Student 1: dadem002
Student 2: dsale010
Team Number: 56

Lab 1 - Write Up

proc.h: Added int exit_status to struct proc because we need each proc's own status.

proc.c:

- put the additional parameters to respective functions.
- Added exit_status for current **PCB's** a certain status code by updating the pointer to the object's **exit_status** variable.[**curproc->exit_status = status;**]
- For wait and waitpid we updated the status pointer with **current PCB's** exit_status. This way, on every PCBs return we would have that PCB's own status code.
    - [*status = p->exit_status;] this updates the status pointer with current PCB's own exit_status
- In waitpid we added an if statement inside the process loop (which checks all the processes) that helps to identify the process that has the same pid as it was given in waitpid's pid parameter. **If there is a match**, then we run the usual wait code:
    - if(p->pid == pid) → this will make sure that the process we are waiting for has the parameter pid
- In waitpid added an else condition for WNOHANG option, if the child is not a zombie, it returns 0 so that we understand that child is still running.

param.h:

- Added #define WNOHANG 1, so that when WNOHANG option is called, it will enumerate it to 1.

sysproc.c:

- Included parameter collection in exit, wait and waitpid functions.
- Used argint for int values such as: exit()'s status, waitpid's option
- Used argptr for pointer's that are used in waitpid and wait functions.

usys.S:

- Added waitpid as a new system call function

defs.h:

- Updated functions exit, wait and waitpid with the new parameters that are wanted from proc.c

user.h, **syscall.c** and **syscall.h:**

- Added waitpid function calls to each of the files.

Makefile:

- Updated CPU from 2 to 1
- Added new test file lab1test.c to UPROGS and EXTRA