MP3 - fourDudes Issue Tracker - jhweil2, zega2, rsharm12, scpuri2

Timestamp	Description	Type	Status	Priority	Solution	Assigned To	Date Completed
17/10/2016 22:41:19	This is a test description of an issue encountered in MP3	Other	Completed	1 Show Stopper	This is a fake solution to the test issue	Charlie	
17/10/2016 22:55:16	Keyboard handler output misaligned with keyboard input	Bug	Completed		The array which stored the mapping from scancodes to ascii characters was missing one value after the row of numbers, which offset the rest of the mapping by one character.	Jack	
17/10/2016 23:00:58	Only one interrupt being handled at a time	Bug	Completed		send_eoi function in i8259.c was not properly calculating the EOI signal and was sending the same signal to the master and the slave in the case of an interrupt on a slave IRQ line.	Jack	
17/10/2016 23:02:43	Spurious interrupt calling the default interrupt handler and in turn masking all lower priority interrupts.	Bug	Completed		Turned out to be interrupt coming from IRQ0 on the PIC for unknown reasons. Added a handler for this case that simply ignores the interrupt and sends EOI response to PIC.	Rahul	
17/10/2016 23:05:28	Enabling paging caused crash almost immediately.	Bug	Completed		Needed to reorder the manipulation of control registers in assembly routine as part of init_paging function in paging.c. By enabling paging with cr0 before having finished setting cr3 and cr4 the system was crashing immediately on next instruction since 4MB pages were not enabled (and kernel code resides in one of these pages).	Jack	
17/10/2016 23:07:16	No page fault exception generated for invalid memory access between 0 and 4MB.	Bug	Completed		Had changed initialization of page directory to mark all entries as present with intention of changing it back. Forgot to change back, so all memory was left marked present and therefore considered valid.	Charlie	
17/10/2016 23:12:10	Suspicion of iret instructions issued in C corrupting stack for interrupt service routines.	Investigation	Completed		Created file asm_wrapper.S that acts as a wrapper for ISRs to manage the stack, save registers, and then call the appropriate C handler.	Saurav	
17/10/2016 23:13:26	Exception handlers are not using assembly wrappers since for now they do not return. In the future, they will need assembly wrappers in order to preserve registers, get error codes and associated debug info, and issue an iret when appropriate.	Enhancement	Completed		Utilized the assembly wrapper function and implemented squashing of user level programs that generated exceptions. Exceptions now return via iret when appropriate, still no handling of associated debug info.	Saurav	
17/10/2016 23:15:02	System call handler (entry 0x80 in the IDT) is not handled with appropriate assembly linkage. Will need to be separated out into its own assembly file with a more complex wrapper than asm_wrapper.S currently includes before serious work on system calls can begin.	Enhancement	Completed		It now has its own separate function in assembly which in turn calls specific C functions depending on the value of eax. In turn, these functions call other functions specific to the file type.	Charlie	
24/10/2016 22:43:54	We could not correctly read the file titled "verylargetxtwithverylongname.txt"	Investigation	Completed			Jack	
24/10/2016 22:50:34	Read_data was unable to correctly access the data blocks associated with the given inodes.	Bug	Completed		One line of code had an improper type cast which, when changed to a pointer, made the function work correctly.	Jack	
24/10/2016 22:53:04	Page fault exception when trying to access the filesys_img.	Bug	Completed		Wrote a helper function in paging.c to mark present the pages related to the filesys_img.	Charlie	
24/10/2016 22:55:34	When the term_write writes eighty characters and then a newline, it skips a line.	Bug	Completed		Consolidated some of the case statements so that it only changes line once per line.	Rahul	
24/10/2016 23:00:11	O/C/R/W for different devices are written, but there is no jump table that system calls use to access these handlers.	Skipped Task	Completed		Added assembly linkage for syscallls so that all syscalls from the user level will call the correct functions in the kernel.	Rahul	
24/10/2016 23:03:29	Never wrote a separate read function for when the file type is directory.	Skipped Task	Completed		Wrote one when the grade sheet was released and it was revealed to be necessary.	Rahul	
26/10/2016 21:37:07	Time test	Other	In Progress	1 Show Stopper		Jack	
08/11/2016 21:33:35	Getting a TSS exception when we tried to do a context switch into a new process.	Investigation	Completed		Realized that there is really no need to utilize the NT flag in the EFLAGS register.	Saurav	
08/11/2016 21:36:41	The system crashes when we try to open too many tasks or close out of all the tasks (specifically, calling "exit" in the original shell).	Enhancement	Completed		Set up a macro MAX_PROCESSES that controls how many process can be run, and if a new process is called that exceeds this number, it will fail to execute. Likewise, closing out of the last shell will lead to a new shell being immediately executed.	Saurav	

MP3 - fourDudes Issue Tracker - jhweil2, zega2, rsharm12, scpuri2

Timestamp	Description	Type	Status	Priority	Solution	Assigned To	Date Completed
08/11/2016 21:40:03	Added the capability to squash user level programs (via sys_halt) from the exception handlers in exceptions.c.	Enhancement	Completed			Saurav	
08/11/2016 21:41:15	User programs are not able to read keyboard input when using the appropriate syscall.	Bug	Completed		At the beginning of the key_read function, added an sti() so that it could correctly receive interrupts.	Charlie	
08/11/2016 21:48:39	Typing during a key_read function overwrites information written to the screen by a term_write function, which is counter-intuitive to how the UI should work.	Bug	Completed		Made all of the different functions that write to the screen share the same screen_x and screen_y variables so that they cannot overwrite each other. This leads to a cleaner UI.	Charlie	
08/11/2016 21:52:46	Expanded functionality of paginc.c to support multiple processes and simpler mapping with virtual memory. Added more page directories to support multiple processes organized as a 2d array adding to the original array in checkpoint 1. Also implemented a map page function for extended (4MB pages) for use with sys-execute.	Enhancement	Completed			Charlie	
10/11/2016 23:06:19	Executing more than 6 programs in a row (one task id, given a full PCB size) results in a page fault. It was then discovered that the number of successful programs that can be executed is related to the size of the PCB.	Bug	Completed	2 Urgent	The dependency on the size of the PCB drove us to analyze the kernel memory associated with a given process. After tracing the EBP/ESP values on successive executions of a given program we saw clearly that there was some consistent offset added to the values of both each time. This led us to reviewing the code related to the EBP/ESP and realized that the esp0 value in the TSS was being updated with altered ESP and apparently needed our kernel EBP given that it does not change during execution.	Jack	
10/11/2016 23:16:34	The user code for grep is getting stuck in an infinite loop after searching through all files.	Bug	Completed		It was determined that the directory read function was never returning zero after reading the last file. Adding an extra check to see that all files have been read ended this problem.	Charlie	
17/11/2016 18:37:01	Pressing the F# keys to change terminals was not working. But using other keys worked.	Bug	Completed	4 Needs to be addressed	Realized that an if clause was not allowing keystrokes beyond a certain value to be read. Changed this to include F1-3.	Saurav	
04/12/2016 05:15:41	Running a large amount of processes would cause the OS to restart, but restart to a working condition.	Bug	Completed	4 Needs to be addressed	By increasing the size of the pcb struct, we could see the parent ebp was slowly inching forward to the point that it would cause the OS to reboot.	Rahul	
04/12/2016 05:17:56	After running fish then attempting to run pingpong, the video memory would not update on a given terminal until forced by a terminal switch.	Bug	Completed	3 Wrong Functionality	This was solved by remapping the video memory in sys_execute immediately after loading the program.	Rahul	
04/12/2016 05:22:05	get_args was not clearing the pcb buffer prior to trying to fill it with the new args, so it was keeping part of the previous arg input. Through gdb, we determined the args buffer was not cleaned correctly and therefore the buffer had garbage.	Bug	Completed	2 Urgent	The solution was to clear the buffer before copying new arg so that the buffer is just the argument followed by a null character.	Charlie	
04/12/2016 05:27:59	Noticeable lag when running multiple RTC based programs like pingpong. It looked like scheduling but the lag time was too severe to be scheduling.	Bug	Completed	2 Urgent	We set a separate flag for each terminal and had the rtc_read function return to all terminals. This completely removed basically all delay/lag in the OS.	Jack	
04/12/2016 22:57:19	F-keys pressed and released one would cause the buffer to end and were unable to type until newline was given.	Bug	Completed	5 Should be addressed	Bug in the keyread for the fkeys because the fkeys were not mapped in the array so the behavior is unexpected.	Jack	