**Estimation of maximum frequency of operation**

**(Same guidelines apply to the minimum voltage problem)**

Aim: To estimate the maximum frequency of operation of an SN74LS393 IC using the Advantest T2000 tester.

Material provided:

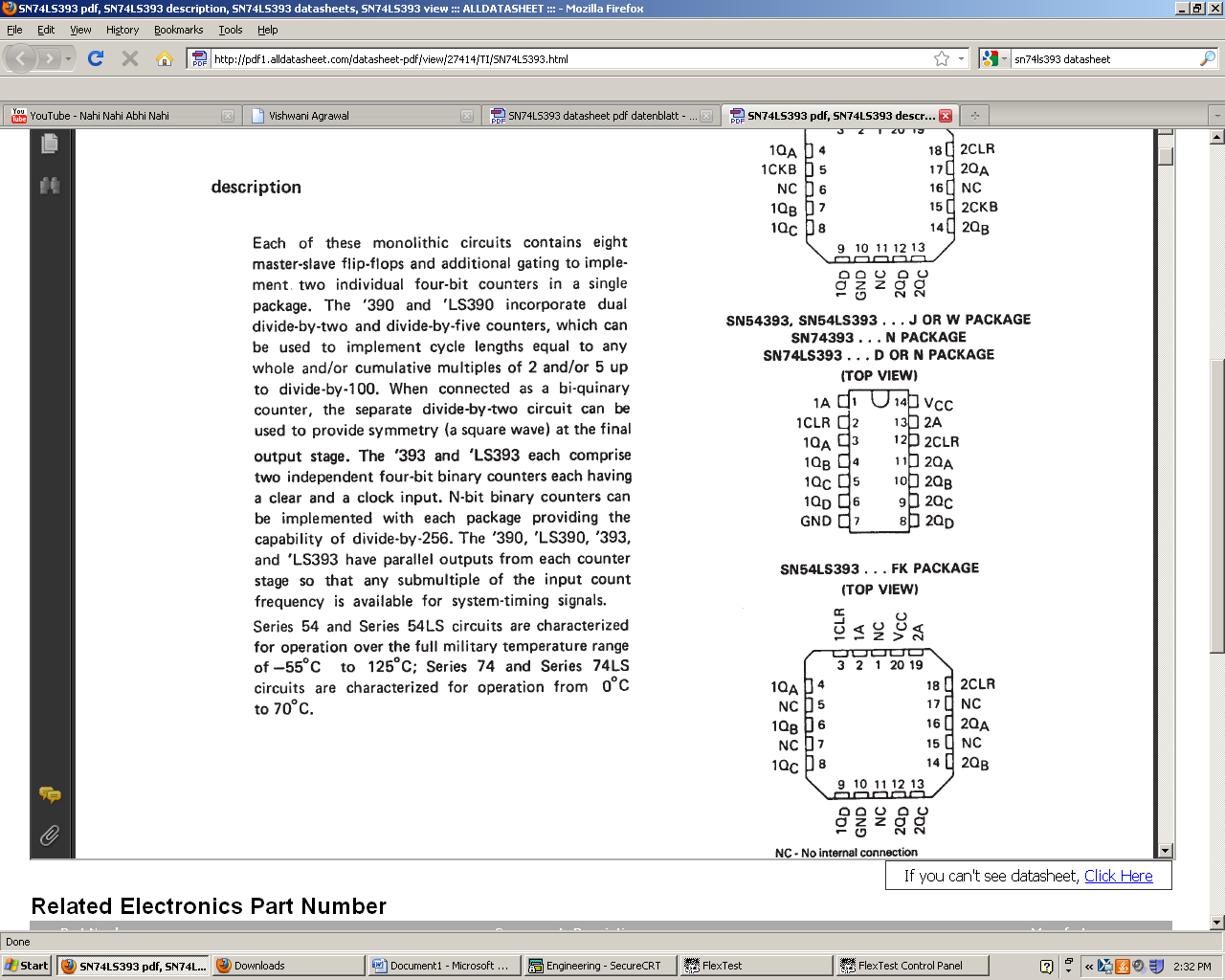
1. Approximate VHDL model of the circuitry in the SN74LS393 IC.

2. Test program required to test the chip on the T2000 tester.

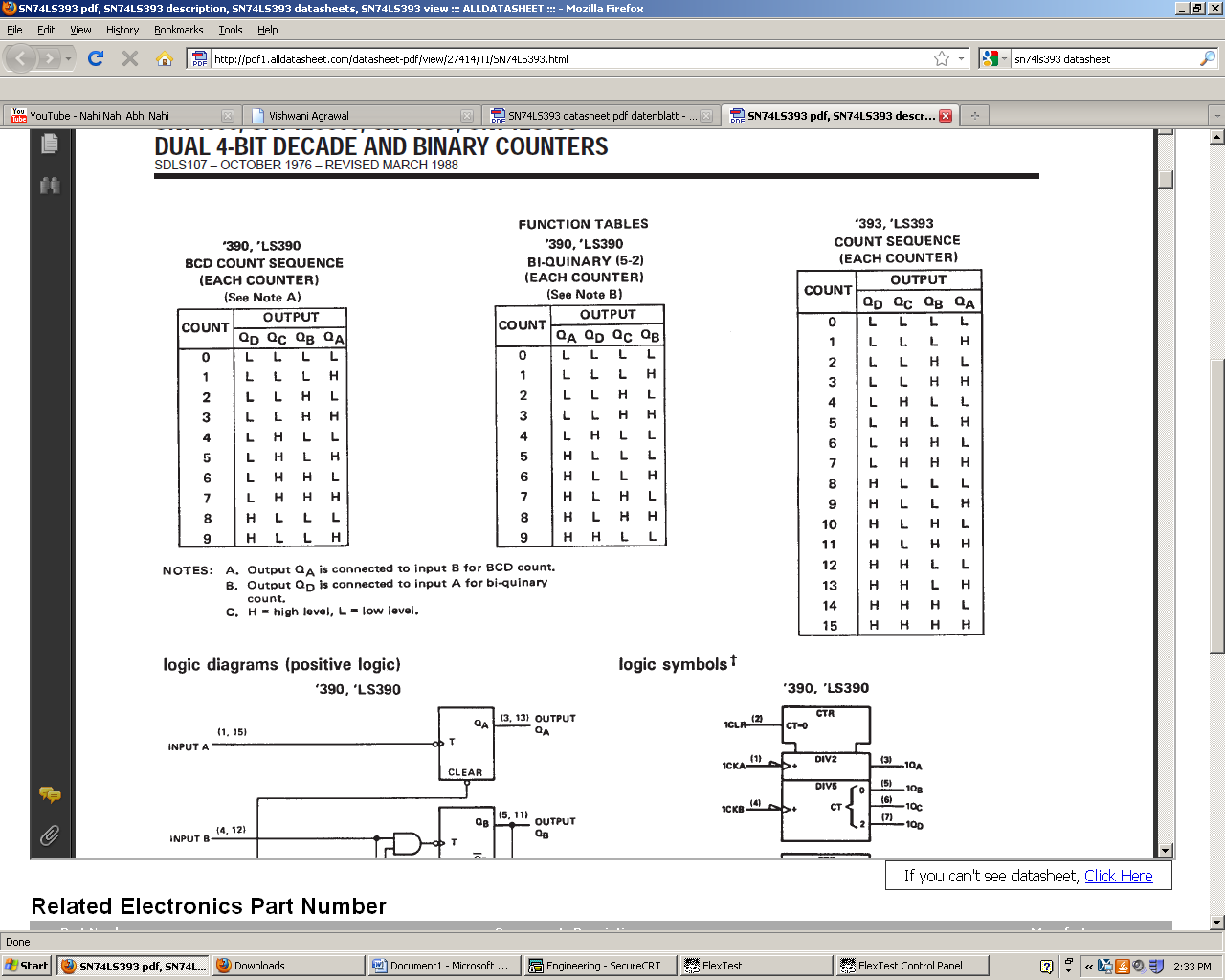
Description:

The SN74LS393 IC comprises two independent four-bit binary counters each having a clear (CLR) and a clock (A) input. It has parallel outputs (QA, QB, QC, QD) from each counter stage.

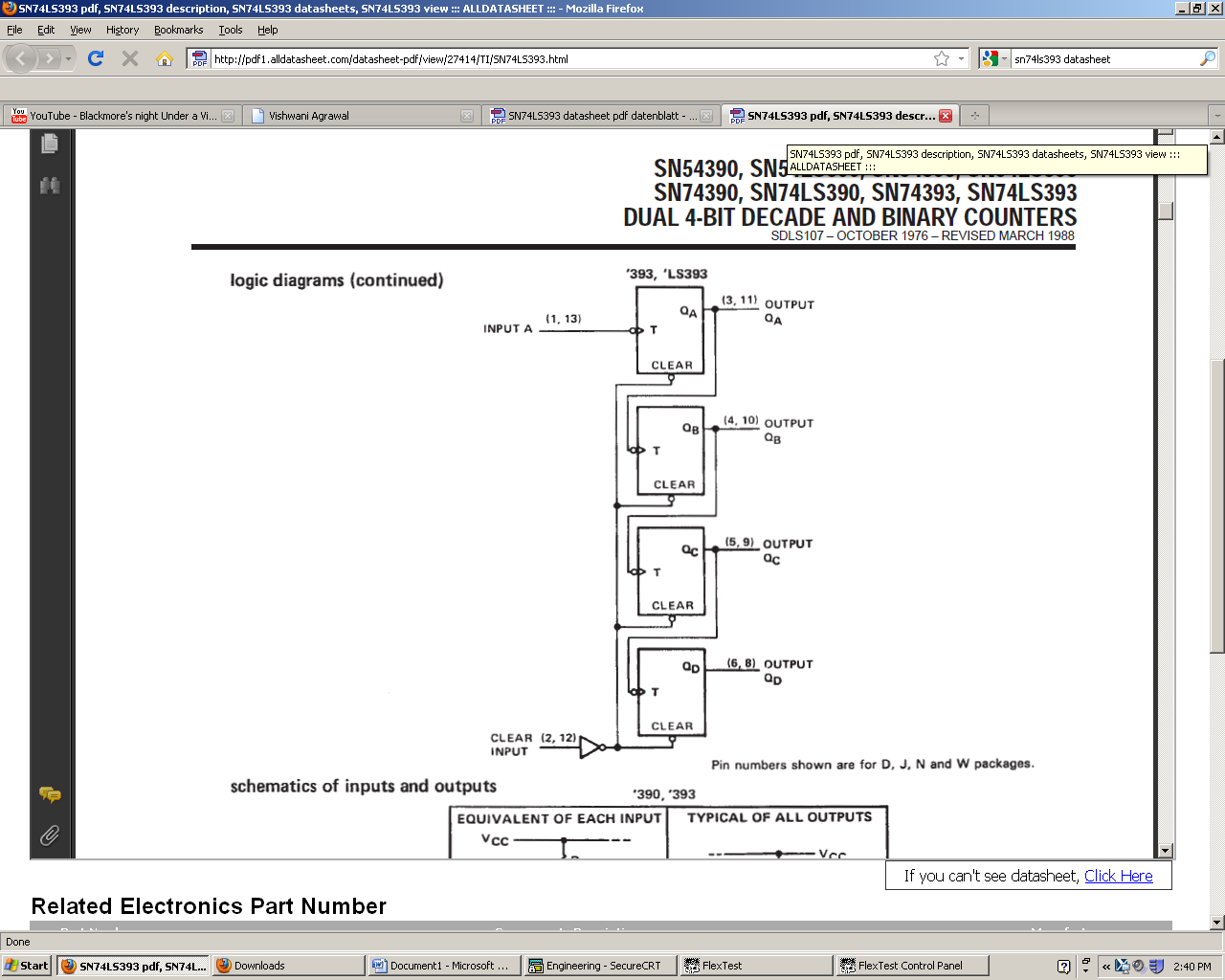
Pin Diagram:



Count Sequence of each counter:



Logic Diagram of each counter:



Understanding the test program:

The test program is written to apply test vectors at varying clock frequencies.

The test program resides in the directory:

D:\T2000Install\UserSDK\examples\OAI\SimpleTPL

It has the following sub directories:

1. OTPLSrc

2. TestClasses

3. TestPlans

4. PLists

5. Patterns

The following files should be present in the OTPLSrc directory:

1. asicbins.bdefs

2. DiagPBSpec.spec: This file holds information about voltage and timing specifications of the chip

3. level.lvl: This file specifies what voltage should be applied to different pins

4. timing.tim: This file defines the timing waveforms eg. NRZ, RZ, etc.

5. timingmap.tmap: This file maps the timing information to the different pins

6. testcondition.tcg: This file maps different tests with the timing information specified in the .tim and .tmap files

7. testplan.tpl: This file defines the different tests and specifies the order in which they should be executed

8. uservar.usrv: This file contains the user variables used in the other files

The following files should be present in the TestPlans directory:

1. socket.soc: This file specifies which pin on the IC is connected to which tester channel on the performance board on the tester head

2. pindesc.pin: This file is used to categorize pins into groups. For instance, all input pins are grouped into the ‘inpins’ group and all the output pins are grouped into the ‘outpins’ group

The following files should be present in the Patterns directory:

1. setup.pxr

2. pat1.pat: This file contains the test vectors to be applied on the IC

The following file should be present in the PLists directory:

1. Pattern.plist: This file indicates the names of the pattern files to be used for the test

The test program is compiled and placed in the TestPlans directory as follows: (Already performed. Need not be repeated.)

1. Open DOS command prompt.

**2.** **C:> D:**

**3.** **D:>cd T2000Install/UserSDK/examples/OAI/SimpleTPL/OTPLOutput**

**4.** **set PATH=%PATH%;C:\Program Files\Microsoft Visual Studio 8\Common7\IDE**

**5.** **set v=%VS80COMNTOOLS%vsvars32.bat**

6. Copy all files from OTPLSrc directory to OTPLOutput directory

**7. oai\_occ testplan.tpl**

**8.** **devenv OTPLTestPlan.vcproj /build Release**

9. Copy OTPLTestPlan.dll from OTPLOutput/Release directory to TestPlans directory

Procedure:

Three different pattern files are used to estimate the frequency of operation:

1. Random vectors obtained by analyzing the circuit: The critical path should be activated with the random vectors

2. Vectors obtained to test stuck-at faults

3. Vectors obtained to test transition faults

Generating (stuck-at and transition) test vectors using MentorGraphics FlexTest ATPG:

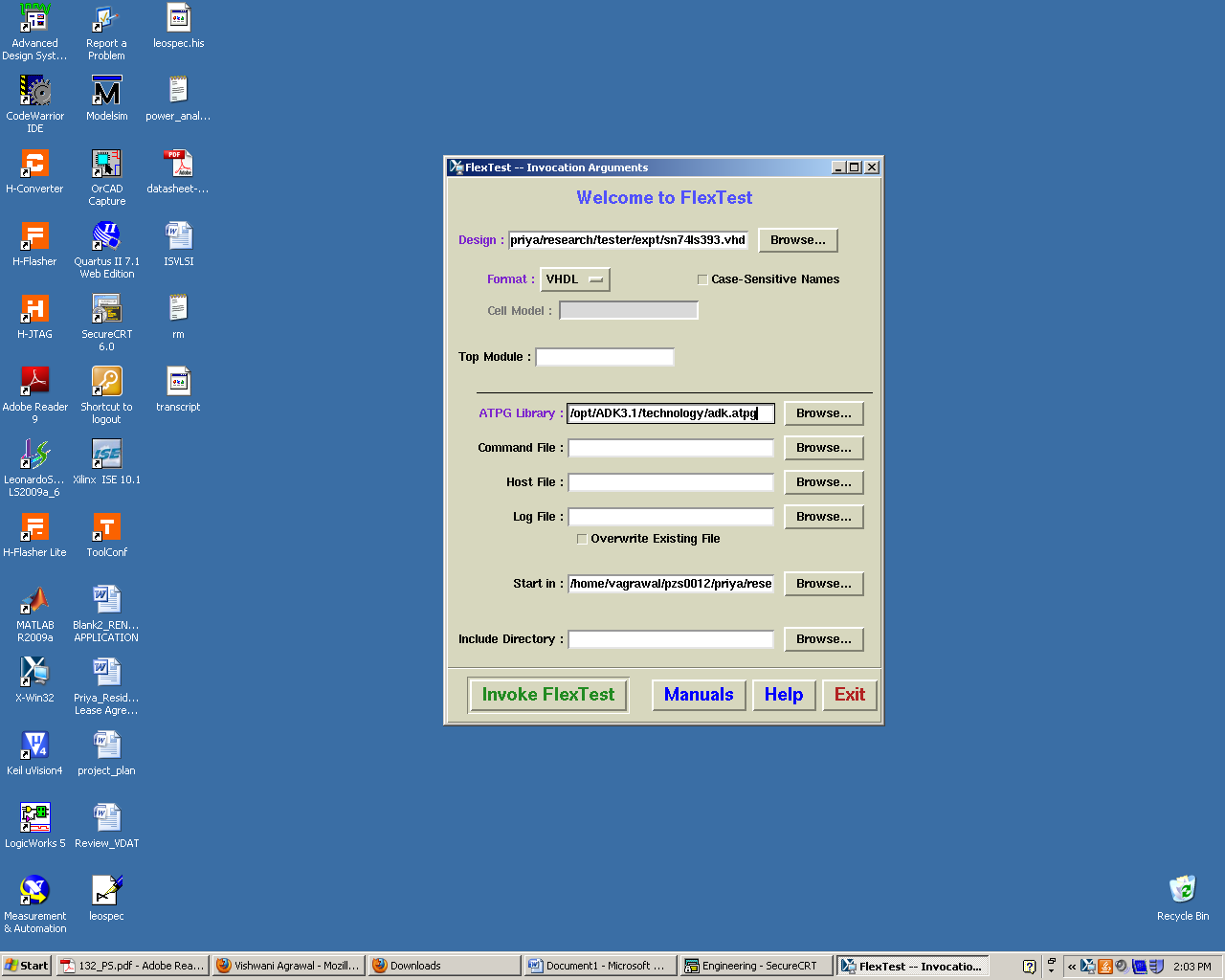
i) Open X-Win 32 (available on the Desktop in the computers in Lab 310)

ii) Open SecureCRT (available on the Desktop in the computers in Lab 310) and sign in using AU user-id and password. Connect to a UNIX server eg. coyote

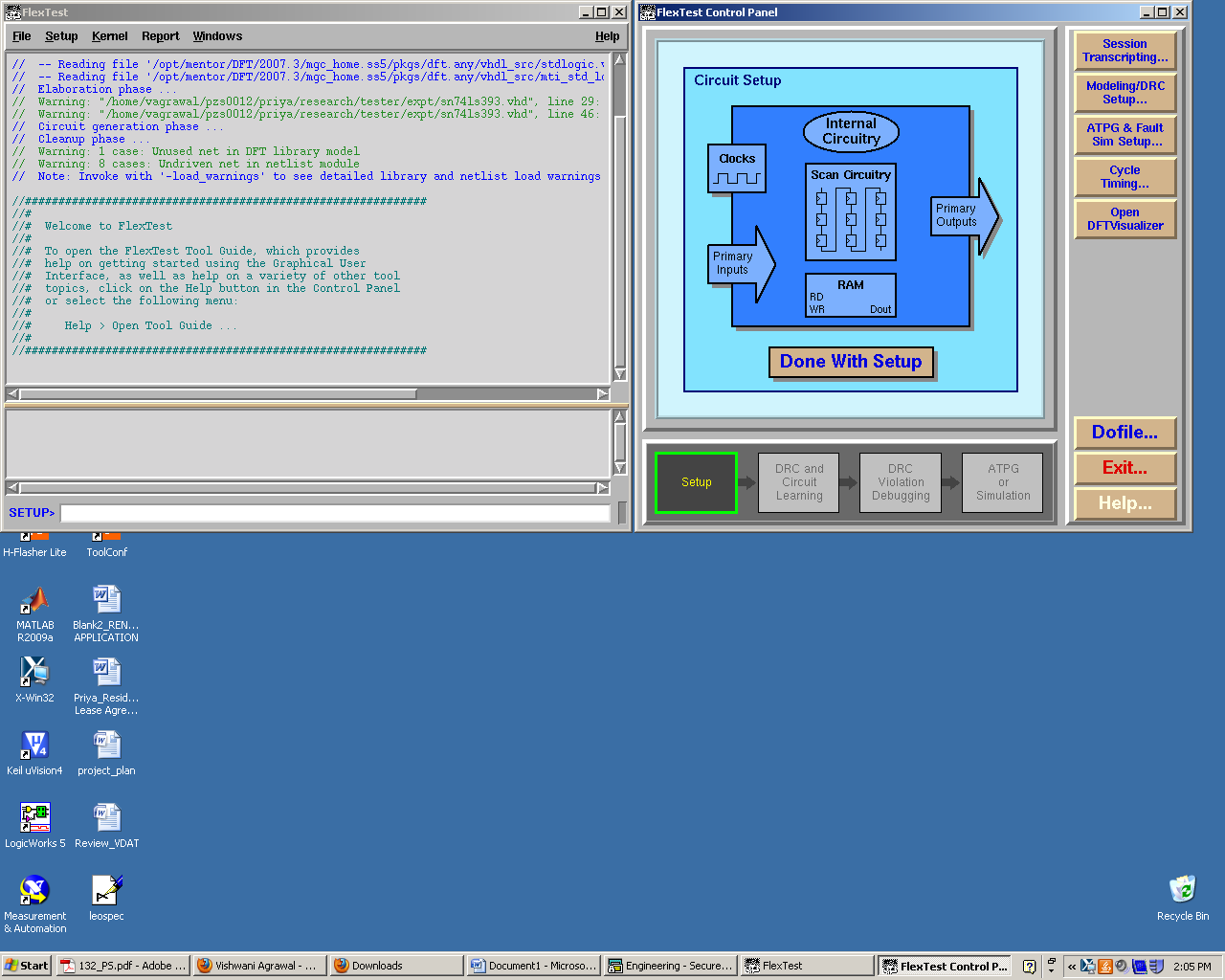
iii) Change the working directory to the directory in which you have placed the vhdl file provided to you

eg. cd /home/vagrawal/pzs0012

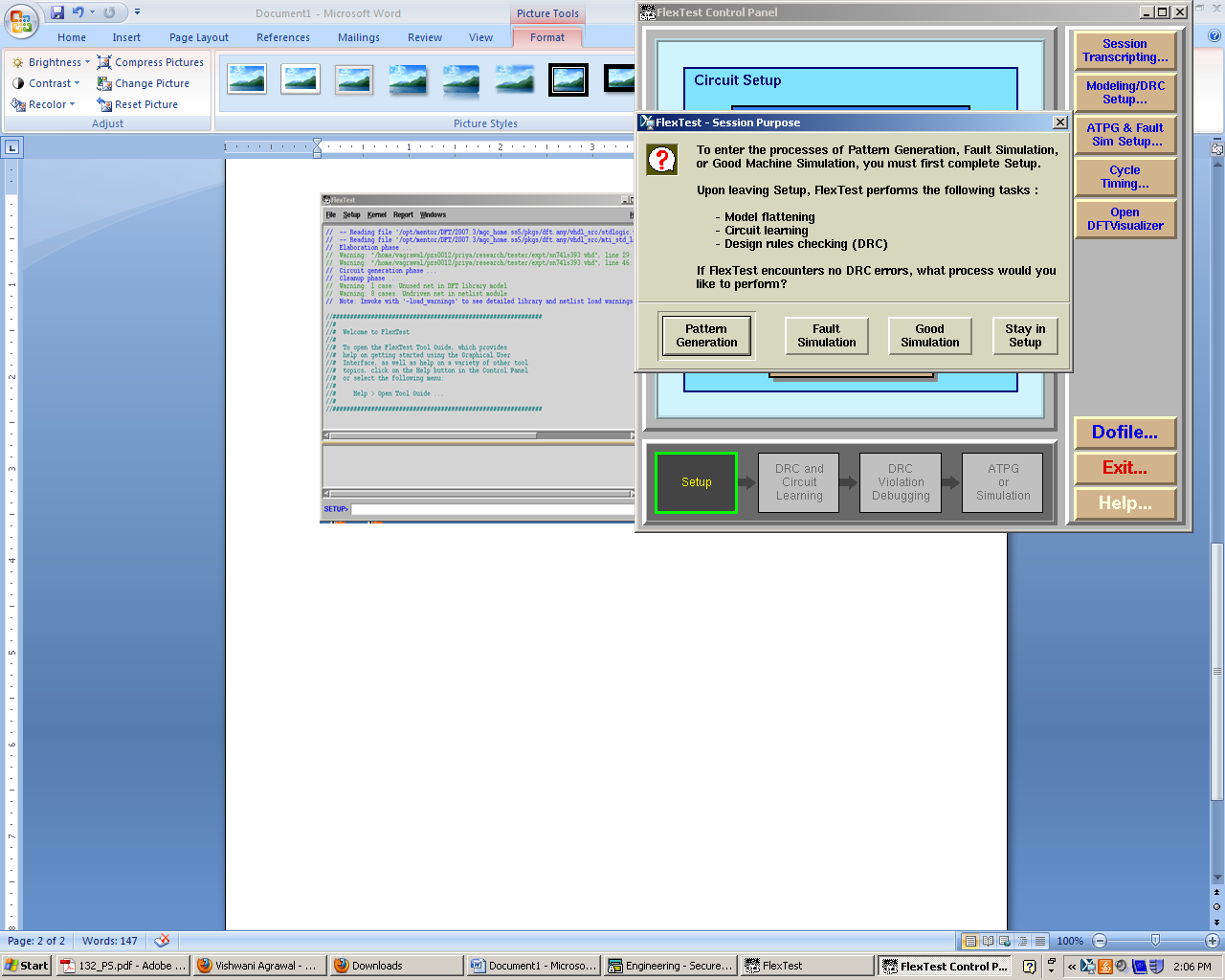
iv) Invoke flextest using the command flextest. Choose the vhdl design and ATPG library ($ADK/technology/adk.atpg) and click on ‘Invoke FlexTest’



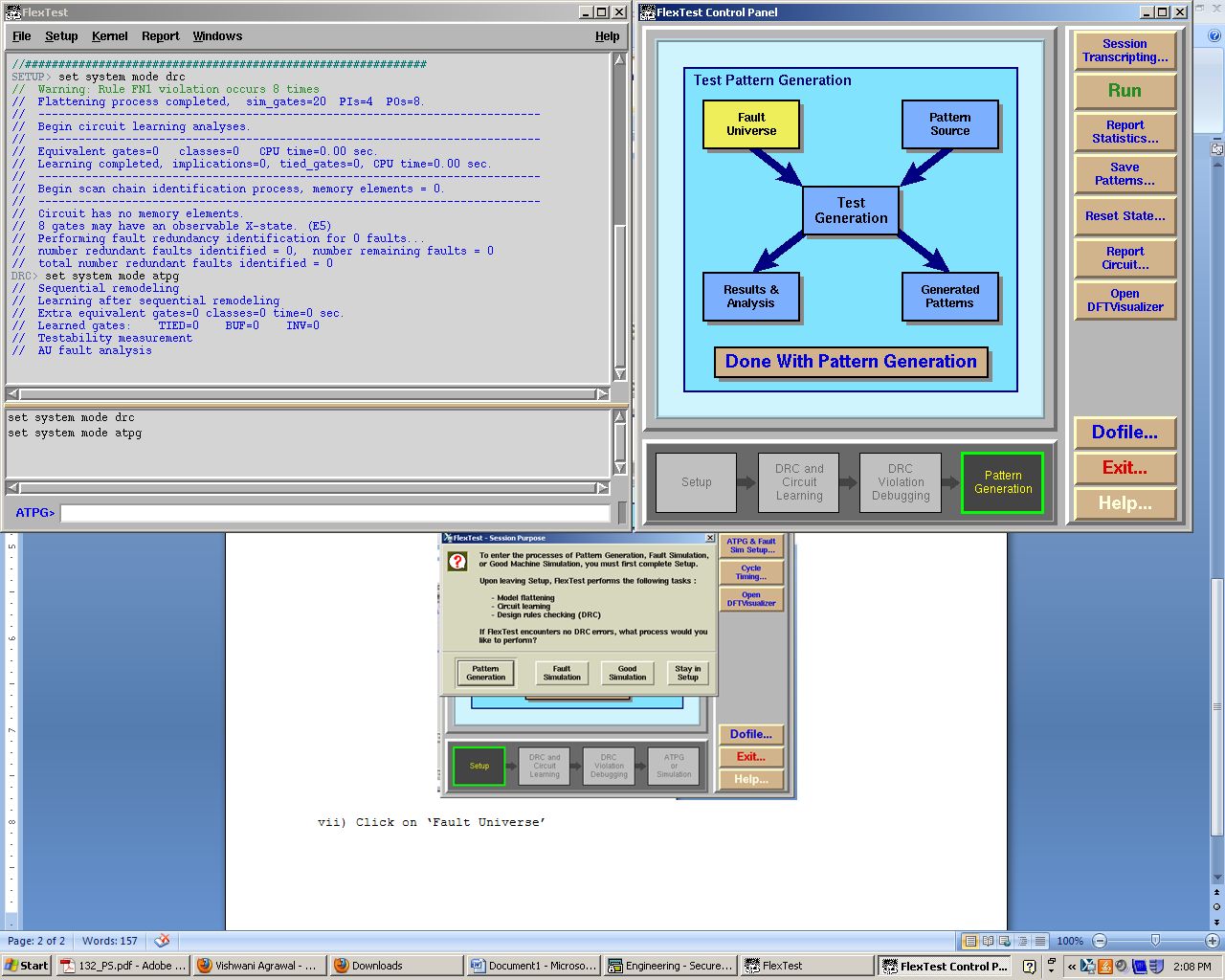
v) Click on ‘Done with setup’



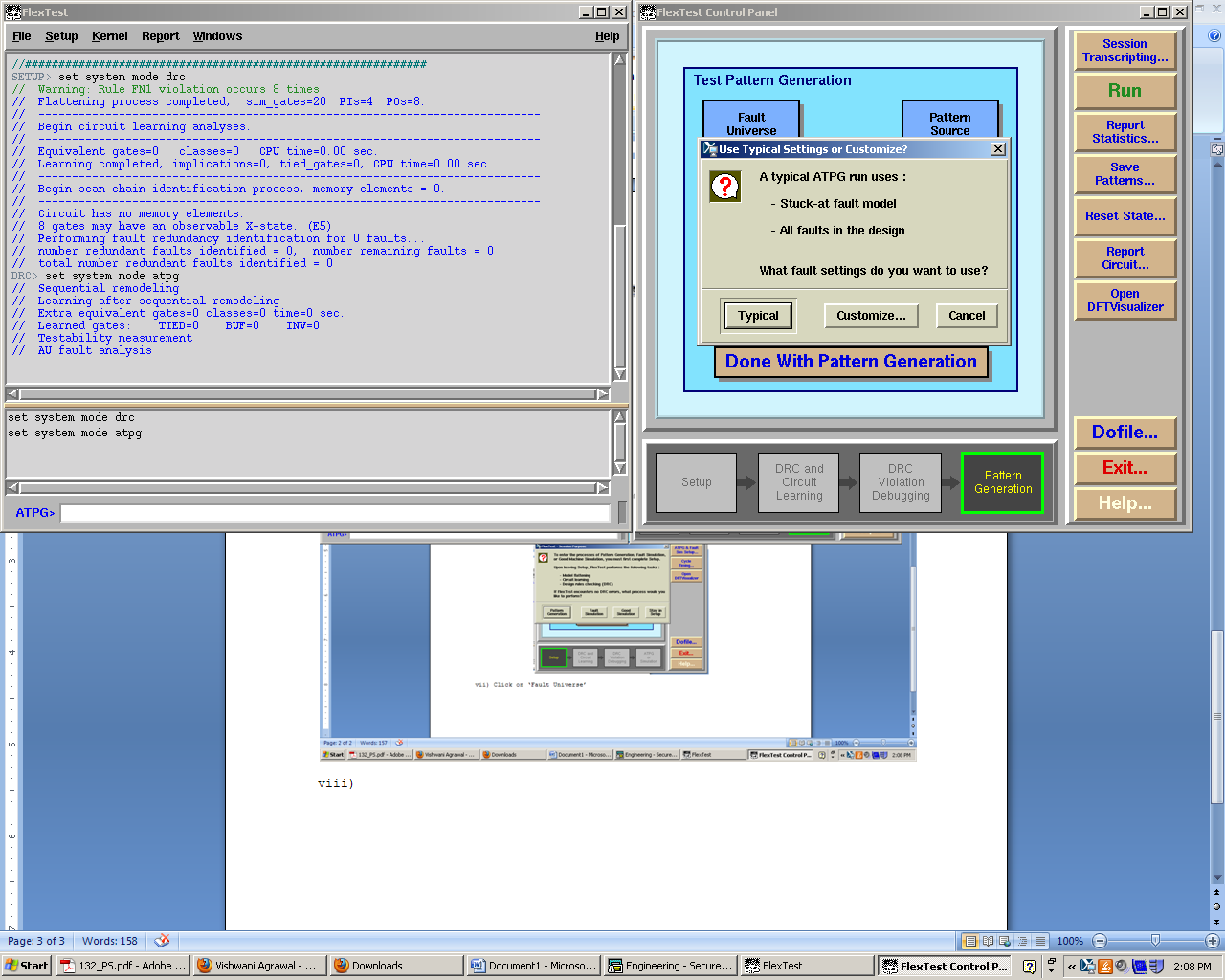
vi) Click on ‘Pattern Generation’



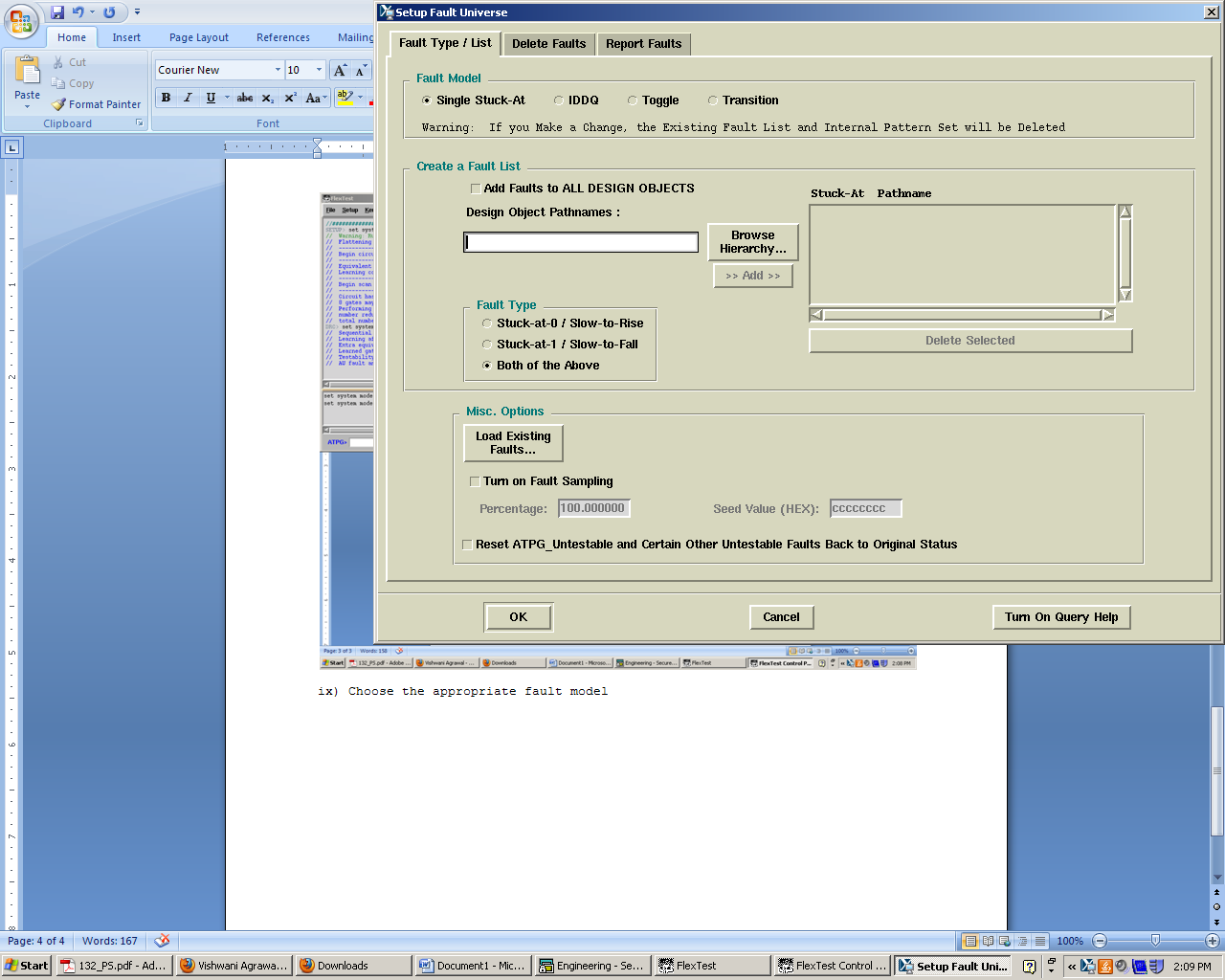
vii) Click on ‘Fault Universe’



viii) Click on ‘Customize’



ix) Choose the appropriate fault model and click on ‘OK’



x) Click on ‘Run’ and choose ‘add all faults’ or ‘custom fault list’ as necessary.

xi) Once the patterns are generated, click on ‘Save patterns’ to save the patterns in the required location

Converting the patterns to tester readable format:

Order of input pins: a1 a2 clr1 clr2

Order of output pins: qa1 qb1 qc1 qd1 qa2 qb2 qc2 qd2

If patterns obtained from FlexTest are in the following format:

CYCLE = 0;

FORCE "ibus" "0111" 0;

MEASURE "obus\_1" "00000000" 1;

CYCLE = 1;

FORCE "ibus" "0100" 0;

MEASURE "obus\_1" "00000000" 1;

or if random patterns to activate critical path are as follows:

(a1,a2,clr1,clr2,qa1,qb1,qc1,qd1,qa2,qb2,qc2,qd2) = (011100000000)

(a1,a2,clr1,clr2,qa1,qb1,qc1,qd1,qa2,qb2,qc2,qd2) = (010000000000)

they should be rewritten as:

NOP { V { inpins=0111; outpins=LLLLLLLL; } W {allpins=wfs1;}}

NOP { V { inpins=0100; outpins=LLLLLLLL; } }

NOP { V { inpins=0111; outpins=LLLLLLLL; } W {allpins=wfs2;}}

NOP { V { inpins=0100; outpins=LLLLLLLL; } }

NOP { V { inpins=0111; outpins=LLLLLLLL; } W {allpins=wfs3;}}

NOP { V { inpins=0100; outpins=LLLLLLLL; } }

NOP { V { inpins=0111; outpins=LLLLLLLL; } W {allpins=wfs4;}}

EXIT { V { inpins=0100; outpins=LLLLLLLL; } }

Running the test on the tester:

1. Once the tester is switched on, it needs a warm-up time of 30 minutes.

2. Go to directory:

D:/T2000Install/UserSDK/examples/OAI/SimpleTPL/Patterns

3. Open the pat1.pat file and paste the random patterns (in the format shown in the previous section) that activate the critical path.

4. Open the DOS command prompt and type in the following commands:

C:>D:

D:>cd T2000Install/UserSDK/examples/OAI/SimpleTPL/Patterns

oai\_patcom –s socket.soc pat1.pat

t2kctrl start

5. This should open a control panel. Choose File/Load Test Plan and browse for the OTPLTestPlan.dll file in D:/T2000Install/UserSDK/examples/OAI/SimpleTPL/TestPlans and click OK

6. Choose ‘Flow Editor’ from the tools tab.

7. Right click on the different test icons (There should be 30 test icons, each for running the test patterns with a time period range of 10 ns eg. 300 ns to 290 ns. Each test runs the test patterns at 4 different frequencies in steps of 2.5 ns eg. at 300 ns, 297.5ns, 295 ns and 292.5 ns .) and choose ‘Suspend on Fail’

8. Click on Execute flow

9. When a test passes, it is highlighted in green. If it fails, it is highlighted in red. For instance, if the test for 200 to 190 ns is highlighted in red, it means that the chip has to be operated at a time period greater than 200 ns. To know the time period of failure to an accuracy of 2.5 ns,

i) Double click on the test icon

ii) Double click on ‘DiagPBPat’

iii) Right click on ‘pat1’ and choose ‘Pattern Editor’

iv) Run the patterns again. The patterns that failed are highlighted in red. By examining the pattern number, the exact time period at which the IC fails can be found to an accuracy of 2.5 ns

10. Repeat the above steps for the stuck-at and transition vectors

Interpreting the results:

Note down the time periods at which the IC fails for the different pattern sets. Compare them. If they are different, try to reason out why they are different. Which pattern set is most effective in estimating the frequency of operation? Why? Does this confirm to the theoretical expectation?