SOFTWARE-IN-THE-LOOP TESTING FOR PRODUCT TESTING SOFWARE IN LABEL TEST PROCESS

CHESTER LAURENCE BARCELON TAN

A dissertation submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Computer Science

School of Computing
Faculty of Engineering
Universiti Teknologi Malaysia

SEPTEMBER 2018

ACKNOWLEDGEMENT

I would first like to thank my thesis advisor Assoc. Prof. Dr. Dayang Horhayati Binti Abang Jawawi, for her endless support, advices, and encouragement to my research studies. She diligently have been giving direction to me during the course of my research every time I needed it.

I would also like to express my gratitude to Finisar Company for giving us this opportunity to further extend our studies and even sponsoring the financial expenses. Special thanks to my colleagues and friends for the support and ideas they have given throughout my research studies.

Finally, I must express my very profound gratitude to my parents and family for providing me with unfailing support and continuous encouragement throught my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

ABSTRACT

Manufacturing software are widely being used in automating production lines as it makes the process faster, ensure safety of the workers, and costs less in long term. As technology advances, the software used also updates as its job requirement for these types of software goes more complex. To ensure that the software does it job, software testing is performed. However, performing manual software testing consumes a lot of time and manpower, so it has become costly for business. By looking for solutions, some manufacturing companies has found ways to overcome this software testing issue by utilizing virtual testing methods. In this study, it is proposed to use Software in the Loop (SIL) method on testing ParTest. SIL is increasingly known in testing software embedded in microchips. This technique is used to allow the development of the software in parallel with the development of the hardware without waiting for the actual hardware to be ready. Through a technology survey, the SIL implement ParTest is evaluated according to its usefulness. By implementing this method, it shows that it can help produce better software and shorten the turnaround time for projects, which results to a more cost efficient method.

ABSTRAK

Perisian pembuatan digunakan secara meluas dalam mengautomasikan garisan produksi kerana ia membuat proses lebih cepat, menjamin keselamatan para pekerja, dan kos kurang dalam jangka panjang. Sebagai kemajuan teknologi, perisian yang digunakan juga kemas kini sebagai keperluan pekerjaan untuk jenis perisian ini menjadi lebih kompleks. Untuk memastikan perisian itu berfungsi, ujian perisian dilakukan. Walau bagaimanapun, melaksanakan ujian perisian manual menggunakan banyak masa dan tenaga kerja, jadi ia menjadi mahal untuk perniagaan. Dengan mencari penyelesaian, sesetengah syarikat pembuatan telah menemui cara untuk mengatasi masalah pengujian perisian ini dengan menggunakan kaedah ujian maya. Dalam kajian ini, dicadangkan untuk menggunakan perisian dalam Loop (SIL) pada ujian ParTest. SIL semakin dikenali dalam perisian ujian yang tertanam dalam mikrocip. Teknik ini digunakan untuk membolehkan pembangunan perisian selari dengan perkembangan perkakasan tanpa menunggu perkakasan sebenar siap. Melalui tinjauan teknologi, SIL melaksanakan ParTest dinilai berdasarkan kegunaannya. Dengan melaksanakan kaedah ini, ia menunjukkan bahawa ia dapat membantu menghasilkan perisian yang lebih baik dan memendekkan masa pemulihan untuk projek, yang menghasilkan kaedah yang lebih efisien.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

BIU	-	Behavioral Intention to Use
DUT	-	Device under Test
HIL	-	Hardware-in-the-Loop
JSON	-	JavaScript Object Notation
MIL	-	Model-in-the-Loop
PEOU	-	Perceived Ease of Use
PU	-	Perceived of Usefulness
SIL	-	Software-in-the-Loop
TAM	-	Technology Acceptance Model
VM	-	Virtual Module

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Overview

In the current age of globalization, being able to rapidly respond to customers' requests is fundamental to win against competitors and gain new customers. It is about speed and cost, which means that being able to respond fast and contain costs (Sangregorio, 2015), but this is sometimes at the expense of quality. In a manufacturing company, the production relies heavily to automated software to test their products. A downtime to debug the faulty software would take adequate amount of time, thus could lead to missed production shipments. At worst, a buggy software can be released in production, which will lead the company to ship bad units to customer, and the customers will validate the units then catch the bad units, this will cause the company a lot of money and bad reputation among its customers.

Software testing is an important process in the software development life cycle, because it improves the quality of the software in terms of security, reliability, performance, and compatibility. It is commonly believed that the earlier a software bug is found, the cheaper it is to fix it.

Having an establish software testing system for the production software will definitely help prevent buggy software to be released in the production line. Though the idea of producing bug-free software is not possible, software testing can assess the level of reliability of the software, accordingly will give the confidence on the software.

The amount of time required for software development determines how rapidly a new products can be introduced. The developed software shall be tested on the workstation to be validated, but as more verification tests are done, more down time is induced to the workstation, thus hindering production output. In order to accommodate the rapid product development, the virtual testing concept is to be used, which allow testing of the software under a simulated environment.

1.2 Problem Background

Depending on the products produced, manufacturing companies nowadays utilizes the current generation's computational power available, which are far more dominant than few decades back to assemble, calibrate, and validate their products. Car manufacturers uses robotics to assemble their car products. Semiconductor companies uses test boards and apparatus to calibrate and verify their products. This additional computational power has enabled software to perform complex application (Nouman et al, 2016).

Software applications are used to automate manufacturing processes. It has been the integral part of the modern industrial manufacturing plant. There are a number of reasons why to automate manufacturing processes. It improves worker's safety as machines will substitute human workers from a dangerous environment. It reduces labor cost, as machine can replace human at certain processes. It will also increase production output as there will be no idle time for the machine. It will also improve product quality as it will eradicate human error, which is cause by manual routines that makes workers bored.

As workers' jobs are passed on to the machines, the responsibility to perform the job accurately is passed on to the software developers. Software developers are human too and can also make mistakes, so the automation software they do are not perfect as well. There will be bugs in the software that may cause quality issue or cause a production line to be down. That's why software applications must be validated and verified at a certain requirement level prior to be release for production usage.

There are challenges in testing automated software application in the production line. The table below shows some of the problems and reasons encountered in the production line.

Problems	**Reasons**
Limited Time	Production machine is for production output
Limited Resources	Not all equipment and devices will have extra for debugging and development purposes Few manpower for numerous diverse products
Replication of bugs and errors	Some bugs are hard to replicate in real machines as it needs to perform certain actions at certain sequence
Potential damage to products	Repeated testing on the same product will deteriorate the product and may change its characteristics already

Virtual prototyping can be an alternative or complement to the real manual testing to resolve such problems (Kim et al, 2017). There are already existing and proposed methods in performing virtual prototyping. Some even already has the available tools in performing it, especially for car manufacturers and embedded systems. MATLAB and National Instruments are few examples of the virtualization tool that already contains software replications of cars and embedded systems, so the only focus of the user is to perform tests on their software.

1.3 Problem Statement

Finisar manufactures a variety of optical product and is in a continuous research to provide new ones and growing requirements of the customers. Software automation enables each product to be manufactured efficiently and with quality, however, as new variety products are introduced and growing requirements, the software will also undergo in a continuous development as well. Along with supporting the new requests, the software should still be able to support legacy product requirements, and this is where the company falls short.

The automated manufacturing software used in Finisar is called ParTest, which is used to calibrate and test the optical transceivers. The software runs in a host PC and communicates with the transceiver modules through a device-under-test (DUT) board, and then controls several other instruments to measure the transceiver modules characteristics. The typical setup can be seen in Figure 1.1. When the software is released in the production line, operators will scan the barcode of the transceiver module and plug the transceiver module to the DUT board to start testing. The scanning of the barcode will let ParTest identify what type of process step need to be performed then waits for the transceiver modules to be inserted.

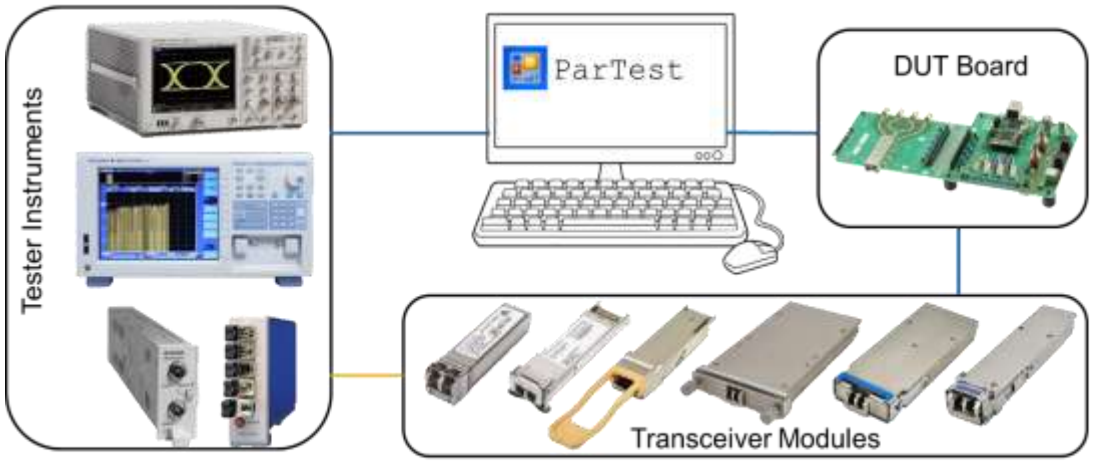

Figure 1.1 Finisar’s typical tester setup

New products will also use the same software to perform product testing. Most of the time, these new products share the same characteristics of the old products, so they can reuse the same set of codes, but some will have new requirement as well, thus will need new iteration of software release. New release of ParTest is either due to new set of codes or code modification. With code modification, it will need to be tested for the new requirement as well as the old requirements as it can change the old behavior, thus software testing is necessary.

Manual testing of ParTest consumes a lot of time to perform, as it needs to wait for a testing setup to be available, and with a variety of products for ParTest to be verified with, it will be an unending task. Replication of bugs and error is also time consuming as the user will need to perform certain tasks at specific sequence to repeat error, and sometimes it still does not appear. A developer will perform the required code changes to the software, borrows a production tester, test the code changes, debug errors and issues encountered, then finally release back the production tester along with the new software. This is the typical flow for the developer to validate the software for the product using the new software features, but not for the legacy products.

Unlike car manufacturers or embedded system, there is no ready testing system software for optical transceivers like MATLAB, which has the tools and virtual representation of the components needed. However, similar approach can be used to test ParTest, in which devices and equipment used by ParTest will be virtualized, so it can be tested in a virtual or simulated environment.

Regression tests need to be perform for the legacy product to have the confidence to use the newly released software. With virtualization, this can be done easily in a non-production tester, avoiding the difficulty of long duration of manual testing of thousands of products. A developer creating a new code for a new product can test ParTest in a real set of hardware, but may test legacy products in the simulated environment for backward compatibility.

The virtual prototyping method is usually done in embedded system and also usually on a fresh software. In this study, we are trying to use the same method, but under a different circumstances. The software to be tested, ParTest, is not an embedded system program nor is a fresh software. Code modification of ParTest for this study is allowed and need to done in order to perform software testing in a virtual environment.

Therefore, the research question is "*Does the virtual environment approach suitable for manufacturing software under the environment that is similar to Finisar?*"

1.4 Research Aim and Objectives

The aim of this research is to propose a software testing system for manufacturing software, which is suitable for Finisar environment, in order to test the software without any usage of physical hardware or equipment.

This research consists set of objectives to be achieved that lead to the research process:

- To identify the software testing method to be used for testing Finisar's manufacturing software, ParTest, which will overcome the current issues and limitations.
- To propose and apply a testing system using the selected software testing method and implements the system to ParTest.
- To evaluate and validate the proposed software testing system's usability with ParTest as the subject of the proposed system.

1.5 Scope

This research is meant for improving the quality of the manufacturing software under the Finisar environment. In Figure 1.2, it shows the different processes a transceiver module goes through, and each process requires the use of ParTest. The scope of the study will be the following:

- This study only focuses on the software testing method for a manufacturing software, similar to Finisar, leaving out the rest of areas, such as test case generation.
- This study only focuses on a single process of transceiver module testing, the LABEL process, to limit the number of involved instruments.
- This study only focuses on a functional testing (Black-Box), validating that the software output is the expected output.

1.6 Significance of Study

This study is intended to help create a software testing system specifically for Finisar manufacturing site and other industries with the similar environment of limited resources. Development and testing of software application in manufacturing site will have reduce downtime in the production line. It helps start the development of the software even though the hardware is not yet complete as long as the hardware capabilities are defined.

1.7 Dissertation Organization

This research paper is made up of seven chapters. In the Chapter 1, it discusses the research introduction, problem background, problem statement, research aims, research goals, scope, and significance of the study. In the Chapter 2, it presents the overview of software testing and further discussion on chosen methods of software-in-the-loop, virtual environment, and virtual prototyping. In the Chapter 3, the research methodology is detailed. In the Chapter 4, the proposed testing system and its framework is discussed and detailed. In the Chapter 5, a couple of case study is performed and analyzed. In the Chapter 6, the evaluation of the testing system is discussed. In the Chapter 7, the conclusion and results of this research paper is discussed.

REFERENCES

Abdullah, M. Z. T. & Mahrin, M. N. B., (2011), Component Based Testing for Vision System Development Platform (VSDP), MSc Thesis, Universiti Teknologi, Malaysia.

Alvares, M., Marwala, T. & de Lima Neto, F. B., (2013), Applications of Computational Intelligence for Static Software Checking Against Memory Corruption Vulnerabilities, *Computational Intelligence in Cyber Security,* 16-19 April

As'sahra, N. F., (2015), Test Case Prioritization Technique Using Sequence Diagram and Labeled Transition Systems in Regression Testing, MSc Thesis, Universiti Teknologi, Malaysia

Bennett, T. & Wennberg, P., (2003), Maintaining Verification Test Consistency Between Executable Specifications and Embedded Software in a Virtual System Integration Laboratory Environment, 28th Annual NASA Goddard Software Engineering Workshop, page 1-8

Bhattacharya, S., Kanjilal, A. & Sengupta, S., (2010), Tools and Techniques for Model Based Testing, *Handbook of Research on Software Engineering and Productivity Technologies: Implications of Globalization,* page 226-249

Casolino, G. M., Tir, M. A., Andreoli, A., Albanesi, M., & Marignetti, F., (2016), Software-in-the-loop Simulation of a Test System for Automotive Electric Drives, IEEE, page 1882-1887

Demers, S., Gopalakrishnan, P. & Kant, L., (2007), A Generic Solution to Software-in-the-Loop, Applied Research, Telcordia Technologies

Ekssan, S. N. B. M., (2017), Enhanced Educational Robotics Feature Model With Pedagogical Element in Software Product Line, MSc Thesis, Universiti Teknologi, Malaysia

Elhagari, U. T. F., (2015), A Model-Based Testing Framework for Trusted Platform Module, Phd Thesis, Universiti Teknologi, Malaysia

Jones B., (2001), The Automation of Software Validation using Evolutionary Computation. *Telecommunications Optimization: Heuristic and Adaptive Techniques, John Wiley & Sons*, page 265-283

Kapur, P., Singh, G., Sachdeva, N. & Tickoo, A., (2014), Measuring Software Testing Efficiency Using Two-Way Assessment Technique, *Infocom Technologies and Optimization, 3rd International Conference,* 8-10 October, page 1-6

Kim, B. G., Kashiba, Y., Dai, S. Y. & Shiraichi, S., (2016), Testing Autonomous Vehicle Software in the Virtual Prototyping Environment, IEEE Embedded Systems Letter, March 2017, page 5-8

Liu, H., Jin, M. & Liu, C., (2010), Construction of the Simulating Environment for Testing Distributed Embedded Software, *Computer Science and Education in 5th International Conference*, 24-27 August, page 97-101

Luo, L., (2001), *Software Testing Techniques: Technology Maturation and Research Strategy,* page 1-20

Mutter, F., Gareis, S., & Schatz, B., (2011), Model-Driver In-the-Loop Validation: Simulation-Based Testing of UAV Software Using Virtual Environments, *2011 18th IEEE Internation Conference and Workshops on Engineering of Computer-Based Systems,* page 269-275

Nouman, M., Pervez, U. & Hasan, O., (2016), Software Testing: A Survey and Tutorial on White and Black-Box Testing of C/C++ Programs, *Region 10 Symposium (TENSYMP)*, 9-11 May, page 225-230

Palmieri, M., (2012), System Testing in a Simulated Environment, MSc Thesis, Mälardalen University, Sweden

Sangregorio, P., Cologni A. L. & Previdi, F., (2015), Modular Automatic Generation of Automation Software for Manufacturing Machines, Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI), 16-18 September

SFF Committee, (2001), *INF-8074i Specification for SFP (Small Formfactor Pluggable) Transceiver Rev 1.0*, 12 May.

Zulkefli, H.S., (2004), “Engineering Maintenance System (EMESYS) Maintenance Module Using .Net Platform”, Master Degree Award, August, Centre for Advanced Software Engineering, Universiti Teknologi Malaysia, Pages 1-106