Project 1 Free Response Question

Q: Read this article: https://medium.com/@flag_seeker/xv6-system-calls-how-it-works-c541408f21ff and project description file. Write summary about each file you have read/changed and explain why they are important. (user.h, syscall.h, syscall.c, usys.S, sysproc.c, proc.c, proc.h, vector.S, trapasm.S, trap.c) (1pt each)

Answer:

- `user.h`
  - I have modified the file in order to add `trace` and `date` system call.
  - This file specifies what system calls or library functions are available to user-level process.
- `syscall.h`
  - I have added the macros to define `SYS_trace` and `SYS_date` an integer for its system call number
  - This file defines the index of each system call so that the program can map the system call name to an integer as defined in this file.
- `syscall.c`
  - I have added the entry for `sys_trace` and `sys_date` in `syscalls` array, a new array to map each system call index to its name, and the logic to print the process information when trace mode switch is turned on
  - This file defines a few fetch-data operation, the array of `syscalls` to map syscall index to specific syscall function, and the logic to make the system call and store returned value to the `eax` register in the current process. A lot of system-related operations are not available without this file.
- `usys.S`
  - I have added two patterns for `trace` and `date` such that I add two blocks of code to move value to specified register and call interrupt after substitution by the macro
  - This file specifies the process to move `SYS_{syscall name}` to `%eax` and call interrupt when any system call is called by the user-level process. Without this, the user may not assign value to `%eax` pointer correctly and fail to call the corresponding system call.
- `sysproc.c`
  - I have added the `sys_date` and `sys_trace` function to implement the logic of reading current time or modifying the process's `tms(trace mode switch)` field
  - This file defines the behaviour of many system-related calls. It will carry out the necessary check and perform the relevant task with kernel-level priviledge or call the corresponding function. Without this file, the validation of certain value is not enforced.
- `proc.c`
  - I have added an additional step for initialization to assign `totalsyscall` field of a to-be-allocated process to `0`
  - This file defines relevant functions used by a process. The behaviour of the process is unclear without this file.
- `proc.h`

    - I have added two fields `tms` and `totalsyscall` to the definition of `proc struct`
    - This file defines the `struct` or `enum` that are relevant to process.
- `vector.S`
    - I did not touch this file during the project 1
    - This file defines a long list of interrupt vectors and their corresponding procedure to store value. It is important as it will execute appropriate interrupt service based on the type of interrupt
- `trapasm.S`
    - I did not touch this file during the project 1
    - This file defines the logic to save and restore CPU's state during trap and ensures the proper context switch
- `trap.c`
    - I did not touch this file during the project 1
    - This file defines the kernel-level routine to handle different traps and manages the transition betwene user and kernel mode.