# AUTOMATED REGRESSION ENVIRONMENT FOR IMPROVING DESIGN VERIFICATION PRODUCTIVITY

By
Keshav Rath
(Roll No. 2013104)

## SUPERVISOR(S):

**EXTERNAL:**
Mrs. RAMA KOWSALYA NAMBURI
Senior Project Manager
Canon ISDC, Bangalore

**INTERNAL:**
Prof. P.N.KONDEKAR
Professor
PDPM IIIT DM Jabalpur

**Electronics and Communication Engineering (B.Tech 2013)**

**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, DESIGN AND MANUFACTURING, JABALPUR**

Report

(16th May 2016 – 15th June 2016)

# 1. INTRODUCTION

During this period, I have been told to revise some old concepts and at the same time told to learn new topics so as to get familiar with the given codes (task) and requirements. The topics that I have told to be familiar are:

- PERL scripting language
- SystemVerilog
- Linux Commands and Shell Scripting

Some description of the topics I have studied are stated in this report.

# 2. PERL

PERL stands for Practical Extraction and Report Language. It is a scripting language that has been developed by Larry Wall in 1987. This language was initially developed as a general purpose Unix Scripting Language but later it became a different independent Language.

In this Scripting Language the concepts studied are:

- The requirement for setting up an environment in UNIX
- The first hello world program so to check the environment, i.e.

  print “Hello World\n”;
- The semantic rules of the language. (CASE SENSITIVE)
- Identifiers (variables, keywords, function, class, module) and escape characters/sequences.
- The three basic data types:
  - Scalars – Simple variables used to store numbers, strings or reference (address of a variable). Preceded by dollar sign ($).
  - Arrays – Ordered list of scalars that can be accessed by numeric index starting with zero. Preceded by at sign (@)
  - Hashes – Unordered sets of values that can be accessed by using ‘keys’. Preceded by percent sign (%)

- Scalars in detail - Numeric scalars, string scalars, scalar operations (addition, subtraction, multiplication, division, concatenation), v strings (v.1.20.300.400), special literals (_FILE_, _LINE_, _PACKAGE_).
- Arrays in details - Creation, accessing elements, sequential number array, array size, adding and removing elements, slicing, replacing, transforming into strings and vice-versa, merging arrays, selecting elements from lists
- Hashes in detail – Creation, accessing elements using key/value, extracting values, extracting keys, size of hash, checking for existence, adding and removing elements
- Conditional statements – if, else, elsif, unless, switch (case), ? : operator
- Loops – loop types (for, while, do…while, for each, unless) and control statements (next, last, continue, redo, goto) and the infinite loop
- Operators – Arithmetic, Equality. Logical, Assignment, Bitwise, Logical, Quote-Like, Miscellaneous
- Date and Time – Current Time, GMT, Formatting, Epoch time (the number of seconds that have lapsed from a given date),
- Subroutines (Function in other programming languages is called subroutines in PERL.) – Defining and calling
- References – Scalar data type that holds the location of another value which could be a scalar, array or hashes.
    - Creating references
    - Dereferencing
    - Circular References
    - References to functions
- Format – Defining a format, Using a format, Defining a report header and footer, defining a pagination
- File handling – Opening and closing files, Reading and writing files, renaming a file, deleting an existing file, positioning inside a file, file information
- Directories – Displaying all files and folders, creating new directories, removing existing directories, changing a directory

- Error handling – ternary function, warn and die function, carp and cluck function, crock, confess
- Special variables- Variables that have a special meaning in PERL. They can be categorized into:
    - Global Scalar Special Variables.
    - Global Array Special Variables.
    - Global Hash Special Variables.
    - Global Special Filehandles.
    - Global Special Constants.
    - Regular Expression Special Variables.
    - Filehandles Special Variables.
- Coding Standards – Some standards are maintained by programmers so as to make the code easier to read, maintain and understand.
- Regular Expression – String of characters that define a pattern or patters you are viewing. Eg: sed, grep, awk. There are three regular expression operator:
    - Matching regular expression
    - Substitute regular expression
    - Transliterate regular expression
- Sending emails: sending plain messages, sending html messages, sending attachments, using SMTP servers.
- OOPs
    - Objects basics – Any OOP can be understood by these basic terms
        - Object – reference to a data type that knows what class it belongs to.
        - Class – A package that contains corresponding methods required to create and manipulate objects
        - Method – Subroutine, defined within the package,
    - Defining a class
    - Defining methods
    - Inheritance
    - Method Overriding
    - Default Autoloading

    - Destructors and Garbage Collections
- Excel sheets – Write scripts so as to format any excel sheet according to users' desire. Reading and writing into a file. Formatting the sheet, creating new excel sheets and workbook. For using this, modules can be downloaded from the Comprehensive Perl Archive Network (CPAN).

## 3. SystemVerilog

This is an extension of Verilog. It is a combination of hardware description language and hardware verification language. The feature-set of SystemVerilog can be divided into two distinct roles:

- SystemVerilog for RTL design is an extension of Verilog-2005; all features of that language are available in SystemVerilog.
- SystemVerilog for verification uses extensive object-oriented programming techniques and is more closely related to Java than Verilog.

The concepts studied under System Verilog are:

- A brief introduction of Verilog
    - Introduction to the various steps in Verilog
        - Specification
        - High Level Design
        - Low Level Design
        - RTL (Register-Transfer Level) coding
        - Verification
        - Synthesis
    - Definition of modules, ports(input, output, bi-directional), drivers, data types, operators
    - Control statements (if else, case, while, for loop, repeat)
    - Variable assignment for combinational elements (assign, always) sequential elements (always statement) and testbenches (initial statement)
    - Definition of always and assign statements

    - Tasks and functions with their minor differences
        - Tasks can have delays, but functions cannot
        - Functions can return a value, but tasks cannot
    - Testbenches
- Literals – Integer and Logic, Real, Time, String, Arrays, Structures
- Data Types –Integer data types (2 state and 4 state), null, strings, event data type, user defined data type, enumerated data types, type-casting (static and dynamic)
- Arrays – packed and unpacked arrays, multi-dimensional arrays, dynamic arrays and methods, associative arrays and methods, array methods , queues and methods,
- Structures and Unions – packed, unpacked, tagged
- Operators – assignment operators, logical and bit-wise operators, wild equality and inequality, operators' precedence and associativity, concatenation, streaming operators and set membership operators.
- Procedural statement and control flow – selection statements, loop statements, jump statements, final blocks, named blocks, disable block, event control, level-sensitive sequence control
- Processes – always assignments (always_comb, always_latch, always_ff), continuous assignments, fork join (join all, join any, join none), fork control (wait fork and disable fork)
- Subroutines – Tasks and Functions, discarding function-return values, Argument Passing (pass by value, pass by reference, pass by name, default argument values, optional argument lists)
- OOPs (Object Oriented Programming) –
    - Introduction,
    - Objects (member and methods),
    - Constructors(for initialization),
    - Classes (Introduction)
    - Assignment of classes,
    - Inheritance
        - Subclasses and Superclasses
        - Overriding (Members)

    - Data hiding and encapsulation with access specifiers (private, protected and public)
    - Polymorphism
    - Virtual classes (abstract classes)
    - Parameterized classes (type, value, generic, extending)
    - Nested Classes
    - Constants ( global constant, constant class, instance constant)
    - Statics - static classes, static methods, static lifetime method
    - Casting - as tasks as well as functions
    - Copy – shallow copy, deep copy, cloning
    - Scope resolution operator, null, external declarations
    - Differences between classes and structures
    - Typedef classes – forward reference, and circular dependency
    - Multiple inheritance and method overloading
- Randomization
    - Random Variables (declaration, rand and randc modifiers)
    - Randomization methods (pre built-in methods, pre randomize, post randomize,
    - Constraint blocks
    - Inline constraints - adding constraints to already existing constraints
    - Global constraints – adding constraints between variables of different classes
    - Static constraints
    - Constraint modes
    - Randomization Controllability – additional controllability by giving maximum and minimum values
    - Randomizing objects –
    - CRV (Constrained Random Verification)
    - Verilog CRV
    - System Verilog CRV
- Coverage - It is defined as the extent or degree to which something is observed, analyzed, and reported. In typical methodologies you would
        - Write the test plan.

- Write the tests
- Perform functional coverage to cover everything specified in the test plan.
- Perform code coverage to identify any redundant or overlooked items.
- Update the test plan to include any new/untouched scenarios or remove any redundancy.
- Iterate back through the functional coverage.

There are two types of coverage: functional coverage and code coverage. Each type can be broken down into smaller categories, each with its own interpretation as to how well the design has been exercised.

- Functional Coverage - Functional coverage brings the specification, the test plan, and the actual tests together by providing information about which functions have or have not been exercised in the design. This can be further categorized into:
  - Application Coverage -
  - Interface Coverage
  - Structural Coverage
- Code Coverage – This coverage tells us about the degree to which the code that has been exercised. This information is very valuable but should be appropriately utilized. This can be further categorized into:
  - Statement/Line Coverage –This coverage tells us whether or not a code has been exercised or not.
  - Expression Coverage – This generally deals with codes involving expression, especially Boolean expression. This coverage tells whether the expression has been exercised in every way possible.
  - State Machine Coverage -This coverage provides information on the visited transitions, arcs, or states in a finite state machine. It can also provide the actual path taken through the state machine.

# 4. LINUX COMMANDS

Linux is a Unix-like and mostly POSIX (Portable Operating System Interface) -compliant computer operating system (OS) assembled under the model of free and open-source

software development and distribution. Linux was originally developed as a free operating system for personal computers based on the Intel x86 architecture, but has since been ported to more computer hardware platforms than any other operating system. Some of the most popular mainstream Linux distributions are Arch Linux, CentOS, Debian, Fedora, Gentoo Linux, Linux Mint, Mageia, openSUSE and Ubuntu, together with commercial distributions such as Red Hat Enterprise Linux and SUSE Linux Enterprise Server.

In this topic, the concepts studied are:

- The architecture:
    - Kernel
    - Shell
    - Commands and Utilities
    - Files and Directories
- System Bootup – Logging in, changing passwords, Finding out yourself, Logging Out, System Shutdown.
- File Management – Knowing the three types of files
    - Ordinary files – data, text, program instructions
    - Directories – Equivalent of Folders in Windows
    - Special Files – Files that provide access to hardware such as hard drive, modems and Ethernet adapters

  Listing (including hidden), creating, editing, copying, renaming, deleting and miscellaneous such as counting words in a file
- Directories – Knowing about home directories, listing, creating, changing, renaming, deleting
- File Permission – Permission indicators, file access modes, directory access modes, changing permission, changing owner(the person who has the file) and group(the set of people who can access the file)
- Environment – Setting the path, the '.profile' file (administrator), environment variables
- Basic utilities – Printing files, Sending emails
- Pipes and Filters – grep command ( this command globally searches for a regular expression and print all lines containing it), sort command

- Processes – Starting a process, Foreground processes, Background processes, listing running processes, stopping processes, parent and child processes, zombie and orphan processes, daemon processes
- Network Communication Utility – Ping Utility, ftp utility, telnet utility, finger utility
- Vi- editor – starting the vi editor, operation modes (command mode, insert mode), getting out of vi editor, moving within a file, control commands, deleting characters, copy and paste commands, advanced commands,

## 5. SHELL SCRIPTING

A shell script is a computer program designed to be run by the UNIX shell, a command-line interpreter. The various dialects of shell scripts are considered to be scripting languages.

The concepts studied are:

- Variables – Defining, accessing, read-only variables, unsetting (deleting), Special Variables
- Arrays – Creating and accessing
- Basic operators – Arithmetic, Relational, Boolean, String, File Test
- Decision Making – if….else, case….esac
- Loops – While, For, until, select
- Loop Control – break and continue
- Quoting Mechanism – Single quotes, double quotes, backslash

## 6. CONCLUSION

The above topics are studied in order to get a good idea of the work to be done ahead.

# REFERENCES


- www.tutorialspoint.com
- www.wikipedia.org
- www.testbench.in
- www.referencedesigner.com/tutorials/verilog
- www.asic-world.com/verilog/veritut.html
- www.asic-world.com/systemverilog/tutorial.html
- www.youtube.com