MPX-OS VER 2.70 Programmers Manual

PREPARED BY

Username programming group

Brandon Steel

James Jackman

Chelsey Randolph

Mohammad Alzayer

How To Use This Manual

This manual is intended for programmers who intend to write their own programs with this OS and give insight into what each function in every file is intended to do and give example inputs and outputs.

This OS utilizes the C-programming language

Table of Contents

h. **delete_pcb**
i. **block_pcb**
j. **unblock_pcb**
k. **suspend_pcb**
l. **resume _pcb**
m. **set_pcb_priority(char name[30], int new_priority)**
n. **show_pcb**
o. **show_ready**
p. **show_blocked**
q. **show_all**

1.1 Command Handler - comhand.c

a. comhand()

This function uses a command buffer and a menu based system in order to execute commands the user inputs. The input is a name of a function that will be executed.

The function itself has no defined input, though it does read user input from the system requirement write in order to execute the users command.

b. version()

Version is a function with no input and a write to the terminal for an output, it outputs a character array for the current version of the MPX operating system.

c. displayAllCommands()

This function will display all the commands that are available for the user to use.

d. inputHelp(char helpBuffer[])

inputHelp is the help function that is menu based similar to the command handler. The input is the word “help” + function name. This function will describe what the other functions do.

Parameters:

char helpBuffer[]: the buffer containing the input from the user

e. shutDown()

This function first prompts the user with a confirmation message, then either exits back to comhand or shuts down the system.

f. displayMenu()

At the start of our system, this will display our “Username” logo with the menu once.

1.2 Date - date.c

g. **getDate()**

This function gets the current date that is set on the system.

h. **setDate()**

This function allows the user to set a new date on the system.

1.3 Time- time.c

i. **setTime()**

This function allows the user to set a new time on the system.

j. **getTime()**

This function gets the current time that is set on the system.

k. **BCDToDEC(int num)**

This function converts a binary number to a decimal number.

Parameters:

int num: the number to be converted

l. **DECToBCD(int num)**

This function converts a decimal number to a binary number.

Parameters:

int num: the number to be converted

1.4 Polling - polling.c

m. **init_polling(char *buffer, int *count)**

This function collects input from the user's keyboard.

Parameters:

char *buffer : pointer that represents the input buffer

int *count: pointer that represents the size of the input buffer

2.1 PCB - pcb.c

a. allocate_pcb()

Allocate_pc is the allocate function that is type of internal functions to set up the pcb in memory and returns the size of the pcb.

b. setup_pcb(char *name, int pclass, int priority)

Setup_pcb is an internal function that includes a name of a process, the process class, and a priority. This function is used to allocate the memory to the new pcb by calling allocate_pcb(). The process name it must be at least 8 characters. The priority must be a number from 0 through 9. The process class must be either 0 for the user or 1 for the system.

Parameters:

char *name : pointer that represents the name of the process.

int pclass : an integer that represents the class classes whether it's 0 or 1.

int priority: an integer that represents the priority number.

c. free_pcb(PCB* pcb)

free_pcb is an internal function that's allow the user to free the pcb in memory by sys_free_mem().

Parameters:

PCB* pcb : a pointer that points to the process control block in the stack function in head file.

d. find_pcb(char *process_name)

Find_pcb is an internal function that allows the user to find a specific pcb. This function checks the four linked lists (ready_queue, blocked_queue, suspend_ready_queue, and suspend_block queue) for the given pcb name. *Parameters*:

char* process_name : is a pointer that is used to compare the name given by the user to the current names of processes.

e. **insert_pcb(PCB *pcb)**

Insert_pcb is an internal function that allows the user to insert a pcb in an appropriate queue.

Parameters:

PCB *pcb: a pointer to the pcb which the user wants to insert into a specific queue

f. **remove_pcb(PCB *pcb)**

remove_pcb is an internal function that allows the user to remove a pcb from the queue in which it is currently stored.

Parameters:

PCB *pcb: a pointer to the pcb which the user wants to remove from a specific queue.

g. **create_pcb()**

This function will call setup_pcb and insert_pcb to insert the new pcb in the appropriate queue. It also checks the data being sent in from the user to make sure it meets the requirements from the system.

h. **delete_pcb(char name[30])**

This function will take a pcb name, find it, remove it, and then free all associated memory.

Parameters:

char name[30]: name of the pcb the user wants to delete.

i. **block_pcb(char name[30])**

This function will take a pcb name the user has given and set its state to blocked.

Parameters:

char name[30]: name of the pcb the user wants to block.

j. **unblock_pcb(char name[30])**

This function will take a pcb name the user has given and set its state to ublocked.

Parameters:

char name[30]: name of the pcb the user wants to unblock.

k. **suspend_pcb(char name[30])**

This function will take a pcb name the user has given and set its state to suspended.

Parameters:

char name[30]: name of the pcb the user wants to suspend.

l. **resume_pcb(char name[30])**

This function will take a pcb name the user has given and set its state to not suspended.

Parameters:

char name[30]: name of the pcb the user wants to unsuspend.

m. **set_pcb_priority(char name[30], int new_priority)**

This function will take a pcb name the user has given and set its new priority to the given integer.

Parameters:

char name[30]: name of the pcb the user wants to set its priority.

int new_priority: integer of the new priority.

n. show_pcb(char name[30])

This function will take a pcb name the user has given and displays information about its state.

Parameters:

char name[30]: name of the pcb the user wants to display.

o. show_ready()

This function shows all the pcbs in the ready queue.

p. show_blocked()

This function shows all the pcbs in the blocked queue.

q. show_all()

This function shows all the pcbs in the system.