OS Allstars' MPX Project

User Manual

Authored by David Maxson, Nathan Grey, Lennon Jones, and Brad Kersting

Current Version: 4.0 [R3/R4]

Last Updated: 18-Mar-21

MPX Startup

In the terminal, enter

```
qemu-system-i386 -nographic -kernel kernel.bin -s
```

to load the MPX Core.

Upon startup, you will be greeted by the MPX startup message:

```
  Welcome to OS Allstar's MPX Project. Enter help for a list of
commands.

>>
```

At this point the core has been booted and the MPX system is now waiting to receive user commands.

MPX Terminal

Any time you encounter "`>>`" followed by a blinking cursor, MPX is waiting for a command from the user.

This MPX implementation features a Linux-style command line menu system, simply type the desired command. The backspace key will delete the most recent typed character and the left and right arrow keys can be used to navigate the cursor through your input in the left and right directions, respectively. When you have typed the desired command, press enter to submit the request.

Currently no implemented commands have optional clauses, commands are as is. Later versions will document the handling of optional command clauses.

If your input was not recognized or otherwise determined invalid by the system, MPX will notify you.

Otherwise, your entered command will be handled appropriately.

MPX Commands (as of v4.0)

Basic commands:

`help`

This command in its current iteration simply lists all the commands available to the user (similar to this section of the User Manual). As further version of the MPX system unfold this will most likely change to be followed by the command you are looking for information about. For example, `help settime` would give the user information specifically about the `settime` command.

`version`

This command prints to the screen the current version of the MPX system, in this case v1.0

`shutdown`

This command will initiate the shutdown process for the MPX system. You will be asked to confirm your selection before the system shuts down.

`gettime`

This command will print the current time to the screen, as it is stored in the RTC registers

`settime`

This command will prompt the user to enter a user desired time, then set that value to the RTC registers

`getdate`

This command will print the current date to the screen, as it is stored in the RTC registers

`setdate`

This command will prompt the user to enter a user desired date, then set that value to the RTC registers

Process Control Block (PCB) Commands:

`deletepcb [pcb_name]`

This command will delete a selected pcb from all 4 of the PCB queues, removing them completely from the system. `pcb_name` must be a valid PCB already in existence.

`suspendpcb [pcb_name]`

This command will set the selected PCB's state to **suspended** and insert it into the appropriate PCB queue. `pcb_name` must be a valid PCB already in existence.

`resumepcb [pcb_name]`

This command will set the selected PCB's state to **unsuspended** and insert it into the appropriate PCB queue. `pcb_name` must be a valid PCB already in existence.

`setpriority [pcb_name]`

This command will set the selected PCB's priority to a new user desired priority, possibly changing its location withing the queues. `pcb_name` and `pcb_priority` must both be valid (*see `createpcb` for valid inputs for these parameters*.)

`showpcb [pcb_name]`

This command will display the attributes of a selected PCB in the terminal. These attributes include `process_name, class, state, suspended_status,` and `priority` `showreadypcb`

This command will display all processes contained within the two ready queues in the terminal.

`showblkpcb`

This command will display all processes contained within the two blocked queues in the terminal.

`showpcbs`

This commands will display all existing processes, in all four queues, in the terminal.

New system commands for dispatching:

`yield` *(R3 only)*

This command simply causes an interrupt at line 60, causing the system to be able to execute processes other than the command handler

`loadr3`

This command will load 5 processes into memory, each with varying priority, to allow the testing of the MPX's dispatching capabilities

`inf`

This command will create the Infinite process, and add it to the ready, not suspended queue. This process is an application, not a system process, therefore it can be deleted if the user suspends it.

`Alarm` *(System Process)*

This command will prompt the user to enter a time and message for an alarm to be printed to the console. Time must be entered in HH:MM:SS format.

Note: System processes cannot be edited by the user.