NC State University

Department of Electrical and Computer Engineering

ECE 565: Fall 2023

Project 1

by

RAGHUL SRINIVASAN

UnityID: rsriniv7

(200483357)

Questions

Q1. What is the maximum number of processes accepted by Xinu? Where is it defined?

Maximum number of processes is defined by "**NPROC**". It is defined in **config/Configration** and the value is set to 100. It is overwritten by **config/conf.h** (to **100)** which in turn is overwritten to **8** by **include/process.h.**

Q2. What does Xinu define as an "illegal PID"?

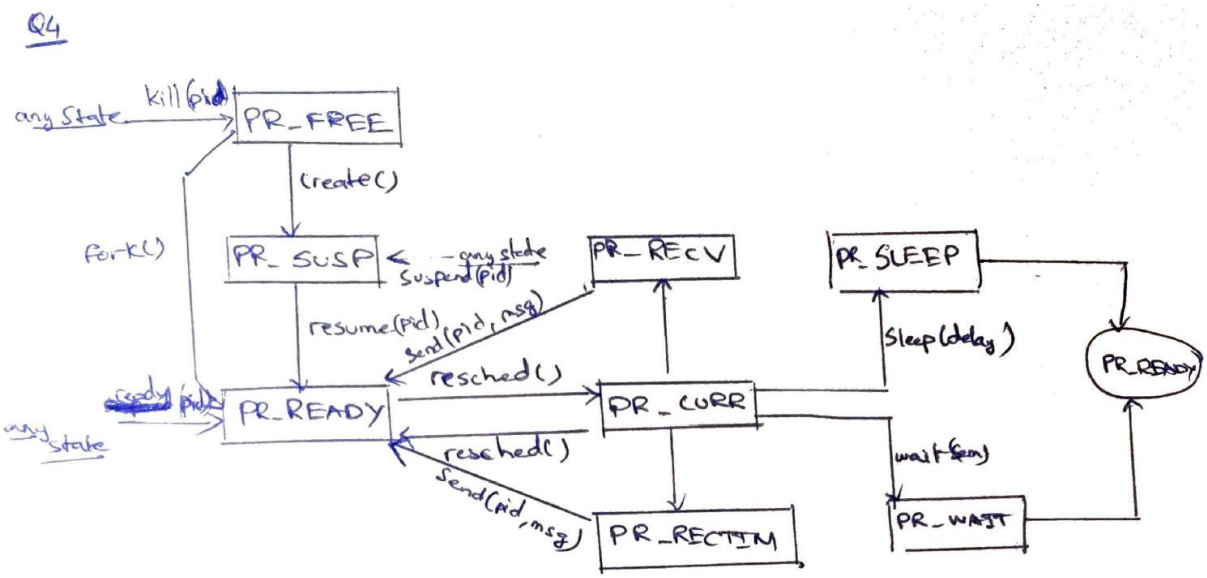
PID is illegal if

- PID < 0
- PID > NPROC
- Corresponding state of the process in process table is PR_FREE (proctab[pid].prstate)

Q3. What is the default stack size Xinu assigns each process? Where is it defined?

Default stack size is 65536 bytes. Defined in include/process.h (#define INITSTK)

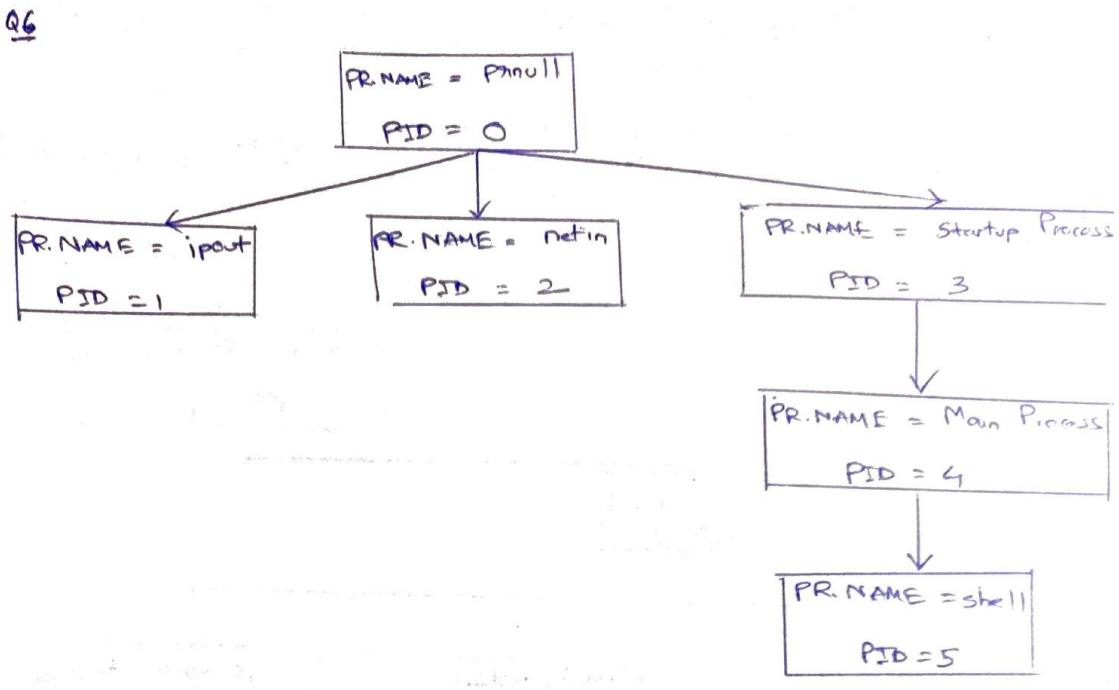
Q4. Draw Xinu's process state diagram.



Q5. When is the shell process created?

Shell Process is created by main process (in system/main.c). In main process, Shell process is re-created whenever shell process is ended.

Q6. Draw Xinu's process tree (including the name and identifier of each process) when the initialization is complete.



Q7. what is the effect of the "receive()" call in the test cases provided? What would happen if that function call was not present?

Receive() will make the parent process to wait until the child process sends a message (via prmsg parameter in process table).

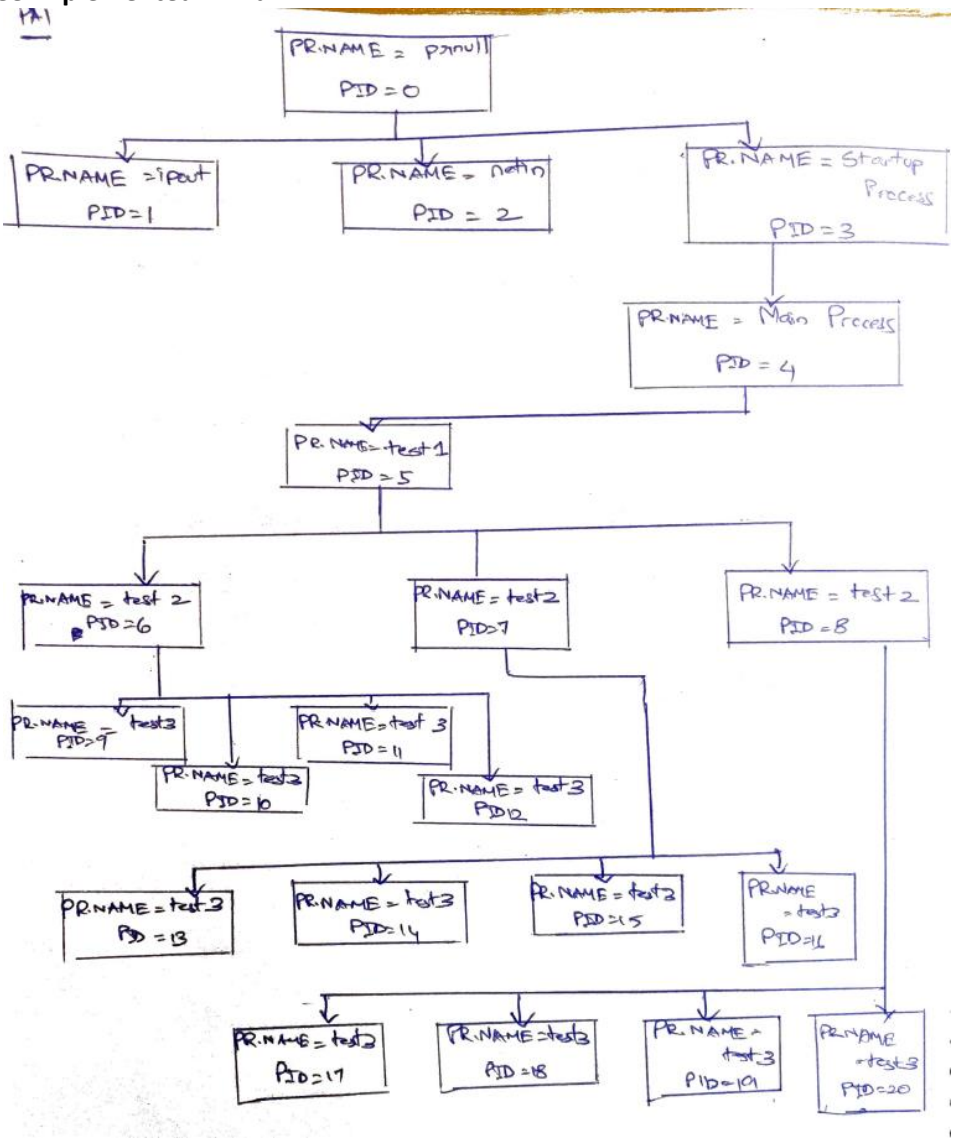
If receive() is not present, parent process wont wait for child process and hence parent process might finish before child processes resulting in varying output.

Programming Assignment 1(Cascading termination)

Below files have been modified for the implementation of cascading termination.

- In include/process.h, **user_process** parameter is added in process control table to differentiate between user processes and system processes.
- In system/create.c, **user_process** is set to accordingly (true if it is user process, false if system process)
- In system/initialize.c, **user_process** is set to "false" for nullproc.
- In system/kill.c, if the process to be killed is a user_process, implemented recursive calling of kill() is implemented to kill all its children and grand-children.
- In system/main.kill, test case with two generation of user-processes are generated and selectively killed to check the process tree if only the relevent child processes are killed.

Process tree implemented in main.kill



Programming Assignment 2(Process creation and stack handling)

Below files have been modified for the implementation of Process creation and stack handling.

- In include/prototypes.h, sync_printf(), fork() and newpid() are added.
- In system/fork.c, implemented fork implementation.
    - Get all parent process parameters from PCB and save it.
    - Create a new process and set its state to PR_READY (child process).
    - Copy stack contents from parent to child process.
    - Update all the base pointers of the child by adding the offset.
    - Get the instruction from which child is to start executing and update in child stack.
    - Initialize all the registers and flags for the child.
    - Insert the child process entry to ready queue for the scheduler to know that the child process is ready.