./

Learning Report – LINUX OS AND PROGRAMMING

Course Code: <CODE>



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver.Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **Approved By** | **Remarks/Revision Details** |
|  | 02-03-2021 | HARSHITHA R |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Contents

￼

[Case study: Design & Link with Libraries 4](#_Toc65616340)

Ignore this page

# Contents:

Activity 1: Case study: Design and link with the libraries

Activity 2: OS programming about threads, Processes and syscalls

# 

# Activity 1

# Case study: Design & Link with Libraries

* Created source file for mystring, myutils, bit\_set\_reset.
* Created header files for mystring, myutils, bit\_set\_reset
* Created test.c which have functions of source files
* Generated the Makefile to compile all source files and link with test code
* Mystring file consists of following functions:
* Mystrlen-It will give the length of the string
* Mystrcpy-It will copy the string
* Mystrcat-It will add the two string
* Mystrcmp-It will compare the two strings
* Myutils file consists of following functions
* Factorial –It will perform factorial of a number
* isPrime – It will check if a number is prime or not
* isPalindrome –it will check the palindrome of a number
* vsum- It will give the sum of variable input
* Bit\_set\_reset consists of following functions:
* Set – Set the bit position
* Reset – Set all the bit position to zero
* Flip – It will flip the bit values
* Query

* Created object files for all the above-mentioned files
* Generated libmystr.a with string functions

libmyutils.a with utility functions

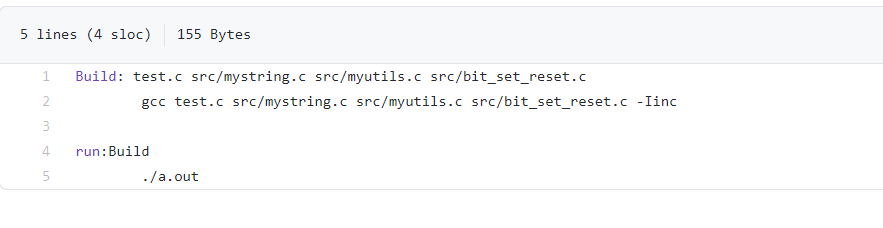
bit\_set\_reset.a with bit functions

* Above are the static libraries generated during the process
* Generated shared object files

\* libmystr.so with string functions

\* libmyutils.so with utility functions

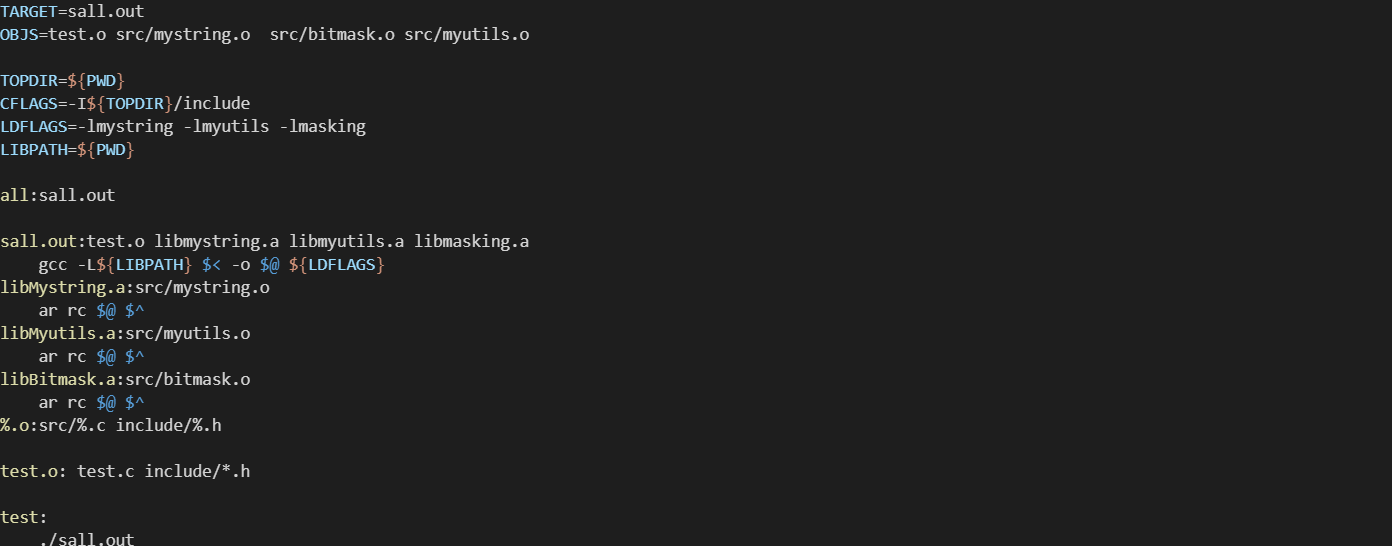
\* libbit\_set\_reset.so with bit introspection functions



This picture shows the Makefile for source files and header files

## C:\Users\user pc\Desktop\statics.png

This picture above is the code of Makefile for static library

The picture above is the Makefile for Dynamic linking library

# ACTIVITY 2:

## Description:

* A System call is a mechanism which provides the interface between a process and operating system. It offers the service of the operating system to the user programs via API.
* A process is the instance of a computer program that is being executed by one or many threads. Depending on the operating system (OS), a process may be made up of multiple threads of execution that execute instructions concurrently.
* The Shell is the layer of programming that understands and executes the commands a user enters. As the outer layer of an operating system, a shell can be contrasted with the kernel, the operating system’s inmost layer or core of services.
* fork() creates a new process by duplicating the process, the new process, referred to as child, is an exact duplicate of the calling process.
* Return value of fork(), the PID of the child process is returned in the parent and ) is returned in the child. On failure –1 is returned in the parent.
* The execl() functions takes the path of the executable binary files as the first and second argument. The execl() system function runs the command and prints the output.
* The thread is a basic unit of CPU utilization, consisting of a program counter, a stack, and set of registers and thread ID.
* Types of thread User level thread is a user managed thread and kernel level thread is a OS managed thread acting on kernel, an operating system core.
* Multiple thread within the same application can run in parallel on multiple processors and a blocking system call need not block the entire process.

## Learning Outcomes:

* We could be able to write the program to copy one file contents to other using open, read, write, close system calls.
* a program to send specific signal to a target process.
* To design the minishell and program to compile & link any program within child process by launching gcc using execl and program to build multifile program using fork & exec.
* We could be able to write a program to print current time periodically.

## Challenges faced

* Understanding the threads and multithreading was a challenge.
* Difference between PID and PPID.

Github link: <https://github.com/99003531/LINUX_01>