Software Test Automation of Electronic Controller Unit

Shreeji Mahendra Pathak
Post-Graduation Student, Department of Manufacturing Engineering And Industrial Management, College Of Engineering Pune [COEP], India
pathaksm19.prod@coep.ac.in

Dr. Prakash D. Pantawane
Head Of Department, Department Of Manufacturing Engineering And Industrial Management, College Of Engineering Pune [COEP], India
hod.prod@coep.ac.in

Dr. Rajiv B.
Associate Professor, Department Of Manufacturing Engineering And Industrial Management, College Of Engineering Pune [COEP], India
rbh.prod@coep.ac.in

Abstract

In this paper, a different approach for software test automation of automotive Electronic Controller Unit is proposed where automation tool is based on the LabVIEW and TestStand platform. Canoe is used for simulation of environment. Parameters of ECU is accessed using XCP protocol. Communication between LabVIEW and Canoe is established using FDX i.e., Fast Data Exchange. FDX works on UDP protocol. CAN signals cannot be directly accessed by LabVIEW with use of FDX only hence CAPL node is used for accessing CAN signals in case of Unified Diagnostics Service related testcases. Hardware is integrated using VT system.

Keywords
Electronic Control Unit (ECU), Canoe, LabVIEW, TestStand, Test automation tool, CAPL, VT system.

1. Introduction
Today product development of Mechatronics systems in automotive sector is consist of many phases like mechanical design, software design, development, integration, testing, etc. Testing is major part in a product development cycle. There is need to increase the efficiency in product testing and to make the process more productive and increase the quality of testing while being cost effective and less time consuming (Kumar et al. 2018). In product development cycle of automotive world let's consider case of any Electronic Controller Unit of body module, the software which is flashed in ECU needs to be tested many times before final release (Vinaya 2020). Many companies follow method of manual testing using different testing tools which requires more time and there are chances of human error in testing (Yuan et al. 2012). As some projects come along with short span of development cycle it is important to submit Test Report as soon as possible (Tarik 2015). For that purpose, companies are seeking software testing tools that offer a comprehensive solution that helps in achieving above requirements.

In coming days as well as today also in automotive industry lot of comfort features are getting introduced in cars. Different power operated modules are introduced with lot of functionalities (Rajashree et al. 2016, . For such products testing is also needed and to gain efficiency in testing different industries are trying to automate hardware test as well as software test. New mechatronics systems are complex has lot of I/O channels and also works on different types of protocols along with real time control (Jenny et al. 2019, Eun et al. 2009, Dietmar et al. 2014, Claus et al. 2017). A test automation tool that will perform test of functionalities of DUT is highly desirable for this, mainly to achieve high level accuracy, precision and to reduce time!

In this test automation LabVIEW is used along with TestStand and Canoe. Test sequences are written in the excel sheet using different key words.

2. Methodology

Methodology which is used in this automation project is explained in 5 different sections as given below in Figure 1.

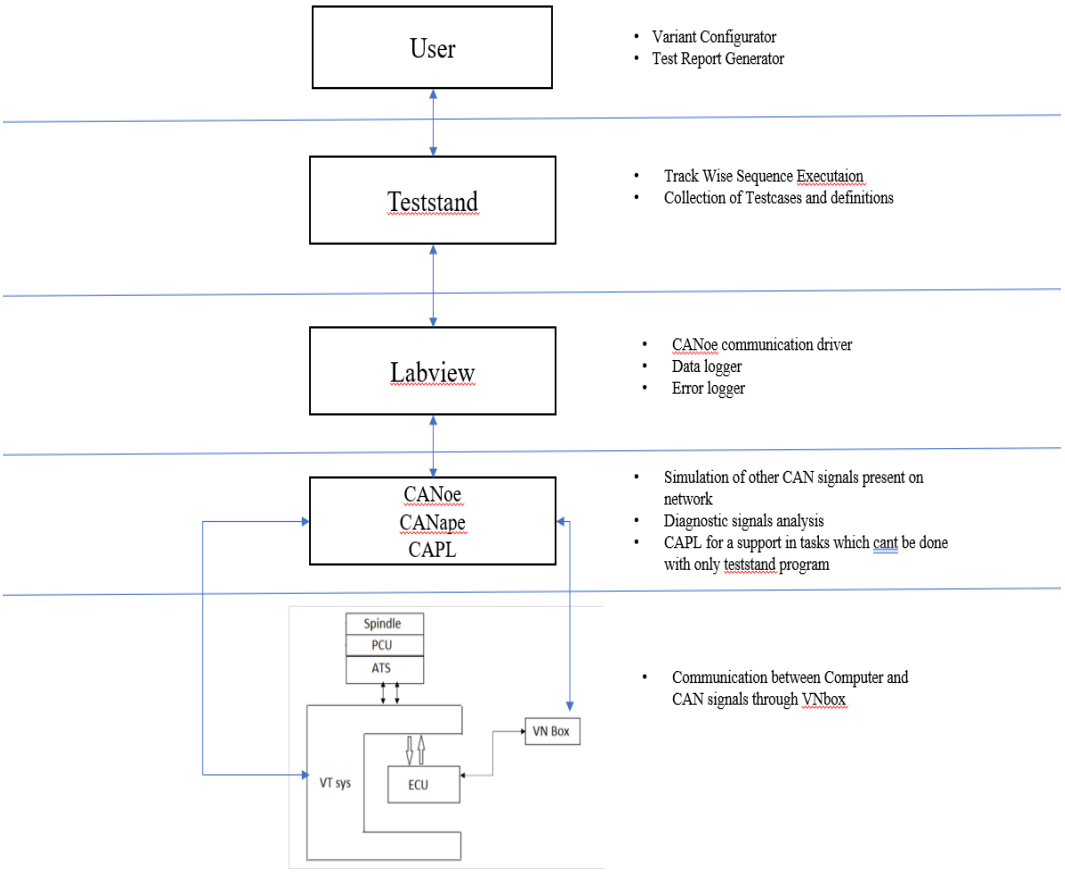

Figure 1. System Architecture

2.1. User Layer

It is a LabVIEW made user interface. Main purpose of this layer will be configuration of the new project which will be going under test automation first time. Testcases are written in the excel sheet using 5 different tabs as given below-

1. Mapped signals
2. Definitions
3. Testcases
4. Table Logic
5. Diagnostics

All these data in excel sheet is needed to be converted into a format which is understandable for LabVIEW for generating sequence file. Hence first step towards it is to convert data written in excel sheet into CSV. For that purpose, macros are developed in excel sheet. Those CSV converted files are then converted into TestStand sequence with use of LabVIEW. In LabVIEW there is TestStand API pallet for conversion of CSV data to sequence, subsequence, and steps in TestStand (Figure 2).

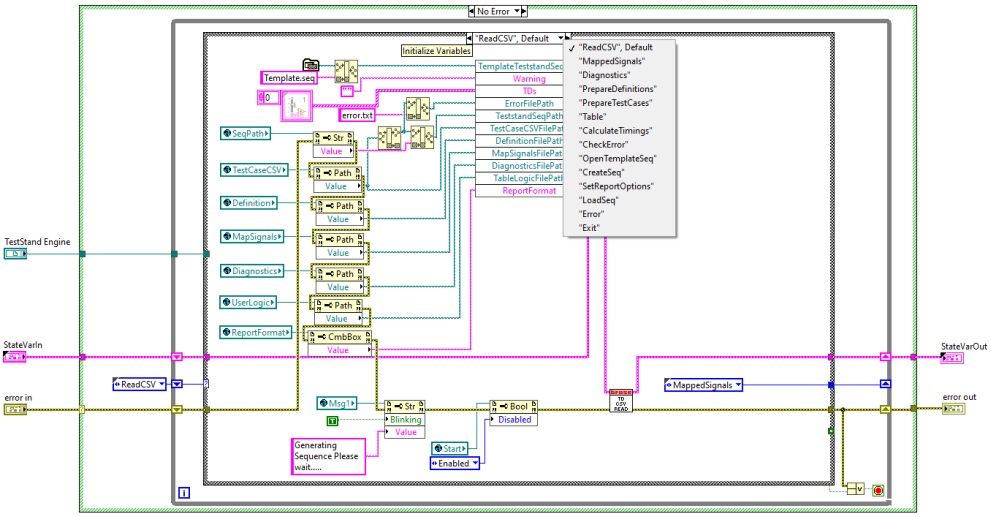

Figure 2. LabVIEW SubVI for converting CSV to TestStand Sequence file.

In the figure given below it is shown the provision for browsing different CSV files. After adding paths Generate button needs to be pressed which converts CSV files in to TestStand sequence file with help of LabVIEW. After generation of sequence file, tool shows message pop up that generation is completed. But in case of any Testcase format related error it pops up error window. Then sequence is loaded in to TestStand with help of TestStand API function in LabVIEW with help of "Load Sequence" button on user interface.

While developing tool various keyword syntax are decided for writing testcase. Keywords which are used in writing testcase are shown in a table given below in Table 1.

Table 1. Keywords used in automation TD writing

Sr.no	Keywords	Meaning	Example
1	{}	To call Definition	{Standardinit}
2	()	To call Map signals	(closeSW)=1
3	[wait]	To wait for particular Timeout	[wait] 10ms
4	[waitfor timeout=2000ms]	To wait until required condition executes or until timeout occurs.	[waitfor timeout=2000ms](PCU_Plus)=1
6	[check]	To validate if the required condition is pass or fail.	[check](MechanicalEndPosLearn)=1
7	>=	Greater than or equal to	[waitfor timeout=2000ms](PCU_Plus)>=1
8	<=	Less than or equal to	[waitfor timeout=2000ms](PCU_Plus)<=1
9	>	Greater than	[waitfor timeout=2000ms](PCU_Plus)>1
10	<	Less than	[waitfor timeout=2000ms](PCU_Plus)<1
11	=	Equal to	[check](MechanicalEndPosLearn)=1

12	!=	Not equal to	[check](MechanicalEndPosLearn)!=1
13	+	Addition	(VehicleSpeed)=(VehicleMotionSpeedLimit)+5

In LabVIEW it is programmed to find out if the written sequence in excel sheet has any mismatch from the rules or format which are defined before. If mismatch is present, then tool pops up the message shown in figure below. And the errors are logged into a notepad file (Figure 3).

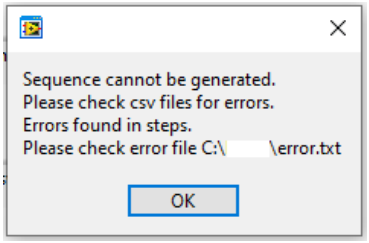

Figure 3. Error Pop up

When user click on the Generate button first task LabVIEW does is that it finds out format related errors present in written sequences. As well as errors it also indicates missing definitions and mapped signals. This feature helps user to identify easily that where format related error has been occurred, but functionality and logic related error has to be taken care by user himself/herself.

This layer is also associated with the test report which is generated after the test execution. Current step execution, expected results, actual results also can be seen on the LabVIEW user interface of automation tool.
In this layer it is also needed to add automation related CAPL files and system variables. Also FDX signals should be added and XCP configurations are required to be done (Figure 4).

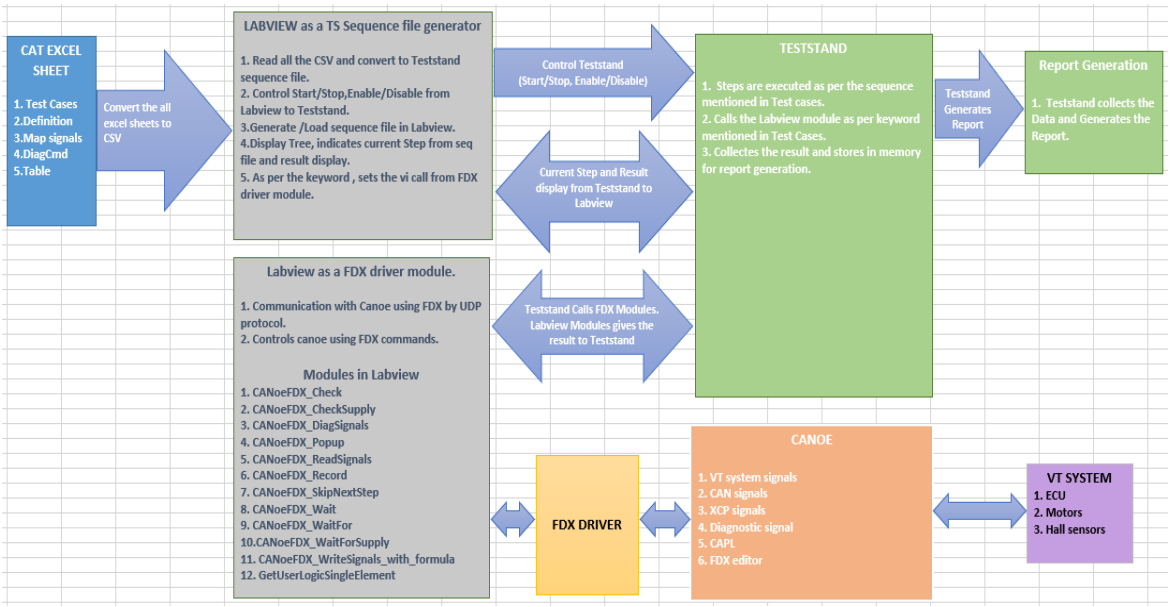

Figure 4. System Architecture

2.2 TestStand layer

NI TestStand is an open test management tool for developing, customizing, and implementing testcases. TestStand plays important role in project to run the actual test sequence. TestStand steps call the LabVIEW FDX module which communicates with Canoe (Figure 5 and Figure 6).

TestStand is also used for generation of Report. It is an inbuilt feature in TestStand. FDX module's status output and result output is given as input to the TestStand report. TestStand Report consist of 3 different inbuilt formats.

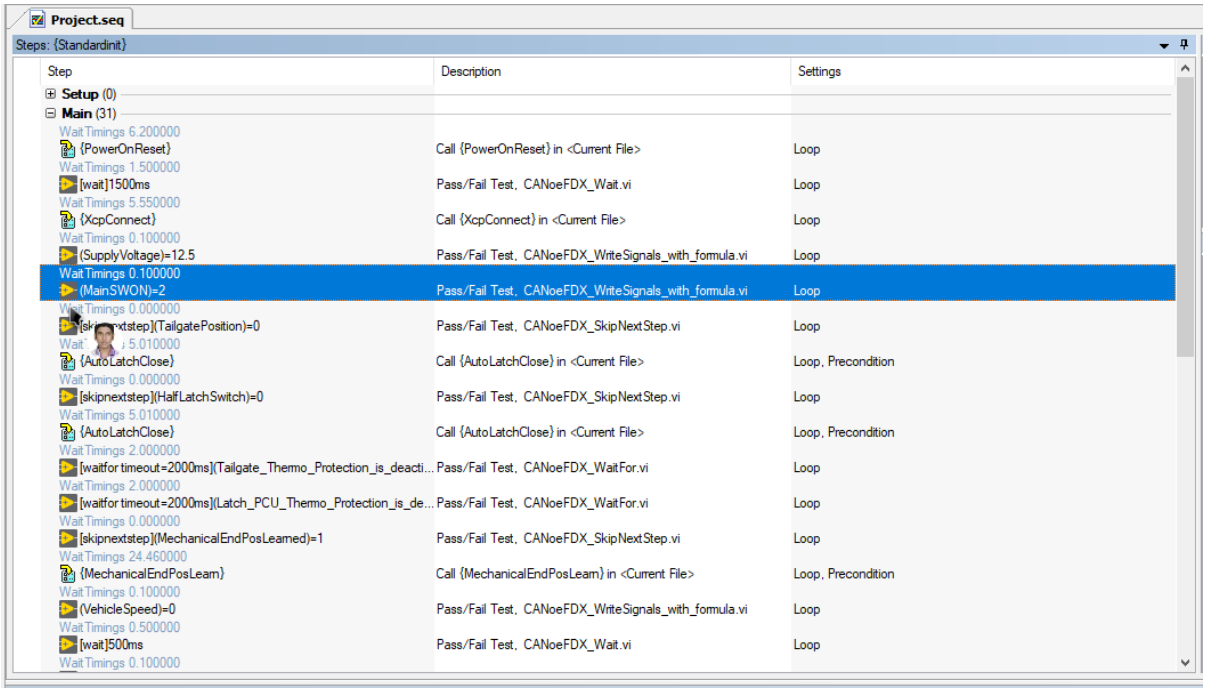

Figure 5. TestStand Sequence Steps

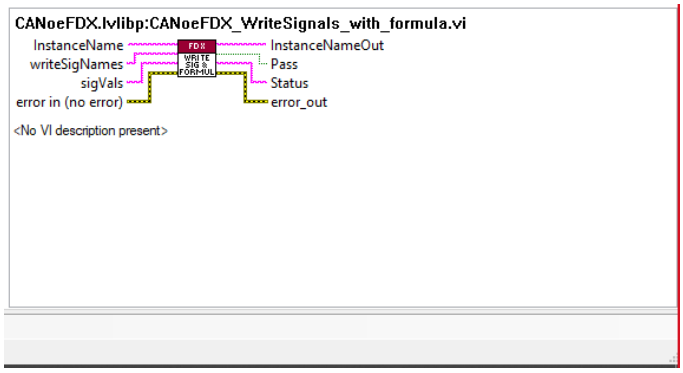

Figure 6. FDX Module VI representation

2.3 LabVIEW-

Main tasks performed with use of LabVIEW are as follows
User Interface –
After creating a sequence file using LabVIEW it is loaded in to TestStand and at the same time it is loaded into a tree (Live Display). When TestStand starts executing steps, each step calls LabVIEW FDX module to transfer mapped signals and its values to CANoe using FDX protocol. FDX protocol sets the signal with its set value. As steps are executed, live step execution and its result i.e., actual signal values are displayed in tree.

FDX module
There are various FDX modules-
FDX Check VI-
This VI is programmed in such a way that from canoe it will fetch the current value of the mapped signal which is mentioned in the step and compare it with the given input value. If values are matched, then step status is set as “Pass” and vice versa.

FDX Diag Signal VI-
This VI deals with the diagnostic services. In a diagnostic tab in excel sheet different diagnostic requests are written. Along with request expected response is also written by the user. Every testcase has testcase number.

Both diagnostic request and response are written in diagnostics sheet along with the testcase number so that tool will differentiate different responses for same request.
While diagnostic step execution request which is to be sent is managed by this VI. As well as the sending of request, receiving response related to same and comparing it with the expected reply is also managed in this VI. In the diagnostic request first byte indicate the length of the request. In excel sheet where requests are written there is no need of writing the first byte i.e. byte length of the request. This VI counts the entered bytes and automatically sends the first byte of UDS request as byte length of request. For example, if user has to send request 03 19 02 09 then only 19 02 09 is written in the diagnostic tab. This data is transferred to CAPL system variables which takes care of CAN communication using canoe. Same happens in case of receiving diagnostics response. CAPL takes care of the byte length of response and saves it in system variable. And actual response bytes are separated in another system variable which is then accessed by this VI.

After receiving of diagnostics response in LabVIEW, it is matched with the expected response which is written in the excel sheet by the user. There are 2 facilities provided- i. Pass/Fail depending up on exact matching of the response with the provided response. ii. Pass/Fail depending up on whether the actual response is in between the range of the given 2 responses.
FDX Record VI-

Many times, in a test sequence while running, value of any mapped signal is needed to be saved temporarily to match up with some other parameter or same parameter further in the sequence. For recording of current value of that mapped signal this VI is used. Mapped signal is saved with the name of [Record1]. [Record1], [Record2], [Record3] likewise up to [Record10] variables can be used.

FDX Read Signals VI-
Read Signals VI is used in FDX Check VI. For fetching the current value of the mapped signal Read Signals VI is used. In FDX protocol every mapped signal has its unique group ID. This group ID is used in Read Signals VI and through UDP protocol current value of the Canoe mapped signal is accessed by the LabVIEW.
These are short explanations of some FDX VIs which are used in automation tool.

2.4 Canoe, XCP, CAPL-

Canoe is used throughout in process of ECU software development. On Canoe different CAN messages are simulated which are sent from the other ECUs. Along with Canoe XCP is also used for accessing internal ECU parameters for automation. CAPL codes are used for some operations which cannot be performed by automation scripts directly using LabVIEW. For example, XCP Connect/ Disconnect, Diagnostics request, Buzzer pattern detection, pulsating response detection on any digital pin, etc.

Handling of diagnostics request with use of CAPL-
Diagnostics request is nothing but sending CAN message on CAN bus and detecting a response related to the same. LabVIEW cannot directly access CAN bus. Hence for automation sequence, sending and receiving of CAN messages is done using CAPL node in Canoe. Input to the CAPL node is given through system variables and input to those system variables are given by LabVIEW as per the test sequence in TestStand. In same way receiving is happened. CAPL node detects the response of diagnostic request and saves it in system variable. Some system variables are used as flags for acknowledgement to LabVIEW tool that response has been detected and present in system variables.

2.5 VT system, VNbox, Wiring harness, ECU, DUT system-

It is the physical layer of the system where actual POT set up is connected to VT system. Different inputs and outputs can be simulated using VT system. Also, communication between computer and CAN signals that is ECU is done in this layer using VNbox. In some cases, Cancase also can be used. Connections which are coming from VT system and vice versa are arranged as a standard which can be used for different OEMs in same standard. So that the connection time is reduced for a tester and no further configuration changes will be needed in future.

3. Procedure to Follow to Start Test Automation

1. Map hardware input outputs in the excel sheet with proper naming.
2. Map ECU signals(parameters) in the excel sheet with proper naming.
3. Configuration of the Canoe simulation of particular project according to the VT system connections.
4. Adding CAPL nodes for different functionalities of automation.

5. Importing system variable files into Canoe simulation.
6. Mapping of the database paths into the TestStand tool. Till now the setup related part is completed. Testcase related part starts now onwards.
7. Using reference of mapped signals write different definitions of required steps in testing like "LatchOpen", "LatchClose", "LearnMechanicalEndPosition", "PowerOnReset", etc.
8. Using this definitions test sequence for different testcases are written in such a way that most of them will be used as it is for new project.
9. All this mapped signals, definitions and testcase sequences are written in the Excel sheets. And these sheets are then converted into CSV files.
10. Load this generated CSV files into the LabVIEW. LabVIEW will generate the sequence accordingly.
11. Start test.
12. Live steps can be seen on the TestStand user interface with the actual and expected values.
13. In the end report is generated into XML format.

4. Results

A simple test case is explained here.
A testcase for sleep current. When ECU is in sleep mode at that time it should consume some specific amount of current. For that in testcase it should have some checks like ECU is in sleep mode and current consumption in specified limits.

4.1 Mapped signals

This is the actual format in which signals are mapped in excel sheet. Rest bus signals, ECU signals and VT system variables has different address pattern. Accordingly, they are mapped in the mapping section. (Figure 7).

#	TesterPresent	BasicUDS::TesterPresent::Active
#	TesterPresentCycleTime	BasicUDS::TesterPresent::CycleTime
#	Antipinch1Pressed	VTS::M2_Ch1::RelayShortCircuit
#	Antipinch1Unpressed	VTS::M2_Ch1::RelayShortCircuit
#	Antipinch2Pressed	VTS::M2_Ch2::RelayShortCircuit
#	Antipinch2Unpressed	VTS::M2_Ch2::RelayShortCircuit

Figure 7. Mapped signals

4.2 Definitions

Example of definition which is made by use of mapped signals is shown below.
Definitions are of different functionalities which are used again and again in most of the testcases. In development phase of automation tool different types of keywords and writing formats can be made using TestStand according to our need (Figure 8)

DEF-17	{WaitTillBasicPosition}		[waitfor timeout = 9000ms] (TailgatePosition) = (BasicPosition) * 0.95[inrange](BasicPosition) * 1.05,
DEF-18	{Antipinch1Trigger}		(Antipinch1Pressed) = 1, [wait] 500ms, (Antipinch1Unpressed) = 0,

Figure 8. Definitions

4.3 Testcase

Testcases are written as per the Test description. Different checks are introduced in between steps. Example of testcase is given below in Figure 9.

TD no	Test case name	Test Initial Conditions	Test Sequence
55	Current Test	{XcpDisconnect},	{ManualLatchOpen}, [wait]1000ms, {AutoLatchClose}, (VDCA9Msg) = 0, [wait]5000ms, (NetworkManagement) = 1, [wait]10000ms, (Vout2_Active) = 1, [wait]10000ms, [waitfor timeout = 10000ms](PS1_AvgCurrent) = 0.000030[inrange]0.000100,

Figure 9. Testcase

4.4 TestStand Execution window

In this window current step execution is displayed. In different tab actual and expected value of variable at moment of execution is displayed (Figure 10).

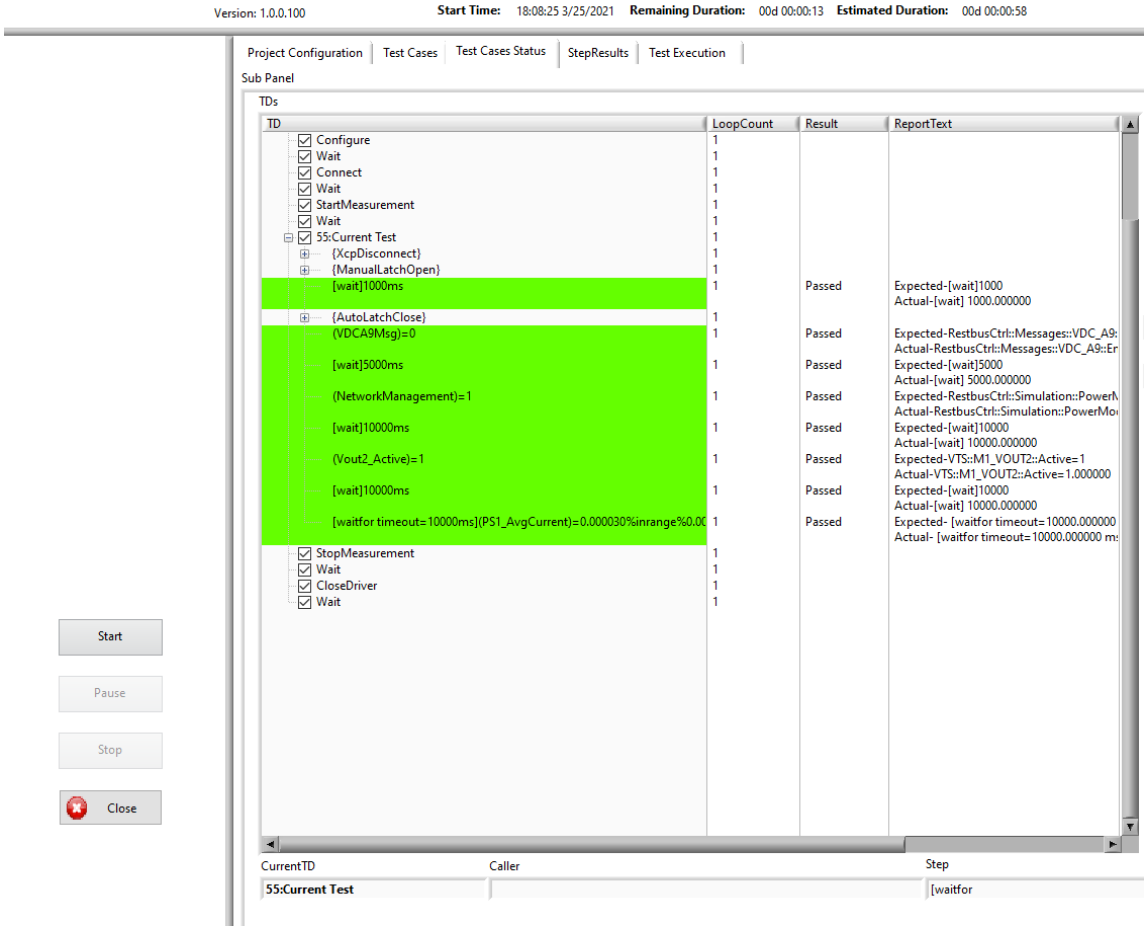

Figure 10. TestStand Execution window

4.5 Report window

After execution of the testcases report is generated in the XML format. Format of the XML report can be changed according to the need. Example of report is shown below in Figure 11.

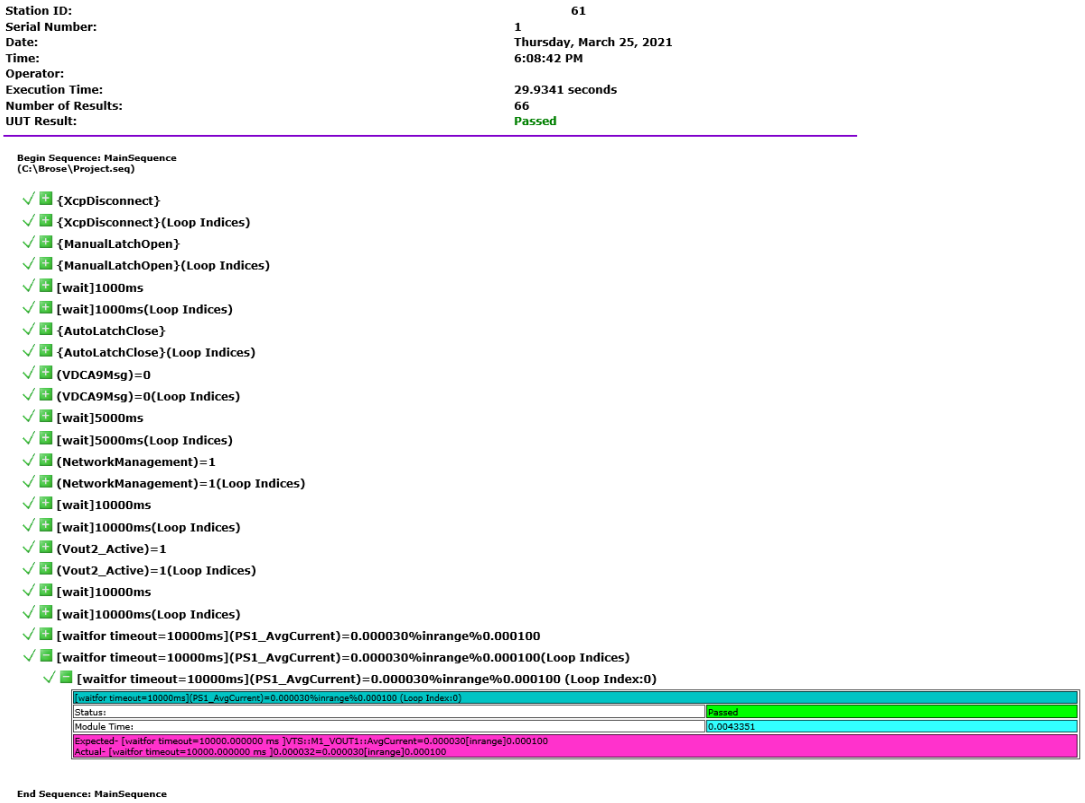

Figure 11. Report Window.

5. Advantage

• Automation tests get recorded and this allows you to reuse and execute the same kind of testing and repetitive tasks multiple times
• After scripts and configuration, test automation requires less time and resources to execute defined scope, compared to manual testing
• Consistent test runs hence we get reliable and consistent results
• Less error caused by human mistake
• No manual error in the test.
• Overnight test is possible and hence resource utilization will be better for organization.
• High reusability of the test case reduces development time (Table 2)

Table 2. Comparison between manual and automated testing.

Testing Method	Time Required (Approx.)
Manual Testing	8 Weeks
Automated Testing	2 Weeks

6. Future Scope

1. Initially these mapped signals, definitions and testcases are written for project “A”. If the definitions and testcases are written in a generic way then after some modifications it is possible to use same testcases and definitions for other projects also. Which will reduce development time of the testcase and definitions.
2. In the end of the testcase graph of selected variables can be introduced in the report. Which will be useful for identification of the exact reason of failure in case of failed testcase.

7. Conclusion

a. Automated testing tool provides robust ECU software logic to verify on vehicle. Smart and highly accurate as well as precise testing can be developed for body modules features across different OEM model's platforms.

b. Most importantly testing time will be reduced hence resource utilization will be better for organization.
c. Not only in automobile sector but also in many other sectors like robotics or industrial automation test automation is need for manufacturing and development process.

References

Claus Klammer, Rudolf Ramler, A Journey from Manual Testing to Automated Test Generation in an Industry Project, 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), 29, July 2017.

Dietmar Winkler, Reinhard Hametner, Thomas Östreicher, Stefan Biffl, A Framework for Automated Testing of Automation Systems, 1Institute of Software Technology, Vienna University of Technology Favoritenstr. 9-11/188, AT 1040 Vienna, Austria Volume 4, Issue 1, July 2014

Eun Ha Kim, Jong Chae Na, and Seok Moon Ryoo, Implementing an Effective Test Automation Framework, 33rd Annual IEEE International Computer Software and Applications Conference. 2009.

Jenny Li, Andreas Ulrich, Xiaoying Bai, Antonia Bertolino, Advances in test automation for software with special focus on artificial intelligence and machine learning, Software Quality Journal, 26, October 2019.

Sharma, Quantitative Analysis of Automation and Manual Testin, International Journal of Engineering and Innovative Technology (IJEIT), 2020.

Rajashree M Bhide, Pratiksha Raut, Rohini S Jadhav, Vaishali S Kulkarni, Sunil L Tade, Test Automation Tool for Electronic Control Unit's Software Testing, International Journal of Scientific and Engineering Research, olume 7, Issue 4, 2016 . ISSN 2229-5518.

Tarik Sheth, Dr. Santosh Kumar Singh, Software Test Automation- Approach on evaluating test automation tools, International Journal of Scientific and Research Publications, Volume 5, Issue 8, 2015.

Kumar U. R, Kavitha K, Canoe Tool for Ecu Automated Communication Testing, International Journal of nnovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-8 Issue-2S December 2018.

Vinaya CH, Automatic Simulation Measurement and Verification of Inputs and Outputs of Body Control Module, International Journal of Innovative Science and Research Technology ISSN No: -2456-2165, 2020.

Yuan Liu, Zhixue Wang, Ruke Zhuang and Jianhui Ma, The Development of Automobile Body Control Module Simulation Testing System, Advanced Materials Research Vols. 466-467 , pp 1084- 1088 , 2012.Online: 2012-02-10 © (2012) Trans Tech Publications, Switzerland.