SCPI based integrated test and measurement environment using LabVIEW

B Aravind Balaji¹, S Sasikumar¹ and K Ramesh²

¹ Electronics and Communication Engineering, Hindustan Institute of Technology and Science, Chennai, Tamil Nadu, India
² Computer Science Engineering, Hindustan Institute of Technology and Science, Chennai, Tamil Nadu, India

Abstract. Test and Measurement processes play a vital role on the quality of hardware functionality testing. Automating these processes provides a better reduction in “time to market” and is more concerned on achieving higher test coverage area when it comes to mass production. Functional testing deals with the observation on the characteristics of the hardware to determine whether it replicates the needs for which it was actually designed. This can be achieved using set of Test and Measurement Instruments which can replicate the environmental condition in which the hardware is to be deployed. This paper provides an environment to integrate set of measurement instruments in a single platform (LabVIEW) and automate the measurement control using command language know as Standard Command for Programmable Instrumentation (SCPI). Script based control commands provide an easy configuration and remote control of instruments. And also programming interface control using Virtual Instrument Software Architecture (VISA) to control instruments using Ethernet and USB protocol.

1. Introduction

Hardware functionality test is the last passage in the assemble cycle to assess the performance of the purpose for which a hardware was structured. Functional testing duplicates the nature of the environment where the hardware is intended to work. Different properties are to be considered in resembling the environment in which they are to be deployed. The necessary generation and acquisition of those properties to create such an environment for the hardware is done using a set of different instruments that are incorporated together and automated using graphical language (LabVIEW). This research work gives a comprehensive exposure on execution of an integrated environment that gives a structure to building such test and measurement.

Conventional remote control of the instruments lack in standardization on two aspects namely programming compatibility and interchangeability. There are various instrument control methods [1] which provide a solution for this drawback. Generally control are carried out by sending a set of commands from the Test system controller(Test application) and ASCII characters are often the ones which are used for commands and the response are revert back from the instrument to the controller in ASCII form[2]. This was not standardized, each instrument contrast from one another which result in a lack of compatibility in programming same instruments of different vendors resulting in reprogramming the whole process with different commands depending upon the vendor. SCPI provides a solution for this providing a standardized script based commands followed by different instrument vendor making programming compatibility simpler [3] [4]. SCPI is simpler since it’s in natural human readable form.
Similarly communication between instruments and test system controller are carried out using different communication protocol such as serial protocol (RS232), Ethernet protocol (TCP, UDP), USB protocol, GPIB [3] [4]. Interchangeability between such protocol results in reprogramming the configuration parameters during updating. VISA is an Application program Interface which provides software based link between the instrument and the controller regardless to the type of protocol in which they are connected. For instance the VISA Instruction to send the script based commands to an instrument is same regardless to the communication protocol of the instrument [3] [4] [6]. LabVIEW acts as test system controller sending such script based instructions or commands to the array of test and measurement instruments simultaneously in order to manage the test environment to the Unit Under Test (UUT) or Hardware [9]. Automated test patterns are generated to observe and evaluate the response. Test report generation is done to evaluate for further analysis.

2. Design flow of integrated environment

The figure 1 shows the design flow diagram of integrated environment which is segregated of four consecutive level, Starting with the test system controller (LabVIEW) it is a graphical programming language based virtual instrumentation workbench which observes, evaluate and automates the measurements and also encapsulate the string based SCPI commands.

![Figure 1. Design Flow of Integrated Environment.](image)

SCPI is an ASCII-based programming language which utilizes a progressive structure known as a tree system [10]. The related commands are gathered under a typical hub or root illustrated with an example below indicates the tree system for current and voltage configuration of a multimeter with arrow marks.

```plaintext
CONFigure:
  → CURRent:
    → AC:RANGE{<range>|MIN|MAX|DEF}
    → AC:RANGE?|MINimum|MAXimum|DEFault|

VOLTage:
  ← AC:RANGE?
```

Where:
- CONFigure - Root identifier.
- CURRent - Second level identifier (under CONFigure).
- VOLTage - Second level identifier (under CONFigure).
- AC - Third level identifier (under CURRent and VOLTage).
- Colon (:) - Isolates back to back identifiers.
- Braces ({})) - Parameter choice.
- Angle brackets (<>) - Specify value.
- Vertical bar (|) - Splits parameters.
- Semicolon (;) - Isolates identifiers within same root identifier.
- Question mark (?) - Query.

VISA a software based architecture which is in form of APIs provides Write and Read function between instruments and controller application via any form of communication protocol [12]. The communication is invoked between them in the form of sessions each session connect the appropriate...
channel in which the resources (instruments) are connected that are indicated with unique resource name and followed by hardware array which is the set of instruments that are to be controlled and monitored. Each instrument is connected in form of different communication protocol such as Ethernet protocol (TCP /UDP), USB protocol and GPIB protocol. And finally the Hardware (UUT) is connected to the array of test instruments. The data flow can be bidirectional, for example the required properties can be queried by sending a string based SCPI command encapsulated by the test system controller (LabVIEW) to the appropriate instrument via VISA driver and the instrument response back to the application with the related data for the query via VISA driver.

3. Software design framework
Figure 2 shows the software design frame work which can be categorized with different layers out of which the VISA plays a vital role and it is the lower level/ Base layer of the framework and its acts as a back bone. Multiple VISA sessions are invoked for communication between array of Test and Measurement Instruments such as programmable AC source, DC Electronic load, multimeter, CRO etc.

![Figure 2. Design Flow of Integrated Environment.](image)

These instruments replicate the source inputs and desired payload to the UUT and observe the response from the UUT. This frame work provides the integration of these Test and measurement instruments within a platform operating concurrently. Multiple VISA sessions are invoked by mean of parallel processing with unique channel for each instrument under the array.

VISA Library supports Write and Read functions in order to send commands and read the response back from the instrument each session starts with open session which provides a connection path between the instrument and the test system controller. This block is provided with all the required parameters such as a unique name/address, type of protocol (USB/TCP/UDP/GPIB), and Port number of the appropriate instruments to be connection. Once the connection is established, depending upon the required operation the read and write functions are used to instruct the instruments to respond. Once the response is acknowledged the session is closed, this process is repeated for all the other VISA Sessions concurrently.

The query is transferred to the instruments in form of string based ASCII codes (SCPI commands) via VISA Write functions and the response from the instruments are acquired using VISA Read function depending upon the different ways to replicate and observe the response of the UUT to evaluate it. The test cases are developed and are executed in a sequential process. There response from the instruments are monitored and compared with the desired expectation limits using windowing techniques these reading are tabulated in a report form to after each and every test cases subsequently.

4. Programming design
LabVIEW is a virtual instrumentation Automation tool used in industrial automations. Programs are based on graphical block representation and can be simply dragged, dropped and connected together.
The figure 3 below represents the design of a part of single test case algorithm illustrating the concurrent operation of sourcing 100V to the UUT using an AC source and measuring the AC voltage and current. The execution of the blocks are from left to right one after the another, parallel processing is possible by placing the block parallel to each others.

**Figure 3.** Test case part A.

VISA session acts as the back bone of the program in commanding the array of instruments depending upon the test cases. The session starts with VISA Open block that initiates the communication with the instruments and the resource name determines which instrument is connected. Open blocks are placed out of the loop which allows the session to be active until all the test case are executed which reduces the execution time cycle. The VISA Write and Read function are used to send and receive the strings to the instruments VISA Flush can be used to clear the I/O buffer in order to clear the previous query in the buffer which provides a better data refresh rate. The SCPI commands are parser in form string constants and are fetched to the instruments via Write function and similarly responds from the instrument are read by the Read function subsequently once the command is sent via write function and these received strings are unparsed using subset strings for further observation.

For example the figure 4 Shows the return string from AC source for a query to measure AC voltage i.e. 2.2808E+02 which mean its $2.28 \times 10^2 = 228$V. The return string is in form of array of strings which are to be subset or split into characters using indexing.

**Figure 4.** Subset string.

Here it has 0-9 index values, Subset block is used to split the array depending upon the offset and length of strings to be subset. Depending upon the test algorithm each process is carried out in sequential form one after another using flat Sequential within a test case. The entire program is looped under a case structure which provides a sequential operation of test cases one by one. Figure 5 illustrates remaining part of a test case algorithm.
The return string is split into subsets and is observed if the properties are within specific range using windowing method. If the response is not within desired range the test is halted automatically. An interlock mechanism is provided to ensure that whenever the desired range is not achieved the entire instrument array is reset to initial position and the power provided to the instruments is turned off terminating the program. This prevents the instruments from damage due to faulty UUT. The reports generation plays a vital role in the whole test process logging the response of the UUT in each and every test case throughout the test process in an excel Sheet automatically, with unique serial number for each UUT for identification of each UUTs response for further post observation query. VISA Close session is invoked at the end of the whole test process to terminate the connection.

5. Application implementation

With the reference of the Design Environment, The paper takes Functional testing software design of a SMPS for testing its integrity and efficiency. The design consists of a Digital Storage Oscilloscope (DSO), Multimeter, AC source and Electronic load. Cased and controlled via Ethernet and USB protocol using VISA Architecture. Test cases are developed depending upon the required characteristics to be evaluated. Test case measurement include Nominal load Current measurement, Power measurement, Ripple measurement, Calibration, Protection circuitry and MOSFET Switching Speed. Graphical User Interface (GUI) in the figure 6 is developed in such a way to provide a live visual status of the testing process using indicators and display of appropriate outputs parameters. Time stamping is provided for report generation and browser options for selecting the desired Excel sheet. Auto Rest control is provided to rest the program and instruments to initial condition. Dropdown menu is provided to choose between automated functional test and Auto reset.

Figure 5. Test case part B.

Figure 6. Graphical user interface.
6. Result
Thus the desired implementation of the functional test environment for SMPS is implemented and the test was successfully performed connecting the SMPS to the test equipment built with array of test. The figure 7 (a) shows the hardware response of the test equipment during runtime. A 24V SMPS is used as prototype for evaluating the test environment and test cases are developed to cross check the performance of the SMPS and maximum test coverage area is achieved. Fault diagnosis and interlock mechanism was cross check by testing a faulty board. Final output report was generated with appropriate reading printed in a excel sheet shown in figure 7(b) The plan strategy has perceptibly Decreased the “time to market” and shorted the development time.

Figure 7. (a) Hardware response (b) Final report.

7. Future scope
This proposed worked can be further more enhanced by focusing more on Internet Of Things (Iot) based instrument control and fault diagnosis, prescient and prescriptive examination and data base management system based report generation for query based observation.

8. References
[1] Zhenjie zeng, Xiaofei zhu, Shiju qi, Xiaowei shen and Shangjun 2018 preliminary Exploration of Instrument control Technology IOP conf. Ser: Mater. Sci Eng, 394 052064
[2] Wanzeng cai, Binwen wang and Shoulong zhang 2017 Remote Control and Data Acquisition of Multiple Oscilloscopes Using LabVIEW IEEE trans. International conference Computer Technology Electronics and Communication 920-24
[3] Liu guili and Kong quancun 2013 Design of Virtual Oscilloscope Based on GPIB Interface and SCPI 11th IEEE International Conference Electronic Measurement & Instruments 294-98
[4] P Sumathi and Deol peter 2019 Instrument control through GPIB-USB Communication with LabVIEW IEEE 28th international symposium on Industrial Electronics 1583-88
[5] Elena daria tica , Lucian andrei perisoara and Alexandru Vasile 2016 An arduino platform for
remote control and bus testing of programmable instruments \textit{IEEE Trans 22nd International Symposium Design and Technology in Electronic Packaging} 308-311

[6] Wei cheng and Fenggui wang 2015 Remote automatic test system based on MATLAB using VISA over LAN \textit{IEEE International Conference on Advanced Mechatronic Systems} 384-87

[7] Jiang chao, Xu wu-bin and Li bing 2013 Design of Instrument Control System Based on LabVIEW \textit{TELKOMNIKA Indonesian Journal of Electrical Engineering} 11 6 3427-32

[8] S Reinhardt, C Butschkow, S Geissler, A Dirnaichner, F Olbrich, C E Lane, D Schröer, and A K Hüttel 2019 Lab::Measurement—A portable and extensible framework for controlling lab equipment and conducting measurements \textit{Elsevier Computer Physics Communications} 234 216-22

[9] Chance elliott, Vipin vijayakumar and Wesley zink. 2007 National Instruments LabVIEW: A Programming Environment for Laboratory Automation and Measurement \textit{Journal of the Association for Laboratory Automation} 12 1 17-24

[10] SCPI Consortium, Standard commands for programmable instrumentation. (SCPI) http://www.ivifoundation.org/docs/scpi-99.pdf

[11] National Instruments Corp, USA, LabVIEW Measurements Manual [M] 2003:5-4-5-10 http://fy.chalmers.se/~f7xlh/LV_Kurs/LV_Meas_Man_2003.pdf

[12] National Instruments Virtual Instrument Software Architecture (VISA) https://www.ni.com/visa/