DC Parametric Test and IDDQ Test Using Advantest T2000 ATE

by

Jialin Ding

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
August 1, 2015

Keywords: ATE, DC parametric test, ATPG, IDDQ test, Module Trigger

Copyright 2015 by Jialin Ding

Approved by

Victor Nelson, Professor of Electrical and Computer Engineering
Vishwani Agrawal, James J. Danaher Professor of Electrical and Computer Engineering
Adit Singh, James B. Davis Professor of Electrical and Computer Engineering

Abstract

For purpose of improving quality of devices before shipping to customers, VLSI testing methods have been developed to detect defective devices effectively by using automatic test equipment (ATE). To test a chip properly, it is required to test a number of devices to characterize the device and determine manufacturing defects that should be corrected before full production of the device.

Since the characterization and production tests need to be undertaken fast and effectively, automatic test equipment is employed to test each device.

This thesis presents test methods for characterization test, including contact test, input and output voltage level test, input leakage test, IDD current test, power supply voltage bump test and IDDQ test by using the ADVANTEST T2000 test system. This thesis also provides test procedure and results for the each test.

For each of these tests, test results are presented to show the characteristics of each of several devices, implemented with different technologies.

Acknowledgments

I express my deepest gratitude to my advisor, Dr. Victor Nelson, whose advices and patience have guided me throughout my thesis research. Without his guidance, it may be very difficult to finish this thesis research.

I would also express my appreciation to Dr. Vishwani Agrawal and Dr. Adit Singh to be my thesis committee. By reading the book Essentials of Electronic Testing that written by Dr. Michael Bushnell and Dr. Vishwani Agrawal, I acquired a lot of knowledge in VLSI testing, which really helped me understand the basic concept of VLSI testing. Also, the course VLSI testing taught by Dr. Adit Singh is useful that grows my understanding of VLSI testing.

Moreover, I thank to my friend Kunpeng, Baohu who supported me in my thesis research.

Lastly, I also appreciate my parents who always stand by and support me.

Table of Contents

Abstract ii

Acknowledgments iii

List of Figures viii

List of Tables xii

1 Introduction to VLSI Testing 1

1.1 Fault modeling 1

1.2 Testing Procedure 2

1.2.1 Simulation 2

1.2.2 Device debug and characterization test 3

1.2.3 Production test 3

1.2.4 Burn-in test 4

1.3 Typical characterization test program flow 4

1.3.1 Functional test 5

1.3.2 Contact test 5

1.3.3 DC parametric test 6

1.3.4 Power supply current test 6

1.3.5 IDDQ testing 6

1.3.6 AC parametric test 7

1.4 Production test 7

1.5 Thesis goal and organization 7

2 Automatic Test Equipment 9

2.1 Mainframe 9

2.2 Test head 10

2.2.1 DPS500mA 10
2.2.2 Module trigger 13
2.2.3 250MDM 16
2.3 Test program 18
2.3.1 Pin description file 19
2.3.2 Socket file 20
2.3.3 Specification file 21
2.3.4 Levels file 22
2.3.5 Timing and timing map files 23
2.3.6 Test condition file 26
2.3.7 Pattern file 27
2.3.8 Testplan file 28
3 DC Parametric Test 30
3.1 Contact test 31
3.1.1 Test method 31
3.1.2 Test procedure 33
3.1.3 Test program 34
3.1.4 Test result 35
3.2 Input leakage current test 38
3.2.1 Test procedure 40
3.2.2 Test Program 41
3.2.3 Test result 43
3.3 Input threshold voltage test 45
3.3.1 Margin search 46
3.3.2 Test procedure 46
3.3.3 Test Program 47
3.3.4 Test result 48

3.4 Output voltage test 50
3.4.1 Test method 50
3.4.2 Test procedure 50
3.4.3 Test condition 51
3.4.4 Test result 51
3.5 Gross IDD current test 53
3.5.1 Test method 53
3.5.2 Test procedure 53
3.5.3 Test Program 54
3.5.4 Test result 55
3.6 Static IDD current test 55
3.6.1 Test method 56
3.6.2 Test procedure 56
3.6.3 Test Program 57
3.6.4 Test result 58
3.7 Power supply bump test 59
3.7.1 Test method 59
3.7.2 Test procedure 62
3.7.3 Test Program 62
3.7.4 Test result 64
4 IDDQ test 66
4.1 Reason for IDDQ testing 66
4.2 IDDQ theory 67
4.3 Different types of physical defects 68
4.3.1 Bridging fault 68
4.3.2 Gate oxide short (GOS) 71
4.4 IDDQ test vectors 72

4.5 Generating IDDQ test vectors by ATPG 73
4.6 Program example 76
4.6.1 Pattern file 76
4.6.2 Testplan File 77
4.6.3 Levels File 78
4.7 Test program 79
4.8 Test result 80
5 Conclusion and Future Work 85
5.1 Conclusion 85
5.2 Future work 86
5.2.1 Testing output current of devices 86
5.2.2 Emulating IDDQ faults in Field Programmable Gate Arrays (FPGA) 86
Bibliography 87

List of Figures

1.1 General characterization test procedure by ATE [6]. 4

2.1 Simple diagram of the DPS channel [10]. 12

2.2 The operation of a DPS channel [10]. 13

2.3 Pattern generator controls the measurement ADC [10]. 14

2.4 Example of programmed setup for module trigger. 14

2.5 Both SNDT instruction in pattern file and PMDTR in testplan file can determine when to activate module trigger signal to high [11]. 15

2.6 The simple schematic of a 250MDM channel, which is also a PMU [11]. . 18

2.7 Block diagram of test program files on T2000 test system [13]. 19

2.8 Example of a pin description file. 20

2.9 Example of a socket file. 21

2.10 Example of a specification file. 22

2.11 Example of a levels file. 23

2.12 Example of a timing file. 24

2.13 Example of a timing map file. 24

2.14 Example of waveform. 26

2.15 Example of a test condition file. 27

2.16 Example of a pattern file. 28

2.17 Example of a testplan file. 29

2.18 Example of the test flow item that shows in Flow Editor GUI. 29

3.1 The protective diode of DUT. 32

3.2 Contact test setup diagram. 33

3.3 Contact test defect diagram. 33

3.4 Program setup of the levels file. 35

3.5 Program setup of the testplan file. 35

3.6 Voltage result of contact test on device 74HC393 captured from the Console Window. 36

3.7 Input leakage current test. 39

3.8 Program setup of the levels file for IIL test. 42

3.9 Program setup of the testplan file for IIL and IIH test. 43

3.10 The test result of IIL test for 74HC393 captured from the Console Window. 43

3.11 The test result of IIH test for 74HC393 captured from the Console Window. 44

3.12 Program setup of the levels file for VIH test. 47

3.13 Program setup of the testplan file for VIH test. 48

3.14 Program setup of the pattern file for VIH test. 48
3.15 The test result of VIH on the device 74HC393. 48
3.16 The test result of VIL on the device 74HC393. 49
3.17 Program setup of the testplan file for VOH test. 51
3.18 The test result of VOH on the device 74HC393. 52
3.19 The test result of VOL on the device 74HC393. 52
3.20 Program setup of the levels file for gross IDD current test. 54
3.21 Program setup of the testplan file for gross IDD current test. 55
3.22 Static IDD current test setup diagram [6]. 56
3.23 Program setup of the levels files for static IDD current test. 57
3.24 Static IDD current test result of device 74HC163. 58
3.25 Static IDD current test result of device 74LS163. 58
3.26 The connection from PMDTR0-1 on digital module to VBMP0-1 on DPS module [10]. 60
3.27 The supply voltage is fluctuated depending on SNDT and PMDTR0-1 [10]. 61
3.28 Define module trigger signals in the testplan file. 62
3.29 Pattern file example of voltage bump test. 62
3.30 The program setup of the levels file for voltage bump test. 63
3.31 The pattern file used in voltage bump test of the device 74HC393. 64

3.32 The function error in voltage bump test. 65

4.1 IDDQ fault. 68

4.2 Photos of short circuits in physical layouts. 69

4.3 Analysis of short circuit resulting from a bridging fault. 69

4.4 Gate oxide short models. 71

4.5 Do file example for generating IDDQ test vectors. 74

4.6 Statistics of the ATPG produced IDDQ test vectors. 75

4.7 The generated IDDQ test pattern file. 75

4.8 An example pattern file for IDDQ test. 76

4.9 Module trigger changed their values by controlled SNDT instruction [11]. 77

4.10 A part of the testplan file for setting IDDQ test. 77

4.11 The parameters of the levels file for IDDQ test. 78

4.12 The pattern file for IDDQ test. 80

4.13 The IDDQ test result when VIH is 4.5 V and VIL is 0V. 82

4.14 The IDDQ test result when VIH is 4.5 V and VIL is 2.1 V. 83

List of Tables

2.1 DPS500mA specification [10]. 11

2.2 Pin electronics driver specifications [12]. 16

2.3 Pin electronics comparator specification [12]. 17

3.1 Testing result of 74393 chips (Unit for all values is V). 37

3.2 Testing result of 74163 chips (Unit for all values is V). 38

3.3 Test result of 74393 chips (Unit for all values is A). 44

3.4 Test result of 74163 chips (Unit for all values is A) 45

3.5 Input threshold voltage test result of 74393 chips (Unit for all values is V). 49

3.6 Input threshold voltage test result of 74163 chips (Unit for all values is V). 49

3.7 Output driving voltage test result of 74393 chips (Unit for all values is V). 52

3.8 Output driving voltage test result of 74163 chips (Unit for all values is V). 52

3.9 Gross IDD current result of 74393 chips 55

3.10 Gross IDD current result of 74163 chips 55

Chapter 1

Introduction to VLSI Testing

Since the first integrated circuit was invented in 1958, Very Large Scale Integrated Circuit (VLSI) technology has significantly improved in the past decades [1]. Defects created during the fabrication process cannot be ignored since even a tiny defect can make an entire integrated circuit (IC) unusable. For purposes of improving quality of devices before shipping to customers, VLSI testing methods have been developed to detect defective devices effectively by using automatic test equipment (ATE) [2].

Testing a device can be performed at different points from design concept through full device fabrication. Since there are many types of defects, such as bridging faults and stuck-at faults, that need to be detected, a number of different test methods are often used on an IC.

1.1 Fault modeling

A particular fault is the result of a specific type of defect in a VLSI circuit. The existence of a fault in a circuit will likely cause failures of operation. By modeling the effects of each type of fault, one can predict the consequences of a particular fault on the operation of an IC.

There are some basic fault models that are frequently used in VLSI circuit design and test.

Bridging fault

A bridging fault refers to two or more signal lines shorted together. Sometimes, a bridging fault can be regarded as 1-dominant (OR bridge) when the neighbor shorted

line is forced to 1. It can also be treated as 0-dominant (AND bridge) when the neighbor shorted line is forced to 0. In some cases, a bridging fault does not affect the logic state of a neighbor signal line, because of high impedance between two signal lines. However, in this case, the circuit may experience some timing faults. [2].

Stuck-at fault

At the gate level, a stuck-at fault means a node of a circuit is constantly 0 or 1. These nodes can be inputs or outputs of logic gates or flip-flops [2].

Transistor fault

This fault can be described at the transistor level of a logic gate. This kind of fault often behaves as a stuck-open or stuck-on fault. In a stuck-open fault, the transistor is always off (it does not conduct current). In a stuck-on fault, the transistor is always on (it always conducts) [4].

Transition delay fault

With a delay fault, functions of a device may be correct but signal transitions are slower (or rarely more quickly) than normal [4].

1.2 Testing Procedure

VLSI test is required at different stages of design and fabrication of an IC. Each of the tests has its own purpose, as described below.

1.2.1 Simulation

When the design of circuit is being developed with EDA tools, the design needs to be simulated to ensure all functions are correct before fabrication. In this step, a netlist of a design such as a Verilog or VHDL file that has whole structure of the

circuit, is required to be simulated. Once a circuit design has been fully verified, this design can be sent to tape out [5].

1.2.2 Device debug and characterization test

Some of the initial fabricated devices are used as samples to verify the operation of a design, using full functional test and parametric tests (including DC and AC parametric tests) before the design is sent to production. The purpose of characterization test is determining limits of device parameters at which the device will be operational. The purpose of functional test is mainly detecting whether a device has stuck-at faults or other faults that affect logic states. These samples need to be tested in different operating environments to observe how behavior changes based on different operating conditions. Some specific sequences of input test vectors may lead to device failures. If these failures are analyzed deeply, one can find problems in the fabrication process and the chosen test limits. These sample chips can be compared to the designed simulation model. If a device fails initial test, it must be diagnosed to find out the root cause of the failures [5].

1.2.3 Production test

After characterization test, a device will be sent to full scale production. Production test is used to distinguish faulty devices from good devices. Moreover, devices need to be tested on Automatic Test Equipment (ATE). Since a large number of devices must be tested, the test time plays an important role in this step, requiring the design of efficient tests. At this stage, all failed devices are not required to be analyzed to determine the root causes of the failures. During production testing, it is sufficient to simply discard faulty devices [5].

1.2.4 Burn-in test

Research has found that some devices fail in early stages of operation. However, after fabrication these devices can successfully pass the tests listed above. Therefore, burn-in test is often performed to accelerate the life of device to make some devices fail before shipping to customers. During the test, production tests can be performed at high temperature and voltage. In this stage, infant mortality failures and freak failures are tested. In infant mortality failures, an extreme environment will accelerate devices to fail by burning out weak resistive lines. In freak failures, a device works properly in normal conditions but fail after a long time. Thus, these devices are put in burn-in test for a long period [4].

1.3 Typical characterization test program flow

In characterization test, there are always a number of tests that are designed for testing electrical parameters. In this step, engineers are required to set up both ATE and test programs. Figure 1.1 shows the general sequence of test operations performed on an ATE during characterization testing. Some of these tests are also used in production testing.

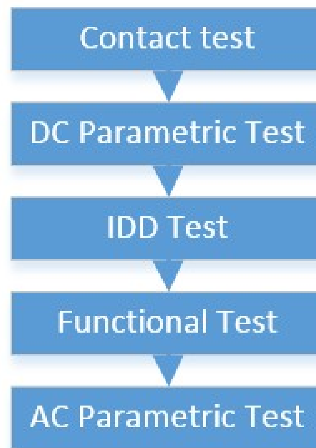


Figure 1.1: General characterization test procedure by ATE [6].

1.3.1 Functional test

In functional test, a sequence of test vectors is applied to a device to check whether the output functions are the same as expected. In current technology, integrated circuits are very large; therefore, there may be many test vectors required to verify all functions. For example, if a device has 60 pins, and each pin has binary state (0 or 1), the number of possible test vectors is

$$2^{60} = 1.15 * 10^{18} \tag{1.1}$$

Assuming that the 250MDMA module of the ADVANTEST T2000 test system is used for doing functional test (this module supports up to 250MHz frequency, or a test period of 4ns), the time for this system to finish all functional tests on the device is

$$1.15 * 10^{18} * 4\,ns = 4.6 * 10^{9}\,s = 1281023\,hours \tag{1.2}$$

In this case, to finish all possible tests will consume a very long time. This is unacceptable. For this reason, other tests such as IDDQ test are employed to reduce test time. For complex devices, functional test performs well in testing stuck-at faults, which IDDQ has limited ability to detect.

1.3.2 Contact test

During device packaging, there is the possibility that a package pin is shorted to other pins or a package pin is open to the internal die. Due to these reasons, checking pin contacts of each device can save time for testing devices, by preventing further tests from being performed on a device with faulty pins [6].

1.3.3 DC parametric test

In this stage, device DC parameters such as leakage current, and input and output voltage are tested to ensure tested values are within the electrical specifications. Margin search test (mentioned in input voltage test in Chapter 3) is used to locate precise input threshold voltages and output driving voltages, then a limited set of values is used for production test [7]. The details of these tests are described in Chapter 3.

1.3.4 Power supply current test

Power supply current tests usually include gross IDD current test, static IDD current test, dynamic IDD current test and power supply bump test. Power supply current test can measure current consumed when device is powered up. In this test, measuring the current value from a power supply can determine if some physical defects exist inside the chip, such as defects caused by under-etching or photo mask misalignment. Usually, this test is performed after contact test and DC parametric test [6].

1.3.5 IDDQ testing

IDDQ testing is a supplemental test method to find bridging and other faults in CMOS circuits by measuring quiescent power supply current. IDDQ testing has become a standard test technique since the end of the 20th century to decrease PPM levels. In addition, IDDQ testing can also discover some potential reliability failures that do not result in any logic failures in functional test. However, IDDQ test is unable to replace functional test because it is limited in detecting many stuck-at faults. So it can be regarded as a supplemental test to reduce overall testing time [8].

In contrast to dynamic IDD test, IDDQ test is required to measure power supply current in steady state after executing each test vector. Hence, in order to test

for a bridging fault, the IDDQ test vector set generated by automatic test pattern generation (ATPG) is to toggle on and off as many transistors as possible.

1.3.6 AC parametric test

The main purpose of AC parametric test is to make sure that the device can meet all timing specifications. AC parametric tests include propagation delay, rise/fall time, setup time, hold time, pulse width and maximum clock frequency tests [6].

1.4 Production test

Production test is typically performed by using GO/NOGO testing. The electrical values used in the test are defined after characterization test. If a measured value is outside a designed range, this device can be judged to be failed. In order to minimize the cost of test, test time is required to be reduced as much as possible. Again, the functional test is performed to detect stuck-at faults or other faults that affect logic states in production test.

This test method is same as characterization test, except without fault diagnosis. Once a device fails to pass a test, this device can be thrown away directly without further testing.

1.5 Thesis goal and organization

There has been much work on developing functional tests. Therefore, design and implementation of DC parametric tests and IDDQ tests for the ADVANTEST T2000 ATE are explored in this thesis.

The thesis is divided into five chapters. The organization of chapters is as followed:

Chapter 2 illustrates the structure and the functions of the ATE.

Chapter 3 describes test strategies and results of contact tests and DC parametric tests performed on the ATE.

Chapter 4 discusses test method and results of IDDQ testing on the ATE and IDDQ test pattern generation using FastScan.

Chapter 5 summarizes the work performed and provides in this project and suggestions for future work.

Chapter 2

Automatic Test Equipment

After a device product has been sent to a foundry to produce chips, it is required to test a number of devices to characterize the device and determine manufacturing defects that should be corrected before full production of the device. Furthermore, after characterization of a device has been confirmed, production test is performed to filter out bad devices that have fabrication defects from the entire group of devices. These steps can guarantee that devices shipped to customers have low defect levels. The characterization and production tests need to be undertaken fast and effectively. Therefore, ATE is employed to test each device. The test engineer designs and sets up the automatic test program and lets the ATE apply the test.

In the experiments performed for this thesis study, the Advantest T2000GS Test System is used for performing contact tests, DC parametric tests and IDDQ tests. The T2000GS test system is an entry level system that contains three units, the mainframe, a user interface console and a test head.

2.1 Mainframe

The mainframe of the T2000GS contains a system controller, site controller and bus switch. The system controller is the primary connection between the user and the test system, and runs operating system Windows XP with test development tools. The user must create a test program before running the test on the site controller. The system controller also provides power supplies for the whole T2000 test system. The site controller executes testplans on each device under test (DUT). The bus switch provides connections between the site controller and DUT [9].

2.2 Test head

In the test head, different modules with different functions are available to test DUTs. The system used in this work contains a programmable power supply module (DPS500mA) that provides up to 500mA power supply current for up to 32 channels, a 250MHz digital module (250MDM) that controls up to 128 DUT pins, and a synchronous generator/matrix 16 module (SGM16) [9].

2.2.1 DPS500mA

The main purpose of a Device Power Supply (DPS) is to provide power to a DUT and measure current or voltage at DUT power supply pins.

The DPS500mA module has 32 channels, and each channel of the DPS500mA can either be controlled independently or programmed in parallel with other channels to supply more current to the DUT. As device power requirements vary so much, multiple power supply voltages are required for many DUTs.

Table 2.1 shows that the driving voltage can be any value from -2V to 8V or from -2V to 12V (voltage in range from -2V to 8V has better precision). The resolution of the voltage in the DPS500mA module is 1mV. However, the actual drive voltage accuracy is about $\pm(0.1\%+10mV)$. That means if the 6V is programmed, the voltage of the power supply can vary from 5.984V to 6.016V. Furthermore, the DPS module provides a current source mode that generates up to 500mA for each DPS channel. Maximum measurement memory means 256 measurements can be made and stored when using trace mode. There are two modes for for measurements. One is trace mode that traces every sampling point and display in the Console Window. The other one is average mode that traces every sampling point but only calculates the average value of all sampling values.

Table 2.1: DPS500mA specification [10].

Resource	Value	Resource	Value
Channel	32ch/Module	VSIM measurement	5uA/50uA/500uA/ 5mA/50mA/500mA
Drive voltage range	-2V to 8V / -2V to 12V	VSIM measurement accuracy	5mA range: $\pm(0.20\%8uA2uA/V)$
Drive voltage resolution	1mV	Max Output Current(Ganged)	16A/6.4A
Drive voltage accuracy	$\pm(0.1\% + 10mV)$	Max Meas. Memory - Trace Mode Method	256
Output current range	500mA/200mA	Max number of Vbump supported	4 (2 bits)

The DPS module is used in this work for measuring power supply current in voltage bump tests and IDDQ tests.

Figure 2.1 shows the simplified schematic of a DPS measurement channel. The DPS specifications, circuitry, number of Range-Register, Tester Interface Unit (TIU) interconnections, etc. are different for each DPS model. The DPS channels apply their test conditions on the Force (F) lines. When applying a voltage, the applied voltage is sensed via sense (S) lines, which are used by the DPS channel to properly maintain the desired voltage at the DUT. The current measurements are done by measuring the voltage across the range registers, offering smaller measurement current ranges with better accuracies. In order to achieve the best accuracies, some connections from the DPS Modules to the DUT are especially important [10].

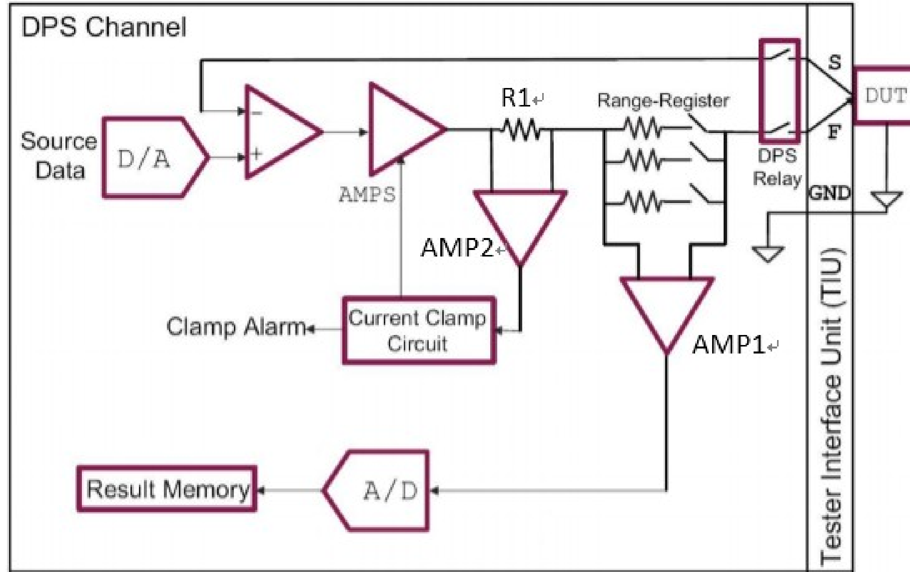


Figure 2.1: Simple diagram of the DPS channel [10].

In a typical Voltage Source Current Measure (VSIM) operation, the engineer programs a voltage source, by applying a binary value, corresponding to the desired voltage, to a DAC. The DAC output is sent through two amplifiers and a resistor, R1, which is used for the current measure. The value of the voltage at the DUT is sensed and fed back to the first amplifier. If it does not match the programmed voltage, adjustments are made to maintain the desired voltage at the DUT. To measure the supplied current, a differential amplifier sends the voltage across the Range-Resistor to an ADC, which measures that voltage. The data is read back as current, based on the voltage value of across the selected Range-Resistor. Different resistors (range registers) can be switched in to create smaller current ranges with greater accuracy. The current is determined by measuring the voltage across them. In addition, the resistor R1 is used to determine whether the current for power supply is outside the programmed current range. If the current from AMP2 is out of the current range, the AMPS will immediately turn off to protect the DUT [10].

For example, in Figure 2.2, the source DAC is programmed to 1V. That goes through the adjusting amps and ideally is 1V. This is amplified to 1.0005V to adjust

for the drop across the Range-Resistor and losses in the overall path to the DUT. 0.0005V is measured across the Range-Resistor and the digital data is stored in result memory. So the power supply current drawn by the DUT is

$$\frac{0.0005\,V}{0.03125\,Ohms} = 16\,mA \tag{2.1}$$

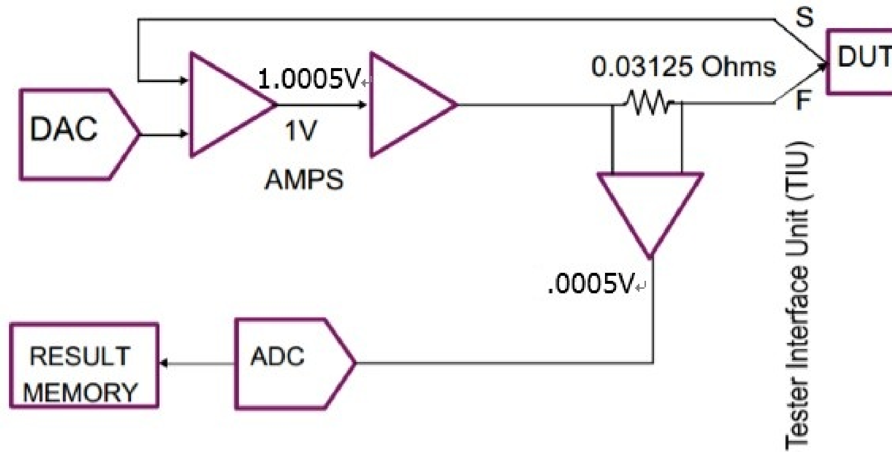


Figure 2.2: The operation of a DPS channel [10].

2.2.2 Module trigger

As shown in Figure 2.3, a module trigger is a resource to control an external module, with trigger signal produced by the Pattern Generator (PG).

In a DPS module, a trigger controls when to take current or voltage measurements on power supply channels during a test. As shown in Figure 2.3, the DPS500mA module has one trigger signal input, which controls all 32 channels in the module by starting and stopping the current measure ADC, described earlier. In the thesis, signal pin DIMTRG is used for controlling the ADC. HIMTRG and LIMTRG are used in other DPS modules.

The module trigger resource name and assignment are specified in the socket and pin description files as shown in Figure 2.4(a) and Figure 2.4(b).As shown in

Figure 2.5, the user programs the timing of a trigger signal with an 8-bit binary value supplied with the SNDT instruction of the PG. Our T2000 250Mbps Digital Module has four triggers, PMDTR0 - PMDTR3 [10].

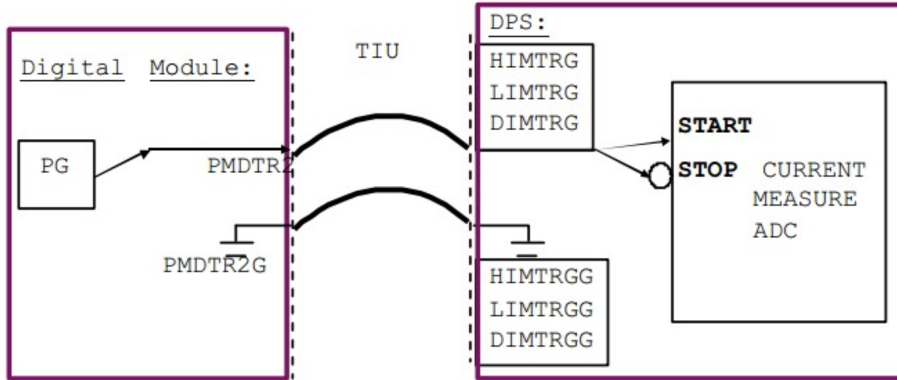


Figure 2.3: Pattern generator controls the measurement ADC [10].

```
Resource moduletrigger
{
PMDTR0;
PMDTR1;
PMDTR2;
PMDTR3;
}
```

(a) Program setup of module trigger in pin description file.

```
Resource moduletrigger
{
PMDTR0 1003.129;
PMDTR1 1003.130;
PMDTR2 2003.131;
PMDTR3 2003.132;
}
```

(b) Program setup of module trigger in socket file.

Figure 2.4: Example of programmed setup for module trigger.

As shown in Figure 2.5, the trigger signal is generated when a particular bit, which is the logical AND of a value assigned to PMDTR and an operand from the SNDT instruction, becomes 1. The data 0x1 is assigned to PMDTR0 when the Test object is created and the SNDT operand data changes to 0x1, 0x2, 0x3, and 0x4 while the pattern executes. For example, as shown in Figure 2.5, we chose PMDTR0 to be 0x1 (00000001 in binary format) in the testplan file and SNDT to be 0x1 (00000001 in binary format) in the pattern file. The logic AND of these two numbers is 00000001, which means the pin Trigger 0 (PMDTR0) is activated.

The example below illustrates why PMDTR0 and PMDTR3 are on for the instruction SNDT 0x1:

0000 0001 (SNDT 0x1) & 0000 0001 (PMDTR0 0x1) = 0000 0001=1

0000 0001 (SNDT 0x1) & 0000 0010 (PMDTR1 0x2) = 0000 0000=0

0000 0001 (SNDT 0x1) & 0000 0100 (PMDTR2 0x4) = 0000 0000=0

0000 0001 (SNDT 0x1) & 0000 0001 (PMDTR3 0x1) = 0000 0001=1

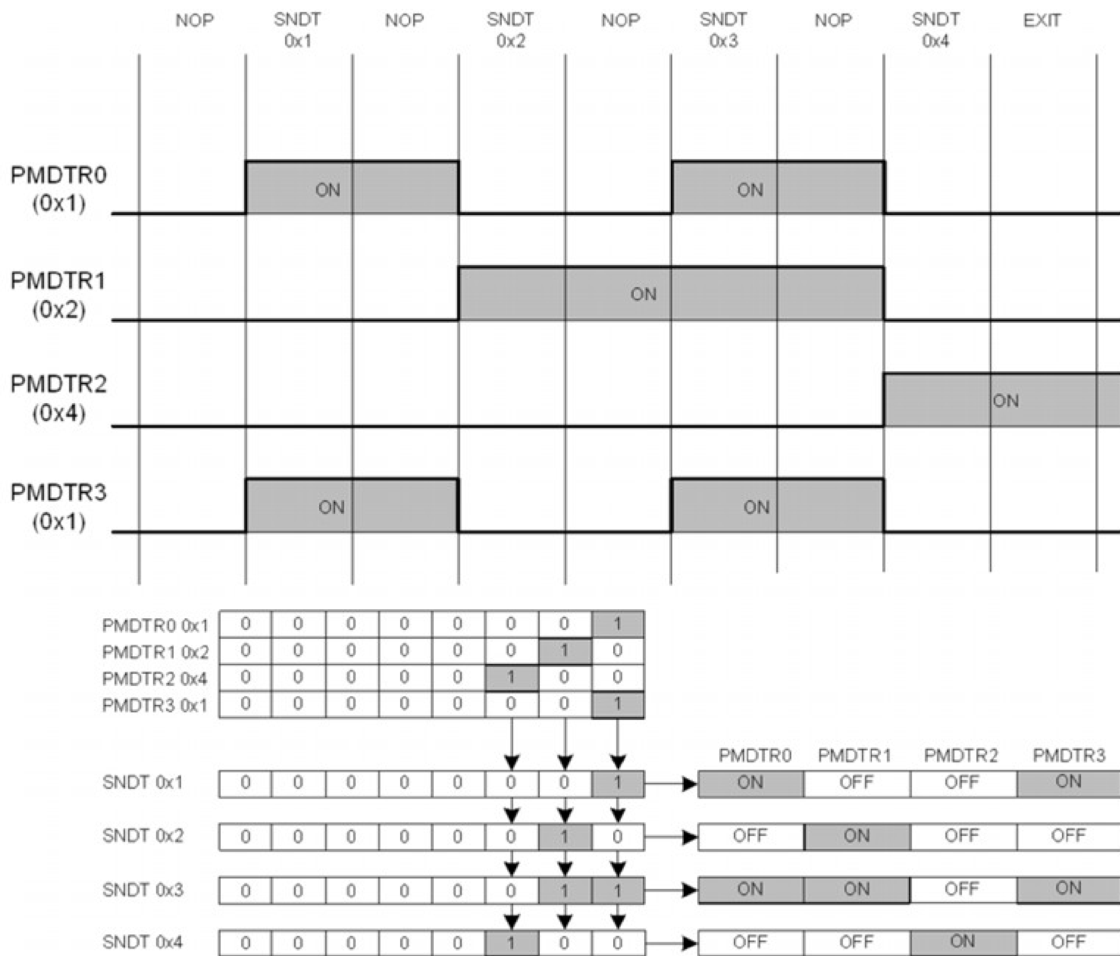


Figure 2.5: Both SNDT instruction in pattern file and PMDTR in testplan file can determine when to activate module trigger signal to high [11].

2.2.3 250MDM

The T2000 128 channel 250Mbps digital module (250MDM) is exclusively designed for the T2000 test system, which is a test system using the OPENSTAR standard. Installing the module in the test head of a T2000 test system allows tests to run with a rate of up to 250Mbps. The module has 128 digital I/O pins. Additional modules can be installed to accommodate higher pin counts of target devices, as well as allowing multiple devices to be tested in parallel [12].

Tables 2.2 and 2.3 list specifications of the driver and comparator, respectively, of the pin electronics section of the 250Mbps digital module.

As indicated in Table 2.2, each channel of the 250MDM can provide VIH value from -1.15V to 7V and VIL from -1.25V to 5.9V. Table 2.3 shows that the output voltage comparison level can be set to -1.25V to 6.75V for both VOH and VOL.

Table 2.2: Pin electronics driver specifications [12].

Item		Value	Item		Value
Voltage range	VIH	-1.15V to +7.0V	Minimum pulse width	50%	4.0ns (at 3V)
	VIL	-1.25V to +5.9V		50%	2.5ns (at 1V)
Voltage amplitude (VIH-VIL)		0.1V to 8.0V	Driver minimum on time	Hi-Z	10ns
Voltage resolution		2mV		VTT	5.0ns
Transition time	20% to 80%	1.2ns (at 3V)	Driver minimum off time	Hi-Z	10ns
	20% to 80%	1ns (at 1V)		VTT	5.0ns

A pin electronics driver is utilized for driving voltages VIL and VIH to input pins of a DUT. A pin electronics comparator is used for setting up comparison levels so the test system is able to judge whether the output voltage of a DUT pin is within the range of VOH and VOL.

Table 2.3: Pin electronics comparator specification [12].

<table>
<tr><th colspan="2">Item</th><th>Value</th><th colspan="2">Item</th><th>Value</th></tr>
<tr><td rowspan="2">Voltage range</td><td>VOH</td><td>-1.25V to +6.75V</td><td rowspan="2">Equivalent transition time</td><td>Hi-Z: 20% to 80%</td><td>2.4ns (at 3V)</td></tr>
<tr><td>VOL</td><td>-1.25V to +6.75V</td><td>VTT: 20% to 80%</td><td>2.0ns (at 1V)</td></tr>
<tr><td colspan="2">Minimum voltage</td><td>0.0V</td><td colspan="2">Window strobe minimum on time</td><td>4.0ns</td></tr>
<tr><td colspan="2">Voltage resolution</td><td>2mV</td><td colspan="2">Window strobe minimum off time</td><td>4.0ns</td></tr>
</table>

In the 250MDM, each of the 128 channels also has a parametric measurement unit (PMU), as shown in the diagram of Figure 2.6, which is similar to that of the DPS500mA shown in Figure 2.1, and therefore the mechanism is identical.

For example, in VSIM mode, the user programs a voltage source value, applied to a DAC. This data is sent through Main Amplifier, Buffer Amplifier and a resistor (this resistor is selected when using different current measurement ranges) to apply a voltage to a DUT pin. The voltage value at the DUT is sensed and fed back to determine if the voltage applied to DUT is identical to the programmed voltage value. The clamp amplifier senses the current flow through the 5 Ohm or 50 Ohm clamp resistor, and outputs current to the main amplifier. The main amplifier can decide whether the programmed voltage is able to continue to force the DUT by comparing a clamp limit to the current value from the clamp amplifier. To measure the DUT input current, the current amplifier (IM Amp) sends the voltage across the range resistor to an ADC, which measures the voltage. The data is read back as current, based on the value of the range resistor.

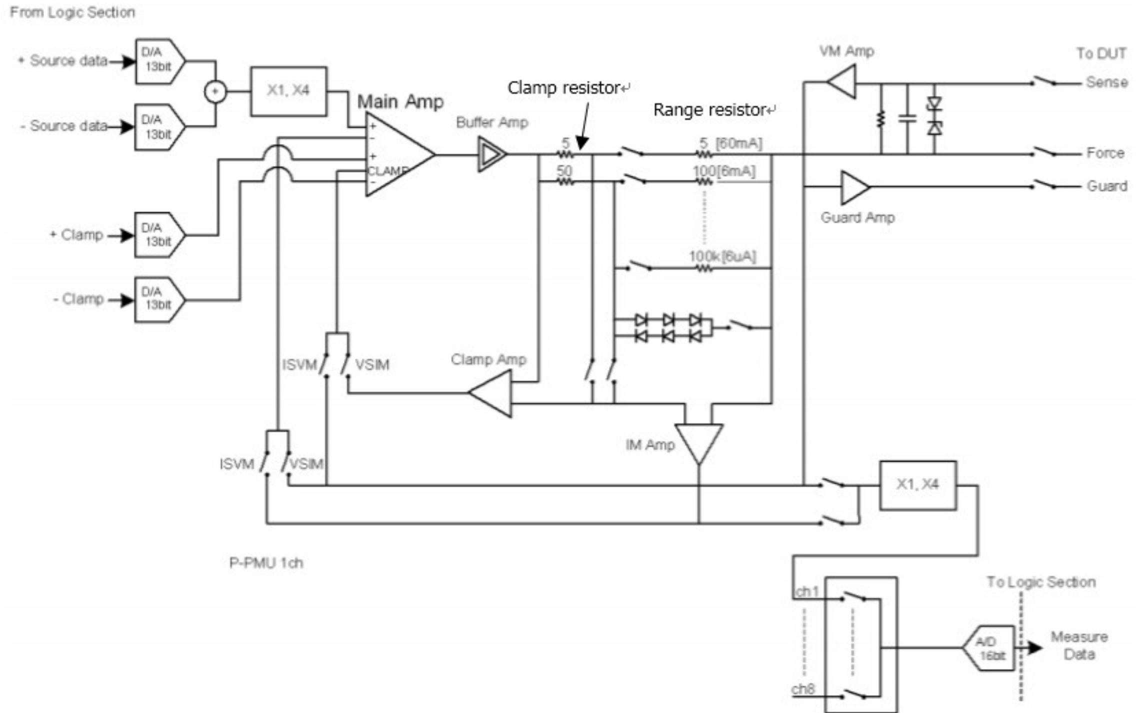


Figure 2.6: The simple schematic of a 250MDM channel, which is also a PMU [11].

2.3 Test program

Test programs are created by a user to perform tests on the ATE. The user must provide three main inputs: a test program, test vectors, and analog test waverforms. Test vectors can be typically generated by an APTG tool, such as Tessent FastScan [26].

The Advantest T2000GS test system uses the Open Architecture Test System Test Programming Language (OTPL). This programming language defines test information for signal pins, timing information, DC signal levels and test patterns. A block diagram of files that make up a test program on the T2000 test system is shown in Figure 2.7. The components of a test program are described in the following paragraphs [13].

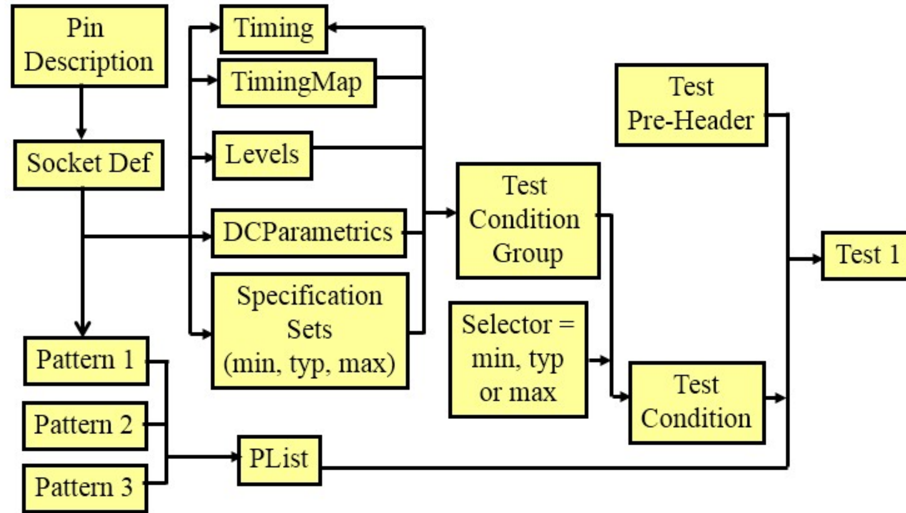


Figure 2.7: Block diagram of test program files on T2000 test system [13].

2.3.1 Pin description file

The pin description file (*.pin) defines the names of the DUT signal and power supply pins. Typically, signal pins are defined with digital pin resource "dpin" and power supply pins are defined with digital power supply 500mA resource "dps500mA". In the pin description file, pins can be grouped together to make them easy to reference in other test program files. Figure 2.8 shows the format of the pin file. A "resource" refers to pins in a specified module. For example, "Resource AT.Digital.dpin" means use the 250DMD module to define DUT signal pins. "Resource dps500mA" means use the DPS500mA module to define power supply pin VDD [13].

```
PinDescription
{
    Resource AT.Digital.dpin
    {
            A1;     # Pin are defined
            CLR1;
            QA1;
            QB1;
            QC1;
            QD1;
            QD2;
            QC2;
            QB2;
            QA2;
            CLR2;
            A2;

        Group inpins    # Group input pins
        {
            A1, A2, CLR1, CLR2
        }
        Group outpins
        {
            QA1, QB1, QC1, QD1, QA2, QB2, QC2, QD2
        }
    }
    Resource dps500mA
    {
        VDD;
    }
```

Figure 2.8: Example of a pin description file.

2.3.2 Socket file

In the socket file, signal pins of a device are mapped to ATE connectors and pins. Figure 2.9 is an example of socket file. In this example, DUT input signal pin A1 is associated with ATE connector "1003.1", which refers to pin 1 on connector 1003. Every time the ATE forces a signal to pin 1003.1, that signal is applied to pin A1 of the device [13].

```
SocketDef
{
    DUTType DiagPB
    {
        PinDescription pindesc.pin;

        DUT 1
        {
            SiteController 1;

            Resource AT.Digital.dpin
            {
                A1      1003.1;
                CLR1    1003.2;
                QA1     1003.3;
                QB1     1003.4;
                QC1     1003.5;
                QD1     1003.6;
                QD2     1003.58;
                QC2     1003.59;
                QB2     1003.60;
                QA2     1003.61;
                CLR2    1003.62;
                A2      1003.63;
            }

            Resource dps500mA
            {
                VDD    2010.27;
            }
```

Figure 2.9: Example of a socket file.

2.3.3 Specification file

This file defines device specifications such as power supply voltage, input threshold voltage, output driving voltage, slew rate and timing related parameters. In this file, multiple values can be specified for each parameter, such as min, typical and max values. In addition, the unit of each parameter is clearly stated, so it can be directly used in other files. The parameters of a device are acquired from its datasheet. Figure 2.10 is example of a specification file, using specification data of the 74LS393 IC (4-bit binary counter). For a particular test, one set of values will be selected (ex. all min values) in levels, test condition and testplan files. Also, the variables defined in the specification file can be used in other files [13].

```
Import uservar.usrv;

SpecificationSet DiagPBSpec(min, typ, max)
{

        Voltage vforce = 4.75V,5V,5.25V;

        Current ich     = 20mA, 100mA, 200mA;

        Current icl = -400mA, -1600mA, -2400mA;

        VoltageSlew slewrate= 78.125;

        Voltage vih = 2V;

        Voltage vil = 0V;

        Voltage voh = 2.5V, 3.4V, 3.4V;

        Voltage vol = 0.35V, 0.35V, 0.5V;

        #                       min             typ             max
        Time tStrbH     =       50nS,           50nS,           50nS;
        Time tStrbL     =       50nS,           50nS,           50nS;
}
```

Figure 2.10: Example of a specification file.

2.3.4 Levels file

This file is able to define what kind of signal should be associated with various pin resources. It defines voltage and current levels at each pin or pin group. For instance, as shown in Tables 2.2 and 2.3, since VIH range is defined from -1.15V to 7V in the pin electronics specification, the VIH values in the levels file cannot exceed this range. In addition, it also states what testing mode will be applied during the test. For instance, the levels file can indicate whether current source voltage measurement (ISVM) or voltage source current measurement (VISM) is to be used in the test. Figure 2.11 is an example of a levels file. In this levels file, the variables, such as vforce, vih, vil, voh and vol, are assumed to have been defined in the specification file [13].

```
Levels Lvl1
{
        VDD
        {
                VSRange = 7V;   # Voltage supply range
                VForce = vforce;# Voltage value, it can either define in specification file or level file
                DpsRelay = CLOSE;# Close the relay between DUT and test head
                PowerSequence = ON;
        }
        Delay 3mS;

        inpins
        {
                VIH = vih; #specification file defined the variable vih, vil, voh and vol
                VIL = vil;
                PinOutRelay = CLOSE;
                PowerSequence = ON;
        }


        outpins
        {
                VOH = voh;
                VOL = vol;
                PinOutRelay = CLOSE;
                PowerSequence = ON;
        }
}
```

Figure 2.11: Example of a levels file.

2.3.5 Timing and timing map files

The timing file defines test vector periods and waveforms (signal transition times) for each pin or pin group. A timing file can define up to four different periods and eight waveforms. The values defined in a pattern file should be applied in accordance with the timing file. In a pattern file, the waveform of a vector can be directly defined using the variable defined in timing map file [13].

Figure 2.12, an example of a timing file, shows waveform and period tables. In this example, the period table defines four test period values, one waveform table define input pin in timing, and four different waveform tables define output pin sampling times.

Figure 2.13, an example of a timing map file, that defines waveform sets, each of which associates a period with a specified set of waveform tables. In this example, the timing map file defines four different waveform sets; the detailed waveform parameters of seq1, seq2, seq3 and seq4 are defined in the timing file shown in Figure 2.12.

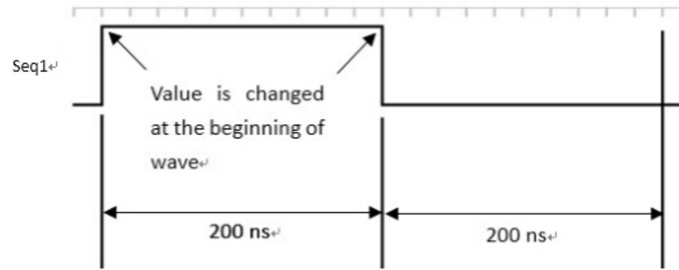
```
Timing Tim_200_to_190
{
        CommonSection
        {
                Domain default
                {
                        PeriodTable
                        {
                                Period per0 { 200nS; }
                                Period per1 { 197.5nS; }
                                Period per2 { 195nS; }
                                Period per3 { 192.5nS; }
                        }
                        Pin inpins
                        {
                                WaveformTable seq1
                                {
                                        { 1 { U@0nS,E1; } }
                                        { 0 { D@0nS,E1; } }
                                }
                                WaveformTable seq2
                                {
                                        { 1 { U@40nS,E1;D@140nS,E1 } }
                                        { 0 { D@40nS,E1 } }

                                }
                        }
                        Pin outpins
                        {
                                WaveformTable seq1
                                {
                                        { H { H@199.5nS,E5; } }
                                        { L { L@199.5nS,E6; } }
                                }
                                WaveformTable seq2
                                {
                                        { H { H@197nS,E5; } }
                                        { L { L@197nS,E6; } }
                                }
                                WaveformTable seq3
                                {
                                        { H { H@194.5nS,E5; } }
                                        { L { L@194.5nS,E6; } }
                                }
                                WaveformTable seq4
                                {
                                        { H { H@192nS,E5; } }
                                        { L { L@192nS,E6; } }
                                }
                        }
                }
        }
}
```

Figure 2.12: Example of a timing file.

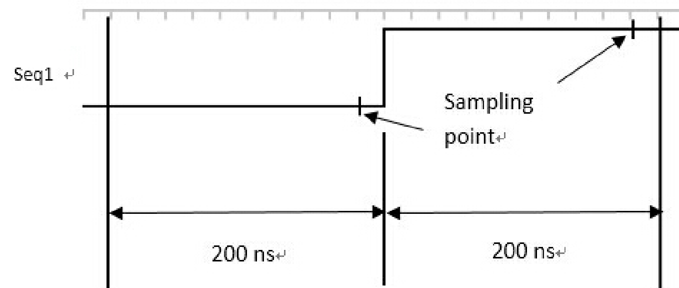
```
TimingMap TMap1
{
        Domain default
        {
                WaveformMap
                {
                        PinFormat { inpins, outpins }
                        wfs1, per0, { seq1, seq1 }
                        wfs2, per1, { seq1, seq2 }
                        wfs3, per2, { seq1, seq3 }
                        wfs4, per3, { seq1, seq4 }
                }
        }
}
```

Figure 2.13: Example of a timing map file.

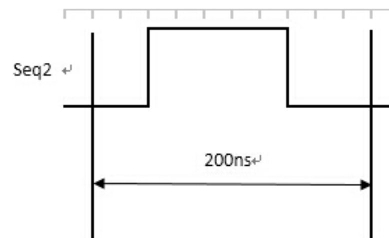
Figure 2.14 is an example that shows what the timing file and map represent. In this example, waveform set wfs1 of the timing map of Figure 2.13 is used, which has a test period per0, defined as 200ns in the timing file of Figure 2.12. Figure 2.14(a) shows an input signal to the DUT. The waveform seq1 is defined in timing map file shown in Figure 2.13, which inputs applied at the beginning of the test period. Figure 2.14(b) shows the output sampling point of 199.5 ns for each period, corresponding to waveform table seq1. The tester will do a comparison between the preset VOH/VOL and the actual voltage value at the sampling point. Figure 2.14(c) shows that input voltage levels can also change within a period, to create clocks and related signal.



(a) The waveform to an input of DUT.



(b) The waveform of an output of DUT.



(c) Example of an input.

Figure 2.14: Example of waveform.

2.3.6 Test condition file

This file defines what levels of voltage or current from the specification and which timing parameters should be chosen for a given test. Each test is defined with a specific set of test conditions. Figure 2.15 is an example of a test condition file. In this example, the specification set DiagPbSpec shown in Figure 2.10, the levels Lvl1 shown Figure 2.11, the timing Tim_200_to_190 shown in Figure 2.14, and the

timing map TMap1 shown in Figure 2.15 are combined to create this test condition group [13].

```
TestConditionGroup DiagPBTCG_200_to_190
{
        SpecificationSet DiagPBSpec;
        Levels Lvl1;
        Timings
        {
                Timing = Tim_200_to_190;
                TimingMap = TMap1;
        }
}
```

Figure 2.15: Example of a test condition file.

2.3.7 Pattern file

In a pattern file, input test vectors and expected outputs are defined to verify circuit functions in each test period. In addition, the system can automatically produce different types of patterns, such as by algorithmic pattern generation (ALPG) and SCAN pattern generation. Users can also specify instructions for each test vector to control test pattern application. Figure 2.16 is an example of a pattern file. In this example, the timing of waveform set wfs1 is applied to the vectors, as defined in the timing file and the timing map file shown in Figure 2.12 and Figure 2.13, respectively. In this pattern file, the variables inpins and outpins are defined in the pin description file shown in Figure 2.8. In this example, the first input vector is 0111, which applies 0 to pin A1, 1 to pin A2, 1 to pin CLR1, and 1 to pin CLR2. The output vector LLLLLLLL means expecting a low voltage level at all of the outputs when applying this pattern. [13].

```
NOP  {  V { {inpins=0111; outpins=LLLLLLLL;} # V indicates voltage levels
                              {inpins=1100; outpins=LLLLLLLL;}
                              {inpins=0000; outpins=HLLLHLLL;}
                              {inpins=1100; outpins=HLLLHLLL;}
                              {inpins=0000; outpins=LHLLLHLL;} W{allpins=wfs1;} }
```

Figure 2.16: Example of a pattern file.

2.3.8 Testplan file

The testplan file determines which test conditions and specifications should be combined in a specific test. It also defines test organization and what type of test class, such as DC parametric test and power supply test, to use during each step of the test. Figure 2.17 is an example of a test plan file. In the example, the test initializes the DatalogSetup block, and then performs the function test of the DiagPBFunctionalTest_200_to_190. If DiagPBFunctionalTest_200_to_190 block passes the test, it will proceed to block DiagPBFunctionalTest_190_to_180. Otherwise, it will proceed to 1 which means this test is failed. Each block in a DUTFlow is a DUT flow item, as shown in Figure 2.17. To facilitate testing, the built-in tool Flow Editor GUI shows each DUT flow item as a block of a test plan, as illustrated in Figure 2.18. The user can edit the test plan via the GUI, and observe the execution flow of a test [13].

```
TestCondition TC_200_to_190
{
        TestConditionGroup = DiagPBTCG_200_to_190;
        Selector = typ;
}
}
Test FunctionalTest DiagPBFunctionalTest_200_to_190
{
        PListParam = DiagPBPat;
        TestConditionParam = TC_200_to_190;
}
Test FunctionalTest DiagPBFunctionalTest_190_to_180
{
        PListParam = DiagPBPat;
        TestConditionParam = TC_200_to_190;
}
DUTFlow FlowMain
{
        DUTFlowItem DatalogSetupFlow DatalogSetup
        {
                Result 0
                {
                        GoTo FlowMain_200_to_190;
                }
        }
        DUTFlowItem FlowMain_200_to_190 DiagPBFunctionalTest_200_to_190
        {
                Result 0
                {
                        GoTo FlowMain_190_to_180;
                }

                Result 1
                {
                        Return 1;
                }
        }
        DUTFlowItem FlowMain_190_to_180 DiagPBFunctionalTest_190_to_180
        {
                Result 0
                {
                        Return 0;
                }

                Result 1
                {
                        Return 1;
                }
        }
```

Figure 2.17: Example of a testplan file.

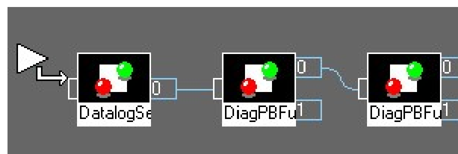


Figure 2.18: Example of the test flow item that shows in Flow Editor GUI.

Chapter 3

DC Parametric Test

After integrated circuit device fabrication, DC parametric tests ensure that various electrical parameters of a circuit device are within its specifications. Generally speaking, DC parametric testing includes [6]:

1. Contact test (Open and Short)
2. Input leakage current test (IIL, IIH)
3. Input threshold voltage test (VIL, VIH)
4. Output driving voltage test (VOL, VOH)
5. Power consumption test (Gross, Static)
6. Power supply bump test

The first four tests use the PMU of the 250MDMA, and the fifth and sixth tests employ the DPS500mA (DPS) module.

In some DC parametric tests, before performing the test, a device needs to be preset to some specific state. For example, if IIL is the parameter to be tested, the power supply of this device should be preconditioned to a maximum level. When using the PMU, some settling time is required before taking a measurement on the device [14].

In the experiments performed for this thesis, two standard digital functions, each implemented in three different technologies, are utilized to determine differences between those chips. The chips used in the test include 74HC393, 74HCT393, 74LS393, 74HC163, 74HCT163 and 74LS163. Function 74393 is a 4-bit binary ripple counter [16] and the 74163 is a 4-bit synchronous presettable binary counter [15]. Here, 74HC means using CMOS technology. Chips using CMOS technology have

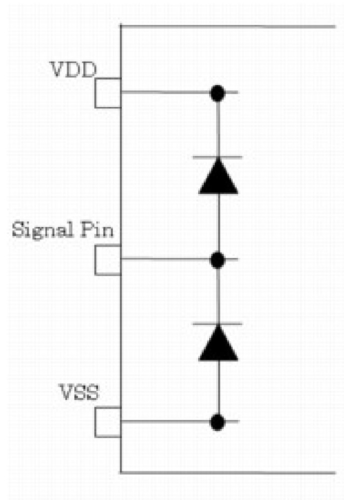
merits such as working over a wider range of voltages with lower power consumption than 74LS series chips. 74LS series chips, made with Schottky transistors, have advantages like reducing the stored charge and operating at higher frequency than 74HC series chips. 74HCT series chips mainly use CMOS technology but with TTL-compatible inputs and outputs to make HCT chip voltage levels compatible with LS series and other TTLchips.

3.1 Contact test

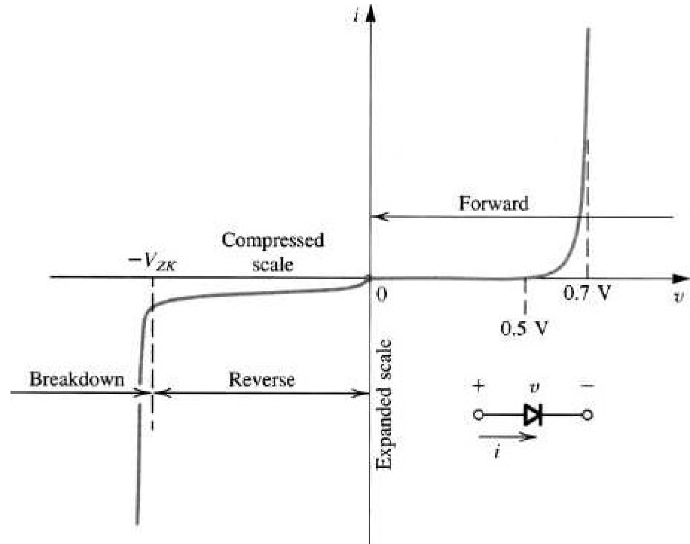
Open and short tests are also named continuity tests, and mainly check two problems. One is device pins open and the other is shorted device pins. The main causes of opens and shorts are physical defects occurring in the fabrication of die or the packaging of die. The defects include missing, broken, or incomplete bond wires, shorted metal pins and pins damaged by static electricity. This test is typically done first because there is a possibility that the electrical contacts between the ATE and the DUT or between wafer and wafer probe card are defective. Open and short tests are simple tests that can be finished rapidly to decrease the average time of testing a faulty device by identifying faulty devices before applying more costly tests. In this test, all input and output signal pins are forced to a fixed current to measure voltage between these pins. If the voltage is out of range, we can treat this device as faulty [6] [7].

3.1.1 Test method

Generally speaking, each device pin is usually connected to protective diodes as shown in Figure 3.1(a). If a forward bias current is applied to either of these diodes, a voltage drop occurs between the two terminals of the diode, which is illustrated in the V-I characteristic in Figure 3.1(b). By using this V-I characteristic, a signal pin can be checked to see whether the connection is defective.



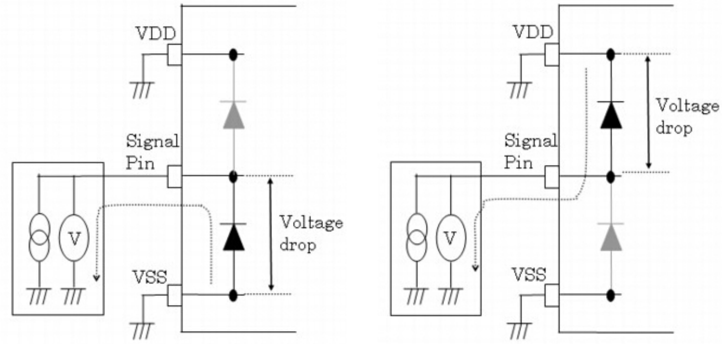
(a) Protective diode [7].



(b) V-I characteristic plot of diode [3].

Figure 3.1: The protective diode of DUT.

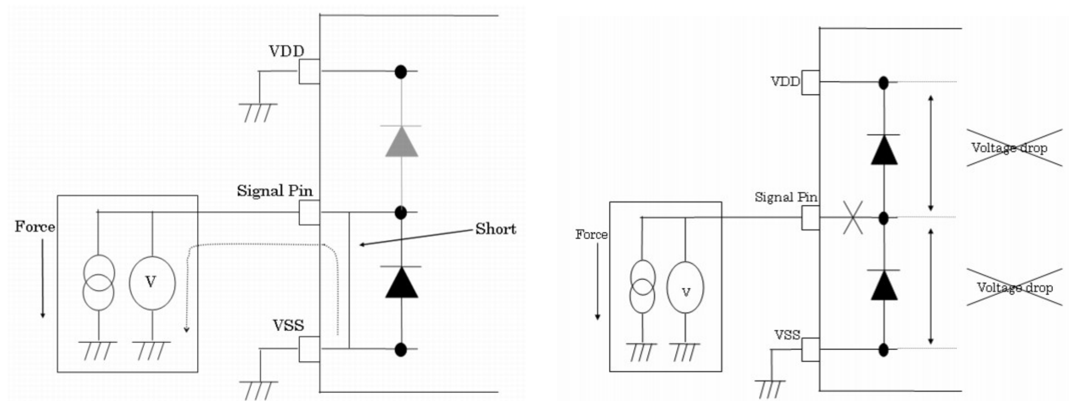
To test open and short faults, the GND level (0V) should be applied on all VDD, GND and signal pins, except for target pins, as shown in Figure 3.2. This is used to check whether signal pins are open circuited or short circuited to other signal pins or power supply pins. A fixed current is applied to each device signal pin from the build-in PMU to measure the electric potential (voltage) across each signal pin. If the signal pins are correctly connected without opens and shorts, as shown in Figure 3.2, then if a positive current is applied through signal pins, the current will flow through the diode and back to the current source to form a voltage drop across diode. By using ISVM, this voltage drop can be measured by the PMU and compared to the expected value.



(a) Forward bias the diode [7]. (b) Reversed bias the diode. [7]

Figure 3.2: Contact test setup diagram.

However, if the target pin is shorted to other device pins or GND, as shown in Figure 3.3(a), the measured voltage is the same electric potential of 0V. If a disconnection, as shown in Figure 3.3(b), is inside of the device or the device interface, then the applied current flows through an infinite resistance; this measured voltage will reach a clamp voltage level, causing this test to fail.



(a) Short circuited between signal pin and ground. (b) Pin open.

Figure 3.3: Contact test defect diagram.

3.1.2 Test procedure

The steps listed below are applied to all six test chips.

1. Apply 0 V to all power supply pins in the levels file.

2. Apply 0 V to all signal pins in the levels file.

3. Apply a forward bias current to the target signal pins in the levels file to measure voltage on each signal pin, one by one.

4. In the testplan file, set up the PMU module, serial testing mode and settling time for testing.

5. Load and run the test plan and view results in the Console Window.

6. If the measured voltage is larger than -0.1 V, the signal pin is considered short-circuited.

7. If the measured voltage is smaller than -1.0 V, the signal pin is open.

8. If the measured voltage is within the range of -0.1 V to -1.0 V, the DUT pin is correctly connected to the device interface and within the DUT.

3.1.3 Test program

The statements of the program files used in the contact test are shown in Figure 3.4 and Figure 3.5. In order to measure the voltage of the signal pins, -200 uA current is applied to each pin to measure the threshold voltage.

```
DCParametrics hc393_open
{
        allpins
        {
                OPMode                  = ISVM; # current source voltage measurement
                ISRange                 = 600uA; # current source range
                IForce                  = -200uA;# applying -200uA to each pin serially
                VMRange                 = 2V;   # voltage measurement range
                VClampHi                = 1V;  # high clamp voltage
                VClampLo                = -2V; # low clamp voltage
                JudgeUpperVoltageVal    = -0.1V; # high failed voltage
                JudgeLowerVoltageVal    = -1V; # low failed voltage
        }
}
```

(a) Setting up testing mode and relevant parameter.

```
Levels hc393_open_levels
{
        VDD{
          VSRange       = 6V;  # voltage source range to pin "VDD"
          VForce        = 0V; # connecting ground to pin "VDD"
          DpsRelay      = CLOSE; #Close the relay between DUT and test head
          PowerSequence  = ON;

        }
        allpins
        {
          PinOutRelay    = CLOSE;
          FIXL           = 0V;  # applying fixed 0 V to all signal pins
          PowerSequence  = ON;

        }
        Delay 3mS;
}
```

(b) Setting up voltage levels to VDD and signal pin.

Figure 3.4: Program setup of the levels file.

```
Test DCParametricPMUTest hc393_open_test {
     TestConditionParam  = hc393_open_TC;
     TestConditionParam  = hc393_open_PMU_TC;
     MeasurementPin      = "allpins"; # measure all signal pins
     PMURelayControl     = "On";     # turn on PMU
     MeasMode            = "Serial";  # measure each pin on by one
     SettlingTimePreMeasure   = "1mS"; # wait time after the power is turned on
     SettlingTimePostMeasure  = "1mS"; # wait time after the power is turned off
}
```

Figure 3.5: Program setup of the testplan file.

3.1.4 Test result

Figure 3.6 shows the measured voltages when -200uA current is applied to each signal pin of device 74HC393; the unit of each value is V. The result indicates all pins are operational within the limit (from -1V to -0.1V).

```
DUT ID 1:DCParametric test hc393_open_test execution succeeded.
        PinName          Value        Result    Status
        A1               -0.724       Pass      Valid
        A2               -0.726       Pass      Valid
        CLR1             -0.724       Pass      Valid
        CLR2             -0.726       Pass      Valid
        QA1              -0.514       Pass      Valid
        QB1              -0.512       Pass      Valid
        QC1              -0.514       Pass      Valid
        QD1              -0.514       Pass      Valid
        QA2              -0.51        Pass      Valid
        QB2              -0.51        Pass      Valid
        QC2              -0.512       Pass      Valid
        QD2              -0.512       Pass      Valid
```

Figure 3.6: Voltage result of contact test on device 74HC393 captured from the Console Window.

The results shown in Table 3.1 and Table 3.2 show that all six chips pass the contact test. By analyzing the data, it can be seen that measured pin voltages of LS chips are higher than HC and HCT chips. For each chip, test voltages of inputs are almost identical between HC and HCT devices. In addition, voltages of inputs on LS devices are much higher than that for HC and HCT devices. Output voltages are almost nearly the same on all of the devices.

Table 3.1: Testing result of 74393 chips (Unit for all values is V).

		74HC393	73HCT393	74LS393
Inputs	A1	-0.724	-0.716	-0.558
	A2	-0.726	-0.718	-0.558
	CLR1	-0.724	-0.716	-0.56
	CLR2	-0.726	-0.716	-0.558
Outpus	QA1	-0.514	-0.582	-0.608
	QB1	-0.512	-0.578	-0.606
	QC1	-0.514	-0.584	-0.604
	QD1	-0.514	-0.588	-0.61
	QA2	-0.51	-0.586	-0.608
	QB2	-0.51	-0.586	-0.606
	QC2	-0.512	-0.586	-0.602
	QD2	-0.512	-0.588	-0.608

Table 3.2: Testing result of 74163 chips (Unit for all values is V).

		74HC163	73HCT163	74LS163
Inputs	A	-0.688	-0.724	-0.432
	B	-0.688	-0.724	-0.43
	C	-0.69	-0.724	-0.43
	D	-0.688	-0.724	-0.43
	CLR	-0.69	-0.724	-0.432
	CLK	-0.69	-0.724	-0.45
	ENP	-0.688	-0.724	-0.43
	ENT	-0.688	-0.722	-0.43
	LOAD	-0.688	-0.724	-0.43
Outputs	RCO	-0.556	-0.586	-0.61
	QA	-0.556	-0.604	-0.562
	QB	-0.554	-0.604	-0.57
	QC	-0.55	-0.59	-0.578
	QD	-0.552	-0.59	-0.592

3.2 Input leakage current test

Input Leakage Current Testing is a voltage source current measurement (VSIM) test that checks whether leakage current of all input signal pins is within the design specifications. Since the ideal input pin has infinite resistance, the input pins do not sink or source more than the current stated in the specification. Based on the measured current, resistance from input pins to VDD and GND can be measured, and some defects in the device can be detected [7].

High leakage current on a signal pin is usually caused by defects of fabrication. Sometimes, static electricity may also lead to damage of a signal pin. Because of the

physical defects, a low resistive path has possibility to be formed between signal pins or between signal pins and power supply pins. If a faulty device has a low resistance between a targeted pin and another pin under test, high leakage current will be formed because of the voltage drop across this low resistance path [6].

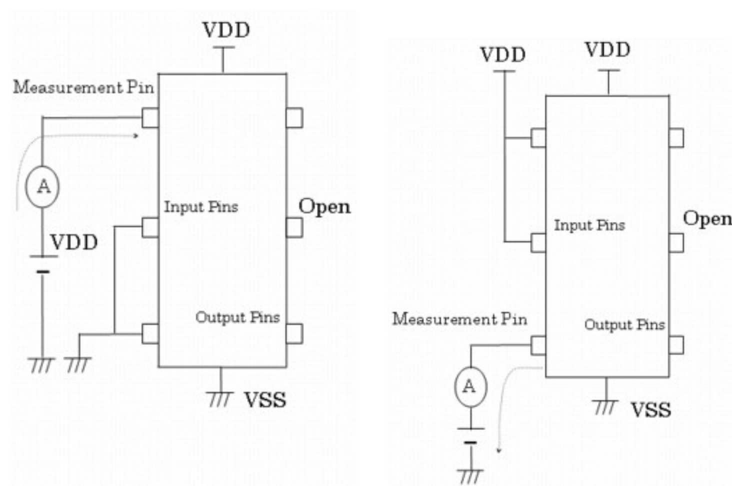
This test is typically performed after Contact Test.

IIH

When doing IIH test, voltage VDD, the same as the power supply, is applied to the target pin and GND voltage to non-target pins. All output pins are left open (the relays from the DUT to the tester are open). The PMU will apply a voltage to the target pin and measure the current flow through it. Figure 3.7(a) illustrates IIH test setup [7].

IIL

IIL is tested as shown in Figure 3.7(b). Testing IIL is similar to IIH. GND is applied to the target pin and VDD to non-target pins. All of the output pins are again kept opened [7].



(a) IIH test setup. [7]   (b) IIL test setup. [7]

Figure 3.7: Input leakage current test.

Serial and Parallel method

In testing IIL and IIH, there are two testing approaches. One is serial and the other is parallel. As the name implies, in serial mode, the input pins, output pins and power supply pins are configured to the settings shown in Figure 3.7. Pins are tested serially by repeatedly implementing the same test on each individual target pin.

Parallel measurement is performed by driving low simultaneously all input pins, to take measurements of IIL. In the 250MDM, a PMU is provided for each pin of the tester. Parallel test is performed faster than in serial mode. However, since it gives all the input pins the same voltage level, leakage currents between input pins cannot be detected [7].

3.2.1 Test procedure

Generally, the IIH and IIL test procedure can be summarized as follows.

IIH test procedure

1. Apply the maximum power supply voltage to the power supply pin in the levels file.

2. Apply the same voltage as that of GND to the non-target input signal pins in the levels file.

3. Set up all output pins to be opened to the tester in the levels file.

4. In the level file, apply the same voltage as that of the power supply to the target pin.

5. In the testplan file, set up PMU module, serial testing mode and settling time for testing.

6. Measure the current value which flows into the target pin. Test results are displayed in the Console Window.

7. Determine whether the measured current value is within the specification of the device.

IIL test procedure

1. Apply the maximum power supply voltage to the power supply pin in the levels file.

2. Apply the same voltage as that of the power supply to the non-target input signal pins in the levels file.

3. Set up all output pins to be opened to the tester in the levels file.

4. In the levels file, apply the same voltage as that of GND to the target pin.

5. In the testplan file, set up the PMU module, serial testing mode and settling time for testing.

6. Measure the current value which flows into the target pin. Test results are displayed in the Console Window.

7. Determine whether the measured current value is within the specification of the device.

3.2.2 Test Program

Figure 3.8 and Figure 3.9 are test program examples for testing input leakage current.

```
DCParametrics Leakage_I
{
        inpins1
        {
                OPMode                  = VSIM; #Voltage source current measurement
                VSRange                 = 7V;  #Voltage source range
                VForce                  = vin; # The applied voltage to inputs, vin is 0V defined
                                                # in specification file
                IMRange                 = 1mA;  # Current measurement ranage
                IClampHi                = 1mA;  # High clamp current
                IClampLo                = -1mA; # Low clamp current
                JudgeUpperCurrentVal    = 1mA;  # High failed current
                JudgeLowerCurrentVal    = -1mA; # Low failed current
        }
```

(a) Program setup for VDD, non-target input pins and output pins.

```
Levels ls393_DCLeakage_IIL_levels
{
        VDD{
                VSRange         = 7V;   # Voltage source range
                VForce          = vcc_apply; # Apply VDD to power supply voltage
                DpsRelay        = CLOSE;
                PowerSequence = ON;
        }

        Delay 3mS;

        inpins
        {
                FIXH    = vih;   # Apply input signal pins to vih (same voltage as VDD)
                PowerSequence =ON;
                PinOutRelay = CLOSE;
        }
        outpins
        {
                PowerSequence =OFF; # Leave output signal pins open
                PinOutRelay = OPEN;
        }
        Delay 3mS;
}
```

(b) Program setup for target input pins.

Figure 3.8: Program setup of the levels file for IIL test.

In Figure 3.9, the statement "MeasurementPin = inpins" indicates that all input pins are to be tested. The statement "MeasMode = Serial" means that serial measurement mode is to be used.

```
Test DCParametricPMUTest DCLeakage_low {
     TestConditionParam  = DC_IIL_levels;
     TestConditionParam  = DC_IIL_params;
     MeasurementPin      = "inpins";  # only measure inputs
     PMURelayControl     = "On";
     MeasMode            = "Serial"; # measurement mode is serial
     SettlingTimePreMeasure   = "3mS";
     SettlingTimePostMeasure  = "0.5mS";
     ExecEndCondition         = 1;
}

Test DCParametricPMUTest DCLeakage_high {
     TestConditionParam  = DC_IIH_levels;
     TestConditionParam  = DC_IIH_params;
     MeasurementPin      = "inpins";
     PMURelayControl     = "On";
     MeasMode            = "Serial";
     SettlingTimePreMeasure   = "3mS";
     SettlingTimePostMeasure  = "0.5mS";
     ExecEndCondition         = 1;
}
```

Figure 3.9: Program setup of the testplan file for IIL and IIH test.

3.2.3 Test result

Figure 3.10 is the result of IIL test for the 74HC393 using serial mode. The unit of value is A. Since minimum current test range is 5uA with 0.01uA precision, input leakage current is so small that it is unable to be measured accurately. In the specification, input leakage current is 0.1uA maximum. Therefore in testing results, 1e-8 is equivalent to 0.01uA and 2e-8 is 0.02uA, which are smaller than the maximum value. Based on the results, this device passes IIL testing.

```
DUT ID 1:DCParametric test DCLeakage_low execution succeeded.
     PinName         Value       Result    Status
     A1              -1e-008     Pass      Valid
     A2              -1e-008     Pass      Valid
     CLR1            -1e-008     Pass      Valid
     CLR2            -2e-008     Pass      Valid
```

Figure 3.10: The test result of IIL test for 74HC393 captured from the Console Window.

Figure 3.11 is the result of IIH test for 74HC393 using serial mode. The results are also within specification, which means this device passes the test of IIH.

```
DUT ID 1:DCParametric test DCLeakage_high execution succeeded.
    PinName         Value       Result     Status
    A1              1e-008      Pass       Valid
    A2              -1e-008     Pass       Valid
    CLR1            2e-008      Pass       Valid
    CLR2            -2e-008     Pass       Valid
```

Figure 3.11: The test result of IIH test for 74HC393 captured from the Console Window.

According to Table 3.3 and Table 3.4, all HC and HCT device pins have very small input leakage currents compared with LS devices.

Table 3.3: Test result of 74393 chips (Unit for all values is A).

		74HC393	74HCT393	74LS393
IIH				
Inputs	A1	1e-008	2e-008	-0.000128
	A2	-1e-008	-1e-008	-0.000144
	CLR1	2e-008	2e-008	-1.6e-005
	CLR2	-2e-008	-3e-008	-1.6e-005
IIL				
Inputs	A1	-1e-008	-1e-008	-0.000352
	A2	-1e-008	-1e-008	-0.000336
	CLR1	-1e-008	-1e-008	-0.000208
	CLR2	-2e-008	-2e-008	-0.00024

Table 3.4: Test result of 74163 chips (Unit for all values is A)

		74HC163	74HCT163	74LS163
IIH				
Inputs	A	-5e-008	-5e-008	-1.6e-005
	B	-2e-008	-2e-008	-1.6e-005
	C	-6e-008	-6e-008	-1.6e-005
	D	-4e-008	-4e-008	-1.6e-005
	CLR	1e-008	1e-008	1.6e-005
	CLK	1e-008	2e-008	-1.6e-005
	ENP	1e-008	2e-008	-4.8e-005
	ENT	-6e-008	-6e-008	-1.6e-005
	LOAD	1e-008	1e-008	-3.2e-005
IIL				
Inputs	A	-2e-008	-2e-008	-0.000144
	B	-2e-008	-2e-008	-0.000128
	C	-1e-008	-1e-008	-0.000112
	D	-1e-008	-1e-008	-0.000128
	CLR	-1e-008	-2e-008	-0.00016
	CLK	-1e-008	-1e-008	-1.6e-005
	ENP	-1e-008	-2e-008	-0.000144
	ENT	-2e-008	-1e-008	-0.000144
	LOAD	-2e-008	-2e-008	-0.00016

3.3 Input threshold voltage test

The input voltage test checks whether a device operates properly for input voltage values in the device electrical specification. The input threshold voltage can be usually

classified by two levels: VIH is high level input voltage threshold and VIL is low level input voltage threshold [6].

When using the T2000 test system, there are two types of input voltage tests. One can perform a GO/NOGO test by performing functional test and looking for output errors if VIH/VIL are out of range. The other uses the margin search method to obtain the exact VIH or VIL level.

3.3.1 Margin search

A margin search test tries different input voltages to test if functions are correct for these voltage levels. The input voltage test using the margin search method obtains the input voltage threshold value while performing functional tests to search for the pass/fail transition point by changing the input voltage level over a range of values. The margin search method is mainly used for detecting the maximum and minimum value of each parameters [7].

In the test, output voltage levels are the same as the GO/NOGO test. The input voltage (VIH or VIL) that is to be tested is varied over a range of values.

3.3.2 Test procedure

The input voltage test using the margin search method is performed according to the following procedure.

1. Apply power supply voltage in the levels file.

3. Set an input signal level in the levels file.

4. Set output level conditions in the levels file.

5. Set signal timing in the timing file.

6. Set up test vectors in the pattern file.

7. Load test patterns and set up the margin search test parameter in the test plan file.

8. Execute the test patterns.

9. Check the test pattern execution results in the Console Window.

10. Change the input signal levels.

11. Return to step 7 if input signal level is lower than the stop value.

12. Determine whether the input signal level is within the specification of the device.

13. Search will be terminated after the last test value has been performed.

3.3.3 Test Program

Figure 3.12 is an example of a levels file for testing the 74HC393. In this program, normal values of VIH and VIL are assigned.

```
Levels AllPins_Levels
{
        inpins
        {

                VIH = vih;      # vih and vil are defined in specification file
                VIL = vil;      # the values of vih and vil are 3.15 V and 1.35 V respectively
                PinOutRelay = CLOSE;
                PowerSequence = ON;
        }
        outpins
        {
                VOH = voh;      # the values of voh and vol are 4.32 V and 0.15 V respectively
                VOL = vol;
                PinOutRelay = CLOSE;
                PowerSequence = ON;
        }
        Delay 1mS;
}
```

Figure 3.12: Program setup of the levels file for VIH test.

In the testplan shown in Figure 3.13, the Algorithm is the "Margin", which means searching VIH from lower voltage to higher voltage step by step. In the next line, for "SearchVarParam", vih represents the variable we want to change, 1.5 V is start value, 0.01 V is the increment of variable at each test, 100 is the number of tests we wish to perform.

```
Test SearchTest Search_InputVoltageTest_High
{
    PListParam          = DiagPBPat;
    TestConditionParam  = InputVoltage_AllDps_Levels_Typ_TC;
    TestConditionParam  = InputVoltage_AllPins_Levels_Typ_TC;
    TestConditionParam  = FunctionalTest_Timing_Typ_TC ;

    Algorithm      = "Margin";
    SearchVarParam = "x,InputVoltage_AllPins_Levels_Typ_TC, vih, 1.5V, 0.01V, 100";
}
```

Figure 3.13: Program setup of the testplan file for VIH test.

Figure 3.14 is the set of random test patterns used for these margin search test. The test patterns were created manually for this experiment.

```
NOP { V {inpins=0000; outpins=XXXXXXXX;} W{inpins=wfs1; outpins=wfs1;}}
        NOP { V {inpins=0011; outpins=LLLLLLLL;} }
        NOP { V {inpins=0011; outpins=LLLLLLLL;} }
        NOP { V {inpins=0000; outpins=LLLLLLLL;} }
        NOP { V {inpins=1100; outpins=LLLLLLLL;} }
        NOP { V {inpins=0000; outpins=HLLLHLLL;} }
        NOP { V {inpins=1100; outpins=HLLLHLLL;} }
        NOP { V {inpins=0000; outpins=LHLLLHLL;} }
        NOP { V {inpins=1100; outpins=LHLLLHLL;} }
        NOP { V {inpins=0000; outpins=HHLLHHLL;} }
        NOP { V {inpins=1100; outpins=HHLLHHLL;} }
        NOP { V {inpins=0000; outpins=LLHLLLHL;} }
        NOP { V {inpins=1100; outpins=LLHLLLHL;} }
        NOP { V {inpins=0000; outpins=HLHLHLHL;} }
```

Figure 3.14: Program setup of the pattern file for VIH test.

3.3.4 Test result

Figure 3.15 shows margin search test of VIH for device 74HC393. This plot provides information such as start value, stop value, resolution and the VIH levels that pass functional test. The symbol "." means the device using this voltage level does not pass functional test. The symbol "*" indicates this device passes the functional test at this voltage level. Figure 3.15 shows that VIH values from 1.98V to 2.49V can make this device function correctly.

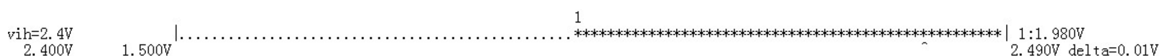


Figure 3.15: The test result of VIH on the device 74HC393.

Figure 3.16 shows margin search test results of VIL on the device 74HC393. Based on the result in Figure 3.16, the functional test is correct for VIL values from 1V to 1.56V.

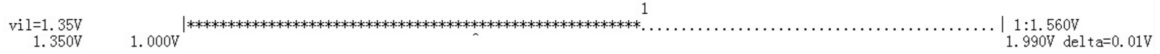


Figure 3.16: The test result of VIL on the device 74HC393.

Table 3.5 and Table 3.6 illustrate that the difference between VIH and VIL in LS devices is normally larger than in HC and HCT devices. In addition, test results are not always consistent with the specifications on the datasheet. For example, when testing the 74LS393, the specified minimum VIH is 2 V and maximum VIL is 0.8 V. However, the measured maximum VIL was found to be 0.66V.

Table 3.5: Input threshold voltage test result of 74393 chips (Unit for all values is V).

	74HC393	74HCT393	74LS393
VIH (Specified(Min))	3.15	2.0	2.0
VIH (Measured)	1.98	1.35	1.49
VIL (Specified(Max))	1.35	0.8	0.8
VIL (Measured)	1.56	0.84	0.66

Table 3.6: Input threshold voltage test result of 74163 chips (Unit for all values is V).

	74HC163	74HCT163	74LS163
VIH (Specified(Min))	3.15	2.0	2.0
VIH (Measured)	2.52	1.58	1.55
VIL (Specified(Max))	1.35	0.8	0.8
VIL (Measured)	2.12	1.33	1.04

3.4 Output voltage test

The output voltage test checks whether the voltage on an output pin satisfies the specification under the load conditions designated in the device electrical specification. The output voltage is classified as VOH, the high level output voltage, and VOL, the low level output voltage [6].

In the T2000 test system, margin search was used to find out where output voltages exceed their specified threshold values of VOH and VOL.

3.4.1 Test method

The output voltage test, using the margin search method, obtains the output voltage value while performing functional tests, to search for the pass/fail transition point by changing the comparison voltage value. This method is used when tests using the PMU cannot be performed (when the output voltage level is not stable) [7].

In testing VOH, power supply voltage is set to a typical value of the specification, and typical values are applied to all input pins. Output pins are configured to change the high level comparison voltage, VOH, while keeping low level comparison voltage to a typical value.

Similarly, in VOL testing, input voltage level is the same as for VOH testing, and the low level comparison voltage, VOL, is changed over a range.

3.4.2 Test procedure

The output voltage test using the margin search method is performed according to the following procedure.

1. Apply power supply voltage to the power supply pins in the levels file.
2. Set the input signal levels to input pins in the levels file.
3. Set output voltage levels in the levels file.
4. Set signal timing in the timing file.

5. Set up test vectors for functional test in the pattern file

6. Load test patterns and select margin search method in the test plan file.

7. Execute the test patterns.

8. Check the test pattern execution results in the Console Window.

9. Change the output signal comparison level.

10. Return to step 7 after the change of output signal comparison level.

11. Determine whether the output signal comparison level is within the specification of the device.

3.4.3 Test condition

When testing VOH and VOL, there are some parameters that should be changed in the testplan as shown in Figure 3.17. The levels file and patterns are the same as the test of VIH and VIL, as shown in Figure 3.12 and Figure 3.14, respectively.

According to Figure 3.17 and Figure 3.18, voh is changed from 3.98 V to 4.97 V for this test.

```
Test SearchTest Search_InputVoltageTest_High
{
     PListParam          = DiagPBPat;
     TestConditionParam  = InputVoltage_AllDps_Levels_Typ_TC;
     TestConditionParam  = InputVoltage_AllPins_Levels_Typ_TC;
     TestConditionParam  = FunctionalTest_Timing_Typ_TC ;


     Algorithm      = "Margin";
     SearchVarParam = "x,InputVoltage_AllPins_Levels_Typ_TC, voh, 3.98V, 0.01V, 100";
}
```

Figure 3.17: Program setup of the testplan file for VOH test.

3.4.4 Test result

Figure 3.18 shows the result of VOH testing for the device 74HC393. The result shows that the upper limit of device 74HC393 for VOH is 4.47V.

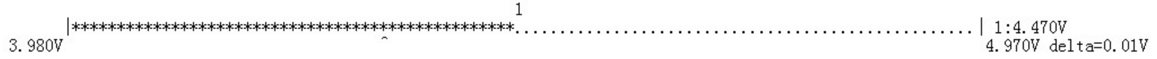


Figure 3.18: The test result of VOH on the device 74HC393.

Figure 3.19 shows the result of VOL testing for the device 74HC393. The result shows that the lower limit of device 74HC393 for VOH is 0.11V.

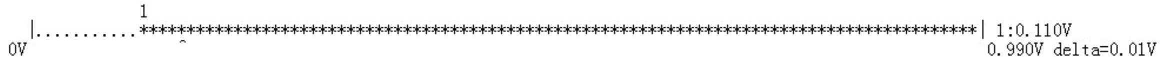


Figure 3.19: The test result of VOL on the device 74HC393.

Table 3.7 and Table 3.8 show VOH and VOL, respectively, for the six chips.

Table 3.7: Output driving voltage test result of 74393 chips (Unit for all values is V).

	74HC393	74HCT393	74LS393
VOH (Specified(Min))	3.98	3.98	2.7
VOH (Measured)	4.47	4.47	3.75
VOL (Specified(Max))	0.33	0.33	0.5
VOL (Measured)	0.11	0.1	0.25

Table 3.8: Output driving voltage test result of 74163 chips (Unit for all values is V).

	74HC163	74HCT163	74LS163
VOH (Specified(Min))	3.98	3.98	2.7
VOH (Measured)	4.5	4.5	2.4
VOL (Specified(Max))	0.26	0.26	0.5
VOL (Measured)	0.03	0.04	0.22

3.5 Gross IDD current test

Gross IDD current test is used to measure resistance between VDD and GND. For most cases, this test is performed after continuity test to determine if subsequent tests, after Gross IDD current test, are worth doing. This test is performed immediately after contact test. If functional test is performed on a faulty device, it may lead to high current on the power supply that affects the test system. If gross IDD is performed first, a high current that exceeds the clamp current value will turn off the power supply, which can protect the test system [6].

3.5.1 Test method

Since this test measures the current flow through the power supply, it utilizes the DPS module to take the measurement. All input pins are driven low, and output pins are left open. Current flow through the power supply pin is measured gross IDD current [17].

3.5.2 Test procedure

1. Apply power supply voltage to the power supply pin in the levels file.
2. Set level conditions of input signal pins in the levels file.
3. Set all input signal pins to 0 V in the levels file.
4. In the levels file, open the pinout relays of the output pins.
5. Set up power supply current measurement in the test plan file.
6. Measure a current value on the target power supply pin and view the results in the Console Window.
7. Determine whether the measured current value is within the specification of the device.

3.5.3 Test Program

Figure 3.20 is the levels file for this test. The key items are OPMode, IClampHi, IClampLo, JudgeUpperCurrentVal, JudgeLowerCurrentVal and SamplingMode.

```
DCParametrics staticICC_setting
{
        VDD
        {
                OPMode                  = VSIM; # Voltage source current measurement
                VSRange                 = 7V;   # Voltage source range
                VForce                  = 6V;   # Apply 6V to VDD
                IMRange                 = 50uA; # Current measurement
                IClampHi                = 10uA; # High clamp current
                IClampLo                = -10uA;# Low clamp current
                JudgeUpperCurrentVal    = 15uA; # High failed current
                JudgeLowerCurrentVal    = 0uA;  # Low failed current
                SamplingMode            = Average; #Sampling current every 100us and calculate the average value
                DpsRelay                = CLOSE;
                PowerSequence           = ON;
        }
        Delay 1mS;

}
```

(a) Program setup for VDD of DUT.

```
Levels GrossIDD_levels
{
        Delay 3mS;
        inpins
        {
                FIXL = 0V;
                PinOutRelay = CLOSE;
                PowerSequence = ON;
        }
        outpins
        {
                PinOutRelay = OPEN;

        }
        Delay 3mS;
}
```

(b) Program setup for input and output pins.

Figure 3.20: Program setup of the levels file for gross IDD current test.

Figure 3.21 shows one section of the testplan file. The key items are MeasurementPins and SettlingTime.

```
Test ATDCParametricPowerSupplyTest Gross_I_DCParametricTest
{
        TestDescription        = "Gross Power Supply  Current Test";
        TestConditionParam     = IDD_L_levels;
        TestConditionParam     = IDD_L_params;
        Datalog
        {
                Mode                   = AllLog, # display result in Console Window
                Style                  = Detailed
        }
        PowerSupplyParam
        {
                MeasurementPins = "VDD",
                SettlingTime    = 3mS   # settling time before perform measurement
        }
        PinOutRelay
        {
                Mode                   = Close
        }
}
```

Figure 3.21: Program setup of the testplan file for gross IDD current test.

3.5.4 Test result

The gross IDD current measurement result are tabulated in Table 3.9 and Table 3.10.

Table 3.9: Gross IDD current result of 74393 chips

	74HC393	74HCT393	74LS393
Current of VDD	3.48uA	3.16uA	12.46mA

Table 3.10: Gross IDD current result of 74163 chips

	74HC163	74HCT163	74LS163
Current of VDD	3.24uA	2.96uA	22.2mA

3.6 Static IDD current test

Static IDD current means the current is measured when the device is not active. This test is important for portable devices that operate on batteries. Since a portable device is always in standby mode, high static IDD current will cause higher power

consumption, resulting in short battery life. If a device has higher static IDD current than its specification, it may have some physical defects [6].

3.6.1 Test method

Figure 3.22 illustrates how static IDD test is performed. Before measuring power supply current, the test supplies maximum VDD voltage. Secondly, exact precondition patterns should be applied to input signal pins to set this device in steady-state mode [17]. In the case of the 74HC393, since there is not a specific steady state, a random test vector can be used to measure static IDD current. In the static IDD current test, the pinout relays of output pins are opened so that the power supply current value will not become larger by the influence of the digital module connected to the DUT output pins. After the device has been configured to steady state mode, current measurement is performed to comparing the measured value with the value in the specification.

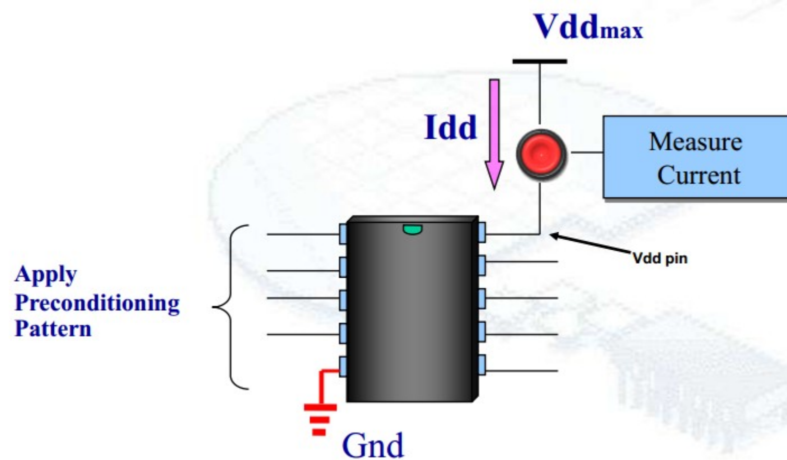


Figure 3.22: Static IDD current test setup diagram [6].

3.6.2 Test procedure

The static power-supply-current test is performed according to the following procedure.

1. Apply power supply voltage to the power supply pin in the levels file.

2. Set level conditions of input signal pins in the levels file.

3. Set output level conditions in the levels file.

4. Set timing conditions of input signal pins in the timing file.

5. Set up the module trigger and load test patterns in the test plan file.

6. Execute the test patterns to set the DUT to the stand-by state.

7. Open the pinout relays of the output pins in the levels file.

8. Measure a current value of the target power supply pin.

9. Determine whether the measured current value is within the specification of the device.

3.6.3 Test Program

Figure 3.23 is the example of the levels file that defines the measurement mode. The key items are OPMode, IClampHi, IClampLo, JudgeUpperCurrentVal, JudgeLowerCurrentVal, SamplingMode, TriggerControlStart and TriggerControlStop. TriggerControlStart and TriggerControlStop are used for controlling the start and stop of the current measurement.

```
DCParametrics staticICC_setting
{
        VDD
        {
                OPMode                  = VSIM; # Voltage source current measurement
                VSRange                 = 7V;   # Voltage source range
                VForce                  = 6V;   # Apply 6V to VDD
                IMRange                 = 50uA; # Current measurement
                IClampHi                = 50uA; # High clamp current
                IClampLo                = -50uA;# Low clamp current
                JudgeUpperCurrentVal    = 15uA; # High failed current
                JudgeLowerCurrentVal    = 0uA;  # Low failed current
                SamplingMode            = Trace; #Sampling current every 100us and calculate the average value
                TriggerControlStart     = External; # Module trigger signal control start sampling
                TriggerControlStop      = External; # Module trigger signal control stop sampling
                DpsRelay                = CLOSE;
                PowerSequence           = ON;
        }
        Delay 1mS;
```

Figure 3.23: Program setup of the levels files for static IDD current test.

3.6.4 Test result

In static IDD current test, the device is usually set to steady-state mode. However, when testing a chip like the 74HC393, there is no specific steady state, and therefore this device can be set to an arbitrary state to measure quiescent power supply current.

In this case, the test preconditions this device to all inputs at low voltage level and all outputs at high voltage level, and then static current flow through the power supply is measured.

However, when measuring static IDD current of the 74163 chips, the static IDD current will decrease to almost 0 A over time except for 74LS163. For instance, if a settling time is long enough, the static IDD current will reduce from hundreds of microamps to hundreds of nanoamps (about 400nA) in devices 74HC163 and 74HCT163 as shown in Figure 3.24. The static ICC current (24mA) for 74LS163 is shown in Figure 3.25.

DUT	Index	Result	Status	Value	Judge Hi	Judge Lo	Act.Cnt	Pin Name
1	1	PASS	Valid	412.0nA	10.00uA	0.000A	1	VDD
1	2	PASS	Valid	410.0nA	10.00uA	0.000A	1	VDD
1	3	PASS	Valid	408.0nA	10.00uA	0.000A	1	VDD
1	4	PASS	Valid	406.0nA	10.00uA	0.000A	1	VDD
1	5	PASS	Valid	406.0nA	10.00uA	0.000A	1	VDD
1	6	PASS	Valid	406.0nA	10.00uA	0.000A	1	VDD
1	7	PASS	Valid	406.0nA	10.00uA	0.000A	1	VDD
1	8	PASS	Valid	406.0nA	10.00uA	0.000A	1	VDD
1	9	PASS	Valid	404.0nA	10.00uA	0.000A	1	VDD
1	10	PASS	Valid	404.0nA	10.00uA	0.000A	1	VDD

Figure 3.24: Static IDD current test result of device 74HC163.

DUT	Index	Result	Status	Value	Judge Hi	Judge Lo	Act.Cnt	Pin Name
1	1	PASS	Valid	24.00mA	50.00mA	0.000A	1	VDD
1	2	PASS	Valid	24.00mA	50.00mA	0.000A	1	VDD
1	3	PASS	Valid	24.20mA	50.00mA	0.000A	1	VDD
1	4	PASS	Valid	24.00mA	50.00mA	0.000A	1	VDD
1	5	PASS	Valid	23.80mA	50.00mA	0.000A	1	VDD
1	6	PASS	Valid	24.00mA	50.00mA	0.000A	1	VDD
1	7	PASS	Valid	24.00mA	50.00mA	0.000A	1	VDD
1	8	PASS	Valid	24.00mA	50.00mA	0.000A	1	VDD
1	9	PASS	Valid	24.00mA	50.00mA	0.000A	1	VDD
1	10	PASS	Valid	24.00mA	50.00mA	0.000A	1	VDD

Figure 3.25: Static IDD current test result of device 74LS163.

3.7 Power supply bump test

The power supply bump test checks whether a device functions properly if the power supply voltage fluctuates while the device is operating.

3.7.1 Test method

To execute the power supply voltage bump test, the same conditions are used as the functional test of the DUTs. Power supply voltage is fluctuated during pattern execution to check that the DUT is not influenced by the fluctuation.

The DPS module has a power supply bump test function (VBUMP function), as illustrated in Figure 3.26. By using the power supply bump test function, the power supply voltage can be changed while executing a test. The DPS module can hold four voltage values to be used in the test. The voltage value changes at times determined by an external trigger signal. A two-bit external trigger signal is required for the power supply bump test. The external trigger signal is supplied from the module trigger of the 250MHz Digital Module as shown in Figure 3.26 [10].

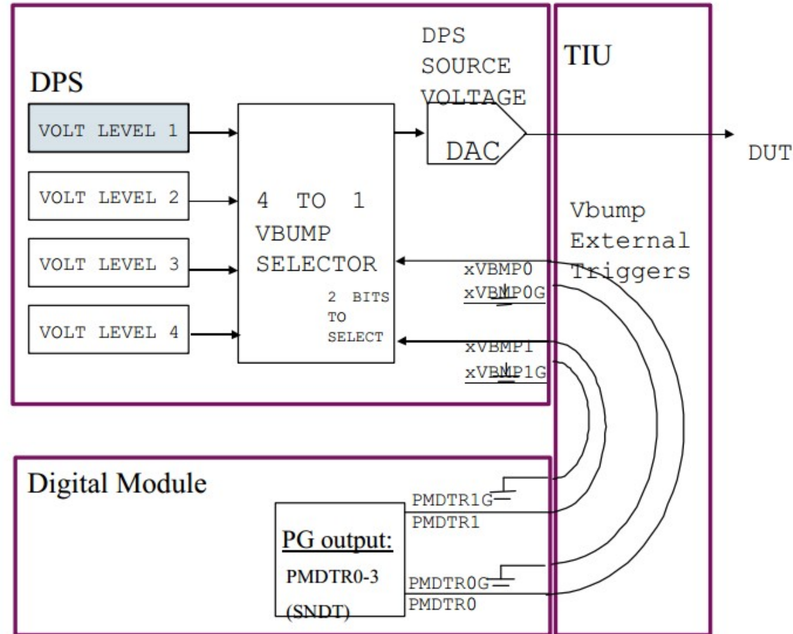


Figure 3.26: The connection from PMDTR0-1 on digital module to VBMP0-1 on DPS module [10].

Vbump trigger control

DPS500mA provides four user selected voltages for a voltage bump test. Which voltage is applied to an operating device depends on the two trigger signals sent from the pattern generators of an active digital module. Figure 3.26 illustrates the 4 to 1 selector for the DPS source voltage DAC. The selected value is determined by signal VBMP0-1 trigger, supplied from the 250MDM digital module [10].

As shown in the example of Figure 3.27, four SNDT instructions are used in the pattern file to generate triggers to the DPS module. The four voltage values are programmed with the selector numbers as shown. The SNDT parameter and module triggers can determine which voltage value is applied to the device. For instance, the parameter of module trigger ”PMDTR0” is 0x1 and ”PMDTR1” is 0x2, which are the parameters in the testplan file shown in Figure 3.28. 0x1 means 01 in binary format. In Figure 3.29, the pattern file shows that the SNDT instruction parameters are from

0x0 to 0x3, which are 00, 01, 10 and 11 in the binary form. The final outputs of "PMDTR0" are decided by the logic AND of the "PMDTR0" bit from the testplan file and SNDT parameter from the pattern file. In this case, in the pattern with SNDT 0x0 as shown in Figure 3.29, "PMDTR0" is 0 since the logic AND of "PMDTR0" pattern (01) and SNDT 0x0 pattern (00) is 00. For the same reason, "PMDTR1" is 0 since logic AND of "PMDTR1" bit (10) and SNDT 0x0 bit (00) is 00. However, for the test pattern with instruction SNDT 0x1, the logic AND of "PMDTR0" bit (01) and SNDT 0x1 bit (01) is 01, and logic AND of "PMDTR1" bit (10) and SNDT 0x1 bit (01) is 00. The result shows how power supply voltage changes when vbump0-1 signals are varied in Figure 3.27. As mentioned in the manual [10], the minimum period of a test vector is 4 us in voltage bump tests.

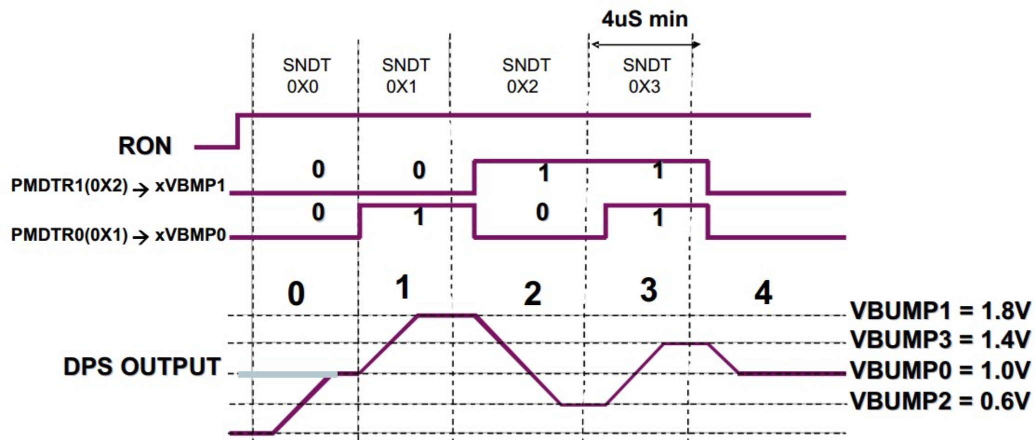


Figure 3.27: The supply voltage is fluctuated depending on SNDT and PMDTR0-1 [10].

```
ModuleTrigger
{
        Pin             =       "PMDTR0",
        BitSelect       = 0x1
}
ModuleTrigger
{
        Pin             =       "PMDTR1",
        BitSelect       = 0x2
}
```

Figure 3.28: Define module trigger signals in the testplan file.

```
NOP { V {inpins=1100; outpins=LLLLLLLL;} }
NOP { V {inpins=0000; outpins=HLLLHLLL;} }
SNDT 0x1 { V {inpins=1100; outpins=HLLLHLLL;} } #Select voltage level 2
NOP { V {inpins=0000; outpins=LHLLLHLL;} }
SNDT 0x2 { V {inpins=1100; outpins=LLHLLLHL;} } #Select voltage level 3
NOP { V {inpins=0000; outpins=HLHLHLHL;} }
SNDT 0x3 { V {inpins=1100; outpins=HHHLHHHL;} } #Select voltage level 4
NOP { V {inpins=0000; outpins=LLLHLLLH;} }
SNDT 0x0 { V {inpins=0000; outpins=HHLHHHLH;} } #Select voltage level 1
NOP { V {inpins=1100; outpins=HHLHHHLH;} }
```

Figure 3.29: Pattern file example of voltage bump test.

3.7.2 Test procedure

The power supply bump test is performed according to the following procedure.

1. Apply power supply voltage to the power supply pins in the levels file.
2. Set a signal level in the levels file.
3. Set load conditions in the levels file.
4. Set signal timing in the timing file.
5. Execute the test patterns.
6. Fluctuate the power supply voltage during test pattern execution.
7. Check whether the logic outputs are correct or not.

3.7.3 Test Program

Part of the testplan file, defining the module trigger, is displayed in Figure 3.28. The levels file for this test is shown in Figure 3.30. The levels file defines the four

VForce values. The pattern file with the four SNDT instructions is shown in Figure 3.31. The NOP instruction means no a specified operation but just runs patterns without any action.

```
VDD
{
        VSRange                                 = 7V;
        VBumpMode                               = Enable;
        VForce                                  = 4.4V; # voltage level 1
        VForce                                  = 4.3V; # voltage level 2
        VForce                                  = 4.5V; # voltage level 3
        VForce                                  = 4.6V; # voltage level 4
        IClampHi                                = 1mA;
        IClampLo                                = -1mA;
        DpsRelay                                = CLOSE;
        PowerSequence                   = ON;
}
```

Figure 3.30: The program setup of the levels file for voltage bump test.

According to the test mechanism that illustrates in the section 3.7.1 of this chapter, the vectors between "SNDT 0x0" and "SNDT 0x1" will be applied with 4.4 V as the power supply voltage.

The vectors between "SNDT 0x1" and "SNDT 0x2" will be applied with power supply voltage =4.3V. The vectors between "SNDT 0x2" and "SNDT 0x3" will be applied with power supply voltage =4.5V. The vectors between "SNDT 0x3" and "SNDT 0x0" will be applied with power supply voltage =4.6V. The vectors between "SNDT 0x0" and "EXIT" will be applied with power supply voltage =4.4V.

```
        SNDT 0x0 { V {inpins=0000; outpins=XXXXXXXX;} W{inpins=wfs1;
outpins=wfs1;}}
        NOP { V {inpins=0011; outpins=LLLLLLLL;} }
        NOP { V {inpins=0011; outpins=LLLLLLLL;} }
        NOP { V {inpins=0000; outpins=LLLLLLLL;} }
        NOP { V {inpins=1100; outpins=LLLLLLLL;} }
        NOP { V {inpins=0000; outpins=HLLLHLLL;} }
        SNDT 0x1 { V {inpins=1100; outpins=HLLLHLLL;} }
        NOP { V {inpins=0000; outpins=LHLLLHLL;} }
        NOP { V {inpins=1100; outpins=LHLLLHLL;} }
        NOP { V {inpins=0000; outpins=HHLLHHLL;} }
        NOP { V {inpins=1100; outpins=HHLLHHLL;} }
        NOP { V {inpins=0000; outpins=LLHLLLHL;} }
        SNDT 0x2 { V {inpins=1100; outpins=LLHLLLHL;} }
        NOP { V {inpins=0000; outpins=HLHLHLHL;} }
        NOP { V {inpins=1100; outpins=HLHLHLHL;} }
        NOP { V {inpins=0000; outpins=LHHLLHHL;} }
        NOP { V {inpins=1100; outpins=LHHLLHHL;} }
        NOP { V {inpins=0000; outpins=HHHLHHHL;} }
        SNDT 0x3 { V {inpins=1100; outpins=HHHLHHHL;} }
        NOP { V {inpins=0000; outpins=LLLHLLLH;} }
        NOP { V {inpins=1100; outpins=LLLHLLLH;} }
        NOP { V {inpins=0000; outpins=HLLHHLLH;} }
        NOP { V {inpins=1100; outpins=HLLHHLLH;} }
        NOP { V {inpins=0000; outpins=LHLHLHLH;} }
        NOP { V {inpins=1100; outpins=LHLHLHLH;} }
        SNDT 0x0 { V {inpins=0000; outpins=HHLHHHLH;} }
        NOP { V {inpins=1100; outpins=HHLHHHLH;} }
        NOP { V {inpins=0000; outpins=LLHHLLHH;} }
        NOP { V {inpins=1100; outpins=LLHHLLHH;} }
        NOP { V {inpins=0000; outpins=HLHHHLHH;} }
        NOP { V {inpins=1100; outpins=HLHHHLHH;} }
        NOP { V {inpins=0000; outpins=LHHHLHHH;} }
        NOP { V {inpins=1100; outpins=LHHHLHHH;} }
        NOP { V {inpins=0000; outpins=HHHHHHHH;} }
        NOP { V {inpins=0000; outpins=HHHHHHHH;} }
        EXIT { V {inpins=0000; outpins=HHHHHHHH;} }
```

Figure 3.31: The pattern file used in voltage bump test of the device 74HC393.

3.7.4 Test result

The specified VDD for all HC devices is from 2 V to 6 V. The specified VDD for all HCT devices is from 4.5 V to 5.5 V. The specified VCC for all LS devices is from 4.5 V to 5.5V.

When the test condition described above is applied, all of the six chips passed the power supply bump test for values between 4.3V and 4.6V.

However, if a voltage level is changed to a much lower voltage level, like changing from 4.3V to 3.9V, all six devices fail functional test. The failed information is displayed in Figure 3.32. In addition, power supply voltage can directly determine the output voltage levels. Therefore, the output driving voltage is decreased with a lower power supply voltage. For this reason, if the preset comparison voltage level VOH in the test program file is low enough, changing the power supply voltage over a wide range can also make both HC and LS devices work properly. For example, the device 74LS393 has a minimum power supply voltage 4.5V on its datasheet, and the minimum output driving voltage level VOH is 2.4V. In testing, only when power supply voltage is below 3.5V does this device output a voltage below the minimum VOH. In contrast, if the power supply voltage of a device 74HC393 is 3.5V, the high output voltage level is around 3.4V.

Figure 3.32 shows which patterns failed. Symbol ”#” means the output values are not correct for the applied pattern.

```
                                        AACCQQQQQQQQ
                                        12LLABCDABCD
                                          RR11112222
                                          12
MainAddr   AppAddr    Cycle      SS         ########
>>>> Pattern : ls393
       0x5                      6   0 M 0000HLLLHLLL
                                            #   #
      0x1a                     27   0 M 1100HHLHHHLH
                                            ## ### #
      0x1b                     28   0 M 0000LLHHLLHH
                                              ##  ##
      0x1c                     29   0 M 1100LLHHLLHH
                                              ##  ##
      0x1d                     30   0 M 0000HLHHHLHH
                                            # ### ##
      0x1e                     31   0 M 1100HLHHHLHH
                                            # ### ##
      0x1f                     32   0 M 0000LHHHLHHH
                                             ### ###
      0x20                     33   0 M 1100LHHHLHHH
                                             ### ###
      0x21                     34   0 M 0000HHHHHHHH
                                            ########
      0x22                     35   0 M 0000HHHHHHHH
                                            ########
      0x23                     36   0 M 0000HHHHHHHH
                                            ########
```

Figure 3.32: The function error in voltage bump test.

Chapter 4

IDDQ test

In the 1980s, the yields of IC products increased due to their quality and reliability remarkably improving in this period. The failure rates dramatically decreased from 1000 dpm (defects per million) to 50 dpm. In the same period, IBM proposed a goal to reduce failure rate to 3.4 dpm. This approach is called ”zero defects”. However, the conventional test methods would need to increase test fault coverage, burn-in coverage and electrostatic discharge damage awareness if they were to reach this failure rate. For the sake of improving IC test coverage much easier, IDDQ current testing was carried out to boost test coverage. The stuck-at fault model is typically used for functional test. However, when it comes to resistive bridging faults, tests designed to detect stuck-at faults have limited ability to detect them [2].

4.1 Reason for IDDQ testing

The main reason for IDDQ testing is because it is extremely cost effective and it detects many physical defects. For instance, it may double test cost to increase stuck-at fault test coverage from 80% to 90% - 95% [18]. However, a small IDDQ test set is able to reach the same level of test coverage. Test coverage can be increased to almost 100% when IDDQ testing is applied as a supplement to functional test [18]. However, IDDQ testing does not check the functionality if power supply current is the only object observed. So this is the reason why IDDQ is used as a supplemental test.

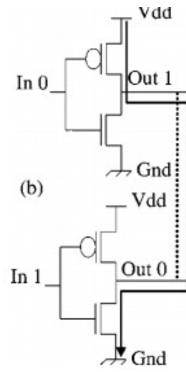
Due to the reason that functional testing can be time consuming when test coverage is above 90%, IDDQ testing is a good supplemental test to shorten the time

to market. For example, 60% functional fault coverage plus 20 IDDQ vectors might raise the test coverage up to 95%. Moreover, a faulty circuit can be often determined by observing power supply current. Without verifying logic outputs. In addition, research has shown that IDDQ testing is capable of detecting some defects that do not result in logic failures but may lead to timing related faults (e.g. resistive bridging and sub-threshold leakage faults) [8].

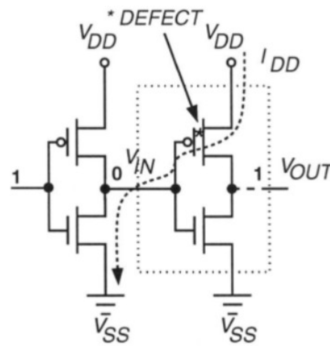
4.2 IDDQ theory

In a CMOS circuit, the dissipated current of a defect-free circuit is almost zero or very small when in steady state. But in the case of a gate-oxide short or internal short circuit, a short circuit is formed from the power supply (VDD) of the circuit to ground (GND). Due to this reason, a noticeably high current can be easily observed through the power supply. The power supply current of a faulty circuit can be higher than that of a fault-free circuit by orders of magnitude. Hence, the faulty circuit and fault-free circuit would be distinguished in this way [2].

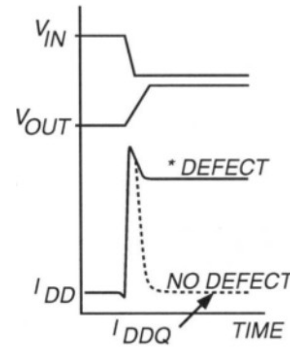
Figures 4.1(a) and 4.1(b) show examples of short circuit paths and Figure 4.1(c) indicates how power supply current changes in a faulty circuit. As shown in Figure 4.1(a), one defect is because of the conduction between outputs of two inverters. A PMOS transistor is on in one inverter and an NMOS transistor is on in the other inverter, forming a conducting path leading to a high current flow from VDD to GND. In Figure 4.1(b) another defect, named gate-oxide short, makes the PMOS transistor of the second inverter shorted, which results in conduction from VDD through the PMOS and NMOS transistor to GND. Therefore, Figure 4.1(c) illustrates that IDDQ current is low if the circuit is defect free. By contrast, IDDQ current is high after current has subsided if this circuit has a gate oxide shorted defect. Thus, measuring power supply current is able to find out whether a circuit is faulty or not.



(a) Example of a bridging fault. [19]



(b) Example of a gate oxide short in a CMOS transistor. [2]



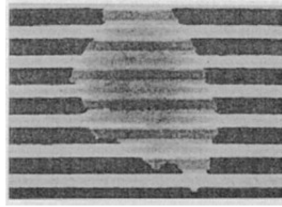
(c) The power supply current change in a faulty circuit. [2]

Figure 4.1: IDDQ fault.

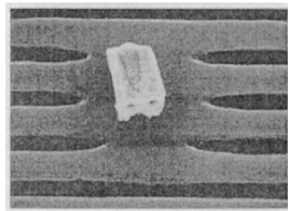
4.3 Different types of physical defects

4.3.1 Bridging fault

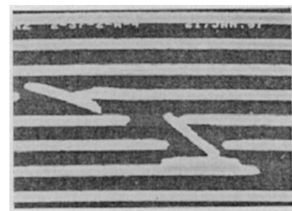
A bridging fault is a physical defect that links multiple metal lines together. Pictures of real physical bridging defect are displayed in Figure 4.2.



(a) The unexposed photoresist causes a short circuit among several metal lines. [20]



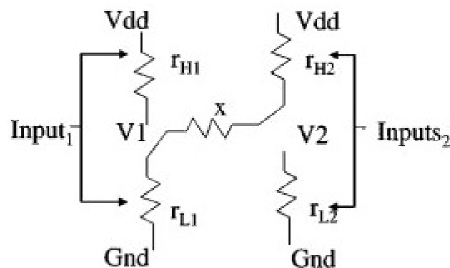
(b) A particle on the metal mask leads to four metal lines shorted. [20]



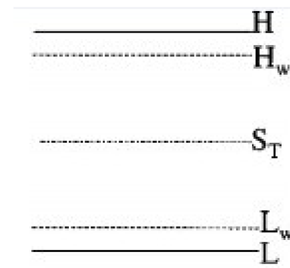
(c) Shorts and breaks of metal lines are caused by a scratch in the photoresist. [20]

Figure 4.2: Photos of short circuits in physical layouts.

Bridging faults are not always absolute shorts. In many cases low resistive and high resistive shorts can also cause circuit failure [2].



(a) Simple potential divider model for bridging. [21]



(b) Definition of voltage levels. [21]

Figure 4.3: Analysis of short circuit resulting from a bridging fault.

The behavior of a bridging fault can be explained by the potential divider rule illustrated in Figure 4.3. The outputs of two logic elements are indicated by V1 and V2, and the bridge resistance is shown as x. The resistance that connects the L(H) node to ground(Vdd) is r_L (r_H). If the minimum output high voltage is H_W, the H-level noise margin is N_H, the maximum output low voltage is L_W, the L-level noise margin is N_L, and the switching threshold voltage is S_T. So the voltages at the output nodes V_1 and V_2 are

$$V_1 = Vdd[r_{L1}/(r_{L1} + x + r_{H2})] \tag{4.1}$$

and

$$V_2 = Vdd[(r_{L1} + x)/(r_{L1} + x + r_{H2})] \tag{4.2}$$

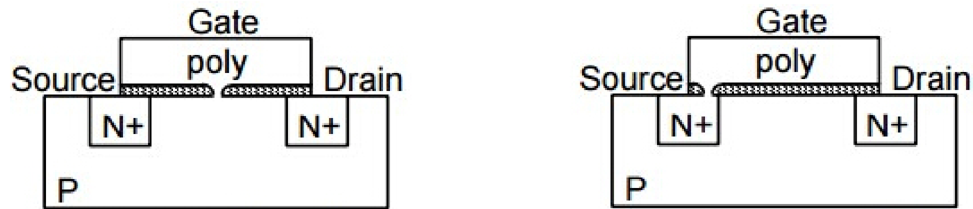
When $x = \infty$, there is no bridging fault. Assume that V1 $=$ 0 and V2 $=$ Vdd. If x $=$ 0, V_1 and V_2 can be 1/2Vdd when r_{L1} and r_{H2} are equivalent. This means the bridge resistance determines whether it induces a faulty logic value or not. Since 1/2Vdd may result in an unknown logic value and a high bridge resistance may cause changes in Hw and Lw to reduce reliability. For example, if the resistance $x + r_{H2} = 2 * r_{L1}$, the V_1 will be half of the original value that changes the output driving voltage level. For this reason, IDDQ is a good test method to detect a bridging fault.

Research has showed that most cases of bridging faults are low resistance while only approximately 20% behave as high resistance [22] [23]. However, high resistance bridging faults may not cause any logic failures, but may have slight effects on noise margin, thus reducing reliability. Therefore, IDDQ testing was designed for testing such bridging faults. As shown in Figure 4.1(a), the conduction path from VDD to GND due to a bridging fault significantly boosts the value of power supply current.

The IDDQ testing method does not require fault propagation; it only needs to check power supply current.

4.3.2 Gate oxide short (GOS)

Gate oxide defects cause pinholes, micropores, dendrites, trapped charge and direct shorts to diffusion. These kind of defects happen frequently, and result in an unexpected current flow through the gate oxide of a MOSFET transistor. GOS is a defect that forms a low resistance path between gate oxide and underlying silicon. According to the location of the defect, GOS can be divided into two types. One is gate to transistor channel short, which is shown in Figure 4.4(a). The other one is a gate to diffusion short that is illustrated in Figure 4.4(b). As shown by the example in Figure 4.1(b), since the PMOS transistor gate oxide is shorted to diffusion, VDD is directly connected to GND when the NMOS transistor of the first inverter is conducting.



(a) Shorted circuit path from gate to substrate. [25]

(b) Shorted circuit path from gate to source diffusion. [25]

Figure 4.4: Gate oxide short models.

In a majority of cases, GOS may result in reliability degradation (e.g threshold voltage change) and slow switching time. Only avalanche breakdown and subsequent shorts will cause function to be abnormal. There are some papers that present accurate models of gate oxide shorts [24]. Generally speaking, functional testing is not a good way to detect gate oxide defects. IDDQ testing is relatively efficient in detecting this kind of fault via abnormal power supply current.

4.4 IDDQ test vectors

Since IDDQ testing requires taking measurements of quiescent power supply current, it does not need to verify pin voltage. This test only needs to implement a sequence of IDDQ test vectors and measure power supply current after power supply current has subsided for each test vector. The IDDQ test vectors generated by E-DA tools mainly detect gate-oxide faults, adjacent bridging faults or stuck-on faults. Devices can be judged to have internal manufacturing defects if excessive current occurs.

IDDQ testing does not propagate IDDQ fault effects to output signal pins but excessive current passes through the power supply pin. For this reason, the generated IDDQ test vector set may be very compact and still cover a high percentage of faults. IDDQ testing can be treated as a supplemental test method to increase overall test coverage achieved by functional test.

IDDQ test vectors can be produced in the following three ways [26]:

Every-vector. This methodology basically applies every vector in a functional test vector set to observe current on the power supply. However, applying each test vector of a functional test is not efficient because taking a current measurement is very slow compared with voltage test.

Supplemental. This method can save time by using a compact set of special IDDQ test vectors. The generated IDDQ test vectors are intended to toggle as many values as possible to trigger potential bridging faults. Supplemental IDDQ test vectors can be generated by FastScan.

Selective. This method intelligently chooses a compact test vector set from the test vector list of a functional test. FastScan can also select test vectors for IDDQ test from external pattern files.

In the experiments performed in this study, Tessent FastScan [26] is used to produce IDDQ test vector sets by supplemental and selective test methodologies.

IDDQ test vectors can be classified into two test vector types [26].

Ideal. Applying ideal IDDQ test vectors to a device can produce nearly zero quiescent power supply current during testing of a good device. Most methodologies would like to see this result.

Non-ideal. This kind of IDDQ test vector will produce a small quiescent current that flows through the power supply when testing a good circuit.

Two CMOS device types can be considered when perform IDDQ test [26]:

Fully static. All states of this kind of device consume nearly zero IDDQ current. These circuits do not have pull up or pull down resistors, and there is only one active driver at a time on tri-state buses. Therefore, any vector applied to this circuit will result in a very small quiescent current.

Resistive. In resistive CMOS circuits, high IDDQ current will flow through the power supply by applying IDDQ test vectors, due to pull-up and pull-down resistors and tri-state buses that these circuits have.

4.5 Generating IDDQ test vectors by ATPG

This section describes the process of generating IDDQ test vectors with FastScan. The command to start up FastScan is:

fastscan testckt.v -lib $ADK/technology/adk.atpg -Dofile dofile_name

"Testckt.v" is the circuit netlist file for which we want to generate IDDQ test vectors. "-lib $ADK/technology/adk.atpg" is a library file. "-Dofile dofile_name" means use a do file of commands to generate IDDQ test vectors. The do file example used in this work is shown in Figure 4.5.

```
set system mode atpg
set fault type iddq
add faults -all
set pattern source internal
create patterns -auto
run
write fault atpg.flt -replace
save patterns atpg.pat -replace
report faults > atpg.faults
report statistics > atpg.stats
exit
```

Figure 4.5: Do file example for generating IDDQ test vectors.

The command "set fault type iddq" causes the produced test vector to be designed for IDDQ testing. This command can also be changed to generate test vectors for toggle, stuck-at or path delay faults. This simple example used supplemental generation methodology because the pattern source is set to internal, rather than selecting IDDQ test vectors from an external pattern file.

The generated test vector set will be saved to file "atpg.pat", for our simple example. In this example, we imported a Verilog netlist of the 74HC163, a binary counter, created from a schematic in its datatsheet, to get IDDQ test vectors. There is also a command that is capable of selecting a number of IDDQ vectors from the generated supplemental IDDQ test vectors.

Figure 4.6 shows the statistics of this simple example. Since the 74HC163 chip does not have a scan chain in its schematic, the test coverage is only 58.16% by using the generated patterns shown in Figure 4.7.

Figure 4.7 shows detailed test vectors for performing this IDDQ test. For instance, in pattern 0, "force "PI" "001010001" 0" means apply the input vector "001010001" to primary inputs as step 0. Figure 4.7(a) shows the corresponding signal pin for each test vector. For example, in pattern 0, only input signal pins "D2", "CET" and "CP" are set to high in this pattern. The statements: "measure "PO" "XXXXX" 1" and "measure IDDQ ALL 1" indicate measuring primary output

values and quiescent power supply current in step 1. Since IDDQ testing measures current flow through the power supply, it does not care about output logic. The resulting test coverage is 58% even though logic outputs are not observed.

```
Statistics:
    Test Coverage   =  58.16%
    Total Faults    =  282
        DS (det_simulation)     =  164
        AU (atpg_untestable)    =  118
    Total            Patterns   =  5
```

Figure 4.6: Statistics of the ATPG produced IDDQ test vectors.

```
SETUP =

    declare input bus "PI" = "/D0", "/D1", "/D2", "/D3", "/CET",
         "/CEP", "/PE", "/MR", "/CP";

    declare output bus "PO" = "/Q0", "/Q1", "/Q2", "/Q3", "/TC";
```

(a) The signal pin sequence.

```
pattern = 0;
force   "PI" "001010001" 0;
measure "PO" "XXXXX" 1;
measure IDDQ ALL 1;

pattern = 1;
force   "PI" "100010110" 0;
measure "PO" "XXXXX" 1;
measure IDDQ ALL 1;

pattern = 2;
force   "PI" "000001100" 0;
measure "PO" "XXXX0" 1;
measure IDDQ ALL 1;

pattern = 3;
force   "PI" "111111011" 0;
measure "PO" "XXXXX" 1;
measure IDDQ ALL 1;

pattern = 4;
force   "PI" "000000010" 0;
measure "PO" "XXXX0" 1;
measure IDDQ ALL 1;
```

(b) IDDQ test vectors.

Figure 4.7: The generated IDDQ test pattern file.

4.6 Program example

4.6.1 Pattern file

Figure 4.8 is an example of a T2000 pattern file that can be used for testing power supply current. In this file, instruction SNDT is used to cause IDDQ current to be measured between "SNDT 0x1" to "SNDT 0x2".

The SNDT instruction controls the module trigger signal in order to attain synchronization with the DPS500mA. The 250Mbps Digital Module has four trigger signals that are transmitted to the performance board. The SNDT instruction controls trigger signal outputs using eight-bit data (0 to 255) specified for the operand, as explained in the section module trigger of Chapter 2.

```
NOP { V {inpins=0000; outpins=LLLLLLLL;} W{inpins=wfs3; outpins=wfs3;} }
NOP { V {inpins=0011; outpins=LLLLLLLL;} W{inpins=wfs2; outpins=wfs2;} }
NOP { V {inpins=1100; outpins=LLLLLLLL;} W{inpins=wfs1; outpins=wfs1;} }
SNDT 0x1 { V {inpins=0000; outpins=HLLLHLLL;} }
NOP { V {inpins=1100; outpins=HLLLHLLL;} }
NOP { V {inpins=0000; outpins=LHLLLHLL;} }
SNDT 0x2 { V {inpins=0000; outpins=LLLLLLLL;} }
EXIT { V {inpins=0000; outpins=LLLLLLLL;} }
```

Figure 4.8: An example pattern file for IDDQ test.

The example of Figure 4.9 shows how module triggers change based on the SNDT instructions in the pattern file of Figure 4.8. In the picture, module trigger 0 is forced high in instruction "SNDT 0x1" and remains high for the following two "NOP" instructions (NOP means no operation) before instruction "SNDT 0x2". Since Module Trigger 0 is the only module trigger pin used during this IDDQ test, the test system only responds to Module Trigger 0. That means power supply current is measured in the period when this signal is at a high level.

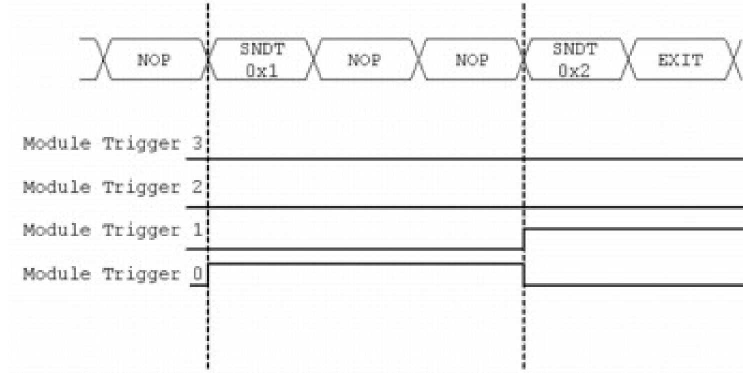


Figure 4.9: Module trigger changed their values by controlled SNDT instruction [11].

4.6.2 Testplan File

Figure 4.10 is the OPTL code used in the testplan file. In Figure 4.10, "Datalog" indicates the results to be displayed in the Console Window, a built-in tool of the T2000 Test System Software. With the help of this tool, any fault result can be easily observed and then we can determine which kind of fault the DUT has.

```
Datalog
{
    Mode            = AllLog,
    Style           = Detailed
}
PowerSupplyParam
{
    MeasurementPins = "VDD",
    SettlingTime    = 1mS
}
ModuleTrigger
{
    Pin             = "PMDTR0",
    BitSelect       = 0x1
}
```

Figure 4.10: A part of the testplan file for setting IDDQ test.

In the block "PowerSupplyParam" in Figure 4.10, "MeasurementPins" defines which pin of the device is to be measured. "SettlingTime" determines how much time

is needed to wait after module trigger signal has been set to the high level. In IDDQ testing, it is usually required to wait some time to allow IDDQ to subside before measuring.

In the "ModuleTrigger" section, "Pin" indicates the name of the module trigger pin. In this case, Module Trigger 0 (PMDTR0) is used. "BitSelect" is an output control bit that determines which module trigger signal will be turned on in the operation. This variable can be changed to 0x1, 0x2, 0x3 and 0x4 for Module Triggers 0-3, respectively. The example presented in Chapter 2 illustrates module trigger operation in detail.

4.6.3 Levels File

In the levels file shown in Figure 4.11, voltage source current measurement (VSIM) is selected for measuring current flow through the VDD. Pin current measurement range (IMRange) is defined as 5mA. High clamp current and low clamp current are set to 1mA and -1mA, respectively. The statement "JudgeUpperCurrentVal = 3mA" means when current from VDD exceeds 3mA, it judges that this device has failed the IDDQ test. This design measures current over a period of 100uS, which is controlled by the module trigger.

```
VDD
{
    OPMode                  = VSIM; #voltage source current measurement
    VSRange                 = 7V;   #voltage source range
    VForce                  = 4.5V; #apply 5 V to VDD
    IMRange                 = 5mA;  #current measurement range
    IClampHi                = 50mA; #current exceed this value will turn off power
    IClampLo                = -50mA;#current lower than this value will turn off power
    JudgeUpperCurrentVal    = 3mA;  #current exceed 3mA will judge to fail
    JudgeLowerCurrentVal    = 0uA;  #current lower than 0A will be judged to fail
    SamplingMode            = Trace;#display current on each sampling point
    TriggerControlStart     = External;#control sampling start point by trigger
    TriggerControlStop      = External;#control sampling stop point by trigger
    DpsRelay                = CLOSE;    #close the relay between device and tester
    PowerSequence           = ON;   #give VDD a value when in operation
```

Figure 4.11: The parameters of the levels file for IDDQ test.

4.7 Test program

Figure 4.10 is the setting of the testplan file and Figure 4.11 is the setting of the levels file for IDDQ test. Figure 4.12 is a pattern file that measures IDDQ current of a 74HC163 between ”SNDT 0x1” to ”SNDT 0x2”. The period for each vector is 1ms. Since IDDQ current is the quiescent VDD current after applying a test vector to inputs, it requires current to be subsided to a stable value before determining if the current is appropriate. In this example, each IDDQ test vector is required to operate for 6ms to subside IDDQ current. The cycle for each pattern is 1ms, so an IDDQ test vector is applied for seven consecutive cycles. Among the seven cycles, since the settling time is 6ms so that the first six cycles are used for subsiding current to stable value. Therefore measurements only sample the current in the last 1ms of each vector. Measurements can sample 10 current values for each vector since the sampling interval is 100us.

```
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=0; CLK=0; outpins=LLLLL;} W{inpins=wfs3; outpins=wfs3;}}
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=0; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=1; outpins=LLLLL;} }

SNDT 0x1 { V {inputs=1000; LOAD=1; ENP=0; ENT=1; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=1000; LOAD=1; ENP=0; ENT=1; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=1000; LOAD=1; ENP=0; ENT=1; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=1000; LOAD=1; ENP=0; ENT=1; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=1000; LOAD=1; ENP=0; ENT=1; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=1000; LOAD=1; ENP=0; ENT=1; CLR=1; CLK=0; outpins=LLLLL;} }
SNDT 0x2 { V {inputs=1000; LOAD=1; ENP=0; ENT=1; CLR=1; CLK=0; outpins=LLLLL;} }

SNDT 0x1 { V {inputs=0000; LOAD=1; ENP=1; ENT=0; CLR=0; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=1; ENP=1; ENT=0; CLR=0; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=1; ENP=1; ENT=0; CLR=0; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=1; ENP=1; ENT=0; CLR=0; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=1; ENP=1; ENT=0; CLR=0; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=1; ENP=1; ENT=0; CLR=0; CLK=0; outpins=LLLLL;} }
SNDT 0x2 { V {inputs=0000; LOAD=1; ENP=1; ENT=0; CLR=0; CLK=0; outpins=LLLLL;} }

SNDT 0x1 { V {inputs=1111; LOAD=0; ENP=1; ENT=1; CLR=1; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=1111; LOAD=0; ENP=1; ENT=1; CLR=1; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=1111; LOAD=0; ENP=1; ENT=1; CLR=1; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=1111; LOAD=0; ENP=1; ENT=1; CLR=1; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=1111; LOAD=0; ENP=1; ENT=1; CLR=1; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=1111; LOAD=0; ENP=1; ENT=1; CLR=1; CLK=1; outpins=LLLLL;} }
SNDT 0x2 { V {inputs=1111; LOAD=0; ENP=1; ENT=1; CLR=1; CLK=1; outpins=LLLLL;} }

SNDT 0x1 { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=0; outpins=LLLLL;} }
NOP { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=0; outpins=LLLLL;} }
SNDT 0x2 { V {inputs=0000; LOAD=0; ENP=0; ENT=0; CLR=1; CLK=0; outpins=LLLLL;} }

SNDT 0x1 { V {inputs=0010; LOAD=0; ENP=0; ENT=1; CLR=0; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=0010; LOAD=0; ENP=0; ENT=1; CLR=0; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=0010; LOAD=0; ENP=0; ENT=1; CLR=0; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=0010; LOAD=0; ENP=0; ENT=1; CLR=0; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=0010; LOAD=0; ENP=0; ENT=1; CLR=0; CLK=1; outpins=LLLLL;} }
NOP { V {inputs=0010; LOAD=0; ENP=0; ENT=1; CLR=0; CLK=1; outpins=LLLLL;} }
SNDT 0x2 { V {inputs=0010; LOAD=0; ENP=0; ENT=1; CLR=0; CLK=1; outpins=LLLLL;} }

EXIT { V {inputs=1000; LOAD=1; ENP=1; ENT=1; CLR=1; CLK=0; outpins=LLLLL;} }
```

Figure 4.12: The pattern file for IDDQ test.

4.8 Test result

Device 74HC163 is used to perform IDDQ test by applying IDDQ test vectors generated by FastScan.

Since different input voltages will affect quiescent IDD current significantly, in this test, various input voltage levels are applied to observe the difference.

The measurements are taken when VIH is 4.5V and VIL is set to 0 V. The current in these input conditions is what we primary concern in IDDQ test.

Figure 4.13 displays all measured values during the IDDQ test. It is apparent that the current value of each IDDQ test vector is stable. Measurements 1-10 are

the testing result of the first IDDQ test vector, the measured IDDQ is about 350 nA. Measurements 11-20 are the testing result of the second IDDQ test vector, the measured IDDQ is about 188 nA. Measurements 21-30 are the testing result of the third IDDQ test vector, the measured IDDQ is about 322 nA. Measurements 31-40 are the testing result of the fourth IDDQ test vector, the measured IDDQ is about 238 nA. Measurements 41-50 are the testing result of the fifth IDDQ test vector, the measured IDDQ is about 214 nA.

DUT	Index	Result	Status	Value	Judge Hi	Judge Lo	Act.Cnt	Pin Name
1	1	PASS	Valid	352.0nA	5.000uA	0.000A	1	VDD
1	2	PASS	Valid	352.0nA	5.000uA	0.000A	1	VDD
1	3	PASS	Valid	352.0nA	5.000uA	0.000A	1	VDD
1	4	PASS	Valid	352.0nA	5.000uA	0.000A	1	VDD
1	5	PASS	Valid	350.0nA	5.000uA	0.000A	1	VDD
1	6	PASS	Valid	348.0nA	5.000uA	0.000A	1	VDD
1	7	PASS	Valid	344.0nA	5.000uA	0.000A	1	VDD
1	8	PASS	Valid	344.0nA	5.000uA	0.000A	1	VDD
1	9	PASS	Valid	342.0nA	5.000uA	0.000A	1	VDD
1	10	PASS	Valid	342.0nA	5.000uA	0.000A	1	VDD
1	11	PASS	Valid	190.0nA	5.000uA	0.000A	1	VDD
1	12	PASS	Valid	190.0nA	5.000uA	0.000A	1	VDD
1	13	PASS	Valid	190.0nA	5.000uA	0.000A	1	VDD
1	14	PASS	Valid	190.0nA	5.000uA	0.000A	1	VDD
1	15	PASS	Valid	188.0nA	5.000uA	0.000A	1	VDD
1	16	PASS	Valid	188.0nA	5.000uA	0.000A	1	VDD
1	17	PASS	Valid	184.0nA	5.000uA	0.000A	1	VDD
1	18	PASS	Valid	184.0nA	5.000uA	0.000A	1	VDD
1	19	PASS	Valid	184.0nA	5.000uA	0.000A	1	VDD
1	20	PASS	Valid	184.0nA	5.000uA	0.000A	1	VDD
1	21	PASS	Valid	322.0nA	5.000uA	0.000A	1	VDD
1	22	PASS	Valid	324.0nA	5.000uA	0.000A	1	VDD
1	23	PASS	Valid	322.0nA	5.000uA	0.000A	1	VDD
1	24	PASS	Valid	322.0nA	5.000uA	0.000A	1	VDD
1	25	PASS	Valid	322.0nA	5.000uA	0.000A	1	VDD
1	26	PASS	Valid	322.0nA	5.000uA	0.000A	1	VDD
1	27	PASS	Valid	324.0nA	5.000uA	0.000A	1	VDD
1	28	PASS	Valid	324.0nA	5.000uA	0.000A	1	VDD
1	29	PASS	Valid	326.0nA	5.000uA	0.000A	1	VDD
1	30	PASS	Valid	326.0nA	5.000uA	0.000A	1	VDD
1	31	PASS	Valid	240.0nA	5.000uA	0.000A	1	VDD
1	32	PASS	Valid	238.0nA	5.000uA	0.000A	1	VDD
1	33	PASS	Valid	238.0nA	5.000uA	0.000A	1	VDD
1	34	PASS	Valid	238.0nA	5.000uA	0.000A	1	VDD
1	35	PASS	Valid	236.0nA	5.000uA	0.000A	1	VDD
1	36	PASS	Valid	236.0nA	5.000uA	0.000A	1	VDD
1	37	PASS	Valid	236.0nA	5.000uA	0.000A	1	VDD
1	38	PASS	Valid	236.0nA	5.000uA	0.000A	1	VDD
1	39	PASS	Valid	238.0nA	5.000uA	0.000A	1	VDD
1	40	PASS	Valid	238.0nA	5.000uA	0.000A	1	VDD
1	41	PASS	Valid	216.0nA	5.000uA	0.000A	1	VDD
1	42	PASS	Valid	216.0nA	5.000uA	0.000A	1	VDD
1	43	PASS	Valid	214.0nA	5.000uA	0.000A	1	VDD
1	44	PASS	Valid	214.0nA	5.000uA	0.000A	1	VDD
1	45	PASS	Valid	214.0nA	5.000uA	0.000A	1	VDD
1	46	PASS	Valid	214.0nA	5.000uA	0.000A	1	VDD
1	47	PASS	Valid	214.0nA	5.000uA	0.000A	1	VDD
1	48	PASS	Valid	214.0nA	5.000uA	0.000A	1	VDD
1	49	PASS	Valid	214.0nA	5.000uA	0.000A	1	VDD
1	50	PASS	Valid	212.0nA	5.000uA	0.000A	1	VDD

Figure 4.13: The IDDQ test result when VIH is 4.5 V and VIL is 0V.

Different input values can cause different internal gates to turn on or turn off. Therefore, IDDQ current is highly dependent on input values.

However, when both VIH and VIL are changed to different values, for instance, VIH is set to 4.5V and VIL is set to 2.1V, then, input voltages are still effective to make this device fully functional. The results are shown in Figure 4.14.

In Figure 4.14, all of IDDQ test vectors that result in much higher current than the currents shown in Figure 4.13.

DUT	Index	Result	Status	Value	Judge Hi	Judge Lo	Act.Cnt	Pin Name
1	1	PASS	Valid	3.400mA	50.00mA	0.000A	1	VDD
1	2	PASS	Valid	3.400mA	50.00mA	0.000A	1	VDD
1	3	PASS	Valid	3.400mA	50.00mA	0.000A	1	VDD
1	4	PASS	Valid	3.400mA	50.00mA	0.000A	1	VDD
1	5	PASS	Valid	3.420mA	50.00mA	0.000A	1	VDD
1	6	PASS	Valid	3.400mA	50.00mA	0.000A	1	VDD
1	7	PASS	Valid	3.400mA	50.00mA	0.000A	1	VDD
1	8	PASS	Valid	3.400mA	50.00mA	0.000A	1	VDD
1	9	PASS	Valid	3.420mA	50.00mA	0.000A	1	VDD
1	10	PASS	Valid	3.400mA	50.00mA	0.000A	1	VDD
1	11	PASS	Valid	4.540mA	50.00mA	0.000A	1	VDD
1	12	PASS	Valid	4.540mA	50.00mA	0.000A	1	VDD
1	13	PASS	Valid	4.540mA	50.00mA	0.000A	1	VDD
1	14	PASS	Valid	4.560mA	50.00mA	0.000A	1	VDD
1	15	PASS	Valid	4.540mA	50.00mA	0.000A	1	VDD
1	16	PASS	Valid	4.540mA	50.00mA	0.000A	1	VDD
1	17	PASS	Valid	4.540mA	50.00mA	0.000A	1	VDD
1	18	PASS	Valid	4.560mA	50.00mA	0.000A	1	VDD
1	19	PASS	Valid	4.540mA	50.00mA	0.000A	1	VDD
1	20	PASS	Valid	4.540mA	50.00mA	0.000A	1	VDD
1	21	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	22	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	23	PASS	Valid	760.0uA	50.00mA	0.000A	1	VDD
1	24	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	25	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	26	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	27	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	28	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	29	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	30	PASS	Valid	740.0uA	50.00mA	0.000A	1	VDD
1	31	PASS	Valid	5.260mA	50.00mA	0.000A	1	VDD
1	32	PASS	Valid	5.260mA	50.00mA	0.000A	1	VDD
1	33	PASS	Valid	5.260mA	50.00mA	0.000A	1	VDD
1	34	PASS	Valid	5.240mA	50.00mA	0.000A	1	VDD
1	35	PASS	Valid	5.260mA	50.00mA	0.000A	1	VDD
1	36	PASS	Valid	5.260mA	50.00mA	0.000A	1	VDD
1	37	PASS	Valid	5.260mA	50.00mA	0.000A	1	VDD
1	38	PASS	Valid	5.260mA	50.00mA	0.000A	1	VDD
1	39	PASS	Valid	5.240mA	50.00mA	0.000A	1	VDD
1	40	PASS	Valid	5.260mA	50.00mA	0.000A	1	VDD
1	41	PASS	Valid	3.580mA	50.00mA	0.000A	1	VDD
1	42	PASS	Valid	3.580mA	50.00mA	0.000A	1	VDD
1	43	PASS	Valid	3.580mA	50.00mA	0.000A	1	VDD
1	44	PASS	Valid	3.560mA	50.00mA	0.000A	1	VDD
1	45	PASS	Valid	3.580mA	50.00mA	0.000A	1	VDD
1	46	PASS	Valid	3.580mA	50.00mA	0.000A	1	VDD
1	47	PASS	Valid	3.560mA	50.00mA	0.000A	1	VDD
1	48	PASS	Valid	3.580mA	50.00mA	0.000A	1	VDD
1	49	PASS	Valid	3.580mA	50.00mA	0.000A	1	VDD
1	50	PASS	Valid	3.580mA	50.00mA	0.000A	1	VDD

Figure 4.14: The IDDQ test result when VIH is 4.5 V and VIL is 2.1 V.

At this time, there are no convenient methods for creating IDDQ faults in tested devices. Therefore, our experiments simply explored how to perform IDDQ test on the ATE.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

In the experiments described in this thesis, we demonstrated how to design and perform DC parametric tests and IDDQ tests on the T2000 test system. In Chapter 2, we introduced the function and the specification of the T2000 tester system DPS500mA and 250MDM modules. The DPS500mA module can perform voltage source current measurement (VSIM) to measure the current flow through the power supply pin of a device. The 250MDM module is used in the experiments mainly for measuring the voltage and current on input signal pins and the output signal pins of a device. Chapter 2 also illustrates the syntax of the test program that needs to be executed on the tester.

DC parametric test is discussed in Chapter 3. In both characterization test and production test, DC parametric test is usually performed first. This chapter describes seven tests that were designed and performed on the six 74-series chips. The tests include contact test, input leakage test, input threshold voltage test, output driving voltage test, gross IDD current test, static IDD current test and supply voltage bump test. Each of the tests used different program setup on the T2000 to measure the required parameters. The main purpose of these tests is verifying that the device operates within its specifications for various electrical parameters.

IDDQ test, discussed in Chapter 4, is also one of the test techniques used in both characterization test and production test. Since IDDQ test is especially useful in detecting bridging faults, it can be applied as a supplemental functional test. The IDDQ test vectors are generated by ATPG tools like FastScan. With help of the

IDDQ test vectors, test coverage can be improved with much fewer test vectors when compared to functional test. For example in the experiment, FastScan only generated five IDDQ test patterns for the device 74HC163 to cover more than 50% of its faults.

In conclusion, DC parametric test and IDDQ test are efficient test methods that reduce the overall test time. DC parametric test and IDDQ test are equally important to functional test. Without DC parametric test and IDDQ test, a functional test may require too many test vectors to ensure a system working properly, which increases the cost of a device.

5.2 Future work

5.2.1 Testing output current of devices

The T2000 test system has the ability to measure the currents of output signal pins when a DUT has been loaded. However, it requires the test program to set up properly to avoid errors. Errors were shown in the GUI when we tried to test the output currents in some situations. If this test is working properly, it can measure the current of outputs so that compares with specification.

5.2.2 Emulating IDDQ faults in Field Programmable Gate Arrays (FPGA)

An FPGA is an integrated circuit designed to be configured by a customer or a designer after manufacturing. Stuck-at faults can be easily emulated in an FPGA. For Xilinx FPGAs, the Xilinx Design Language (XDL) and FPGA Editor can be used to configure internal wires and Look-Up Tables (LUT) manually. Therefore, the FPGA configuration can be modified to emulate some IDDQ fault in an FPGA by employing XDL and FPGA Editor. However, we were unable to emulate bridging faults in FPGA in this experiment.

There are a number of papers that refer to fault injection for FPGA [27] [28].

Bibliography

[1] "The first monolithic integrated circuits", http://homepages.nildram.co.uk/ w̃ylie/ICs/monolith.htm.(accessed on June 1,2015).

[2] M.L. Bushnell and V. D. Agrawal, "Essentials of Electronic Testing for Digital, Memory & Mixed-Signal VLSI Circuits", Boston, Springer, 2000.

[3] "Diode i-v Characteristics", http://people.seas.harvard.edu/ jones/es154/ lectures/lecture_2/diode_characteristics/diode_characteristics.html.(accessed on June 1,2015).

[4] P. Venkataramani, "Reducing ATE Test Time by Voltage and Frequency Scaling" , 2014.

[5] I.A. Grout, "Integrated Circuit Test Engineering Modern Techniques", Springer, pp7.

[6] TERADYNE, "Fundamental Of Testing On ATE" , 2007.

[7] ADVANTEST Corporation, "AT Standard Test Class Programming Manual" , 2002.

[8] H.T. Vierhaus, W. Meyer and U.Glaser, "CMOS bridges and resistive transistor faults: IDDQ versus delay effects," *Proc. IEEE Int. Test Conference*, pp. 83–91, 1993.

[9] ADVANTEST Corporation, "T2000 Hardware Architecture Overview v2.3-SWB".

[10] ADVANTEST Corporation, "T2000 DEVICE POWER SUPPLIES (DP-S)MODULES COURSE STUDENT WORKBOOK" , 2008.

[11] ADVANTEST Corporation, "DM250Mbps Programming Manual" , 2002.

[12] ADVANTEST Corporation, "T2000 128ch 250Mbps Digital Module Training Text" , 2008.

[13] ADVANTEST Corporation, "OTPL Software Overview v2.3-SWB".

[14] J.M Chang, "A Study of the Optimizationof DC Parametric Tests," *Proc. IEEE Int. Test Conference*, pp. 478–487, 1990.

[15] NXP Semiconductors, "74HC163; 74HCT163 datasheet" , 2014.

[16] Philips Semiconductors, "74HC393; 74HCT393 datasheet" , 2014.

[17] "DC Test Theory," http://data.eefocus.com/myspace/1/6351/bbs/1185160533/5f75a39d.pdf. (accessed on June 1,2015).

[18] F. Zarrinfar and R. Rajsuman, "Automated Iddq testing from CAD to manufacturing," *Proc. IEEE Int. Workshop on Iddq Testing*, pp. 48–51, 1995.

[19] R. Rajsuman, "Iddq testing for CMOS VLSI," *Proceedings of the IEEE*, vol. 88, no. 4, pp. 544–568, 2000.

[20] W. Maly, "Realistic fault modeling for VLSI testing," *IEEE Design Auto. Conf.*, pp. 173–180, 1987.

[21] Y. K. Malaiya, A. P. Jayasumana, and R. Rajsuman, "A detailed examination of bridging faults," *Int. Conf. on Computer Design*, pp. 78–81, 1986.

[22] R. R. Montanes, E. M. J. G. Bruls, and J. Figueras, "Bridging defect resistance measurement in a CMOS process," *Int. Test Conf.*, pp. 892–899, 1992.

[23] M. Favalli, M. Dalpasso, P. Olivo, and B. Ricco, "Analysis of resistive bridging fault detection in BiCMOS digital ICs," *IEEE Trans. VLSI*, pp. 342–355, 1993.

[24] M. Syrzycki, "Modeling of gate oxide shorts in MOS transistors," *IEEE Trans. CAD*, pp. 193–202, 1989.

[25] Chen-Wei Lin, Mango C.-T. Chao, and Chih-Chieh Hsu, "Novel Circuit-Level Model for Gate Oxide Short and its Testing Method in SRAMs," *IEEE Trans. VLSI Systems*, vol. 22, no. 6, pp. 1294–1307, 2014.

[26] Mentor Graphics, "Scan and ATPG Process Guide (DFTAdvisor and Tessent FastScan)" , 2011.

[27] T. Slaughter and C. Stroud, "Fault injection emulation for field programmable analog arrays," *Proc. ITCOM*, 2001.

[28] D. de Andres, J.C. Ruiz, D. Gil and P. Gil, "Fast Emulation of Permanent Faults in VLSI Systems," *International Conference Field Programmable Logic and Applications, 2006*, pp. 1–6, 2006.