**VVDN Training Syllabus**

**Rev: A0-08**

**25thMay, 2018**

***VVDN Training Syllabus***

***Phase-I***

***For Freshers***

**VVDN Contact:**

[mentors\_chn@vvdntech.com](mailto:mentors_chn@vvdntech.com)

**REVISION HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **DATE** | **REVISION** | **DESCRIPTION** | **AUTHOR** |
| 02/06/2014 | A0-01 | Initial Version | Mentors Team |
| 22/07/2014 | A0-02 | Updated C programming section  Replaced reference web links with document | Mentors Team |
| 17/11/2014 | A0-03 | Updated C programming section and included Instruction to new joiners & target days for individual C programming topics | Mentors Team |
| 02/06/2015 | A0-04 | Updated Status mail format &Removed Exercises in C programming section | Mentors Team |
| 18/09/2015 | A0-05 | Adding minicom, wildcards/expressions in Linux commands.  Adding getopt and getoptlong in C programming.  Addition to points under Coding style and HW concepts | Mentors Team |
| 18/02/2016 | A0-06 | Updated linux commands section Elaborated syllabus Replaced broken mediawiki links with local SVN path Added few reference links | Mentors Team |
| 01/06/2017 | A0-07 | Assessment Details added.  Added reference links  Module owner and target date updated  Syllabus split into two phase training | Mentors Team |
| 25/05/2018 | A0-08 | Linux Commands Updated  Segmented Network Topics wrt Layers  Added Network Concepts & Commands | Mentors Team |

Table of Contents

[Table of Contents 3](#_Toc484417015)

[1. Introduction 6](#_Toc484417016)

[1.1. VVDN Process 6](#_Toc484417017)

[1.2. Mailing Ethics 6](#_Toc484417018)

[1.3. Art of Communication 6](#_Toc484417019)

[1.4. Instruction to new joiners 6](#_Toc484417020)

[1.4.1. VVDN etiquette 6](#_Toc484417021)

[1.4.2. Do's / Don'ts in VVDN 6](#_Toc484417022)

[1.4.3. Status mail format 7](#_Toc484417023)

[1.4.4. Training Assessment 7](#_Toc484417024)

[2. Linux Commands 8](#_Toc484417025)

[2.1. Super user 8](#_Toc484417026)

[2.2. Package installation 8](#_Toc484417027)

[2.3. Files/Directory Basics 8](#_Toc484417028)

[2.4. File viewing 8](#_Toc484417029)

[2.5. File properties 9](#_Toc484417030)

[2.6. File location 9](#_Toc484417031)

[2.7. File text identification 9](#_Toc484417032)

[2.8. File comparison 9](#_Toc484417033)

[2.9. Disk and file-system 9](#_Toc484417034)

[2.10. Runtime level management 9](#_Toc484417035)

[2.11. System monitoring 10](#_Toc484417036)

[2.12. Compressing/Uncompressing 10](#_Toc484417037)

[2.13. Job scheduling 10](#_Toc484417038)

[2.14. Miscellaneous 10](#_Toc484417039)

[2.15. Shell variables 11](#_Toc484417042)

[2.16. Shell configuration file 11](#_Toc484417043)

[2.17. Wildcards 11](#_Toc484417043)

[3. Networking 1](#_Toc484417048)1

[3.1. Physical Layer 1](#_Toc484417049)1

[3.2. MAC Layer 1](#_Toc484417049)1

[3.3. IP Layer 1](#_Toc484417050)2

[3.4. Layer 4 1](#_Toc484417051)3

[3.5. Commands and Protocols 1](#_Toc484417052)3

[4. Networking Tools 1](#_Toc484417057)4

[4.1. Networking tools 1](#_Toc484417059)4

[4.2. Packet capture Tools 1](#_Toc484417061)4

[4.3. Packet Generators 1](#_Toc484417062)4

[5. Editors / Code Navigation 1](#_Toc484417044)1

[5.1. Editor features (Vim or Emacs editor) 1](#_Toc484417045)5

[5.2. Ctags/Etags 1](#_Toc484417046)5

[5.3. Cscope 1](#_Toc484417047)5

[5.4. Taglist 1](#_Toc484417043)5

[6. Software Version Control tools 1](#_Toc484417063)5

[6.1. SVN 1](#_Toc484417064)5

[6.2. GIT 1](#_Toc484417065)6

[7. Coding Style 1](#_Toc484417066)6

[8. Basic C 1](#_Toc484417067)7

[8.1. Introduction 1](#_Toc484417068)7

[8.2. Fundamentals of C 1](#_Toc484417069)7

[8.3. Storage Class 1](#_Toc484417070)7

[8.4. Control Statements and Loops 1](#_Toc484417071)7

[8.5. Pointer basics + Arrays 1](#_Toc484417072)7

[8.5.1. Pointer 1](#_Toc484417073)7

[8.5.2. Arrays 1](#_Toc484417074)8

[8.6. Functions 1](#_Toc484417075)8

[8.7. Structures &Unions 1](#_Toc484417076)8

[9. Scripting 1](#_Toc484417077)8

[9.1. Linux Shell script 1](#_Toc484417078)8

[10. H/W Concepts 1](#_Toc484417079)9

[11. Module and Targets 1](#_Toc484417080)9

19.1 C Programming module target days ………………………………………………………………………………………… 20

# Introduction

## VVDN Process

* <https://sites.google.com/a/vvdntech.com/docshare>

## Mailing Ethics

* <http://192.168.101.10/svn/trainee/documents/trng_ref_docs/VVDN_INDx_TRNG_Mailing_Ehics_A0-01.pdf>

## Art of Communication

* <http://192.168.101.10/svn/trainee/documents/trng_ref_docs/VVDN_INDx_TRNG_Art_of_communication.ppt>

## Instruction to new joiners

### VVDN etiquette

* Don't address anyone by sir / madam through mail or in person. You call them by their names irrespective of their designation or age.
* Make sure you send a status mail at the end of every day.
* Don't forget to fill the time-sheet at the end of every day. (You can find Timesheet in VVDN MediaWiki)
* Leave
  + If you need to take a planned leave, you should apply for it through Orange HRM portal (in VVDN MediaWiki) and if leave approved, you should intimate admin (admin\_chennai@vvdntech.in).
  + For sick/unplanned leave, you should intimate your RM by phone/email and update HRM on return to duty.
* Please keep your reporting manager updated regarding the status of your training as well as in project.
* Use following generic signature in all your mails.  
  With Warm Regards,  
  XxxxxXxxxx  
  VVDN Technologies Pvt. Ltd.  
  Cell: +91 xxxxxxxxxx | Skype: xxxxx

### Do's / Don'ts in VVDN

* **LAB Maintenance**  
  Don't bring water / snacks inside the lab.

Avoid keeping water bottle near boards.

* **Office Timing**  
  Working hours: 9AM to 6.30PM  
  Everyone must be in office by 9AM and can leave office by 6.30PM  
  If anyone wants to work after 6.30PM to complete tasks on same day, he can work up to 8.30PM without any intimation to admin  
  Office will be closed by 8.30PM by office boys and it is not encouraged to work thereafter at office, in general.  
  If anyone wants to work after 8.30PM, he has to get the permission from his RM and intimate Chennai Admin team.
* **AHU Room**  
  For your own safety, please do not enter in to AHU room for any reason.
* **Misc**  
  Please take care of all the resources that you are using. Power off all the machines / devices including PC, monitor, board, etc. when you are leaving / not using. 'Save Electricity

### Status mail format

TO: Respective RM

Subject should be in this format: *“***TRNG\_BATCH\_##: Daily Status DD-MM-YY***”*

Mail content:

Hi <Name of the person(s) mentioned in TO>

**Task:**

List of tasks assigned

**Status:**

Work done on particular day (Briefly explain on your own words)

**Pending:**

Remaining task and assignments

**With Warm Regards,**  
 <Your Full Name>  
 VVDN Technologies Pvt Ltd  
 Cell : | Skype :

### Training Assessment

During training period trainees will be assessed based on the assessment scores and various factors trainees will be split into team for different domain training.

Phase-I Assessment:

* Linux commands to Basic C (Sec 2 to Sec 8)

NOTE: During assessment internet will be blocked/monitored.

During assessment the review request format

> Summary Section:

The summary section should be as follows:

**“[Assessment] [Assessment\_No.Prog\_no] : <prog summary>”**

> People Section:

Add "mentors\_chn" ID in "People" section while sending review.

# Linux Commands

Note: The Linux commands need to be practiced along with wildcards wherever applicable. (<http://tldp.org/LDP/GNU-Linux-Tools-Summary/html/x11655.htm>).

Learn Linux directory structure (<https://en.wikipedia.org/wiki/Filesystem_Hierarchy_Standard>)

## Super user

* sudo
* su

## Package installation

* yum (for Fedora)
* apt-get (for Ubuntu)

## Files/Directory Basics

* cp [with options] <-l, -a, -r, -f, -i, -v, --help>
* mv [with options] <-l, -a, -f, -i, -v, --help>
* rm [with options] <-i, -r>
* ln [with option] <-s>
* cd [with parameters] < .., dir\_path, /, ~/ >
* pwd
* mkdir<-p>
* rmdir
* ls [with options] <-l, -h, -a, -R, --help>
* tree

## File viewing

* cat
* less
* head [with options] <-c, -n>
* tail [with options] <-c, -n, -f>
* more

## File properties

* stat
* wc [with options] <-c, -b, -l>
* du [with options] <-s, -h, -a>
* file
* touch [with options] <-c, -a, -m>
* chown
* chmod

## File location

* find [with options] <-P, -L, -name, -size, -perm, -user, -wholename.>
* locate
* which
* whereis

## File text identification

* grep [with options] <-n, -i, -c, -r>
  + regular expression (https://www.cs.tut.fi/~jkorpela/perl/regexp.html)

## File comparison

* diff [with options] <-u, -N, -R, -i>
* comm
* cmp
* vimdiff (For vim users)

## Disk and file-system

* df <-h>
* mount [with options] <-o, -t>
* umount
* fdisk
* mkfs.ext\*, mkfs.vfat, mkfs.ntfs

## Runtime level management

* exit
* reboot
* shutdown <-c, +n(where n is an integer)>

## System monitoring

* uptime
* w, who, whoami
* pstree <-p>
* top [with options] <-d, -i, -n. –p>
* ps [with options] <-a, -u, -x, -e, -L>
* uname [with options] <-a, -s, -n, -r, -v, -m, -p, o>
* lsmod
* lspci [-v]
* lshw
* free
* dmesg [with options] <-c, -C, -D>
* insmod
* Kill [-SIGTERM, - SIGABRT, -SIGSEGV, -SIGKILL, -SIGHUP]
* killall

## Compressing/Uncompressing

* tar [with options] <-c, -x, -v, -f, -z, -j>
* zip
* unzip
* rar
* gzip
* bzip2

## Job scheduling

* sleep
* watch <-n>
* renice

## Miscellaneous

* man
* info
* clear
* reset
* ar <-r, -s>
* env
* export
* echo
* passwd
* pipe (|)
* redirection (<, >)
* append (>>)
* xargs
* tee
* awk
* cut
* sed
* pgrep

## Shell variables

* PWD
* OLDPWD (used by cd -)
* PATH
* SHELL
* ? (Stores value of last executed command)
* EDITOR
* Create/Display/Clear Custom shell variables

## Shell configuration file

* .bashrc file(Alias)

## Wildcards

* \*, &, /, []

# Networking

## 3.1 Physical layer

* Types of cables (RJ45 & SFP)

## 3.2 MAC Layer

* Format of MAC
* Types of MAC
* Use of different MAC
* L2 devices [Switch and Access Point]
* Bridge
* Layer 2 Protocols
* PPP
* L2TP
* LLDP
* STP
* Difference between IEEE 802.11 & IEEE 802.3
* VLAN
* What is VLAN? Why and When we need?
* Tagging and Untagging
* Trunk port and access port
* Packet analysis
* Commands
* iwconfig
* ifconfig [change MAC address]
* brctl

## 3.3 IP layer

* Types of IP address.
* IPv4 / IPv6
* Public & Private IP
* Classes of IP address
* Subnetting & Supernetting.
* How to assign IP, Subnet, Gateway
* How DHCP taking IP
* How ARP working
* How Ping working
* L3 Devices - Routers
* Protocols
* ARP
* IGMP
* IPSec
* Routing & NATing
* Commands
* arp
* ifconfig
* route
* traceroute
* dig
* nslookup
* iptables, ebtables
* ip

## 3.4 Layer 4

* TCP
* UDP
* ICMP
* Commands
* netstat

## 3.5 Commands and Protocols

* Bootp
* DNS
* SSH
* FTP, SFTP, TFTP
* HTTP/HTTPS/XML
* Radius
* TLS
* NTP
* Curl
* Commands
* ssh
* scp
* curl
* Telnet
* iptraf

# Networking Tools

## Networking tools

* http://192.168.101.10/svn/trainee/documents/trng\_ref\_docs/Network%20Commands\_tools.pdf

## Packet capture Tools

* Wireshark
* tcpdump [with options] <-i, -s, -w>

## Packet Generators

* Iperf[with options] <-s, -c, -d, -w, -u, -b, -p>
* Ostinato
* Devices used for generating packets : IXIA, EXFO

# Editors / Code Navigation

## Editor features (Vim or Emacs editor)

* Choose any 1 editor
* Learn editor basics with vimtutor / emacs tutorial
* Search a string,
* Replace a string with other
* Copy a block of code and paste at different place
* Move a block of code to another place
* Find a variable / function name and use them (auto fill CTRL+P or CTRL+N)
* Go to starting of a function
* Go to end of a function
* Split the editor into multiple windows(vsplit / hsplit) and open other file
* Traverse between multiple windows
* Copy between multiple windows
* Jump to specific line
* Set paste
* Alignment
* Expand spaces
* Enable syntax
* Search special characters
* Recording
* Diff mode
* Ignore case
* Browse man page
* Execute commands (without leaving editor)
* Append output of external command
* Learn about vim plugins / emacs extensions
* Configuration file (.vimrc / .emacs)
* Exercise
  + Just practice the features
  + vimtutor

## Ctags/Etags

* Navigate to symbol definition and return back in editor, choose the tags file from editor, update the tags file from editor
* Practice ts, tp, tn ctags options on editor
* Exercise
  + Go through any project codebase

## Cscope

* Options: -b, -R, -q, -c, -d
* Find the C symbol in codebase, Find the global definition, Find functions called by this function, Find functions calling this function, Find this text string, Change this text string, Find this egrep pattern, Find this file, Find files #including this file. update cscope.out file from editor, using cs find options on editor, Integrated use of editor with cscope & shortcuts and plugins for editor (ctrl+\ for vim)
* Exercise
  + Go through any project codebase

## 5.4 Taglist

* Tlistopen

# Software Version Control tools

## SVN

* <http://192.168.101.10/svn/trainee/documents/trng_ref_docs/VVDN_INDx_TRNG_SVN_Essentials_A0-01.pdf>
* svn checkout, svn commit, svn info, svn status, svn merge(Try reverse merge also), svn add, svn copy, svn rm etc.
* svn diff and patch command
* Exercise
  + Create a repo add and updates files in the same, Create a project tree in repository (include branches, trunk, tags, etc.)
  + Add a dummy project to the repo, create a branch, update files in branch and merge it to the trunk, reflect the changes made in trunk in the branch, create a tag

## GIT

* git init, git add , git commit, git diff, git status, git show
* git branch, git checkout, git clone, git merge, git remote , git reset, git fetch, git pull, git review, git stash, git cherry-pick
* Refer man pages gittutorial and Practice using <https://try.github.io>
* Configuration file (.gitconfig, .gitignore, .gitreview)
* Exercise:
  + Same as SVN (try with git commands)

# Coding Style

Refer following documents in http://192.168.101.10/svn/trainee/documents/trng\_ref\_docs/Coding\_style/

* Coding style
* Commenting style
* Indentation style
* Variable nomenclature

Following points should be taken care before posting any code for review:

1. Compilation of the “src” files should be done using the ‘*–Wall*’ option in gcc and no warning/error should be left out.
2. All the patches should be scanned using these scripts for coding style. It will throw warning/error when it finds unexpected coding.

/usr/src/linux-headers-`uname –r`/scripts/checkpatch.pl <file\_name.patch>

1. Use this script to remove whitespaces

/usr/src/linux-headers-`uname –r`/scripts/cleanfile <src\_file>

OR

/usr/src/linux-headers-`uname –r`/scripts/cleanpatch <file\_name.patch>

OR

Use vim scripts to remove unwanted space in your program

<http://vim.wikia.com/wiki/Remove_unwanted_spaces>

1. You should use 4 spaces, not tabs, in all files, except makefiles and config files.

<http://vim.wikia.com/wiki/Converting_tabs_to_spaces>

# Basic C

Prerequisites:

* SVN – Section 4.1
* Learn Review board usage.

Add files to the review board, sending review request, updating review request with new diff file, deleting the review request. Reference: <https://www.reviewboard.org/docs/manual/2.0/>

## Introduction

* Why C in embedded systems
* Compilation stages
* GCC Compiler & its options (-E -s -C -g -o)
* Computer program memory

## Fundamentals of C

* Data types, Variables and Constants
* Data type Modifiers (size, sign, constant, volatile)
* User defined data types
* Typecasting
* Operators and Expression

**Note:**  For this section’s Exercise programs, don’t use Arrays/functions.

## Storage Class

* Automatic
* Static
* Extern
* Register

## Control Statements and Loops

* If statement, if else statement , else if ladder, switch statement
* for, while, do while

**Note:** For this section’s Exercise programs, implement using control statements only. Don’t use arrays/functions.

## Pointer basics + Arrays

### Pointer

* The purpose of pointers
* Defining and declaring pointers
* Pointer Assignment
* Generic and Null Pointer
* Pointer Arithmetic

### Arrays

* What is array and why is it necessary?
* Defining, initializing and using arrays
* Multi-dimensional arrays
* Macro definition, expansion and its usage in array declaration
* Character Array: Strings

## Functions

* Pass by value / reference
* Returning values from Function
* Recursive Functions
* Call Back Functions
* Implications on Stack
* Library v/s User defined function
* Passing variable no of arguments
* Passing an array as argument to function & Returning Array from function

**Note:** For this section’s Exercise programs, Try to make the code modular by using user-defined functions.

## Structures &Unions

* Structures
* Unions

# Scripting

## Linux Shell script

* Description and types of scripting languages
* Exercise
  + Write a script to filter and show list of files/directories which exceeds specific size limit. Path and size limit need to be get from user at runtime.
  + Implement hangman game using scripts

# H/W Concepts

* Handling PCB
* ESD
* Power Supply selection
* Using LAB Equipment
* Over view of components and modules
* Clock and Reset
* External and Internal Pull-up / Pull-down.
* Strapping pins
* Pin Multiplexing
* JTAG
* Schematics, Board file, Netname.
* Board bring-up sequence (h/w perspective).

# Module and Targets

|  |  |  |
| --- | --- | --- |
| S.No | Module | Target (Days) |
| 1 | Introduction | 1 |
| 2 | Linux Commands | 2 |
| 3 | Networking | 4 |
| 4 | Networking tools | 1 |
| 5 | Editors / Code Navigation | 3 |
| 7 | Software Version Control | 1 |
| 8 | Coding style | 1 |
| 9 | Basic C | 10 |
| 10 | Assessment | 1 |
| 11 | Hardware Concepts | 1 |
| 12 | Scripting | 2 |

# 19.1. C Programming module target days

|  |  |  |
| --- | --- | --- |
| **S.No** | **Topics** | **Module Target** |
| 1 | Introduction | 1 |
| 2 | Fundamentals of C | 2 |
| 3 | Control Statements and Loops | 1 |
| 4 | Pointer basics + Arrays | 2 |
| 5 | Functions | 2 |
| 6 | Structures & Unions | 2 |
|  | TOTAL TARGET DAYS | 10 |