# EFFECTS OF SPL DOMAIN ENGINEERING ON TESTING COST AND MAINTAINABILITY

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

ZİYA ŞENBAYRAK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

JANUARY 2013

Approval of the thesis:

**EFFECTS OF SPL DOMAIN ENGINEERING ON TESTING COST AND MAINTAINABILITY**

submitted by **ZİYA ŞENBAYRAK** in partial fulfillment of the requirement for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen ____________________
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. İsmet Erkmen ____________________
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. Semih Bilgen ____________________
Supervisor, **Electrical and Electronics Engineering Dept., METU**

**Examining Committee Members:**

Assoc. Prof. Dr. Cüneyt F. Bazlamaçcı ____________________
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Semih Bilgen ____________________
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Ece Güran Schmidt ____________________
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. Ali H. Doğru ____________________
Computer Engineering Department, METU

S.Eren BALCI, M.Sc. ____________________
Test Systems and Processes Design Dept., ASELSAN

**Date:** ______________

**I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.**

Name, Last name: ZİYA ŞENBAYRAK
Signature :

# ABSTRACT

EFFECTS OF SPL DOMAIN ENGINEERING
ON TESTING COST AND MAINTAINABILITY

Şenbayrak, Ziya
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Semih Bilgen

January 2013, 80 pages

A software product line (*SPL*) consists of a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way. Together with testing of final deliverable products developed within the SPL, called Integration Testing, particularly important in this context is the way individual hardware as well as software components in an SPL are tested and certified for usage within the SPL. This study investigates specific approaches and techniques proposed in the literature for unit testing in the SPL context. Problems inherent to this issue were studied and possible solutions aiming towards systematic and effective testing of hardware as well as software units in SPLs have been proposed. The specific problems of SPL testing in ASELSAN were investigated in the light of these possible solutions and their applicability as well as their benefits were quantitatively assessed.

Keywords: Software Product Lines, Board Test, Domain Engineering

# ÖZ

YAZILIM ÜRÜN HATLARI ALAN MÜHENDİSLİĞİNİN
TEST MALİYET VE İDAMESİ ÜZERİNDEKİ ETKİLERİ

Şenbayrak, Ziya
Yüksek Lisans, Elektrik Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Semih Bilgen

Ocak 2013,80 sayfa

Yazılım Ürün Hattı (SPL), belirli bir pazar kesiminin veya hedefinin özel ihtiyaçlarını karşılayan, ortak bir kontrollü özellik kümesini paylaşan, tanımlı bir yolla ortak bir çekirdek yapı üzerine kurulmuş yazılım yoğunluklu sistemler dizisidir. SPL içinde üretilen son teslimat ürünlerinin test edilmesi sırasında, Tutarlılık Testi ile birlikte, önemli olan SPL içindeki özgün yazılım ve donanım bileşenlerinin test edilmesi ve SPL içinde kullanılabilmeleri için onaylanmalarıdır. Bu çalışmada, SPL deki yazılım ve donanım birimlerinin testleri için literatürde önerilen yaklaşım ve teknikler incelenmiştir. Bu konuya özgü problemler üzerinde çalışılıp, sistematik ve etkili birim testlerini hedefleyen olası çözümler önerilmiştir. ASELSAN özelinde birim testlerinde gözlemlenen sorunlar bağlamında bu çözümlerin uygulanabilirlikleri incelenerek, yararları sayısal biçimde değerlendirilmiştir.

Anahtar Kelimeler: Yazılım Ürün Hattı, Kart Test, Alan Mühendisliği

***To those I have sacrificed…***

## ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my supervisor Prof Dr. Semih BİLGEN for his guidance, advice, criticism, encouragement and insight throughout the completion of the thesis.

I am indebted to all of my friends and colleagues for their support and encouragements. I am also grateful to ASELSAN Inc. for the facilities that made my work easier.

Finally, I am grateful to my family for their continuous support and encouragements.

# TABLE OF CONTENTS

# LIST OF FIGURES

FIGURES

# LIST OF TABLES

TABLES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| AE | Application Engineering |
| ATE | Automatic Test Equipment |
| ATEMS | Automatic Test Equipment Management Software |
| BS | Boundary Scan |
| BSTE | Board Specific Test Equipment |
| BSTS | Board Specific Test Software |
| DE | Domain Engineering |
| HW | Hardware |
| ICT | In Circuit Test |
| MBSTE | Manual Board Specific Test Equipment |
| PL | Product line |
| PT | Project T |
| PZ | Project Z |
| PCB | Printed Circuit Board |
| SPL | Software product line |
| SW | Software |
| UUT | Unit Under Test |

# CHAPTER 1

# INTRODUCTION

In software product lines (*SPL*), all efforts are basically grouped in two different but related areas which are domain engineering and application engineering. In domain engineering, focus is basically on the core assets of the product line, which will be used for all products of this family. On the other hand, in application engineering a specific product takes all attention.

While testing a *SPL*, test plan and test cases should also obey this organization which means there should be some test cases which only focus on the core assets of the SPL and applied to them. In addition to these test cases, there should be some product specific test cases which try to discover any problem in a specific product. [2]

In general, in order to test hardware, a test setup including both hardware and software part is designed. Making a specific setup for every product increases testing costs dramatically and once the production of a specific product is over; easy reuse of that hardware is nearly impossible without a serious effort to adapt it to the new product. In order to overcome this problem and build economically efficient test setups, modular testing equipment is preferred. Such modular testing equipment is designed with reuse in mind, and they allow testing more than one kind of product with small adjustments. They increase the initial cost of the system whereas significant reduction of test cost per product become possible, when reuse of the system hardware for different products is taken into account. [7]

The present study aims to increase efficiency of hardware testing in an economic manner, by applying *SPL* methods to the hardware testing area. In other words, while attempting to implement *SPL* techniques in hardware engineering in general, illustrating the effectiveness of borrowing relevant solutions from the realm of *SPL* to hardware product lines in the context of system testing constitutes the main aim of this study.

Within the context of this study, in the domain engineering phase, possible hardware requirements of the products which are produced by the facility were considered and commonalities were detected. By using these commonalities an Automatic Test Equipment (*ATE*) for this product family was designed. This *ATE* constitutes a common interface to connect a product specific hardware adaptor.

In the application engineering phase, product specific requirements were considered and the required adaptor which is used to connect hardware to the family specific *ATE*, was designed. This method decreased the cost of test setups dramatically since all setups share same resources on the family *ATE*. Some scheduling problems can be encountered as a result of one *ATE* for the whole family but that can be solved by duplicating the *ATE*.

ASELSAN produces electronic equipment for air, marine and land combat/defense vehicles. As an expected result of being a defense company ASELSAN does not mass produce. And also it has a very large number of R&D projects some of which are not going to be produced in future.

Under this condition product specific investment on testing equipment would become infeasible within a few years. When costs of testing hardware and the burden of importing this equipment from other countries are considered, the need for reusable hardware testing equipment becomes clear especially for the national defense industry.

After careful domain engineering, a general purpose *ATE* were designed. And this automation and standardization of testing decreased the documentation time because technicians apply only the standard procedure to start the test of the electronic board, rest of the product specific actions are held by the *ATE* automatically by using product specific test software which was developed by engineers.

*ATE* was not only bring reusability to the testing hardware but also make hardware testing more efficient.Efficiency of the automatic test setup can be measured in terms of engineering development time, technician's application time, testing time, total investment cost, initial setup time, mechanical design and implementation costs. [8]

Implementation of proposed ATE system was held by ASELSAN test design team where the author is one of the members. Above mentioned metrics are recorded for both automatic and manual test systems by the author specifically for this study. Recorded metrics were analyzed and compared in the scope of this study to clarify the strong and weak points of the proposed solution.

The rest of this document is structured as follows: in chapter 2, a review of the literature on *SPL* and on hardware testing is presented. Chapter 3 presents the hardware testing problem in ASELSAN, and the approach for applying techniques borrowed from SPL on these problems to decrease costs while increasing efficiency and testing quality. Chapter 4 explains the implementation of the proposed solution in detail and gives information about the final implemented system. Chapter 5 discusses the concrete gains derived from applying the proposed solution, in terms of the measurements obtained from earlier projects and those in which the proposed approach has been applied. The last chapter concludes the study by outlining the achievements, overviewing the shortcomings, limitations and suggesting directions for future work.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 Software Product Line

Software has become the key element of everyday life in the last two decades. It has been used nearly everywhere from supermarket checkouts to the pay system of public transportation. IEEE Standard definition for software [1] is;

*Software: Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.*

*Computer program: A combination of computer instructions and data definitions that en-able computer hardware to perform computational or control functions.*

Software technology has difficulties which make it impossible to increase not only the productivity of the development process but also performance of the final software in the order of magnitudes.

These difficulties are grouped in two different categories by Brooks. First one is the *Essential difficulties* and it contains the problems which are the consequences of the software's nature. [6]

Essential difficulties are [6];

Complexity: Software systems have enormous number of states which makes it impossible to define, review, test and visualize each of the states individually. Every state describes a unique input combination for the software system that is why using simplified models to solve the problems is not applicable for this case.

Conformity: Software systems are used in nearly every industry with every possible interface and the common expectations from these systems is to successfully conform to the required interfaces.

Changeability: After software systems are finalized at a point in time modification requests start not only to use the software on a different domain that it is not designed for, but also to overcome the replacement problems of wore out/obsolete hardware systems that software system works together. Both of these cases mean major extensions/modifications on the functions. Therefore, software systems should embrace the change in every possible environmental variable to maximize the ease of maintainability of the system.

Invisibility: Hierarchical diagrams are inadequate to create a complete visualization of the software systems' interior architecture. This hinders not only developers' understanding about the concept of the system but also communication among the designers and customers.

Second category *is Accidental Difficulties* which are the difficulties caused by programmers' mistake or development tools inadequacy. Problems of this category can be solved by using proper methods. Examples of problems solved in the last five decades are;

- Mistakenly written machine level codes related problems were solved by using high level languages which brought simplicity to the coding activities.

- Problems related to having limited time for debugging period were solved by using time shared computers which increase the productivity of the programmers.

- Using Unified development environments attacks to the problem caused by using numbers of small individual programs together, and solves it by providing libraries.

Comprehensive software design and elaborate implementation of this design can prevent the software system from *Accidental Difficulties.* However, there is no method to overcome the *Essential Difficulties*.

Despite having no solution to this *Essential Difficulties*, cost of these problems to the products can be minimized by maximizing the reuse of the design assets and implemented code wherever possible.

Wegner [9] states that software industry has much more *Essential Difficulties* than any other known industries. This is why software industry should find a way to overcome the cost of these inevitable problems and proposed method for the cost minimization is the enhancing reuse whenever possible.

<u>What is Reuse?</u>

In "*Ad hoc reuse*" approach, anything written by developers (module, objects, etc. ) are collected under a company reuse library and when a new project is started developers are forced to use the pieces in that library. While developers first built that pieces "reuse" was not in their mind as a result of that using these code pieces and modifying them often take more time than building new ones from scratch . [19]

In contrast to the "*Ad hoc reuse*" attitude, "*systematic reuse*" methods are high order, planned and managed reuse actions not only on the technical sides but also on the nontechnical (documentation, organization, etc.) sides. From the beginning the costly parts of the projects (domain models, test cases, etc.) are planned and designed to reuse in other systems. [20]

Another traditional reuse attitude is "*clone and own*" method. At the beginning of the new project, developers barrow whatever they can from previous similar projects and modify these according to the new requirements. It definitely gives some economical advantages to the company but at the end of the day, the company will have two entirely different systems which means two different maintenance procedure and effort will be held by the company. [20]

Three different reuse methodologies are described and analyzed above. Un-planned nature of '*ad hoc reuse'* and difficulties faced in the modification steps of "*clone and own*" method clearly put "*systematic reuse*" one step forward.

*"Systematic reuse"* methodologies can be effectively applied on product families rather than unrelated individual products. Parnas defines product families as the set of software programs that shares common properties as much as possible and differentiate from each other by some individual properties. According to Parnas, design and development actions of the software product families should be planned to complete properties that are shared by all the family first .After completion of the common structure , design and development actions can be dedicated to the special properties of the individual products.[10]

Defining a software family also means defining a domain that all possibilities will be covered by designed products. And a software component which is developed for a defined domain can be systematically reused by any other software belongs to that domain.

Defining a product family and a domain then building a structure to develop this family is called as *product line (PL)* and specifically *software product line* (*SPL*) in software engineering literature.

Designing and implementing a *SPL* cost much higher than developing a single product. In industries which produce physical goods this establishing cost of the *PL* is spread through the individual products and mass production of the designed goods solve this economical problem. But this is not applicable in software industry.

There are mainly two different approaches used in the industries for the economical projections and plans, namely *Economy of Scale* and *Economy of Scope*. [5]

*Economy of Scale* approach is based on the mass production idea which means producing and selling enormous numbers of the same design. By increasing the number of sold product and decreasing the production cost, high development and design costs can be added to the price of the individual products without affecting the price strategy on the market.

*Economy of Scope* approach focuses on the cases where there are multiple product designs but they all have some distinct properties. In this economic model high design and development cost is shared by these different product designs.

In the software industry production costs can be neglected compared to the design and development costs of the systems. Therefore in the industry *Economy of Scope* approach is generally accepted. Moreover, there are some software application areas where producers can sell more than one copy of their software like word processors, in such cases *Economy of Scale* can also be taken into consideration.

Generally, companies order and use the software to gain some competitive advantages which means that it is nearly impossible for the software developer company to use same product for another customer. However if the developer company use family of the product approach and develop its products to maximize the reuse, it can benefit from splitting the development costs more than one project budget as it is proposed in Economy *of Scope* context.

Cusumano's [4] studies about software industry in Japan show that nearly 90 percent of the software that is produced in a company in a year is similar to the computer programs which were produced the previous year at that company.

Both Cusumano's study and *Economy of Scope* approach which was mentioned above, direct the software development industry to be more reuse oriented and systematic.

*PL* approach, which is supported by Parnas' theory mentioned above, is the key architecture for systematic reuse and it will be discussed in detail later.

*SPL* is defined [3] as "set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way." By using base of common assets, producing a new software system just becomes integration of modules rather than creation (programming).

Non-technical parts of the projects have bigger problems than the technical parts in terms of reuse. Only developing some piece of reusable code library will not solve the reuse problem without an organizational support and defined methodical process for design and construction phases.

By getting similar final products in terms of core assets not by chance but by careful planning gives the company strategic strength and economic advances. These are;

- Improved time to market as a result of increasing productivity.
- Increased personnel mobility as a result of commonality between the products of the company.
- Skillful developer teams as a result of having more time to learn and adopt new technologies to the company's *know-how*.

Beyond these benefits, usage of the *SPL* methodologies increases the complexities of the project's phases and required effort for completion.

- <u>Specifying requirements</u> not for one system but for a *PL* requires more time and complex analysis.
- Considering more than one system and their variation points at the <u>architectural design</u> phase requires more talented system architects and complex analysis.
- Inserting built-in variation points to the <u>software components</u> to be used in different members of the family brings extra effort on algorithm development and debugging phases.
- Creating <u>test cases and scripts</u> for more than one product by taking the variation points into consideration requires complicated analysis.

These extra efforts and costs become meaningful via systematic reuse when they are shared by the product family members. This is the key point for successful *SPL* implementation.

*"Bottom-up"* and "*top to bottom"* are two distinct implementation strategies of the SPL. At the "*top to bottom"* method, final products of the PL are designed and developed. After finalizing the products, core assets are extracted from these final products. "T*op to bottom"* SPL implementation strategy does not have any phase for taking the future product family members into consideration at the designs which clearly place this strategy to the neighborhood of the ad hoc reuse methodologies.[3]

*"Bottom-up"* methodology dictates to develop core assets from scratch and after completion of this development phase, new products are built by using these assets. This methodology considers all possible new products of the company at the core asset development's design phase. [3]

In SPL projects there are two kinds of engineering activities; actions related to core asset planning, designing and developing is called as *Domain Engineering*. And the actions related to an individual product development are called as *Application Engineering*.

### 2.1.1 Domain Engineering

Domain engineer has *'Product Constraints', 'Production Constraints', 'Production Strategy'* and *'Pre Existing Assets Info'* at the beginning of the domain analysis step, which contains information about commonalities of the product family, expected future technological developments, quality requirements, required standards, production agenda and desired assembling strategy of line members.[5]

'*Software Product Line Scope', 'Core Assets'* and '*Production Plan'* are the three outcomes of the domain engineering phase [3]:

*SPL scope* defines the specifications of the products that can be produced by this *PL*; in other words it defines the capabilities of the *PL.*

*PL* scoping is the key phase of domain engineering activities for the success of the *SPL* methodology. Having an unnecessarily large *PL* scope, i.e. increasing variant and decision points in core assets to enlarge family variety, results in unmaintainable and error prone components. On the other side, if a narrow product line scope is chosen, developed core assets will not be generic enough to support future growth of the product line. Furthermore, after a period of time core assets will become useless for newly developed products which implicitly means that turning back the investments of the company to the core assets will not be possible.

In order to have a right *PL* scope, domain experts must do the scoping by using knowledge about the business, market competition, available technologies, business goals of the firm, and the capabilities of developer teams.

Scoping is also an iterative action which means that after constructing product line accordingly decided product line scope; scope will not remain static for life time of the *PL*. It will evolve continuously in order to keep itself updated as a result of changing market conditions and technological developments.

*Core Assets* are the fundamental ingredients of the production activity on the *PL*, which are consist of every reusable software components, documents, designs.

Carefully analyzed and designed component architectures result in a common architecture at all *PL* members and this ensures the economic advantages by not only reducing design, test and maintenance costs of the new products, but also increasing safety and reliability of the *PL* members.

*SPL* architecture, i.e. skeleton of every *PL* member, is emerged by these engineered components therefore the designer should have experience on designing complex system architectures and has profound knowledge about not only the scope of the *SPL* but also available technologies on the market. Insufficient or careless design of the *PL* architecture will definitely results in the collapse of the whole *PL* system both in technical and economical meanings.

*Production Plan* defines how to use *core assets* to build specific products on product line and which methods should be used to set variation points of the software components to get specific product.

### 2.1.2 Application Engineering

Relationship between the general product line and specific products is shown in Figure 2.1.

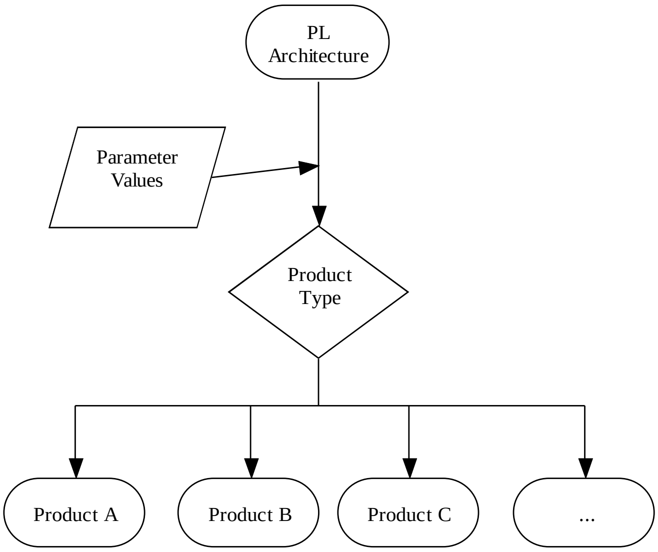



***Figure 2.1-Creating product from product line***

***<u>( All material presented in photographs, figures, tables, etc. are original, unless explicitly stated and referenced otherwise)</u>***

As described in the beginning of this section application engineering covers all the activities in designing and producing specific member of the product family.

Application Engineering activities basically use product requirements and outputs of the Domain engineering phase. Product requirements, which are the key inputs of application engineering activities, are generally consist of technical specifications of the specific family member.

In Application Engineering phase main purpose is to analyze above mentioned inputs, decide which core assets will be used and what additional components will be developed for these specific products. Also, pre-defined variation /decision points of the core assets are decided and set inside the Application Engineering activity scope by considering product requirements.

Main focus of the all Application Engineering phase is to maximize core asset usage as much as requirements and standards permit. [5]

As a consequence of the iterative nature of the *SPL* methodology, if necessary some actions can be taken to update some core assets in this phase and new version of update core assets can be used when they are finalized.

## 2.2 Board Testing

General flow of the electronic hardware tests are showed as in Figure 2.2.

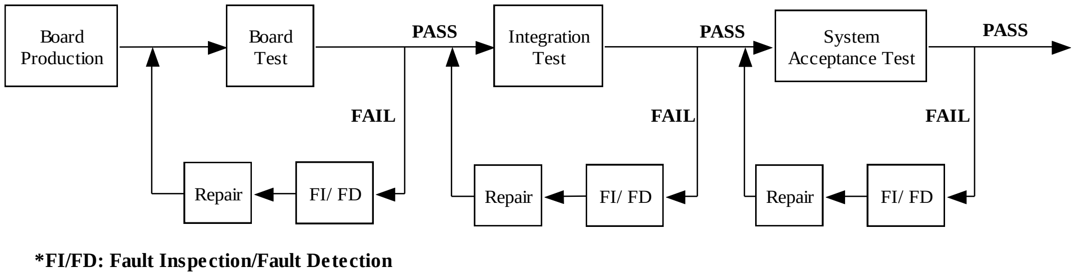



***Figure 2.2- Electronic Hardware Test Flow***

The present study will just focus on the board test level of these testing activities. Integration testing and System Acceptance tests are out of the scope of this study.

In [14], the main purpose of board testing is summarized as;

- To confirm that each component performs its required function;
- To confirm that the components are interconnected in the correct manner; and
- To confirm that the components in the product interact correctly and that the product performs its intended function.

Board testing was started as a manual process, where experienced technicians use some probes, multimeters and detailed testing procedure to verify the given board.

In the manual probing of circuit card process, technician uses reference documents to find the probing location when the probing steps of the testing procedure order to do. After placing probe in to the place found from reference document, technician will hold probe until required time elapsed to apply/measure desired signal. Testing logic and strategy of this manual process ignores the possibility that technician can touch on the unintended location, or does not apply sufficient pressure to have proper contact or leaves probing before measurements are done.

Trust in the operator is the essential property of every manual process which clearly makes manual testing methodology an error prone system which can easily give erroneous decisions about components or boards. In best scenario this wrong decision will cause the re-testing of the board and wasting operator time and decreasing test equipment throughput.

Also it is definite that using reference documents to find each probing locations, placing probes, and taking measurements one by one enormously increase testing time of the board.

The demand for electronic boards increased drastically, after chips were getting cheaper and capacity of the boards increased exponentially. As a result of this, manufacturing industry started to complain and blame manual testing as being slow, error prone, and being over cost.

### 2.2.1 In Circuit Testing

*In circuit test* (*ICT*) methodology extends the coverage of the manual testing and also decrease design and application time of the tests. Moreover, it is a white box testing approach, which means not only the functions of the *unit under test* (*UUT*) but also internal structures of the unit are tested in this methodology. [13]

The main aim of the manufacturing industry is to obtain good boards in an economical manner. According to the analysis made on the manufacturing data, high volume of the problem is caused by poor workmanship which results in improperly oriented or bad soldered components. [12]

In *ICT* methodology, designer uses board topology and component part list to decide the location of probes which will be used to apply signals and measure the responses. Probes are placed on a test fixture which is called as bed of nails.

Example of a conducting probe is showed in Figure 2.3. A standard bed of nail includes tens of these probes on it to take required measurements as seen in Figure 2.4.

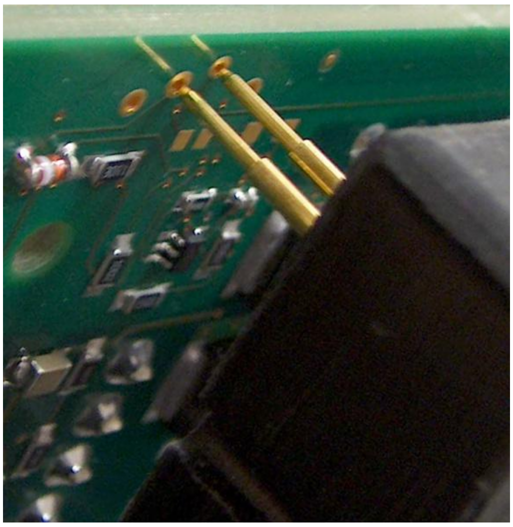


***Figure 2.3- Example of Conducting Probes***

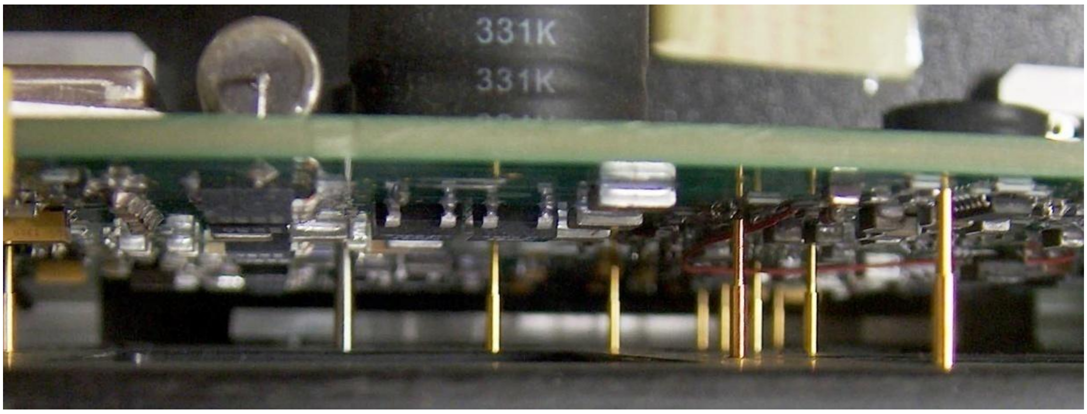



***Figure 2.4-Example of Bed of Nails***

Conducting probes of *ICT* test fixture is used to detect opens and shorts at the circuitry as well as missing, wrong or improperly oriented components. As a philosophy *ICT* tests generally do not have any good or bad assumptions about components until it is tested.

Compared to the manual test technics, *ICT* drastically decreases testing time, increases ease of operation and maintenance. [12]

Above mentioned properties of the *ICT*, makes it desirable testing technique for decades, but as consequences of changing board topologies and architectures, this technique became insufficient to fulfill required testing coverage.

Two of these problems which make conventional *ICT* less desirable are listed below;

- Increased board sizes and crowded part lists start to make impossible to probe every required point on the board one by one using pins and increase required designing time of the tests. In the industry, robotic probes are proposed to solve this problem.

Basically, a couple of programmable robot arms will do the same thing with static probes which are continuously contacting the pre-defined location.

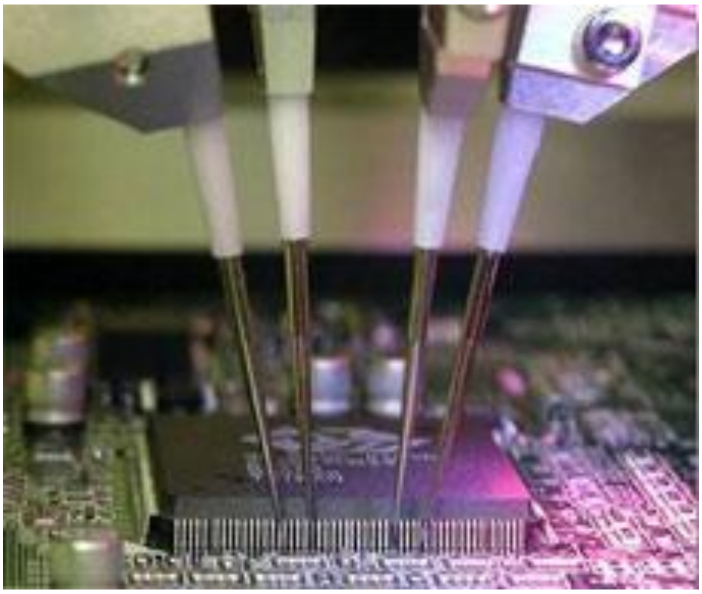


***Figure 2.5-Moving robotic probes***

This proposed probing technique brings a new problem with itself which is having no control on the applied force to the component and solder joint. This incapability of controlling the applied force can give harm to the board and component surfaces.

In order to overcome above mentioned problem, advanced robotic systems are designed whose general name is flying probe. [23]

Details of flying probe machines and information about their design will not be given in this study.

Rapid development of the semiconductor industry results in smaller and much more complex components. Some of these complex components like FPGAs (Field Programmable Gate Arrays) have hundreds of pins on it. In addition to being out numbered it is harder to test this pins because they are not basic resistors where you can apply a signal and wait a linear response.

Also these hundreds of pins are so close to each other which make impossible to use the new invented flying probe techniques.

Applying functional tests is the trivial solution of this problem but industry pioneers have proposed a new method to overcome the testing problem of complex semiconductors, which is named ***boundary scan testing.***

### 2.2.2 Boundary Scan Testing

Boundary Scan Testing technique is proposed to solve the problems emerged after surface mount, miniaturized components, multi-chip modules, and complex, application specific integrated circuits' introduction in the industry. Technique claims to extend the testing coverage on the boards by replacing *ICT* techniques' component direct contact principle with a newly proposed testing architecture, which mainly removes the physical barrier introduced by the limited access to the test points. [15]

Boundary Scan architecture is designed and proposed by the Joint Test Action Group and IEEE has standardized this architecture [14]. Actually boundary scan technique is a collective method where integrated circuit manufacturers must apply architecture to their *IC*s, board designers must consider this architecture while constructing the topology and test designers must use the given topology and described methods to increase coverage.

Integrated circuits are called boundary scan compatible (or *IEEE 1149.1 compatible*) if the required integral architecture is implemented by the manufacturer of the chip. [14]

IEEE 1149.1 compatible *IC*s are designed and manufactured as having shift registers between each device pin and internal logic of the chip. A reference model for the defined architecture of IEEE 1149.1 is shown in Figure 2.6.

Standard definitions of the major concepts of the boundary scan architecture can be found in [14].

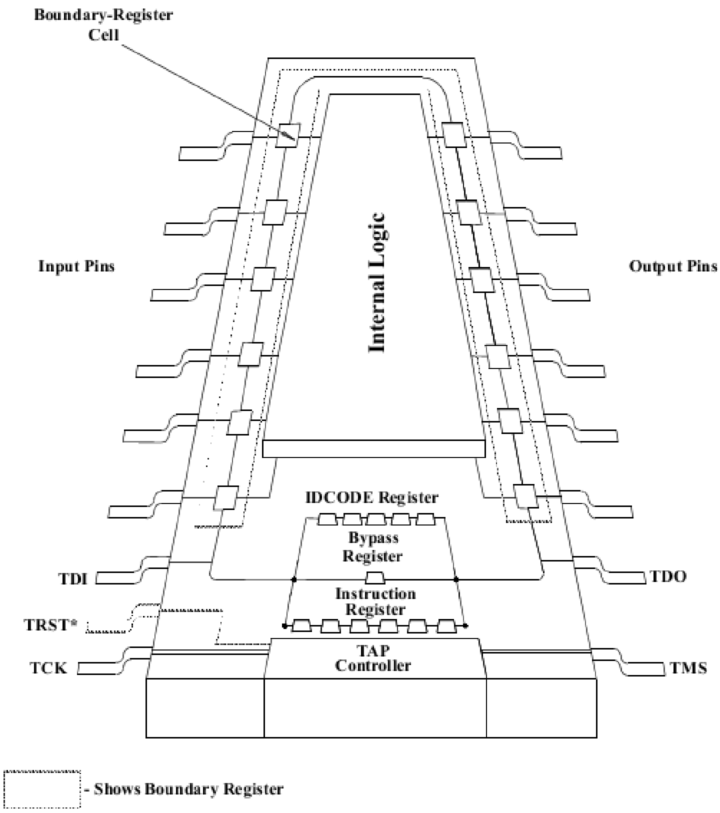



***Figure 2.6-Simplified IEEE 1149.1 Compatible IC internal Structure [14]***

Each shift register of this topology is called a boundary scan cell and these cells give the ability to monitor inputs and control outputs of the every pin on the chip. Connection of these individual cells together constructs the data register chain and is called as the boundary register.

By using data registers of the components and applying a pre-defined testing vector of the logic values to the *TDI*, test coverage of board can be maximized in all boundary scan compatible boards.

There are many discussions in the literature about test vector creation philosophy and standardized methods but details of these discussions and patterns are not given in the scope of this study.

Although boundary scan patterns maximize test coverage ideally on a boundary scan compatible board, in reality boards have conventional and non-boundary scan compatible components. These mixed type boards illuminate some deficiencies of the boundary scan method which are [15]:

- Shorts on the *pure nets* (nets between two boundary scan compatible components) can successfully be detected whereas shorts on the *impure nets* (nets between boundary scan compatible and non-boundary scan components) can only be reported when the non-boundary scan net is not in a high impedance state.

- Faults on pure nets can not be detected by boundary scan infrastructure if a conventional tristate component is also connected to the nets.

- Boundary scan test methodology can not detect missing component on the board if the component do not have any connection with boundary scan compatible *IC*s.

Test designers in the industry who are responsible for deciding on the testing method and equipment generally prefer using a combined system for tests which includes both *ICT* probes and boundary scan infrastructure. By this way both listed deficiencies of *ICT* techniques and boundary scan methodologies can be overcome by this mixed type device.

### 2.2.3 Functional Testing

Functional testing is basically simulating the working conditions of the board in terms of inputs and measuring the outputs of the board. These measurements are compared by the expected outputs of the boards.

Designing functional tests is a time consuming processes since it requires deep information about the design, functionalities and requirements of the board.

In another perspective, designing functional tests is hard because it requires expert test engineers who know the required communication protocols and available industrial equipment to simulate required signals.

Although it is hard to establish a functional test, they are inevitable tests. Combination of *ICT* and Boundary Scan test can guarantee that there is no missing components on the board and all components have right orientation, good solder, working connections, with a very high success rates. However the historical debate states that "*having all components on the board and in the right place does not guarantee that the board is functioning well*." [12] Therefore an additional functional test is required to be sure whether the board is working or not. Functional testing is important especially in defense like industries where the functions of the boards are critical and risk of having a malfunctioned board cannot be taken.

## 2.3 Evaluation Metrics of Test Equipment and Methods

Test equipment cost not only significant amount of engineering effort at design phase but also large instrumentation budgets at implementation phase. As a consequence of this great amount of spending, the industry has tried to find a method to evaluate the final system and compare the results with the old used system if it is possible. [8]

Every company in the industry has different priorities and benefit calculation philosophy, therefore several evaluation metrics can be found in the usage.

In this section, evaluation metrics and their mechanisms will be explained.

- **Test Equipment Setup Time:** This metric measures required time to make the test equipment ready for test. Time can be measured in minutes, seconds or hours according to the companies or investigators choice. This metric is used at the industry to compare two test systems in terms of the operational speed and time management. [33] [34]

- **Labor Skill:** The minimum skills that operator must have to use the test equipment are used as an evaluation metric. In general, hourly wage of the operators which increase by increasing skills, are used as a numeric comparison criteria. This metric gives a chance to compare systems in terms of the extra training cost that they introduce to company. [16][33]

- **Test Development Time:** This metric measures the required engineer time to complete the overall test set. When the high wages of the engineers are taken in to account, this metric could easily be used for economical comparison. On the other hand, for a company who has market pressure on its product could use this metric to choose the quicker solution for its testing problem. [8][31][33]

- **Test Coverage:** The percentage of tested area on the boards in overall board topology is used as an evaluation metric. 100% test coverage which means testing all testable areas, is the ultimate desire of the companies but it is not realistic. Therefore test systems are compared by using their coverage percentages to decide which is better for the company. [32]

- **Test Hardware Cost:** Initial capital costs of the test systems are used as an evaluation criterion in the industry. Hardware cost includes the test resources, technicians' man hour to build the system and other supporting materials. Cost can be measured with the currency decided by company, but the general attitude is using the *USD* in evaluations. [8][33]

- **Verification and Validation Time:** With this metric, test strategies and methods are compared in terms of their verification/validation requirements for each new designed board. For a modular system, only verified part is the board specific designed equipment where as for the manual strategy all used components and instruments must be included in the verification/validation process. Therefore two systems can be compered by using this metric to compare their required time between the developments, when the development processes are completed and test equipment is officially on the line.[31]

- **Fault Detection Rate:** There are two different calculation methods for this metric. The first method uses the ratio of faulty boards in total produced boards as a comparison criterion. On the other hand, the second method uses the percentage of rejected boards at high level tests (i.e. integration and system acceptance tests) in whole board level tested ones. These both attitudes give different insights about testing and production processes, the appropriate method should be chosen by company. [28][30]

- **Overall Operator Time:** This metric compares two test systems in terms of the required human resources for test processes. This time span starts at the moment that operator start to make the equipment ready and ends at the moment that he/she finalize test process by putting test equipment to the appropriate places. Spent time can be measured in minutes, seconds or hours according to the company's or investigator's choice. [31][33]

- **Test Run Time:** Measures the time spent from the moment that first case is run until the test report is generated. Spent time can be measured in minutes, seconds or hours according to the companies or investigators choice. This metric shows the required time for just testing a board and gives a chance to make realistic testing plans. Two test systems can be compared by using this metric to point out which system is quick at testing operation.[8][21][31]

# CHAPTER 3

# PROBLEM DEFINITON AND SOLUTION

Reliability is a key issue in military defense electronics industry. It is imperative in a combat functioning equipment since no one cares about how much money would be taken from warranty after the operation [17]. Therefore, it is important to detect the failures of the equipment correctly and quickly, more than any other industries.

Military electronics are complex and they are also high technology devices. Therefore, the desired coverage levels of the tests are higher than standard industry requirements; i.e. almost complete functional coverage is mandatory. These special properties of the military electronic industry directly affect the test design processes, designing a testing environment for these complex systems requires plenty of engineering time and source. Both of these result in an increase in the total testing cost of the final product.

Total numbers of the manufactured final systems are in very low volume in military electronic industry when compared with mass production industries. Therefore, the testing cost mentioned above will be shared by a limited number of final products, which means test development phase becomes one of the key factors that determines the cost of the final product.

As it has been mentioned above, testing cost becomes a major element of the production budget. In other words, to obtain cheaper final systems, companies should reduce the testing costs.

Reducing the testing cost can be achieved in two ways;

- <u>Decreasing the scope of the tests</u>: With this method engineering and test resource costs can be reduced dramatically but it also means that final products lost their reliability as a result of narrowing the fault coverage. In other words, decreasing the testing scope introduces a tradeoff between costs and reliability, but this tradeoff is usually unacceptable due to the special requirements of the military electronics market.

- <u>Increasing the efficiency of the test design procedures:</u> As mentioned above, these are complex systems so designing similar test routines and testing devices for each system one by one consumes too much engineering time. Making modular designs which allow the reuse of design and other resources can increase the design efficiency and decrease the cost of testing for the special device.

Although military electronics companies are designing devices for different operational environments *(air, marine and land),* final products are similar in terms of their electronic board topologies. The features that differentiate them from each other are generally about physical design *(aerodynamic, radar invisible, etc.)* or endurance requirements.

Designing test software, equipment, and procedures for each new system from scratch, leads to the requirement of verification for this newly designed test components. Also, as a result of being a new design, reliability of these components are not very high when compared with the pre- tested and used designs.

Having modular test equipment and software designs, which can be used for different products of the company, not only decreases the cumulative cost of developing tests for more than one system, but it also increases the reliability of the tests by verifying them again and again on different systems.

It is clearly shown in [18] that automatic test equipment (*ATE*) increases the efficiency of the testing procedure and decreases the total testing costs. Also, [22] proposes that *ATE*s can fullfill the requirements which are being fast, cheap, automatic and able to be used by unskilled labor force, for testing.

By using *ATE*s [18] it has been shown that:

- Required skill level expectancy from test operators can be decreased which means low waged, low experienced operators can be hired to apply test procedures.
- Overall test application time can be decreased up to %48 percent which directly means to increase test equipment throughput.
- Diagnosis time of the faulty boards can be decreased up to %76 percent.

By combining [18]'s study and efficiency incremental solutions mentioned above, the newly designed test equipment should be automatic and modular, which not only increases the test reusability but also decreases the test application time.

The concepts that have been discussed for the general military electronics industry is also applicable for ASELSAN which are;

- Low volume of manufacturing,
- Demand for high reliability,
- Demand for low price,
- Testing complex, high technology systems.

In ASELSAN, similar boards are produced for different projects and come to the test design phase at different times.

These boards generally have the same communication peripherals, nearly the same power requirements and slightly different topologies.

In the conventional test development procedure of ASELSAN, test equipment and software of these nearly the same boards (in terms of testing perspective) are designed separately and generally by a different design engineer. This causes a waste of engineering time and test resources. Moreover, boards are not tested in the same manner due to having different test designers.

The proposed method to solve these long standing problems can be formulated as;

i. By using domain engineering principles of the *SPL*, investigations will be made on the newly designed boards.

ii. By using domain information that will be mined from the investigations, a domain based test system will be designed.

iii. The test system will have a modular architecture. This architecture will consist of two parts; a stable part (*ATE*) and a changeable part.

iv. The required measurement devices, power sources, oscilloscope, relays, signal generators, and simulation devices will be ready in the stable part.

v. Changeable part of the test system will be a test fixture which will have been specially designed for the board. This fixture can include cable connections and any other special equipment for the test of that specific board.

vi. The designed *ATE* will have general software, which runs independently from the tested board. This general software will initialize the required instruments and also it will make sure that a standard test report is generated by the board test.

vii. Each board which has its own test fixture will also have test software. This board specific test software will be an input to the main software of the *ATE* mentioned above.

In 1950s, *ATE* technology was proposed to solve the problems of military maintenance procedures. The engineers of those first *ATE* systems had a common mistake which was adding any available instruments on the market to the design to confront possible future needs. As a consequence of containing several instruments, most of which would never be used throughout the *ATE* lifecycle, overall production cost of *ATE* system increased enormously and industry came to the decision point about whether to give up the *ATE* methodology or revise the *ATE*'s design philosophy. The design attitudes which have been given above with the numbers **i** and **ii**, are proposed to decrease the total initial capital cost of the *ATE* system without causing any insufficiency in the tests. [24][25][26]

The industry prefers modular *ATE* design which has been described at numbers **iii, iv,** and **v**, not only to manage and plan product line related problems separately from product specific ones but also to isolate the general test equipment topology from board specific requirements. This attitude proposes two main modules, the module which contains measurement, switch, simulation, and signaling devices is called as stable part in this study (**iv**) but it has different names in the different studies like core equipment or universal common module. The board specific module which has wirings from board to stable parts and required special equipment is called as changeable part where it can be found as interface device, dedicated module, interface test adaptor or *UUT* adaptor in different studies.[8][16][25][27][29]

It is a known fact that *"no ATE is complete without its software"*. [24] But the studies show that the cost of test software has become the main portion of the total test capital cost. [31] Therefore, the modular test software development approach which has been described in **vi** and **vii,** is proposed by the developers. The module whose duties are initializing, controlling the instruments, detecting broken *ATE* parts, realizing the general purpose operator interface is called as system software or management software in the related studies (as in **vi**). [28][29] The second module which is responsible for applying test related signals to the board and making comparisons according to the given limits, is called as application software or testing software(as in **vii**). [28][29] By the help of this separation programming time is significantly decreased and as a consequence of operator interfaces commonality, human based errors are decreased. [31]

By applying the methodology mentioned above, a standard approach to board testing will be established for the domain which has been investigated.

This standardization of test development process is expected to result in;

- **Decreased test development time:** Using the *ATE* system and its management software simplifies test development process to the setting the parameters of devices and mapping board signals to *ATE* resources. These simplifications linearly decrease the required engineer time to develop test. [31][35][37]

- **Decreased test hardware cost:** The total cost of *ATE* system will be shared by the whole focused board family. This will dramatically decrease the instrument ownership cost of the test system. In addition to this, there will also be a significant decrease in the board specific test equipment manufacturing errors as a consequence of simplifying it to the mapping between board and *ATE*. The dramatic decreases both in instrumentation and material costs will lead a decrease in total test hardware cost. [25][33][37]

- **Decreased set up time:** Proposed solution simplifies the test setup procedure in a way that the only required actions are connecting board specific test equipment to the *ATE* and starting the general *ATE* software. This simplifications and automations decrease the required operator time to start the test. [33] [34]

- **Increased test coverage**: Capabilities of test environment are increased by the use of *ATE*s and also, design engineers start to spend more time on the actions, which will increase the test coverage after the simplifications at the development phase grant extra time to them. Automation also speeds up the test procedures, as a result of that, the test cases which were intentionally subtracted from test procedures to shorten the test time will be added back to the test sets. Having a larger test set and engineering effort will result in an increase in test coverage. [30][32]

In the next chapter, details about application of the methodology mentioned above in ASELSAN's tests will be presented and results will be analyzed.

# CHAPTER 4

# IMPLEMENTATION

Low production volume in the defense electronic industry is mentioned above. As a consequence of this fact, all actions related with cost minimization should obey the "*economy of scope*" approach which is explained in Chapter 2.

This approach basically states to increase the commonalities of the final products in order to share the total design cost of the systems. That is also the main idea of *SPL* approach and the board test system which will be designed and implemented in this chapter use this main idea.

By using domain engineering strategy of the *SPL* approach, appropriate scope for the test system will be decided later in this chapter.

In this implementation, a specific project of the ASELSAN had been taken into account and from this point on, the project will be called as Project T (*PT*) because of confidentiality issues.

*PT* was designed and developed for the Turkish Air Force which means that it shall obey all the applicable regulations and principles of air combat vehicles. This complex system consists of **95** individual modules. Although these modules differ from each other in their functionalities and physical properties, they can be assembled in four major groups.

- *Electronic Boards* group consists of any module which is implemented on a *PCB*;
- *Optical Modules* group consists of any module which is related to the optical functionality of the overall system;
- *Wiring Modules* group is responsible for the connection of the other modules with each other;
- *Others* group is collection of the modules which are not the part of the above mentioned groups. Member list of this group includes heater fans, special design heat sinks, spring sets, coolant materials, electric motors.

Distribution of these **95** modules to the above mentioned groups can be seen in Figure 4.1.

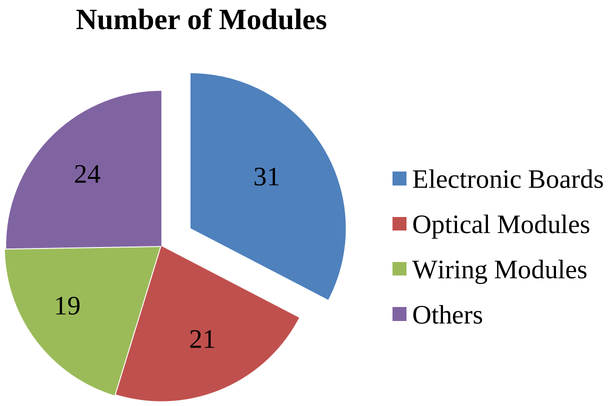


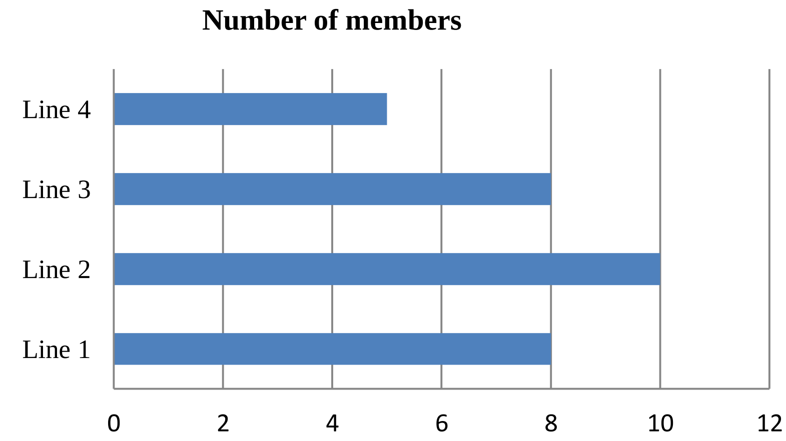

***Figure 4.1 - Distribution of the modules in PT to four categories***

All of these modules need to be tested before being integrated to the final system which means testing procedures and techniques for these materials should be designed. These four groups have different characterizations and therefore different testing approaches should be applied to them.

In the focus of this study, *electronic boards* group which has 31 members had been investigated. All the members of the group have different functional behaviors and the final system expects distinct duties from them. Trying to design a system that tests all of these distinct members, results in enlarging the scope of the system to inefficient borders. Therefore, another grouping will be appropriate for the sake of efficiency.

By analyzing the main functions of the boards and the company *production strategy* which is stated as the required input of the domain engineering at [5], four different families are generated in *PT* which are;

- **Line 1:** Power generation, power conversion and high voltage boards.
- **Line 2:** Communication boards, processor boards, digital boards with a specific duty on the system.
- **Line 3:** Flex, backplane or main boards which are only has hardwired lines.
- **Line 4:** Outsourced boards which are designed by ASELSAN but will be produced and tested by sub-contractors.




| | Line 1 | Line 2 | Line 3 | Line 4 |
|---|---|---|---|---|
| Number of members | 8 | 10 | 8 | 5 |

***Figure 4.2- Distribution of Electronic Boards of PT to the Families***

The focus of the present study was narrowed down to the ***Line 2*** family which is the most crowded and technologically complicated family. This decision drawn the border line to the scope of the investigations which was done to develop the *ATE* to test ***Line 2*** family members and similar products of the future projects.

Decisions about requirements for both hardware and software architecture of the *ATE* will be taken at the below by using the results of the detail analysis and futuristic projections.

## 4.1 Hardware Design

As mentioned in chapter 3, designed hardware have two blocks. First block which is called as ***ATE***, will be used by whole focused family members and was designed by using domain engineering principles. Second block which is called as board specific test equipment (***BSTE***), was designed and used for just one specific member of the family, this design and development actions of *BSTE* depend on the application engineering point of view.

### 4.1.1 ATE Design

Input requirements of the family members, expected outputs and functional observations are analyzed to design the *ATE* of the system. In addition to these factors, available off the shelf components and possible future members of the family are considered to have a maintainable and efficient product line testing system. Market and industry knowledge become key ingredients to have efficient and applicable test system design which means experiences of the designer will have a great role in this phase.

All required hardware modules will be analyzed and determined below.

#### 4.1.1.1 Power Source Devices

Voltage requirements of the family members are listed at Table 4.1.

***Table 4.1-Line 2 Family Members' Power Requirements***

| **Board** | **Voltage (V)** | **Max.Power (W)** |
|---|---:|---|
| m1 | +5.0 | 10 |
| | +3.3 | 5 |
| m2 | +3.3 | 9 |
| m3 | +3.3 | 5 |
| | +5.0 | 8 |
| | -5.0 | 8 |
| | +12.0 | 35 |
| | -12.0 | 35 |
| m4 | +5.0 | 5 |
| | +3.3 | 5 |
| m5 | +5.0 | 5 |
| | +3.3 | 6 |
| m6 | +5.0 | 15 |
| m7 | +3.3 | 3 |
| | +5.0 | 2,5 |
| | -5.0 | 2,5 |
| | +12.0 | 40 |
| | -12.0 | 40 |

*Table 4.1-Line 2 Family Members' Power Requirements(Continued)*

| | | |
|---|---|---|
| m8 | +3.3 | 2 |
| | +5.0 | 10 |
| | -5.0 | 10 |
| m9 | +28.0 | 95 |
| | +12.0 | 45 |
| | +5.0 | 2,5 |
| | +3.3 | 1 |
| m10 | +3.3 | 2 |
| | +5.0 | 3 |

Requirements for five different voltage levels which were used at the boards are given above. The purpose of this study is to design a test system that can test all the family members. Therefore, the meaningful data of Table 4.1 are the maximum power requirements of each voltage levels which will draw the border lines to the power limits of *ATE*.

By inspecting Table 4.1, maximum Voltage/Current values can be decided as;

| | | |
|---|---|---|
| Voltage Level 3.3 V | Voltage Level 5 V | Voltage Level -5 V |
| Max. Power= 6 W | Max. Power= 15 W | Max. Power= 10 W |
| Max. Current= ~1.85 A | Max. Current= 3A | Max. Current= 2A |
| Voltage Level 12 V | Voltage Level -12 V | Voltage Level 28 V |
| Max. Power= 45 W | Max. Power= 40 W | Max. Power= 95 W |
| Max. Current= 3,75 A | Max. Current= ~3.35 A | Max. Current= ~3.40 A |

*Minimum number of power sources* which is required to boot all family members in their respective test sequence is five because of the fact that family member ***m7*** needs that number of sources.

In addition to booting up the *UUT*, extra power necessities can exists in the test setups. Using specially designed test boards to test *UUTs* are very common in ASELSAN, as can be seen in Figure 4.3., **16%** of all boards' and **60%** of boundary scan (*BS*) compatible boards' test setups (Figure 4.4.) consist *BS* test boards. In this perspective adding one more power source to the designed system makes sense in order to test all family members under any required condition.

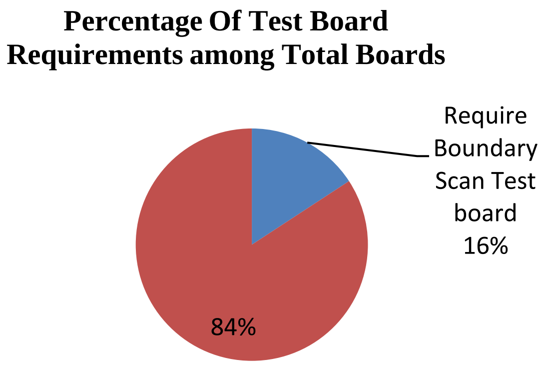



***Figure 4.3- Percentage of Test Board Requirements among Total Boards***

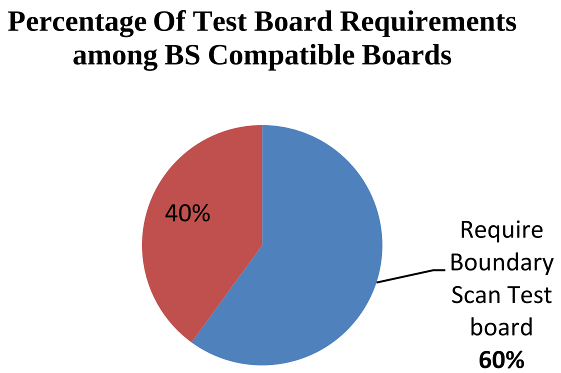



***Figure 4.4-Percentage of Test Board Requirements among BS Compatible Boards***

Number of total power sources was decided as six above. At the point of deciding specifications and brand of these six sources, *preexisting company assets*, as mentioned in 2.1.1, have great importance. Using pre-used brand and products minimized the required time to define maintenance and calibration procedure of these devices. This also drastically decreased the amount of time required to learn to manage this device by using its application programming interface (*API*) or dynamic-link library (*DLL*), supplied by manufacturer.

By above mentioned considerations power source devices of *brand X* were chosen as;

***Table 4.2-Power Source Devices Specification List***

| **Source Name** | **Specifications** |
|---|---|
| Pwr Source 1 | Max.Voltage:**8 V** Max.Power:**50W** Max.Current:**6.25A** |
| Pwr Source 2 | Max.Voltage:**8 V** Max.Power:**50W** Max.Current:**6.25A** |
| Pwr Source 3 | Max.Voltage:**35 V** Max.Power:**105W** Max.Current:**3A** |
| Pwr Source 4 | Max.Voltage:**35 V** Max.Power:**105W** Max.Current:**3A** |
| Pwr Source 5 | Max.Voltage:**60 V** Max.Power:**750W** Max.Current:**12.5A** |
| Pwr Source 6 | Max.Voltage:**60 V** Max.Power:**750W** Max.Current:**12.5A** |

Choosing all sources with maximum ratings among off the shelf sources was the easiest way to decide sources but the purpose of this study is to decrease the total testing cost. Therefore; to decrease the cost of total *ATE*, sources were selected by carefully planning.

*Pwr source 1* and *pwr source 2* which have low voltage and power ratings, fullfill the low power requirement of voltage levels 3.3 and 5.

*Pwr source 3* and *pwr source 4* which have high voltage rating but low current rating, can be used to supply voltage levels of 12 and -12.

*Pwr source 5* and *pwr source 6* are chosen to prevent the system to be useless against future member of the family which has high power and voltage requirements.

#### 4.1.1.2 Communication Protocols

The industry has hundreds of different protocols, implementing a system which has the ability of applying all these protocols are not only impossible but also irrational. Therefore; protocols that are used by the line 2 family were investigated and the findings were listed in Table 4.3.

***Table 4.3-Line 2 Family Members' Communication Protocols***

| Board | Protocol | #of channels |
|---|---|---|
| m1 | Rs422 | 2 |
| m2 | ARINC-708 | 1 |
| m3 | Rs232 | 2 |
| | Rs422 | 1 |
| | Rs485 | 1 |
| | MLT-STD-1553 | 2 |
| m5 | ARINC-429 | 8 |
| m8 | MLT-STD-1553 | 1 |
| m9 | Rs232 | 3 |
| m10 | Rs232 | 2 |
| | Rs422 | 1 |
| | Rs485 | 1 |

Below devices were chosen among the off the shelf components in order to implement the communication requirements of the line 2 family.

***Table 4.4-Communication Devices Specification List***

| Device Name | Specifications |
|---|---|
| CommDev 1 | 2 channels MLT-STD-1553 communication |
| CommDev 2 | 10 channels ARINC-429 communication |
| CommDev 3 | 2 channels ARINC-708 communication |
| CommDev 4 | 4 channels configurable Rs232/422/485 communication |

#### 4.1.1.3 Boundary Scan Devices

Decisions about boundary scan had two main subjects. The first one was whether or not to implement boundary scan infrastructure to the *ATE*. Investigations, which were summarized as Figure 4.5, on the *Line 2* family boards show that 80 % of the members have *BS* compatible components.

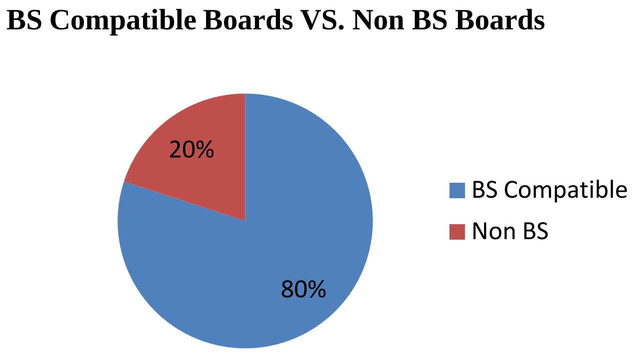



***Figure 4.5-BS Compatible Boards VS. Non BS Boards in Line 2 Family***

To increase the test coverage of **80 %** of the family boundary scan infrastructure was decided to implement and in the next phase, a second question arises: What kind of hardware will be chosen?

As an industry standard, boundary scan hardware does not work individually, to make it functional, the responsible person must purchase and install license files and application software. These software and license bring extra cost to the system and not only to avoid this extra cost but also to have companywide similar infrastructure, *Brand A*'s products had been selected for use in this study. ASELSAN has network licenses for *Brand A*'s products which means any *ATE* connected to the ASELSAN network can use the software without any extra charges.

*Brand A* has several boundary scan hardware, so, to choose the correct model information about BS topologies of the family members were required. By investigating family members, the topology was constructed.

***Table 4.5-Number of Scan Chains Line 2 members have***

| **Board** | **Number of Scan Chain** |
|---|---|
| m1 | 1 |
| m2 | 2 |
| m3 | 1 |
| m4 | 1 |
| m5 | 1 |
| m8 | 2 |
| m9 | 2 |
| m10 | 1 |

*Brand A* has one port and 4 port *BS* hardware, by using Table 4.5, four port hardware was chosen, whose specifications can be seen below.

***Table 4.6-BS Device Specification List***

| **Device Name** | **Specifications** |
|---|---|
| BS Device | 4 port, 1.8/2.5/3.3/5.0 V, PCI Boundary Scan Hw |

#### 4.1.1.4 Digital Input/output Devices

Digital output signals are used to set a board signal to *HIGH* or *LOW* states where as the digital inputs are used to read the state of the signals.

*HIGH* state voltage level was decided by board design therefore; to set the specifications of digital IO device, *line 2* family was investigated.

5 volts, 3.3 volts and '*Low voltage differential signaling*' were observed as three different digital signaling standards on the family. To fulfill these three standards above devices were selected;

***Table 4.7-Digital Input Output Devices Specifications List***

| **Device Name** | **Specifications** |
|---|---|
| DIO Device 1 | 32 channel , 1.8/2.5/3.3/5V level adjustable PXI DIO |
| DIO Device 2 | 32 channel , 1.8/2.5/3.3/5V level adjustable PXI DIO |
| DIO Device 3 | 16 channel , LVDS PXI DIO |
| DIO Device 4 | 16 channel , LVDS PXI DIO |

#### 4.1.1.5 Analog Output Devices

Analog outputs are used to set board signals to the desired voltage levels; applied voltage level is related to the design and desired function of the board.

Range of applicable analog voltage level was the key criterion to choose the correct device therefore line 2 family was investigated.

Range for analog signals was decided as ±10 volt, so below device was chosen for the system.

***Table 4.8 -Analog Output Device Specifications List***

| **Device Name** | **Specifications** |
|---|---|
| AOut Device | 16 channel, ±10 volts range PXI Analog Output |

#### 4.1.1.6 Measurement Devices

Investigations on *line 2* show that oscilloscope and multimeter are two required measurement devices of the *ATE*.

Having more than one oscilloscope or multimeter devices on the *ATE* is not acceptable in ASELSAN's design culture. Therefore, software controlled multiplexers and relay modules were preferred to populate the inputs of these measurement devices.

Analysis on the *line 2* boards resulted in the topologies seen at Figure 4.6 and Figure 4.7 which used devices of Table 4.9.

*Table 4.9-Measurement Devices Device Specifications List*

| Device Name | Specifications |
|---|---|
| Measurment Device 1 | 6 ½ Digit, 300V-1A Multimeter |
| Measurment Device 2 | 2 Channel,1 Trigger,150 Mhz, PXI, Osilloscope |

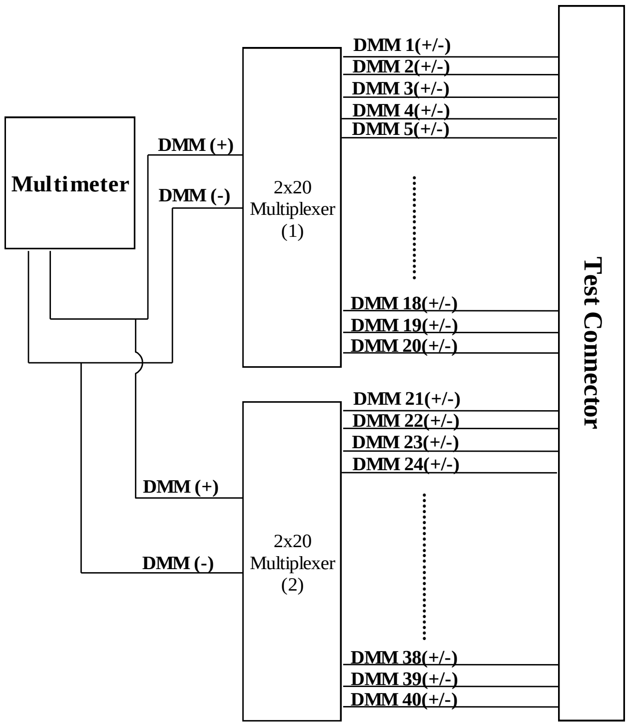



*Figure 4.6-Populating Multimeter Inputs via Multiplexers*

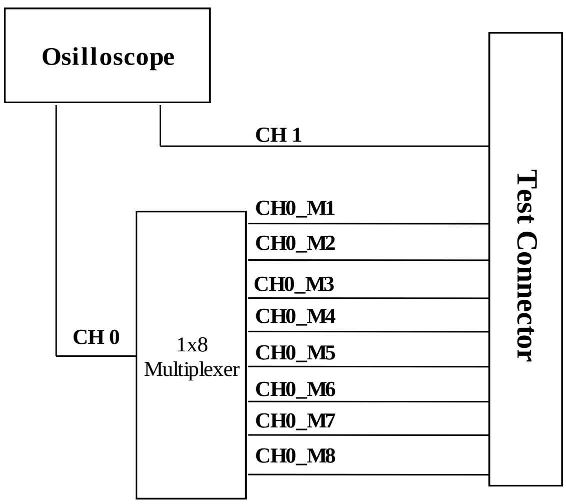



*Figure 4.7-Populating Osilloscope Input via Multiplexer*

The final system layout which had been designed throughout section 4.1.1 can be seen at Figure 4.8.

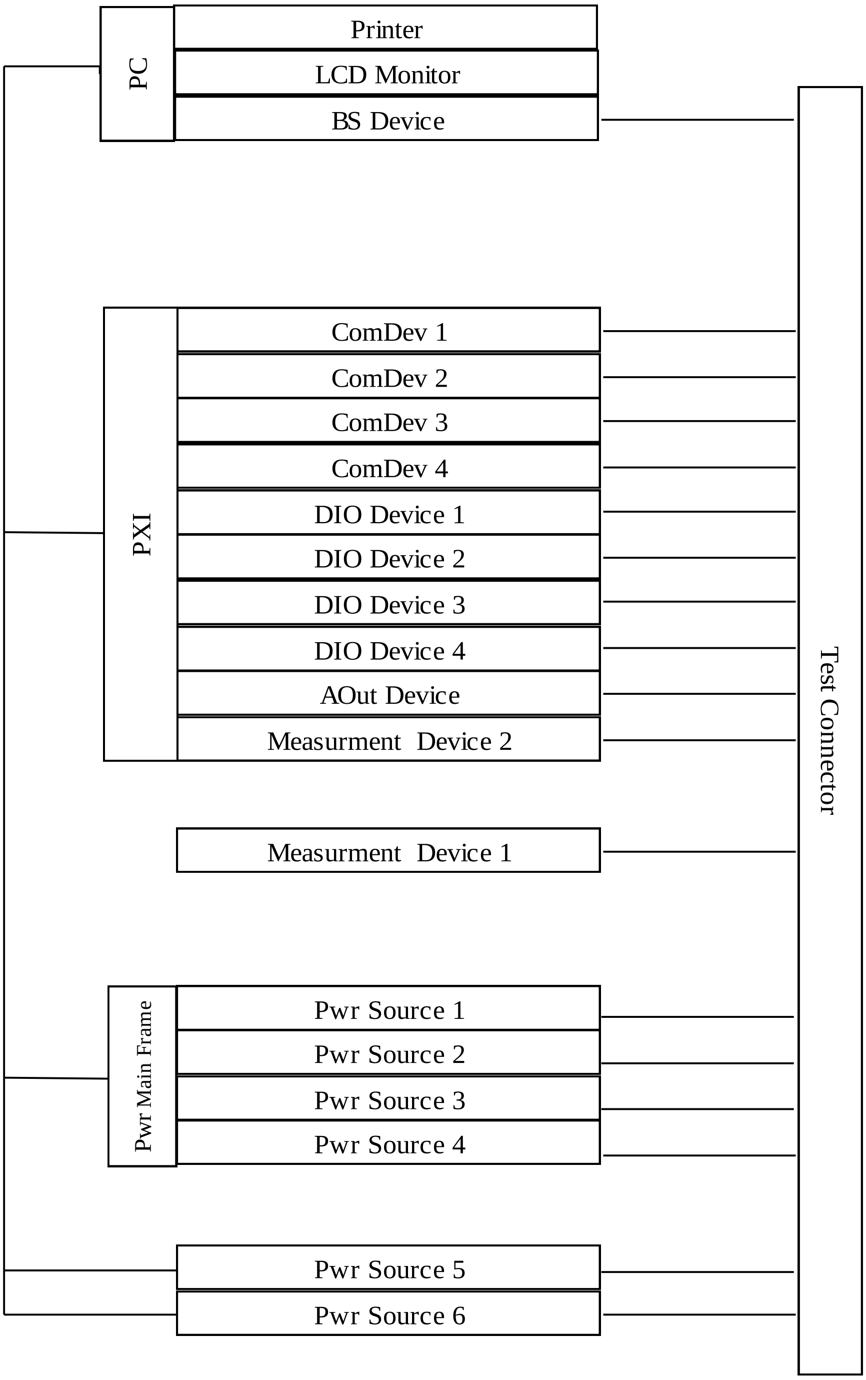



***Figure 4.8-ATE Over All System Layout***

***Figure 4.9-Final Look of the Designed ATE***

### 4.1.2 BSTE Design

*BSTE*s are designed for specific boards and their main duty is being a bridge between electronic board and general purpose *ATE*.

*BSTEs* which are designed in ASELSAN, do not include any active or complex components as a design criteria of the company. In other words, wirings from board connector, test point or probed location to the *ATE*'s specific resources are the only desired elements of the *BSTE*s. On the other hand, in some cases it is inevitable to add some extra equipment to the *BSTE*, like adding a cooling fan for a processor board. In such cases designers are allowed to use extra components by giving the control of this extra component to the *ATE* via appropriate wiring.

The design of *BSTE* is started after *BSTS* design is finalized, which is explained in detail at 4.2.2. At *BSTS* design step test methods and requirements are stated clearly and *BSTE* designer, who generally is the same engineer with *BSTS* designer, uses this information to decide whatever he/she needs to implement that test methods physically.

Steps of *BSTE* design are;

- By looking at test methods engineers decide whether probing is required or not. If probing is required, the coordinates of the probing locations are decided and appropriate mechanical probing tool is requested from mechanical design engineers of the department.

- By looking at test methods, a mapping from board signals to the *ATE* resources is created by the designer to show the technicians which connections should be done.

- Board orientation on the *BSTE* is decided by considering *ATE* connection and possible fault inspection/fault detection activities of technicians for faulty boards.

- Standards of the cables which will be used for transmitting signals from board to *ATE*, are decided by considering applicable standards and previous experiences.

A *BSTE* which were designed and developed for the line 2 family can be seen at Figure 4.10.

***Figure 4.10-BSTE example***

## 4.2 Software Design

Software architecture of the test system consists one general and one board specific software block whose names are automatic test equipment management software (***ATEMS***) and board specific test software (***BSTS***). As in hardware section design and development actions of *ATEMS* part consist domain engineering point of view whereas *BSTS* part consists application engineering.

Test system is intended to be used by test operators and test designers whose use cases are given at Figure 4.11.

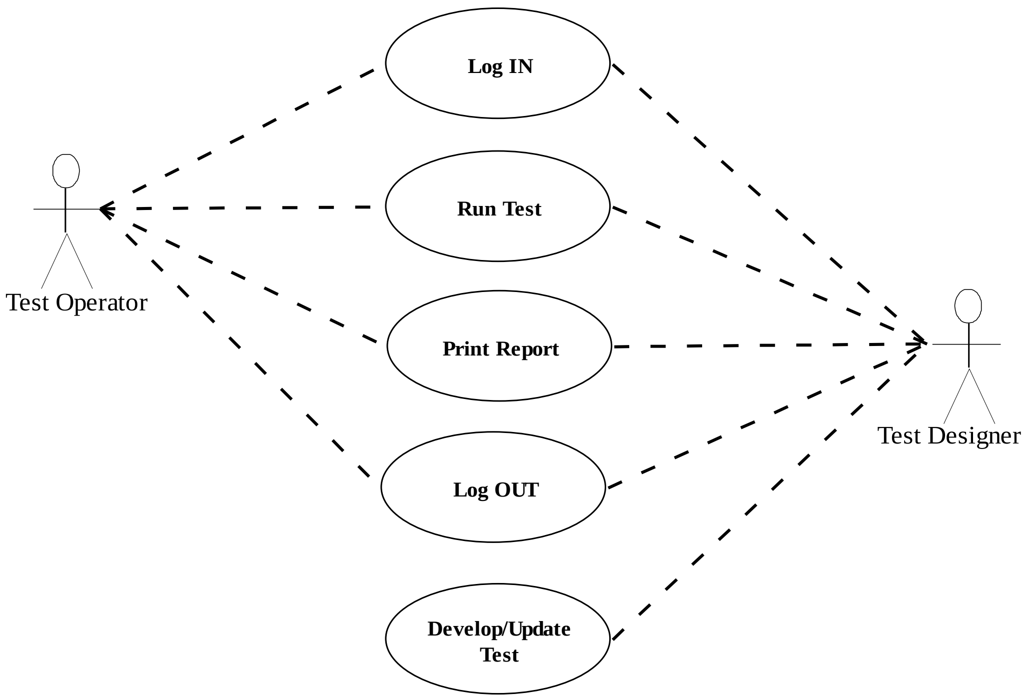



***Figure 4.11- Test System Use Case Diagram***

Software system which is the combination of *ATEMS* and any given *BSTS* should realize above use cases. By investigating above use cases, scope and functions of not only *ATEMS* but also *BSTS* was defined.

Details of *ATEMS* and *BSTS* architecture are given below.

### 4.2.1 Design of *ATEMS*

*ATEMS* is the block that handles the standard procedures of the *ATE* system for every test execution. Setting an appropriate scope for *ATEMS* decreased the cost of each individual *BSTS* by narrowing the scope of engineering work done for each individual boards.

- Log IN
- Log Out
- Run Test
- Print Report

Above four use cases which were taken from Figure 4.11, are standard procedures which means they do not include any board specific information. Therefore; these four use cases must be considered as the focus of *ATEMS.*

*"Develop/Update Test"* use case represents test development activity which is held by engineers. After completion of design phase of the board specific test, the designer decides on the required resources for the test. Actions after that point on can be divided into two main categories.

- **Manage Devices:** At this action block designer initializes the devices and sets the required parameters.
- **Compare Results:** The designer builds a system which compares the result sent by devices and expected results.

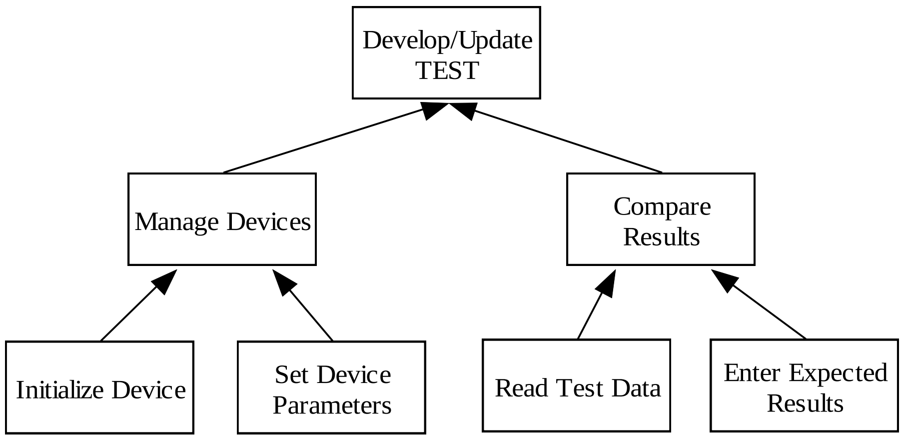



***Figure 4.12-Break Down of Develop/Update Test Action***

Both of these action categories were repeated several times inside the test software wherever necessary.

*Compare Results* category is totally board dependent, but on the other hand *Manage Devices* category includes parts which are suitable for reuse.

Device initializations are very straightforward procedures, which are done by using device *APIs* or *DLLs*. These actions return device *handles* to the programmer in order to let designers to control the device later in the program.

As a consequence of being an inevitable and standard procedure, initialization related actions were included in the scope of *ATEMS*.

Although device parameters are decided and set by the developer according to the board design, this effort can be minimized by using modular device management software components. Therefore, at the *ATEMS* development phase by careful domain engineering system's all required software assets must be planned and developed.

Developers of the *BSTS* which was explained in detail at next section , must use these assets to design their tests.

*ATEMS* was designed by considering above mentioned use cases and actions. The descriptions of main program blocks are;

- **Log in:** Checks the user to decide whether he/she is a designer or operator and grants rights accordingly.
- **Decide Action:** If the user is a designer there are two operation possibilities; run test or develop test.
- **Initialize Devices:** *ATE* resources are initialized and device handles are stored on system.
- **Acknowledge *BSTE*:** Every *BSTE* has a unique hard coded id and *ATEMS* will recognize which *BSTE* is connected to *ATE.*
- **Decide Related BSTS:** *ATEMS* checks the *BSTE* and *BSTS* relation table to decide appropriate *BSTS*.
- **Call Related BSTS:** BSTS of the specific board is called with the resource handles.
- **Get BSTS Results**: After *BSTS* operation is finished, stored test data is read and parsed by *ATEMS*.
- **Generate/Print Report**: A report file according to the ASELSAN standards is generated and sent to the printer.
- **Release Device Handles**: Release action for whole devices are done at this step.

The flow diagram of above program blocks can be seen at Figure 4.13.

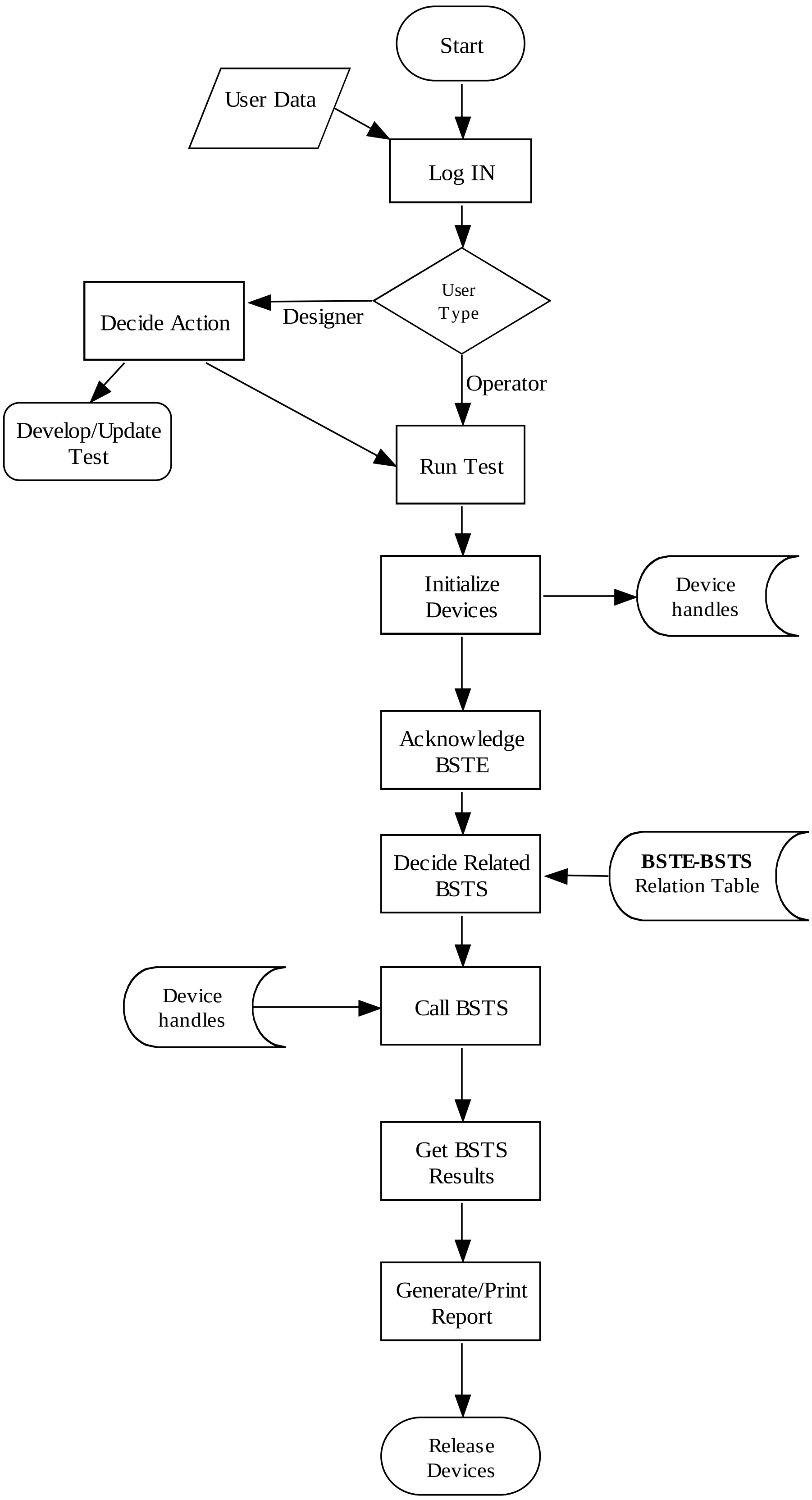



***Figure 4.13-ATEMS Program Flow***

In addition to these program blocks shown in Figure 4.13, management libraries for all the resources mentioned in hardware section, were designed and developed in the *ATEMS* development phase.

These libraries are ready to use for the development of the *BSTS* which will run on this *ATE.*

### 4.2.2 Design of *BSTS*

*BSTS* part is developed by the responsible test designer of each board, separately. Developers must fulfill architectural requirements that *ATEMS* state and provide the required parameters.

In other words, *ATEMS* behaves like a black-box to the *BSTS* which must take handle as an input and gives predefined outputs. These predefined outputs are;

- **Test Results:** Pass or fail information of each individual test step.
- **Test Limits:** Numeric limits or pass/fail criteria of the test steps.
- **Test Measurements**: Measurements read from devices or entered by operators about related test steps.

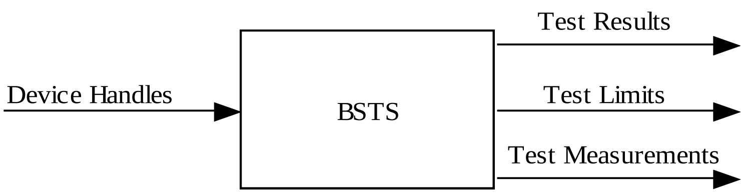



***Figure 4.14-Black Box representation of BSTS***

Although *BSTS* seen as a black box by *ATEMS* (as seen at Figure 4.14), internal design of *BSTS* consists several steps and require investigation in detail on the focused board to have a proper testing method and increase the test coverage.

*BSTS* design and development steps are;

**a.** Board topology is analyzed by the design engineer in detail.

**b.** Test methods are decided for each testable part of the board.

**c.** Limits or pass/fail criteria of each test step is defined.

**d.** Required test resources are decided

**e.** Resources are compared with available resources on the *ATE*. If test design requirements contradicts with available resources designer should return to **step b** and review his/her test design to find alternative methods.

**f.** Test flow which shows the order of test steps is decided

**g.** Conditional relations between test steps are decided. Preconditions and post conditions of each test steps are listed.

**h.** For each test step required state of resources are decided and parameters to set devices to these states are listed.

**i.** Resources are set to the desired states by using parameters defined in **step h** and device management libraries that are developed in *ATEMS* phase.

**j.** Obtained results are compared with the limits defined at **step** c**.**

**k.** Pass/Fail decision of the test is given by the information at **step j.**

*BSTS* have their own revision controls, separate form *ATEMS* and must be updated whenever the related board is revised or a defect about board test methodology is reported.

# CHAPTER 5

# EVALUATION

Test system implementation which is elaborately explained in chapter 0 is proposed to solve the problem defined in chapter 0. In this chapter, the effectiveness of the proposed solution against the manual testing systems will be investigated by using the real life results which are taken from the usage of designed *ATE* by ASELSAN personnel.

In order to obtain reliable comparison possibility, manually tested boards which was used to decide the efficiency of the proposed automatic and modular method, was chosen from a project that is also designed for air forces like *PT.*

The project which has manual board tests, from this point on, will be called as ***Project Z (PZ)***. As a consequence of being developed for the same customer (air forces), ***PZ*** was subjected to the same performance requirements and standards. These similarities gave a chance to have an objective comparison about test devices.

***PZ*** has 8 boards which are similar with the *line 2 family's* boards in terms of technical specifications. Throughout this study these 8 specific electronic boards had been used for the comparison purposes and called as ***L1, L2….L8***, respectively.

In section 2.3, evaluation metrics and explanations about their usage were given.

For this study comparison and performance evaluation of developed methodology was based on the metrics below:

- **Test Equipment Setup Time:** Metric shows the elapsed time from operators start to setting the test system till he/she runs the test plus required time to returning test system to the initial conditions. This metric will be used in <u>minutes.</u>

- **Test Run Time:** Metric shows the elapsed time from the operator run the test until the test report is generated. All test related actions, warnings and messages are included in this time span. This metric will be used in <u>minutes.</u>

- **Test Development Time:** Metric shows time spent for the specific board's test. This time span starts from the moment that engineer starts to inspect the board topology and ends at verification tests which are done with company quality control engineers. When the cost of one engineer hour to the company is considered, the most expensive part of a test system becomes this metric. [16] This metric will be used in <u>hours.</u>

- **Test Hardware Cost:** Metric shows specific board's total test equipment cost which includes test equipment's material cost and technicians' time spent to build. This metric will be used in <u>US dollars.</u>

- **Fault Detection Rate:** There is more than one method to decide this metric. Percentage of faulty boards in total produced board number can be used to compare both above mentioned projects. But this method could not isolate the board production faults therefore is not appropriate to compare two different projects whose production lines are different. In this study, the number of boards which passed the board test but detected as faulty at system integrations tests will be considered. The percentage of insufficient tested boards in total produced amount will be used as a detection rate. This metric will be used in percentage (%).

Some of the metrics listed at 2.3 was not used in this study. These metrics and the reasons were given below.

- **Labor Skill**: In this study labor skill metric is not applicable because there is no classification between test technicians at ASELSAN. All test designs are done as if it would be used by unskilled technicians then the operator of the test is chosen by the department manager by considering technicians' schedules and the test schedules.

- **Test Coverage**: ASELSAN does not have any established methodology or procedure to measure the board test coverage and generate coverage reports. Therefore before using this metric, one should design and establish a proper methodology for the factory. This process requires significant effort and domain analysis that is why "*test coverage*" metric is not considered in the scope of the study.

- **Verification and Validation Time**: In ASELSAN, the verification and validation processes that are held with quality control departments are standard and straightforward. In the other words it is not possible to modify the processes according to the test methodologies. Therefore this metric could not be used for comparing two test systems at ASELSAN's case.

- **Over all Operator Time**: Over all operator time data is the sum of "***test run time***" and "***test equipment setup time***" for the ASELSAN's case. Therefore in this study to have a more insight about processes, "***test run time***" and "***test equipment setup time***" metrics are chosen to be used at evaluation steps. Using "*over all operator time*" metric while the above mentioned two metrics in use, will not give any extra insight about test systems performance or efficiency that is why this metric is omitted in this study.

Metrics explained above were applied to both *PT* and *PZ* boards and results were presented below.

## 5.1 Test Equipment Setup Time

*PT* has an automatic testing system which was designed and developed at above sections. Therefore setup phase of the *PT line 2* board family only included these actions;

i. Connect related *BSTE* to the *ATE*

ii. Connect board to the *BSTE*

iii. Start test sequence

iv. Disconnect board from *BSTE*

v. Disconnect *BSTE* from *ATE*

Time that test operator spent for *PT's* members were listed at Table 5.1.

***Table 5.1- PT member's Test Setup Time***

| **Board** | **Setup Time (min)** |
|---|---|
| m1 | 4,4 |
| m2 | 4,4 |
| m3 | 13,3 |
| m4 | 13,3 |
| m5 | 4,4 |
| m6 | 4,4 |
| m7 | 4,4 |
| m8 | 4,4 |
| m9 | 13,3 |
| m10 | 4,4 |
| ***Total Setup Time $_{PT}$ =71,0 min*** | |

By inspecting Table 5.1.

$$\boldsymbol{Average\ Setup\ Time_{PT}} = \frac{Total\ Setup\ Time\ _{PT}}{\#\ of\ PT's\ members} = 7{,}1\ minutes \qquad (5.1\text{-}1)$$

The electronic boards of *PZ* have manual board specific test equipment (*MBSTE*) which means setup phase of this family include these actions;

i. Operator should read test procedure and learn required resources.

ii. Operator should bring the resources listed in test procedure to the test table.

*iii.* Connections between resources and *MBSTE* should be done by looking test procedure.

*iv.* *UUT* should be connected to the *MBSTE.*

*v.* Test should be started by following steps in test procedure.

*vi.* *UUT* should be disconnected from *MBSTE.*

*vii.* Resources should be disconnected from *MBSTE.*

*viii.* Resources should be carried to the appropriate stocking place.

Examples of *MBSTEs* can be seen in Figure 5.1.


***Figure 5.1-Example of MBSTE***

Time that test operator spent for *PZ's* members were listed at Table 5.2

*Table 5.2- PZ member's Test Setup Time*

| **Board** | **Setup Time (min)** |
|---|---|
| L1 | 13,3 |
| L2 | 20,4 |
| L3 | 13,3 |
| L4 | 20,4 |
| L5 | 16,0 |
| L6 | 13,3 |
| L7 | 20,4 |
| L8 | 13,3 |
| ***Total Setup Time $_{PZ}$=130,6*** | |

By inspecting Table 5.2;

$$\textbf{\textit{Average Setup Time}}_{PZ} = \frac{Total\ Setup\ Time_{\ PZ}}{\#\ of\ PZ'\ s\ members} = 16{,}3\ \ minutes \qquad (5.1\text{-}2)$$

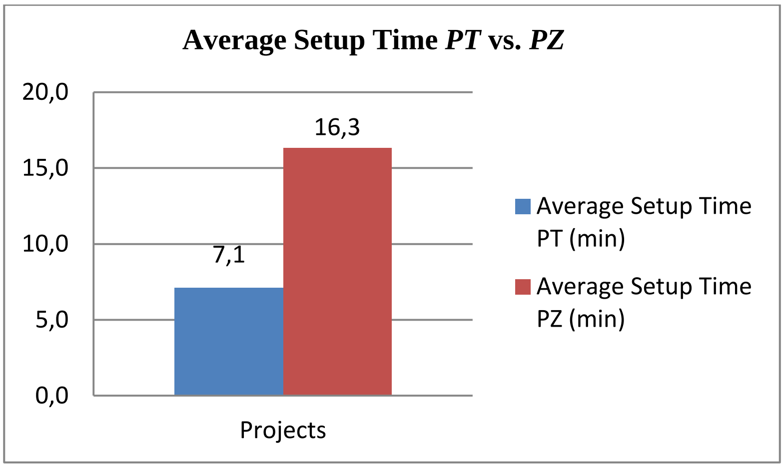



***Figure 5.2-Average Setup Time comparison of PT vs. PZ***

By inspecting Figure 5.2;

$$\boldsymbol{S} = \frac{Average\ Setup\ Time_{PZ}\ \text{-}\ Average\ Setup\ Time_{PT}}{Average\ Setup\ Time_{PZ}}\ x100 = 56{,}4\% \qquad (5.1\text{-}3)$$

where *S* is the percentage of the saved test run time.

By comparing the results given at Figure 5.2, it can be stated that automatic test strategy saves **56,4%** of the required setup time.

## 5.2 Test Run Time

Boards of *PT* have their own *BSTS* which were explained in section 4.1.2, *BSTS* are integrated with ATE therefore only actions that operators must do throughout test are;

i. Running the *BSTS*

ii. Following the warning messages on the computer screen, if exists.

Recorded test time for *PT's* members can be seen at Table 5.3

***Table 5.3- PT member's Test Run Time***

| **Board** | **Test Time (min)** |
|---|---|
| m1 | 16,0 |
| m2 | 16,0 |
| m3 | 20,4 |
| m4 | 20,4 |
| m5 | 24,0 |
| m6 | 16,0 |
| m7 | 24,0 |
| m8 | 19,9 |
| m9 | 26,7 |
| m10 | 16,0 |
| ***Total Run Time*** $_{PT}$**= 199,5min** | |

By inspecting Table 5.3

$$\textbf{\textit{Average Test Run Time}}_{PT} = \frac{Total\ Run Time\ _{PT}}{\#\, of\ \ PT's\ members} = 19{,}9\ \ minutes \qquad (5.2\text{-}1)$$

Test operators have test procedures and *MBSTE*s to test *PZ*'s boards. As a consequence of not having *ATE* and test software, testing activity of *PZ* members becomes;

i. Operator follows the test procedure step by step.

ii. Operator takes measurements by any required measurement device when procedure requests.

iii. Measurement results are stored by operator to a predefined test log document.

iv. Operator applies signals to *UUT* when procedure requests.

Time that spent for testing of each *PZ* member was listed at Table 5.4.

***Table 5.4-PZ member's Test Run Time***

| Board | Test Time (min) |
|---|---|
| L1 | 90,0 |
| L2 | 120,0 |
| L3 | 26,7 |
| L4 | 120,0 |
| L5 | 60,0 |
| L6 | 60,0 |
| L7 | 150,0 |
| L8 | 50,0 |
| ***Total Run Time $_{PZ}$=676,6*** | |

By inspecting Table 5.4;

$$\boldsymbol{Average\ Test\ Run\ Time_{PZ}} = \frac{Total\ Run\ Time\ _{PZ}}{\#\, of\ \ PZ'\, s\ members} = 84{,}6\ \ minutes \qquad (5.2\text{-}2)$$

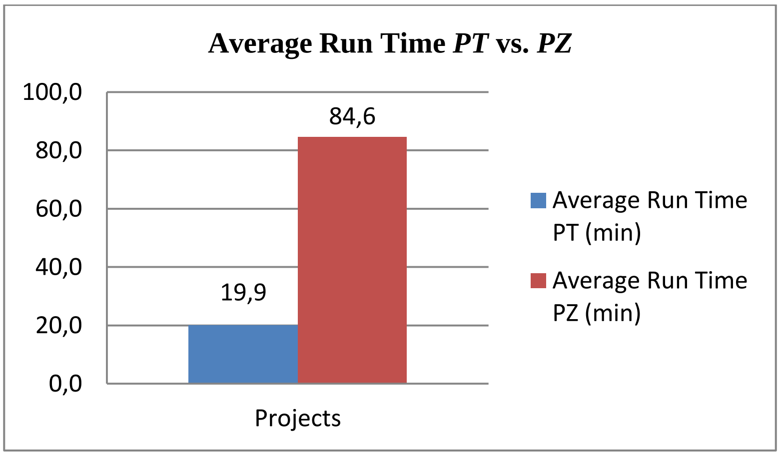



***Figure 5.3-Average Run Time comparison PT vs. PZ***

By inspecting Figure 5.3;

$$\boldsymbol{S} = \frac{Average\ Run\ Time_{PZ}\ \text{-}\ Average\ Run\ Time_{PT}}{Average\ Run\ Time_{PZ}}\ x100 = 76{,}5\% \qquad (5.2\text{-}3)$$

Here, $S$ is the percentage of the saved test run time.

By looking the value of $S$, it can be stated that proposed methodology decrease the test time in the order of **76,5 %.**

## 5.3 Test Development Time

Test development activity of proposed automatic testing approach includes;

i. Design and implement BSTS: Developer uses the assets which are built at *ATEMS* phase to construct related BSTS. This significantly decreases the development effort.

ii. Design and construct BSTE: Equipment uses *ATE's* standard connectors to connect the devices. Therefore, BSTE design is downgraded to preparation of cable mappings.

iii. Preparation of test procedure: As a consequence of using automatic test equipment, test procedure becomes very straightforward. Test procedure basically tells operator how to run the test and follow the warnings on the screen. (see **appendix A**)

iv. BSTS and BSTE integration: Complete test flow is applied to the "*known good board (i.e. golden board)"*several times to validate test methodology and be sure from functionality

v. Test set verification: Test procedure, *BSTS* and *BSTE* are verified by multiple runs and statistical repeatability calculations on the results. This activity is done by the participation of the company quality control engineers.

Unfortunately, stored test development time data set did not have above mentioned granularity. The data on hand shows the total amount of time to spent on each board and they were listed below at Table 5.5

***Table 5.5-PT members' test development time (hour)***

| **Board** | **Development Time (hour)** |
|---|---|
| m1 | 91,6 |
| m2 | 92,9 |
| m3 | 121,7 |
| m4 | 119,5 |
| m5 | 102,5 |
| m6 | 97,6 |
| m7 | 99,6 |
| m8 | 104,8 |
| m9 | 152,0 |
| m10 | 112,4 |
| ***Total Test Development Time*** $_{PT}$**=1094,4 hour** | |

By inspecting Table 5.5 ;

$$\textbf{\textit{Average Test Development Time}}_{PT} = \frac{Total\ Dev.Time_{\ PT}}{\#\,of\ \ PT's\ members} = 109{,}4\ \ hours \tag{5.3-1}$$

On the other hand manual test development activities include;

i. Design and construct *MBSTE*: Connectors to connect the required resources are decided and placed into the test equipment. Mapping of wirings from board to devices are prepared and the cable specifications are decided.

ii. Preparation of test procedure: Test procedures need to be prepared in detail as a consequence of being operator dependent test. This procedure will guide the operator through the whole test. At measurement steps it must tell the operator where the probes must be placed, what the expected values and limits are. At the steps where signals should be applied to the board, procedure should explain how to set devices to give required signal and where or when this signal must be applied. Also, the procedure must direct operator to write down the measurement which he/she read to the test log document. (see ***appendix B***)

iii. Procedure and *MBSTE* integration: Complete test flow is applied to the "*known good board (i.e. golden board)"* several times to validate test methodology and be sure from procedures' guidance. As a consequence of manual tests' time consuming nature, this step elapses the main portion of the development time.

iv. Test set verification: Test procedure and *MBSTE* are verified by multiple runs and statistical calculations on the results. This activity is done by the participation of the company quality control engineers.

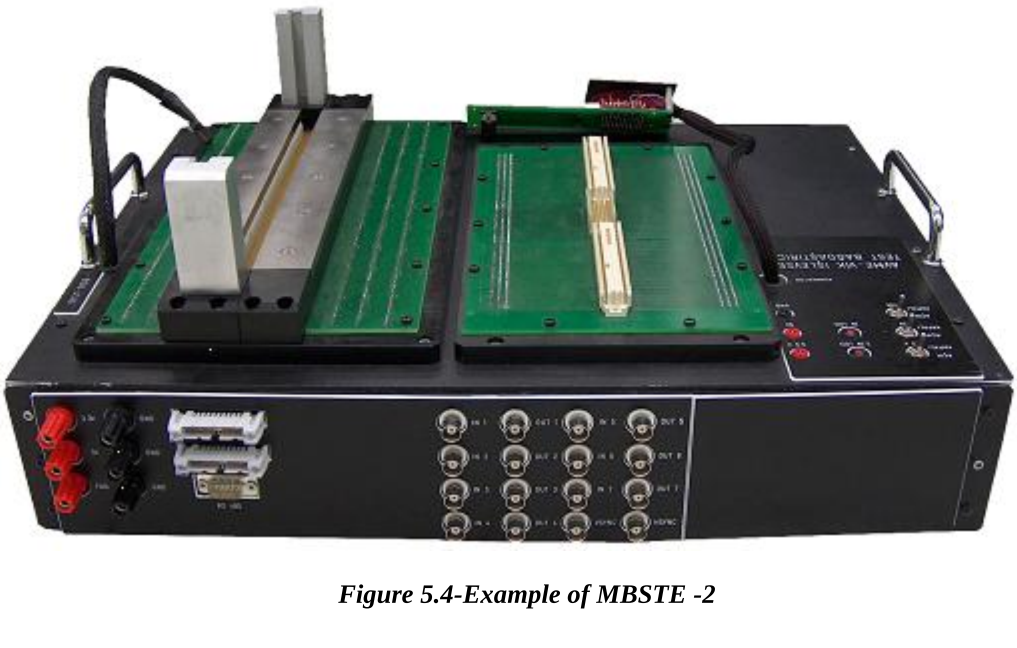


***Figure 5.4-Example of MBSTE -2***

Time spent for developing tests to the *PZ* family members were listed below.

***Table 5.6- PZ members' test development time (hour)***

| Board | Development Time (hour) |
|---|---|
| L1 | 158,7 |
| L2 | 182,3 |
| L3 | 135,6 |
| L4 | 185,5 |
| L5 | 148,7 |
| L6 | 142,5 |
| L7 | 196,4 |
| L8 | 138,6 |
| ***Total Test Development Time*** $_{PZ}$**=1288,4 hours** | |

By inspecting Table 5.6 ;

$$\boldsymbol{Average\ Test\ Development\ Time}_{PZ} = \frac{Total\ Dev.Time\ _{PZ}}{\#\,of\ \ PZ'\,s\ members} = 161{,}0\ \ hours \qquad (5.3\text{-}2)$$

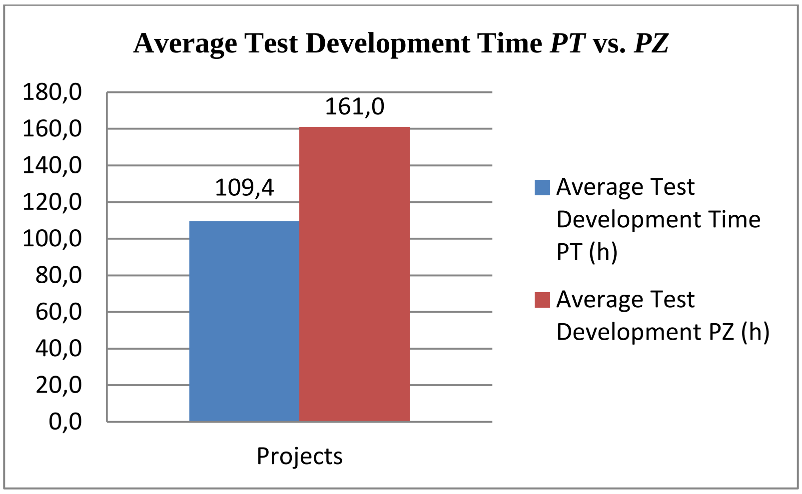



***Figure 5.5- Average Development Time comparison PT vs. PZ (hours)***

By inspecting Figure 5.5;

where *S* is the percentage of the saved test development time.

$$\boldsymbol{S} = \frac{Average\ Dev.Time_{PZ} - Average\ Dev.Time_{PT}}{Average\ Dev.Time_{PZ}} x100 = 32{,}0\% \qquad (5.3\text{-}3)$$

By looking the value of *S*, it can be stated that proposed methodology decreases the test development time in the order of **32,0 %.**

## 5.4 Test Hardware Cost

Critical decision of hardware cost calculations is whether to include resource cost in the test system cost or not.

In this study, the cost of devices in the *ATE* was not added to the total hardware cost of the board tests for *PT*. Also, in the same manner, the cost of used resources was not added to the total hardware cost of the *PZ.*

Devices in *ATE* and resources used in manual tests are general purpose equipment which means that will be used in more than one test design and family. Therefore, adding the cost of these general purpose devices to just one family, brings a misleading cost overhead to the test equipment.

Although the *ATE*, in this study, was developed for the *PT's line 2* family which has 10 members, today *it* has 23 board tests. If the cost of the *ATE* is shared by the boards tested by it, this 13 board should be considered. But adding data from other projects for evaluation purposes makes it impossible to compare *PT* and *PZ.* It is also applicable for *PZ* whose resources are used in different projects, too.

Therefore, in this study, only material costs of the test equipment and technician hour spent to build equipment was considered as a hardware cost. *(Technician hours will be converted to the USD by considering wages)*

Hardware costs of *PT*'s members were listed in Table 5.7.

***Table 5.7- Hardware Costs of PT's members ($)***

| **Board** | **Hardware Cost ($)** |
|---|---|
| m1 | 15750,0 |
| m2 | 14000,0 |
| m3 | 21000,0 |
| m4 | 19750,0 |
| m5 | 15800,0 |
| m6 | 17500,0 |
| m7 | 18750,0 |
| m8 | 16000,0 |
| m9 | 21750,0 |
| m10 | 16750,0 |
| ***Total Hardware Cost*** $_{PT}$**=177050,0$** | |

By inspecting Table 5.7;

$$\textbf{\textit{Average Hw Cost}}_{PT} = \frac{Total\ Hw\,Cost_{PT}}{\#\,of\ \ PT'\,s\ members} = 17705{,}0\$ \qquad (5.4\text{-}1)$$

Hardware costs of *PZ's* members were listed at Table 5.8

***Table 5.8- Hardware Costs of PZ's members ($)***

| **Board** | **Hardware Cost ($)** |
|---|---|
| L1 | 15900,0 |
| L2 | 18750,0 |
| L3 | 15250,0 |
| L4 | 22300,0 |
| L5 | 17800,0 |
| L6 | 19250,0 |
| L7 | 25750,0 |
| L8 | 14500,0 |
| ***Total Hardware Cost*** $_{PZ}$**=149500,0$** | |

By inspecting Table 5.8;

$$\textbf{\textit{Average Hw Cost}}_{PZ} = \frac{Total\ Hw\,Cost_{PZ}}{\#\,of\ \ PZ'\,s\ members} = 18687{,}5\$ \qquad (5.4\text{-}2)$$

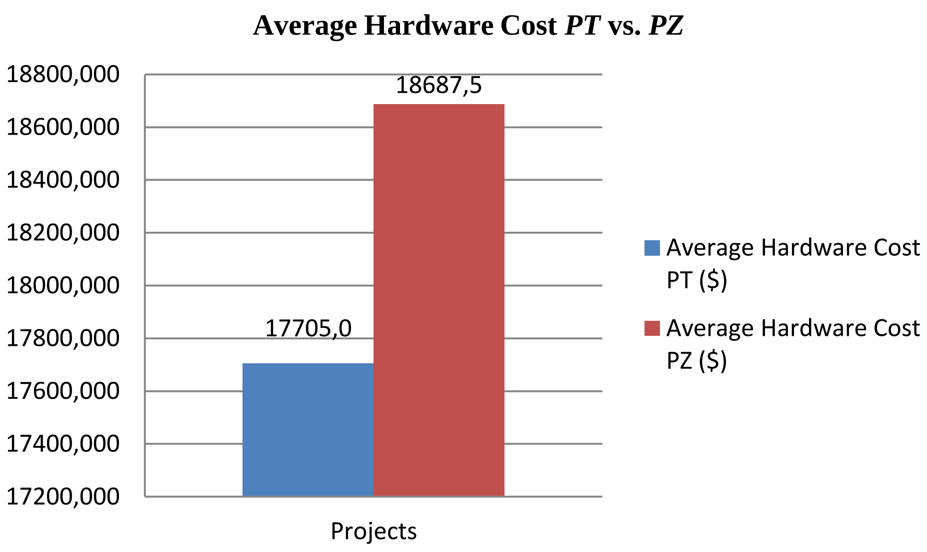



***Figure 5.6-Average Hardware cost comparison PT vs. PZ($)***

By inspecting Figure 5.6

where ***S*** is the percentage of the cost decrements.

$$\boldsymbol{S} = \frac{Average\,Hw\,Cost_{PZ} \text{ - } Average\,Hw\,Cost_{PT}}{Average\,Hw\,Cost_{PZ}} x100 = 5{,}3\% \qquad (5.4\text{-}3)$$

By looking at the value of *S*, it can be stated that the difference between hardware costs of proposed methodology and manual method is only **5,3 %** (~1000$).

## 5.5 Fault Detection Rate

Electronic boards are subjected to more than one level of tests. Board test is the first level examination which is applied to the boards. Integration and system acceptance tests are the next levels, respectively. ***(As seen in figure 2.2)***

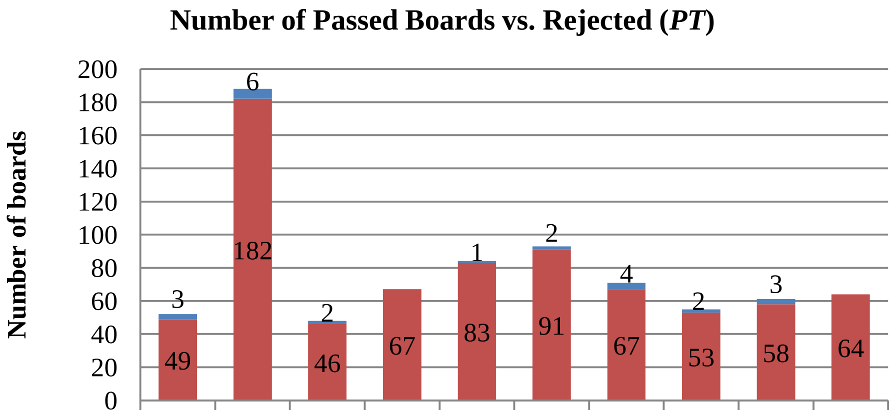
Fault detection rate metric shows the rate of boards which passed the board level examination and then rejected at either at integration or system acceptance levels.

$$\textbf{\textit{Fault Detection Rate}}_{x} = \frac{\text{Total Board Production - Rejected Boards}}{\text{Total Board Production}} x100 \qquad (5.5\text{-}1)$$

Number of produced and rejected boards of the *PT* can be seen in Table 5.9.

***Table 5.9- Production & Rejection Numbers of PT's boards***

| **Boards** | **# of Produced Boards** | **# of Rejected Boards** |
|---|---|---|
| m1 | 52 | 3 |
| m2 | 188 | 6 |
| m3 | 48 | 2 |
| m4 | 67 | 0 |
| m5 | 84 | 1 |
| m6 | 93 | 2 |
| m7 | 71 | 4 |
| m8 | 55 | 2 |
| m9 | 61 | 3 |
| m10 | 64 | 0 |
| ***Total Production*** $_{PT}$ **= 783 boards** | | |
| ***Total Rejection*** $_{PT}$ **= 23 boards** | | |




| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Rejected | 3 | 6 | 2 | 0 | 1 | 2 | 4 | 2 | 3 | 0 |
| Passed | 49 | 182 | 46 | 67 | 83 | 91 | 67 | 53 | 58 | 64 |

***Figure 5.7- Distribution of Passed and Rejected boards on PT's members***

By inspecting Table 5.9 ;

$$\textit{Fault Detection Rate}_{PT} = \frac{\text{Total Board Production - Rejected Boards}}{\text{Total Board Production}} x100 = 97{,}1\% \qquad (5.5\text{-}2)$$

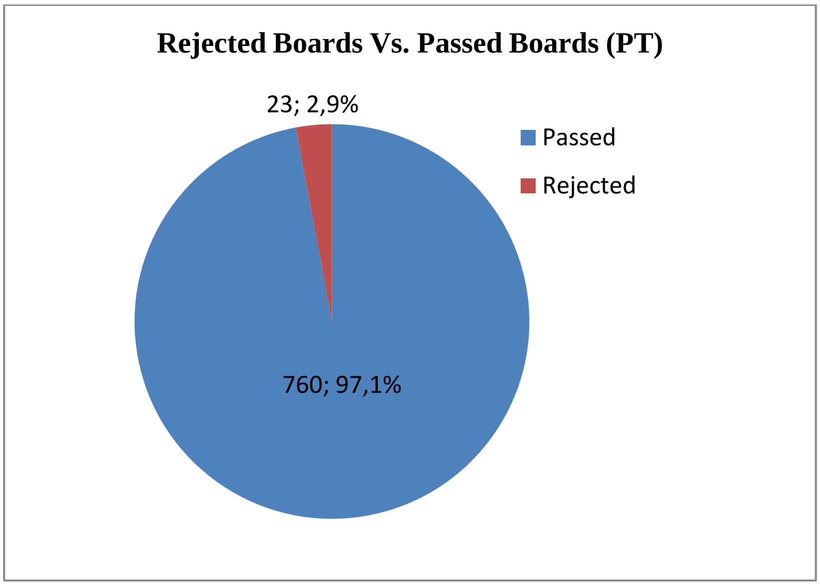



***Figure 5.8- Total number of Passed and Rejected boards in PT***

Number of produced and rejected boards of the ***PZ*** can be seen in Table 5.10.

***Table 5.10- Production & Rejection Numbers of PZ's boards***

| **Boards** | **# of Produced Boards** | **# of Rejected Boards** |
|---|---|---|
| L1 | 376 | 23 |
| L2 | 165 | 12 |
| L3 | 63 | 4 |
| L4 | 138 | 7 |
| L5 | 446 | 28 |
| L6 | 99 | 4 |
| L7 | 81 | 7 |
| L8 | 48 | 3 |
| ***Total Production*** $_{PZ}$ **= 1416 boards** | | |
| ***Total Rejection*** $_{PZ}$ **= 88 boards** | | |

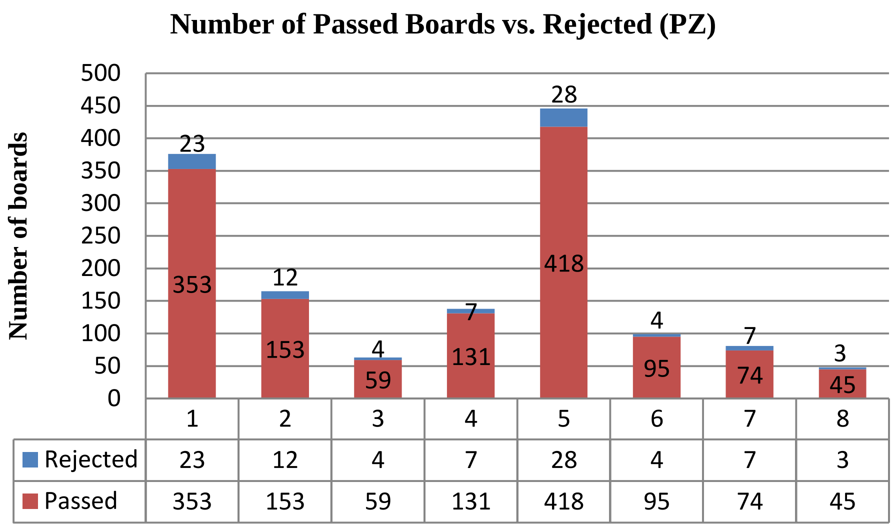



***Figure 5.9- Distribution of Passed and Rejected boards on PZ's members***

By inspecting Table 5.10 ;

$$\textbf{\textit{Fault Detection Rate}}_{PZ} = \frac{\text{Total Board Production - Rejected Boards}}{\text{Total Board Production}} x100 = 93,8\% \qquad (5.5\text{-}3)$$

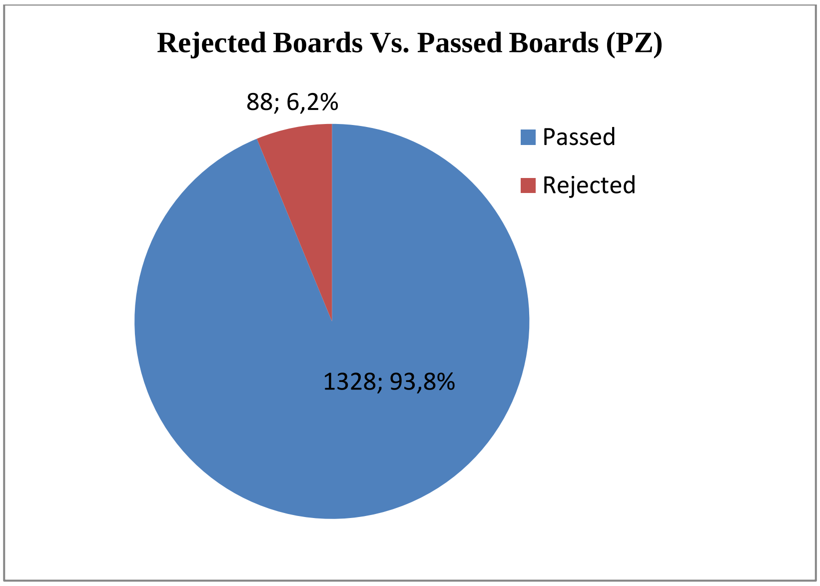



***Figure 5.10- Total number of Passed and Rejected boards in PZ***

The results at 5.5-2 and 5.5-3 shows that **2,9 *%*** of the ***PT's* boards** and **6,2 *%*** of ***PZ's* boards** were rejected by the high level tests. These rates showed that fault detection rate of proposed *ATE* method is better than manual testing methods.

When a board is rejected by a high level test, this means that one of below three possibilities has occurred.

- **Operator Fault:** Test operator done something wrong when applying the board test procedure in the first time. And this cannot be detected by the test system.

- **Insufficient Board Test:** Applied test methods are insufficient to detect and isolate error types that rejected boards have. When insufficiency is detected, test equipment and software are revised immediately by the engineers.

- **False Alarm:** Boards are in a good condition which means system level test gave false alarm.

To understand the reason of the rejection, one should apply the action plan which was outlined in Figure 5.11

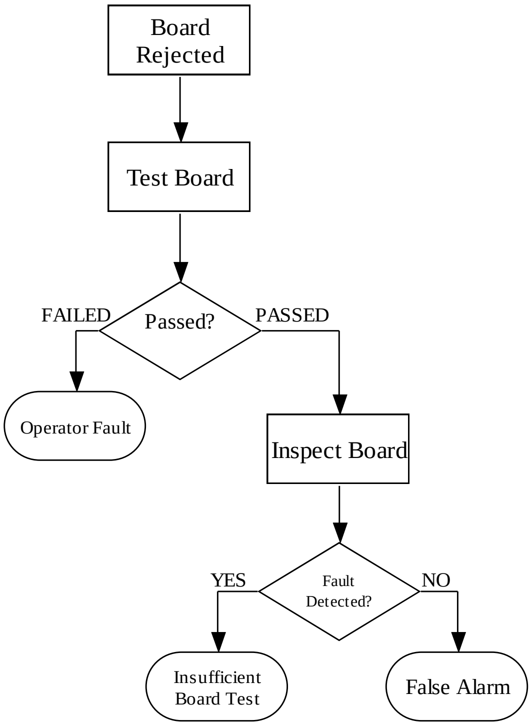



***Figure 5.11-Rejected Board Action Plan***

In this study, board tests were run again for the rejected boards of the *PT* and *PZ* by test operators. By this way percentage of the operator faults in overall rejections for both methodologies were calculated.

The results of this retest action for *PT*'s boards can be seen in Table 5.11

***Table 5.11- Distribution of Rejected Boards on Insufficient Test and Operator Faults in PT***

| **Boards** | **# of Produced Boards** | **# of Rejected Boards** | **# of Insufficient Tests** | **# of Operator Faults** |
|---|---|---|---|---|
| m1 | 52 | 3 | 2 | 1 |
| m2 | 188 | 6 | 3 | 3 |
| m3 | 48 | 2 | 0 | 2 |
| m4 | 67 | 0 | 0 | 0 |
| m5 | 84 | 1 | 0 | 1 |
| m6 | 93 | 2 | 2 | 0 |
| m7 | 71 | 4 | 3 | 1 |
| m8 | 55 | 2 | 2 | 0 |
| m9 | 61 | 3 | 2 | 1 |
| m10 | 64 | 0 | 0 | 0 |

By inspecting Table 5.11 ;

***Total Number of Rejected Boards***$_{PT}$ =23 (5.5-4)

***Total Number of Insufficient Test***$_{PT}$ =14 (5.5-5)

***Total Number of Operator Faults***$_{PT}$ =9 (5.5-6)

$$\textbf{\textit{Percentage of Operator Faults}}_{PT} = \frac{\text{Total Operator Faults}}{\text{Total Rejected Boards}} x100 = 39{,}1\% \tag{5.5-7}$$

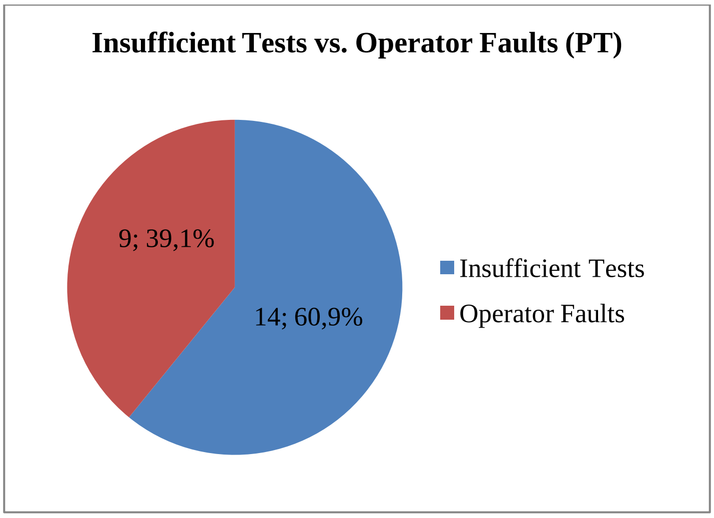



***Figure 5.12- Insufficient Tests vs. Operator Faults in PT***

The results of this retest action for *PZ*'s boards can be seen in Table 5.12.

***Table 5.12-Distribution of Rejected Boards on Insufficient Test and Operator Faults in PZ***

| Boards | # of Produced Boards | # of Rejected Boards | # of Insufficient Tests | # of Operator Faults |
|---|---|---|---|---|
| L1 | 376 | 23 | 3 | 20 |
| L2 | 165 | 12 | 2 | 10 |
| L3 | 63 | 4 | 0 | 4 |
| L4 | 138 | 7 | 2 | 5 |
| L5 | 446 | 28 | 4 | 24 |
| L6 | 99 | 4 | 0 | 4 |
| L7 | 81 | 7 | 3 | 4 |
| L8 | 48 | 3 | 3 | 0 |

By inspecting Table 5.12;

***Total Number of Rejected Boards***$_{PZ}$ =88 (5.5-8)

***Total Number of Insufficient Test***$_{PZ}$ =17 (5.5-9)

***Total Number of Operator Faults***$_{PZ}$ =71 (5.5-10)

$$\textbf{\textit{Percentage of Operator Faults}}_{PZ} = \frac{\text{Total Operator Faults}}{\text{Total Rejected Boards}} x100 = 80{,}7\% \quad (5.5\text{-}11)$$

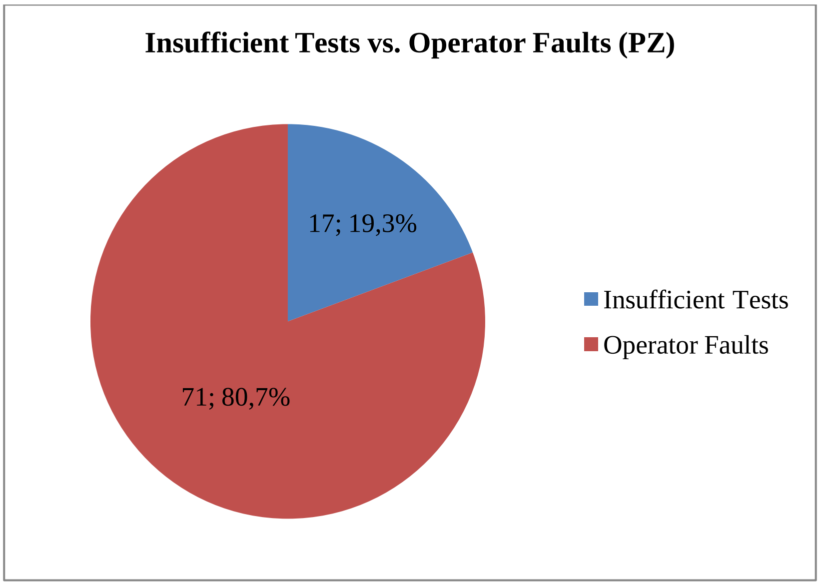



***Figure 5.13-Insufficient Tests vs. Operator Faults in PZ***

From the above analysis it can be said that the proposed solution not only increases the fault detection rate, from **93,8%** to **97,1%,** but also significantly decreases the operator related faults, from **80,7%** to **39,1 %.** (*Formulas* ***5.5-3, 5.5-2, 5.5-11, 5.5-7*** *respectively*)

# CHAPTER 6

# CONCLUSION

In this thesis study, benefits of modular automatic board test system design were investigated and contributions were pointed out by using metrics.

Domain engineering principles and common design steps in the literature were investigated. Current board test methodologies were listed by using several sources and by using this information the border line of the thesis work had been drawn.

In order to fairly evaluate the proposed methodology, metrics that have been accepted by the industry and published in the literature had been investigated. These metrics were explained in detail in section 2.3. Then, the proposed methodology was applied to a specific board family and a proper test system was implemented. The performance and efficiency evaluations of this system were done by comparing the results of the above mentioned metrics with a specific manual test system's results.

In this study, one of ASELSAN's projects which has been carried out for the Turkish Air Force was investigated. The focused board family was chosen as the group which handles communication, processing and digital input/output duties of the system. As proposed in chapter 0, general purpose test equipment called *ATE* and board specific test equipment called *BSTE* were designed in this study. In addition to this hardware, general management software (*ATEMS*) and software that applies the test cases (*BSTS*) to the board were designed and implemented by the ASELSAN test design department where the author is a member.

The data set that was used to evaluate the proposed system in chapter 0 was obtained from the usage of the systems at ASELSAN's MGEO factory by the author for this study.

In chapter 0, the proposed solution was compared with a manual test system which does not use the domain engineering principles.

In chapter 0 it was shown that

- Proposed test system decreases the average spent time to make test system ready by **56.4%.** (***formula 5.1-3***);
- Automatization of test procedures and methods decrease the average test run time by **76.5%** ( ***formula 5.2-3***);
- Usage of device management libraries and decreases within recurring engineering duties resulted in a dramatic reduction of the average test development time. The reduction is in the order of **32,0%** ( ***formula 5.3-3***);

- The great amount of initial hardware investments do not increase the hardware cost of the board tests. In contrast, decreased technician effort resulted in a reduction of the total cost. The reduction is in the order of **5,3%** ( ***formula 5.4-3***);

- The investigations done on the rejected boards at high level tests (i.e. integration tests and system acceptance tests) have shown that board rejection rate of the old method was **6,2%** whereas the proposed method has achieved a **2,9%** rejection rate ( ***formula 5.5-2 and 5.5-3***);

- The investigations done on the faulty boards have shown that the ratio of operator related errors in overall errors is **80,7 %** in the manual test systems, whereas this ratio is **39,1**% with the proposed solution ( ***formula 5.5-7 and 5.5-11***).

The test system was designed to test the *PT family boards,* the number of the boards that belong to the family was given as 10 in chapter 0. Presently in ASELSAN, there are 23 different board tests which were designed and ready to be used with the implemented test system. The increase of the number of boards from 10 to 23 clearly shows that the implemented test system fullfills the requirements of the new family members.

As stated above, the proposed solution reduced operator related errors to 39,1% . Although it is an improvement in terms of the test processes, future studies should aim to further reduce this ratio by increasing the level of automation of test procedures. In this manner, the possibility of applying image processing techniques to the board test equipment to automate the tests which require operators' observations, should be investigated in future studies.

To have a more precise analysis and gain more insight about process, companies should define the test development and application steps in detail. After establishing these concrete definitions, organizations should start to record elapsed time information for each step more accurately. In ASELSAN's case which is used in this study, the "*test development time"* data set contains the time spent at the validation/verification, test software development, test hardware design, test resource allocation steps. These steps are very different from each other and have different internal dynamics. Therefore, this data set gives a general idea about development actions but it can not be used to make deep analysis on the desired process.

This general test equipment was designed bearing all test requirements of the family in mind. Therefore, while testing a board which requires just a few resources, remaining devices on the system stay untouched. With increasing demand for this test system, it became inevitable to find a way to use system efficiently. In this manner, designing a test software algorithm and apparatus for the test equipment which will allow the test of more than one board at a time, should be considered as a future area of work in this subject.

# REFERENCES


[1] IEEE Standards Board, “IEEE Standard Glossary of Software Engineering Terminology”, 1990.

[2] McGregor, D. J., “Testing a software Product line”, Carnegie Mellon Software Engineering Institute Technical Report, 2001.

[3] Clements, P., Northrop, L., “Software Product Lines: Practices and Patterns”, Addison-Wesley, 2005.

[4] Cusumano, M. A.,”Japan’s Software Factories”, Oxford: Oxford University Press,1991.

[5] Greenfield, J., Short, K., “Software factories: assembling applications with patterns, models, frameworks, and tools “, Willey Publishing, 2004.

[6] Brooks,F., “No silver Bullet: Essence and Accidents of Software Engineering”, Computer Magazine,pp. 10-18, 1987

[7] Tampico,J., Rose,B.,H., “Minimizing Production Costs Through Modular Automatic Test Equipment”, IRE Transactions on Industrial Electronics,pp. 19-24, April 1959.

[8] Nguyen, H., D., Miller,I.,A., “Cost Impact of Automated Acceptance Testing of Electrical Ground Support Equipment for Spacecraft Testing”, IEEE AutoTestCon 2011

[9]Wegner, P. “Research Directions in Software Technology”, Proceedings of the 3rd International Conference on Software Engineering, pp. 243-259, 1978

[10] Parnas, L., D.,” On the Design and Development of Program Families”, IEEE Transactions on Software Engineering, pp. 1-9, March 1976.

[11] Schlotzhauer, O., E., “The Real World Board Test Effectiveness”, IEEE Design & Test of Computers, pp. 16-23, 1988

[12] Hults, M., C., Schwedner, A.,F., Grossman, E.,S., “In-Circuit Test Systems-An Evolution” , IEEE Transaction on Manufacturing Technology , pp. 42-48, December 1975.

[13] Shim, H., J., Cho, S. H., Kim, S., ” A New Probing System for the In- Circuit Test of a PCB”, IEEE International Conference on Robotics and Automation, pp. 590-595, April 1996.

[14] IEEE Standards Board, “IEEE Standard Test Access Port and Boundary-Scan Architecture”, 1990.

[15] Albee, A., “A Practical Guide to Combining ICT & Boundary Scan Testing”, IEEE International Test Conference, pp. 487-494, 2001

[16] Joel, D., “Factors Influencing the Acquisition of Low Cost Deployable Automatic Test

Equipment”, IEEE AUTOTESTCON, pp. 227-235, 1995

[17] Barnett, N., “In-Service Reliability, Maintainability and Testability Demonstrations”, Proceedings Annual Reliability and Maintainability Symposium, pp. 587-592, 2003

[18] Dorf, C., R., Hall , T., “The Benefits of Using ATE System in The Testing Of Printed Circuit Boards”, IEEE AUTOTESTCON, pp. 25-29, 1988

[19] Eriksson,M. , “An introduction to Software Product Line Development”, Proceedings of Ume's Seventh Student Conference in Computing Science, pp. 26-37, 2003

[20] SEI,” A Framework for Software Product Line Practice”,

http://www.sei.cmu.edu/productlines/frame_report/ , last visited: 06 January 2013

[21] Zhang, S., Choi, M., Park, N., Lombardi, F., “Probabilistic Balancing of Fault Coverage and Test Cost in Combined Built-In Self-Test/Automated Test Equipment Testing Environment”, IEEE DFT, 2004

[22] Williams, J., B., “What Additional Standards and Methods Are Needed In Non-Destructive Testing”, Symposium on Standardization of in Non-Destructive Testing, pp. 84-87, 1962

[23]Shim, J., Kim, S., Cho, H., “A New Probing System for the In- Circuit Test of a PCB”, Proceedings of IEEE ICRA, 1996

[24] Dyer, S., “Survey of instrumentation and measurement”, Wiley, pp. 25-33, 2001

[25] Brogan, L., “Reducing Costs Through The Use Of Special Purpose Automatic Test Equipment (SPATE)”, IEEE AUTOTESTCON, 1994

[26] Gosling,W. , “Twenty Years of ATE”, IEEE ITC, pp.3-6, 1989

[27] Carleton, D., “B-2 Automatic Test Equipment”, IEEE NAECON, pp. 1003-1009, 1993

[28] Mingfei, X., Hongfeng, W., Mingqi, X., “Automatic Test System for Some Kind General Electronic Equipment”, IEEE ICEMI, pp.73-75, 2011

[29] Hernan, J., Manrique, F., “New Generation of Automatic Test Bench (STTES) For Eurofighter Typhoon's Avionic Units”, IEEE AUTOTESTCON, pp.261-266, 2006

[30] Verma, A., Robinson, C., Butkovich, S., “Production Test Effectiveness of Combined Automated Inspection and ICT Test Strategies”, IEEE ITC, pp. 393-402, 2004

[31] Scheck, G., “Reducing the Cost of ATE Software Development”, IEEE AUTOTESTCON, 2012

[32] Kapur, R., Chandramouli, R., Williams, T., “Strategies for Low-Cost Test”, IEEE Design & Test of Computers, pp.47-54, 2001

[33] Crosher, R., “The Economics of Operator Participation in Automatic Testing” , IEEE Radio and Electronic Engineer, pp. 35-42, 1971

[34] Fan, D., Roehling, S., Carruth, R., “Case Study - Using STIL as Test Pattern Language”, IEEE ITC, pp.811-817, 2003

[35] Gillette, G., “A Single Board Test System: Changing the Test Paradigm”, IEEE ITC, pp. 880-885, 1995

[36] Aranha, E., Borba, P., “ManualTEST: Improving Collection of Manual Test Execution Data in Empirical Studies”, Empirical Software Engineering Latin-American Workshop (ESELAW), 2008

[37] Elmore, T., “Case Study: Modular Ate System Design”, IEEE AUTOTESTCON, pp. 576-583, 2000

# APPENDIX A

## EXAMPLE OF AN AUTOMATIC TEST PROCEDURE

**1. AMAÇ**

1.1. Bu doküman XXXXXX projesi kapsamında yer alan XXXX-XXX parça numaralı “XXXXX” kartına uygulanan üretim kabul testi işlemlerini ayrıntılı bir şekilde anlatır ve uygulanmasını sağlar.

**2. UYARILAR**

2.1. Bu doküman yalnızca XXXX-XXX parça numaralı “Navigation I/O” kartına uygulanabilir.

2.2. Ortam sıcaklığının 24±5 C, bağıl nemin %25-60 olması gerekmektedir.

2.3. **Bu dökümanda, aksi belirtilmedikçe, atıfta bulunulan tüm yazılım, donanım ve dokümanların son sürümleri esas alınır.**

**3. İLGİLİ DOKÜMANLAR**

3.1. Bu birim ve test ile ilgili olarak ilişik belirtilen ve başvurulabilecek dokümanlar aşağıdaki **Tablo-1’**de belirtilmiştir.

Tablo 1 Referans Doküman ve Formlar

| **Doküman Türü** | **Doküman Numarası** | **Dil Kodu** | **Doküman Tanımı** |
|---|---|---|---|
| 8888 | XXXXX | 000 | TBDK |
| 888 | XXXXX | 000 | ŞEMATİK |

**4. TEST DONATIMI LİSTESİ**

4.1. Aşağıdaki **Tablo-2**’de verilen donatımın varlığını, ölçümleme (kalibrasyon) tarihlerinin geçerli olduğunu ve son versiyon olduklarını doğrulayınız.

Tablo 2 Test Donatımı Listesi

| **DONATIM** | **P/N** | **Firma** | **Kalibrasyon** |
|---|---|---|---|
| XXXXX Otomatik Test Cihazı | T-XXX | ASELSAN | Gerektirir |
| Genel Kullanım Yazılımı, XXX | TS-XXXX | ASELSAN | Gerektirmez |
| XXXX Kartı Test Yazılımı | TS-000XX | ASELSAN | Gerektirmez |
| XXXXX Kartı Test Ekipmanı | TE-00XX | ASELSAN | Gerektirmez |
| BS Test Kartı | - | AAAAAA | Gerektirmez |

**5.TEST YÖNERGESİ**

**5.1.** Test sistemi açık değil ise yeşil sistem açma düğmesi ile sistemi açtıktan sonra sistem üzerindeki tüm cihazları da açınız.

**DİKKAT!**

Sistem açma düğmesine basıldığında bağdaştırıcı yatağı herhangi bir uyarı verilmeksizin kenetli ise açılacak, açık ise kenetlenecek şekilde hareket edebilmektedir. Bu nedenle üzerinde ya da yakınında bir şey bulunmamasına dikkat ediniz.

**5.2.** Test bilgisayarını çalıştırınız. "XXXX" kullanıcı adı ve "aaaa" şifresini kullanarak XXXNET ortamına giriş yapınız.

**5.3.** Test bilgisayarının masaüstünde yer alan "**Test Exec**" kısa yoluna çift tıklayarak test arayüz programını çalıştırınız. Masaüstünde bu kısa yol mevcut değil ise ***Start/Programs/National Instruments/Teststand 3.5/Operator Interfaces/Labwindows-CVI*** menü yolu ile de program çalıştırılabilir. Size özel verilen kullanıcı adı ve şifreyi giriniz.

**5.4.** Test edilecek kartı bağdaştırıcı üzerine dikkatlice yerleştiriniz. Test düzeneğinde yer alan "JTAG" konnektörünü test edilen kartın "J1" konnektörüne takınız. Test düzeneğinde yer alan "BSIO-JTAG" konnektörünü BSIO kartın "J6" konnektörüne takınız.

**5.5.** Pencerenin sağ-üst tarafında yer alan "**Single Pass**" düğmesine tıklayarak testi başlatınız.

**DİKKAT!**

Devam eden adımlarda, yazılım tarafından görüntülenecek mesajlarda kullanıcının seçimi doğrultusunda bağdaştırıcı yatağı -sesli bir uyarı verildikten sonra- hareket edebilir.

**5.6.** Yazılım, sistemdeki cihazların varlığını kontrol ettikten ve cihazları ilklendirdikten sonra bağdaştırıcı denetlemesi yapacaktır. Sistemde, kenetli durumda bir bağdaştırıcı yok ise yazılım bir bağdaştırıcı takmanız konusunda uyaracaktır. Bu işlem için 5.7 adımına geçiniz. Kenetli durumda bir bağdaştırıcı var ise yazılım mevcut bağdaştırıcı ile devam etme ya da bağdaştırıcıyı değiştirme seçeneklerini görüntüler. Mevcut bağdaştırıcı **TE-00XXX** ise 5.8 adımına geçiniz. Aksi takdirde DEĞİŞTİR düğmesine tıklayınız.

**5.7.** **TE-00XXX** numaralı test bağdaştırıcısını, etiketi kendinizden tarafa gelecek şekilde ve test sistemi konektörüyle bağdaştırıcı konektörü uyumlu olacak şekilde yerleştiriniz. Sistem bağdaştırıcı yatağındaki sabitleyici çıkıntıların bağdaştırıcı konektörünün alt tarafındaki deliklere girmesini sağlayınız ve ardından kilitleme kolunu kullanarak bağdaştırıcıyı kilitleyiniz. Bu işlem sonunda bağdaştırıcının yatağa düzgün ve sağlamca yerleştiğinden emin olunuz.

**DİKKAT!**

Bağdaştırıcı yerleşimindeki herhangi bir uyumsuzluk hem test sistemine hem de bağdaştırıcıya onarılması güç zararlar verebilir. Bu nedenle yerleştirme işlemi dikkatle ve özenle yapılmalı, ehemmiyeti göz ardı edilmemelidir.

Bu bağlamda hasar görmüş, farklılaşmış ya da uyumsuz görünen bağdaştırıcı, sistem yatağı, konektörler, bağlantı elemanları vs. var ise sistemi kullanmayıp ilgililerine haber veriniz.

**5.8.** Bağdaştırıcı sistem konektörü ile kenetlendikten sonra yazılım tarafından tanınacaktır. Bu aşamada ekranda görüntülenecek mesajları dikkatle takip ediniz. Son olarak, eğer henüz test edilecek birimin seri numarası girilmemiş ise seri numarası girmeniz istenecektir. Seri numarasının doğru girilmesi test sonuçlarının sabit diske doğru kaydedilmesi ve ilerde tekrar ulaşılabilmesi açısından önemlidir.

**5.9.** Test adımları çalışmaya başladıktan sonra ekranda teste özel mesajlar görüntülenebilir. Bu durumlarda yazılı olarak belirtilen uyarı ve komutlara uyarak testi tamamlayınız.

**5.10.** Test bitiminde kısa test çıktısı sorgusu ekrana gelecek ve onayladığınızda tek sayfa çıktı basılacaktır. Arayüz penceresinin üst tarafındaki "Report" sekmesinde ayrıntılı test çıktısı görüntülenecektir. Bu uzun test çıktısını almak için sekme üst kısmında yer alan "Print" düğmesini kullanabilirsiniz.

**5.11.** Yeni bir birim test etmek için önce menüden "File/Close Execution" seçeneğine tıklayınız. Ardından pencerenin sol tarafındaki "Sequence Files" sekmesine tıkladıktan sonra 5.4 adımına dönünüz.

**5.12.** Farklı bir kullanıcı testlere devam edecek ise menüden "File/Logout" seçeneğine tıklayınız. Kullanıcı adı ve şifre sorgulama ekranı görüntülenecektir. Bu bilgileri girdikten sonra 5.4 adımına dönünüz.

**5.13.** Teste devam etmeyecek iseniz programı kapatmak için menüden "File/Exit" seçeneğine tıklayınız. Sistemi tamamen kapatmak için bilgisayarı kapattıktan sonra sistem üzerindeki tüm birimleri kapatıp kırmızı sistem kapatma düğmesine basınız.

**DİKKAT!**

Sistem açma ve kapama esnasında bağdaştırıcı yatağı herhangi bir uyarı verilmeksizin kenetli ise açılacak, açık ise kenetlenecek şekilde hareket edebilmektedir. Bu nedenle üzerinde ya da yakınında bir şey bulunmamasına dikkat ediniz.

# APPENDIX B

## EXAMPLE OF A MANUAL TEST PROCEDURE

1. **AMAÇ**
   - **1.1.** Bu doküman **XXXXX** parça numaralı **"TBDK, XXXXXX"** biriminin fonksiyonel test yöntemini, ayrıntılı bir şekilde anlatır.
2. **UYARILAR**
   - **2.1.** Bu doküman yalnızca **XXXX** parça numaralı **"TBDK, XXXXX"** birimine uygulanabilir.
   - **2.2.** Test edilecek karta dokunmadan önce mutlaka ESD bileziğini takınız. Kartta bulunan elemanlar durgun yüke hassas elemanlardır.
   - **2.3.** Ortam sıcaklığının 25±7 C, bağıl nemin %25-60 olması gerekmektedir.
   - **2.4.** Sonunda (**KAYIT**) uyarısı olan paragraflarda alınan tüm ölçümler **FORM-0XXX** test veri sayfasına işlenmelidir.
   - **2.5.** **XXXX Kartı güç kaynaklarından 8.00 A değerine varan akım çekmektedir. Bu nedenle test esnasında kullandığınız malzemelerin yalıtımına ve test yönergesini eksiksiz uygulamaya özen gösteriniz.**

3. **REFERANS DOKÜMANLAR VE FORMLAR**
   - **3.1.** Referans dokümanlar ve formlar tabloda belirtilmiştir.

| **Referans Doküman ve Formlar** | | | |
|---|---|---|---|
| **Doküman Türü** | **Doküman Numarası** | **Dil Kodu** | **Doküman Tanımı** |
| 888 | XXXXX | 000 | TBDK, XXX |
| 888 | XXXXX | 000 | ŞEMATİK, XXXX |
| 888 | XXXX | 000 | TEST GEREKLİLİKLERİ, XXXX |
| 888 | FORM-0XXX | 000 | XXXX Kartı Fonksiyonel Test Veri Formu |
| 888 | TS-000XXX | 000 | XXX Kartı İşlevsel Test Yazılımı |

Tablo 3: Referans Dokümanlar ve Formlar

**4.TEST DONATIMI LİSTESİ**
   - **4.1.** Aşağıdaki tabloda verilen donatımların varlığını, ayarlama tarihlerinin geçerli olduğunu ve sürüm numaralarını doğrulayınız.

| **Donatım** | **Model** | **Firma** | **Ayarlama** |
|---|---|---|---|
| GÖZCÜ-V1 Güç Kartı İşlevsel Test Bağdaştırıcısı | TE-00XX | ASELSAN | Gerektirmez. |
| Güç Kaynağı ( 32V-10 A )( veya eşdeğeri ) | HHHH | AAAAA | Gerektirir. |
| Güç Kaynağı ( 32V-10 A )( veya eşdeğeri ) | HHHH | AAAAA | Gerektirir. |
| Güç Kaynağı ( 30V-3 A )( veya eşdeğeri ) | HHHH | AAAAA | Gerektirir. |
| Güç Kaynağı ( 30V-3 A )( veya eşdeğeri ) | HHHH | AAAAA | Gerektirir. |
| Güç Kaynağı ( 30V-3 A )( veya eşdeğeri ) | HHHH | AAAAA | Gerektirir. |
| Sayısal Multimetre ( veya eşdeğeri ) | HHHH | AAAAA | Gerektirir. |

*Tablo 4: Test Donatımı*

**5.TEST YÖNERGESİ**

**5.1.** Bu yönergede **“KAPLAMA ÖNCESİ TEST”** ve **“KAPLAMA SONRASI TEST”** adımları aynı adımlardır.

**NOT: “KAPLAMA SONRASI TEST”** durumunda kaplamaya zarar vermemek için kart üzerinden ölçüm alınması gereken noktalardan ölçüm almayınız. Test veri formunda bu testleri **“GEÇTİ”** olarak kabul ediniz. Test veri formuna **“Test Tipi”** bölümüne ilgili test tipini işaretleyiniz.

**NOT: GÖZCÜ-V1_5 Güç Kartı güç kaynaklarından 8.00 A değerine varan akım çekmektedir. Bu nedenle test esnasında kullandığınız malzemelerin yalıtımına ve test yönergesini eksiksiz uygulamaya özen gösteriniz.**

**5.2.** Test kutusu üzerindeki **“KART_KAPALI”** ve **“KART BEKLEMEDE”** anahtarlarının **“AÇIK”** diğer anahtarların **”KAPALI”** konumda olduğunu kontrol ediniz.

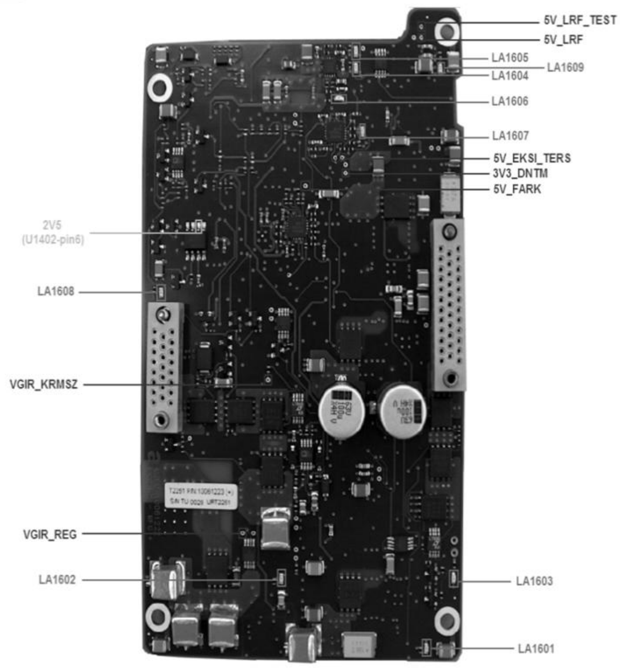
**5.3.** Güç giriş bağlantılarını test bağdaştırıcısının arkasındaki güç giriş noktalarından belirtildiği şekilde yapınız, güç girişlerini tablodaki değerlere göre ayarlayınız. **VHARICI** girişine 10 A akım, **VPIL** girişine 3 A akım sağlayabilen güç kaynağını bağlayarak tabloda belirtilen güç girişi bağlantılarını yapınız, gerilim ve akım değerlerini tabloda belirtilen değerlere ayarlayınız.

| **Güç Girişi Adı** | **Güç Girişi Gerilim Değeri** | **Akım Limiti** |
|---|---|---|
| VPIL | 8.00 ± 0.05 V | 2.00 A |
| VHARICI | 12.00 ± 0.05 V | 2.00 A |
| 12V_SOG_TEST | 12.00 ± 0.05 V | 100.00 mA |
| 12V_TEST | 12.00 ± 0.05 V | 100.00 mA |
| 5V_TEST | 5.00 ± 0.05 V | 100.00 mA |
| 5V_EKSI_TEST | -5.00 ± 0.05 V | 100.00 mA |
| 3V3_TEST | 3.30 ± 0.05 V | 100.00 mA |

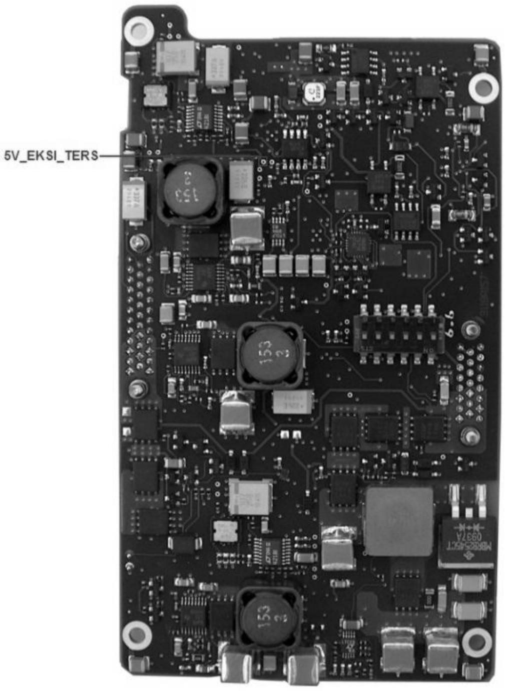
**5.4.** **“TBDK, XXXX ”** birimini test kutusu üzerindeki yapıya **Resim-1** ‘de görüldüğü gibi dikkatlice yerleştiriniz. **KN1** ve **KN2** konektör bağlantılarını üzerlerinde belirtilen numaralara dikkat ederek yapınız.

***Resim-1 – Test Kutusu Kart Yerleşimi***

**5.5.** Test sırasında gözlemlenmesi ya da ölçüm alınması istenen bazı noktalar kart üzerinde bulunmaktadır. Bu noktalardan ölçüm almak için **“Resim-2”** ve **“Resim3”** üzerinde gösterilen test noktalarını kullanınız.




***Resim-2 – Test Noktalarının Kart Üzerindeki Gösterimleri ( Ön Yüz )***




***Resim-3 – Test Noktalarının Kart Üzerindeki Gösterimleri ( Arka Yüz )***

### 5.6. AÇIK DEVRE TESTLERİ

- Tablodaki noktalardan direnç ölçümü yapınız. **KAYIT.**

| Ölçüm Noktası "+" Kutbu | Ölçüm Noktası "-" Kutbu | Alt Limit | Üst Limit |
|---|---|---|---|
| VHARICI | VGIR_RTN | 1.0 kΩ | - |
| VPIL | VGIR_RTN | 1.0 kΩ | - |
| VHARICI | VPIL | 1.0 kΩ | - |
| VGIR_KRMSZ *(Resim-2)* | VGIR_RTN | 1.0 kΩ | - |
| VGIR | VGIR_RTN | 1.0 kΩ | - |
| VGIR_REG *(Resim-2)* | VGIR_RTN | 1.0 kΩ | - |
| KART_KAPALI | VGIR_RTN | 1.0 kΩ | - |
| 12V_SOG | GND | 1.0 kΩ | - |
| 12V | GND | 1.0 kΩ | - |
| 5V | GND | 75 Ω | 90 Ω |
| 5V_EKSI | GND | 0.5 kΩ | - |
| 3V3 | GND | 40 Ω | 60 Ω |
| 3V3_DNTM *(Resim-2)* | GND | 1.0 kΩ | - |
| 5V_LRF | GND | 1.0 kΩ | - |
| KART_BEKLEMEDE | GND | 1.0 kΩ | - |

### 5.7. KISA DEVRE TESTLERİ

- Tablodaki noktalardan direnç ölçümü yapınız. **KAYIT.**

| Ölçüm Noktası "+" Kutbu | Ölçüm Noktası "-" Kutbu | Ölçüm Limiti (ohm) |
|---|---|---|
| GND | VGIR_RTN | ≤ 2.0 Ω |
| 12V_SOG | 12V_SOG _TEST | ≤ 2.0 Ω |
| 12V | 12V_TEST | ≤ 2.0 Ω |
| 3V3 | 3V3_TEST | ≤ 2.0 Ω |
| 5V | 5V_TEST | ≤ 2.0 Ω |
| 5V_EKSI | 5V_EKSI_TEST | ≤ 2.0 Ω |

### 5.8. GÜÇ GİRİŞİ TESTLERİ

#### 5.8.1. "VPIL" Girişinden Çekilen Akım Kontrolü:

- **"KART_KAPALI"** anahtarını **"KAPALI"** konumuna getiriniz.
- **"KART BEKLEMEDE"** anahtarını **"BEKLEMEDE"** konumuna getiriniz.
- **"VPIL"** anahtarını **"AÇIK"** konumuna getiriniz.
- Güç kaynağından çekilen akım değerini ölçünüz. **KAYIT**.

| Güç Girişi Adı | Alt Limit | Üst Limit |
|---|---|---|
| VPIL | 0.00 mA | 20.00 mA |

- **“VPIL”** anahtarını **“KAPALI”** konumuna getiriniz.

### 5.8.2. “VHARICI” Girişinden Çekilen Akım Kontrolü:

- **“VHARICI”** anahtarını **“AÇIK”** konumuna getiriniz.
- Güç kaynağından çekilen akım değerini ölçünüz. **KAYIT.**

| Güç Girişi Adı | Alt Limit | Üst Limit |
|---|---|---|
| VHARICI | 0.00 mA | 20.00 mA |

- **“VHARICI”** anahtarını **“KAPALI”** konumuna getiriniz.

## 5.9. SICAKLIK ALGILAYICILARI TESTİ

**NOT:** Sıcaklık algılayıcıları testinde ölçülen sıcaklık değeri test kutusunun ön yüzünde bulunan ekranda sayı olarak yazacaktır. ***Resim-4*** ‘e göre U1501 algılayıcısından ölçülen değer 26 ºC, U1503 algılayıcısından ölçülen değer ise 28 ºC ‘dir. Yani sağdaki iki basamaklı rakam ilk ölçümü, diğer 2 basamaklı rakam ise ikinci ölçüm değerini göstermektedir.


***Resim-4 – Sıcaklık Algılayıcı Bilgisi***

Ayrıca iki adet uyarı kodu vardır.

- **E1E1:** Hat üzerinde bulunan iki algılayıcı ile de haberleşme yapılamadığı anlamına gelmektedir.
- **E2:** Algılayıcılardan biri ile haberleşme yapıldığı diğeri ile haberleşme yapılamadığı durumdur. Uyarının yazdığı taraftaki algılayıcı ile haberleşme kurulamamış demektir.

### 5.9.1. SIC_OLC_IC Sıcaklık Algılayıcı Hattı Testi

- **“VPIL”** anahtarını **“AÇIK”** konumuna getiriniz.
- **“KART_KAPALI”** anahtarını **“AÇIK”** konumuna getiriniz.
- **“KART_BEKLEMEDE”** anahtarını **“AÇIK”** konumuna getiriniz.
- **“ALGILAYICI”** anahtarını **“ SIC_OLC_IC ”** konumuna getiriniz.
- **“SICAKLIK TEST”** anahtarını **“AÇIK”** konumuna getiriniz.
- Tablodaki ölçümleri alınız. **KAYIT.**

| Algılayıcı Adı | Alt Limit | Üst Limit |
|---|---|---|
| U1501 | 16 ºC | 50 ºC |
| U1503 | 16 ºC | 50 ºC |

- **“SICAKLIK TEST”** anahtarını **“KAPALI”** konumuna getiriniz.

### 5.9.2. SIC_OLC_DIS Sıcaklık Algılayıcı Hattı Testi

- **“ALGILAYICI”** anahtarını **“ SIC_OLC_DIS ”** konumuna getiriniz.
- **“SICAKLIK TEST”** anahtarını **“AÇIK”** konumuna getiriniz.
- Tablodaki ölçümleri alınız. **KAYIT.**

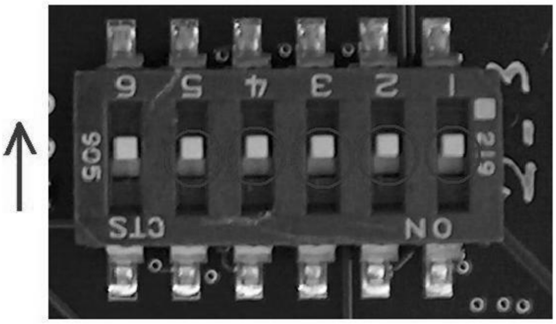
| Algılayıcı Adı | Alt Limit | Üst Limit |
|---|---|---|
| U1502 | 16 ºC | 50 ºC |
| U1504 | 16 ºC | 50 ºC |

- **“SICAKLIK TEST”** anahtarını **“KAPALI”** konumuna getiriniz.
- **“KART_BEKLEMEDE”** anahtarını **“BEKLEMEDE”** konumuna getiriniz.
- **“KART_KAPALI”** anahtarını **“KAPALI”** konumuna getiriniz.
- **“VPIL”** anahtarını **“KAPALI”** konumuna getiriniz.

## 5.10. 12V_SOG SİNYALİ ÇIKIŞ HATA TESTLERİ

### 5.10.1. SW1001 Anahtarı Kontrolü

Kart üzerinde bulunan SW1001 anahtarında 1, 2, 3, 4, 5 numaralı anahtarları yukarı konumuna **( OFF konumuna )** getiriniz.



***Resim-5 – Anahtarlar OFF konumunda***

Tablodaki noktalar arasından direnç ölçümü yapınız. **KAYIT.**

| **Ölçüm Noktası "+" Kutbu** | **Ölçüm Noktası "-" Kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| 12V_SOG | 12V_SOG _TEST | 10.0 kΩ | - |
| 12V | 12V_TEST | 10.0 kΩ | - |
| 3V3 | 3V3_TEST | 10.0 kΩ | - |
| 5V | 5V_TEST | 10.0 kΩ | - |
| 5V_EKSI | 5V_EKSI_TEST | 10.0 kΩ | - |

- **"12V_SOG_TEST", "12V_TEST", "3V3_TEST", "5V_TEST", "5V_EKSI_TEST"** anahtarlarını **"AÇIK"** konumuna getiriniz.
- Tablodaki ölçümleri alınız. **KAYIT**

| **Ölçüm Noktası "+" Kutbu** | **Ölçüm Noktası "-" Kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| 12V_SOG_TEST | GND | 11.900 V | 12.100 V |
| 12V_TEST | GND | 11.900 V | 12.100 V |
| 3V3_TEST | GND | 3.200 V | 3.400 V |
| 5V_TEST | GND | 4.900 V | 5.100 V |
| 5V_EKSI_TEST | GND | -5.100 V | -4.900 V |

- **"12V_SOG_YUK1"** anahtarını **"AÇIK"** konumuna getiriniz.
- **"VPIL"** giriş gerilimini **"8.00 V"** değerine ayarlayınız.
- **"VPIL"** anahtarını **"AÇIK"** konumuna getiriniz.
- **"KART_KAPALI"** anahtarını **"AÇIK"** konumuna getiriniz.
- **"KART_BEKLEMEDE"** anahtarını **"AÇIK"** konumuna getiriniz.

### 5.10.2. 12V_SOG Gerilimi Yüksek Gerilim Hatası Testi

**5.10.2.1. 12V_SOG_TEST** gerilimini **12.000 V** değerine ayarlayınız.

Tablodaki noktalardan ölçüm alınız. **KAYIT.**

| **Ölçüm Noktası "+" kutbu** | **Ölçüm Noktası "-" kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 7.500 V | 8.000 V |

**5.10.2.2. 12V_SOG_TEST** gerilimini **12.453 V** değerine ayarlayınız.

Tablodaki noktalardan ölçüm alınız. **KAYIT.**

| **Ölçüm Noktası "+" kutbu** | **Ölçüm Noktası "-" kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 7.500 V | 8.000 V |

**5.10.2.3. 12V_SOG_TEST** gerilimini **13.356 V** değerine ayarlayınız.

Tablodaki noktalardan ölçüm alınız. **KAYIT.**

| Ölçüm Noktası “+” kutbu | Ölçüm Noktası “-” kutbu | Alt Limit | Üst Limit |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 0.000 V | 0.200 V |

Kart üzerindeki ikaz ışıklarının durumlarını kontrol ediniz. **KAYIT.**

| İkaz Işığı Adı | Tanım | Gözlem |
|---|---|---|
| LA1601 | VGIR_REG | Kapalı |
| LA1602 | 12V_SOG | Kapalı |
| LA1603 | 12V | Kapalı |
| LA1604 | 3V3 | Kapalı |
| LA1605 | 5V | Kapalı |
| LA1606 | 5V_EKSI | Kapalı |
| LA1607 | 3V3_DNTM | Açık |
| LA1608 | KILIT_UYARI | Kapalı |
| LA1609 | 5V_LRF | Kapalı |

**5.10.2.4. 12V_SOG_TEST g**erilimini **12.000 V** değerine ayarlayınız.

Tablodaki noktalardan ölçüm alınız. **KAYIT.**

| Ölçüm Noktası “+” kutbu | Ölçüm Noktası “-” kutbu | Alt Limit | Üst Limit |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 0.000 V | 0.200 V |

Kart üzerindeki ikaz ışıklarının durumlarını kontrol ediniz. **KAYIT.**

| İkaz Işığı Adı | Tanım | Gözlem |
|---|---|---|
| LA1601 | VGIR_REG | Kapalı |
| LA1602 | 12V_SOG | Kapalı |
| LA1603 | 12V | Kapalı |
| LA1604 | 3V3 | Kapalı |
| LA1605 | 5V | Kapalı |
| LA1606 | 5V_EKSI | Kapalı |
| LA1607 | 3V3_DNTM | Açık |
| LA1608 | KILIT_UYARI | Kapalı |
| LA1609 | 5V_LRF | Kapalı |

**5.10.2.5.**

- **“KART_BEKLEMEDE”** anahtarını **“BEKLEMEDE”** konumuna getiriniz.
- **“KART_KAPALI”** anahtarını **“KAPALI”** konumuna getiriniz
- **“KART_KAPALI”** anahtarını **“AÇIK”** konumuna getiriniz.
- **“KART_BEKLEMEDE”** anahtarını **“AÇIK”** konumuna getiriniz.

Tablodaki noktalardan ölçüm alınız. **KAYIT.**

| **Ölçüm Noktası “+” kutbu** | **Ölçüm Noktası “-” kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 7.500 V | 8.000 V |

Kart üzerindeki ikaz ışıklarının durumlarını kontrol ediniz. **KAYIT.**

| **İkaz Işığı Adı** | **Tanım** | **Gözlem** |
|---|---|---|
| LA1601 | VGIR_REG | Açık |
| LA1602 | 12V_SOG | Açık |
| LA1603 | 12V | Açık |
| LA1604 | 3V3 | Açık |
| LA1605 | 5V | Açık |
| LA1606 | 5V_EKSI | Açık |
| LA1607 | 3V3_DNTM | Açık |
| LA1608 | KILIT_UYARI | Kapalı |
| LA1609 | 5V_LRF | Açık |

### 5.10.3. 12V_SOG Gerilimi Düşük Gerilim Hatası Testi

**5.10.3.1. 12V_SOG_TEST** gerilimini **11.593 V** değerine ayarlayınız.

Tablodaki ölçümleri alınız. **KAYIT.**

| **Ölçüm Noktası “+” kutbu** | **Ölçüm Noktası “-” kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 7.500 V | 8.000 V |

**5.10.3.2. 12V_SOG_TEST** gerilimini **10.810 V** değerine ayarlayınız.

Tablodaki ölçümleri alınız. **KAYIT.**

| **Ölçüm Noktası “+” kutbu** | **Ölçüm Noktası “-” kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 0.000 V | 0.200 V |

Kart üzerindeki ikaz ışıklarının durumlarını kontrol ediniz. **KAYIT.**

| **İkaz Işığı Adı** | **Tanım** | **Gözlem** |
|---|---|---|
| LA1601 | VGIR_REG | Kapalı |
| LA1602 | 12V_SOG | Kapalı |
| LA1603 | 12V | Kapalı |
| LA1604 | 3V3 | Kapalı |
| LA1605 | 5V | Kapalı |
| LA1606 | 5V_EKSI | Kapalı |
| LA1607 | 3V3_DNTM | Açık |
| LA1608 | KILIT_UYARI | Kapalı |
| LA1609 | 5V_LRF | Kapalı |

**5.10.3.3. 12V_SOG_TEST g**erilimini **12.000 V** değerine ayarlayınız.

Tablodaki ölçümleri alınız. **KAYIT.**

| **Ölçüm Noktası "+" kutbu** | **Ölçüm Noktası "-" kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 0.000 V | 0.200 V |

Kart üzerindeki ikaz ışıklarının durumlarını kontrol ediniz. **KAYIT.**

| **İkaz Işığı Adı** | **Tanım** | **Gözlem** |
|---|---|---|
| LA1601 | VGIR_REG | Kapalı |
| LA1602 | 12V_SOG | Kapalı |
| LA1603 | 12V | Kapalı |
| LA1604 | 3V3 | Kapalı |
| LA1605 | 5V | Kapalı |
| LA1606 | 5V_EKSI | Kapalı |
| LA1607 | 3V3_DNTM | Açık |
| LA1608 | KILIT_UYARI | Kapalı |
| LA1609 | 5V_LRF | Kapalı |

**5.10.3.4.**

- **"KART_BEKLEMEDE"** anahtarını **"BEKLEMEDE"** konumuna getiriniz.
- **"KART_KAPALI"** anahtarını **"KAPALI"** konumuna getiriniz
- **"KART_KAPALI"** anahtarını **"AÇIK"** konumuna getiriniz.
- **"KART_BEKLEMEDE"** anahtarını **"AÇIK"** konumuna getiriniz.

Tablodaki noktalardan ölçüm alınız. **KAYIT.**

| **Ölçüm Noktası "+" kutbu** | **Ölçüm Noktası "-" kutbu** | **Alt Limit** | **Üst Limit** |
|---|---|---|---|
| GUC_CIKIS_HATA | VGIR_RTN | 7.500 V | 8.000 V |

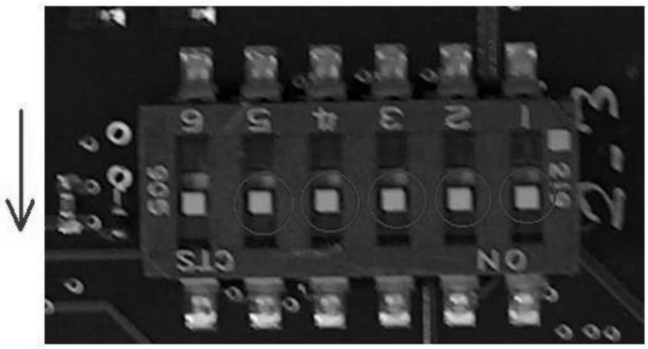
Kart üzerindeki ikaz ışıklarının durumlarını kontrol ediniz. **KAYIT.**

| **İkaz Işığı Adı** | **Tanım** | **Gözlem** |
|---|---|---|
| LA1601 | VGIR_REG | Açık |
| LA1602 | 12V_SOG | Açık |
| LA1603 | 12V | Açık |
| LA1604 | 3V3 | Açık |
| LA1605 | 5V | Açık |
| LA1606 | 5V_EKSI | Açık |
| LA1607 | 3V3_DNTM | Açık |
| LA1608 | KILIT_UYARI | Kapalı |
| LA1609 | 5V_LRF | Açık |

## 5.11. TEST SONLANDIRMA KONTROLÜ

### 5.11.1. SW1001 Anahtarı Kontrolü

Kart üzerinde bulunan SW1001 anahtarında 1, 2, 3, 4, 5 numaralı anahtarları aşağı konumuna **(ON konumuna)** getiriniz.



***Resim-6 – Anahtarlar ON konumunda***

Tablodaki noktalardan direnç ölçümü yapınız. **KAYIT.**

| **Ölçüm Noktası "+" Kutbu** | **Ölçüm Noktası "-" Kutbu** | **Ölçüm Limiti (ohm)** |
|---|---|---|
| 12V_SOG | 12V_SOG _TEST | ≤ 2.0 Ω |
| 12V | 12V_TEST | ≤ 2.0 Ω |
| 3V3 | 3V3_TEST | ≤ 2.0 Ω |
| 5V | 5V_TEST | ≤ 2.0 Ω |
| 5V_EKSI | 5V_EKSI_TEST | ≤ 2.0 Ω |

### 5.11.2. İkaz Işıkları Kontrolü

- **"VPIL"** anahtarını **"AÇIK"** konumuna getiriniz.
- **"KART_KAPALI"** anahtarını **"AÇIK"** konumuna getiriniz.
- **"KART_BEKLEMEDE"** anahtarını **"AÇIK"** konumuna getiriniz.

Kart üzerindeki ikaz ışıklarını kontrol ediniz. **KAYIT.**

| **İkaz Işığı Adı** | **Tanım** | **Gözlem** |
|---|---|---|
| LA1601 | VGIR_REG | Açık |
| LA1602 | 12V_SOG | Açık |
| LA1603 | 12V | Açık |
| LA1604 | 3V3 | Açık |
| LA1605 | 5V | Açık |
| LA1606 | 5V_EKSI | Açık |
| LA1607 | 3V3_DNTM | Açık |
| LA1608 | KILIT_UYARI | Kapalı |
| LA1609 | 5V_LRF | Açık |

- **"VPIL"** anahtarını **"KAPALI"** konumuna getiriniz.
- **"KART_BEKLEMEDE"** anahtarını **"BEKLEMEDE"** konumuna getiriniz.
- **"KART_KAPALI"** anahtarını **"KAPALI"** konumuna getiriniz
- **"12V_SOG_YUK1"** anahtarını **"KAPALI"** konumuna getiriniz.

**<u>Yeni bir kart testi için 5.1 test adımına gidiniz.</u>**