industries in a positive way – supporting evidence based verification methodologies to address systematic failures or Common Cause Fault (CCF). Because of the size and complexity of real-world systems, model-based engineering has evolved to be tool-supported. Tools like MathWorks Simulink, ANSYS SCADE, and AREVA SIVAT are but a few examples of model-based engineering tools.

In model-based testing, a model of the desired behavior of the system under test (SUT) is the starting point for the assessment. The models are used to help the tester understand functional interactions, relationships, and behaviors that the device should have based on its requirements. These models typically include a functional model of the SUT, a requirements/specification model that captures the functionality of the SUT, and an input model that represents the nominal input sets and anticipated erroneous input sets. An important aspect of the model-based testing or verification of software that is often overlooked, but is nonetheless crucial, is the accurate representation of the digital I&C hardware that executes the software. It is important to capture the hardware/software interactions in complex digital I&C systems in order to correctly model the functionality of the system safety functions. Until recently, testers typically used actual hardware articles to represent the underlying hardware. While using actual hardware in testing represents the most accurate representation possible, there are challenges with hardware-in-the-loop. Namely, using

actual hardware is inefficient in terms of testing concurrent articles, has limited observability and controllability for observing the error pathology of software functions, and is expensive in time and cost.

The reported preliminary research in this paper is to evaluate a relatively new technology in model-based engineering called "Hyper-Accurate Virtual Hardware" to support the testing and verification of software components of digital I&C systems. Virtualized hardware platforms and environments provide a means to develop micro-architecture representations of the embedded digital I&C system that execute the same binary code as would run on the actual hardware – so called hyper-accurate. One of the claimed benefits of virtualized hardware is the enhanced controllability and observability of the hardware/software system representation through advanced debug features which allows for the detailed analysis of execution behaviors. In addition, virtualized testing environments allow for plant models to be in the loop, thus placing the testing into plant context. While other industries have been using virtualized hardware platforms for the testing of safety critical systems, as seen in [2], there is little research or results regarding the use of virtualized hardware platforms in the NPP community. The use of virtual models of hardware/software digital I&C systems introduces the ability to communicate testing artifacts, concerns, assumptions and evidence to stakeholders (regulatory staff) via portable executable models to support safety arguments of the digital I&C systems.

2 VIRTUALIZED PLATFORMS

Virtualized platforms resemble accurate representations of actual hardware articles, and provide a variety of advantages when compared to, or used as a complementary means to, the current practice of testing hardware in-the-loop. The various concerns associated with the licensing of nuclear I&C devices are discussed in this section, including the observed issues related to verification and validation. Additionally, this section outlines the incorporation of the use of virtualized hardware platforms for testing within the industrial domain. The advantages and limitations of using virtualized hardware when compared to testing with actual hardware-in-the-loop are analyzed, and the observed experiences thus far, with regard to using virtualized hardware simulators for supporting the verification and validation and licensing of embedded systems, are reported.

2.1 Common Concerns

Nuclear I&C devices in particular require exhaustive techniques with regard to verification and validation, for obvious reasons, including the assurance of safety properties. The licensing of typical nuclear I&C hardware devices is a stringent, intricate process that takes a large amount of time and clear communication between design and manufacturing organizations, and governing bodies within the nuclear domain. This process becomes more complex when software is introduced, and even more convoluted when models of hardware executing software is introduced, as is the case for virtualized models. Safety properties must be ensured, primarily with regard to requirements and specifications, to include the operational profile context of the evaluated hardware or software, in order to guarantee that the article being evaluated is suitable for operation within the NPP domain.

A variety of verification and validation techniques exist for testing the hardware/software article, which are typically intensive and used in an exhaustive manner, usually including some form of formal verification technique. The process of licensing and validating and verifying a hardware/software article is expensive in time and computational cost, but is necessary within the nuclear power domain for obvious safety-related reasons. The same processes may be applied to virtual platforms in order to ensure that the appropriate requirement, specification, and property metrics remain valid throughout the testing process. Accurate virtual modeling of the hardware and software is the first step toward applying this technique to a nuclear digital I&C system, followed by various testing techniques for verification and validation assurance. As previously mentioned, the models allow the tester the ability to understand the functional interactions, relationships, and behaviors that the device should have based on its requirements. Ensuring

that the model is accurate and complete according to the reference hardware/software aids in capturing the hardware/software interactions in complex digital I&C systems, which subsequently assists in correctly modeling the functionality of system safety functions.

2.2 Benefits and Limitations of Virtualized Platforms

While using actual hardware articles in testing to represent the underlying hardware supplies the most accurate representation possible, there are several challenges with using hardware-in-the-loop for testing. Particularly, testing using actual hardware-in-the-loop is inefficient in terms of testing concurrent articles, the approach has limited observability and controllability for observing the error pathology of software functions, and is expensive in time and cost. Using virtual platforms as an alternative means to testing helps to overcome these obstacles. Virtual hardware modeling offers several benefits with regard to testing, efficiency, and verification and validation techniques. Given that the virtual platform is an accurate representation of the hardware article being tested, which is the primary challenge of the virtual platform approach, testing may proceed in the same manner as would be for the actual hardware article. Virtual platforms are able to produce quick, repeatable results, and input parameters can easily be varied and simulated rapidly using scripting techniques. The simple adjustment and reusability of virtual platforms alleviates the time and cost obstacles associated with actual hardware-in-the-loop testing. Additionally, these abilities simply cannot be accomplished in an efficient, feasible manner solely using actual hardware.

Various testing tools exist for increasing the observability and controllability of the virtual platform. The use of virtual platform emulators supplies the user with enhanced observability and controllability of the hardware/software system representation, specifically by applying simulation manipulation commands such as tracing, adding breakpoints, and the ability to inspect and control the platform during simulation. These tools allow for the straightforward and smooth collection of data for evaluation and the detailed analysis of execution behaviors. In addition, virtualized testing environments allow for plant models to be in the loop, thus placing the testing into plant context. Another noted benefit of testing with virtual platforms is the capability of automating tests, configurations, and injections, which requires computational cost and human effort when practiced with actual hardware articles. Lastly, the use of virtual models of hardware/software digital I&C systems introduces the ability to communicate testing artifacts, concerns, assumptions and evidence to stakeholders (regulatory staff) via portable executable models in order to support safety arguments of the digital I&C systems.

The limitations associated with the use of virtualized platforms lie almost exclusively in the design process of creating the virtual platform model. This is due to the fact that actual testing of the virtual platform is done in a simulated environment, which is a fairly straightforward process. With regard to the design process, designers must be in constant communication with device manufacturers in order to ensure that the virtual hardware being modeled is an accurate representation of the actual hardware used in the operational context. Additionally, the operational context and requirements and specifications of the device must be included in the design of the virtual hardware executing the associated software, in order to ensure that the most realistic model of the hardware is represented as it is used in the real world. The use of virtual platform emulators ensures that, given the correct design parameters, the virtual platform is an accurate representation of the hardware that it is modeled after. Careful attention must be paid when researching the specific components, requirements, and specifications in order to design an accurate virtual model of the referenced hardware and software.

2.3 Evidence for Virtual Platform Application

Currently, industrial evidence exists for the use of virtual digital I&C systems for supporting licensing and verification and validation. The first approval of simulation validation testing for digital I&C systems is seen in [2]. AREVA is the first and only organization thus far to receive approval from the Nuclear Regulatory Commission (NRC) for a full plant application of a safety-related digital I&C system (TELEPERM® XS) and for a simulation validation testing tool, SIVAT. The AREVA TELEPERM® XS

digital I&C system is a state-of-the-art, fully digital system, used to monitor and control the processes and equipment of a nuclear plant, and is expected to improve overall plant reliability and performance. The SIVAT tool is the first safety-related software simulation tool approved by the NRC to perform formal validation testing on digital I&C systems, and will enable nuclear utilities to more accurately and easily test systems. The result of the application of this tool in the nuclear domain contributes to the operational context while significantly reducing risks associated with the licensing, installation, and operation of a digital I&C upgrade system.

Model-based engineering and verification methods and practices for embedded safety systems have successfully transformed various safety critical industries in a positive way. The use of virtual platforms for certification and verification and validation purposes have been applied to the industry standards in [3] and [4] for the IEC 61508 and the ISO 26262 industries, respectively. The results of the application of virtual platforms in these domains has led to improved requirement specifications for product development at the software level, to include functional software safety requirements (and verification of these requirements), software unit design, implementation, and integration and testing. The observed effects of applying virtual platforms to these industries supports evidence based verification methodologies and aids in the licensing and verification and validation of digital I&C systems.

As previously noted, the application of virtualized hardware has been tested in various industrial domains with empirical, documented results. Although there is little research or results regarding the use of virtualized hardware platforms in the NPP community, the same methodology and processes may be applied within the nuclear domain, for the verification of digital I&C systems. Using the requirements, specifications, and operational context information specific to the NPP domain, sufficient testing fidelity may be assessed with various testing techniques. The approach used by our research effort will be outlined in the following section, and various testing techniques will be described with regard to assuring the aforementioned level of fidelity and other analyzed testing criteria, such as coverage and safety properties. Additionally, using the established virtual platform industry examples previously mentioned, as well as tools such as AREVA SIVAT, the application of virtual platform modeling and testing may be transitioned to include devices within the nuclear power domain in a smooth manner.

3 THE SMART SENSOR

To sufficiently validate the approach of using virtual hardware models in general, we have developed a virtual smart sensor hardware platform using the Imperas virtual platform emulator OVPsim. The functionality of OVPsim and the associated modeling process will be discussed more thoroughly in the following section. Using the smart sensor hardware article, a virtual model has been created and is currently being tested in order to validate our approach. Optimistically, the virtual smart sensor platform will serve as a means of guidance for applying the approach within the nuclear context, using hardware articles commonly found in the NPP community.

The smart sensor prototype, originally developed by the VCU Unmanned Aerial Vehicle (UAV) laboratory, is a temperature and pressure measurement device. The prototype represents a sensor that would be found in the digital I&C system of a nuclear power plant. The actual smart sensor that our virtual model represents is capable of measuring pressure and temperature data, advanced calibration and factory characterization, self-checking for the confirmation of calibration and communication parameters, advanced filtering and analysis of input and output data, and support for various output formats. A brief overview of the structure of the actual smart sensor hardware and software, which has been used as a reference for our virtual model, is now discussed.

3.1 Smart Sensor Architecture

The smart sensor developed by the VCU UAV laboratory consists of hardware and software components, including an internal operating system known as ChibiOS. The hardware element of the smart

sensor includes an ARM Cortex M4 processor, several memory components and buses, and a variety of communication peripherals for the communication of data and simulation instructions, among other components. The smart sensor hardware model is seen in Fig. 1.

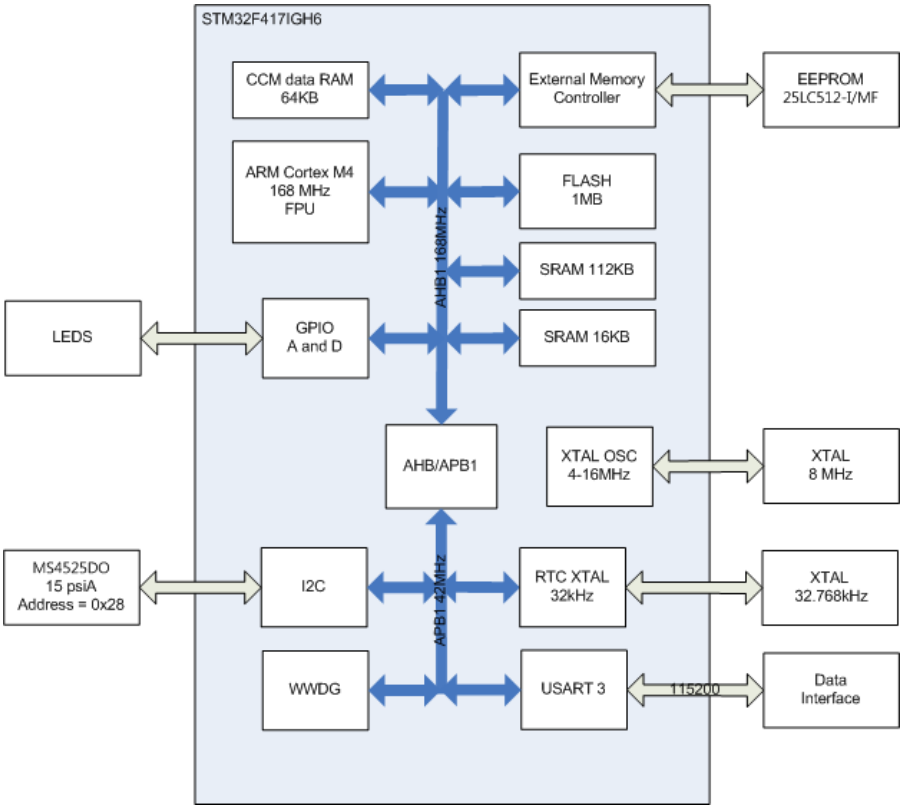

Figure 1. Smart Sensor Hardware Model

The software aspect of the smart sensor includes a real-time operating system, ChibiOS, which provides the platform with predictable and deterministic execution behavior. Additionally, the operating system includes a variety of device drivers for well-known industry interfaces as well as a clear and concise application programming interface (API). Collectively, these aspects allow for the quick integration and development of a target application. The software layers for the smart sensor include the user application layer, layers for the sensor and hardware drivers, and the ChibiOS kernel layer. The collection of smart sensor software layers is displayed in the smart sensor software model, which is seen in Fig. 2.

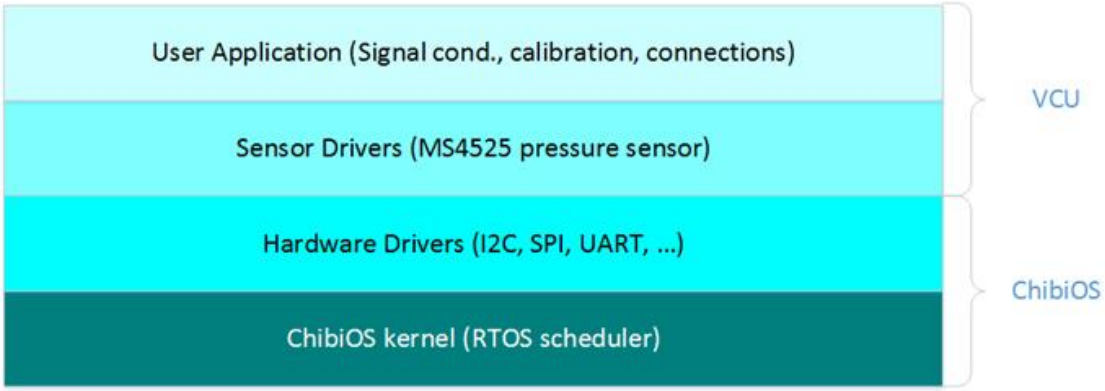

Figure 2. Smart Sensor Software Model

The development of the virtual smart sensor platform is relatively straightforward and will be discussed comprehensively in the following section. Following the development of the virtual platform, various testing techniques and methods may be applied for verification and validation purposes. The process of developing a virtual platform, testing, and analyzing results may be applied to a verified industry-standard embedded system, which is the overall goal of this research effort. Future work with regard to the specific testing of the smart sensor is discussed further throughout the remainder of this paper.

4 VIRTUALIZED HARDWARE SIMULATORS

Several virtual platform emulators exist for the virtual modeling and testing of actual hardware articles. Virtual platform emulators operate as full system hardware simulators and maintain the ability to execute at real-time or near real-time with instruction cycle accuracy. The abilities presented by virtual platform emulators grant the user the ability of virtually modeling hardware as closely as possible to the actual hardware article that is being represented. Additionally, virtual platform emulators include the capability of simulating embedded systems running real application code, which ensures that models are an accurate representation of the actual embedded system being analyzed, in their respective operational context. The process of virtual modeling and testing using virtual platform emulators can be easily applied to digital I&C systems within the nuclear power domain to aid in the support of verification and validation.

4.1 The OVPsim Simulator

The Imperas product OVPsim was chosen as the virtual platform emulator to be used specifically for the smart sensor application for various reasons when compared with several other virtual platform emulators. OVPsim is a commercial tool that is capable of virtual platform and modeling technology. There are three main components of virtual platform emulators: The APIs that enable a model to be written using application code, a library of previously generated models for reference, and a simulator that executes the created model. The primary focus of OVPsim is to accelerate the adoption of the new style of developing embedded software by means of virtual modeling., a focus which is shared by our approach.

The process of platform development in OVPsim is relatively straightforward: The user creates a simulation model, such as a platform, using application code. The application code is then compiled into an executable with the option of connecting it to a debugger, which provides a very efficient embedded software development environment. After successful compilation and execution of the program, a variety of APIs and debuggers are available for further testing and simulation. An example of the OVPsim platform creation and testing process, in comparison with the process of testing using real hardware, is seen in Fig. 3.

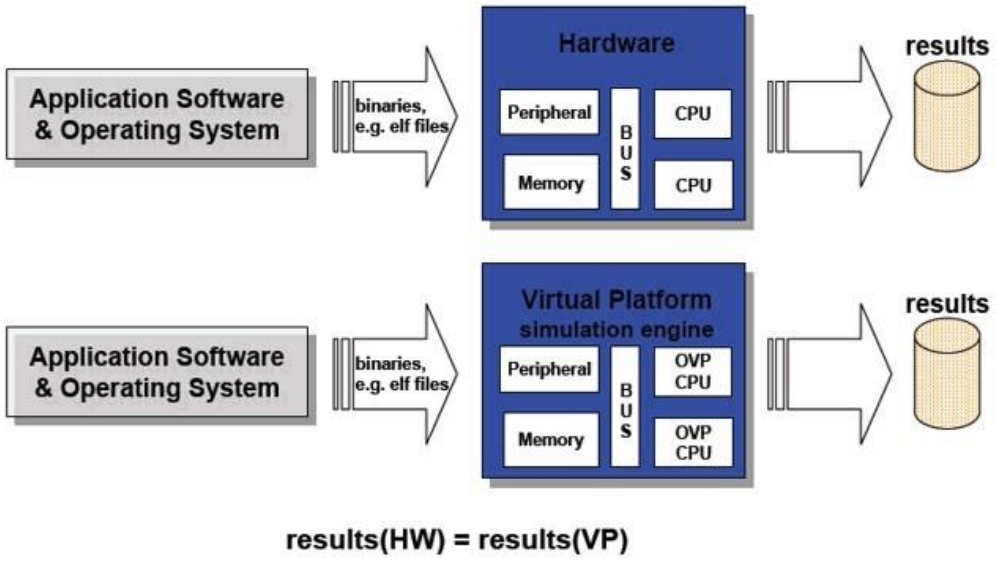

Figure 3. OVPsim Virtual Platform Creation and Testing Process, in Comparison with the Process of Testing Using Actual Hardware

It is worth noting that the virtual platform and hardware components seen in Fig. 3 are identical in nature, only differing in the fact that the CPU components of the virtual platform are OVP-specific. In other words, the virtual platform is an accurate virtually generated (modeled) representation of the actual hardware being tested. One of the specific advantage of using OVPsim is the ease of providing models to others for testing in external simulation environments. This allows for collaborative and cooperative efforts between design and testing organizations, and, when applied in the nuclear context, assures the accurate representation of the model and testing parameters in the operational context. These methods further support the verification of digital I&C systems within the NPP domain. OVPsim offers a variety of debugging tools, APIs, and software verification, analysis, and profiling (VAP) tools that assist in analyzing the simulated data. The built-in and external testing tools that facilitate testing in OVPsim will be discussed more intensively in the following subsections.

The overall process of developing and testing a virtual model includes researching the hardware and software elements that the actual hardware article is comprised of, creating the model in application code, generating an executable, and testing the final model with various testing techniques for the verification and validation of application-specific criteria. The entire process is being combined into a single environment with a graphical user interface (GUI) front-end for ease-of-use. The environment is currently in development as a part of our research approach, and will include automation principles for the efficient testing of the model. Once finished, the environment will only require an input model in order to test and generate results. More specifically, once developed, this process can be utilized very easily within the NPP domain. The same aforementioned processes and methods applied for the smart sensor will then be able to be applied to any model, primarily those currently in use in the nuclear context. A similar example to our approach is seen in [5], which displays the creation of a virtual platform for the model-based design of software for a dependable cyber-physical system. A more detailed analysis of the aforementioned development and testing process, with strict regard to the smart sensor application, may be found in [6].

4.2 Testing Methods

Following the development of the virtual platform, testing techniques such as model-based testing, mutation testing, and fault injection methods may be applied for a thorough analysis of the platform and the corresponding input/output data. The goal of applying the aforementioned testing techniques is to achieve substantial verification and validation results, primarily with regard to addressing the problems and concerns associated with the licensing of nuclear digital I&C systems. The testing methods will be briefly outlined in this section in order to understand the applicability of these methods in aiding the support of the use of digital I&C systems in the NPP community. The definitions and specific applications of each of the respective testing techniques are beyond the scope of this paper. However, examples, related tools, and associated difficulties with regard to each of the respective methods may be found in the relevant literature.

Many suggested model-based testing strategies exist for the smooth and efficient development, design, and testing of models. A variety of these approaches are found in [7], [8], [9], [10], and [11]. More specific tools for the analysis of data using the model-based testing approach may be found in [12], [13], [14], [15], [16], [17], [18], and [19]. Simulink provides an alternative model-based tool approach, with their tool-sets included in the Simulink Design Verifier environment and the MathWorks tool chain. The tool-sets consist of precise verification and validation tools that use formal methods to ensure the verification of requirements and test coverage metrics. The various difficulties associated with model-based testing are discussed in [15].

Mutation testing methods may be applied to the source code, software requirements, and design specifications of the virtual platform model. The testing of software requirements and design specifications support the applicability of the simulated model in a practical way by ensuring that requirements and constraints in the operational context remain valid both during and after the simulation execution. The application of real-world requirements and specifications can be easily applied to the established nuclear context and associated digital I&C systems. Various examples of the mutation testing of hardware and software in general are seen in [20], [21], [22], [23], [24], [25], and [26]. As within the model-based testing domain, obstacles exist within the mutation testing domain, and further discussion of these challenges may be found in [27] and [28].

Various types of fault injection methods exist for the testing of hardware and software in order to determine the efficiency and effectiveness of the system fault detection and fault tolerance mechanisms, with respect to a set of hypothesized faults and the operational context of the SUT. Fault injection may be applied both on the physical hardware and the virtual emulation of the platform in order to compare results, further ensuring that the model is an accurate representation of the hardware being modeled in the operational context. The focus of using fault injection on virtual platforms is to inject defects associated with requirement flaws in the design of the system operating in its respective context. Fault injection campaigns may be invoked at any time during the design and testing process, allowing for a full analysis of the virtual platform and its associated functionality. An example of fault injection in virtual platforms and the associated effects is seen in [5].

Lastly, a combination of the distinct testing methods may be applied in order to assess and improve the software dependability, including the assurance of verification and validation metrics. Co-analysis techniques, such as the combination of fault injection and mutation testing for example, are seen in [28], [29], and [30].

4.3 Smart Sensor Testing Approach

The smart sensor virtual platform developed specifically for our application will be subject to a variety of the aforementioned testing methods. More specifically, the smart sensor will initially be tested using various model-based testing methods and mutation testing. Model-based testing methods will be utilized for the tracing of requirements and design specifications during simulation, in order to ensure the correct operation and reliability of various testing metrics, such as safety properties, for example. In addition to

applying the model-based testing methods to the virtual model, a strong mutation testing technique will be applied to the source code of the virtual hardware in order to identify any observable differences between the original model and its associated variants.

VCU is currently developing a testing process framework which will encompass all of the testing methods and criteria described thus far into a single environment for ease-of-use. Using the appropriate scripting techniques, a user will be able to provide an input model, and the testing processes will be run, storing all logged data in a database for further data analysis. Included in the testing process framework will be the ability to change files and profile aspects of the testing process. The option of changing the profile aspects of the testing process is a specific benefit for the argument of transitioning to the testing of digital I&C systems within the nuclear domain. With strict regard to the nuclear domain, the user will have the ability to make various parameter changes, set initial conditions, and simulate the component under different profile aspects according to various specific operational context settings, such as low-power mode or high-power mode, for example. From this point, a variety of cost reduction, automation, and efficiency techniques may be applied in order to refine the testing process to a more feasible one.

With strict regard to testing using OVPsim, various commands are built-in for debugging functionality, and OVPsim maintains the ability to hook up to any external debugger that supports the GNU Debugger (GDB) Remote Serial Protocol (RSP). The collective debugging features incorporate commands for tagging, tracing, adding break-points, and various simulation manipulation techniques. Additionally, OVPsim supports a wide variety of devices to aid in the debugging and testing of the virtual platform. Two specific examples of options for further analysis include the GUI-based source code debugging of virtual platforms and embedded software, iGui, and the VAP tools mentioned earlier. Both of these tools are included with the original OVPsim package and may be invoked during simulation to perform a number of tasks. The virtual platform emulator tools may be turned on and off from the platform definition, the external interactive debugger, or from script files. External debuggers such as iGui include commands for inspection and control of the platform and processor state while simulating the application on OVPsim, such as displaying processor register values, stepping of simulation instructions, and setting breakpoints for data collection and evaluation. The VAP tools provide extensive visibility into the booting and running of the software on a virtual platform, which aids developers in identifying functional bugs and examining system performance, profiling, and coverage information. The execution trace of the virtual platform simulation may be analyzed by using the VAP tools in order to evaluate exactly how the platform is executing. The VAP tools provide similar commands as those displayed for the iGui tool, including the ability to trace function calls, line codes, and options for analyzing and debugging programs running on the processor and for the booting and running of the operating system.

Although the aforementioned tools and methods are used explicitly within the OVPsim environment, the concepts may be applied smoothly elsewhere, such as within the NPP domain. The aforementioned tools are used only as examples of the capabilities of testing virtual platforms, and, using an industry-verified embedded system, the same principles and types of tools may be applied to ease the verification and validation of digital I&C systems operating in a nuclear context. One of the noted capabilities of virtual platform testing tools is the ability to trace requirements. Ensuring correct requirement functionality and operation is critical, specifically within the nuclear power domain, in order to ensure the high reliability of safety properties. Supplementary options are available for the instrumentation and automation of testing within the virtual platform emulator environment in general, further facilitating the testing of virtual platforms efficiently for use in verification and validation assurance. Furthermore, various cost reduction techniques exist with respect to each of the previously described testing domains, which may be applied in order to further increase the efficiency of testing virtual platforms in a more practical manner.

5 FUTURE WORK

Minor adjustments must be made both within the scope of the specific approach used for the virtual smart sensor platform and for the application of the virtual modeling and testing process for use within the nuclear domain for the support of verification of digital I&C systems. The required modifications will be discussed briefly in this section.

5.1 Testing Principles

With regard to the specific approach used for the virtual smart sensor platform, there are various areas that require slight attention. A more comprehensive outline of the proposed work for the near future, with related organizations and as part of a larger project, can be found in [6]. As a brief overview, we plan to extend the current testing process to include a more diverse application of the discussed testing methods, such as the inclusion of fault injection for specific operational testing, for example. In addition, we intend to include more definite requirements and design specifications for the smart sensor, according to the nuclear operation context in which the testing process will be applied for our specific application.

The testing principles used in the smart sensor modeling and testing approach may be extended to include various additional aspects of each distinct testing domain. Currently, the strong mutation testing technique is being applied to the source code of the smart sensor. We plan to extend this technique to include weak or firm mutation, which allows for the monitoring and tagging of state data as well as the explicit monitoring and tagging of output data provided by the strong mutation testing technique. Additionally, the monitoring of nominal system behavior may be extended with new fault behaviors to yield an extended system model. Automated or tool assisted analysis will be applied to the virtual smart sensor platform for the generation of safety artifacts, such as fault trees, from the extended system model. Co-analysis techniques such as those previously described will also be applied to the smart sensor in order to attain highly reliable coverage metrics and ensure that the smart sensor is as accurate a reflection as possible to the actual hardware article that it is modeling. Lastly, we plan to refine the previously described testing process framework to include automated functionality and execution for the rapid testing of multiple versions of models according to distinctly defined parameters and operational context settings.

In order to apply the proposed testing techniques to the smart sensor and to the virtual platform testing domain in general, a unified view of the disparate testing techniques must be constructed. By surveying the state of the art in all of the aforementioned testing method domains, common threads and integration principles to support the testing of virtual platforms must be established and applied to the model under test. The application of these principles will aid in the design and development of a more concrete testing process which supports these testing methods. The established methodology, shaped around the more complete testing process, may then be applied to a variety of operational contexts, to include the NPP domain. Ideally, the testing process and integration principles will also incorporate the virtual platform emulator environment, in our case, OVPsim, to include the associated tools and models.

5.2 Nuclear Domain Application

The smart sensor prototype developed in OVPsim is intended to be used as a guide in order to extend the virtual modeling and testing process of hardware in general. More specifically, using the process outlined in this paper, we plan to apply the same techniques and methodology to an industry-verified nuclear I&C system. Using the various verification and validation approaches mentioned, the verification of the virtual model of any verified industry-standard nuclear I&C system will be relatively straightforward, with the only notable differences from the virtual smart sensor platform approach being variations of the parameters, requirements, specifications, and context, and the accurate modeling of particular components of the specific hardware article to be modeled. The same modeling and testing guidelines as used for the virtual smart sensor platform will be followed for a specific nuclear digital I&C system.

An alternative approach is to use an industry-standard nuclear digital I&C system component as a reference model in order to add various functionalities, requirements, design specifications, and components to the current smart sensor according more specifically to the nuclear context. Currently the smart sensor is relatively lightweight, as it does not include diagnostics and consists of only minor calibration configurations. Extending the smart sensor functionality will ensure that it is a more accurate, representative model of the I&C components currently in use in the nuclear domain. Using either a real-world industry-standard I&C embedded system component, or the revised smart sensor platform according to nuclear context parameters and a reference component, the modeling process outlined in this paper will serve as a guideline for the testing methodology and application of verification and validation techniques for virtual platforms. Ideally, the testing methodology will serve as a proof-of-concept method for assuring that virtual platforms are an efficient, practical means of testing, and ensure that virtual platforms may serve as a complement to the current practice of testing using hardware-in-the-loop.

6 CONCLUSIONS

We have effectively evaluated the current status of virtualized platforms in the nuclear domain, and analyzed the effects of applying the technology compared with the current practice of testing using hardware-in-the-loop, including both the benefits and limitations of each approach. The practical applicability of virtual platforms has also been introduced, displaying evidence provided from the application of the technology in multiple industrial domains. The process of modeling and testing has been applied to the smart sensor prototype in order to serve as a guide for the virtual modeling and testing of hardware/software in general. The modeling and testing of virtual platforms using virtual platform emulators, namely the OVPsim simulator, has been examined and various testing methods for the collection and analysis of data has been analyzed. Finally, proposed work has been described with regard to applying the technology of virtual hardware modeling in a more practical, realistic approach and improving testing methods for the efficient modeling and testing of virtual platforms.

Virtual hardware platforms have been analyzed and the findings have been presented with regard to the efficacy and viability of this technology as a complementary means to actual hardware-in-the-loop testing Factory Acceptance Testing for NPP digital I&C systems. By applying the methodologies used in this paper as a guide, the transition from modeling and testing the smart sensor prototype to the modeling and testing of actual industry-verified digital I&C embedded systems can become a realistic, feasible possibility. The testing process and environment utilized in our methodology provides a straightforward approach to virtually modeling hardware systems in general for testing, extending the generality of this work. Additionally, the processes and methods described in this work may be applied in the nuclear domain in order support the testing and verification of virtual models of digital I&C systems as a complementary means to actual hardware-in-the-loop testing.

7 ACKNOWLEDGMENTS

The material presented in this document incorporates work applied to a larger project supported by the Department of Energy. We would like to thank several research organizations and institutions for their contributions and collaboration efforts with VCU, including the University of Tennessee: Dr. Richard T. Wood, and Ohio State University: Dr. Carol Smidts and Boyuan Li.

8 REFERENCES

1. “Open Virtual Platforms™ (OVP™) Website,” http://www.ovpworld.org/ (2017).
2. “NRC Approval of New Simulation Validation Test Tool for Teleperm Digital I&C System: SIVAT,” http://us.areva.com/EN/home-1950/nrc-ic-simulation-approval.html (2012).

3. "Functional Safety and IEC 61508," http://www.iec.ch/functionalsafety/ (2017).
4. "ISO 26262-6:2011 Road Vehicles – Functional Safety – Part 6: Product Development at the Software Level," https://www.iso.org/standard/51362.html (2011).
5. M. Becker, C. Kuznik, and W. Mueller, "Virtual Platforms for Model-Based Design of Dependable Cyber-Physical System Software," *2014 17th Euromicro Conference on Digital System Design*, Verona, pp.246-253 (2014).
6. R. Wood et al., "Development and Demonstration of a Model Based Assessment Process for Qualification of Embedded Digital Devices in Nuclear Power Applications First Annual Progress Report," (2016).
7. A. Sinha and C. Smidts, "HOTTest: A Model-Based Test Design Technique for Enhanced Testing of Domain-Specific Applications," *ACM Trans. Softw. Eng. Methodol.*, **vol. 15**, pp.242-278 (2006).
8. A. Sinha and C. Smidts, "An Experimental Evaluation of a Higher-Ordered-Typed-Functional Specification-Based Test-Generation Technique," *Empirical Softw. Eng.*, **vol. 11**, pp.173-202 (2006).
9. A. D. Brucker and J. Julliand, "Editorial for the Special Issue of STVR on Tests and Proofs Volume 2: Tests and Proofs for Improving the Generation Time and Quality of Test Data Suites," **vol. 24**, pp.591–592 (2014).
10. A. C. D. Neto, R. Subramanyan, M. Vieira, and G. H. Travassos, "A Survey on Model-Based Testing Approaches: A Systematic Review," *1st ACM international workshop on Empirical assessment of software engineering languages and technologies: held in conjunction with the 22nd IEEE/ACM International Conference on Automated Software Engineering (ASE) 2007*, New York, NY, USA, pp.31–36 (2007).
11. S. R. Dalal et al., "Model-Based Testing in Practice," *Proceedings of the 21st International Conference on Software Engineering*, New York, NY, USA, pp.285-294 (1999).
12. A. Hartman and K. Nagin, "The AGEDIS Tools for Model Based Testing," *Proceedings of the 2004 ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA '04)*, New York, NY, USA, pp.129-132 (2004).
13. J. Tretmans and E. Brinksma, "TorX: Automated Model Based Testing," *First European Conference on Model-Driven Software Engineering*, Nuremberg, Germany, December 11-12, pp.31-43 (2003)
14. M. Shafique and Y. Labiche, "A Systematic Review of State-Based Test Tools," *Int. J. Softw. Tools Technol. Transf.*, **vol. 17**, pp.59–76 (2015).
15. V. Pantelic et al., "A Toolset for Simulink: Improving Software Engineering Practices in Development with Simulink," *2015 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD)*, Angers, pp.1-12 (2015).
16. K. Ghani, J. A. Clark, and Y. Zhan, "Comparing Algorithms for Search-Based Test Data Generation of Matlab® Simulink® Models," *2009 IEEE Congress on Evolutionary Computation*, Trondheim, pp.2940–2947 (2009).
17. M. W. Whalen, A. Murugesan, S. Rayadurgam, and M. P. E. Heimdahl, "Structuring Simulink Models for Verification and Reuse," *Proceedings of the 6th International Workshop on Modeling in Software Engineering (MiSE 2014)*, New York, NY, USA, pp.19-24 (2014).
18. Y. Zhan and J. A. Clark, "A Search-Based Framework for Automatic Testing of MATLAB/Simulink Models," *Journal of Systems and Software*, **vol. 81**, pp.262–285 (2008).
19. G. Fraser, F. Wotawa, and P. E. Ammann, "Testing with Model Checkers: A Survey," *Softw. Test. Verif. Reliab.*, **vol. 19**, pp.215-261 (2009).

20. K. Maruchi, H. Shin, and M. Sakai, "MC/DC-Like Structural Coverage Criteria for Function Block Diagrams," *2014 IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops*, Cleveland, OH, pp.253-259 (2014).
21. H. Chockler, D. Kroening, and M. Purandare, "Computing Mutation Coverage in Interpolation-Based Model Checking," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **vol. 31**, pp.765-778 (2012).
22. P. Ammann, W. Ding, and D. Xu, "Using a Model Checker to Test Safety Properties," *Proceedings Seventh IEEE International Conference on Engineering of Complex Computer Systems*, Skovde, pp.212-221 (2001).
23. G. Fraser and F. Wotawa, "Using Model-Checkers for Mutation-Based Test-Case Generation, Coverage Analysis and Specification Analysis," *Software Engineering Advances, International Conference on*, Tahiti, pp.16-16 (2001).
24. A. Abbasinasab, M. Mohammadi, S. Mohammadi, S. Yanushkevich, and M. Smith, "Mutant Fault Injection in Functional Properties of a Model to Improve Coverage Metrics," *2011 14th Euromicro Conference on Digital System Design*, Oulu, pp.422-425 (2011).
25. S. Nica and F. Wotawa, "EqMutDetect – A Tool for Equivalent Mutant Detection in Embedded Systems," *Proceedings of the 10th International Workshop on Intelligent Solutions in Embedded Systems*, Klagenfurt, pp.57-62 (2012).
26. G. Fraser and F. Wotawa, "Mutant Minimization for Model-Checker Based Test-Case Generation," *Testing: Academic and Industrial Conference Practice and Research Techniques – MUTATION (TAICPART-MUTATION 2007)*, Windsor, pp.161-168 (2007).
27. J. Zhang and S. K. Gupta, "Using Hardware Testing Approaches to Improve Software Testing: Undetectable Mutant Identification," *2016 IEEE 34th VLSI Test Symposium (VTS)*, Las Vegas, NV, pp.1-6 (2016).
28. M. Kooli, A. Bosio, P. Benoit, and L. Torres, "Software Testing and Software Fault Injection," *2015 10th International Conference on Design & Technology of Integrated Systems in Nanoscale Era (DTIS)*, Naples, pp.1-6 (2015).
29. L.T.M. Hanh and N.T. Binh, "Mutation Operators for Simulink Models," *2012 Fourth International Conference on Knowledge and Systems Engineering*, Danang, pp.54-59 (2012).
30. I. Pill, I. Rubil, F. Wotawa, and M. Nica, "SIMULTATE: A Toolset for Fault Injection and Mutation Testing of Simulink Models," *2016 IEEE Ninth International Conference on Software Testing, Verification, and Validation Workshops (ICSTW)*, Chicago, IL, pp.168-173 (2016).