ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 3, May 2012

Development of Testing Environment for Embedded Systems

Miss. Ashwini M. Motghare, Prof. Swapnili P. Karmore

***Abstract* — Quality is the life of embedded systems, and the testing is a basic guarantee for stable and reliable operation of the embedded systems. Testing is an important part in the development of any system as it represents the ultimate verification and validation of specification, design and code. The goal of testing is to design a series of test cases that has the highest likelihood of finding most of the errors with a minimum amount of time and effort. The techniques used to test the embedded systems provide systematic guidance for designing tests that exercise the internal logic of Embedded System components and test the input and output domains of the program to uncover errors in program function, behavior and performance. Simulation can be used as an alternative to the actual target system for a significant portion of the testing effort, saving developers time and money, as well as increasing test coverage and providing better debugging facilities, it becomes much more necessary to construct embedded systems testing environment.**

***Keywords* — Embedded Systems, Simulation Testing Environment, *Embedded system testing environment (ESTE)* .**

I. INTRODUCTION

Testing is an important part in the development of any system as it represents the ultimate verification and validation of specification, design and code. Once source code has been generated, whether it represents hardware or software, the system must be thoroughly tested to uncover as many errors as possible before delivery to the customer. The goal of testing is to design a series of test cases that has the highest likelihood of finding most of the errors with a minimum amount of time and effort. The techniques used to test the embedded systems provide systematic guidance for designing tests that exercise the internal logic of Embedded System components and test the input and output domains of the program to uncover errors in program function, behavior and performance.

Testing is a process centered on the goal of finding defects in a system. It may be for debugging reasons or acceptance reasons – trying to find defects is an essential part of every test process. Although the whole world agrees that it is much better to prevent defects than to find and correct them, reality is that currently unable to produce defect-free systems. Testing is an essential element in system development – it helps to improve the quality of the system. The ultimate goal of testing is to provide the organization with well-informed advice on how to precede – advice based on observed defects related to requirements of the system (either explicitly defined or implicitly assumed). The testing is itself does not directly improve the quality of the system. But it does indirectly, by providing a clear insight to the observed weaknesses of the system and the associated risks for the organization. This enables management to make better informed decisions on allocating resources for improving the quality of the system.

To achieve these test goals, every test process contains activities for planning what is needed, specifying what should be tested, and executing those test cases. A generic test approach can be defined providing the basic structured approach for organizing a well-controlled test process.

As a consequence of the shorter development period, various simulators have been used to test functional behavior of the embedded control systems. Simulation can be used as an alternative to the actual target system for a significant portion of the testing effort, saving developers time and money, as well as increasing test coverage and providing better debugging facilities. This cover the technical issues involved in creating simulated test systems, as well as the business aspects and benefits. Simulating a system has always carried the advantage of increased insight and flexibility, at a cost in execution speed and timing fidelity the real machine. However, until recently, use of simulation technology for large scale embedded systems software development and testing has been fairly limited.

Hardware designers for processors, supporting chip sets, systems-on-chip, and servers have always made use of simulation in order to model hardware early. Simulation is used for performance evaluation, to test various ideas for implementation, and to validate that a system works as intended. Initial firmware bring-up and ports of operating system codes to new embedded and other computers is quite often performed using simulation tools, as the real hardware is typically not available early enough.

II. EMBEDDED SYSTEMS DESIGN

A. Embedded Systems

An embedded system is a special purpose computer that is used inside of a device. For example, a microwave oven contains an embedded system that accepts input from panel, controls the LCD display and turns on and off heating elements that cook the food.

***Manuscript received May 15, 2012*.**

***Miss. Ashwini M. Motghare**, Embedded System & Computing, G. H. Raisoni College of Engineering Autonomous Institute, e-mail: ashwini.motghare@gamil.com, Nagpur, India, Mobile No: 08928751154.*

***Prof. Swapnili P. Karmore,** Embedded System & Computing G. H. Raisoni College of Engineering Autonomous Institute, Nagpur, India, Mobile No: 09823437807, e-mail: swapravin@gmail.com.*

An embedded system is a part of a product with which an end user does not directly interact or control. Embedded systems possess the following Characteristics.

- Embedded systems utilize mechanics, electronics, and hardware and software technologies that are closely related to each other. Due to the high correlative dependence between software and hardware, the functionality and performance of embedded systems are implemented via cooperation between hardware and software.
- Embedded systems often have real-time requirements, i.e., correctness is partially a function of time. Most embedded systems are controlled in real-time, and usually adopted to fulfill those tasks with high real-time requirements.
- Many embedded systems must be robust, i.e., their behavior must always be controlled, even during system failure.

Embedded systems usually have limited hardware resources, memory size and running speed are restricted the embedded CPU and target environment.

B. *Structure of embedded systems*

An embedded system is a specialized computer system that is part of a larger system of machine. Typically, an embedded system is housed on a single microprocessor board with the program stored in ROM. Some examples of applications of embedded systems are consumer electronics, telecommunications, automobiles, and plant control. Although the application domains are different from each other, they have the common structure in functional configuration. Figure 1 shows a layered structure including hardware platform, hardware-dependent software, application software, and application programming interfaces. Application program interface are necessary for communication between the hardware-dependent software, and application layer of software. The hardware- dependent software is connected with the physical hardware and network. Real-time operating system and device drivers are closely coupled with hardware platform. From viewpoint of an application domain, performance and size the constraints that usually determine the hardware platform [3].

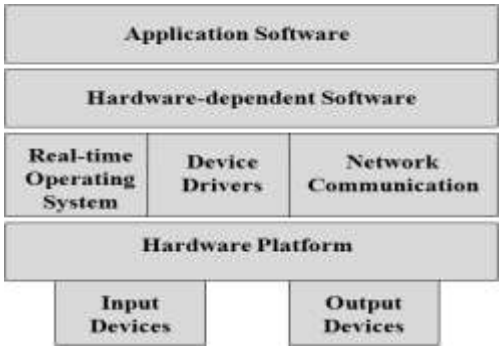

Figure1. Functional structure of embedded systems

III. TESTING ENVIRONMENT

In view of the complexity of embedded system, testing environments are constantly changing because of different characteristics of system under test (SUT) and different testing restrictions in the process of embedded system testing. That means we should developed different testing environment for different SUT or even for different testing requirement for the same SUT, which causes the specialization of testing environment. The specialization of testing environment leads some problems: Low reusability, Low reliability, Poor effect. So it is necessary to research how to ensure the testing environment has ability to follow the changing of the requirements. We attempt to solve this problem by developing testing environment framework [5] [6] with good reusability.

A. *Test Environment Architecture*

Embedded system testing environment (ESTE) is a kind of computer system that orients to embedded system testing. Using ESTE, tester can organize the inputs of SUT, complete configuration, drive test process and collect the output of SUT in order to realize the real-time testing for the embedded systems.

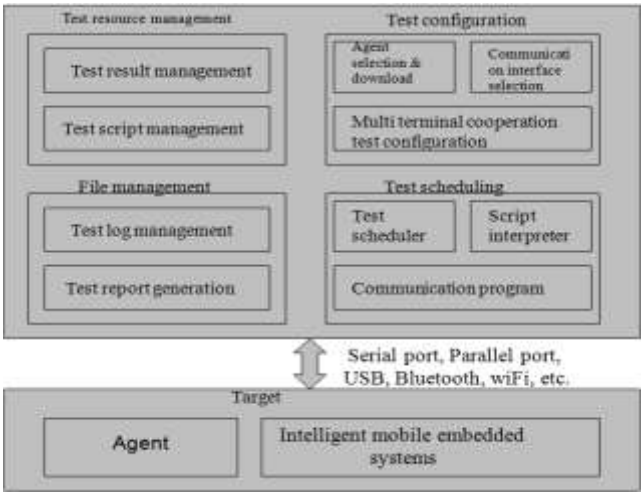

Figure2. Architecture of ESTE

Fig. 2 shows the architecture of ESTE which is composed of test scheduling, test resource management, test configuration, file management. As it is shown in figure 2, the test environment runs on the host where there are abundant resources.

The functions of testing environment host are

1. Test Scheduling - Completing the monitoring and scheduling during test process, and realizing the automatic testing with nobody by the use of script-driven method.
2. Test Resource Management – Managing test resources during test process, depositing and getting test results and recording the test script.
3. Testing Environment Configuration – allowing the tester to configure the testing environment such as selecting and downloading the agent, selecting the interfaces between host and the target machine, and configure the multi terminal cooperating testing.
4. File Management – Recording the testing log information and forming test report.

B. *Test process*

Erroneous embedded systems may cause serious accidents related to human safety. Therefore it is essential to detect any errors in the embedded systems through various test processes. The embedded systems are often subjected to change to meet customers need and functionalities, so that it is desirable to automate testing to improve the efficiency of process. Since there are huge number of test cases to execute in embedded systems, automation is required to maximize coverage and reliability.

Test automation consists of following three steps: Test case generation, test execution and result analysis. Test generator allow tester to generate test cases that can be executed automatically in testing environment. Test analyzer is developed in order to analyze test results. It compares test result from the simulation of the behavior model with the expected result. Target-suite provides automated testing environment for system under test. Target-suite consists of hardware equipment and control software. Here hardware equipment means system under test and control software consists of target-practice and target monitor. Target-practice which can map I/O signals of the functional model with a real ECU model. The other is Target-Monitor which analyzes and shows the execution result on PC monitor. Fig. 3 shows the picture of target-suite.

Figure3. Target-Suite

IV. EVALUATIONS

Growing Technology and their development increasingly influence Automation in industry. And, it plays an important role in the global economy and in our daily lives. Almost all the process monitoring systems installed as a part of plant or production process are basically Digital Control Systems DCS connected by digital networks. The purpose of automation has shifted from increasing productivity and reducing costs, to broader issues, such as increasing safety, quality and flexibility in the manufacturing process. With the help of DCS and wireless network, it is possible to make both startup activity and operational routines of a complicated process much easier and more efficient. DCS also offer process modeling and simulation.

We demonstrate the use of evolutionary testing for functional testing in an industrial setting by applying the developed solution to the testing of Process Flow Automation System through Simulating Environment.

Developed system consists of four servo motors whose functioning is to monitor and test serial production of process flow automation system. System also consists of input channel which is connected to proximity sensors for giving input signals to the system. Task of proximity sensors is to provide input signals in the form of 0's and 1's (low and high) to the input channels. Depending on the input signals given by input channels corresponding servo motors work on. By setting the sequence of servo motors developer can set numbers of functional task. Since there are four input channels so user can get 16 different combinations.

Figure4. Process Flow Automation System in Simulating Environment

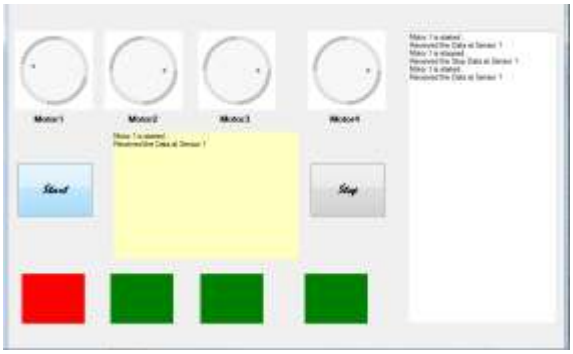

Figure5. Result of Process Flow Automation System in Simulating Environment

Fig. 4 describes process flow automation system in simulating environment. To test sequence of process flow automation in simulation Testing Environment, firstly it is connected to simulation testing environment through serial port as per the given APIs of the selected device. By Selecting corresponding port tester can make connection to device. After clicking on start button of testing environment user can test sequence of process flow automation in simulation Testing environment. And result of working corresponding servo motors will be display on simulating test environment as shown in Fig.

V. CONCLUSION

The complexity of the embedded systems is ever increasing while high system quality is demanded at the same time. With the continuously growing software and system complexity in electronic control unit and shorting release cycles, the need for efficient testing grows. To increase the

reliability, quality and consistency of embedded systems at development stage, it becomes much more necessary to construct embedded system testing environment.

REFERENCES

[1] Byeongdo Kang, Young-Jik Kwon, Roger Y. Lee, “A Design and Test Technique for Embedded Software”, IEEE, 2005.
[2] Pei Tian, Kai Wang, Kai Qiang, " Construction of Distributed Embedded Software Testing Environment", IEEE, 2009.
[3] Alberto Sangiovanni-Vincentelli and Grant Martin, “Platform-Based Design and Software Design Methodology for Embedded systems”, IEEE Design and Test Computers, November-December, 2001.
[4] Dae-Hyun Kum, Joonwoo Son, Seon-bang Lee, “Automated Testing for Automotive Embedded Systems”, SICE-ICASE, 2006.
[5] D. Parsons, A. Rashid, A. Speck, and A. Telea, “A Framewoek for Object Oriented Frameworks Design”, Technology of Object Oriented Languages And Systems, 1999.
[6] W. T Tsai, Y. Na, and R. Paul, “Adaptive Scenario-Based Object Oriented Frameworks for Testing Embedded Systems, “Computer Software and Applications Conference, 2002.
[7] Yongfeng YIN, Bin LIU, Guoliang ZHANG, “On Framework Oriented Embedded Software Testing Development Environment”, IEEE, 2009.
[8] Kandl, S.Kirner, R. Puschner, "Development of Framework Automated Systematic Testing of Safty-Critical Embedded Systems”, IEEE, Intelligent Solutions in Embedded Systems, 2006.
[9] D. L. kaleita and N. Hartmann, "Test Development Challenges for Evolving Automotive Electronic Technologies", SAE, 2004-21-0015, 2004.
[9] Bringmann, E.Kramer, “Model-Based Testing Automotive Systems”, IEEE, Software Testing, Verification, and Validation, 2008.
[10] Bringmann, E.Kramer, “Model-Based Testing Automotive Systems”, IEEE, Software Testing, Verification, and Validation, 2008.
[11] Lindlar, F. Windisch, “A Search-Based Approach to Functional Hardware-in-the-Loop Testing”, IEEE, 2010.
[12] Siegl, S.Caliebe, “Improving Model-Based Verification of Embedded System by Analyzing Component Dependences”, IEEE, Industrial Embedded Systems (SIES), 2011.
[13] N. H. Lee and S. D. Cha, "Generation Test Sequences from a Set of MSCs", The International Journal of Computer and Telecommunications Networking, Volume 42, Issue 3, Page 405 - 417, 2003.
[14] Y. G. Kim, H. S. Hong, D. H. Bae and S.D.Cha,"Test Cases Generation from UML State Diagrams", IEE proceedings, online no. 199990602, 1999.
[15] R. L. Probert, H. Ural and A.W.Williams, "Rapid Generation of Functional Tests using MSCs, SDL and TTCN", Computer Communications, Volume 24, Issues 3-4, Page 374-393, 2001.

Ashwini Motghare is student of M. E. in Embedded System & Computing at G. H. Raisoni College of Engineering Autonomous Institution, Nagpur Maharashtra.

Prof. Swapnili P. Karmore is Professor of Embedded System & Computing at G. H. Raisoni College of Engineering Autonomous Institution, Nagpur Maharashtra.