ADVANTEST “T2000 GS” IC Tester

VLSI Design & Test Seminar
Victor P. Nelson
Presentation outline

• IC testing process
• Tester architecture
• Device test fixture
• Test plan elements
• Creation of test vectors
• Running tests
IC testing process

For each test vector:
1. Apply test vector to DUT input pins
2. Activate clock
3. Sample DUT outputs
4. Compare sampled to expected outputs

Input Test Vector

Device Under Test (DUT)

Sampled Outputs

Expected Outputs

Detected Error

Test period $T_p$

Apply clock pulse

Apply input vector

Check outputs
Other IC tests

• Contact test - test chips’ package pin opens/shorts
• Input pin DC parametric tests (signal pins)
  • Input leakage current
  • Input threshold voltage
• Output pin DC parametric tests (signal pins)
  • Driving voltage test
• IDD test (power supply pins)
  • Test for VDD current
  • Gross, Static, Dynamic and IDDQ
• AC parametric tests
  • Test quality of output signal and signal timing parameters
ADVANTEST T2000 SoC Test Systems

Source: https://www.advantest.com/products/ic-test-systems/t2000
T2000 System Scalability

Source: https://www.advantest.com/products/ic-test-systems/t2000
ADVANTEST T2000 GS Test System

GS Mainframe:
- system controller
- site controller

Performance board:
• DUT socket pins connect to module channels via HIFIX (High Fidelity Tester Access Fixture)

Test Head:
• 13 module slots
  - I/O pin electronics
  - power supplies
• HIFIX
• Optional:
  - handler (volume test)
  - wafer prober
  - manipulator
T2000 GS computing architecture

System controller
- User GUI to the test system to develop and store test plans and patterns
- Sends commands to site controller

Site Controller
- One per DUT *(we have only one)*
- Executes test plans on the DUT
  - Controls test instrument modules
  - Returns results to user

Bus Switch
- Configured by a socket file
- Connects site controller to test modules

Test Instrument Modules
- I/O pin electronics
- Power supplies
Test instrument modules
(up to 12 in T2000 GS test head)

- 800MDM (800/500Mbps)
  - 250MDMA (250Mbps)
    - 128 I/O pins per module
    - SCAN Test Function
    - Memory Test Function
    - High Voltage Digital Pin
    - Source Synchronous Tests (800MDM only)
    - Multi Time Domain
    - Histogram Engine (250MDMA only)

- 12GWSGA
  - 4VSG/4VSA x 4MUX per module
  - VSG: 6GHz, VSA: 12GHz
  - High C/N Mode and High Speed Setting Mode

- 6.5GDM (6.5Gbps)
  - Differential 8I + 8O per module
  - Header Hunt Function for Non-deterministic Latency
  - Clock Data Recovery for Embedded Clock Interfaces
  - Clock Tracking Mode for Source Synchronous Interfaces

Device Power Supply

- LCDPS
  - 8ch per module
  - 4A Maximum Output

- HCDPS
  - 4ch per module
  - 16A Maximum Output

- DPS500mA
  - 32ch per module
  - 500mA Maximum Output

AAWG for Audio
  - Audio AWG
    - 8 AWG ch per module (4 L/R pair)
    - 24bit/200Ksps

Audio DGT
  - 8 DGT ch per module (4 L/R pair)
  - 18bit/820Ksps (20bit/51Ksps)

BBWG for BaseBand, Video
  - BaseBand AWG
    - 8 AWG ch per module (4 I/Q pair)
    - 16bit/400Mps

BaseBand DGT
  - 8 DGT ch per module (4 I/Q pair)
  - 16bit/128Mps
  - Hardware DSP Engine

PMU32 for ADC/DAC
  - 32 DC pins per module
  - High Speed DC Linearity Test
  - Wide Coverage of Voltage Range (±0.7V to ±40V)
  - 200mA Maximum Output
### 250MDMA Pin Electronics - Specifications

#### Pin Drivers

<table>
<thead>
<tr>
<th>Voltage Range</th>
<th>250MDMA Specification Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VIH</td>
<td>-1.15 ~ +7.0V</td>
</tr>
<tr>
<td>VIL</td>
<td>-1.25 ~ +5.9V</td>
</tr>
<tr>
<td>Voltage Amplitude</td>
<td>(VIH-VIL)</td>
</tr>
<tr>
<td>Voltage Resolution</td>
<td></td>
</tr>
</tbody>
</table>

**Transition Time**
- 20% ~ 80%: 1.2nS @ 3V
- 20% ~ 80%: 1nS @ 1V

**Timing Edge Resolution:** 7.8125ps

**Number of Timing Edges:** 6 per pin (4 drive/2 compare)

#### Pin Comparators

<table>
<thead>
<tr>
<th>Voltage Range</th>
<th>250MDMA Specification Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOH</td>
<td>-1.25 ~ +6.75V</td>
</tr>
<tr>
<td>VOL</td>
<td>-1.25V ~ +6.75V</td>
</tr>
<tr>
<td>Minimum Voltage (VOH-VOL)</td>
<td></td>
</tr>
<tr>
<td>Voltage Resolution</td>
<td></td>
</tr>
</tbody>
</table>

---

*January 18, 2018*
250MDMA pattern generator and frame processor
DPS500mA (power supply) pin electronics

- Programmed voltage
- Output voltage (less R1/RR drops)
- Detect output current for “clamp”
- Detect output current to measure $I_{DD}/I_{DDQ}$
- Adjust output, based on sensed value
- Sense value
- Force value
- Capture measurements

January 18, 2018
VLSI D&T Seminar - Nelson
Auburn T2000 performance board (test fixture for devices to be tested)

Replaced by one DIP48 ZIF socket
Performance board IC sockets

Power supply connections

To Module Connector
1003.1..32
1003.33..64

To Module Connector
1003.1..24
1003.25..48

To Module Connector
2003.1..32
2003.33..64
Connecting DUT pins to 250MDMA/DPS500ma

DUT: FPGA module

DUT pins to 250MDMA channels

Shorting plug

GND 250MDM channel DUT socket pin GND

G S S G

DUT pin to DPS500mA channel

To DPS500mA module

GND

G S S G

GND 250MDM channel DUT socket pin GND

Shorting plug removed

January 18, 2018 VLSI D&T Seminar - Nelson
TSS (T2000 System Software) Structure

- The primary User Interface with the Tester.
- Allows communication between GUI and User Tools, Test Plan and Test Classes on the Site Controller(s).
- Test Plan resides here along with the Test Classes needed for device test. Interfaces to specific Framework Classes ultimately with Std. Interfaces that translate to module-specific commands.
- Software layers that control the H/W modules from API’s and Functions implemented by the Test Class and Test Plan.
- Module Backplane provides optical/ electrical I/F to individual test modules.
Test Plan

• A test plan (program) is written by a test engineer.
  • Defines the test flow
  • Executes on the Site Controller
    • Controls the modules to test the device
  • Written in OTPL
    • Open Architecture Test Programming Language
  • Uses framework classes
    • Test, Level, Timing, DC Parametrics, User-supplied
  • Configures hardware using standard interfaces
    • test plans interact with common test system hardware components and other test-related objects.
OPTL Test Plan Structure (multiple files)

Pin Description

Socket Def

Pattern 1
Pattern 2
Pattern 3

Specification Sets (min, typ, max)

Levels

Timing

Selector = Min, typ Or max

Test Condition Group

Test Pre-Header

Test 1

Pre-Header

Pattern 1

Pattern 2

Pattern 3

Plist

Specification

Levels

Timing

Pattern 1

Pattern 2

Pattern 3

Plist

.Spec

.Lvl

.Tim

.Tmap

.Tpl

.Pat

.Soc

.Pin

.Plist

-January 18, 2018
Example – Test plan for a 74LS393 dual 4-bit binary counter (14-pin DIP package)
74LS393 “pin description file” (.pin)

DUT pin names and pin groups for timing domains & patterns

Version 1.0.0;
PinDescription
{
   Resource AT.Digital.dpin
   {
      A1;
      CLR1;
      QA1;
      QB1;
      QC1;
      QD1;
      A2;
      CLR2;
      QA2;
      QB2;
      QC2;
      QD2;
   }
   Group inpins1
   {
      A1, A2
   }
   Group inpins2
   {
      CLR1, CLR2
   }
   Group outpins1
   {
      QA1, QB1, QC1, QD1
   }
   Group outpins2
   {
      QA2, QB2, QC2, QD2
   }
   Domain default
   {
      allpins
   }
   DomainGroup DefaultDG
   {
      default
   }
}

Resource dps500mA
{
   VDD;
}

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

Pins controlled/observed as groups in the test plan

All individual pins

(OTPL requires strict formatting)
74LS393 “socket file” (.soc)
Specify DUT pin connections to module channels

Version 1.0.0;
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   1010.2;
            }
            Resource moduletrigger
            {
                PMDTR0 1003.129;
                PMDTR1 1003.130;
                PMDTR2 2003.131;
                PMDTR3 2003.132;
            }
        }
    }
}

Resource dps500mA
{
    VDD   1010.2;
}

DPS500mA connector: 1010.1 .. 32

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

250MDMA connectors: 1003.1 .. 64 2003.1 .. 64

connector.channel

Connector 1003 -> left 64-pin ZIF socket & 48-pin ZIF socket
Connector 2003 -> right 64-pin ZIF socket
74LS393 device “specification file” (.spec)
Device voltage/current specifications (from its data sheet)

Version 1.0;
Import uservar.usrv;
SpecificationSet functional_Specs(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 = 5V;
    Voltage vil = 0V;
    Voltage voh = 2.5V, 3.4V, 3.4V;
    Voltage vol = 0.35V, 0.35V, 0.5V;
}

A “test condition” will select min, typ, or max values
Tester output levels applied to DUT inputs
Thresholds on tester inputs from DUT outputs

Tester  vil/vih  DUT  vol/voh  Tester
Levels file (.lvl)

Voltagess/currents for DUT signal pins,
Force voltages for DUT power supply pins.

Version 1.0;
Import pindesc.pin;
# pindesc.pin declares names:
#        VDD, inpins, outpins
# resource.rsc declares names:
# VSRange, VForce, Relay, VIH, etc.
Levels Lvl1
{

VDD
{
    VSRange = 7V;
    VForce = vforce;
    DpsRelay = CLOSE;
    PowerSequence = ON;
}
Delay 3mS;

inpins
{
    VIH = vih;
    VIL = vil;
    PinOutRelay = CLOSE;
    PowerSequence = ON;
}

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

January 18, 2018 VLSI D&T Seminar - Nelson
Test pattern timing – for each test vector

May define different timing patterns for different pins and/or test steps.
(4 force edges + 2 compare edges per pin)
Timing file (.tim)

Define timing of input transitions and sample times

PeriodTable
{
  
  #Cycle time “rate0” for test freq = 5MHz
  Period rate0 { 200nS; }  

}

#Force times for device inputs
Pin INPCONTROL_PINS
{

  WaveformTable inpctrl
  {
    { 1 { U@0nS; } }
  
    { 0 { D@0nS; } }
  }

}

Test pattern Symbols
Up/Down Transition Time

#Sample times for device outputs
Pin OUTPINS
{

  WaveformTable out
  {
    { H { H@85nS,E5; } }
    { L { L@85nS,E6; } }
    { X { Z@0nS; } }
  }

}

Test pattern Symbols
Sample High/Low Sample Time
Timing file example
This test engineer wanted to repeat tests for different periods to see when chips begin to fail

Version 1.0;
Import pindesc.pin;
# Perform the test with one set of parameters
Timing Tim_300_to_290
{
    CommonSection
    {
        Domain default
        {
            PeriodTable
            {
                Period per0 { 300nS; }
                Period per1 { 297.5nS; }
                Period per2 { 295nS; }
                Period per3 { 292.5nS; }
            }
        }
        Pin inpins
        {
            WaveformTable seq1
            {
                { 1 { U@0nS,E1; } }
                { 0 { D@0nS,E1; } }
            }
        }
    }
    Pin outpins
    {
        WaveformTable seq1
        {
            { H { H@299.5nS,E5; } }
            { L { L@299.5nS,E6; } }
        }
        WaveformTable seq2
        {
            { H { H@297nS,E5; } }
            { L { L@297nS,E6; } }
        }
        WaveformTable seq3
        {
            { H { H@294.5nS,E5; } }
            { L { L@294.5nS,E6; } }
        }
        WaveformTable seq4
        {
            { H { H@292nS,E5; } }
            { L { L@292nS,E6; } }
        }
    }
}

Define 4 periods
Apply all inputs at start of period

4 different output sample times - one for each period
Timing map file (.tmap)
Combine individual pin & rate timings into DUT “waveform sets”

Version 1.0;
Import pindesc.pin;
TimingMap TMap1
{
  Domain default
  {
    WaveformMap
    {
      PinFormat { inpins, outpins }
      wfs1, per0, { seq1, seq1 }
      wfs2, per1, { seq1, seq2 }
      wfs3, per2, { seq1, seq3 }
      wfs4, per3, { seq1, seq4 }
    }
  }
  Waveform Set
  {
    Period value
    Input waveform table
    Output waveform table
  }
  From .pin description file
  Test to be performed 4 times, using different timing, i.e.
  4 different “waveform sets”
  From .tim file
Test condition group file (.tcg)
TCG associates: spec, level, timing & timing map files

Version 1.0;
Import timing.tim;
Import timingmap.tmap;
Import level.lvl;
Import DiagPBSpec.spec;

# A Levels-Only Test Condition Group.
TestConditionGroup DiagPBTCG_300_to_290
{
    SpecificationSet DiagPBSpec;       #from .spec file
    Levels Lvl1;                                    #from .lvl file
    Calibration CalBlock1;                   #from .tim file
    Timings
    {
        Timing = Tim_300_to_290;      #from .tim file
        TimingMap = TMap1;               #from .tmap file
    }
}
Pattern (vector) files
Pin order is defined in .pin file

Sequencing instruction

Vector  Apply  Sample

NOP  { V { inpins=0111; outpins=LLLLLLLLL; } W {allpins=wfs1;}}
NOP  { V { inpins=0100; outpins=LLHLHHLLL; } }  
NOP  { V { inpins=0110; outpins=LLHLHLLL; } }  
NOP  { V { inpins=0110; outpins=HHLLLLLH; } }  
....
NOP  { V { inpins=0111; outpins=LLLLLLLLL; } W {allpins=wfs2;}}
NOP  { V { inpins=0100; outpins=LLLLLLLL; } }  
NOP  { V { inpins=0110; outpins=LLLLLLLL; } }  
NOP  { V { inpins=0110; outpins=LLLLLLLL; } }  
NOP  { V { inpins=0111; outpins=LLLLLLLL; } }  
....
NOP  { V { inpins=0111; outpins=LLLLLLLLL; } W {allpins=wfs3;}}
NOP  { V { inpins=0100; outpins=LLLLLLLLL; } }  
NOP  { V { inpins=0110; outpins=LLLLLLLLL; } }  
....

Waveform set
for timing

Use last waveform set
until a new one is listed
Functional test vectors may be created from simulation results

<table>
<thead>
<tr>
<th>Time (ns)</th>
<th>^A</th>
<th>^B</th>
<th>^Function</th>
<th>^F</th>
<th>^Fb</th>
<th>^Other2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>^S</td>
<td>^Clk1</td>
<td>^Other1</td>
<td>^Clk2</td>
</tr>
<tr>
<td>0.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>18.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>B</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>19.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>21.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>B</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>23.00</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>24.00</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>25.00</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>41.00</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>42.00</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>46.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>47.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>48.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>64.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>65.00</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>69.00</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>70.00</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>71.00</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>87.00</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>88.00</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>92.00</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>93.00</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>94.00</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>110.00</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

One “test cycle”:
- Inputs applied at 23
- Clk1 applied from 24-25
- Clk2 applied from 41-42
- Outputs stable after 42

Partition into “test periods”.
Within each period define:
- Times inputs applied
- Times of clock edges
- Times outputs sampled
Vectors extracted from functional simulation  
(to be translated to T2000 pattern format)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Fct</th>
<th>S</th>
<th>Ck1</th>
<th>Ck2</th>
<th>F</th>
<th>Fb</th>
<th>O1</th>
<th>O2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>4</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>5</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>6</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>8</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>9</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>A</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>B</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>C</td>
<td>2</td>
<td>B</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

For each vector:

Inputs (A,B,Fct) to be applied at start of cycle

Clocks (Ck1,Ck2) to be pulsed during cycle

Outputs (S,F,Fb) to be sampled at end of cycle

Vector 1

Vector 2

....
Tessent *Fastscan* ATPG tool - ASCII test file

(must convert to T2000 test patterns)

```plaintext
SETUP =
   TEST_CYCLE_WIDTH = 1;

DECLARE INPUT BUS "ibus" = "/A0", "/A1", "/A2", "/A3",
            "/B0", "/B1", "/B2", "/B3",
            "/M", "/S3", "/S2", "/S1",
            "/S0", "/C'n";

DECLARE OUTPUT BUS "obus_1" = "/A=E", "/C'n+4", "/F0", "/F1",
            "/F2", "/F3", "/X", "/Y";

END;

CYCLE_TEST =

PATTERN = 0;
   CYCLE = 0;
   FORCE "ibus" "10001100001101" 0;
   MEASURE "obus_1" "01101100" 1;

PATTERN = 1;
   CYCLE = 0;
   FORCE "ibus" "00011000011010" 0;
   MEASURE "obus_1" "00010011" 1;

PATTERN = 2;
   CYCLE = 0;
   FORCE "ibus" "00110000110100" 0;
   MEASURE "obus_1" "00000001" 1;
```

Test pattern inputs/outputs ("primary" inputs/outputs)
Clock specified elsewhere.

January 18, 2018

VLSI D&T Seminar - Nelson
Ex: FPGA boundary scan register test patterns

(Kunpeng Zeng)

NOP{V{TCK=1;TMS=0;TDI=1;TDO=X;}}  #first bit shifted in at TDI
NOP{V{TCK=1;TMS=0;TDI=0;TDO=X;}}
NOP{V{TCK=1;TMS=0;TDI=1;TDO=X;}}
NOP{V{TCK=1;TMS=0;TDI=0;TDO=X;}}
NOP{V{TCK=1;TMS=0;TDI=1;TDO=X;}}
NOP{V{TCK=1;TMS=0;TDI=0;TDO=X;}}
NOP{V{TCK=1;TMS=0;TDI=1;TDO=X;}}
NOP{V{TCK=1;TMS=0;TDI=0;TDO=X;}}

IDXI 363 {V{TCK=1;TMS=0;TDI=X;TDO=X;}}

Shift pattern 10101100 into BSR via the TDI pin

Repeat 363 times to shift pattern through all cells of the FPGA’s BSR

NOP{V{TCK=1;TMS=0;TDI=X;TDO=H;}}  #first bit shifted out at TDO
NOP{V{TCK=1;TMS=0;TDI=X;TDO=L;}}
NOP{V{TCK=1;TMS=0;TDI=X;TDO=H;}}
NOP{V{TCK=1;TMS=0;TDI=X;TDO=L;}}
NOP{V{TCK=1;TMS=0;TDI=X;TDO=H;}}
NOP{V{TCK=1;TMS=0;TDI=X;TDO=L;}}
NOP{V{TCK=1;TMS=0;TDI=X;TDO=H;}}
NOP{V{TCK=1;TMS=0;TDI=X;TDO=L;}}

Verify pattern 10101100 shifted out of BSR via the TDO pin
Test plan (.tpl)

Specify test conditions and test flow

(Standard “header section” of the test plan file omitted)

# Declarations of TestConditions TC1Min, TC1Typ, TC1Max, TC2Min, TC2Typ, etc
TestCondition TC_300_to_290
{
    TestConditionGroup = DiagPBTCG_300_to_290;
    Selector = typ;
}

# Other TestConditions

# Declare a "FunctionalTest", which refers to a C++ test class that runs the test
# and returns a 0, 1 or 2 as a result.
Test FunctionalTest DiagPBFunctionalTest_300_to_290
{
    PListParam = DiagPBPat;
    TestConditionParam = TC_300_to_290;
}

# Other functional tests

continued
# FlowMain is the main flow.

DUTFlow FlowMain

{ # First flow to be executed:

  DUTFlowItem DatalogSetupFlow DatalogSetup
  {
    Result 0 { Property PassFail = "Pass";
               GoTo FlowMain_300_to_290;
    }
  }

  DUTFlowItem FlowMain_300_to_290 DiagPBFunctionalTest_300_to_290
  {
    Result 0 { Property PassFail = "Pass";
               IncrementCounters PassCount;
               GoTo FlowMain_290_to_280;
    }
    Result 1 { Property PassFail = "Fail";
               IncrementCounters FailCount;
               SetBin SoftBins.FailCache3GHz;
               Return 1;
    }
  }

  Continue with this flow item if test passes

  Quit if test fails

  Result 0 = “Pass”
  Result 1 = “Fail”

continued
Test plan example – FPGA

(1) power up, (2) configure FPGA, (3) test the circuit

Test FunctionalTest Functional_power_typ
{
  ## Test Description = "Functional Test for typ values"
  PListParam = powerup;
  TestConditionParam = TC_typpower;
  DebugMode = 0;
}

Test FunctionalTest Functional_dpins_typ
{
  ## Test Description = "Functional Test for DPINS typ for FPGA configuration"
  PListParam = fpgaconfigpat;
  TestConditionParam = TC_typdpins;
  DebugMode = 0;
}

Test FunctionalTest Funct_test
{
  ## Test Description = "Functional Test post configuration"
  PListParam = testpat;
  TestConditionParam = TC_typtest;
  DebugMode = 0;
}
FPGA Test Plan (continued)

DUTFlowItem **FlowMain_Func_power_typ** Functional_power_typ
{
    Result 0 {
        Property PassFail = "Pass";
        GoTo FlowMain_Func_dpins_typ;
    }
    Result 1 {
        Return 1;
    }
}

DUTFlowItem **FlowMain_Func_dpins_typ** Functional_dpins_typ
{
    Result 0 {
        Property PassFail = "Pass";
        GoTo Flowmain_functional_test;
    }
    Result 1 {
        Return 1;
    }
}

DUTFlowItem **Flowmain_functional_test** Funct_test
{
    Result 0 {
        Return 0;
    }
    Result 1 {
        Return 1;
    }
}

---

Power up the FPGA

Download bit file to the FPGA

Test the configured circuit in the FPGA
Running T2000 ATE

- Open **DOS** command prompt
- Go to your **working directory**
- Go to **Pattern directory**
- Type **t2kctrl start**
Loading test plan

From Test Control Panel, select **File->Load TestPlan**

In the “Plan file” load the `.tpl` file from the OTPLSrc folder
Select “User TPL Env” and choose your `.env` file from your work directory
Flow Editor
Control and/or edit the main test flow

From Test Control Panel, select Tools -> flow editor

Control the test:
Start, Stop, Suspend, Reset, Continue
Other test options

• DC Parametric Tests
  – Per-pin parametric measurement unit
  – IDD tests, etc.
• Pattern editor – modify the test patterns
• Oscilloscope tool – study signal waveforms
• SHMOO plots
  – Modify variables over a range and plot #pass/fail vectors
• Test of scan-based designs
• Complex timing (ex. double data rate)
• Binning (hard and soft)
  – Control handler to move failed parts to bins
Pattern Editor

- Pattern editor is opened in test control panel
  
  File -> pattern editor

- Red highlighted patterns are failing

- Waveforms can be displayed by selecting pattern -> right click -> waveform oscilloscope/logic analyzer
Oscilloscope Tool
Verification of JTAG BYPASS register