ELATE: Embedded Low Cost Automatic Test Equipment for FPGA Based Testing of Digital Circuits

Alp Arslan Bayrakci

Gebze Technical University
abayrakci@gtu.edu.tr

Abstract

Testing digital circuits is very crucial but very complex and expensive if high end Automatic Test Equipments (ATE) are employed. In this paper we propose an embedded low cost ATE (ELATE) based on FPGA hardware in communication with computer software, which is able to perform not only the functional tests but also the propagation delay and power consumption tests. The methodology and solution approach explained in this paper can be used to eliminate some generic problems of FPGA based automatic testing as well as to design more precise digital system test equipment by only replacing the underlying hardware components with the better and faster ones.

1. Introduction

Digital circuits surrounding us are continuously increasing in terms of both amount and complexity. Meanwhile, testing digital circuits is getting harder, more expensive yet more crucial.

There are two different approaches in test: high-end automatic test equipment (ATE) and low cost ATE. High end ATE developed by companies like Advantest [1] are very expensive (multi-million dollars) and require advanced engineering skills to be used accurately. This necessitates more approximate and less precise low cost test devices to be used in fast, easy and low cost testing of digital circuits in early phases. Companies like Teseda [2] try to lower the costs. For instance, it had broken $200 per pin cost barrier in 2003 [3].

Costs can be decreased much further using FPGA based hardware and computer software in combination. The advances in FPGA technology supports the development of FPGA based low cost ATE [4, 5, 6]. Low cost FPGA based ATE case study in [7] can perform functional tests of digital circuits. Similar to this paper, it combines the computer software with the FPGA board to build the ATE. It has the advantage of using benchmark HDL to extract test patterns and reconfiguration of FPGA according to the different pin-out configurations of the device under test (DUT). However, it can only perform functional tests and the reconfiguration of the underlying FPGA each time the pin-out changes is a costly process that can be get rid of.

In this paper, we introduce a new embedded low cost ATE (ELATE) methodology that can perform functional, propagation delay (speed) and power consumption tests. ELATE proposes solutions to essential generic problems like different pin-out combination, efficient usage of internal memory storage, which are summarized in Section 2. The capacity limits of ELATE can be easily increased by employing faster and more precise hardware equipments like ADC and FPGA chip without modifying the underlying methodology of ELATE.

Section 3 overviews the architecture of ELATE without getting into detail, Section 4 explains the main solutions proposed in this paper for the problems explained in Section 2 and Section 5 presents the experimental setup and the results for the proposed low cost ATE.

This work is supported by The Scientific and Technological Research Council of Turkey (TUBITAK) under Grant No. 116E296.

2. Motivation

Number of input and output pins of the DUT differs from device to device. Further, the connection scenario of the pins of DUT to the test equipment also varies. Therefore test device must be aware of the pin connections and input output distribution of the DUT in order to perform the tests. Reconfiguring the underlying FPGA based ATE is a solution [7], but for each DUT, a reconfiguration of the FPGA is costly and time consuming. All GPIO connections of the ATE must be easily configured by the user.

Second, the use of memory is an important issue. Unless an efficient memory usage is employed, the tests easily fill up all available memory causing a compulsory interrupt. Just for visualizing the outputs as voltage time graphs in propagation delay tests, all outputs of DUT must be recorded for many times until they all stabilize. Considering that each record is taken in nanosecond intervals, this alone exceeds the memory limits. Similar problem occurs in power consumption tests.

The right time to stop the tests is another ambiguity that must be clarified. During the functional tests the output of the DUT may change more than once and unless the final output value is recorded, the result recorded by ATE would be different than the actual final result of the DUT. It is also important to find the right time to stop the propagation delay and power tests, otherwise the computed delay or power values do not accurately represent the activity corresponding to the current test vector given to the DUT.

Other than these essential problems, there are known issues that must be handled in an ATE design like the automated inputting of the test vectors, performing the user defined test and collecting the results, user customization and easy configuration of the software, software-hardware interfacing, visualizing the results by voltage and current vs time graphics and storing the results both as text files or as graphics.

3. Overview of ELATE

ELATE proposed in this paper aims to solve above listed problems in a generic way so that following the principles explained in this paper anyone having more and better resources can easily upgrade the system to get a better ATE. The overview of ELATE testing system is shown by Fig. 1.

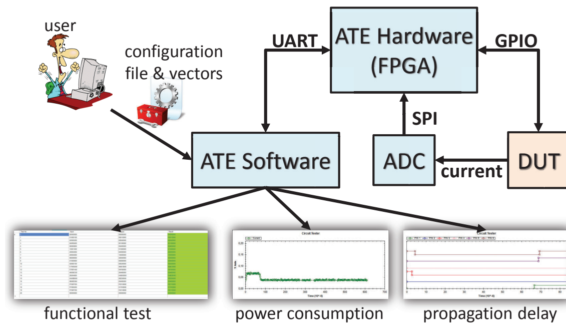


Figure 1. Overview of ELATE

The blue boxes (ATE hardware, ATE software and ADC) in the figure constitute the main structure of ELATE. First of all, user supplies the DUT GPIO connection settings and the input test vectors either as a text file or using the user interface. The settings tab of the software is shown at Fig. 2. The left pane is filled by the user and explains which ELATE pin is connected to which DUT pin and whether it is an input or output. The right pane is again filled by the user and holds the input test vectors in hex format as well as the expected outputs. At the below, there are settings for minimum wait time which shows the amount of time to wait for the output to stabilize and the resistance value of the shunt resistor that is used for power consumption tests. At the top the user specifies the virtual COM port for the corresponding USB connection with the ELATE hardware. The two buttons at the bottom left are used to save all settings and test vectors into a text file and to load these settings from the file respectively. After the settings are ready and the ELATE software is connected with the ELATE hardware, all settings and test vectors are sent to the hardware to be stored inside the FPGA. Then, user may start one of the three tests (functional, propagation delay, power consumption) using the start button located inside the tab of the corresponding test. Test is performed by ELATE hardware by giving each test vector as input to DUT and the results taken from the DUT are sent back to the software.

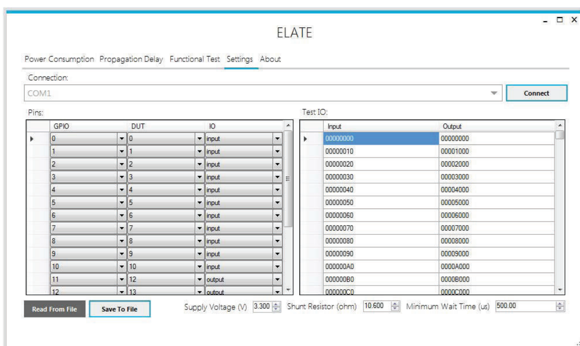


Figure 2. ELATE software settings tab

There are six hardware modules as shown in Fig. 3: $uart$, spi, $test_control$, $functional_test$, $propagation_delay$, $power_consumption$. All of these modules are written as finite state machines (FSM) and in Verilog HDL.

Duplex communication with the software is handled by the $uart$ module whereas spi module is used to communicate with the ADC which is used only in power consumption tests. The incoming input test vectors and settings are taken through $uart$ module. Storing the test vectors to the internal memory and configuring the gpio pins in accordance with the user settings is the main responsibility of $test_control$ module. According to the test selected by the user through software, $test_control$ module sends start command to the corresponding test module and supplies the input test vectors individually.

If the functional test is desired, start signal is sent to $functional_test$ module. The module changes to an active state and waits for the first input test vector. Each test vector is supplied by the $test_control$ module. Each input test vector is given to the corresponding DUT input pins selected as input by the user settings, which is clarified in the next section. The output of the DUT is traced until it stabilizes and the result is recorded to the internal registers of the $functional_test$ module. Then, before continuing on the new test vector, the current result is sent to ELATE software via the $uart$ module. All test vectors are handled by $functional_test$ module in the same manner. At the end, the user visually sees the outputs for all test vectors and whether there is a mismatch between the actual output and the expected one with color codes (green output as in Fig. 5a means a match, while red means a mismatch).

If the propagation delay test is started by the user, this time the $propagation_delay$ module transits into the active state by the incoming start signal. At first, the input test vector is given to DUT. Then, $propagation_delay$ module traces all outputs of the DUT and waits until the outputs get stabilized. Outputs are controlled with a 200MHz clock. This results in an error of at most 4ns, which can be easily improved just by using much faster FPGA architectures supporting faster clocks without modifying the underlying design. Only the changes at the outputs are recorded with time stamps. If the output does not change no new record is taken. All records are stored inside the $memory$ module. The $memory$ module enables 32768 changes to be stored in before the outputs are stabilized. Propagation delay is computed by using the time stamp of the latest changing output of the DUT. All stored data is sent to the ELATE software through $uart$ module to be stored in computer and visualized with graphics by the software. Then, the next test vector is given to DUT and the same operations are performed for each vector until all test vectors are over.

Power tests are handled by $power_consumption$ module. It is activated with a start signal and similarly it handles each test vector one by one. It collects the ADC outputs which represents the voltage values over a shunt resistor connected between the ground and the DUT as commonly preferred. The shunt voltage measurements are done by the ADC after the input test vector is applied. The resultant voltage values are stored in $memory$ module and are sent through UART to the software before switching to the next test vector incoming from the $test_control$ module. ELATE software stores these voltage values and constructs the current-time graph by dividing the shunt resistor voltages to the value of the shunt resistor. Using integral at the current-time graph, the total energy consumption is easily computed. Dividing this energy consumption to the time difference gives the average power consumption which is shown to user and can be saved as a text file.

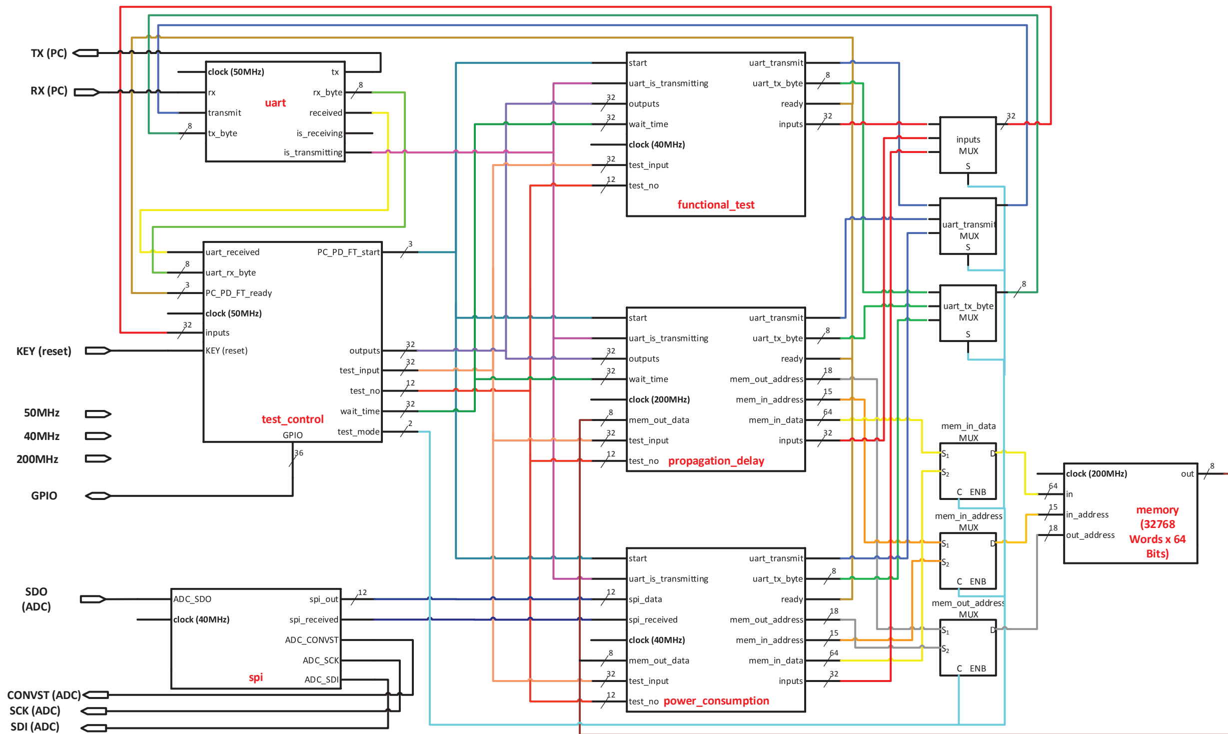


Figure 3. ELATE hardware block diagram

Using a much faster ADC would increase the cost of the ATE but results in much better power consumption estimates again without any major modification at the underlying design or *power_consumption* module.

As referred above, *memory* module in Fig. 3 is shared by two modules: *propagation_delay* and *power_consumption*. This explains the necessity of the three 2x1 multiplexers located at the bottom right of the figure. Each of the three test modules sends the results for the current test vector through the *uart* module. This similarly explains the requirement of the other three multiplexers in Fig. 3.

4. Solutions to Challenges

The main challenges in ATE design are summarized in Section 2. In this section, the proposed solutions are explained.

4.1. Different Input/Output Combination Compatibility

First problem is about the flexibility of ELATE. Different DUTs may have different number of inputs and outputs, which are defined by the user through ELATE software at the left pane shown in Fig. 2. For instance, a DUT may have 8 bit input and 16 bit output whereas another may have 28 bit input and 8 bit output. ELATE must be compatible with any input/output combination provided that the total number of input and output pins do not exceed a certain boundary representing the total number of available pins of the test equipment.

The critical part is to realize the pin settings inside the hardware. In our design, *test_control* module is responsible from the adjustment of GPIO pins of the ELATE hardware according to whatever the user settings dictate. The part of the hardware handling this problem is shown in Fig. 4. In the figure, the leftmost 36x1 multiplexer column is used to select the output pins coming from the DUT. Next to this column of MUXes there are 2x1 multiplexers, which are used to output 0 if the corresponding output is not used as DUT output. The rightmost 32x1 multiplexer column is used to send the selected inputs to the DUT. By this architecture, any GPIO pin can be set as input to DUT (rightmost MUXes) or as output from DUT (leftmost MUXes). In our case study, we assume a total of 36 GPIO pins where at most 32 of them can be used as outputs from DUT. This design saves us from reconfiguration each time the DUT pin out changes. Instead, only the MUX select bits are modified by *test_control* module according to the user settings.

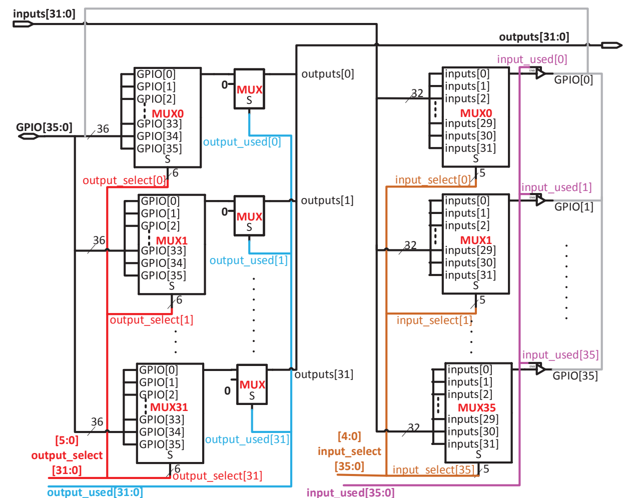


Figure 4. Setting GPIO pins as DUT input or output

4.2. Storage Issue

Test of digital circuits may need huge amounts of data to be stored if not handled carefully. For instance, the propagation delay tests require to capture all outputs of DUT with high frequencies (200MHz in our case) and in the real time data must be stored in a memory in order to be able to show changes at the output signals visually to the user as ELATE software can do. Besides, the capturing process continues until all outputs stabilize, which is unknown a priori. A similar problem exists in power consumption test where the shunt resistor voltage must be recorded at the sampling rate of the ADC.

ELATE hardware prefers "not to store anything until there is an altering". Therefore, $propagation_delay$ module stores the output of the DUT only if the output pins experience a high to low or low to high transition. Otherwise it only increases the time stamp but stores nothing. In other words, a new memory entry is created only if at least one of the DUT outputs is altered. As a result, the number of memory entries cannot go far beyond the number of altering at the DUT outputs. Same strategy is adopted by $power_consumption$ module. Only if there is a significant (more than a predefined threshold) change at the shunt resistor voltage, it is recorded otherwise the old value persists.

Another critical choice is to send all data corresponding to an input test vector to the ELATE software before switching to the next test vector. By this way, all data of the previous vector can be cleared and the next vector can use the same storage locations. This also guarantees that the increasing number of test vectors does not lead to a malfunction due to storage capacity. The data must eventually be sent to the software and whether you send it one by one or send it after collecting all of them does not change the final time all data arrives.

4.3. Time to Stop Issue

For all three tests, right time to stop the test is important as explained in Section 2. For that purpose, the user supplies a minimum waiting time in microseconds through the ELATE software as shown in Fig. 2. In functional test, the test stops if the DUT output does not change for user given amount of time. But if it changes the time counter is reset to zero and the output must again stay stable for the user defined amount of time. This scenario is repeated for each input test vector. In propagation delay tests, similarly all of the DUT outputs must stay stable for user defined amount of time for the test to stop and the delay is computed by taking the difference between the input change instant and the instant of the last changed output. In power consumption test, the recording of the shunt resistor voltage starts when the voltage becomes different from the idle period and the test stops when the voltage returns back to the idle regime voltage values.

If the user estimates an upper bound for the circuit delay, a higher value for the minimum wait time certainly serves the purpose. Otherwise, it can be determined easily by such an adaptive method: at first a very small wait time is given and functional test is performed resulting in many outputs to be different than the expected ones. Then, the user can gradually increase the minimum wait time until the number of correct outputs ceases to increase. Only for the incorrect output results, the user can rerun the functional tests with a much higher minimum wait time in order to be sure that incorrectness is not due to the wait time.

5. Results

ELATE hardware is designed using FSMs written in Verilog on Altera Cyclone V 5CGXFC5C6 FPGA with 6% logic utilization and about 55% memory utilization. Power consumption tests are performed with the help of Linear Technology LTC2308 ADC. ELATE software and user interface are programmed by C#. The communication between ELATE software and hardware is handled through USB and UART protocol. DUTs are connected to FPGA board using 40-pin GPIO. Two different DUTs are used in experiments: Texas Instruments Stellaris Launchpad and Cyclone V GX Starter Kit FPGA board.

Stellaris Launchpad holds a Stellaris microcontroller (MCU), which is programmed using Code Composer Studio so that it accepts 3 inputs and 1 output: 4 bit num1, 4 bit num2 and 3 bit select signal as inputs and 8 bit res signal as output. MCU program performs 8 different arithmetic operations on num1 and num2 according to the value of the select signal: 0. add ($res = num1 + num2$), 1. multiply ($res = num1 * num2$), 2. max ($res = max(num1, num2)$), 3. greatest common divisor, 4. mod ($res = num1\%num2$), 5. in a for loop of $100 \times num1 \times num2$ times accumulate counter in res ($res = res + counter$), 6. num1th element of the fibonacci series written to res, 7. concatenate ($res = num1 + num2 \times 16$).

There is a total of 11 bit input which constitutes 2048 different input test vectors. All possible test vectors are generated and the expected output vectors are computed by a simple C code. ELATE functional test results for all 2048 input test vectors completely overlap the expected outputs which shows the accuracy of the functional test. Also Stellaris is programmed such that the results are sent to the PC terminal to double check ELATE results. Fig. 5a shows the functional test results for addition operation where the leftmost column shows num1 (second digit from right) and num2 (rightmost digit) in hex format, the middle column shows the expected output (fourth and fifth digit from right) and the rightmost column shows the actual output from DUT. As an example case, for test vector 29, num1 is 0xD, num2 is 0x1 and the result is 0xE as expected. Green coloring means a perfect match between the expected and actual output.

Propagation delay and power consumption tests on Stellaris Launchpad consume about a couple of minutes including a minimum wait time (50ms for confidence) for each test vector and the time to send back each result. Results are in accordance with the Analog Discovery oscilloscope and logic analyzer. As an example case, Fig. 5b and 5c demonstrate the ELATE plots for propagation delay and power consumption test respectively. In Table 1, delay results are in micro seconds (us) and power consumptions are in mW. As expected longest delay is because of the long for loop when select is 5. Average fibonacci delay is lower while the worst delay is high. This is because fibonacci output changes only when num1 changes which occurs only 15 times in 256 different input test vectors. Delays can better be examined in Fig. 5d where all delays corresponding to all test vectors are shown as bar graph. Average power does not change from operation to operation because our ADC can not catch the instantaneous changes in current as it is a 500ksps ADC. But the results are very similar to what we get with Analog Discovery oscilloscope.

Lastly, the functional and propagation delay tests for 1024 input test vectors are performed on second DUT, which is Cyclone V GX Starter Kit FPGA development board. The FPGA on board is programmed to perform a 8 bit multiplication.

Table 1. Delay and power results on Stellaris DUT

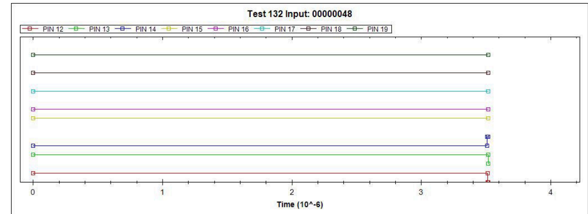
select	operation	avg. delay	worst delay	pow.
0	add	4,92	14,94	4,37
1	multiply	4,87	15,70	4,41
2	max	2,19	11,73	4,09
3	gcd	8,13	32,62	4,40
4	mod	3,31	16,16	4,15
5	long for loop	7268,55	32097,82	4,23
6	fibonacci	4,99	1070,08	4,25
7	concatenate	6,75	14,64	4,31

Therefore, this test circuit has two 8 bit inputs and a 16 bit output. The input/output settings for the second DUT are easily modified through ELATE software without any need for reconfiguration. The functional test results have been verified very similar to Fig. 5a. Propagation delay tests revealed the performance efficiency of FPGA over MCU software. Because the 8 bit multiplications require about 38ns on the average in comparison with 4870ns delay obtained from the multiplication by Stellaris MCU (Table 1). The worst case delay for the FPGA multiplication is about 80ns while for Stellaris multiplication it is 15700ns. Fig. 6 shows the FPGA delays obtained by ELATE for 1024 input test vectors.

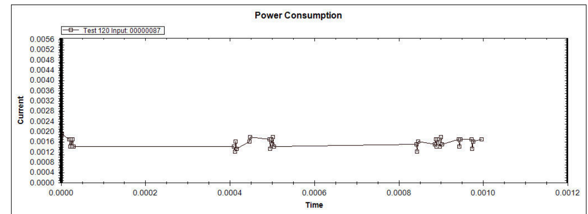
The tests demonstrate the promising results of the proposed ATE bearing in mind the capacity limits like the sampling rate of the employed ADC. Nevertheless, these weaknesses can be eliminated by using more powerful components without altering the design principles of ELATE.

29	000000D1	0000E000	0000E000
30	000000E1	0000F000	0000F000
31	000000F1	00010000	00010000
32	00000002	00002000	00002000
33	00000012	00003000	00003000
34	00000022	00004000	00004000
35	00000032	00005000	00005000
36	00000042	00006000	00006000

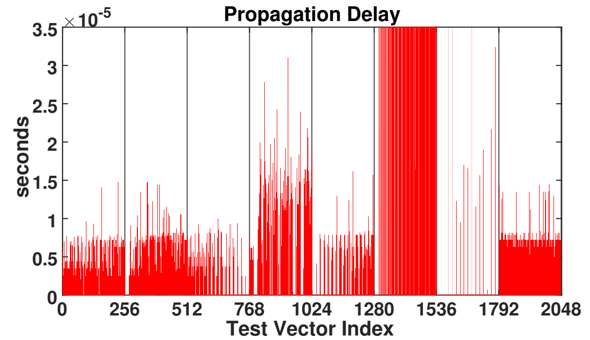
(a) Functional test results for vectors 29 to 36



(b) Propagation delay graph for test vector 132



(c) Current vs time graph for test vector 120



(d) Delay bar graph for 2048 input test vectors

Figure 5. ELATE test results for Stellaris Launchpad

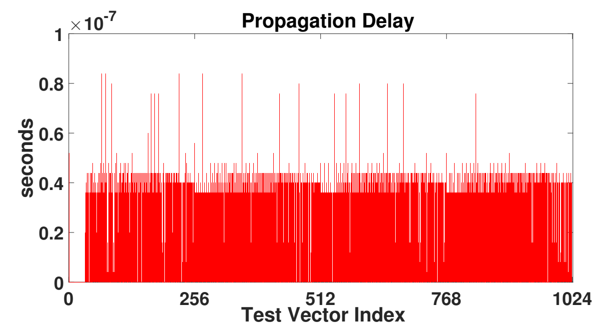


Figure 6. Delay bar graph for Cyclone V development board

6. References

[1] `https://www.advantest.com/products/ic-test-systems`, 'IC Test Systems - Advantest", last accessed: September 2017.

[2] `http://www.teseda.com`, "Teseda Corporation: Silicon Debug, Failure Analysis and Yield Enhancement", last accessed: September 2017.

[3] `http://www.eetimes.com/document.asp?doc_id=1175064`, "Teseda breaks $200 per pin ATE barrier", last accessed: September 2017.

[4] A. L. Crouch, J. C. Potter, A. Khoche and J. Dworak, "FPGA-Based Embedded Tester with a P1687 Command, Control, and Observe-System," *IEEE Design & Test*, vol. 30, no. 5, pp. 6-14, Oct. 2013.

[5] J. H. M. Escobar, J. SachBe, S. Ostendorff and H. D. Wuttke, "Automatic generation of an FPGA based embedded test system for printed circuit board testing," *13th Latin American Test Workshop (LATW)*, Quito, Ecuador, pp. 1-6, 2012.

[6] I. Aleksejev, A. Jutman, S. Devadze, S. Odintsov and T. Wenzel, "FPGA-based synthetic instrumentation for board test," *IEEE International Test Conference*, Anaheim, CA, pp. 1-10, 2012

[7] L. Mostardini, L. Bacciarelli, L. Fanucci, L. Bertini, M. Tonarelli and M. De Marinis, "FPGA-based low-cost automatic test equipment for digital integrated circuits," *IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications*, Rende, pp. 32-37, 2009