Lab #1: System Call Implementation

Group Members

1. Yash Aggarwal (yagga004@ucr.edu)
2. Nityash Gautam (ngaut006@ucr.edu)
3. Parth Bhatt (pbhat029@ucr.edu)

Link to Videos

1. Google Drive
    a. https://drive.google.com/file/d/1dsCuwNWqNFMh9WYTxAoC5fOZWobuIrI_/view?usp=share_link
2. Youtube
    a. https://youtu.be/IjfBkONmJco

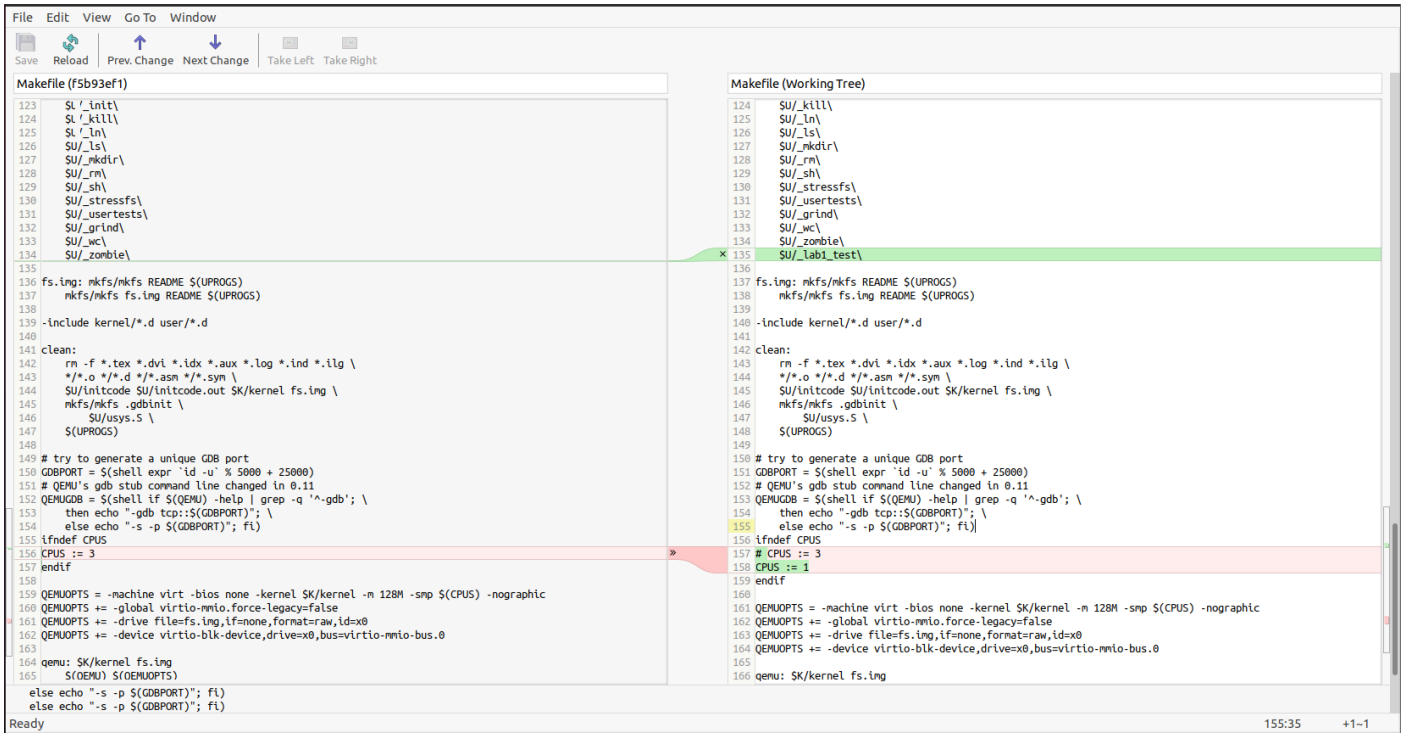
List of Files Modified

Makefile
|- kernel/
    | - defs.h
    | - kalloc.c
    | - proc.c
    | - proc.h
    | - syscall.c
    | - syscall.h
    | - sysproc.c
|- user/
    | - lab1_test.c
    | - user.h
    | - usys.pl

Screenshot of changes and Explanation

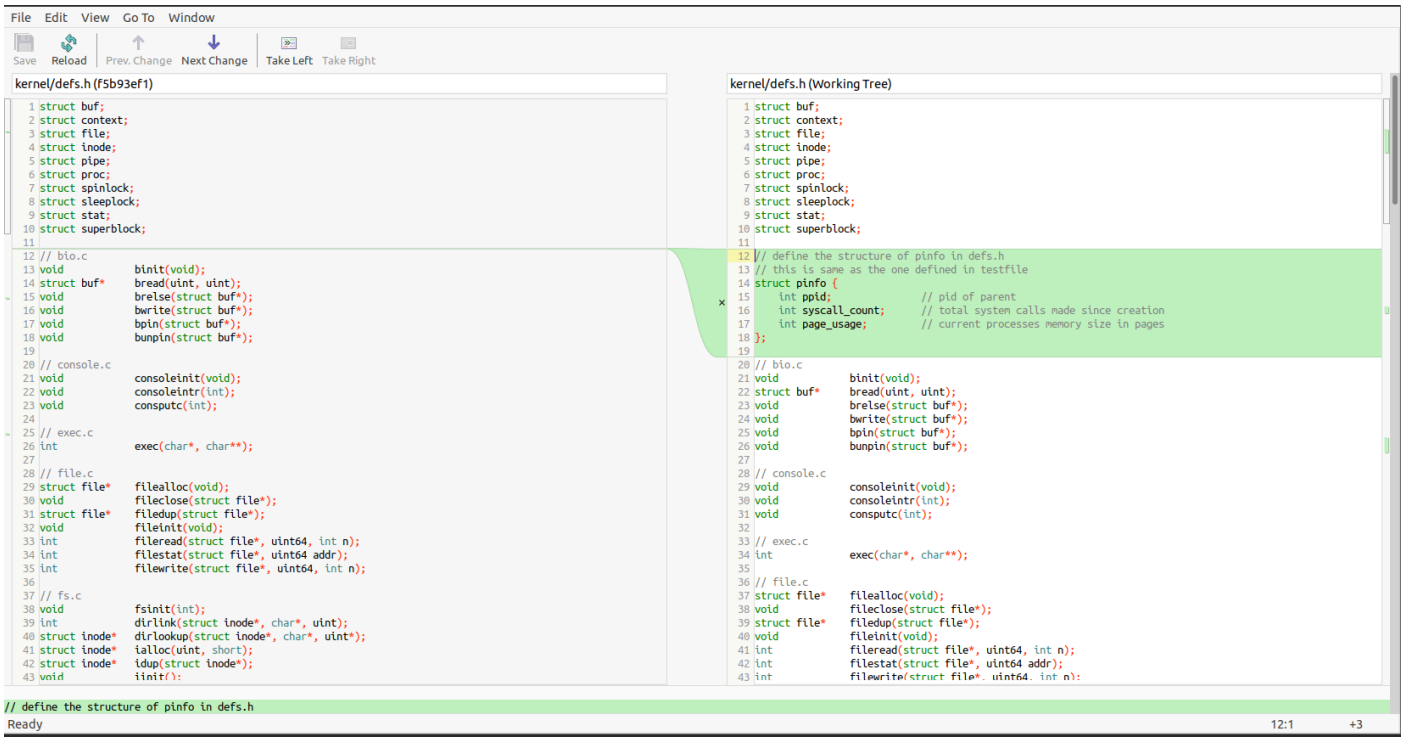
Makefile

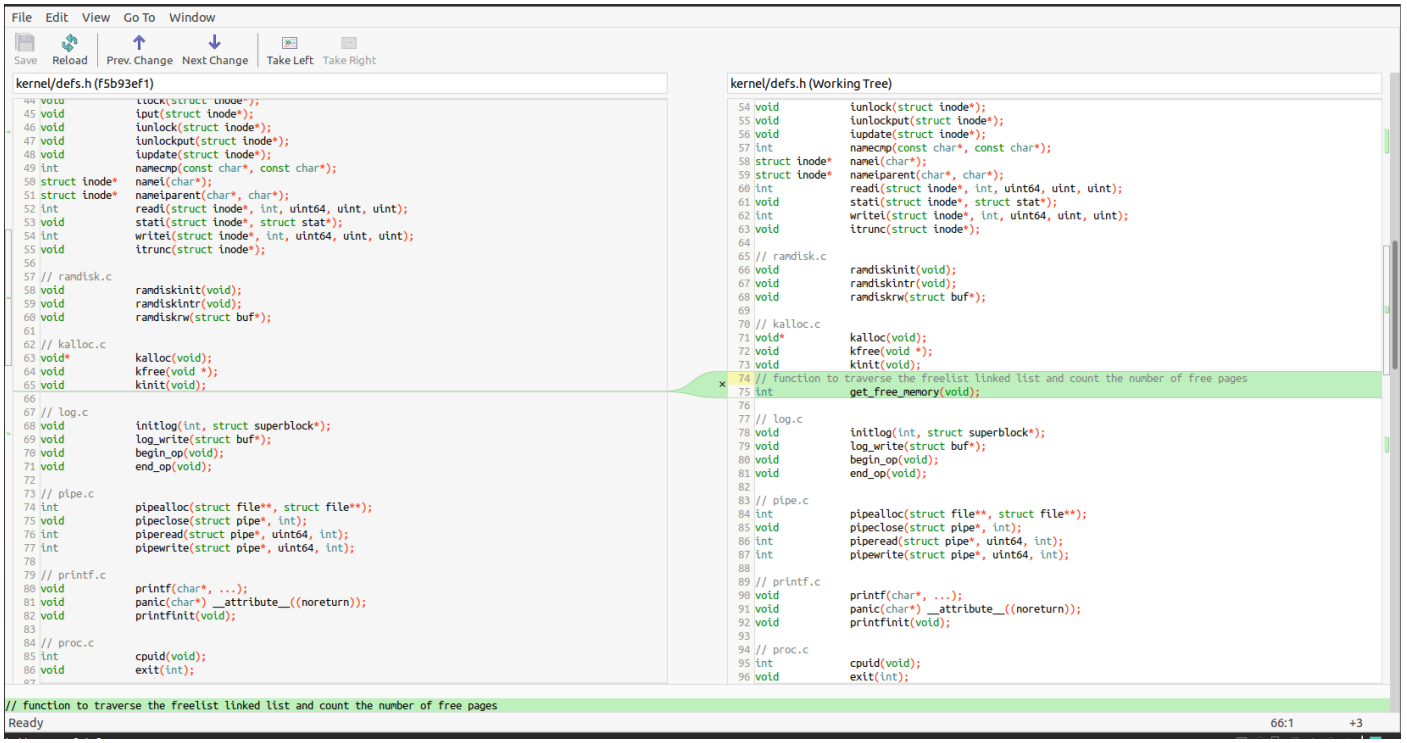
- Added our lab1_test.c file to UPROGS
- Changed the number of CPUs to 1

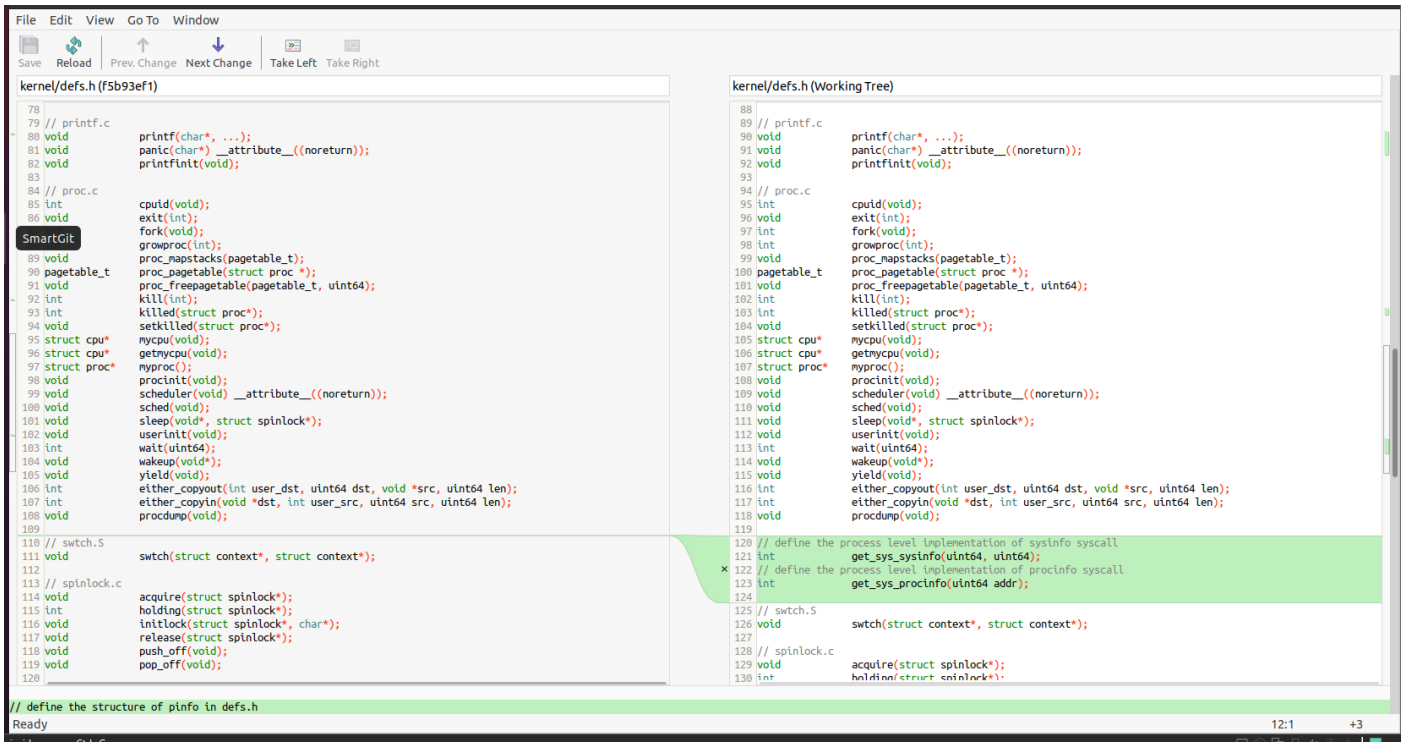


kernel/defs.h

- Create a structure with the required fields
- Create a function *get_free_memory* that returns the number of free pages
- Define the function prototype for sysinfo system call
- Define the function prototype for procinfo system call

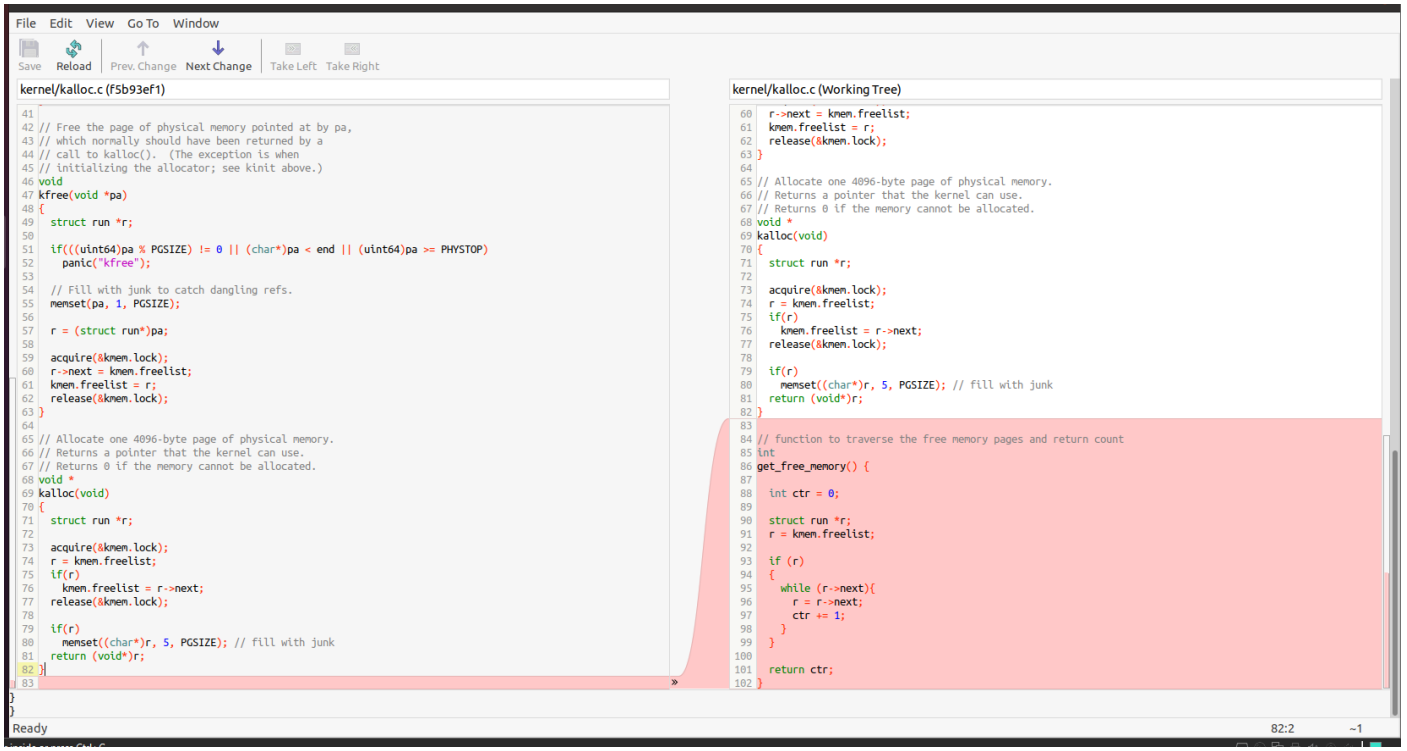






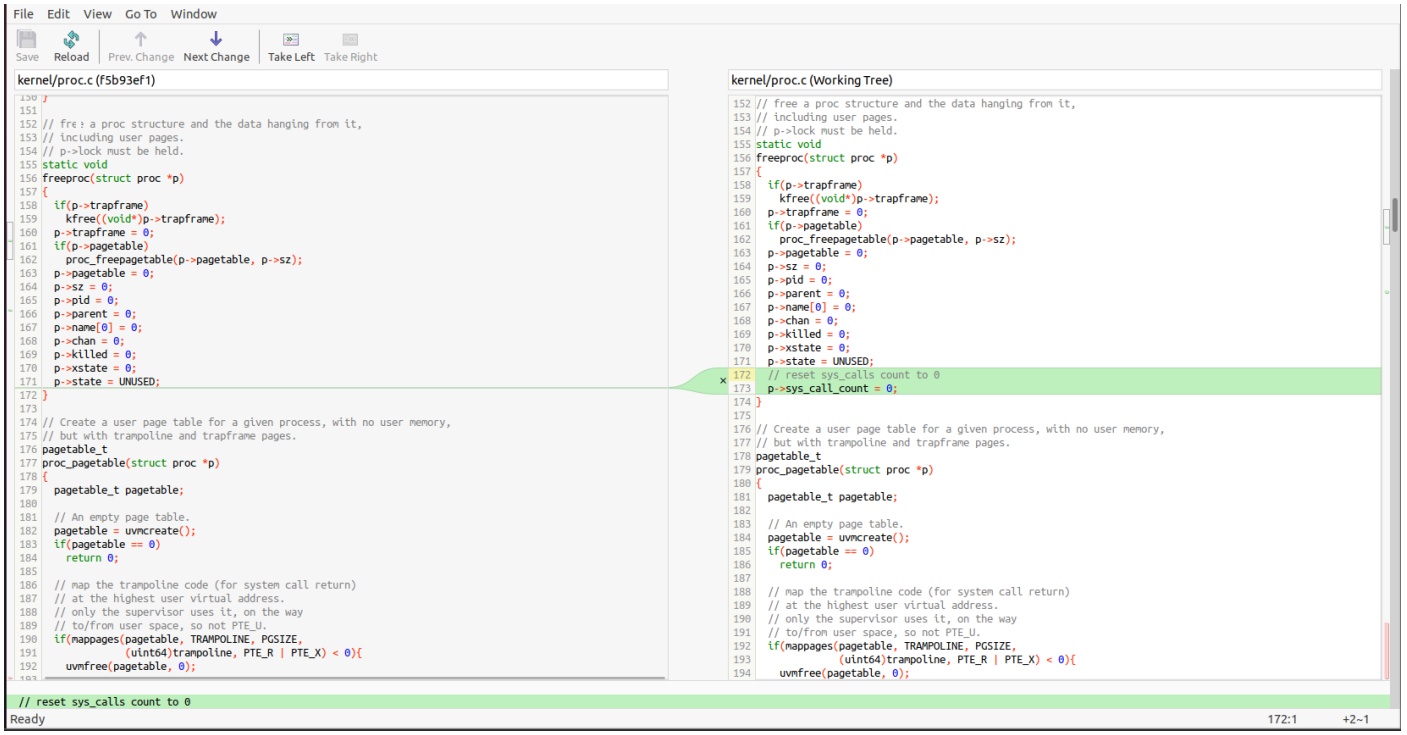
kernel/kalloc.c

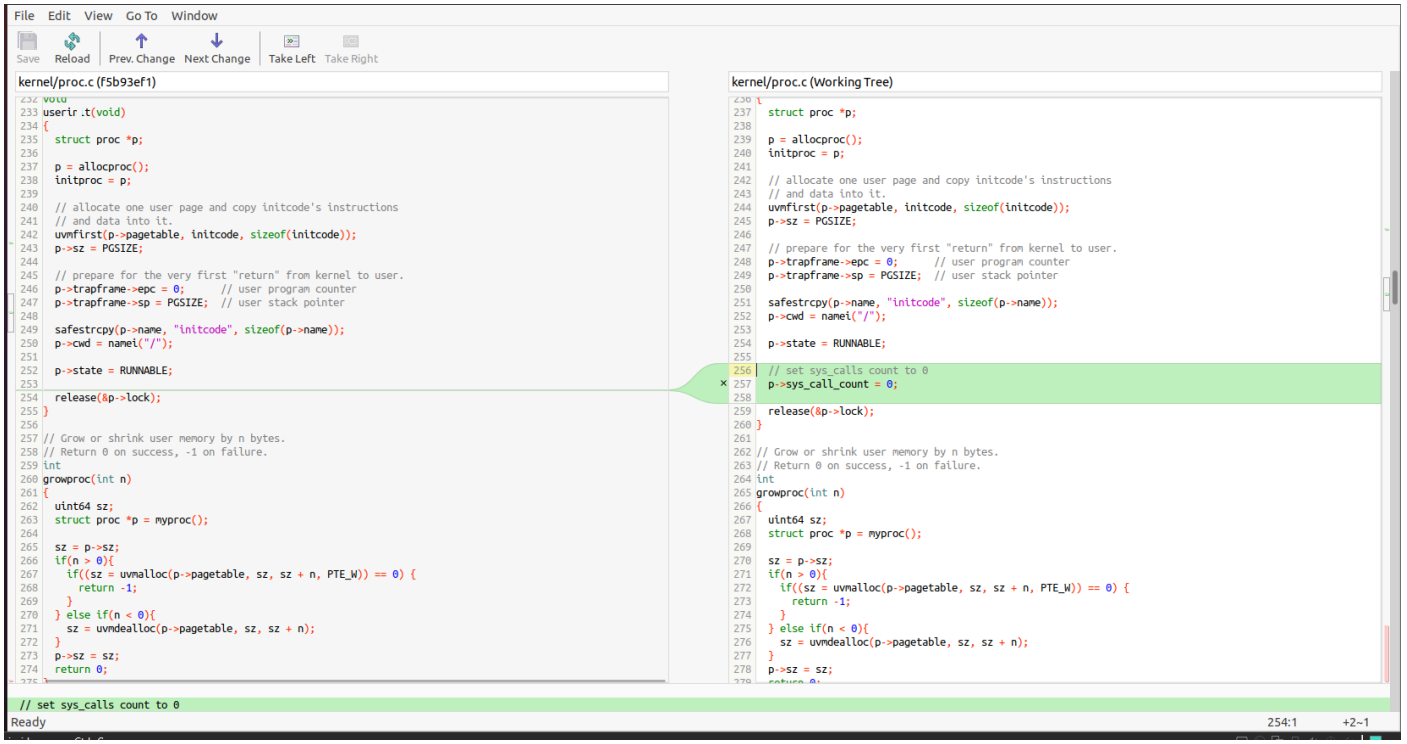
- Write the function body for *get_free_memory* that iterates the freelist linked list and return the number of free pages available.

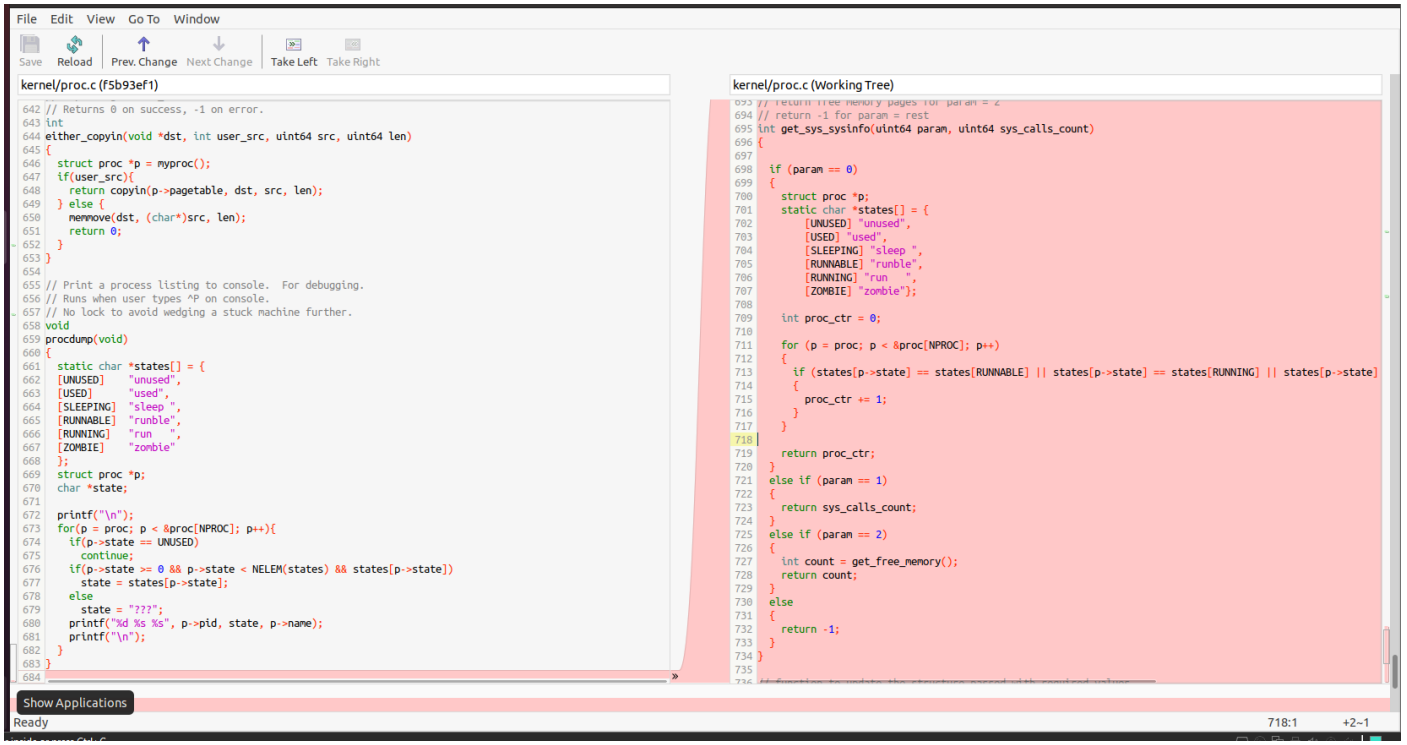


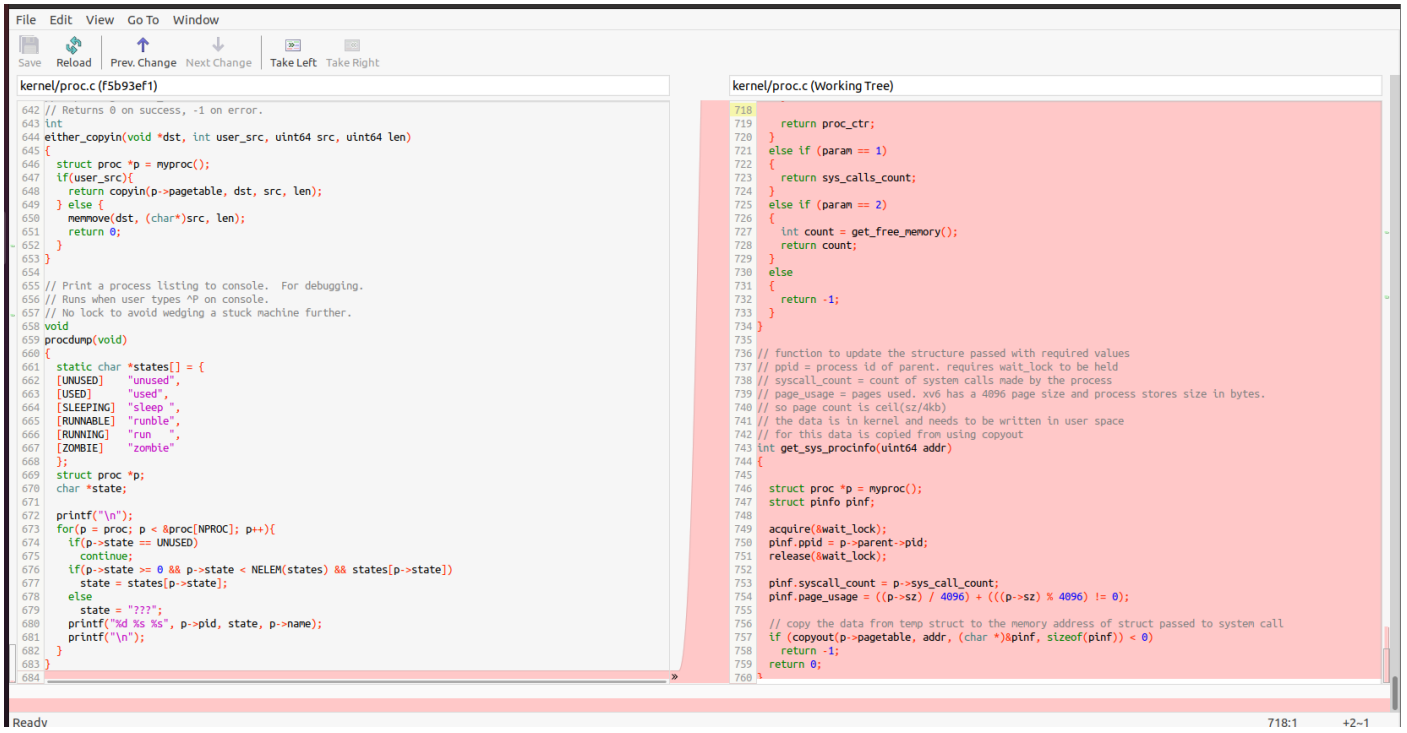
kernel/proc.c

- Set the process level sys_count variable to 0 at the time of process initialization and termination
- Implemented the sys_info system call as per requirements. The function takes parameter as the input (0,1,2) and returns the entities accordingly as per the problem statement. If the parameter passed in the function is anything else than the 3 mentioned parameters, the function will return -1 which implies error.
- Implemented the proc_info system call as per requirements. This function provides information specific to the current process. A pointer of structure *pinfo* is passed to the function. Upon success the function returns **0** while in case of a failure it returns **-1**.



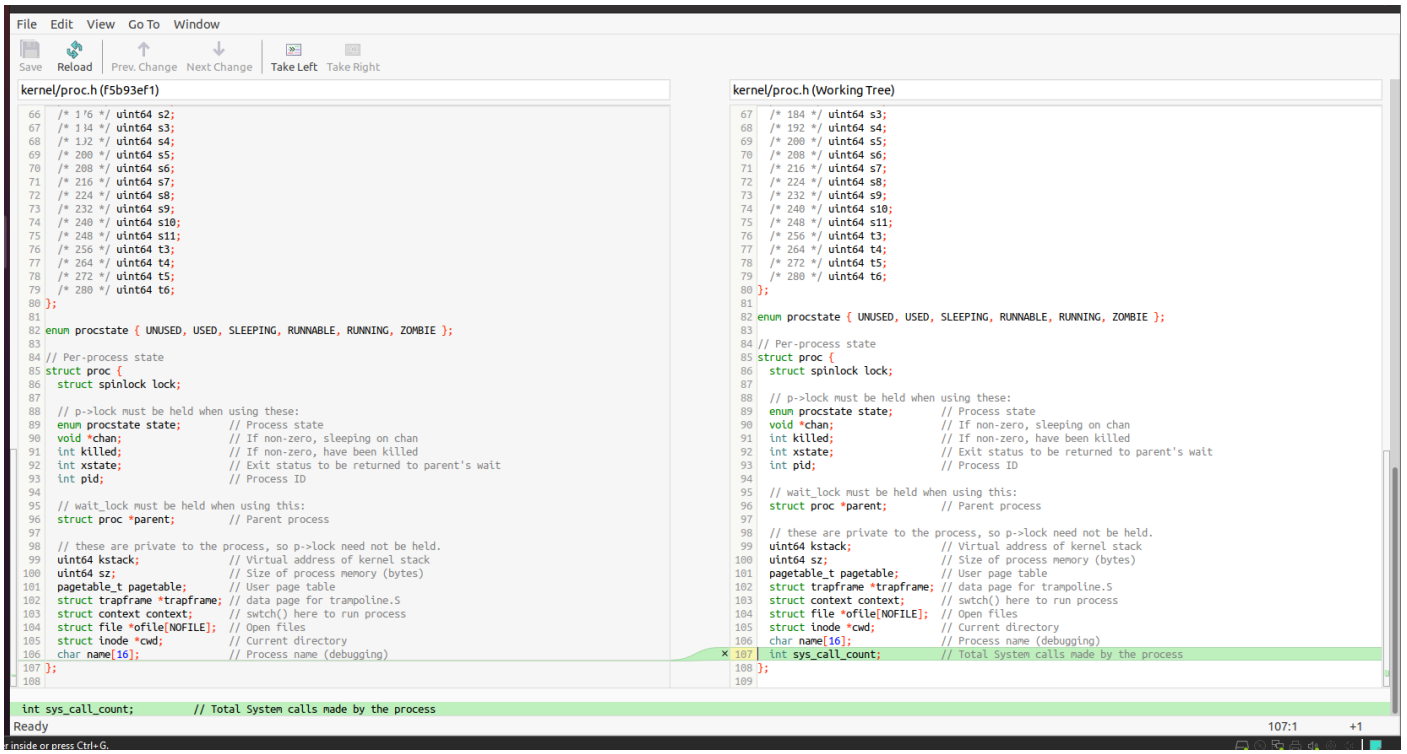

File Edit View Go To Window
Save Reload Prev. Change Next Change Take Left Take Right
kernel/proc.c (f5b93ef1)
kernel/proc.c (Working Tree)
// set sys_calls count to 0
Ready
254:1
+2~1


File Edit View Go To Window
Save Reload Prev. Change Next Change Take Left Take Right
kernel/proc.c (f5b93ef1)
kernel/proc.c (Working Tree)
Show Applications
Ready
718:1
+2~1



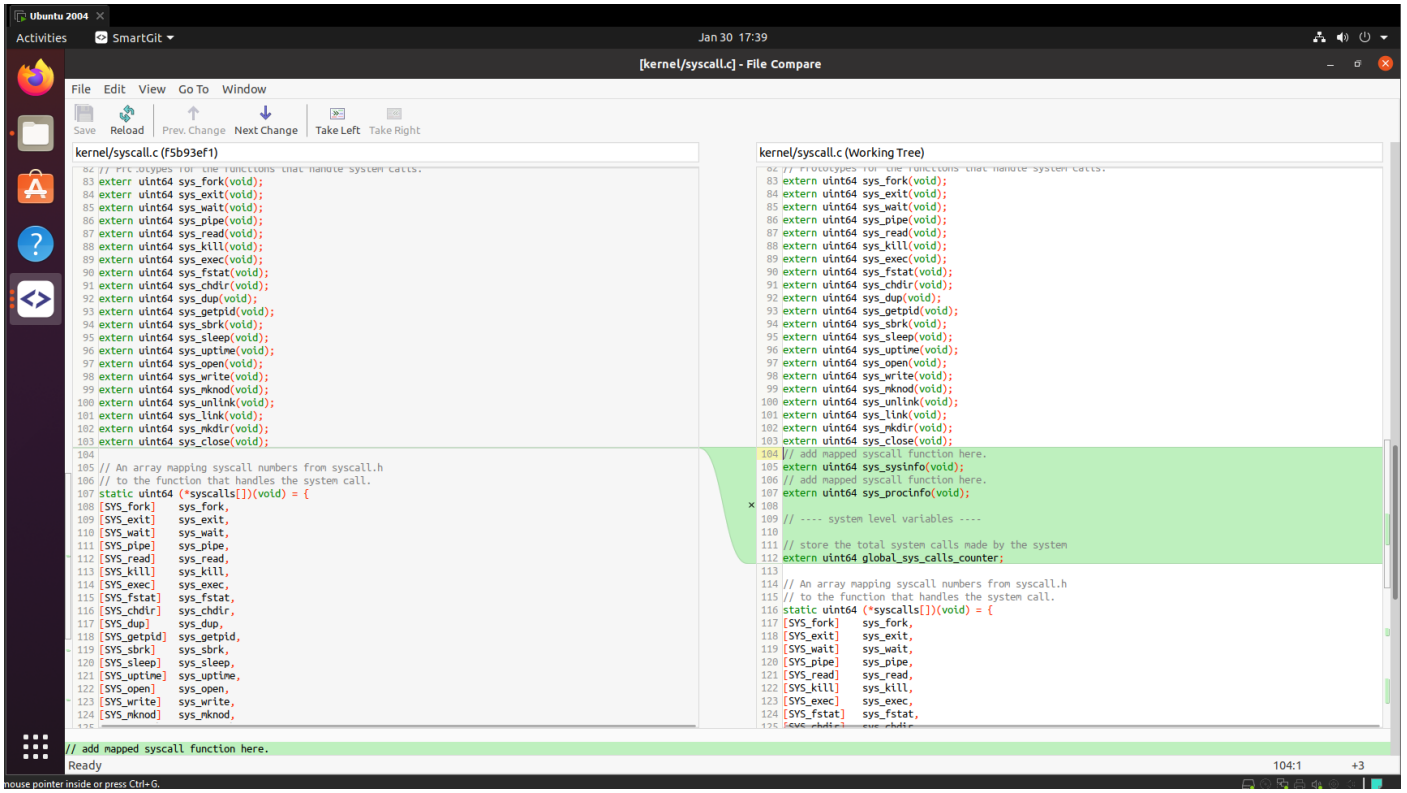
kernel/proc.h

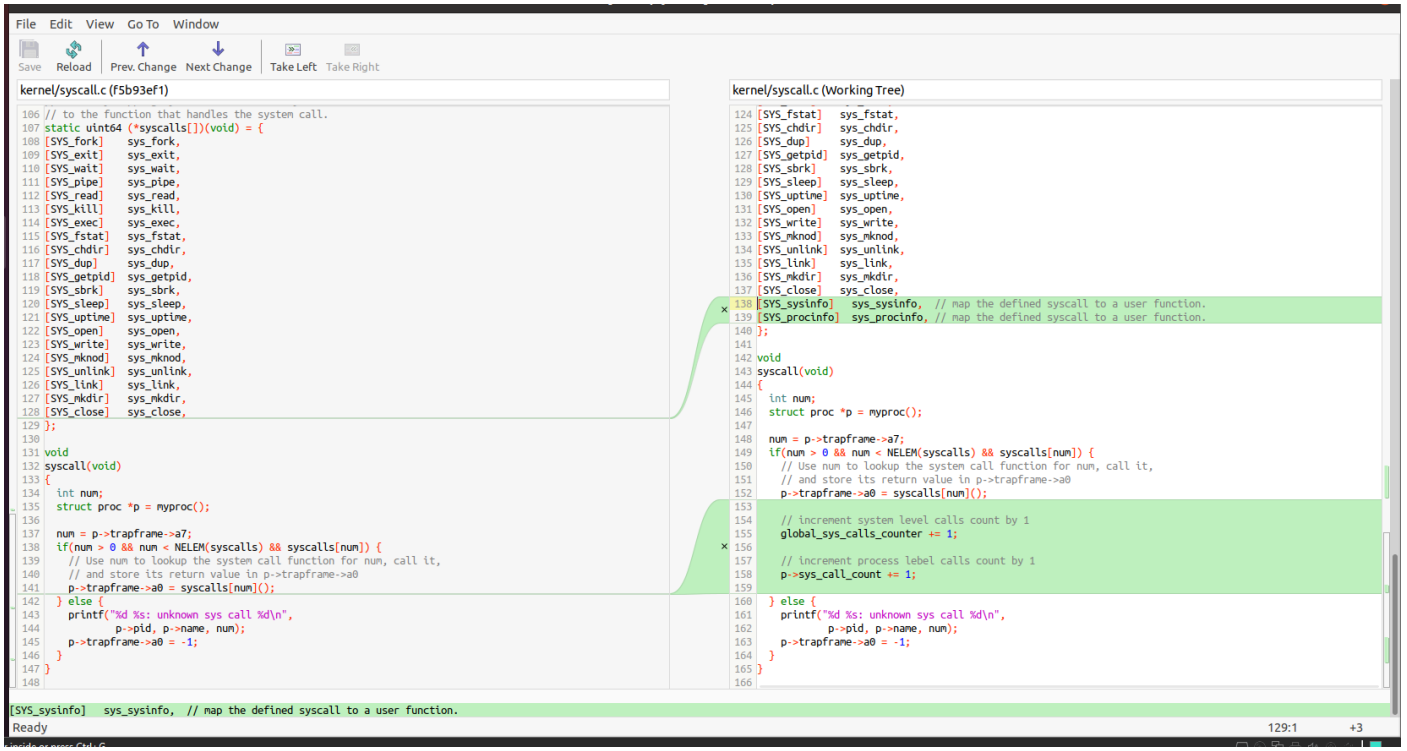
- Added a process level variable to store the number of system calls made



kernel/syscall.c

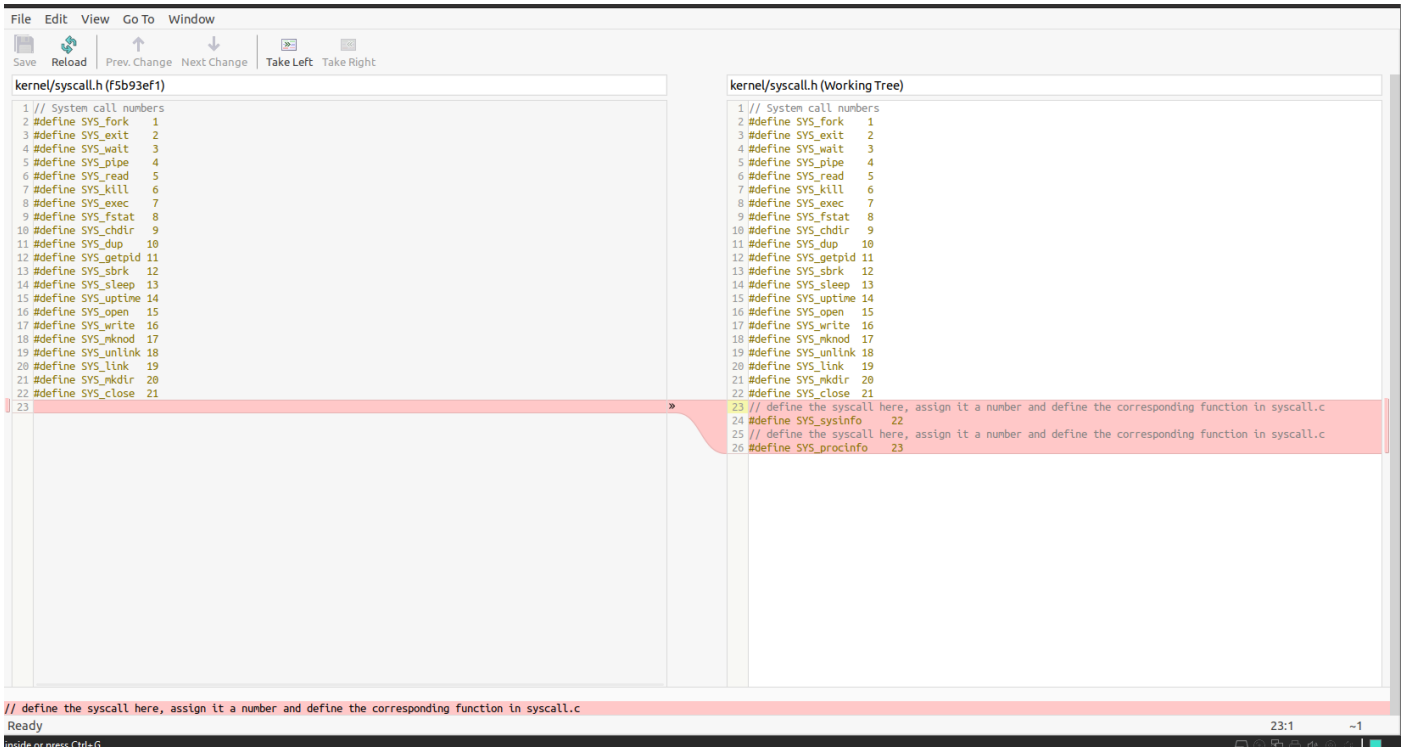
- Created function definitions to be mapped with function calls
- Created system level variable to store all system calls
- Add an entry in syscalls array that maps syscall to its function definitions
- Increment system level syscall counter by 1 after a syscall is made
- Increment the process level syscall counter by 1 after a syscall is made





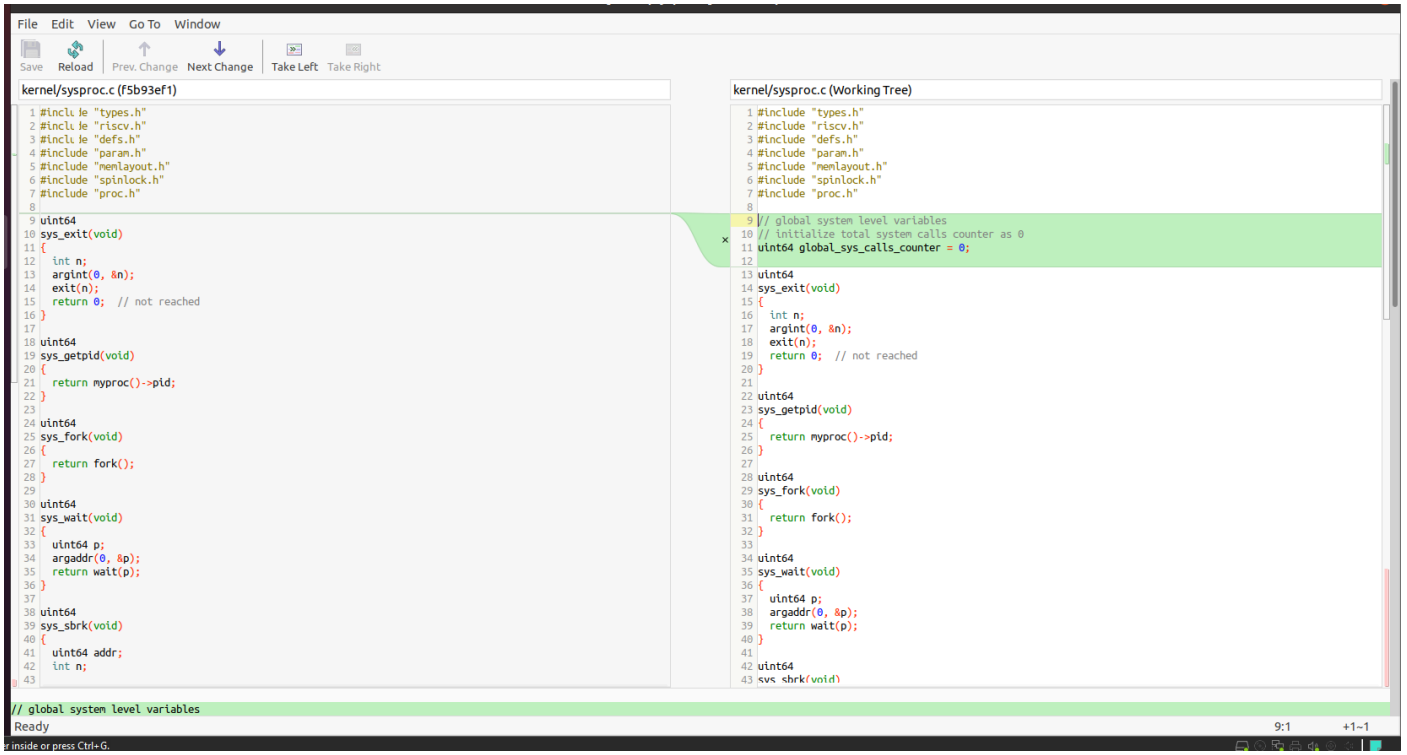
kernel/syscall.h

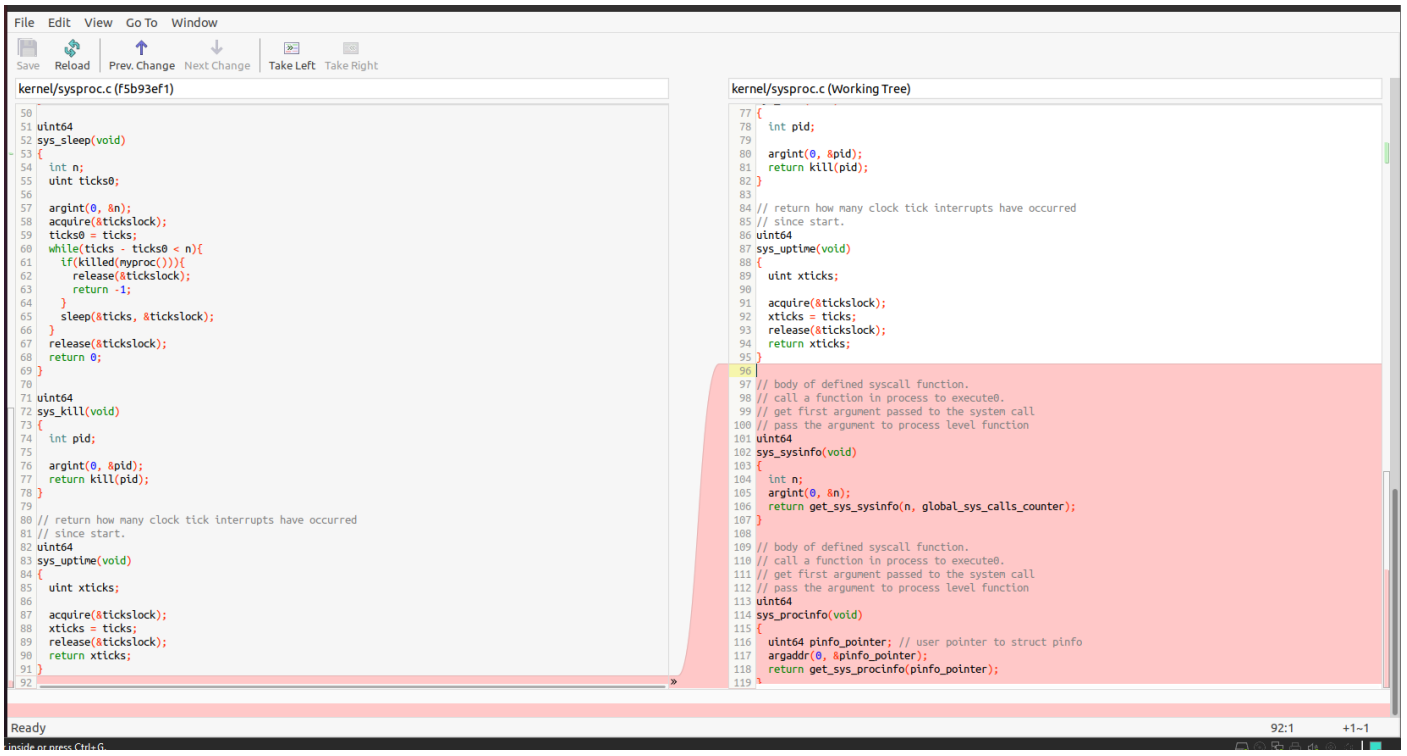
- Add proc_info system call and its number
- Add sys_info system call and its number



kernel/sysproc.c

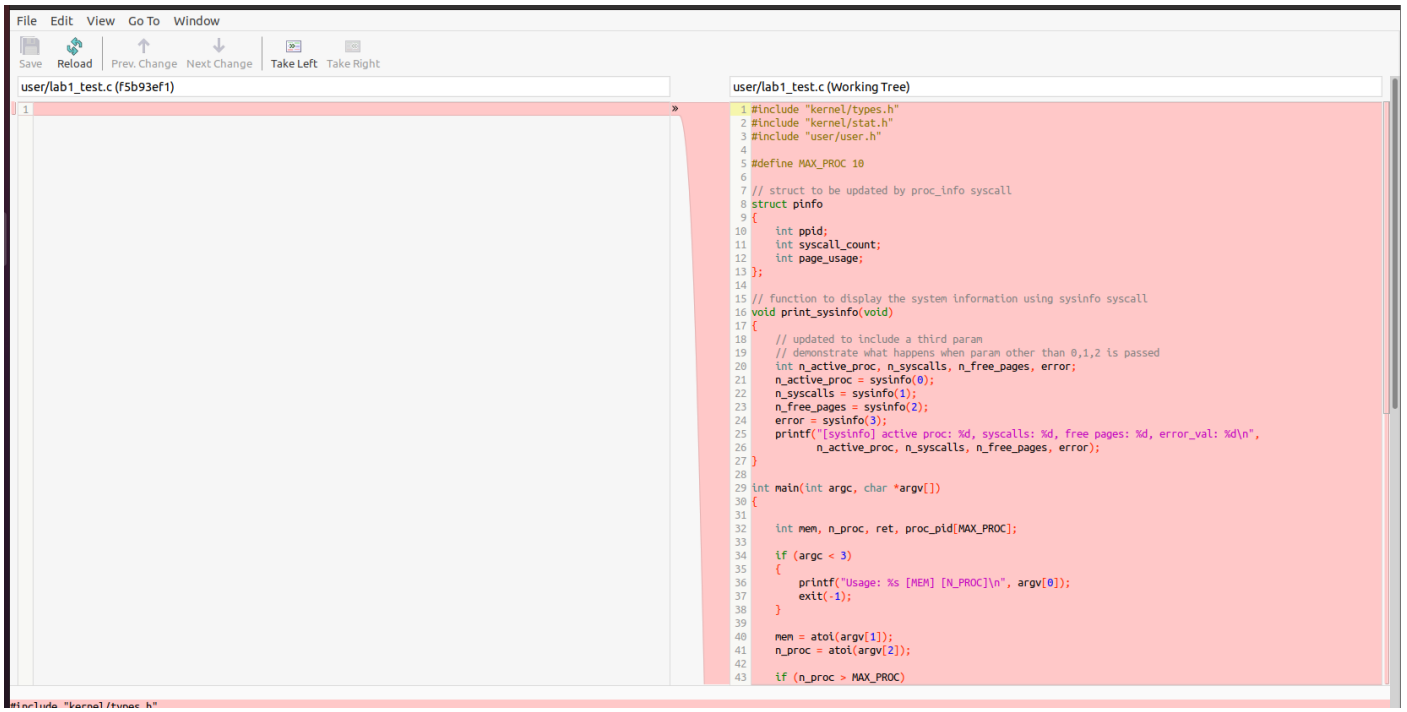
- Initialize the system level syscall counter from 0
- Implement and call sysinfo system call with passed arguments
- Implement and call procinfo system call with passed arguments

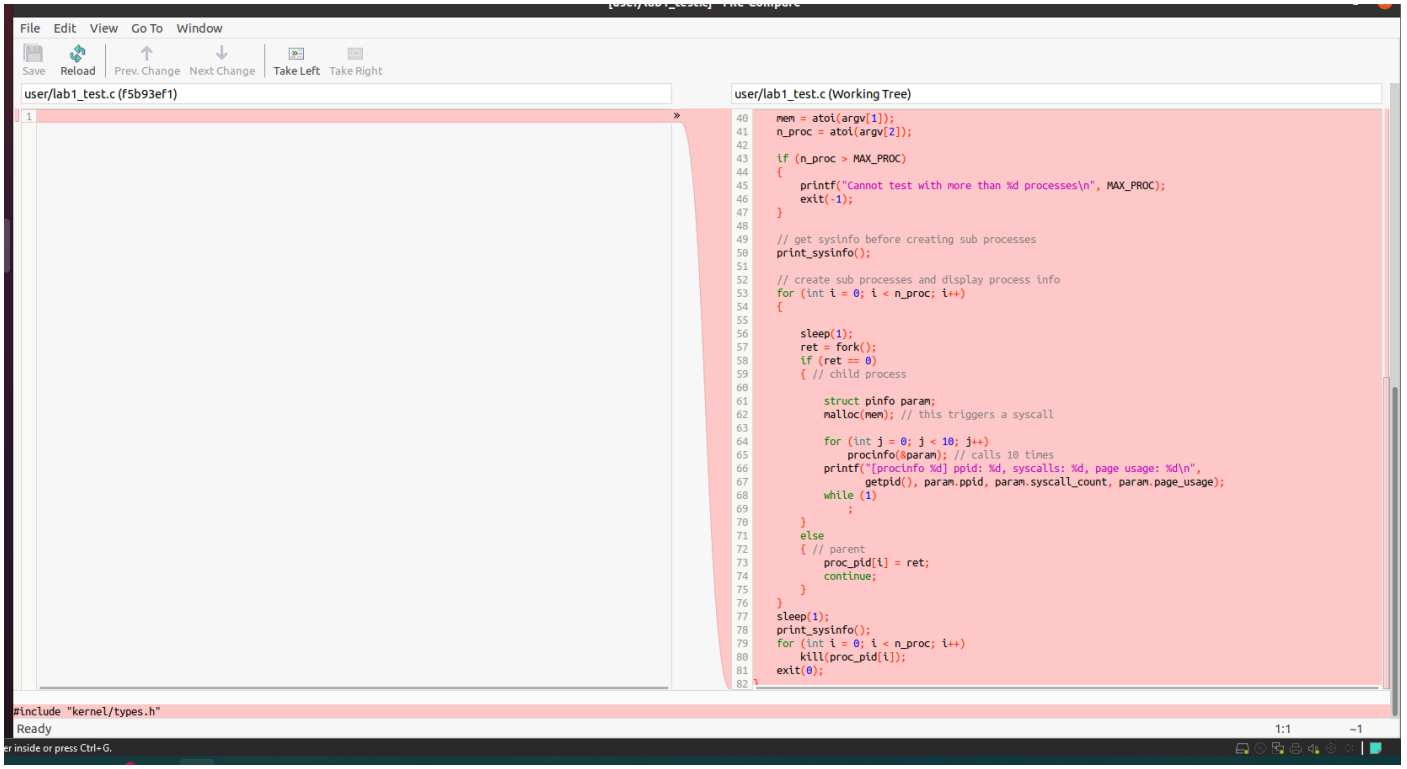




user/lab1_test.c

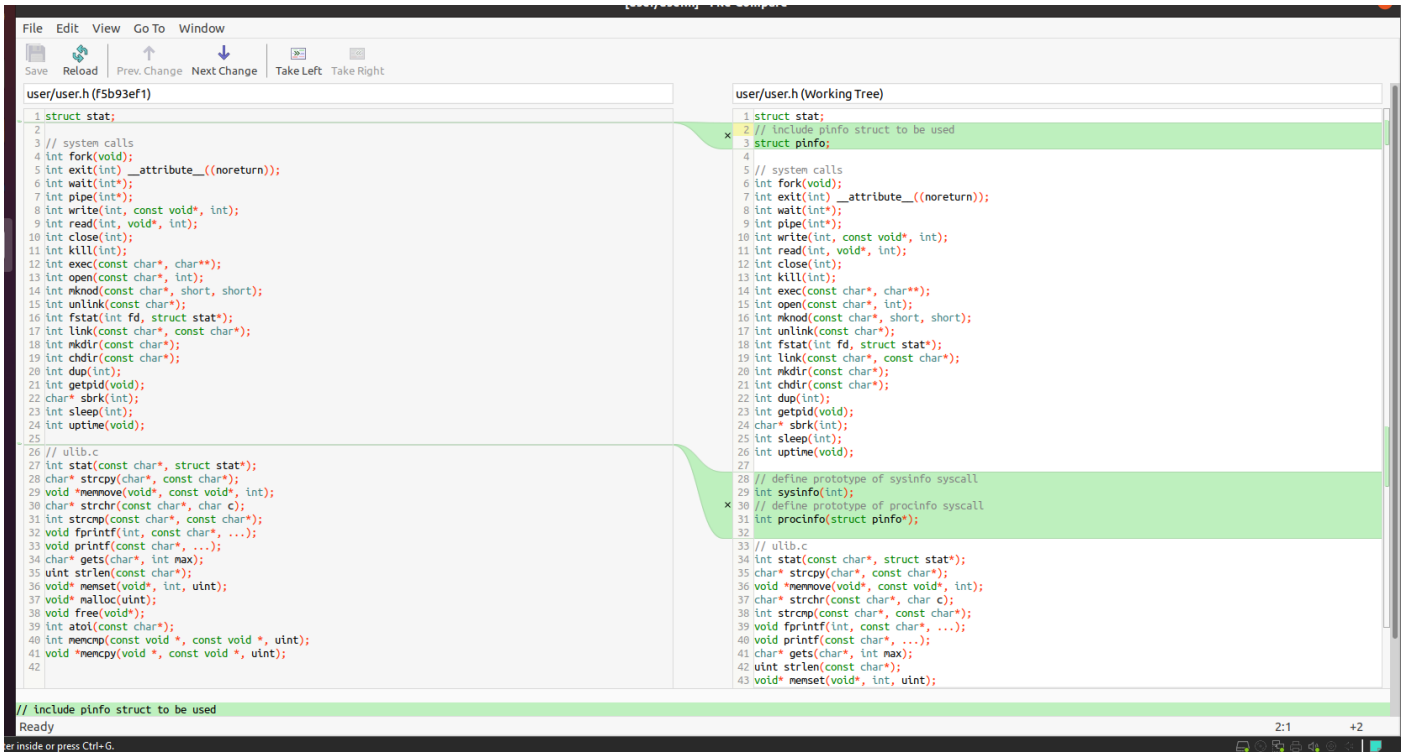
- Changes made to test file as given in lab 1





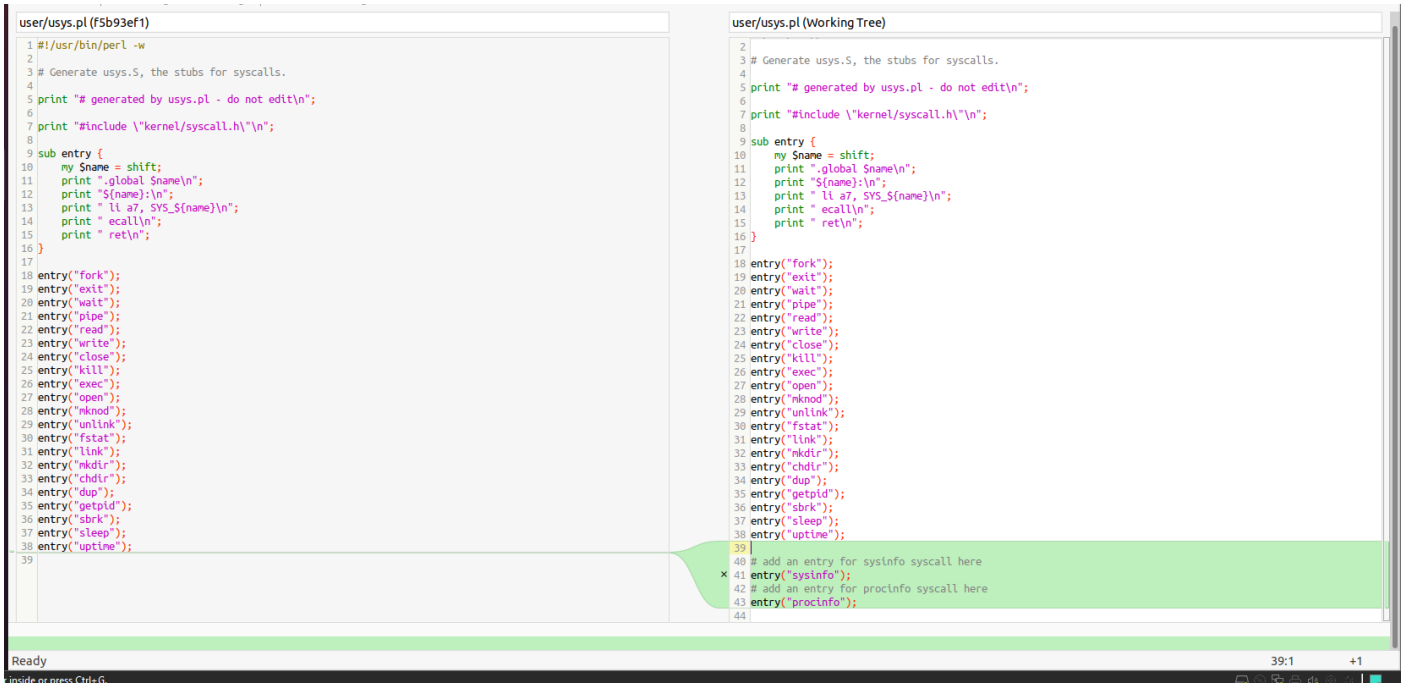
user/user.h

- Declare pinfo struct
- Declare function definitions for sysinfo and procinfo

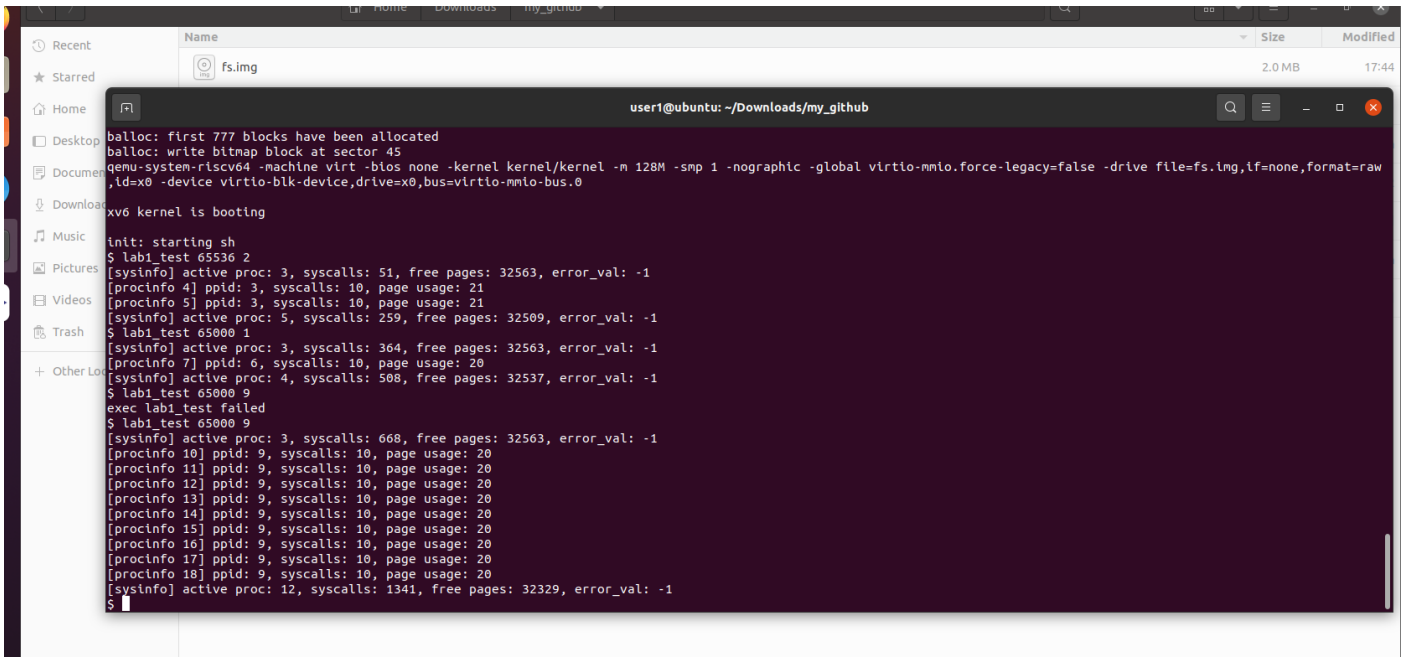


user/usys.pl

- Add an entry for sysinfo system call
- Add an entry for procinfo system call



Results



Summary of contributions

- Member 1: Yash Aggarwal (yagga004@ucr.edu)
  - Implemented system call count, struct passing and data writing in proinfo
  - Implemented page_usage in procinfo
- Member 2: Nityash Gautam (ngaut006@ucr.edu)
  - Implemented PID in procinfo

    - Implement param == 2 for sysinfo - free memory pages
- Member 3: Parth Bhatt (pbhat029@ucr.edu)
    - Implemented param == 1 for sysinfo - total system calls
    - Implemented param == 0 for sysinfo - total active processes
    - Implemented other param handling for sysinfo