

MAMA

Generated by Doxygen 1.9.3

1 MAMA	1
2 Who did what table	3
3 Class Index	5
3.1 Class List	5
4 File Index	7
4.1 File List	7
5 Class Documentation	9
5.1 cmcb_s Struct Reference	9
5.1.1 Detailed Description	9
5.1.2 Member Data Documentation	9
5.1.2.1 addr	10
5.1.2.2 name	10
5.1.2.3 next	10
5.1.2.4 prev	10
5.1.2.5 size	10
5.1.2.6 type	10
5.2 cmd_mapping Struct Reference	11
5.2.1 Member Data Documentation	11
5.2.1.1 cmd_handler	11
5.2.1.2 cmd_name	11
5.2.1.3 default_args	11
5.3 context Struct Reference	11
5.3.1 Detailed Description	12
5.3.2 Member Data Documentation	12
5.3.2.1 cs	12
5.3.2.2 ds	12
5.3.2.3 eax	12
5.3.2.4 ebp	13
5.3.2.5 ebx	13
5.3.2.6 ecx	13
5.3.2.7 edi	13
5.3.2.8 edx	13
5.3.2.9 eflags	13
5.3.2.10 eip	13
5.3.2.11 es	14
5.3.2.12 esi	14
5.3.2.13 esp	14
5.3.2.14 fs	14
5.3.2.15 gs	14
5.4 date_time Struct Reference	14

5.4.1 Member Data Documentation	15
5.4.1.1 day_m	15
5.4.1.2 day_w	15
5.4.1.3 day_y	15
5.4.1.4 hour	15
5.4.1.5 min	15
5.4.1.6 mon	15
5.4.1.7 sec	15
5.4.1.8 year	16
5.5 dcb_t Struct Reference	16
5.5.1 Member Data Documentation	16
5.5.1.1 eflag_p	16
5.5.1.2 oper_status	16
5.5.1.3 ready_state	16
5.5.1.4 ring_buffer	17
5.5.1.5 ring_buffer_head	17
5.5.1.6 ring_buffer_tail	17
5.5.1.7 user_read_buf	17
5.5.1.8 user_read_count	17
5.5.1.9 user_write_buf	17
5.5.1.10 user_write_count	17
5.6 footer Struct Reference	18
5.6.1 Member Data Documentation	18
5.6.1.1 head	18
5.7 gdt_descriptor_struct Struct Reference	18
5.7.1 Member Data Documentation	18
5.7.1.1 base	18
5.7.1.2 limit	19
5.8 gdt_entry_struct Struct Reference	19
5.8.1 Member Data Documentation	19
5.8.1.1 access	19
5.8.1.2 base_high	19
5.8.1.3 base_low	19
5.8.1.4 base_mid	20
5.8.1.5 flags	20
5.8.1.6 limit_low	20
5.9 header Struct Reference	20
5.9.1 Member Data Documentation	20
5.9.1.1 index_id	20
5.9.1.2 size	20
5.10 heap Struct Reference	21
5.10.1 Member Data Documentation	21

5.10.1.1 base	21
5.10.1.2 index	21
5.10.1.3 max_size	21
5.10.1.4 min_size	21
5.11 idt_entry_struct Struct Reference	21
5.11.1 Member Data Documentation	22
5.11.1.1 base_high	22
5.11.1.2 base_low	22
5.11.1.3 flags	22
5.11.1.4 sselect	22
5.11.1.5 zero	22
5.12 idt_struct Struct Reference	23
5.12.1 Member Data Documentation	23
5.12.1.1 base	23
5.12.1.2 limit	23
5.13 index_entry Struct Reference	23
5.13.1 Member Data Documentation	23
5.13.1.1 block	24
5.13.1.2 empty	24
5.13.1.3 size	24
5.14 index_table Struct Reference	24
5.14.1 Member Data Documentation	24
5.14.1.1 id	24
5.14.1.2 table	24
5.15 mcb_queue_s Struct Reference	25
5.15.1 Detailed Description	25
5.15.2 Member Data Documentation	25
5.15.2.1 mcb_queue_type	25
5.15.2.2 mcbq_head	25
5.16 page_dir Struct Reference	25
5.16.1 Member Data Documentation	26
5.16.1.1 tables	26
5.16.1.2 tables_phys	26
5.17 page_entry Struct Reference	26
5.17.1 Member Data Documentation	26
5.17.1.1 accessed	26
5.17.1.2 dirty	27
5.17.1.3 frameaddr	27
5.17.1.4 present	27
5.17.1.5 reserved	27
5.17.1.6 usermode	27
5.17.1.7 writeable	27

5.18 page_table Struct Reference	27
5.18.1 Member Data Documentation	28
5.18.1.1 pages	28
5.19 param Struct Reference	28
5.19.1 Member Data Documentation	28
5.19.1.1 buffer_ptr	28
5.19.1.2 count_ptr	28
5.19.1.3 device_id	28
5.19.1.4 op_code	29
5.20 parsed_args Struct Reference	29
5.20.1 Member Data Documentation	29
5.20.1.1 flag_count	29
5.20.1.2 flags	29
5.20.1.3 named_arg_count	29
5.20.1.4 named_arg_names	30
5.20.1.5 named_arg_values	30
5.20.1.6 unnamed_arg_count	30
5.20.1.7 unnamed_args	30
5.20.1.8 unnamed_args_used_so_far	30
5.21 pcb_node_t Struct Reference	30
5.21.1 Detailed Description	31
5.21.2 Member Data Documentation	31
5.21.2.1 pcb	31
5.21.2.2 pcbn_next_pcb	31
5.21.2.3 pcbn_prev_pcb	31
5.22 pcb_queue Struct Reference	31
5.22.1 Detailed Description	32
5.22.2 Member Data Documentation	32
5.22.2.1 pcbq_count	32
5.22.2.2 pcbq_head	32
5.22.2.3 pcbq_tail	32
5.22.2.4 queue_order	33
5.23 pcb_t Struct Reference	33
5.23.1 Detailed Description	33
5.23.2 Member Data Documentation	33
5.23.2.1 pcb_name	33
5.23.2.2 pcb_priority	34
5.23.2.3 pcb_process_class	34
5.23.2.4 pcb_process_state	34
5.23.2.5 pcb_protection_mode	34
5.23.2.6 pcb_stack_bottom	34
5.23.2.7 pcb_stack_top	34

6 File Documentation	35
6.1 /home/maximillian/Desktop/MAMA/include/core/asm.h File Reference	35
6.2 asm.h	35
6.3 /home/maximillian/Desktop/MAMA/include/core/comhand.h File Reference	35
6.3.1 Function Documentation	35
6.3.1.1 comhand()	35
6.4 comhand.h	36
6.5 /home/maximillian/Desktop/MAMA/include/core/interrupts.h File Reference	36
6.5.1 Function Documentation	36
6.5.1.1 init_irq()	36
6.5.1.2 init_pic()	36
6.6 interrupts.h	36
6.7 /home/maximillian/Desktop/MAMA/include/core/io.h File Reference	37
6.7.1 Macro Definition Documentation	37
6.7.1.1 inb	37
6.7.1.2 outb	37
6.8 io.h	37
6.9 /home/maximillian/Desktop/MAMA/include/core/serial.h File Reference	38
6.9.1 Macro Definition Documentation	38
6.9.1.1 COM1	38
6.9.1.2 COM2	38
6.9.1.3 COM3	38
6.9.1.4 COM4	38
6.9.2 Function Documentation	39
6.9.2.1 init_serial()	39
6.9.2.2 polling()	39
6.9.2.3 serial_print()	39
6.9.2.4 serial_println()	39
6.9.2.5 set_serial_in()	40
6.9.2.6 set_serial_out()	40
6.10 serial.h	40
6.11 /home/maximillian/Desktop/MAMA/include/core/tables.h File Reference	41
6.11.1 Function Documentation	41
6.11.1.1 __attribute__()	41
6.11.1.2 gdt_init_entry()	42
6.11.1.3 idt_set_gate()	42
6.11.1.4 init_gdt()	42
6.11.1.5 init_idt()	42
6.11.2 Variable Documentation	42
6.11.2.1 access	42
6.11.2.2 base	42
6.11.2.3 base_high	43

6.11.2.4 base_low	43
6.11.2.5 base_mid	43
6.11.2.6 flags	43
6.11.2.7 limit	43
6.11.2.8 limit_low	43
6.11.2.9 sselect	43
6.11.2.10 zero	43
6.12 tables.h	44
6.13 /home/maximillian/Desktop/MAMA/include/mem/heap.h File Reference	44
6.13.1 Macro Definition Documentation	45
6.13.1.1 KHEAP_BASE	45
6.13.1.2 KHEAP_MIN	45
6.13.1.3 KHEAP_SIZE	45
6.13.1.4 TABLE_SIZE	45
6.13.2 Function Documentation	45
6.13.2.1 _kmalloc()	45
6.13.2.2 alloc()	46
6.13.2.3 init_kheap()	46
6.13.2.4 kfree()	46
6.13.2.5 kmalloc()	46
6.13.2.6 make_heap()	46
6.14 heap.h	47
6.15 /home/maximillian/Desktop/MAMA/include/mem/paging.h File Reference	48
6.15.1 Macro Definition Documentation	48
6.15.1.1 PAGE_SIZE	48
6.15.2 Function Documentation	48
6.15.2.1 clear_bit()	48
6.15.2.2 first_free()	49
6.15.2.3 get_bit()	49
6.15.2.4 get_page()	49
6.15.2.5 init_paging()	49
6.15.2.6 load_page_dir()	49
6.15.2.7 new_frame()	49
6.15.2.8 set_bit()	49
6.16 paging.h	50
6.17 /home/maximillian/Desktop/MAMA/include/string.h File Reference	51
6.17.1 Function Documentation	51
6.17.1.1 atoi()	51
6.17.1.2 isspace()	51
6.17.1.3 itoa()	51
6.17.1.4 memset()	52
6.17.1.5 strcat()	52

6.17.1.6 strcmp()	52
6.17.1.7 strcpy()	52
6.17.1.8 strlen()	52
6.17.1.9 strtok()	53
6.18 string.h	53
6.19 /home/maximillian/Desktop/MAMA/include/system.h File Reference	54
6.19.1 Macro Definition Documentation	54
6.19.1.1 asm	54
6.19.1.2 cli	54
6.19.1.3 GDT_CS_ID	55
6.19.1.4 GDT_DS_ID	55
6.19.1.5 hlt	55
6.19.1.6 iret	55
6.19.1.7 no_warn	55
6.19.1.8 nop	55
6.19.1.9 NULL	55
6.19.1.10 sti	56
6.19.1.11 volatile	56
6.19.2 Typedef Documentation	56
6.19.2.1 size_t	56
6.19.2.2 u16int	56
6.19.2.3 u32int	56
6.19.2.4 u8int	56
6.19.3 Function Documentation	56
6.19.3.1 klogv()	56
6.19.3.2 kpanic()	57
6.20 system.h	57
6.21 /home/maximillian/Desktop/MAMA/kernel/core/interrupts.c File Reference	57
6.21.1 Macro Definition Documentation	59
6.21.1.1 ICW1	59
6.21.1.2 ICW4	59
6.21.1.3 io_wait	59
6.21.1.4 PIC1	59
6.21.1.5 PIC2	59
6.21.2 Function Documentation	59
6.21.2.1 bounds()	59
6.21.2.2 breakpoint()	60
6.21.2.3 coprocessor()	60
6.21.2.4 coprocessor_segment()	60
6.21.2.5 debug()	60
6.21.2.6 device_not_available()	60
6.21.2.7 divide_error()	60

6.21.2.8 do_bounds()	60
6.21.2.9 do_breakpoint()	60
6.21.2.10 do_coprocessor()	61
6.21.2.11 do_coprocessor_segment()	61
6.21.2.12 do_debug()	61
6.21.2.13 do_device_not_available()	61
6.21.2.14 do_divide_error()	61
6.21.2.15 do_double_fault()	61
6.21.2.16 do_general_protection()	61
6.21.2.17 do_invalid_op()	61
6.21.2.18 do_invalid_tss()	62
6.21.2.19 do_isr()	62
6.21.2.20 do_nmi()	62
6.21.2.21 do_overflow()	62
6.21.2.22 do_page_fault()	62
6.21.2.23 do_reserved()	62
6.21.2.24 do_segment_not_present()	62
6.21.2.25 do_stack_segment()	62
6.21.2.26 double_fault()	63
6.21.2.27 general_protection()	63
6.21.2.28 init_irq()	63
6.21.2.29 init_pic()	63
6.21.2.30 invalid_op()	63
6.21.2.31 invalid_tss()	63
6.21.2.32 isr0()	63
6.21.2.33 nmi()	64
6.21.2.34 overflow()	64
6.21.2.35 page_fault()	64
6.21.2.36 reserved()	64
6.21.2.37 rtc_isr()	64
6.21.2.38 segment_not_present()	64
6.21.2.39 stack_segment()	64
6.21.2.40 sys_call_isr()	64
6.21.3 Variable Documentation	65
6.21.3.1 idt_entries	65
6.22 /home/maximillian/Desktop/MAMA/kernel/core/kmain.c File Reference	65
6.22.1 Function Documentation	65
6.22.1.1 kmain()	65
6.23 /home/maximillian/Desktop/MAMA/kernel/core/serial.c File Reference	66
6.23.1 Macro Definition Documentation	66
6.23.1.1 DELETE	66
6.23.1.2 DOWN_ARROW	67

6.23.1.3 LEFT_ARROW	67
6.23.1.4 NO_ERROR	67
6.23.1.5 RIGHT_ARROW	67
6.23.1.6 UP_ARROW	67
6.23.2 Function Documentation	67
6.23.2.1 consume_special()	67
6.23.2.2 init_serial()	67
6.23.2.3 polling()	67
6.23.2.4 serial_print()	68
6.23.2.5 serial_println()	68
6.23.2.6 set_serial_in()	68
6.23.2.7 set_serial_out()	68
6.23.3 Variable Documentation	68
6.23.3.1 serial_port_in	68
6.23.3.2 serial_port_out	69
6.24 /home/maximillian/Desktop/MAMA/kernel/core/system.c File Reference	69
6.24.1 Function Documentation	69
6.24.1.1 klogv()	69
6.24.1.2 kpanic()	70
6.24.1.3 sys_call()	70
6.24.2 Variable Documentation	70
6.24.2.1 cop	70
6.24.2.2 global_context	70
6.24.2.3 params	70
6.24.2.4 priority_queue	71
6.25 /home/maximillian/Desktop/MAMA/kernel/core/tables.c File Reference	71
6.25.1 Function Documentation	71
6.25.1.1 gdt_init_entry()	71
6.25.1.2 idt_set_gate()	72
6.25.1.3 init_gdt()	72
6.25.1.4 init_idt()	72
6.25.1.5 write_gdt_ptr()	72
6.25.1.6 write_idt_ptr()	72
6.25.2 Variable Documentation	72
6.25.2.1 gdt_entries	72
6.25.2.2 gdt_ptr	73
6.25.2.3 idt_entries	73
6.25.2.4 idt_ptr	73
6.26 /home/maximillian/Desktop/MAMA/kernel/mem/heap.c File Reference	73
6.26.1 Function Documentation	73
6.26.1.1 _kmalloc()	74
6.26.1.2 alloc()	74

6.26.1.3	kmalloc()	74
6.26.1.4	make_heap()	74
6.26.2	Variable Documentation	74
6.26.2.1	__end	74
6.26.2.2	_end	74
6.26.2.3	curr_heap	75
6.26.2.4	end	75
6.26.2.5	kdir	75
6.26.2.6	kheap	75
6.26.2.7	phys_alloc_addr	75
6.27	/home/maximillian/Desktop/MAMA/kernel/mem/paging.c File Reference	75
6.27.1	Function Documentation	76
6.27.1.1	clear_bit()	76
6.27.1.2	find_free()	76
6.27.1.3	get_bit()	76
6.27.1.4	get_page()	76
6.27.1.5	init_paging()	76
6.27.1.6	load_page_dir()	77
6.27.1.7	new_frame()	77
6.27.1.8	set_bit()	77
6.27.2	Variable Documentation	77
6.27.2.1	cdir	77
6.27.2.2	frames	77
6.27.2.3	kdir	77
6.27.2.4	kheap	77
6.27.2.5	mem_size	78
6.27.2.6	nframes	78
6.27.2.7	page_size	78
6.27.2.8	phys_alloc_addr	78
6.28	/home/maximillian/Desktop/MAMA/lib/out.c File Reference	78
6.28.1	Function Documentation	78
6.28.1.1	print()	78
6.28.1.2	putc()	79
6.28.1.3	printf()	79
6.28.1.4	println()	79
6.28.1.5	read()	79
6.29	/home/maximillian/Desktop/MAMA/lib/out.h File Reference	79
6.29.1	Function Documentation	81
6.29.1.1	aliasHelp()	81
6.29.1.2	blockHelp()	81
6.29.1.3	clearHelp()	81
6.29.1.4	cmd_help()	81

6.29.1.5 createpcbHelp()	82
6.29.1.6 deletepcbHelp()	82
6.29.1.7 freealarmHelp()	82
6.29.1.8 getdateHelp()	82
6.29.1.9 gettimeHelp()	82
6.29.1.10 helpHelp()	82
6.29.1.11 helpList()	83
6.29.1.12 isemptyHelp()	83
6.29.1.13 loadr3Help()	83
6.29.1.14 print()	83
6.29.1.15 printf()	83
6.29.1.16 printf()	83
6.29.1.17 printf()	84
6.29.1.18 read()	84
6.29.1.19 resumeallHelp()	84
6.29.1.20 resumeHelp()	84
6.29.1.21 setalarmHelp()	84
6.29.1.22 setdateHelp()	84
6.29.1.23 setpriorityHelp()	85
6.29.1.24 settimeHelp()	85
6.29.1.25 showalarmsHelp()	85
6.29.1.26 showallocHelp()	85
6.29.1.27 showallpcbHelp()	85
6.29.1.28 showblockedpcbHelp()	85
6.29.1.29 showfreeHelp()	86
6.29.1.30 showpcbHelp()	86
6.29.1.31 showreadypcbHelp()	86
6.29.1.32 shutdownHelp()	86
6.29.1.33 suspendHelp()	86
6.29.1.34 unblockHelp()	86
6.29.1.35 versionHelp()	87
6.30 out.h	87
6.31 /home/maximillian/Desktop/MAMA/lib/string.c File Reference	88
6.31.1 Function Documentation	88
6.31.1.1 atoi()	88
6.31.1.2 isspace()	88
6.31.1.3 itoa()	88
6.31.1.4 memset()	89
6.31.1.5 strcat()	89
6.31.1.6 strcmp()	89
6.31.1.7 strcpy()	89
6.31.1.8 strlen()	89

6.31.1.9 strtok()	90
6.32 /home/maximillian/Desktop/MAMA/modules/mpx_supt.c File Reference	90
6.32.1 Function Documentation	90
6.32.1.1 idle()	90
6.32.1.2 mpx_init()	91
6.32.1.3 sys_alloc_mem()	91
6.32.1.4 sys_free_mem()	91
6.32.1.5 sys_req()	91
6.32.1.6 sys_set_free()	91
6.32.1.7 sys_set_malloc()	91
6.32.2 Variable Documentation	91
6.32.2.1 current_module	92
6.32.2.2 params	92
6.32.2.3 student_free	92
6.32.2.4 student_malloc	92
6.33 /home/maximillian/Desktop/MAMA/modules/mpx_supt.h File Reference	92
6.33.1 Macro Definition Documentation	93
6.33.1.1 COM_PORT	93
6.33.1.2 DEFAULT_DEVICE	93
6.33.1.3 EXIT	93
6.33.1.4 FALSE	93
6.33.1.5 IDLE	93
6.33.1.6 INVALID_BUFFER	94
6.33.1.7 INVALID_COUNT	94
6.33.1.8 INVALID_OPERATION	94
6.33.1.9 IO_MODULE	94
6.33.1.10 MEM_MODULE	94
6.33.1.11 MODULE_F	94
6.33.1.12 MODULE_R1	94
6.33.1.13 MODULE_R2	94
6.33.1.14 MODULE_R3	95
6.33.1.15 MODULE_R4	95
6.33.1.16 MODULE_R5	95
6.33.1.17 READ	95
6.33.1.18 TRUE	95
6.33.1.19 WRITE	95
6.33.2 Function Documentation	95
6.33.2.1 idle()	95
6.33.2.2 mpx_init()	96
6.33.2.3 sys_alloc_mem()	96
6.33.2.4 sys_free_mem()	96
6.33.2.5 sys_req()	96

6.33.2.6 <code>sys_set_free()</code>	96
6.33.2.7 <code>sys_set_malloc()</code>	96
6.34 <code>mpx_supt.h</code>	97
6.35 <code>/home/maximillian/Desktop/MAMA/README.md</code> File Reference	98
6.36 <code>/home/maximillian/Desktop/MAMA/serial_driver/driver.c</code> File Reference	98
6.36.1 Macro Definition Documentation	99
6.36.1.1 <code>BASE</code>	99
6.36.1.2 <code>DIVISOR_LATCH_HIGH_BYTE_REGISTER</code>	99
6.36.1.3 <code>DIVISOR_LATCH_LOW_BYTE_REGISTER</code>	99
6.36.1.4 <code>INTERRUPT_ENABLE_REGISTER</code>	99
6.36.1.5 <code>INTERRUPT_IDENTIFICATION_REGISTER</code>	99
6.36.1.6 <code>LINE_CONTROL_REGISTER</code>	99
6.36.1.7 <code>LINE_STATUS_REGISTER</code>	100
6.36.1.8 <code>MODEM_CONTROL_REGISTER</code>	100
6.36.1.9 <code>MODEM_STATUS_REGISTER</code>	100
6.36.1.10 <code>PIC_MASK</code>	100
6.36.1.11 <code>RING_BUFFER_SIZE</code>	100
6.36.1.12 <code>SCRATCH_REGISTER</code>	100
6.36.2 Typedef Documentation	100
6.36.2.1 <code>dcb_t</code>	100
6.36.3 Enumeration Type Documentation	100
6.36.3.1 <code>device_ready_state_t</code>	100
6.36.3.2 <code>device_status_t</code>	101
6.36.4 Function Documentation	101
6.36.4.1 <code>com_close()</code>	101
6.36.4.2 <code>com_open()</code>	101
6.36.4.3 <code>com_read()</code>	101
6.36.4.4 <code>com_write()</code>	102
6.36.5 Variable Documentation	102
6.36.5.1 <code>COM1_control_block</code>	102
6.37 <code>/home/maximillian/Desktop/MAMA/term/args.c</code> File Reference	102
6.37.1 Macro Definition Documentation	103
6.37.1.1 <code>MAX_PARSE_STACK_SIZE</code>	103
6.37.2 Function Documentation	103
6.37.2.1 <code>flag()</code>	103
6.37.2.2 <code>get_token()</code>	103
6.37.2.3 <code>named_arg()</code>	103
6.37.2.4 <code>next_unnamed_arg()</code>	103
6.37.2.5 <code>parse_args()</code>	104
6.37.2.6 <code>stack_empty()</code>	104
6.37.2.7 <code>stack_peek()</code>	104
6.37.2.8 <code>stack_pop()</code>	104

6.37.2.9 stack_push()	104
6.37.3 Variable Documentation	104
6.37.3.1 cur_state	104
6.37.3.2 last_state	104
6.37.3.3 parse_stack	105
6.37.3.4 stack_size	105
6.38 /home/maximillian/Desktop/MAMA/term/args.h File Reference	105
6.38.1 Typedef Documentation	105
6.38.1.1 parsed_args	105
6.38.2 Function Documentation	105
6.38.2.1 parse_args()	105
6.39 args.h	106
6.40 /home/maximillian/Desktop/MAMA/term/ascii/mama.c File Reference	106
6.40.1 Function Documentation	106
6.40.1.1 mama()	106
6.41 /home/maximillian/Desktop/MAMA/term/ascii/mama.h File Reference	106
6.41.1 Function Documentation	107
6.41.1.1 mama()	107
6.42 mama.h	107
6.43 /home/maximillian/Desktop/MAMA/term/cmds/argtest.c File Reference	107
6.43.1 Function Documentation	107
6.43.1.1 cmd_argtest()	107
6.44 /home/maximillian/Desktop/MAMA/term/cmds/echo.c File Reference	107
6.44.1 Function Documentation	108
6.44.1.1 cmd_echo()	108
6.45 /home/maximillian/Desktop/MAMA/help.c File Reference	108
6.45.1 Function Documentation	108
6.45.1.1 cmd_help()	108
6.45.1.2 getdateHelp()	109
6.45.1.3 gettimeHelp()	109
6.45.1.4 helpHelp()	109
6.45.1.5 helpList()	109
6.45.1.6 setdateHelp()	109
6.45.1.7 settimeHelp()	110
6.45.1.8 shutdownHelp()	110
6.45.1.9 versionOs()	110
6.46 /home/maximillian/Desktop/MAMA/term/cmds/help.c File Reference	110
6.46.1 Function Documentation	111
6.46.1.1 aliasHelp()	111
6.46.1.2 blockHelp()	112
6.46.1.3 clearHelp()	112
6.46.1.4 cmd_help()	112

6.46.1.5 createpcbHelp()	112
6.46.1.6 deletepcbHelp()	112
6.46.1.7 freealarmHelp()	113
6.46.1.8 getdateHelp()	113
6.46.1.9 gettimeHelp()	113
6.46.1.10 helpHelp()	113
6.46.1.11 helpList()	113
6.46.1.12 isemptyHelp()	113
6.46.1.13 loadr3Help()	114
6.46.1.14 resumeallHelp()	114
6.46.1.15 resumeHelp()	114
6.46.1.16 setalarmHelp()	114
6.46.1.17 setdateHelp()	114
6.46.1.18 setpriorityHelp()	114
6.46.1.19 settimeHelp()	115
6.46.1.20 showalarmsHelp()	115
6.46.1.21 showallocHelp()	115
6.46.1.22 showallpcbHelp()	115
6.46.1.23 showblockedpcbHelp()	115
6.46.1.24 showfreeHelp()	115
6.46.1.25 showpcbHelp()	116
6.46.1.26 showreadypcbHelp()	116
6.46.1.27 shutdownHelp()	116
6.46.1.28 suspendHelp()	116
6.46.1.29 unblockHelp()	116
6.46.1.30 versionHelp()	116
6.47 /home/maximillian/Desktop/MAMA/term/cmds/shutdown.c File Reference	117
6.47.1 Function Documentation	117
6.47.1.1 cmd_shutdown()	117
6.48 /home/maximillian/Desktop/MAMA/term/cmds/version.c File Reference	117
6.48.1 Function Documentation	118
6.48.1.1 cmd_version()	118
6.49 /home/maximillian/Desktop/MAMA/term/commands.h File Reference	118
6.50 commands.h	118
6.51 /home/maximillian/Desktop/MAMA/term/commhand.c File Reference	119
6.51.1 Typedef Documentation	119
6.51.1.1 cmd_func_t	120
6.51.1.2 cmd_mapping	120
6.51.2 Function Documentation	120
6.51.2.1 cmd_alias()	120
6.51.2.2 commhand()	120
6.51.2.3 extract_cmd_name()	120

6.51.2.4	fetch_cmd_mapping()	120
6.51.2.5	is_name_char()	120
6.51.3	Variable Documentation	121
6.51.3.1	cmd_mappings	121
6.51.3.2	priority_queue	121
6.52	/home/maximillian/Desktop/MAMA/term/commhand.h File Reference	121
6.52.1	Macro Definition Documentation	121
6.52.1.1	MAX_CMD_ARG_NAME_LEN	122
6.52.1.2	MAX_CMD_ARG_VALUE_LEN	122
6.52.1.3	MAX_CMD_COUNT	122
6.52.1.4	MAX_CMD_FLAG_COUNT	122
6.52.1.5	MAX_CMD_HIST_LEN	122
6.52.1.6	MAX_CMD_NAME_LEN	122
6.52.1.7	MAX_CMD_NAMED_ARG_COUNT	122
6.52.1.8	MAX_CMD_STRING_LEN	122
6.52.1.9	MAX_CMD_UNNAMED_ARG_COUNT	123
6.52.2	Function Documentation	123
6.52.2.1	commhand()	123
6.53	commhand.h	123
6.54	/home/maximillian/Desktop/MAMA/term/dispatch/context.c File Reference	123
6.54.1	Function Documentation	124
6.54.1.1	dispatcher()	124
6.54.1.2	loadr3()	124
6.54.1.3	yield()	125
6.55	/home/maximillian/Desktop/MAMA/term/dispatch/context.h File Reference	125
6.55.1	Typedef Documentation	125
6.55.1.1	context	125
6.55.2	Function Documentation	126
6.55.2.1	dispatcher()	126
6.55.2.2	loadr3()	127
6.55.2.3	yield()	127
6.56	context.h	127
6.57	/home/maximillian/Desktop/MAMA/term/dispatch/procsr3.c File Reference	128
6.57.1	Macro Definition Documentation	128
6.57.1.1	RC_1	128
6.57.1.2	RC_2	129
6.57.1.3	RC_3	129
6.57.1.4	RC_4	129
6.57.1.5	RC_5	129
6.57.2	Function Documentation	129
6.57.2.1	proc1()	129
6.57.2.2	proc2()	129

6.57.2.3 proc3()	129
6.57.2.4 proc4()	130
6.57.2.5 proc5()	130
6.57.3 Variable Documentation	130
6.57.3.1 er1	130
6.57.3.2 er2	130
6.57.3.3 er3	130
6.57.3.4 er4	130
6.57.3.5 er5	130
6.57.3.6 erSize	131
6.57.3.7 msg1	131
6.57.3.8 msg2	131
6.57.3.9 msg3	131
6.57.3.10 msg4	131
6.57.3.11 msg5	131
6.57.3.12 msgSize	131
6.58 /home/maximillian/Desktop/MAMA/term/dispatch/procsr3.h File Reference	132
6.58.1 Function Documentation	132
6.58.1.1 proc1()	132
6.58.1.2 proc2()	132
6.58.1.3 proc3()	132
6.58.1.4 proc4()	132
6.58.1.5 proc5()	132
6.59 procsr3.h	133
6.60 /home/maximillian/Desktop/MAMA/term/dnt/dnt.c File Reference	133
6.60.1 Function Documentation	134
6.60.1.1 BCDtol()	134
6.60.1.2 currentTime()	134
6.60.1.3 daysInMonth()	134
6.60.1.4 dispatchAlarm()	135
6.60.1.5 freeAlarm()	135
6.60.1.6 getdate()	135
6.60.1.7 gettime()	136
6.60.1.8 intToDayOfWeek()	136
6.60.1.9 intToMonth()	137
6.60.1.10 ltoBCD()	137
6.60.1.11 setAlarm()	137
6.60.1.12 setdate()	138
6.60.1.13 setDateInMemory()	138
6.60.1.14 settime()	139
6.60.1.15 setTimeInMemory()	139
6.60.1.16 showAlarms()	139

6.60.2 Variable Documentation	140
6.60.2.1 alarms	140
6.60.2.2 current_time	140
6.60.2.3 messages	140
6.61 /home/maximillian/Desktop/MAMA/term/dnt/dnt.h File Reference	140
6.61.1 Macro Definition Documentation	142
6.61.1.1 DAYS_IN_LEAP_YEAR	142
6.61.1.2 DAYS_IN_YEAR	142
6.61.1.3 EPOCH_FIRST_DAY_OF_WEEK_OF_YEAR	142
6.61.1.4 EPOCH_FIRST_DAY_OF_YEAR	142
6.61.1.5 EPOCH_FIRST_MONTH_OF_YEAR	143
6.61.1.6 EPOCH_YEAR	143
6.61.1.7 MAX_DAY	143
6.61.1.8 MAX_HOURS	143
6.61.1.9 MAX_MINUTES	143
6.61.1.10 MAX_MONTH	143
6.61.1.11 MAX_SECONDS	144
6.61.1.12 MAX_YEAR	144
6.61.1.13 MIN	144
6.61.1.14 MIN_DAY	144
6.61.1.15 MIN_MONTH	144
6.61.1.16 MIN_YEAR	144
6.61.2 Function Documentation	144
6.61.2.1 BCDtol()	144
6.61.2.2 currentTime()	145
6.61.2.3 daysInMonth()	145
6.61.2.4 dispatchAlarm()	145
6.61.2.5 freeAlarm()	146
6.61.2.6 getdate()	146
6.61.2.7 gettime()	146
6.61.2.8 intToDayOfWeek()	147
6.61.2.9 intToMonth()	147
6.61.2.10 ltoBCD()	148
6.61.2.11 setAlarm()	149
6.61.2.12 setdate()	149
6.61.2.13 setDateInMemory()	149
6.61.2.14 settime()	150
6.61.2.15 setTimeInMemory()	150
6.61.2.16 showAlarms()	151
6.62 dnt.h	151
6.63 /home/maximillian/Desktop/MAMA/term/history.c File Reference	152
6.63.1 Function Documentation	152

6.63.1.1 circular_next_index()	152
6.63.1.2 circular_prev_index()	153
6.63.1.3 hist_discard_last_frame()	153
6.63.1.4 hist_forward()	153
6.63.1.5 hist_next_frame()	154
6.63.1.6 hist_rewind()	154
6.63.1.7 write_hist_to_buf()	154
6.64 /home/maximillian/Desktop/MAMA/term/history.h File Reference	155
6.64.1 Function Documentation	155
6.64.1.1 hist_forward()	155
6.64.1.2 hist_next_frame()	156
6.64.1.3 hist_rewind()	156
6.65 history.h	156
6.66 /home/maximillian/Desktop/MAMA/term/memory_management/mm.c File Reference	156
6.66.1 Function Documentation	157
6.66.1.1 allocateMemory()	157
6.66.1.2 freeMemory()	158
6.66.1.3 initHeap()	158
6.66.1.4 insertAMCB()	158
6.66.1.5 insertFMCB()	158
6.66.1.6 isEmpty()	159
6.66.1.7 removeAMCB()	159
6.66.1.8 removeFMCB()	159
6.66.1.9 showAllocated()	159
6.66.1.10 showFree()	159
6.66.2 Variable Documentation	159
6.66.2.1 allocated	159
6.66.2.2 amcb	160
6.66.2.3 fmc_b	160
6.66.2.4 free	160
6.66.2.5 start_addr	160
6.67 /home/maximillian/Desktop/MAMA/term/memory_management/mm.h File Reference	160
6.67.1 Typedef Documentation	161
6.67.1.1 cmcb_s	161
6.67.1.2 mcb_queue_s	161
6.67.2 Enumeration Type Documentation	161
6.67.2.1 mcb_state_e	161
6.67.3 Function Documentation	162
6.67.3.1 allocateMemory()	162
6.67.3.2 freeMemory()	162
6.67.3.3 initHeap()	162
6.67.3.4 insertAMCB()	163

6.67.3.5 insertFMCB()	163
6.67.3.6 isEmpty()	163
6.67.3.7 removeAMCB()	163
6.67.3.8 removeFMCB()	164
6.67.3.9 showAllocated()	164
6.67.3.10 showFree()	164
6.68 mm.h	164
6.69 /home/maximillian/Desktop/MAMA/term/cmds/pcb.c File Reference	165
6.70 /home/maximillian/Desktop/MAMA/term/pcb/pcb.c File Reference	165
6.70.1 Function Documentation	166
6.70.1.1 allocatePCB()	166
6.70.1.2 blockPCB()	166
6.70.1.3 createPCB()	167
6.70.1.4 deletePCB()	167
6.70.1.5 findPCB()	168
6.70.1.6 freePCB()	168
6.70.1.7 initPCB()	168
6.70.1.8 insertPCB()	169
6.70.1.9 isSystemProcess()	170
6.70.1.10 removePCB()	170
6.70.1.11 resumeAll()	171
6.70.1.12 resumePCB()	171
6.70.1.13 setPriority()	171
6.70.1.14 setupPCB()	172
6.70.1.15 showAll()	172
6.70.1.16 showBlocked()	173
6.70.1.17 showPCB()	173
6.70.1.18 showReady()	173
6.70.1.19 suspendPCB()	173
6.70.1.20 unblockPCB()	174
6.70.2 Variable Documentation	174
6.70.2.1 f_queue	174
6.70.2.2 fifo_queue	174
6.70.2.3 p_queue	175
6.70.2.4 priority_queue	175
6.71 /home/maximillian/Desktop/MAMA/term/pcb/pcb.h File Reference	175
6.71.1 Macro Definition Documentation	176
6.71.1.1 MAX_NAME_SIZE	177
6.71.1.2 MAX_PRIORITY	177
6.71.1.3 MAX_STACK_SIZE	177
6.71.1.4 MIN_PRIORITY	177
6.71.2 Typedef Documentation	177

6.71.2.1 pcb_node_t	177
6.71.2.2 pcb_queue_t	177
6.71.3 Enumeration Type Documentation	177
6.71.3.1 p_protection_mode_t	177
6.71.3.2 p_state_t	178
6.71.3.3 pcb_queue_order_t	178
6.71.4 Function Documentation	178
6.71.4.1 allocatePCB()	178
6.71.4.2 blockPCB()	179
6.71.4.3 createPCB()	179
6.71.4.4 deletePCB()	179
6.71.4.5 findPCB()	181
6.71.4.6 freePCB()	181
6.71.4.7 initPCB()	182
6.71.4.8 insertPCB()	182
6.71.4.9 isSystemProcess()	182
6.71.4.10 removePCB()	183
6.71.4.11 resumeAll()	183
6.71.4.12 resumePCB()	183
6.71.4.13 setPriority()	184
6.71.4.14 setupPCB()	184
6.71.4.15 showAll()	185
6.71.4.16 showBlocked()	185
6.71.4.17 showPCB()	185
6.71.4.18 showReady()	185
6.71.4.19 suspendPCB()	186
6.71.4.20 unblockPCB()	186
6.72 pcb.h	187
6.73 /home/maximillian/Desktop/MAMA/term/syntax.c File Reference	188
6.73.1 Function Documentation	188
6.73.1.1 changes_state()	189
6.73.1.2 get_state()	189
6.74 /home/maximillian/Desktop/MAMA/term/syntax.h File Reference	189
6.74.1 Enumeration Type Documentation	189
6.74.1.1 SyntaxState	189
6.74.2 Function Documentation	190
6.74.2.1 changes_state()	190
6.74.2.2 get_state()	190
6.75 syntax.h	190
6.76 /home/maximillian/Desktop/MAMA/term/utils.c File Reference	190
6.76.1 Function Documentation	191
6.76.1.1 is_name_char()	191

6.76.1.2 skip_ws()	191
6.77 /home/maximillian/Desktop/MAMA/term/utls.h File Reference	191
6.77.1 Function Documentation	191
6.77.1.1 is_name_char()	192
6.77.1.2 skip_ws()	192
6.78 utls.h	192
6.79 /home/maximillian/Desktop/MAMA/term/cmds/clear.c File Reference	192
6.79.1 Function Documentation	193
6.79.1.1 cmd_clear()	193
6.80 /home/maximillian/Desktop/MAMA/term/visuals/clear.c File Reference	193
6.80.1 Function Documentation	193
6.80.1.1 display_clear()	193
6.81 /home/maximillian/Desktop/MAMA/term/visuals/clear.h File Reference	193
6.81.1 Function Documentation	193
6.81.1.1 display_clear()	193
6.82 clear.h	194
6.83 /home/maximillian/Desktop/MAMA/term/visuals/colorize.c File Reference	194
6.83.1 Macro Definition Documentation	194
6.83.1.1 START_SEQ	194
6.83.2 Enumeration Type Documentation	195
6.83.2.1 Color	195
6.83.3 Function Documentation	196
6.83.3.1 display_bg_color()	196
6.83.3.2 display_fg_color()	196
6.83.3.3 display_italicize()	197
6.83.3.4 display_reset()	197
6.83.3.5 print_color_code()	197
6.84 /home/maximillian/Desktop/MAMA/term/visuals/colorize.h File Reference	197
6.84.1 Enumeration Type Documentation	198
6.84.1.1 Color	198
6.84.2 Function Documentation	198
6.84.2.1 display_bg_color()	198
6.84.2.2 display_fg_color()	199
6.84.2.3 display_italicize()	199
6.84.2.4 display_reset()	199
6.85 colorize.h	199
6.86 /home/maximillian/Desktop/MAMA/term/visuals/cursor.c File Reference	200
6.86.1 Function Documentation	200
6.86.1.1 cursor_down()	200
6.86.1.2 cursor_left()	200
6.86.1.3 cursor_return()	201
6.86.1.4 cursor_right()	201

6.86.1.5 cursor_up()	201
6.87 /home/maximillian/Desktop/MAMA/term/visuals/cursor.h File Reference	201
6.87.1 Function Documentation	202
6.87.1.1 cursor_down()	202
6.87.1.2 cursor_left()	202
6.87.1.3 cursor_return()	202
6.87.1.4 cursor_right()	202
6.87.1.5 cursor_up()	203
6.88 cursor.h	203
6.89 /home/maximillian/Desktop/MAMA/term/visuals/hints.c File Reference	203
6.89.1 Function Documentation	203
6.89.1.1 hint_under_prompt()	204
6.90 /home/maximillian/Desktop/MAMA/term/visuals/hints.h File Reference	204
6.90.1 Function Documentation	204
6.90.1.1 hint_under_prompt()	204
6.91 hints.h	205
6.92 /home/maximillian/Desktop/MAMA/term/visuals/syntax_highlight.c File Reference	205
6.92.1 Function Documentation	206
6.92.1.1 color_for()	206
6.92.1.2 get_state_at()	206
6.92.1.3 switch_to()	206
6.92.1.4 syntax_disable_highlighting()	207
6.92.1.5 syntax_enable_highlighting()	207
6.92.1.6 syntax_handle_char()	207
6.92.1.7 syntax_init()	207
6.92.2 Variable Documentation	208
6.92.2.1 enabled	208
6.92.2.2 newest_switch	208
6.92.2.3 states	208
6.92.2.4 switch_indexes	208
6.93 /home/maximillian/Desktop/MAMA/term/visuals/syntax_highlight.h File Reference	208
6.93.1 Macro Definition Documentation	209
6.93.1.1 MAX_SYNTAX_SWITCHES	209
6.93.1.2 SYNTAX_COLOR_CMD_NAME	209
6.93.1.3 SYNTAX_COLOR_DEFAULT	209
6.93.1.4 SYNTAX_COLOR_DOUBLE_QUOTE_STRING	209
6.93.1.5 SYNTAX_COLOR_PARAM_NAME	209
6.93.1.6 SYNTAX_COLOR_PARAM_VALUE	210
6.93.1.7 SYNTAX_COLOR_SINGLE_QUOTE_STRING	210
6.93.2 Function Documentation	210
6.93.2.1 syntax_disable_highlighting()	210
6.93.2.2 syntax_enable_highlighting()	210

6.93.2.3 syntax_handle_char()	210
6.93.2.4 syntax_init()	211
6.94 syntax_highlight.h	211
6.95 /home/maximillian/Desktop/MAMA/WhoDidWhat.md File Reference	211
Index	213

Chapter 1

MAMA

Check out the [who did what markdown page](#) for a list of contributions from each user during each milestone.

Use the [user manual](#) to find information on available commands.

Use the [programmer manual](#) to find information on individual functions, structs, constants, and other code documentation.

Chapter 2

Who did what table

Update with your contributions every module

	R1	R2	R3	R4	R5	R6
Austin Williams	term/visuals/colorize.c term/visuals/cursor.c term/visuals/syntax_highlight.c term/history.c term/syntax.c term/args.c polling() commhand()				cmd_alias() showFree()	
Maximillian Campbell	polling() commhand() gettime() settime() getdate() setdate() cmd_help() cmd_shutdown() itoa() Setting up doxygen Help pages	allocatePCB() FreePCB() SetupPCB() CreatePCB() UnblockPCB() setPriority ↔ PCB() MAMA.pdf HTML Docs Help Pages	irq.s yield() loadr3() sys_call() dispatcher() Documentation Help Pages	createAlarm() freeAlarm() showAlarms() dispatchAlarm() Documentaion Help Pages	initHeap() allocateMemory() freeMemory() Documentation	

	R1	R2	R3	R4	R5	R6
Mohammad Alenezi	print_color_code() display_fg_color() display_bg_color() display_reset() display_italicize() print_color_code() cursor_left() cursor_right() cursor_down() cursor_up() cursor_return()	FindPCB() Show block processes DeletePCB() BlockPCB()			Show↔ Allocated() display_clear()	
Abdullah Alqallaf	cmd_version() VersionOs() Some of Help.c comments for Manual	Set Priority() Show All Pro- cesses() Show ready processes() Show Block Processes() Show PCB()			isEmpty() freeMemory()	

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

cmcb_s	Complete Memory Control Block (CMBC)	9
cmd_mapping		11
context	Context of the currently operating process	11
date_time		14
dcb_t		16
footer		18
gdt_descriptor_struct		18
gdt_entry_struct		19
header		20
heap		21
idt_entry_struct		21
idt_struct		23
index_entry		23
index_table		24
mcb_queue_s	"Master" controller of the MCB queue	25
page_dir		25
page_entry		26
page_table		27
param		28
parsed_args		29
pcb_node_t	Individual PCB nodes. Each PCB is associated with one node	30
pcb_queue	"Master" controller of the PCB queue	31
pcb_t	Process Control Block Structure	33

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

/home/maximillian/Desktop/MAMA/help.c	108
/home/maximillian/Desktop/MAMA/include/string.h	51
/home/maximillian/Desktop/MAMA/include/system.h	54
/home/maximillian/Desktop/MAMA/include/core/asm.h	35
/home/maximillian/Desktop/MAMA/include/core/comhand.h	35
/home/maximillian/Desktop/MAMA/include/core/interrupts.h	36
/home/maximillian/Desktop/MAMA/include/core/io.h	37
/home/maximillian/Desktop/MAMA/include/core/serial.h	38
/home/maximillian/Desktop/MAMA/include/core/tables.h	41
/home/maximillian/Desktop/MAMA/include/mem/heap.h	44
/home/maximillian/Desktop/MAMA/include/mem/paging.h	48
/home/maximillian/Desktop/MAMA/kernel/core/interrupts.c	57
/home/maximillian/Desktop/MAMA/kernel/core/kmain.c	65
/home/maximillian/Desktop/MAMA/kernel/core/serial.c	66
/home/maximillian/Desktop/MAMA/kernel/core/system.c	69
/home/maximillian/Desktop/MAMA/kernel/core/tables.c	71
/home/maximillian/Desktop/MAMA/kernel/mem/heap.c	73
/home/maximillian/Desktop/MAMA/kernel/mem/paging.c	75
/home/maximillian/Desktop/MAMA/lib/out.c	78
/home/maximillian/Desktop/MAMA/lib/out.h	79
/home/maximillian/Desktop/MAMA/lib/string.c	88
/home/maximillian/Desktop/MAMA/modules/mpx_supt.c	90
/home/maximillian/Desktop/MAMA/modules/mpx_supt.h	92
/home/maximillian/Desktop/MAMA/serial_driver/driver.c	98
/home/maximillian/Desktop/MAMA/term/args.c	102
/home/maximillian/Desktop/MAMA/term/args.h	105
/home/maximillian/Desktop/MAMA/term/commands.h	118
/home/maximillian/Desktop/MAMA/term/commhand.c	119
/home/maximillian/Desktop/MAMA/term/commhand.h	121
/home/maximillian/Desktop/MAMA/term/history.c	152
/home/maximillian/Desktop/MAMA/term/history.h	155
/home/maximillian/Desktop/MAMA/term/syntax.c	188
/home/maximillian/Desktop/MAMA/term/syntax.h	189
/home/maximillian/Desktop/MAMA/term/utils.c	190
/home/maximillian/Desktop/MAMA/term/utils.h	191

/home/maximillian/Desktop/MAMA/term/ascii/mama.c	106
/home/maximillian/Desktop/MAMA/term/ascii/mama.h	106
/home/maximillian/Desktop/MAMA/term/cmds/argtest.c	107
/home/maximillian/Desktop/MAMA/term/cmds/clear.c	192
/home/maximillian/Desktop/MAMA/term/cmds/echo.c	107
/home/maximillian/Desktop/MAMA/term/cmds/help.c	110
/home/maximillian/Desktop/MAMA/term/cmds/pcb.c	165
/home/maximillian/Desktop/MAMA/term/cmds/shutdown.c	117
/home/maximillian/Desktop/MAMA/term/cmds/version.c	117
/home/maximillian/Desktop/MAMA/term/dispatch/context.c	123
/home/maximillian/Desktop/MAMA/term/dispatch/context.h	125
/home/maximillian/Desktop/MAMA/term/dispatch/procsr3.c	128
/home/maximillian/Desktop/MAMA/term/dispatch/procsr3.h	132
/home/maximillian/Desktop/MAMA/term/dnt/dnt.c	133
/home/maximillian/Desktop/MAMA/term/dnt/dnt.h	140
/home/maximillian/Desktop/MAMA/term/memory_management/mm.c	156
/home/maximillian/Desktop/MAMA/term/memory_management/mm.h	160
/home/maximillian/Desktop/MAMA/term/pcb/pcb.c	165
/home/maximillian/Desktop/MAMA/term/pcb/pcb.h	175
/home/maximillian/Desktop/MAMA/term/visuals/clear.c	193
/home/maximillian/Desktop/MAMA/term/visuals/clear.h	193
/home/maximillian/Desktop/MAMA/term/visuals/colorize.c	194
/home/maximillian/Desktop/MAMA/term/visuals/colorize.h	197
/home/maximillian/Desktop/MAMA/term/visuals/cursor.c	200
/home/maximillian/Desktop/MAMA/term/visuals/cursor.h	201
/home/maximillian/Desktop/MAMA/term/visuals/hints.c	203
/home/maximillian/Desktop/MAMA/term/visuals/hints.h	204
/home/maximillian/Desktop/MAMA/term/visuals/syntax_highlight.c	205
/home/maximillian/Desktop/MAMA/term/visuals/syntax_highlight.h	208

Chapter 5

Class Documentation

5.1 cmcb_s Struct Reference

Complete Memory Control Block (CMBC)

```
#include <mm.h>
```

Public Attributes

- [mcb_state_e](#) type
The type of the CMCB
- [u32int](#) addr
Beginning address of the CMCB.
- [u32int](#) size
Size of the CMCB.
- char [name](#) [32]
Name of CMCB.
- struct [cmcb_s](#) * [next](#)
Next CMCB.
- struct [cmcb_s](#) * [prev](#)
Previous CMCB.

5.1.1 Detailed Description

Complete Memory Control Block (CMBC)

5.1.2 Member Data Documentation

5.1.2.1 addr

```
u32int cmcb_s::addr
```

Beginning address of the CMCB.

5.1.2.2 name

```
char cmcb_s::name[32]
```

Name of CMCB.

5.1.2.3 next

```
struct cmcb_s* cmcb_s::next
```

Next CMCB.

5.1.2.4 prev

```
struct cmcb_s* cmcb_s::prev
```

Previous CMCB.

5.1.2.5 size

```
u32int cmcb_s::size
```

Size of the CMCB.

5.1.2.6 type

```
mcb_state_e cmcb_s::type
```

The type of the CMCB

The documentation for this struct was generated from the following file:

- /home/maximillian/Desktop/MAMA/term/memory_management/mm.h

5.2 cmd_mapping Struct Reference

Public Attributes

- char * [cmd_name](#)
- [cmd_func_t](#) cmd_handler
- char * [default_args](#)

5.2.1 Member Data Documentation

5.2.1.1 cmd_handler

[cmd_func_t](#) cmd_mapping::cmd_handler

5.2.1.2 cmd_name

char* cmd_mapping::cmd_name

5.2.1.3 default_args

char* cmd_mapping::default_args

The documentation for this struct was generated from the following file:

- /home/maximillian/Desktop/MAMA/term/[commhand.c](#)

5.3 context Struct Reference

Context of the currently operating process.

```
#include <context.h>
```

Public Attributes

- [u32int gs](#)

Segment registers.

- [u32int fs](#)
- [u32int es](#)
- [u32int ds](#)
- [u32int edi](#)

General purpose registers.

- [u32int esi](#)
- [u32int ebp](#)
- [u32int esp](#)
- [u32int ebx](#)
- [u32int edx](#)
- [u32int ecx](#)
- [u32int eax](#)
- [u32int eip](#)
- [u32int cs](#)
- [u32int eflags](#)

5.3.1 Detailed Description

Context of the currently operating process.

5.3.2 Member Data Documentation

5.3.2.1 cs

[u32int](#) context::cs

5.3.2.2 ds

[u32int](#) context::ds

5.3.2.3 eax

[u32int](#) context::eax

5.3.2.4 ebp

```
u32int context::ebp
```

5.3.2.5 ebx

```
u32int context::ebx
```

5.3.2.6 ecx

```
u32int context::ecx
```

5.3.2.7 edi

```
u32int context::edi
```

General purpose registers.

5.3.2.8 edx

```
u32int context::edx
```

5.3.2.9 eflags

```
u32int context::eflags
```

5.3.2.10 eip

```
u32int context::eip
```

5.3.2.11 es

```
u32int context::es
```

5.3.2.12 esi

```
u32int context::esi
```

5.3.2.13 esp

```
u32int context::esp
```

5.3.2.14 fs

```
u32int context::fs
```

5.3.2.15 gs

```
u32int context::gs
```

Segment registers.

The documentation for this struct was generated from the following file:

- </home/maximillian/Desktop/MAMA/term/dispatch/context.h>

5.4 date_time Struct Reference

```
#include <system.h>
```

Public Attributes

- int [sec](#)
- int [min](#)
- int [hour](#)
- int [day_w](#)
- int [day_m](#)
- int [day_y](#)
- int [mon](#)
- int [year](#)

5.4.1 Member Data Documentation

5.4.1.1 day_m

```
int date_time::day_m
```

5.4.1.2 day_w

```
int date_time::day_w
```

5.4.1.3 day_y

```
int date_time::day_y
```

5.4.1.4 hour

```
int date_time::hour
```

5.4.1.5 min

```
int date_time::min
```

5.4.1.6 mon

```
int date_time::mon
```

5.4.1.7 sec

```
int date_time::sec
```

5.4.1.8 year

```
int date_time::year
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/system.h](#)

5.5 dcb_t Struct Reference

Public Attributes

- [int * eflag_p](#)
- [device_ready_state_t ready_state](#)
- [device_status_t oper_status](#)
- [char * user_read_buf](#)
- [int * user_read_count](#)
- [char * user_write_buf](#)
- [int * user_write_count](#)
- [char ring_buffer \[RING_BUFFER_SIZE\]](#)
- [int ring_buffer_head = 0](#)
- [int ring_buffer_tail = 0](#)

5.5.1 Member Data Documentation

5.5.1.1 eflag_p

```
int* dcb_t::eflag_p
```

5.5.1.2 oper_status

```
device_status_t dcb_t::oper_status
```

5.5.1.3 ready_state

```
device_ready_state_t dcb_t::ready_state
```

5.5.1.4 ring_buffer

```
char dcb_t::ring_buffer[RING_BUFFER_SIZE]
```

5.5.1.5 ring_buffer_head

```
int dcb_t::ring_buffer_head = 0
```

5.5.1.6 ring_buffer_tail

```
int dcb_t::ring_buffer_tail = 0
```

5.5.1.7 user_read_buf

```
char* dcb_t::user_read_buf
```

5.5.1.8 user_read_count

```
int* dcb_t::user_read_count
```

5.5.1.9 user_write_buf

```
char* dcb_t::user_write_buf
```

5.5.1.10 user_write_count

```
int* dcb_t::user_write_count
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/serial_driver/driver.c](#)

5.6 footer Struct Reference

```
#include <heap.h>
```

Public Attributes

- [header head](#)

5.6.1 Member Data Documentation

5.6.1.1 head

```
header footer::head
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/mem/heap.h](#)

5.7 gdt_descriptor_struct Struct Reference

```
#include <tables.h>
```

Public Attributes

- [u16int limit](#)
- [u32int base](#)

5.7.1 Member Data Documentation

5.7.1.1 base

```
u32int gdt_descriptor_struct::base
```

5.7.1.2 limit

```
u16int gdt_descriptor_struct::limit
```

The documentation for this struct was generated from the following file:

- /home/maximillian/Desktop/MAMA/include/core/[tables.h](#)

5.8 gdt_entry_struct Struct Reference

```
#include <tables.h>
```

Public Attributes

- [u16int limit_low](#)
- [u16int base_low](#)
- [u8int base_mid](#)
- [u8int access](#)
- [u8int flags](#)
- [u8int base_high](#)

5.8.1 Member Data Documentation

5.8.1.1 access

```
u8int gdt_entry_struct::access
```

5.8.1.2 base_high

```
u8int gdt_entry_struct::base_high
```

5.8.1.3 base_low

```
u16int gdt_entry_struct::base_low
```

5.8.1.4 base_mid

```
uint gdt_entry_struct::base_mid
```

5.8.1.5 flags

```
uint gdt_entry_struct::flags
```

5.8.1.6 limit_low

```
uint gdt_entry_struct::limit_low
```

The documentation for this struct was generated from the following file:

- </home/maximillian/Desktop/MAMA/include/core/tables.h>

5.9 header Struct Reference

```
#include <heap.h>
```

Public Attributes

- int [size](#)
- int [index_id](#)

5.9.1 Member Data Documentation

5.9.1.1 index_id

```
int header::index_id
```

5.9.1.2 size

```
int header::size
```

The documentation for this struct was generated from the following file:

- </home/maximillian/Desktop/MAMA/include/mem/heap.h>

5.10 heap Struct Reference

```
#include <heap.h>
```

Public Attributes

- [index_table](#) index
- [u32int](#) base
- [u32int](#) max_size
- [u32int](#) min_size

5.10.1 Member Data Documentation

5.10.1.1 base

```
u32int heap::base
```

5.10.1.2 index

```
index\_table heap::index
```

5.10.1.3 max_size

```
u32int heap::max_size
```

5.10.1.4 min_size

```
u32int heap::min_size
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/mem/heap.h](#)

5.11 idt_entry_struct Struct Reference

```
#include <tables.h>
```

Public Attributes

- [u16int base_low](#)
- [u16int sselect](#)
- [u8int zero](#)
- [u8int flags](#)
- [u16int base_high](#)

5.11.1 Member Data Documentation

5.11.1.1 base_high

`u16int idt_entry_struct::base_high`

5.11.1.2 base_low

`u16int idt_entry_struct::base_low`

5.11.1.3 flags

`u8int idt_entry_struct::flags`

5.11.1.4 sselect

`u16int idt_entry_struct::sselect`

5.11.1.5 zero

`u8int idt_entry_struct::zero`

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/core/tables.h](#)

5.12 idt_struct Struct Reference

```
#include <tables.h>
```

Public Attributes

- [u16int limit](#)
- [u32int base](#)

5.12.1 Member Data Documentation

5.12.1.1 base

```
u32int idt_struct::base
```

5.12.1.2 limit

```
u16int idt_struct::limit
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/core/tables.h](#)

5.13 index_entry Struct Reference

```
#include <heap.h>
```

Public Attributes

- [int size](#)
- [int empty](#)
- [u32int block](#)

5.13.1 Member Data Documentation

5.13.1.1 block

```
u32int index_entry::block
```

5.13.1.2 empty

```
int index_entry::empty
```

5.13.1.3 size

```
int index_entry::size
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/mem/heap.h](#)

5.14 index_table Struct Reference

```
#include <heap.h>
```

Public Attributes

- [index_entry](#) table [TABLE_SIZE]
- [int](#) id

5.14.1 Member Data Documentation

5.14.1.1 id

```
int index_table::id
```

5.14.1.2 table

```
index\_entry index_table::table[TABLE_SIZE]
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/mem/heap.h](#)

5.15 mcb_queue_s Struct Reference

"Master" controller of the MCB queue

```
#include <mm.h>
```

Public Attributes

- [cmcb_s * mcbq_head](#)
Head of the MCB queue.
- [mcb_state_e mcb_queue_type](#)
Queue order of the Master controller.

5.15.1 Detailed Description

"Master" controller of the MCB queue

5.15.2 Member Data Documentation

5.15.2.1 mcb_queue_type

```
mcb\_state\_e mcb_queue_s::mcb_queue_type
```

Queue order of the Master controller.

5.15.2.2 mcbq_head

```
cmcb\_s\* mcb_queue_s::mcbq_head
```

Head of the MCB queue.

The documentation for this struct was generated from the following file:

- /home/maximillian/Desktop/MAMA/term/memory_management/[mm.h](#)

5.16 page_dir Struct Reference

```
#include <paging.h>
```

Public Attributes

- [page_table](#) * [tables](#) [1024]
- [u32int](#) [tables_phys](#) [1024]

5.16.1 Member Data Documentation

5.16.1.1 tables

```
page\_table* page\_dir::tables[1024]
```

5.16.1.2 tables_phys

```
u32int page\_dir::tables\_phys[1024]
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/mem/paging.h](#)

5.17 page_entry Struct Reference

```
#include <paging.h>
```

Public Attributes

- [u32int](#) [present](#): 1
- [u32int](#) [writeable](#): 1
- [u32int](#) [usermode](#): 1
- [u32int](#) [accessed](#): 1
- [u32int](#) [dirty](#): 1
- [u32int](#) [reserved](#): 7
- [u32int](#) [frameaddr](#): 20

5.17.1 Member Data Documentation

5.17.1.1 accessed

```
u32int page\_entry::accessed
```

5.17.1.2 dirty

```
u32int page_entry::dirty
```

5.17.1.3 frameaddr

```
u32int page_entry::frameaddr
```

5.17.1.4 present

```
u32int page_entry::present
```

5.17.1.5 reserved

```
u32int page_entry::reserved
```

5.17.1.6 usermode

```
u32int page_entry::usermode
```

5.17.1.7 writeable

```
u32int page_entry::writeable
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/include/mem/paging.h](#)

5.18 page_table Struct Reference

```
#include <paging.h>
```

Public Attributes

- [page_entry pages](#) [1024]

5.18.1 Member Data Documentation

5.18.1.1 pages

`page_entry page_table::pages[1024]`

The documentation for this struct was generated from the following file:

- `/home/maximillian/Desktop/MAMA/include/mem/paging.h`

5.19 param Struct Reference

```
#include <mpx_supt.h>
```

Public Attributes

- `int op_code`
- `int device_id`
- `char * buffer_ptr`
- `int * count_ptr`

5.19.1 Member Data Documentation

5.19.1.1 buffer_ptr

`char* param::buffer_ptr`

5.19.1.2 count_ptr

`int* param::count_ptr`

5.19.1.3 device_id

`int param::device_id`

5.19.1.4 op_code

```
int param::op_code
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/modules/mpx_supt.h](#)

5.20 parsed_args Struct Reference

```
#include <args.h>
```

Public Attributes

- int [flag_count](#)
- int [named_arg_count](#)
- int [unnamed_arg_count](#)
- int [unnamed_args_used_so_far](#)
- char [flags](#) [[MAX_CMD_FLAG_COUNT](#)][[MAX_CMD_ARG_NAME_LEN](#)+1]
- char [named_arg_names](#) [[MAX_CMD_NAMED_ARG_COUNT](#)][[MAX_CMD_ARG_NAME_LEN](#)+1]
- char [named_arg_values](#) [[MAX_CMD_NAMED_ARG_COUNT](#)][[MAX_CMD_ARG_VALUE_LEN](#)+1]
- char [unnamed_args](#) [[MAX_CMD_UNNAMED_ARG_COUNT](#)][[MAX_CMD_ARG_VALUE_LEN](#)+1]

5.20.1 Member Data Documentation

5.20.1.1 flag_count

```
int parsed_args::flag_count
```

5.20.1.2 flags

```
char parsed_args::flags[MAX\_CMD\_FLAG\_COUNT][MAX\_CMD\_ARG\_NAME\_LEN+1]
```

5.20.1.3 named_arg_count

```
int parsed_args::named_arg_count
```

5.20.1.4 named_arg_names

```
char parsed_args::named_arg_names[MAX_CMD_NAMED_ARG_COUNT][MAX_CMD_ARG_NAME_LEN+1]
```

5.20.1.5 named_arg_values

```
char parsed_args::named_arg_values[MAX_CMD_NAMED_ARG_COUNT][MAX_CMD_ARG_VALUE_LEN+1]
```

5.20.1.6 unnamed_arg_count

```
int parsed_args::unnamed_arg_count
```

5.20.1.7 unnamed_args

```
char parsed_args::unnamed_args[MAX_CMD_UNNAMED_ARG_COUNT][MAX_CMD_ARG_VALUE_LEN+1]
```

5.20.1.8 unnamed_args_used_so_far

```
int parsed_args::unnamed_args_used_so_far
```

The documentation for this struct was generated from the following file:

- [/home/maximillian/Desktop/MAMA/term/args.h](#)

5.21 pcb_node_t Struct Reference

Individual PCB nodes. Each PCB is associated with one node.

```
#include <pcb.h>
```

Public Attributes

- struct [pcb_node_t](#) * [pcbn_next_pcb](#)
Pointer to the Next PCB.
- struct [pcb_node_t](#) * [pcbn_prev_pcb](#)
Pointer to the Previous PCB.
- [pcb_t](#) * [pcb](#)
Pointer to PCB.

5.21.1 Detailed Description

Individual PCB nodes. Each PCB is associated with one node.

5.21.2 Member Data Documentation

5.21.2.1 pcb

```
pcb_t* pcb_node_t::pcb
```

Pointer to PCB.

5.21.2.2 pcbn_next_pcb

```
struct pcb_node_t* pcb_node_t::pcbn_next_pcb
```

Pointer to the Next PCB.

5.21.2.3 pcbn_prev_pcb

```
struct pcb_node_t* pcb_node_t::pcbn_prev_pcb
```

Pointer to the Previous PCB.

The documentation for this struct was generated from the following file:

- </home/maximillian/Desktop/MAMA/term/pcb/pcb.h>

5.22 pcb_queue Struct Reference

"Master" controller of the PCB queue

```
#include <pcb.h>
```

Public Attributes

- `int pcbq_count`
Number of PCB's currently in the queue.
- `pcb_node_t * pcbq_head`
Head of the PCB queue.
- `pcb_node_t * pcbq_tail`
Tail of the PCB queue.
- `pcb_queue_order_t queue_order`
Queue order of the Master controller.

5.22.1 Detailed Description

"Master" controller of the PCB queue

5.22.2 Member Data Documentation

5.22.2.1 `pcbq_count`

```
int pcb_queue::pcbq_count
```

Number of PCB's currently in the queue.

5.22.2.2 `pcbq_head`

```
pcb_node_t* pcb_queue::pcbq_head
```

Head of the PCB queue.

5.22.2.3 `pcbq_tail`

```
pcb_node_t* pcb_queue::pcbq_tail
```

Tail of the PCB queue.

5.22.2.4 queue_order

```
pcb_queue_order_t pcb_queue::queue_order
```

Queue order of the Master controller.

The documentation for this struct was generated from the following file:

- /home/maximillian/Desktop/MAMA/term/pcb/pcb.h

5.23 pcb_t Struct Reference

Process Control Block Structure.

```
#include <pcb.h>
```

Public Attributes

- char [pcb_name](#) [32]
PCB Name.
- int [pcb_process_class](#)
Process Class.
- int [pcb_priority](#)
Priority of PCB.
- [p_state_t](#) [pcb_process_state](#)
State of the PCB.
- [p_protection_mode_t](#) [pcb_protection_mode](#)
- unsigned char * [pcb_stack_top](#)
Top of the Stack. Set equal to the stack base + size of the stack.
- unsigned char * [pcb_stack_bottom](#)
Beginning of the Stack.

5.23.1 Detailed Description

Process Control Block Structure.

5.23.2 Member Data Documentation

5.23.2.1 pcb_name

```
char pcb_t::pcb_name[32]
```

PCB Name.

5.23.2.2 `pcb_priority`

```
int pcb_t::pcb_priority
```

Priority of PCB.

5.23.2.3 `pcb_process_class`

```
int pcb_t::pcb_process_class
```

Process Class.

5.23.2.4 `pcb_process_state`

```
p_state_t pcb_t::pcb_process_state
```

State of the PCB.

5.23.2.5 `pcb_protection_mode`

```
p_protection_mode_t pcb_t::pcb_protection_mode
```

5.23.2.6 `pcb_stack_bottom`

```
unsigned char* pcb_t::pcb_stack_bottom
```

Beginning of the Stack.

5.23.2.7 `pcb_stack_top`

```
unsigned char* pcb_t::pcb_stack_top
```

Top of the Stack. Set equal to the stack base + size of the stack.

The documentation for this struct was generated from the following file:

- `/home/maximillian/Desktop/MAMA/term/pcb/pcb.h`

Chapter 6

File Documentation

6.1 /home/maximillian/Desktop/MAMA/include/core/asm.h File Reference

```
#include <system.h>
#include <tables.h>
```

6.2 asm.h

[Go to the documentation of this file.](#)

```
1 #ifndef _ASM_H
2 #define _ASM_H
3
4 #include <system.h>
5 #include <tables.h>
6
7 #endif
```

6.3 /home/maximillian/Desktop/MAMA/include/core/comhand.h File Reference

Functions

- int [comhand](#) ()

6.3.1 Function Documentation

6.3.1.1 comhand()

```
int comhand ( )
```

6.4 comhand.h

[Go to the documentation of this file.](#)

```
1 #ifndef _COMHAND_H
2 #define _COMHAND_H
3
4 int comhand();
5
6 #endif
```

6.5 /home/maximillian/Desktop/MAMA/include/core/interrupts.h File Reference

Functions

- void [init_irq](#) (void)
- void [init_pic](#) (void)

6.5.1 Function Documentation

6.5.1.1 init_irq()

```
void init_irq (
    void )
```

6.5.1.2 init_pic()

```
void init_pic (
    void )
```

6.6 interrupts.h

[Go to the documentation of this file.](#)

```
1 #ifndef _INTERRUPTS_H
2 #define _INTERRUPTS_H
3
4 /*
5  Procedure...: init_irq
6  Description...: Installs the initial interrupt handlers for
7                  the first 32 irq lines. Most do a panic for now.
8 */
9 void init_irq(void);
10
11 /*
12  Procedure...: init_pic
13  Description...: Initializes the programmable interrupt controllers
14                  and performs the necessary remapping of IRQs. Leaves interrupts
15                  turned off.
16 */
17 void init_pic(void);
18
19 #endif
```

6.7 /home/maximillian/Desktop/MAMA/include/core/io.h File Reference

Macros

- `#define outb(port, data) asm volatile ("outb %%al,%%dx" : : "a" (data), "d" (port))`
- `#define inb(port)`

6.7.1 Macro Definition Documentation

6.7.1.1 inb

```
#define inb(  
    port )
```

Value:

```
{  
    unsigned char r;  
    asm volatile ("inb %%dx,%%al": "=a" (r): "d" (port)); \  
    r;  
}
```

6.7.1.2 outb

```
#define outb(  
    port,  
    data ) asm volatile ("outb %%al,%%dx" : : "a" (data), "d" (port))
```

6.8 io.h

[Go to the documentation of this file.](#)

```
1 #ifndef _IO_H
2 #define _IO_H
3
4 /*
5  Procedure...: outb
6  Description...: Write a byte of data to a port.
7 */
8 #define outb(port, data) \  
9  asm volatile ("outb %%al,%%dx" : : "a" (data), "d" (port))
10
11 /*
12  Procedure...: inb
13  Description...: Read a byte of data from a port.
14 */
15 #define inb(port) ({  
16     unsigned char r;  
17     asm volatile ("inb %%dx,%%al": "=a" (r): "d" (port)); \  
18     r;  
19 })
20
21 #endif
```

6.9 /home/maximillian/Desktop/MAMA/include/core/serial.h File Reference

Macros

- #define [COM1](#) 0x3f8
- #define [COM2](#) 0x2f8
- #define [COM3](#) 0x3e8
- #define [COM4](#) 0x2e8

Functions

- int [init_serial](#) (int device)
- int [serial_println](#) (const char *msg)
- int [serial_print](#) (const char *msg)
- int [set_serial_out](#) (int device)
- int [set_serial_in](#) (int device)
- int * [polling](#) (char *buffer, int *count)
Serially poll characters from command line.

6.9.1 Macro Definition Documentation

6.9.1.1 COM1

```
#define COM1 0x3f8
```

6.9.1.2 COM2

```
#define COM2 0x2f8
```

6.9.1.3 COM3

```
#define COM3 0x3e8
```

6.9.1.4 COM4

```
#define COM4 0x2e8
```


6.9.2 Function Documentation

6.9.2.1 init_serial()

```
int init_serial (
    int device )
```

6.9.2.2 polling()

```
int * polling (
    char * buffer,
    int * count )
```

Serially poll characters from command line.

Polls input from keyboard and interprets each character individually as it is entered from the keyboard.

Parameters

<i>buffer</i>	Space allocated for single line on the command line
<i>count</i>	Size of the space allocated

Returns

Returns 0 upon success, -1 upon error

6.9.2.3 serial_print()

```
int serial_print (
    const char * msg )
```

6.9.2.4 serial_println()

```
int serial_println (
    const char * msg )
```

6.9.2.5 set_serial_in()

```
int set_serial_in (
    int device )
```

6.9.2.6 set_serial_out()

```
int set_serial_out (
    int device )
```

6.10 serial.h

[Go to the documentation of this file.](#)

```
1 #ifndef _SERIAL_H
2 #define _SERIAL_H
3
4 #define COM1 0x3f8
5 #define COM2 0x2f8
6 #define COM3 0x3e8
7 #define COM4 0x2e8
8
9 /*
10  Procedure...: init_serial
11  Description...: Initializes devices for user interaction, logging, ...
12 */
13 int init_serial(int device);
14
15 /*
16  Procedure...: serial_println
17  Description...: Writes a message to the active serial output device.
18                  Appends a newline character.
19 */
20 int serial_println(const char *msg);
21
22 /*
23  Procedure...: serial_print
24  Description...: Writes a message to the active serial output device.
25 */
26 int serial_print(const char *msg);
27
28 /*
29  Procedure...: set_serial_out
30  Description...: Sets serial_port_out to the given device address.
31                  All serial output, such as that from serial_println, will be
32                  directed to this device.
33 */
34 int set_serial_out(int device);
35
36 /*
37  Procedure...: set_serial_in
38  Description...: Sets serial_port_in to the given device address.
39                  All serial input, such as console input via a virtual machine,
40                  QEMU/Bochs/etc, will be directed to this device.
41 */
42 int set_serial_in(int device);
43
44 /*
45  Procedure: Polling
46  Description: Gathers keyboard input via the serial port using
47               the technique of polling
48 */
49
50 int *polling(char *buffer, int *count);
51
52 #endif
```

6.11 /home/maximillian/Desktop/MAMA/include/core/tables.h File Reference

```
#include "system.h"
```

Classes

- struct [idt_entry_struct](#)
- struct [idt_struct](#)
- struct [gdt_descriptor_struct](#)
- struct [gdt_entry_struct](#)

Functions

- struct [idt_entry_struct](#) [__attribute__\(\(packed\)\)](#) idt_entry
- void [idt_set_gate](#) (u8int idx, u32int base, u16int sel, u8int flags)
- void [gdt_init_entry](#) (int idx, u32int base, u32int limit, u8int access, u8int flags)
- void [init_idt](#) ()
- void [init_gdt](#) ()

Variables

- u16int [base_low](#)
- u16int [sselect](#)
- u8int [zero](#)
- u8int [flags](#)
- u16int [base_high](#)
- u16int [limit](#)
- u32int [base](#)
- u16int [limit_low](#)
- u8int [base_mid](#)
- u8int [access](#)

6.11.1 Function Documentation

6.11.1.1 [__attribute__\(\)](#)

```
struct gdt\_entry\_struct \_\_attribute\_\_ (  
    (packed) )
```

6.11.1.2 gdt_init_entry()

```
void gdt_init_entry (
    int idx,
    u32int base,
    u32int limit,
    u8int access,
    u8int flags )
```

6.11.1.3 idt_set_gate()

```
void idt_set_gate (
    u8int idx,
    u32int base,
    u16int sel,
    u8int flags )
```

6.11.1.4 init_gdt()

```
void init_gdt ( )
```

6.11.1.5 init_idt()

```
void init_idt ( )
```

6.11.2 Variable Documentation

6.11.2.1 access

`u8int` access

6.11.2.2 base

`u32int` base

6.11.2.3 base_high

`u8int` base_high

6.11.2.4 base_low

`ul6int` base_low

6.11.2.5 base_mid

`u8int` base_mid

6.11.2.6 flags

`u8int` flags

6.11.2.7 limit

`ul6int` limit

6.11.2.8 limit_low

`ul6int` limit_low

6.11.2.9 sselect

`ul6int` sselect

6.11.2.10 zero

`u8int` zero

6.12 tables.h

Go to the documentation of this file.

```

1 #ifndef _TABLES_H
2 #define _TABLES_H
3
4 #include "system.h"
5
6 typedef struct idt_entry_struct
7 {
8     u16int base_low; //offset bits 0..15
9     u16int sselect; //stack selector in gdt or ldt
10    u8int zero; //this stays zero; unused
11    u8int flags; //attributes
12    u16int base_high; //offset bits 16..31
13 }
14 __attribute__((packed)) idt_entry;
15
16 typedef struct idt_struct
17 {
18     u16int limit;
19     u32int base;
20 }
21 __attribute__((packed)) idt_descriptor;
22
23 typedef struct gdt_descriptor_struct
24 {
25     u16int limit;
26     u32int base;
27 }
28 __attribute__((packed)) gdt_descriptor;
29
30 typedef struct gdt_entry_struct
31 {
32     u16int limit_low; //first 16 bits of limit
33     u16int base_low; //first 16 bits of base
34     u8int base_mid; //bits 16-23 of base
35     u8int access; //next 8 bits; access flags
36     u8int flags; //page granularity, size
37     u8int base_high; //last 8 bits of the base
38 }
39 __attribute__((packed)) gdt_entry;
40
41
42 void idt_set_gate(u8int idx, u32int base, u16int sel, u8int flags);
43 void gdt_init_entry(int idx, u32int base, u32int limit, u8int access,
44                     u8int flags);
45
46 void init_idt();
47 void init_gdt();
48
49 #endif

```

6.13 /home/maximillian/Desktop/MAMA/include/mem/heap.h File Reference

Classes

- struct [header](#)
- struct [footer](#)
- struct [index_entry](#)
- struct [index_table](#)
- struct [heap](#)

Macros

- #define [TABLE_SIZE](#) 0x1000
- #define [KHEAP_BASE](#) 0xD000000
- #define [KHEAP_MIN](#) 0x10000
- #define [KHEAP_SIZE](#) 0x1000000

Functions

- `u32int _kmalloc` (`u32int` size, `int` align, `u32int *`phys_addr)
- `u32int kmalloc` (`u32int` size)
- `u32int kfree` ()
- `void init_kheap` ()
- `u32int alloc` (`u32int` size, `heap *`hp, `int` align)
- `heap *` make_heap (`u32int` base, `u32int` max, `u32int` min)

6.13.1 Macro Definition Documentation

6.13.1.1 KHEAP_BASE

```
#define KHEAP_BASE 0xD000000
```

6.13.1.2 KHEAP_MIN

```
#define KHEAP_MIN 0x10000
```

6.13.1.3 KHEAP_SIZE

```
#define KHEAP_SIZE 0x1000000
```

6.13.1.4 TABLE_SIZE

```
#define TABLE_SIZE 0x1000
```

6.13.2 Function Documentation

6.13.2.1 _kmalloc()

```
u32int _kmalloc (  
    u32int size,  
    int align,  
    u32int * phys_addr )
```

6.13.2.2 alloc()

```
u32int alloc (
    u32int size,
    heap * hp,
    int align )
```

6.13.2.3 init_kheap()

```
void init_kheap ( )
```

6.13.2.4 kfree()

```
u32int kfree ( )
```

6.13.2.5 kmalloc()

```
u32int kmalloc (
    u32int size )
```

6.13.2.6 make_heap()

```
heap * make_heap (
    u32int base,
    u32int max,
    u32int min )
```


6.14 heap.h

[Go to the documentation of this file.](#)

```

1  #ifndef _HEAP_H
2  #define _HEAP_H
3
4  /* Kernel heap */
5  #define TABLE_SIZE 0x1000
6  #define KHEAP_BASE 0xD000000
7  #define KHEAP_MIN 0x10000
8  #define KHEAP_SIZE 0x1000000
9
10 /* Heap allocation header */
11 typedef struct {
12     int size;
13     int index_id;
14 } header;
15
16 typedef struct {
17     header head;
18 } footer;
19
20 typedef struct {
21     int size;
22     int empty;
23     u32int block;
24 } index_entry;
25
26 /* Kernel heap index table */
27 typedef struct {
28     index_entry table[TABLE_SIZE];
29     int id;
30 } index_table;
31
32 /* Heap structure */
33 typedef struct {
34     index_table index;
35     u32int base;
36     u32int max_size;
37     u32int min_size;
38 } heap;
39
40 /*
41  Procedure...: _kmalloc
42  Description...: Base-level kernel memory allocation routine. Used to
43                  provide page alignment and access physical addresses of allocations.
44                  Called by kmalloc with align=0, physical_address=0.
45 */
46 u32int _kmalloc(u32int size, int align, u32int *phys_addr);
47
48 /*
49  Procedure...: kmalloc
50  Description...: Standard kernel memory allocation routine. Use this unless you
51                  need to specify alignment or obtain a physical address. Calls _kmalloc.
52 */
53 u32int kmalloc(u32int size);
54
55 /*
56  Procedure...: kfree
57  Description...: Free kernel memory.
58 */
59 u32int kfree();
60
61 /*
62  Procedure...: init_kheap
63  Description...: Initialize the kernel heap, and set it as the current heap.
64 */
65 void init_kheap();
66
67 /*
68  Procedure...: alloc
69  Description...: Allocate some memory using the given heap. Can specify page-alignment.
70 */
71 u32int alloc(u32int size, heap *hp, int align);
72
73 /*
74  Procedure...: make_heap
75  Description...: Create a new heap.
76  Parameters...: base - physical start address of the heap
77                  max - maximum size the heap may grow to
78                  min - minimum/initial size
79 */
80 heap* make_heap(u32int base, u32int max, u32int min);
81
82 #endif

```

6.15 /home/maximillian/Desktop/MAMA/include/mem/paging.h File Reference

```
#include <system.h>
```

Classes

- struct [page_entry](#)
- struct [page_table](#)
- struct [page_dir](#)

Macros

- #define [PAGE_SIZE](#) 0x1000

Functions

- void [set_bit](#) (u32int addr)
- void [clear_bit](#) (u32int addr)
- u32int [get_bit](#) (u32int addr)
- u32int [first_free](#) ()
- void [init_paging](#) ()
- void [load_page_dir](#) (page_dir *new_page_dir)
- page_entry * [get_page](#) (u32int addr, page_dir *dir, int make_table)
- void [new_frame](#) (page_entry *page)

6.15.1 Macro Definition Documentation

6.15.1.1 PAGE_SIZE

```
#define PAGE_SIZE 0x1000
```

6.15.2 Function Documentation

6.15.2.1 clear_bit()

```
void clear_bit (  
    u32int addr )
```

6.15.2.2 first_free()

```
u32int first_free ( )
```

6.15.2.3 get_bit()

```
u32int get_bit (
    u32int addr )
```

6.15.2.4 get_page()

```
page_entry * get_page (
    u32int addr,
    page_dir * dir,
    int make_table )
```

6.15.2.5 init_paging()

```
void init_paging ( )
```

6.15.2.6 load_page_dir()

```
void load_page_dir (
    page_dir * new_page_dir )
```

6.15.2.7 new_frame()

```
void new_frame (
    page_entry * page )
```

6.15.2.8 set_bit()

```
void set_bit (
    u32int addr )
```

6.16 paging.h

[Go to the documentation of this file.](#)

```

1 #ifndef _PAGING_H
2 #define _PAGING_H
3
4 #include <system.h>
5
6 #define PAGE_SIZE 0x1000
7
8 /*
9  Page entry structure
10  Describes a single page in memory
11 */
12 typedef struct {
13     u32int present : 1;
14     u32int writeable : 1;
15     u32int usermode : 1;
16     u32int accessed : 1;
17     u32int dirty : 1;
18     u32int reserved : 7;
19     u32int frameaddr : 20;
20 } page_entry;
21
22 /*
23  Page table structure
24  Contains 1024 pages/frames
25 */
26 typedef struct {
27     page_entry pages[1024];
28 } page_table;
29
30 /*
31  Page directory structure
32  Limited to 1024 tables for now
33 */
34 typedef struct {
35     page_table *tables[1024];
36     u32int tables_phys[1024];
37 } page_dir;
38
39 /*
40  Procedure...: set_bit
41  Description...: Marks a page frame bit as in use (1).
42 */
43 void set_bit(u32int addr);
44
45 /*
46  Procedure...: clear_bit
47  Description...: Marks a page frame bit as free (0).
48 */
49 void clear_bit(u32int addr);
50
51 /*
52  Procedure...: get_bit
53  Description...: Checks if page frame is in use.
54 */
55 u32int get_bit(u32int addr);
56
57 /*
58  Procedure...: first_free
59  Description...: Finds the first free page frame.
60 */
61 u32int first_free();
62
63 /*
64  Procedure...: init_paging
65  Description...: Initializes the kernel page directory and
66                  initial kernel heap area. Performs identity mapping of
67                  the kernel frames such that the virtual addresses are
68                  equivalent to the physical addresses.
69 */
70 void init_paging();
71
72 /*
73  Procedure...: load_page_dir
74  Description...: Sets a page directory as the current
75                  directory and enables paging via the cr0 register.
76                  The cr3 register enables address translation from
77                  linear to physical addresses.
78                  http://en.wikipedia.org/wiki/Control\_register#Control\_registers\_in\_x86\_series
79 */
80 void load_page_dir(page_dir *new_page_dir);
81
82 */

```

```

83  Procedure...: get_page
84  Description...: Finds and returns a page, allocating a new
85  page table if necessary.
86  */
87  page_entry* get_page(u32int addr, page_dir *dir, int make_table);
88
89  /*
90  Procedure...: new_frame
91  Description...: Marks a frame as in use in the frame bitmap,
92  sets up the page, and saves the frame index in the page.
93  */
94  void new_frame(page_entry* page);
95
96  #endif

```

6.17 /home/maximillian/Desktop/MAMA/include/string.h File Reference

```
#include <system.h>
```

Functions

- int [isspace](#) (const char *c)
 - void * [memset](#) (void *s, int c, [size_t](#) n)
 - char * [strcpy](#) (char *s1, const char *s2)
 - char * [strcat](#) (char *s1, const char *s2)
 - int [strlen](#) (const char *s)
 - int [strcmp](#) (const char *s1, const char *s2)
 - char * [strtok](#) (char *s1, const char *s2)
 - int [atoi](#) (const char *s)
 - char * [itoa](#) (int i)
- Converts 32-bit integer to an array of 8-bit characters.*

6.17.1 Function Documentation

6.17.1.1 atoi()

```
int atoi (
    const char * s )
```

6.17.1.2 isspace()

```
int isspace (
    const char * c )
```

6.17.1.3 itoa()

```
char * itoa (
    int i )
```

Converts 32-bit integer to an array of 8-bit characters.

Converts an integer data type by breaking it down into its individual digits. Digits are stored individually into a character array.

Parameters

<i>i</i>	Integer that will be converted into ascii
----------	---

Returns

Returns a pointer to the start of the array of character bytes

6.17.1.4 memset()

```
void * memset (
    void * s,
    int c,
    size_t n )
```

6.17.1.5 strcat()

```
char * strcat (
    char * s1,
    const char * s2 )
```

6.17.1.6 strcmp()

```
int strcmp (
    const char * s1,
    const char * s2 )
```

6.17.1.7 strcpy()

```
char * strcpy (
    char * s1,
    const char * s2 )
```

6.17.1.8 strlen()

```
int strlen (
    const char * s )
```

6.17.1.9 strtok()

```
char * strtok (
    char * s1,
    const char * s2 )
```

6.18 string.h

[Go to the documentation of this file.](#)

```
1 #ifndef _STRING_H
2 #define _STRING_H
3
4 #include <system.h>
5
6 /*
7  Procedure...: isspace
8  Description...: Determine if a character is whitespace.
9  Params...: c-character to check
10 */
11 int isspace(const char *c);
12
13 /*
14  Procedure...: memset
15  Description...: Set a region of memory.
16  Params...: s-destination, c-byte to write, n-count
17 */
18 void* memset(void *s, int c, size_t n);
19
20 /*
21  Procedure...: strcpy
22  Description...: Copy one string to another.
23  Params...: s1-destination, s2-source
24 */
25 char* strcpy(char *s1, const char *s2);
26
27 /*
28  Procedure...: strcat
29  Description...: Concatenate the contents of one string onto another.
30  Params...: s1-destination, s2-source
31 */
32 char* strcat(char *s1, const char *s2);
33
34 /*
35  Procedure...: strlen
36  Description...: Returns the length of a string.
37  Params...: s-input string
38 */
39 int strlen(const char *s);
40
41 /*
42  Procedure...: strcmp
43  Description...: String comparison
44  Params...: s1-string 1, s2-string 2
45 */
46 int strcmp(const char *s1, const char *s2);
47
48 /*
49  Procedure...: strtok
50  Description...: Split string into tokens
51  Params...: s1-string, s2-delimiter
52 */
53 char* strtok(char *s1, const char *s2);
54
55 /*
56  Procedure...: atoi
57  Description...: Convert an ASCII string to an integer
58  Params...: const char *s -- String
59 */
60 int atoi(const char *s);
61
62 char *itoa(int i);
63
64 #endif
```

6.19 /home/maximillian/Desktop/MAMA/include/system.h File Reference

Classes

- struct [date_time](#)

Macros

- #define [NULL](#) 0
- #define [no_warn](#)(p) if (p) while (1) break
- #define [asm](#) __asm__
- #define [volatile](#) __volatile__
- #define [sti](#)() [asm volatile](#) ("sti::")
- #define [cli](#)() [asm volatile](#) ("cli::")
- #define [nop](#)() [asm volatile](#) ("nop::")
- #define [hlt](#)() [asm volatile](#) ("hlt::")
- #define [iret](#)() [asm volatile](#) ("iret::")
- #define [GDT_CS_ID](#) 0x01
- #define [GDT_DS_ID](#) 0x02

Typedefs

- typedef unsigned int [size_t](#)
- typedef unsigned char [u8int](#)
- typedef unsigned short [u16int](#)
- typedef unsigned long [u32int](#)

Functions

- void [klogv](#) (const char *msg)
- void [kpanic](#) (const char *msg)

6.19.1 Macro Definition Documentation

6.19.1.1 [asm](#)

```
#define asm __asm__
```

6.19.1.2 [cli](#)

```
#define cli( ) asm volatile ("cli::")
```


6.19.1.3 GDT_CS_ID

```
#define GDT_CS_ID 0x01
```

6.19.1.4 GDT_DS_ID

```
#define GDT_DS_ID 0x02
```

6.19.1.5 hlt

```
#define hlt( ) asm volatile ("hlt:::")
```

6.19.1.6 iret

```
#define iret( ) asm volatile ("iret:::")
```

6.19.1.7 no_warn

```
#define no_warn(  
    p ) if (p) while (1) break
```

6.19.1.8 nop

```
#define nop( ) asm volatile ("nop:::")
```

6.19.1.9 NULL

```
#define NULL 0
```

6.19.1.10 sti

```
#define sti( ) asm volatile ("sti:::)
```

6.19.1.11 volatile

```
#define volatile __volatile__
```

6.19.2 Typedef Documentation

6.19.2.1 size_t

```
typedef unsigned int size_t
```

6.19.2.2 u16int

```
typedef unsigned short u16int
```

6.19.2.3 u32int

```
typedef unsigned long u32int
```

6.19.2.4 u8int

```
typedef unsigned char u8int
```

6.19.3 Function Documentation

6.19.3.1 klogv()

```
void klogv (  
    const char * msg )
```

6.19.3.2 kpanic()

```
void kpanic (
    const char * msg )
```

6.20 system.h

[Go to the documentation of this file.](#)

```
1 #ifndef _SYSTEM_H
2 #define _SYSTEM_H
3
4 #define NULL 0
5
6 // Suppress 'unused parameter' warnings/errors
7 #define no_warn(p) if (p) while (1) break
8
9 // Allows compilation with gcc -std=c89
10 // May also help avoid naming conflicts
11 #define asm __asm__
12 #define volatile __volatile__
13
14 #define sti()    asm volatile ("sti::") //turn irqs off
15 #define cli()    asm volatile ("cli::") //turn irqs on
16 #define nop()    asm volatile ("nop::") //skip cycle
17 #define hlt()    asm volatile ("hlt::") //halt
18 #define iret()   asm volatile ("iret::") //interrupt return
19
20 #define GDT_CS_ID 0x01 //kernel code segment ID
21 #define GDT_DS_ID 0x02 //kernel data segment ID
22
23 /* System Types */
24 typedef unsigned int    size_t;
25 typedef unsigned char   u8int;
26 typedef unsigned short  u16int;
27 typedef unsigned long   u32int;
28
29 /* Time */
30 typedef struct {
31     int sec;
32     int min;
33     int hour;
34     int day_w;
35     int day_m;
36     int day_y;
37     int mon;
38     int year;
39 } date_time;
40
41 /* Test if interrupts are on */
42 static inline int irq_on()
43 {
44     int f;
45     asm volatile ("pushf\n\t"
46                  "popl %0"
47                  : "=g"(f));
48     return f & (1 << 9);
49 }
50
51 void klogv(const char *msg);
52 void kpanic(const char *msg);
53
54 #endif
```

6.21 /home/maximillian/Desktop/MAMA/kernel/core/interrupts.c File Reference

```
#include <system.h>
#include <core/io.h>
#include <core/serial.h>
#include <core/tables.h>
#include <core/interrupts.h>
```

Macros

- `#define PIC1 0x20`
- `#define PIC2 0xA0`
- `#define ICW1 0x11`
- `#define ICW4 0x01`
- `#define io_wait() asm volatile ("outb $0x80")`

Functions

- void `divide_error` ()
- void `debug` ()
- void `nmi` ()
- void `breakpoint` ()
- void `overflow` ()
- void `bounds` ()
- void `invalid_op` ()
- void `device_not_available` ()
- void `double_fault` ()
- void `coprocessor_segment` ()
- void `invalid_tss` ()
- void `segment_not_present` ()
- void `stack_segment` ()
- void `general_protection` ()
- void `page_fault` ()
- void `reserved` ()
- void `coprocessor` ()
- void `rtc_isr` ()
- void `sys_call_isr` ()
- void `isr0` ()
- void `do_isr` ()
- void `init_irq` (void)
- void `init_pic` (void)
- void `do_divide_error` ()
- void `do_debug` ()
- void `do_nmi` ()
- void `do_breakpoint` ()
- void `do_overflow` ()
- void `do_bounds` ()
- void `do_invalid_op` ()
- void `do_device_not_available` ()
- void `do_double_fault` ()
- void `do_coprocessor_segment` ()
- void `do_invalid_tss` ()
- void `do_segment_not_present` ()
- void `do_stack_segment` ()
- void `do_general_protection` ()
- void `do_page_fault` ()
- void `do_reserved` ()
- void `do_coprocessor` ()

Variables

- idt_entry `idt_entries` [256]

6.21.1 Macro Definition Documentation

6.21.1.1 ICW1

```
#define ICW1 0x11
```

6.21.1.2 ICW4

```
#define ICW4 0x01
```

6.21.1.3 io_wait

```
#define io_wait( ) asm volatile ("outb $0x80")
```

6.21.1.4 PIC1

```
#define PIC1 0x20
```

6.21.1.5 PIC2

```
#define PIC2 0xA0
```

6.21.2 Function Documentation

6.21.2.1 bounds()

```
void bounds ( )
```

6.21.2.2 breakpoint()

```
void breakpoint ( )
```

6.21.2.3 coprocessor()

```
void coprocessor ( )
```

6.21.2.4 coprocessor_segment()

```
void coprocessor_segment ( )
```

6.21.2.5 debug()

```
void debug ( )
```

6.21.2.6 device_not_available()

```
void device_not_available ( )
```

6.21.2.7 divide_error()

```
void divide_error ( )
```

6.21.2.8 do_bounds()

```
void do_bounds ( )
```

6.21.2.9 do_breakpoint()

```
void do_breakpoint ( )
```

6.21.2.10 do_coprocessor()

```
void do_coprocessor ( )
```

6.21.2.11 do_coprocessor_segment()

```
void do_coprocessor_segment ( )
```

6.21.2.12 do_debug()

```
void do_debug ( )
```

6.21.2.13 do_device_not_available()

```
void do_device_not_available ( )
```

6.21.2.14 do_divide_error()

```
void do_divide_error ( )
```

6.21.2.15 do_double_fault()

```
void do_double_fault ( )
```

6.21.2.16 do_general_protection()

```
void do_general_protection ( )
```

6.21.2.17 do_invalid_op()

```
void do_invalid_op ( )
```

6.21.2.18 do_invalid_tss()

```
void do_invalid_tss ( )
```

6.21.2.19 do_isr()

```
void do_isr ( )
```

6.21.2.20 do_nmi()

```
void do_nmi ( )
```

6.21.2.21 do_overflow()

```
void do_overflow ( )
```

6.21.2.22 do_page_fault()

```
void do_page_fault ( )
```

6.21.2.23 do_reserved()

```
void do_reserved ( )
```

6.21.2.24 do_segment_not_present()

```
void do_segment_not_present ( )
```

6.21.2.25 do_stack_segment()

```
void do_stack_segment ( )
```


6.21.2.26 double_fault()

```
void double_fault ( )
```

6.21.2.27 general_protection()

```
void general_protection ( )
```

6.21.2.28 init_irq()

```
void init_irq (
    void )
```

6.21.2.29 init_pic()

```
void init_pic (
    void )
```

6.21.2.30 invalid_op()

```
void invalid_op ( )
```

6.21.2.31 invalid_tss()

```
void invalid_tss ( )
```

6.21.2.32 isr0()

```
void isr0 ( )
```

6.21.2.33 nmi()

```
void nmi ( )
```

6.21.2.34 overflow()

```
void overflow ( )
```

6.21.2.35 page_fault()

```
void page_fault ( )
```

6.21.2.36 reserved()

```
void reserved ( )
```

6.21.2.37 rtc_isr()

```
void rtc_isr ( )
```

6.21.2.38 segment_not_present()

```
void segment_not_present ( )
```

6.21.2.39 stack_segment()

```
void stack_segment ( )
```

6.21.2.40 sys_call_isr()

```
void sys_call_isr ( )
```

6.21.3 Variable Documentation

6.21.3.1 idt_entries

```
idt_entry idt_entries[256] [extern]
```

6.22 /home/maximillian/Desktop/MAMA/kernel/core/kmain.c File Reference

```
#include <stdint.h>
#include <string.h>
#include <system.h>
#include <core/io.h>
#include <core/serial.h>
#include <core/tables.h>
#include <core/interrupts.h>
#include <mem/heap.h>
#include <mem/paging.h>
#include <modules/mpx_supt.h>
#include "term/commhand.c"
#include "term/dispatch/context.h"
#include "term/pcb/pcb.h"
#include "term/dnt/dnt.h"
#include "term/memory_management/mm.h"
```

Functions

- void [kmain](#) (void)

6.22.1 Function Documentation

6.22.1.1 kmain()

```
void kmain (
    void )
```

6.23 /home/maximillian/Desktop/MAMA/kernel/core/serial.c File Reference

```
#include <stdint.h>
#include <string.h>
#include <core/io.h>
#include <core/serial.h>
#include <term/history.h>
#include <term/visuals/syntax_highlight.h>
#include <term/visuals/syntax_highlight.c>
```

Macros

- `#define NO_ERROR 0`
- `#define DELETE 0b00001`
- `#define LEFT_ARROW 0b00010`
- `#define RIGHT_ARROW 0b00100`
- `#define UP_ARROW 0b01000`
- `#define DOWN_ARROW 0b10000`

Functions

- `int init_serial (int device)`
- `int serial_println (const char *msg)`
- `int serial_print (const char *msg)`
- `int set_serial_out (int device)`
- `int set_serial_in (int device)`
- `unsigned int consume_special ()`
- `int * polling (char *buffer, int *count)`
Serially poll characters from command line.

Variables

- `int serial_port_out = 0`
- `int serial_port_in = 0`

6.23.1 Macro Definition Documentation

6.23.1.1 DELETE

```
#define DELETE 0b00001
```

6.23.1.2 DOWN_ARROW

```
#define DOWN_ARROW 0b10000
```

6.23.1.3 LEFT_ARROW

```
#define LEFT_ARROW 0b00010
```

6.23.1.4 NO_ERROR

```
#define NO_ERROR 0
```

6.23.1.5 RIGHT_ARROW

```
#define RIGHT_ARROW 0b00100
```

6.23.1.6 UP_ARROW

```
#define UP_ARROW 0b01000
```

6.23.2 Function Documentation

6.23.2.1 consume_special()

```
unsigned int consume_special ( )
```

6.23.2.2 init_serial()

```
int init_serial (
    int device )
```

6.23.2.3 polling()

```
int * polling (
    char * buffer,
    int * count )
```

Serially poll characters from command line.

Polls input from keyboard and interprets each character individually as it is entered from the keyboard.

Parameters

<i>buffer</i>	Space allocated for single line on the command line
<i>count</i>	Size of the space allocated

Returns

Returns 0 upon success, -1 upon error

6.23.2.4 serial_print()

```
int serial_print (
    const char * msg )
```

6.23.2.5 serial_println()

```
int serial_println (
    const char * msg )
```

6.23.2.6 set_serial_in()

```
int set_serial_in (
    int device )
```

6.23.2.7 set_serial_out()

```
int set_serial_out (
    int device )
```

6.23.3 Variable Documentation**6.23.3.1 serial_port_in**

```
int serial_port_in = 0
```

6.23.3.2 serial_port_out

```
int serial_port_out = 0
```

6.24 /home/maximillian/Desktop/MAMA/kernel/core/system.c File Reference

```
#include <string.h>
#include <system.h>
#include <core/serial.h>
#include <modules/mpx_supt.h>
#include "term/pcb/pcb.h"
#include "term/dispatch/context.h"
#include <lib/out.h>
```

Functions

- void [klogv](#) (const char *msg)
- void [kpanic](#) (const char *msg)
- [u32int](#) * [sys_call](#) ([context](#) *registers)

Called to start interrupt.

Variables

- [pcb_t](#) * [cop](#)
Currently operating process.
- [context](#) * [global_context](#)
Context.
- [pcb_queue_t](#) * [priority_queue](#)
- [param](#) [params](#)

6.24.1 Function Documentation

6.24.1.1 klogv()

```
void klogv (
    const char * msg )
```

6.24.1.2 kpanic()

```
void kpanic (
    const char * msg )
```

6.24.1.3 sys_call()

```
u32int * sys_call (
    context * registers )
```

Called to start interrupt.

Is called by irq to determine the next routine to load

Parameters

<i>registers</i>	Context registers for the current process
------------------	---

Returns

Pointer to the process being loaded

6.24.2 Variable Documentation

6.24.2.1 cop

```
pcb_t* cop
```

Currently operating process.

6.24.2.2 global_context

```
context* global_context
```

Context.

6.24.2.3 params

```
param params [extern]
```


6.24.2.4 priority_queue

`pcb_queue_t* priority_queue` [extern]

6.25 /home/maximillian/Desktop/MAMA/kernel/core/tables.c File Reference

```
#include <string.h>
#include <core/tables.h>
```

Functions

- void `write_gdt_ptr` (`u32int`, `size_t`)
- void `write_idt_ptr` (`u32int`)
- void `idt_set_gate` (`u8int` `idx`, `u32int` `base`, `u16int` `sel`, `u8int` `flags`)
- void `init_idt` ()
- void `gdt_init_entry` (`int` `idx`, `u32int` `base`, `u32int` `limit`, `u8int` `access`, `u8int` `flags`)
- void `init_gdt` ()

Variables

- gdt_descriptor `gdt_ptr`
- gdt_entry `gdt_entries` [5]
- idt_descriptor `idt_ptr`
- idt_entry `idt_entries` [256]

6.25.1 Function Documentation

6.25.1.1 gdt_init_entry()

```
void gdt_init_entry (
    int idx,
    u32int base,
    u32int limit,
    u8int access,
    u8int flags )
```

6.25.1.2 idt_set_gate()

```
void idt_set_gate (
    u8int idx,
    u32int base,
    u16int sel,
    u8int flags )
```

6.25.1.3 init_gdt()

```
void init_gdt ( )
```

6.25.1.4 init_idt()

```
void init_idt ( )
```

6.25.1.5 write_gdt_ptr()

```
void write_gdt_ptr (
    u32int ,
    size_t )
```

6.25.1.6 write_idt_ptr()

```
void write_idt_ptr (
    u32int )
```

6.25.2 Variable Documentation

6.25.2.1 gdt_entries

```
gdt_entry gdt_entries[5]
```

6.25.2.2 gdt_ptr

```
gdt_descriptor gdt_ptr
```

6.25.2.3 idt_entries

```
idt_entry idt_entries[256]
```

6.25.2.4 idt_ptr

```
idt_descriptor idt_ptr
```

6.26 /home/maximillian/Desktop/MAMA/kernel/mem/heap.c File Reference

```
#include <system.h>
#include <string.h>
#include <core/serial.h>
#include <mem/heap.h>
#include <mem/paging.h>
```

Functions

- [u32int _kmalloc](#) ([u32int](#) size, [int](#) page_align, [u32int](#) *phys_addr)
- [u32int kmalloc](#) ([u32int](#) size)
- [u32int alloc](#) ([u32int](#) size, [heap](#) *h, [int](#) align)
- [heap](#) * [make_heap](#) ([u32int](#) base, [u32int](#) max, [u32int](#) min)

Variables

- [heap](#) * [kheap](#) = 0
- [heap](#) * [curr_heap](#) = 0
- [page_dir](#) * [kdir](#)
- [void](#) * [end](#)
- [void](#) [_end](#)
- [void](#) [__end](#)
- [u32int](#) [phys_alloc_addr](#) = ([u32int](#))&[end](#)

6.26.1 Function Documentation

6.26.1.1 `_kmalloc()`

```
u32int _kmalloc (
    u32int size,
    int page_align,
    u32int * phys_addr )
```

6.26.1.2 `alloc()`

```
u32int alloc (
    u32int size,
    heap * h,
    int align )
```

6.26.1.3 `kmalloc()`

```
u32int kmalloc (
    u32int size )
```

6.26.1.4 `make_heap()`

```
heap * make_heap (
    u32int base,
    u32int max,
    u32int min )
```

6.26.2 Variable Documentation

6.26.2.1 `__end`

```
void __end
```

6.26.2.2 `_end`

```
void _end
```

6.26.2.3 curr_heap

```
heap* curr_heap = 0
```

6.26.2.4 end

```
void* end [extern]
```

6.26.2.5 kdir

```
page_dir* kdir [extern]
```

6.26.2.6 kheap

```
heap* kheap = 0
```

6.26.2.7 phys_alloc_addr

```
u32int phys_alloc_addr = (u32int)&end
```

6.27 /home/maximillian/Desktop/MAMA/kernel/mem/paging.c File Reference

```
#include <system.h>
#include <string.h>
#include "mem/heap.h"
#include "mem/paging.h"
```

Functions

- void [set_bit](#) (u32int addr)
- void [clear_bit](#) (u32int addr)
- u32int [get_bit](#) (u32int addr)
- u32int [find_free](#) ()
- [page_entry](#) * [get_page](#) (u32int addr, [page_dir](#) *dir, int make_table)
- void [init_paging](#) ()
- void [load_page_dir](#) ([page_dir](#) *new_dir)
- void [new_frame](#) ([page_entry](#) *page)

Variables

- `u32int mem_size` = 0x4000000
- `u32int page_size` = 0x1000
- `u32int nframes`
- `u32int * frames`
- `page_dir * kdir` = 0
- `page_dir * cdir` = 0
- `u32int phys_alloc_addr`
- `heap * kheap`

6.27.1 Function Documentation

6.27.1.1 `clear_bit()`

```
void clear_bit (
    u32int addr )
```

6.27.1.2 `find_free()`

```
u32int find_free ( )
```

6.27.1.3 `get_bit()`

```
u32int get_bit (
    u32int addr )
```

6.27.1.4 `get_page()`

```
page_entry * get_page (
    u32int addr,
    page_dir * dir,
    int make_table )
```

6.27.1.5 `init_paging()`

```
void init_paging ( )
```

6.27.1.6 load_page_dir()

```
void load_page_dir (
    page_dir * new_dir )
```

6.27.1.7 new_frame()

```
void new_frame (
    page_entry * page )
```

6.27.1.8 set_bit()

```
void set_bit (
    u32int addr )
```

6.27.2 Variable Documentation

6.27.2.1 cdir

```
page_dir* cdir = 0
```

6.27.2.2 frames

```
u32int* frames
```

6.27.2.3 kdir

```
page_dir* kdir = 0
```

6.27.2.4 kheap

```
heap* kheap [extern]
```

6.27.2.5 mem_size

```
u32int mem_size = 0x4000000
```

6.27.2.6 nframes

```
u32int nframes
```

6.27.2.7 page_size

```
u32int page_size = 0x1000
```

6.27.2.8 phys_alloc_addr

```
u32int phys_alloc_addr [extern]
```

6.28 /home/maximillian/Desktop/MAMA/lib/out.c File Reference

```
#include <modules/mpx_supt.h>
#include <stdarg.h>
```

Functions

- int `print` (char *str, int len)
- int `putc` (char c)
- int `println` (char *str, int len)
- void `printf` (char *str,...)
- int `read` (char *buf, int len)

6.28.1 Function Documentation

6.28.1.1 print()

```
int print (
    char * str,
    int len )
```


6.28.1.2 `putc()`

```
int putc (
    char c )
```

6.28.1.3 `printf()`

```
void printf (
    char * str,
    ... )
```

6.28.1.4 `println()`

```
int println (
    char * str,
    int len )
```

6.28.1.5 `read()`

```
int read (
    char * buf,
    int len )
```

6.29 /home/maximillian/Desktop/MAMA/lib/out.h File Reference

Functions

- int [cmd_help](#) (char *command)
Prints help message for command.
- void [gettimeHelp](#) ()
Help page for [gettime\(\)](#) method.
- void [settimeHelp](#) ()
Help page for [settime\(\)](#) method.
- void [getdateHelp](#) ()
Help page for the [getdate\(\)](#) method.
- void [setdateHelp](#) ()
Help page for the [setdate\(\)](#) method.
- void [helpHelp](#) ()
Help page for the help command.
- void [shutdownHelp](#) ()
Help page for the shutdown command.
- void [helpList](#) ()

- Displays a list of common system commands.*
- void [versionHelp](#) ()
Help page for the version command.
- void [createpcbHelp](#) ()
Help page for createpcb.
- void [deletepcbHelp](#) ()
Help page for deletepcb.
- void [showpcbHelp](#) ()
Help page for showpcb.
- void [showallpcbHelp](#) ()
Help page for showallpcb.
- void [showreadypcbHelp](#) ()
Help page for showreadypcb.
- void [showblockedpcbHelp](#) ()
Help page for showblockedpcb.
- void [blockHelp](#) ()
Help page for block.
- void [unblockHelp](#) ()
Help page for unblock.
- void [setpriorityHelp](#) ()
Help page for setpriority.
- void [resumeHelp](#) ()
Help page for resume.
- void [suspendHelp](#) ()
Help page for suspend.
- void [loadr3Help](#) ()
Help page for loadr3.
- void [setalarmHelp](#) ()
Help page for setalarm.
- void [showalarmsHelp](#) ()
Help page for showalarm.
- void [freealarmHelp](#) ()
Help page for freealarm.
- void [resumeallHelp](#) ()
Help page for resumeallpcb.
- void [showallocHelp](#) ()
Help page for showalloc.
- void [showfreeHelp](#) ()
Help page for showfree.
- void [isemptyHelp](#) ()
Help page for isempty.
- void [clearHelp](#) ()
Help page for clear.
- void [aliasHelp](#) ()
Help page for alias.
- int [print](#) (char *, int)
- int [putc](#) (char)
- int [println](#) (char *, int)
- void [printf](#) (char *,...)
- int [read](#) (char *, int)

6.29.1 Function Documentation

6.29.1.1 aliasHelp()

```
void aliasHelp ( )
```

Help page for alias.

Displays the alias help pages

6.29.1.2 blockHelp()

```
void blockHelp ( )
```

Help page for block.

Displays the block help page

6.29.1.3 clearHelp()

```
void clearHelp ( )
```

Help page for clear.

Displays the clear help pages

6.29.1.4 cmd_help()

```
int cmd_help (
    char * command )
```

Prints help message for command.

Prints out a help message and basic syntax for a specific command

Parameters

<i>command</i>	Command which the user needs basic information and syntax for
----------------	---

Returns

1 upon success, -1 upon error

6.29.1.5 createpcbHelp()

```
void createpcbHelp ( )
```

Help page for createpcb.

Displays the createpcb help page

6.29.1.6 deletepcbHelp()

```
void deletepcbHelp ( )
```

Help page for deletepcb.

Displays the deletepcb help page

6.29.1.7 freealarmHelp()

```
void freealarmHelp ( )
```

Help page for freealarm.

Displays the freealarm help page

6.29.1.8 getdateHelp()

```
void getdateHelp ( )
```

Help page for the [getdate\(\)](#) method.

Prints out the name, usage, return and description for the [getdate\(\)](#) method.

6.29.1.9 gettimeHelp()

```
void gettimeHelp ( )
```

Help page for [gettime\(\)](#) method.

Prints out the name, usage, return and description for the [gettime\(\)](#) method.

6.29.1.10 helpHelp()

```
void helpHelp ( )
```

Help page for the help command.

Prints out the name, usage, return and description for the help command.

6.29.1.11 `helpList()`

```
void helpList ( )
```

Displays a list of common system commands.

Displays a list of common system commands for the user.

6.29.1.12 `isemptyHelp()`

```
void isemptyHelp ( )
```

Help page for isempty.

Displays the isempty help pages

6.29.1.13 `loadr3Help()`

```
void loadr3Help ( )
```

Help page for loadr3.

Displays the loadr3 help page

6.29.1.14 `print()`

```
int print (
    char * str,
    int len )
```

6.29.1.15 `putc()`

```
int putc (
    char c )
```

6.29.1.16 `printf()`

```
void printf (
    char * str,
    ... )
```

6.29.1.17 println()

```
int println (
    char * str,
    int len )
```

6.29.1.18 read()

```
int read (
    char * buf,
    int len )
```

6.29.1.19 resumeallHelp()

```
void resumeallHelp ( )
```

Help page for resumeallpcb.

Displays the resumeallpcb help page

6.29.1.20 resumeHelp()

```
void resumeHelp ( )
```

Help page for resume.

Displays the resume help page

6.29.1.21 setalarmHelp()

```
void setalarmHelp ( )
```

Help page for setalarm.

Displays the setalarm help page

6.29.1.22 setdateHelp()

```
void setdateHelp ( )
```

Help page for the [setdate\(\)](#) method.

Prints out the name, usage, and description for the [setdate\(\)](#) method.

6.29.1.23 setpriorityHelp()

```
void setpriorityHelp ( )
```

Help page for setpriority.

Displays the setpriority help page

6.29.1.24 settimeHelp()

```
void settimeHelp ( )
```

Help page for [settime\(\)](#) method.

Prints out the name, usage, and description for the [settime\(\)](#) method.

6.29.1.25 showalarmsHelp()

```
void showalarmsHelp ( )
```

Help page for showalarm.

Displays the showalarm help page

6.29.1.26 showallocHelp()

```
void showallocHelp ( )
```

Help page for showalloc.

Displays the showalloc help page

6.29.1.27 showallpcbHelp()

```
void showallpcbHelp ( )
```

Help page for showallpcb.

Displays the showallpcb help page

6.29.1.28 showblockedpcbHelp()

```
void showblockedpcbHelp ( )
```

Help page for showblockedpcb.

Displays the showblockedpcb help page

6.29.1.29 showfreeHelp()

```
void showfreeHelp ( )
```

Help page for showfree.

Displays the showfree help pages

6.29.1.30 showpcbHelp()

```
void showpcbHelp ( )
```

Help page for showpcb.

Displays the showpcb help page

6.29.1.31 showreadypcbHelp()

```
void showreadypcbHelp ( )
```

Help page for showreadypcb.

Displays the showreadypcb help page

6.29.1.32 shutdownHelp()

```
void shutdownHelp ( )
```

Help page for the shutdown command.

Prints out the name, usage, and description for the shutdown system command.

6.29.1.33 suspendHelp()

```
void suspendHelp ( )
```

Help page for suspend.

Displays the suspend help page

6.29.1.34 unblockHelp()

```
void unblockHelp ( )
```

Help page for unblock.

Displays te unblock help page

6.29.1.35 versionHelp()

```
void versionHelp ( )
```

Help page for the version command.

Displays the current version of the system.

6.30 out.h

[Go to the documentation of this file.](#)

```
1 #ifndef OUT_H
2 #define OUT_H
3
14 int cmd_help(char * command);
15
21 void gettimeHelp();
22
28 void settimeHelp();
29
35 void getdateHelp();
36
42 void setdateHelp();
43
49 void helpHelp();
50
57 void shutdownHelp();
58
64 void helpList();
65
72 void versionHelp();
73
80 void createpcbHelp();
81
87 void deletepcbHelp();
88
94 void showpcbHelp();
95
101 void showallpcbHelp();
102
108 void showreadypcbHelp();
109
115 void showblockedpcbHelp();
116
122 void blockHelp();
123
129 void unblockHelp();
130
136 void setpriorityHelp();
137
143 void resumeHelp();
144
150 void suspendHelp();
151
157 void loadr3Help();
158
164 void setalarmHelp();
165
171 void showalarmsHelp();
172
179 void freealarmHelp();
180
186 void resumeallHelp();
187
193 void showallocHelp();
194
200 void showfreeHelp();
201
207 void isemptyHelp();
208
214 void clearHelp();
215
221 void aliasHelp();
222
223
224 int print(char *, int);
225 int printc(char);
226 int println(char *, int);
227 void printf(char *, ...);
228 int read(char *, int);
229
230 #endif
```

6.31 /home/maximillian/Desktop/MAMA/lib/string.c File Reference

```
#include <system.h>
#include <string.h>
```

Functions

- int [strlen](#) (const char *s)
- char * [strcpy](#) (char *s1, const char *s2)
- int [atoi](#) (const char *s)
- char * [itoa](#) (int value)
Converts 32-bit integer to an array of 8-bit characters.
- int [strcmp](#) (const char *s1, const char *s2)
- char * [strcat](#) (char *s1, const char *s2)
- int [isspace](#) (const char *c)
- void * [memset](#) (void *s, int c, [size_t](#) n)
- char * [strtok](#) (char *s1, const char *s2)

6.31.1 Function Documentation

6.31.1.1 [atoi\(\)](#)

```
int atoi (
    const char * s )
```

6.31.1.2 [isspace\(\)](#)

```
int isspace (
    const char * c )
```

6.31.1.3 [itoa\(\)](#)

```
char * itoa (
    int i )
```

Converts 32-bit integer to an array of 8-bit characters.

Converts an integer data type by breaking it down into its individual digits. Digits are stored individually into a character array.

Parameters

<i>i</i>	Integer that will be converted into ascii
----------	---

Returns

Returns a pointer to the start of the array of character bytes

6.31.1.4 `memset()`

```
void * memset (
    void * s,
    int c,
    size_t n )
```

6.31.1.5 `strcat()`

```
char * strcat (
    char * s1,
    const char * s2 )
```

6.31.1.6 `strcmp()`

```
int strcmp (
    const char * s1,
    const char * s2 )
```

6.31.1.7 `strcpy()`

```
char * strcpy (
    char * s1,
    const char * s2 )
```

6.31.1.8 `strlen()`

```
int strlen (
    const char * s )
```

6.31.1.9 strtok()

```
char * strtok (
    char * s1,
    const char * s2 )
```

6.32 /home/maximillian/Desktop/MAMA/modules/mpx_supt.c File Reference

```
#include "mpx_supt.h"
#include <mem/heap.h>
#include <string.h>
#include <core/serial.h>
#include "../lib/out.h"
```

Functions

- int [sys_req](#) (int op_code, int device_id, char *buffer_ptr, int *count_ptr)
- void [mpx_init](#) (int cur_mod)
- void [sys_set_malloc](#) (u32int)(*func)(u32int))
- void [sys_set_free](#) (int)(*func)(void *))
- void * [sys_alloc_mem](#) (u32int size)
- int [sys_free_mem](#) (void *ptr)
- void [idle](#) ()

Variables

- [param](#) [params](#)
- int [current_module](#) = -1
- u32int(* [student_malloc](#))(u32int)
- int(* [student_free](#))(void *)

6.32.1 Function Documentation

6.32.1.1 idle()

```
void idle ( )
```

6.32.1.2 mpx_init()

```
void mpx_init (
    int cur_mod )
```

6.32.1.3 sys_alloc_mem()

```
void * sys_alloc_mem (
    u32int size )
```

6.32.1.4 sys_free_mem()

```
int sys_free_mem (
    void * ptr )
```

6.32.1.5 sys_req()

```
int sys_req (
    int op_code,
    int device_id,
    char * buffer_ptr,
    int * count_ptr )
```

6.32.1.6 sys_set_free()

```
void sys_set_free (
    int(*) (void *) func )
```

6.32.1.7 sys_set_malloc()

```
void sys_set_malloc (
    u32int(*) (u32int) func )
```

6.32.2 Variable Documentation

6.32.2.1 current_module

```
int current_module = -1
```

6.32.2.2 params

```
param params
```

6.32.2.3 student_free

```
int(* student_free) (void *) (  
    void * )
```

6.32.2.4 student_malloc

```
u32int(* student_malloc) (u32int) (  
    u32int )
```

6.33 /home/maximillian/Desktop/MAMA/modules/mpx_supt.h File Reference

```
#include <system.h>
```

Classes

- struct [param](#)

Macros

- #define [EXIT](#) 0
- #define [IDLE](#) 1
- #define [READ](#) 2
- #define [WRITE](#) 3
- #define [INVALID_OPERATION](#) 4
- #define [TRUE](#) 1
- #define [FALSE](#) 0
- #define [MODULE_R1](#) 0
- #define [MODULE_R2](#) 1
- #define [MODULE_R3](#) 2
- #define [MODULE_R4](#) 4
- #define [MODULE_R5](#) 8
- #define [MODULE_F](#) 9
- #define [IO_MODULE](#) 10
- #define [MEM_MODULE](#) 11
- #define [INVALID_BUFFER](#) 1000
- #define [INVALID_COUNT](#) 2000
- #define [DEFAULT_DEVICE](#) 111
- #define [COM_PORT](#) 222

Functions

- int [sys_req](#) (int op_code, int device_id, char *buffer_ptr, int *count_ptr)
- void [mpx_init](#) (int cur_mod)
- void [sys_set_malloc](#) (u32int)(*func)(u32int))
- void [sys_set_free](#) (int)(*func)(void *)
- void * [sys_alloc_mem](#) (u32int size)
- int [sys_free_mem](#) (void *ptr)
- void [idle](#) ()

6.33.1 Macro Definition Documentation

6.33.1.1 COM_PORT

```
#define COM_PORT 222
```

6.33.1.2 DEFAULT_DEVICE

```
#define DEFAULT_DEVICE 111
```

6.33.1.3 EXIT

```
#define EXIT 0
```

6.33.1.4 FALSE

```
#define FALSE 0
```

6.33.1.5 IDLE

```
#define IDLE 1
```

6.33.1.6 INVALID_BUFFER

```
#define INVALID_BUFFER 1000
```

6.33.1.7 INVALID_COUNT

```
#define INVALID_COUNT 2000
```

6.33.1.8 INVALID_OPERATION

```
#define INVALID_OPERATION 4
```

6.33.1.9 IO_MODULE

```
#define IO_MODULE 10
```

6.33.1.10 MEM_MODULE

```
#define MEM_MODULE 11
```

6.33.1.11 MODULE_F

```
#define MODULE_F 9
```

6.33.1.12 MODULE_R1

```
#define MODULE_R1 0
```

6.33.1.13 MODULE_R2

```
#define MODULE_R2 1
```


6.33.1.14 MODULE_R3

```
#define MODULE_R3 2
```

6.33.1.15 MODULE_R4

```
#define MODULE_R4 4
```

6.33.1.16 MODULE_R5

```
#define MODULE_R5 8
```

6.33.1.17 READ

```
#define READ 2
```

6.33.1.18 TRUE

```
#define TRUE 1
```

6.33.1.19 WRITE

```
#define WRITE 3
```

6.33.2 Function Documentation

6.33.2.1 idle()

```
void idle ( )
```

6.33.2.2 mpx_init()

```
void mpx_init (
    int cur_mod )
```

6.33.2.3 sys_alloc_mem()

```
void * sys_alloc_mem (
    u32int size )
```

6.33.2.4 sys_free_mem()

```
int sys_free_mem (
    void * ptr )
```

6.33.2.5 sys_req()

```
int sys_req (
    int op_code,
    int device_id,
    char * buffer_ptr,
    int * count_ptr )
```

6.33.2.6 sys_set_free()

```
void sys_set_free (
    int (*) (void *) func )
```

6.33.2.7 sys_set_malloc()

```
void sys_set_malloc (
    u32int (*) (u32int) func )
```

6.34 mpx_supt.h

[Go to the documentation of this file.](#)

```

1 #ifndef _MPX_SUPT_H
2 #define _MPX_SUPT_H
3
4 #include <system.h>
5
6 #define EXIT 0
7 #define IDLE 1
8 #define READ 2
9 #define WRITE 3
10 #define INVALID_OPERATION 4
11
12 #define TRUE 1
13 #define FALSE 0
14
15 #define MODULE_R1 0
16 #define MODULE_R2 1
17 #define MODULE_R3 2
18 #define MODULE_R4 4
19 #define MODULE_R5 8
20 #define MODULE_F 9
21 #define IO_MODULE 10
22 #define MEM_MODULE 11
23
24 // error codes
25 #define INVALID_BUFFER 1000
26 #define INVALID_COUNT 2000
27
28 #define DEFAULT_DEVICE 111
29 #define COM_PORT 222
30
31 typedef struct {
32     int op_code;
33     int device_id;
34     char *buffer_ptr;
35     int *count_ptr;
36 } param;
37
38 /*
39  Procedure...: sys_req
40  Description...: Generate interrupt 60H
41  Params...: int op_code one of (IDLE, EXIT, READ, WRITE)
42 */
43 int sys_req( int op_code, int device_id, char *buffer_ptr,
44             int *count_ptr );
45
46 /*
47  Procedure...: mpx_init
48  Description...: Initialize MPX support software
49  Params...: int cur_mod (symbolic constants MODULE_R1, MODULE_R2, etc
50 */
51 void mpx_init(int cur_mod);
52
53 /*
54  Procedure...: sys_set_malloc
55  Description...: Sets the memory allocation function for sys_alloc_mem
56  Params...: Function pointer
57 */
58 void sys_set_malloc(u32int (*func)(u32int));
59
60 /*
61  Procedure...: sys_set_free
62  Description...: Sets the memory free function for sys_free_mem
63  Params...: s1-destination, s2-source
64 */
65 void sys_set_free(int (*func)(void *));
66
67 /*
68  Procedure...: sys_alloc_mem
69  Description...: Allocates a block of memory (similar to malloc)
70  Params...: Number of bytes to allocate
71 */
72 void *sys_alloc_mem(u32int size);
73
74 /*
75  Procedure...: sys_free_mem
76  Description...: Frees memory
77  Params...: Pointer to block of memory to free
78 */
79 int sys_free_mem(void *ptr);
80
81 /*
82  Procedure...: idle

```

```
83  Description...: The idle process
84  Params...: None
85  */
86  void idle();
87
88  #endif
```

6.35 /home/maximillian/Desktop/MAMA/README.md File Reference

6.36 /home/maximillian/Desktop/MAMA/serial_driver/driver.c File Reference

Classes

- struct [dcb_t](#)

Macros

- #define [BASE_COM1](#)
- #define [DIVISOR_LATCH_LOW_BYTE_REGISTER](#) 0
- #define [DIVISOR_LATCH_HIGH_BYTE_REGISTER](#) 1
- #define [INTERRUPT_ENABLE_REGISTER](#) 1
- #define [INTERRUPT_IDENTIFICATION_REGISTER](#) 2
- #define [LINE_CONTROL_REGISTER](#) 3
- #define [MODEM_CONTROL_REGISTER](#) 4
- #define [LINE_STATUS_REGISTER](#) 5
- #define [MODEM_STATUS_REGISTER](#) 6
- #define [SCRATCH_REGISTER](#) 7
- #define [PIC_MASK](#) 0x21;
- #define [RING_BUFFER_SIZE](#) 100

Typedefs

- typedef struct [dcb_t](#) [dcb_t](#)

Enumerations

- enum [device_ready_state_t](#) { [OPEN](#) , [CLOSED](#) }
- enum [device_status_t](#) { [IDLE](#) , [READING](#) , [WRITING](#) }

Functions

- int [com_open](#) (int *eflag_p, int baud_rate)
- int [com_close](#) ()
- int [com_read](#) (char *buf, int *count)
- int [com_write](#) (char *buf, int *count)

Variables

- const `dcb_t` * `COM1_control_block` = `NULL`

6.36.1 Macro Definition Documentation

6.36.1.1 BASE

```
#define BASE COM1
```

6.36.1.2 DIVISOR_LATCH_HIGH_BYTE_REGISTER

```
#define DIVISOR_LATCH_HIGH_BYTE_REGISTER 1
```

6.36.1.3 DIVISOR_LATCH_LOW_BYTE_REGISTER

```
#define DIVISOR_LATCH_LOW_BYTE_REGISTER 0
```

6.36.1.4 INTERRUPT_ENABLE_REGISTER

```
#define INTERRUPT_ENABLE_REGISTER 1
```

6.36.1.5 INTERRUPT_IDENTIFICATION_REGISTER

```
#define INTERRUPT_IDENTIFICATION_REGISTER 2
```

6.36.1.6 LINE_CONTROL_REGISTER

```
#define LINE_CONTROL_REGISTER 3
```

6.36.1.7 LINE_STATUS_REGISTER

```
#define LINE_STATUS_REGISTER 5
```

6.36.1.8 MODEM_CONTROL_REGISTER

```
#define MODEM_CONTROL_REGISTER 4
```

6.36.1.9 MODEM_STATUS_REGISTER

```
#define MODEM_STATUS_REGISTER 6
```

6.36.1.10 PIC_MASK

```
#define PIC_MASK 0x21;
```

6.36.1.11 RING_BUFFER_SIZE

```
#define RING_BUFFER_SIZE 100
```

6.36.1.12 SCRATCH_REGISTER

```
#define SCRATCH_REGISTER 7
```

6.36.2 Typedef Documentation

6.36.2.1 dcb_t

```
typedef struct dcb_t dcb_t
```

6.36.3 Enumeration Type Documentation

6.36.3.1 device_ready_state_t

```
enum device_ready_state_t
```

Enumerator

OPEN	
CLOSED	

6.36.3.2 device_status_t

```
enum device_status_t
```

Enumerator

IDLE	
READING	
WRITING	

6.36.4 Function Documentation

6.36.4.1 com_close()

```
int com_close ( )
```

6.36.4.2 com_open()

```
int com_open (
    int * eflag_p,
    int baud_rate )
```

6.36.4.3 com_read()

```
int com_read (
    char * buf,
    int * count )
```

6.36.4.4 com_write()

```
int com_write (
    char * buf,
    int * count )
```

6.36.5 Variable Documentation

6.36.5.1 COM1_control_block

```
const dcb_t* COM1_control_block = NULL
```

6.37 /home/maximillian/Desktop/MAMA/term/args.c File Reference

```
#include "commhand.h"
#include "utils.h"
#include "args.h"
#include "syntax.h"
#include <lib/out.h>
#include <include/string.h>
```

Macros

- #define `MAX_PARSE_STACK_SIZE` 2

Functions

- int `get_token` (char **, char *, int)
- int `stack_empty` ()
- enum `SyntaxState` `stack_peek` ()
- void `stack_push` (enum `SyntaxState`)
- void `stack_pop` ()
- `parsed_args` * `parse_args` (char *arg_str)
- int `named_arg` (`parsed_args` *args, char *arg_name, char **arg_val)
- int `flag` (`parsed_args` *args, char *flag_name)
- int `next_unnamed_arg` (`parsed_args` *args, char **arg_val)

Variables

- enum `SyntaxState` `parse_stack` [`MAX_PARSE_STACK_SIZE`]
- int `stack_size` = 0
- enum `SyntaxState` `last_state`
- enum `SyntaxState` `cur_state`

6.37.1 Macro Definition Documentation

6.37.1.1 MAX_PARSE_STACK_SIZE

```
#define MAX_PARSE_STACK_SIZE 2
```

6.37.2 Function Documentation

6.37.2.1 flag()

```
int flag (  
    parsed_args * args,  
    char * flag_name )
```

6.37.2.2 get_token()

```
int get_token (  
    char ** arg_str,  
    char * token,  
    int max_token_len )
```

6.37.2.3 named_arg()

```
int named_arg (  
    parsed_args * args,  
    char * arg_name,  
    char ** arg_val )
```

6.37.2.4 next_unnamed_arg()

```
int next_unnamed_arg (  
    parsed_args * args,  
    char ** arg_val )
```

6.37.2.5 parse_args()

```
parsed_args * parse_args (
    char * arg_str )
```

6.37.2.6 stack_empty()

```
int stack_empty ( )
```

6.37.2.7 stack_peek()

```
enum SyntaxState stack_peek ( )
```

6.37.2.8 stack_pop()

```
void stack_pop ( )
```

6.37.2.9 stack_push()

```
void stack_push (
    enum SyntaxState state )
```

6.37.3 Variable Documentation

6.37.3.1 cur_state

```
enum SyntaxState cur_state
```

6.37.3.2 last_state

```
enum SyntaxState last_state
```

6.37.3.3 parse_stack

```
enum SyntaxState parse_stack[MAX_PARSE_STACK_SIZE]
```

6.37.3.4 stack_size

```
int stack_size = 0
```

6.38 /home/maximillian/Desktop/MAMA/term/args.h File Reference

Classes

- struct [parsed_args](#)

Typedefs

- typedef struct [parsed_args](#) [parsed_args](#)

Functions

- [parsed_args](#) * [parse_args](#) (char *)

6.38.1 Typedef Documentation

6.38.1.1 parsed_args

```
typedef struct parsed\_args parsed\_args
```

6.38.2 Function Documentation

6.38.2.1 parse_args()

```
parsed\_args * parse\_args (  
    char * arg_str )
```

6.39 args.h

[Go to the documentation of this file.](#)

```
1 #ifndef ARGS_H
2 #define ARGS_H
3
4 typedef struct parsed_args {
5     int flag_count;
6     int named_arg_count;
7     int unnamed_arg_count;
8     int unnamed_args_used_so_far;
9
10    char flags[MAX_CMD_FLAG_COUNT][MAX_CMD_ARG_NAME_LEN + 1];
11    char named_arg_names[MAX_CMD_NAMED_ARG_COUNT][MAX_CMD_ARG_NAME_LEN + 1];
12    char named_arg_values[MAX_CMD_NAMED_ARG_COUNT][MAX_CMD_ARG_VALUE_LEN + 1];
13    char unnamed_args[MAX_CMD_UNNAMED_ARG_COUNT][MAX_CMD_ARG_VALUE_LEN + 1];
14 } parsed_args;
15
16 parsed_args *parse_args(char *);
17
18 #endif
```

6.40 /home/maximillian/Desktop/MAMA/term/ascii/mama.c File Reference

```
#include "mama.h"
#include "term/dnt/dnt.h"
```

Functions

- void [mama](#) ()
mama ascii art

6.40.1 Function Documentation

6.40.1.1 mama()

```
void mama ( )
```

mama ascii art

One of the intro ascii art.

6.41 /home/maximillian/Desktop/MAMA/term/ascii/mama.h File Reference

Functions

- void [mama](#) ()
mama ascii art

6.41.1 Function Documentation

6.41.1.1 mama()

```
void mama ( )
```

mama ascii art

One of the intro ascii art.

6.42 mama.h

[Go to the documentation of this file.](#)

```
1  
7 void mama();
```

6.43 /home/maximillian/Desktop/MAMA/term/cmds/argtest.c File Reference

```
#include "../args.h"  
#include "../args.c"  
#include <lib/out.h>
```

Functions

- int [cmd_argtest](#) (char *arg_str)

6.43.1 Function Documentation

6.43.1.1 cmd_argtest()

```
int cmd_argtest (  
    char * arg_str )
```

6.44 /home/maximillian/Desktop/MAMA/term/cmds/echo.c File Reference

```
#include <lib/out.h>
```

Functions

- int [cmd_echo](#) (char *arg_str)

6.44.1 Function Documentation

6.44.1.1 cmd_echo()

```
int cmd_echo (  
    char * arg_str )
```

6.45 /home/maximillian/Desktop/MAMA/help.c File Reference

```
#include <lib/out.h>
```

Functions

- int [cmd_help](#) (char *command)
Prints help message for command.
- void [helpList](#) ()
Displays a list of common system commands.
- void [shutdownHelp](#) ()
Help page for the shutdown command.
- void [helpHelp](#) ()
Help page for the help command.
- void [setdateHelp](#) ()
Help page for the [setdate\(\)](#) method.
- void [getdateHelp](#) ()
Help page for the [getdate\(\)](#) method.
- void [gettimeHelp](#) ()
Help page for [gettime\(\)](#) method.
- void [settimeHelp](#) ()
Help page for [settime\(\)](#) method.
- void [versionOs](#) ()

6.45.1 Function Documentation

6.45.1.1 cmd_help()

```
int cmd_help (  
    char * command )
```

Prints help message for command.

Prints out a help message and basic syntax for a specific command

Parameters

<i>command</i>	Command which the user needs basic information and syntax for
----------------	---

Returns

1 upon success, -1 upon error

6.45.1.2 getdateHelp()

```
void getdateHelp ( )
```

Help page for the [getdate\(\)](#) method.

Prints out the name, usage, return and description for the [getdate\(\)](#) method.

6.45.1.3 gettimeHelp()

```
void gettimeHelp ( )
```

Help page for [gettime\(\)](#) method.

Prints out the name, usage, return and description for the [gettime\(\)](#) method.

6.45.1.4 helpHelp()

```
void helpHelp ( )
```

Help page for the help command.

Prints out the name, usage, return and description for the help command.

6.45.1.5 helpList()

```
void helpList ( )
```

Displays a list of common system commands.

Displays a list of common system commands for the user.

6.45.1.6 setdateHelp()

```
void setdateHelp ( )
```

Help page for the [setdate\(\)](#) method.

Prints out the name, usage, and description for the [setdate\(\)](#) method.

6.45.1.7 settimeHelp()

```
void settimeHelp ( )
```

Help page for [settime\(\)](#) method.

Prints out the name, usage, and description for the [settime\(\)](#) method.

6.45.1.8 shutdownHelp()

```
void shutdownHelp ( )
```

Help page for the shutdown command.

Prints out the name, usage, and description for the shutdown system command.

6.45.1.9 versionOs()

```
void versionOs ( )
```

6.46 /home/maximillian/Desktop/MAMA/term/cmds/help.c File Reference

```
#include <lib/out.h>
```

Functions

- int [cmd_help](#) (char *command)
Prints help message for command.
- void [versionHelp](#) ()
Help page for the version command.
- void [helpList](#) ()
Displays a list of common system commands.
- void [shutdownHelp](#) ()
Help page for the shutdown command.
- void [helpHelp](#) ()
Help page for the help command.
- void [clearHelp](#) ()
Help page for clear.
- void [aliasHelp](#) ()
Help page for alias.
- void [setdateHelp](#) ()
Help page for the [setdate\(\)](#) method.
- void [getdateHelp](#) ()
Help page for the [getdate\(\)](#) method.
- void [gettimeHelp](#) ()
Help page for [gettime\(\)](#) method.

- void [settimeHelp](#) ()
Help page for [settime\(\)](#) method.
- void [createpcbHelp](#) ()
Help page for [createpcb](#).
- void [deletepcbHelp](#) ()
Help page for [deletepcb](#).
- void [showpcbHelp](#) ()
Help page for [showpcb](#).
- void [showallpcbHelp](#) ()
Help page for [showallpcb](#).
- void [showreadypcbHelp](#) ()
Help page for [showreadypcb](#).
- void [showblockedpcbHelp](#) ()
Help page for [showblockedpcb](#).
- void [blockHelp](#) ()
Help page for [block](#).
- void [unblockHelp](#) ()
Help page for [unblock](#).
- void [setpriorityHelp](#) ()
Help page for [setpriority](#).
- void [resumeHelp](#) ()
Help page for [resume](#).
- void [suspendHelp](#) ()
Help page for [suspend](#).
- void [loadr3Help](#) ()
Help page for [loadr3](#).
- void [setalarmHelp](#) ()
Help page for [setalarm](#).
- void [showalarmsHelp](#) ()
Help page for [showalarm](#).
- void [freealarmHelp](#) ()
Help page for [freealarm](#).
- void [resumeallHelp](#) ()
Help page for [resumeallpcb](#).
- void [showallocHelp](#) ()
Help page for [showalloc](#).
- void [showfreeHelp](#) ()
Help page for [showfree](#).
- void [isemptyHelp](#) ()
Help page for [isempty](#).

6.46.1 Function Documentation

6.46.1.1 [aliasHelp\(\)](#)

```
void aliasHelp ( )
```

Help page for [alias](#).

Displays the [alias](#) help pages

6.46.1.2 blockHelp()

```
void blockHelp ( )
```

Help page for block.

Displays the block help page

6.46.1.3 clearHelp()

```
void clearHelp ( )
```

Help page for clear.

Displays the clear help pages

6.46.1.4 cmd_help()

```
int cmd_help (
    char * command )
```

Prints help message for command.

Prints out a help message and basic syntax for a specific command

Parameters

<i>command</i>	Command which the user needs basic information and syntax for
----------------	---

Returns

1 upon success, -1 upon error

6.46.1.5 createpcbHelp()

```
void createpcbHelp ( )
```

Help page for createpcb.

Displays the createpcb help page

6.46.1.6 deletpcbHelp()

```
void deletpcbHelp ( )
```

Help page for deletpcb.

Displays the deletpcb help page

6.46.1.7 freealarmHelp()

```
void freealarmHelp ( )
```

Help page for freealarm.

Displays the freealarm help page

6.46.1.8 getdateHelp()

```
void getdateHelp ( )
```

Help page for the [getdate\(\)](#) method.

Prints out the name, usage, return and description for the [getdate\(\)](#) method.

6.46.1.9 gettimeHelp()

```
void gettimeHelp ( )
```

Help page for [gettime\(\)](#) method.

Prints out the name, usage, return and description for the [gettime\(\)](#) method.

6.46.1.10 helpHelp()

```
void helpHelp ( )
```

Help page for the help command.

Prints out the name, usage, return and description for the help command.

6.46.1.11 helpList()

```
void helpList ( )
```

Displays a list of common system commands.

Displays a list of common system commands for the user.

6.46.1.12 isemptyHelp()

```
void isemptyHelp ( )
```

Help page for isempty.

Displays the isempty help pages

6.46.1.13 loadr3Help()

```
void loadr3Help ( )
```

Help page for loadr3.

Displays the loadr3 help page

6.46.1.14 resumeallHelp()

```
void resumeallHelp ( )
```

Help page for resumeallpcb.

Displays the resumeallpcb help page

6.46.1.15 resumeHelp()

```
void resumeHelp ( )
```

Help page for resume.

Displays the resume help page

6.46.1.16 setalarmHelp()

```
void setalarmHelp ( )
```

Help page for setalarm.

Displays the setalarm help page

6.46.1.17 setdateHelp()

```
void setdateHelp ( )
```

Help page for the [setdate\(\)](#) method.

Prints out the name, usage, and description for the [setdate\(\)](#) method.

6.46.1.18 setpriorityHelp()

```
void setpriorityHelp ( )
```

Help page for setpriority.

Displays the setpriority help page

6.46.1.19 settimeHelp()

```
void settimeHelp ( )
```

Help page for [settime\(\)](#) method.

Prints out the name, usage, and description for the [settime\(\)](#) method.

6.46.1.20 showalarmsHelp()

```
void showalarmsHelp ( )
```

Help page for showalarm.

Displays the showalarm help page

6.46.1.21 showallocHelp()

```
void showallocHelp ( )
```

Help page for showalloc.

Displays the showalloc help page

6.46.1.22 showallpcbHelp()

```
void showallpcbHelp ( )
```

Help page for showallpcb.

Displays the showallpcb help page

6.46.1.23 showblockedpcbHelp()

```
void showblockedpcbHelp ( )
```

Help page for showblockedpcb.

Displays the showblockedpcb help page

6.46.1.24 showfreeHelp()

```
void showfreeHelp ( )
```

Help page for showfree.

Displays the showfree help pages

6.46.1.25 showpcbHelp()

```
void showpcbHelp ( )
```

Help page for showpcb.

Displays the showpcb help page

6.46.1.26 showreadypcbHelp()

```
void showreadypcbHelp ( )
```

Help page for showreadypcb.

Displays the showreadypcb help page

6.46.1.27 shutdownHelp()

```
void shutdownHelp ( )
```

Help page for the shutdown command.

Prints out the name, usage, and description for the shutdown system command.

6.46.1.28 suspendHelp()

```
void suspendHelp ( )
```

Help page for suspend.

Displays the suspend help page

6.46.1.29 unblockHelp()

```
void unblockHelp ( )
```

Help page for unblock.

Displays te unblock help page

6.46.1.30 versionHelp()

```
void versionHelp ( )
```

Help page for the version command.

Displays the current version of the system.

6.47 /home/maximillian/Desktop/MAMA/term/cmds/shutdown.c File Reference

```
#include <lib/out.h>
```

Functions

- int `cmd_shutdown` (char *arg_str)
Handler for calls to the shutdown command.

6.47.1 Function Documentation

6.47.1.1 `cmd_shutdown()`

```
int cmd_shutdown (  
    char * arg_str )
```

Handler for calls to the shutdown command.

Prompts for user confirmation before shutting the system down.

Parameters

<code>arg_str</code>	The arguments passed to the shutdown command. Unused by the handler.
----------------------	--

Returns

The exit code of the command, indicating whether or not the user confirmed the request to shutdown the system. Returns 0 if the user confirmed the request, 1 otherwise.

6.48 /home/maximillian/Desktop/MAMA/term/cmds/version.c File Reference

```
#include <lib/out.h>
```

Functions

- int `cmd_version` (char *arg_str)
Handler for the version command.

6.48.1 Function Documentation

6.48.1.1 `cmd_version()`

```
int cmd_version (
    char * arg_str )
```

Handler for the version command.

Prints the current version of the operating system.

Parameters

<code><i>arg_str</i></code>	The arguments passed to the version command. Unused by the handler.
-----------------------------	---

Returns

The exit code of the command, always 0.

6.49 /home/maximillian/Desktop/MAMA/term/commands.h File Reference

```
#include "cmds/help.c"
#include "cmds/shutdown.c"
#include "cmds/echo.c"
#include "cmds/version.c"
#include "cmds/argtest.c"
#include "cmds/pcb.c"
#include "cmds/clear.c"
```

6.50 `commands.h`

[Go to the documentation of this file.](#)

```
1 #ifndef COMMANDS_H
2 #define COMMANDS_H
3
4 #include "cmds/help.c"
5 #include "cmds/shutdown.c"
6 #include "cmds/echo.c"
7 #include "cmds/version.c"
8 #include "cmds/argtest.c"
9 #include "cmds/pcb.c"
10 #include "cmds/clear.c"
11
12 #endif
```


6.51 /home/maximillian/Desktop/MAMA/term/commhand.c File Reference

```
#include <include/string.h>
#include <modules/mpx_supt.h>
#include "visuals/colorize.c"
#include "visuals/clear.c"
#include "history.c"
#include "commhand.h"
#include "commands.h"
#include "visuals/syntax_highlight.h"
#include "visuals/hints.h"
#include <lib/out.c>
#include "dnt/dnt.c"
#include "utils.h"
#include "ascii/mama.c"
#include "dispatch/context.c"
#include "pcb/pcb.c"
#include "memory_management/mm.c"
#include <term/args.h>
```

Classes

- struct [cmd_mapping](#)

Typedefs

- typedef int(* [cmd_func_t](#)) (char *)
- typedef struct [cmd_mapping](#) [cmd_mapping](#)

Functions

- int [cmd_alias](#) (char *)
- int [is_name_char](#) (char)
Returns whether or not the specified character is a valid character in an identifier, such as a command or argument name.
- void [extract_cmd_name](#) (char *, char *, int *, int *)
- [cmd_mapping](#) * [fetch_cmd_mapping](#) (char *)
- void [commhand](#) ()
Displays command line and interprets inputted commands.

Variables

- [cmd_mapping](#) [cmd_mappings](#) [[MAX_CMD_COUNT](#)]
- [pcb_queue_t](#) * [priority_queue](#)

6.51.1 Typedef Documentation

6.51.1.1 cmd_func_t

```
typedef int (* cmd_func_t) (char *)
```

6.51.1.2 cmd_mapping

```
typedef struct cmd_mapping cmd_mapping
```

6.51.2 Function Documentation

6.51.2.1 cmd_alias()

```
int cmd_alias (
    char * arg_str )
```

6.51.2.2 commhand()

```
void commhand ( )
```

Displays command line and interprets inputted commands.

Parses through the input that was polled from the command line and interprets the command that was inputted (typically the first word)

Returns

Returns 0 upon success, -1 upon error

6.51.2.3 extract_cmd_name()

```
void extract_cmd_name (
    char * cmd_str,
    char * cmd_name,
    int * cmd_name_len,
    int * args_start_index )
```

6.51.2.4 fetch_cmd_mapping()

```
cmd_mapping * fetch_cmd_mapping (
    char * cmd_name )
```

6.51.2.5 is_name_char()

```
int is_name_char (
    char c )
```

Returns whether or not the specified character is a valid character in an identifier, such as a command or argument name.

Parameters

<code>c</code>	The character to test.
----------------	------------------------

Returns

True if the specified character `c` is valid in an identifier, false otherwise.

6.51.3 Variable Documentation

6.51.3.1 `cmd_mappings`

```
cmd_mapping cmd_mappings[MAX_CMD_COUNT]
```

6.51.3.2 `priority_queue`

```
pcb_queue_t* priority_queue [extern]
```

6.52 /home/maximillian/Desktop/MAMA/term/commhand.h File Reference

Macros

- `#define MAX_CMD_STRING_LEN 100`
- `#define MAX_CMD_NAME_LEN 30`
- `#define MAX_CMD_HIST_LEN 20`
- `#define MAX_CMD_ARG_NAME_LEN 30`
- `#define MAX_CMD_ARG_VALUE_LEN 40`
- `#define MAX_CMD_FLAG_COUNT 10`
- `#define MAX_CMD_NAMED_ARG_COUNT 10`
- `#define MAX_CMD_UNNAMED_ARG_COUNT 10`
- `#define MAX_CMD_COUNT 200`

Functions

- `void commhand ()`
Displays command line and interprets inputted commands.

6.52.1 Macro Definition Documentation

6.52.1.1 MAX_CMD_ARG_NAME_LEN

```
#define MAX_CMD_ARG_NAME_LEN 30
```

6.52.1.2 MAX_CMD_ARG_VALUE_LEN

```
#define MAX_CMD_ARG_VALUE_LEN 40
```

6.52.1.3 MAX_CMD_COUNT

```
#define MAX_CMD_COUNT 200
```

6.52.1.4 MAX_CMD_FLAG_COUNT

```
#define MAX_CMD_FLAG_COUNT 10
```

6.52.1.5 MAX_CMD_HIST_LEN

```
#define MAX_CMD_HIST_LEN 20
```

6.52.1.6 MAX_CMD_NAME_LEN

```
#define MAX_CMD_NAME_LEN 30
```

6.52.1.7 MAX_CMD_NAMED_ARG_COUNT

```
#define MAX_CMD_NAMED_ARG_COUNT 10
```

6.52.1.8 MAX_CMD_STRING_LEN

```
#define MAX_CMD_STRING_LEN 100
```

6.52.1.9 MAX_CMD_UNNAMED_ARG_COUNT

```
#define MAX_CMD_UNNAMED_ARG_COUNT 10
```

6.52.2 Function Documentation

6.52.2.1 commhand()

```
void commhand ( )
```

Displays command line and interprets inputted commands.

Parses through the input that was polled from the command line and interprets the command that was inputted (typically the first word)

Returns

Returns 0 upon success, -1 upon error

6.53 commhand.h

[Go to the documentation of this file.](#)

```
1 /* the logic for each command the user has to run is contained in a separate file in term/cmds
2  * each file should contain a function to run this command and possibly any helper functions the command
   needs to run
3  * include each of these files below - make sure to add an #include directive if you write a new command
4  */
5 #ifndef COMMHAND_H
6 #define COMMHAND_H
7
8 #define MAX_CMD_STRING_LEN 100
9 #define MAX_CMD_NAME_LEN 30
10 #define MAX_CMD_HIST_LEN 20
11 #define MAX_CMD_ARG_NAME_LEN 30
12 #define MAX_CMD_ARG_VALUE_LEN 40
13 #define MAX_CMD_FLAG_COUNT 10
14 #define MAX_CMD_NAMED_ARG_COUNT 10
15 #define MAX_CMD_UNNAMED_ARG_COUNT 10
16
17 #define MAX_CMD_COUNT 200
18
19 void commhand();
20 #endif
```

6.54 /home/maximillian/Desktop/MAMA/term/dispatch/context.c File Reference

```
#include "context.h"
#include "term/pcb/pcb.h"
#include "procsr3.c"
#include <lib/out.h>
```

Functions

- void `yield` ()
Causes commhand to yield.
- int `loadr3` (char *p)
Loads r3 'processes'.
- `pcb_t` * `dispatcher` (char *name, void(*func)(void))
Stores context on the stack.

6.54.1 Function Documentation

6.54.1.1 dispatcher()

```
pcb_t * dispatcher (
    char * pcb,
    void(*) (void) func )
```

Stores context on the stack.

With a given pcb and method to run, the dispatcher will store context registers onto the PCB stack.

Parameters

<i>pcb</i>	PCB where context is stored
<i>func</i>	Method that is ran within the process

6.54.1.2 loadr3()

```
int loadr3 (
    char * p )
```

Loads r3 'processes'.

Loads all r3 'processes' into memory in a suspended ready state at any priority of the users choosing

Parameters

<i>p</i>	Empty parameter
----------	-----------------

Returns

Returns 0 upon success, 1 upon error **This may change

6.54.1.3 yield()

```
void yield ( )
```

Causes commhand to yield.

Forces commhand to yield to other processes. If any processes are in the ready queue, they will be executed.

6.55 /home/maximillian/Desktop/MAMA/term/dispatch/context.h File Reference

```
#include "term/pcb/pcb.h"
```

Classes

- struct [context](#)
Context of the currently operating process.

Typedefs

- typedef struct [context](#) [context](#)
Context of the currently operating process.

Functions

- void [yield](#) ()
Causes commhand to yield.
- int [loadr3](#) (char *p)
Loads r3 'processes'.
- [pcb_t](#) * [dispatcher](#) (char *pcb, void(*func)(void))
Stores context on the stack.

6.55.1 Typedef Documentation

6.55.1.1 context

```
typedef struct context context
```

Context of the currently operating process.

6.55.2 Function Documentation

6.55.2.1 dispatcher()

```
pcb_t * dispatcher (  
    char * pcb,  
    void(*) (void) func )
```

Stores context on the stack.

With a given pcb and method to run, the dispatcher will store context registers onto the PCB stack.

Parameters

<i>pcb</i>	PCB where context is stored
<i>func</i>	Method that is ran within the process

6.55.2.2 loadr3()

```
int loadr3 (
    char * p )
```

Loads r3 'processes'.

Loads all r3 'processes' into memory in a suspended ready state at any priority of the users choosing

Parameters

<i>p</i>	Empty parameter
----------	-----------------

Returns

Returns 0 upon success, 1 upon error **This may change

6.55.2.3 yield()

```
void yield ( )
```

Causes commhand to yield.

Forces commhand to yield to other processes. If any processes are in the ready queue, they will be executed.

6.56 context.h

[Go to the documentation of this file.](#)

```
1 #ifndef CONTEXT_H
2 #define CONTEXT_H
3
4 #include "term/pcb/pcb.h"
5
6
7 typedef struct context {
8     u32int gs, fs, es, ds;
9
10     u32int edi, esi, ebp, esp, ebx, edx, ecx, eax;
11
12     // Other special registers
13     u32int eip, cs, eflags;
14 } context;
15
16 void yield();
17
18 int loadr3(char * p);
19
20 pcb_t * dispatcher(char * pcb, void (* func) (void));
21
22 #endif
```

6.57 /home/maximillian/Desktop/MAMA/term/dispatch/procsr3.c File Reference

```
#include "../include/system.h"
#include "../include/core/serial.h"
#include "../modules/mpx_supt.h"
#include "procsr3.h"
```

Macros

- #define [RC_1](#) 1
- #define [RC_2](#) 2
- #define [RC_3](#) 3
- #define [RC_4](#) 4
- #define [RC_5](#) 5

Functions

- void [proc1](#) ()
- void [proc2](#) ()
- void [proc3](#) ()
- void [proc4](#) ()
- void [proc5](#) ()

Variables

- char * [msg1](#) = "proc1 dispatched\n"
- char * [msg2](#) = "proc2 dispatched\n"
- char * [msg3](#) = "proc3 dispatched\n"
- char * [msg4](#) = "proc4 dispatched\n"
- char * [msg5](#) = "proc5 dispatched\n"
- int [msgSize](#) = 18
- char * [er1](#) = "proc1 ran after it was terminated"
- char * [er2](#) = "proc2 ran after it was terminated"
- char * [er3](#) = "proc3 ran after it was terminated"
- char * [er4](#) = "proc4 ran after it was terminated"
- char * [er5](#) = "proc5 ran after it was terminated"
- int [erSize](#) = 34

6.57.1 Macro Definition Documentation

6.57.1.1 [RC_1](#)

```
#define RC\_1 1
```

6.57.1.2 RC_2

```
#define RC_2 2
```

6.57.1.3 RC_3

```
#define RC_3 3
```

6.57.1.4 RC_4

```
#define RC_4 4
```

6.57.1.5 RC_5

```
#define RC_5 5
```

6.57.2 Function Documentation

6.57.2.1 proc1()

```
void proc1 ( )
```

6.57.2.2 proc2()

```
void proc2 ( )
```

6.57.2.3 proc3()

```
void proc3 ( )
```

6.57.2.4 `proc4()`

```
void proc4 ( )
```

6.57.2.5 `proc5()`

```
void proc5 ( )
```

6.57.3 Variable Documentation

6.57.3.1 `er1`

```
char* er1 = "proc1 ran after it was terminated"
```

6.57.3.2 `er2`

```
char* er2 = "proc2 ran after it was terminated"
```

6.57.3.3 `er3`

```
char* er3 = "proc3 ran after it was terminated"
```

6.57.3.4 `er4`

```
char* er4 = "proc4 ran after it was terminated"
```

6.57.3.5 `er5`

```
char* er5 = "proc5 ran after it was terminated"
```

6.57.3.6 erSize

```
int erSize = 34
```

6.57.3.7 msg1

```
char* msg1 = "proc1 dispatched\n"
```

6.57.3.8 msg2

```
char* msg2 = "proc2 dispatched\n"
```

6.57.3.9 msg3

```
char* msg3 = "proc3 dispatched\n"
```

6.57.3.10 msg4

```
char* msg4 = "proc4 dispatched\n"
```

6.57.3.11 msg5

```
char* msg5 = "proc5 dispatched\n"
```

6.57.3.12 msgSize

```
int msgSize = 18
```

6.58 /home/maximillian/Desktop/MAMA/term/dispatch/procsr3.h File Reference

Functions

- void [proc1](#) ()
- void [proc2](#) ()
- void [proc3](#) ()
- void [proc4](#) ()
- void [proc5](#) ()

6.58.1 Function Documentation

6.58.1.1 [proc1\(\)](#)

```
void proc1 ( )
```

6.58.1.2 [proc2\(\)](#)

```
void proc2 ( )
```

6.58.1.3 [proc3\(\)](#)

```
void proc3 ( )
```

6.58.1.4 [proc4\(\)](#)

```
void proc4 ( )
```

6.58.1.5 [proc5\(\)](#)

```
void proc5 ( )
```

6.59 procsr3.h

[Go to the documentation of this file.](#)

```
1 #ifndef PROCSR3_H
2 #define PROCSR3_H
3
4 void procl();
5
6 void proc2();
7
8 void proc3();
9
10 void proc4();
11
12 void proc5();
13
14 #endif
```

6.60 /home/maximillian/Desktop/MAMA/term/dnt/dnt.c File Reference

```
#include "dnt.h"
#include <modules/mpx_supt.h>
```

Functions

- int [setdate](#) (char *date)
Sets the date of the system.
- int [setDateInMemory](#) (int month, int day, int year)
Sets the date in memory.
- int [getdate](#) (char *p)
Gets the date of the system.
- int [settime](#) (char *time)
Sets the time of the system.
- void [setTimeInMemory](#) (int hour, int minute, int second)
Sets the time into memory.
- int [gettime](#) (char *p)
Gets the system time.
- unsigned char [ltoBCD](#) (unsigned int value)
Converts 32-bit integer to 8-bit BCD.
- unsigned int [BCDtol](#) (unsigned char value)
Converts 8-bit BCD to 32-bit integer.
- char * [intToMonth](#) (int value)
Converts masked int into string month.
- char * [intToDayOfWeek](#) (int value)
Converts integer to string day of the week.
- int [daysInMonth](#) (int month, int year)
Calculates the number of days in a month.
- int [setAlarm](#) (char *args)
Set an alarm.
- int [showAlarms](#) (char *p)
Show all alarms.
- int [freeAlarm](#) (char *time)
Remove alarm from alarms.
- void [currentTime](#) ()
Current time.
- void [dispatchAlarm](#) ()
Alarm process.

Variables

- char `alarms` [10][6] = { "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0" }
- char `messages` [10][32] = { "\0", "\0" }
- char `current_time` [6]

6.60.1 Function Documentation

6.60.1.1 BCDtoI()

```
unsigned int BCDtoI (
    unsigned char value )
```

Converts 8-bit BCD to 32-bit integer.

Converts an 8-bit BCD unsigned char to a 32-bit unsigned integer.

Parameters

<i>value</i>	8-bit BCD value that will be converted to 32-bit int
--------------	--

Returns

Returns 32-bit unsigned int

6.60.1.2 currentTime()

```
void currentTime ( )
```

Current time.

Gets the current time and stores it into string.

6.60.1.3 daysInMonth()

```
int daysInMonth (
    int month,
    int year )
```

Calculates the number of days in a month.

Calculates the number of days in the month based upon which month it is. If year is divisible by four then it is a leap year and will add 1 day for February for a total of 29 days. Otherwise, February will be 28 days.

Parameters

<i>month</i>	The month in the year (January = 1...December = 12)
<i>year</i>	The year that was being set

Returns

Returns the number of days in the month

6.60.1.4 dispatchAlarm()

```
void dispatchAlarm ( )
```

Alarm process.

The function that will be used during context switching. This will check all alarm times against the current time

6.60.1.5 freeAlarm()

```
int freeAlarm (
    char * alarm )
```

Remove alarm from alarms.

Removes the alarm from the alarm list and 'frees' the spot

Parameters

<i>time</i>	Alarm to remove from list
-------------	---------------------------

Returns

Returns 0 upon success, -1 upon error

6.60.1.6 getdate()

```
int getdate (
    char * p )
```

Gets the date of the system.

Returns a string that represents the current date of the system. This is in the format DayOfWeek, Month Day, Year
Ex: Wednesday, August 25, 2021

Parameters

<i>p</i>	Empty parameter that is required to call this method. Does not do anything.
----------	---

Returns

Returns 1 upon success, -1 upon error

6.60.1.7 gettimeofday()

```
int gettimeofday (  
    char * p )
```

Gets the system time.

Gets the system time from memory by reading from the corresponding memory address. Time will be writtin to the interface in the syntax of Hour:Minute:Second Ex: 10:06:23

Parameters

<i>Empty</i>	parameter that does not do anything. Required in order to call from commhand
--------------	--

Returns

Returns 1 upon success, -1 upon error

6.60.1.8 intToDayOfWeek()

```
char * intToDayOfWeek (  
    int value )
```

Converts integer to string day of the week.

Converts a masked integer into an unmasked string day of the week. The days of the week are Sunday to Saturday and are 1 to 7 respectivley. 1 = Sunday 2 = Monday ... 7 = Saturday

Parameters

<i>value</i>	The masked integer value of month
--------------	-----------------------------------

Returns

Returns the unasked string value of month

6.60.1.9 intToMonth()

```
char * intToMonth (
    int value )
```

Converts masked int into string month.

Converts integer to a string month.

Converts the masked integer value of month and converts it into an unmasked string of month

Parameters

<i>value</i>	Masked integer month
--------------	----------------------

Returns

Returns unmasked string of month

6.60.1.10 ItoBCD()

```
unsigned char ItoBCD (
    unsigned int value )
```

Converts 32-bit integer to 8-bit BCD.

Uses basic arithmetic and bit shifting to convert from 32-bit integer to 8-bit BCD.

Parameters

<i>value</i>	The 32-bit integer that is converted to BCD
--------------	---

Returns

8-bit BCD number as an unsigned char

6.60.1.11 setAlarm()

```
int setAlarm (
    char * args )
```

Set an alarm.

Sets an alarm which will print a user defined message. Alarm will go off at specified time.

Parameters

<i>args</i>	Time and (optional) message
-------------	-----------------------------

Returns

Returns 0 upon success and -1 upon error

6.60.1.12 setdate()

```
int setdate (
    char * date )
```

Sets the date of the system.

Parses the parameter to setdate, breaking the parameter into month, day and year before passing it to setDateInMemory. The basic syntax is month.day.year

Parameters

<i>date</i>	The parameter that is passed with setdate. This string is parsed and each segment is converted to a 32-bit int.
-------------	---

Returns

Returns 1 upon success, -1 upon error

6.60.1.13 setDateInMemory()

```
int setDateInMemory (
    int month,
    int day,
    int year )
```

Sets the date in memory.

Sets the date in memory by assigning the values to the appropriate places in memory. This method is called by the setdate method.

Parameters

<i>month</i>	The month (1 = January ... 12 = December)
<i>day</i>	The day in the month. Can be between 0 and 32
<i>year</i>	The current year. This is a 2-digit number

Returns

Returns 1 upon success, -1 upon error

6.60.1.14 settime()

```
int settime (
    char * time )
```

Sets the time of the system.

Takes the parameter which will be parsed into 32-bit int (later converted to BCD) and sets it into memory. The syntax is Hour.Minute.Second Ex: 10.23.00

Parameters

<i>The</i>	parameter passed with the settime call
------------	--

Returns

Returns 1 upon success, -1 upon error

6.60.1.15 setTimeInMemory()

```
void setTimeInMemory (
    int hour,
    int minute,
    int second )
```

Sets the time into memory.

This method is called by the settime method. Writes the data into memory. First converts all parameter from 32-bit int to 8-bit BCD and then writes to the appropriate address.

Parameters

<i>hour</i>	32-bit int hour
<i>minute</i>	32-bit int minute
<i>second</i>	32-bit int second

6.60.1.16 showAlarms()

```
int showAlarms (
    char * p )
```

Show all alarms.

Print all alarms currently in the alarm list

Parameters

<i>p</i>	Empty parameters
----------	------------------

Returns

Returns 0 upon success, -1 upon error

6.60.2 Variable Documentation

6.60.2.1 alarms

```
char alarms[10][6] = { "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0" }
```

6.60.2.2 current_time

```
char current_time[6]
```

6.60.2.3 messages

```
char messages[10][32] = { "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0", "\0" }
```

6.61 /home/maximillian/Desktop/MAMA/term/dnt/dnt.h File Reference

Macros

- `#define MAX_HOURS 23`
The largest value that the user can set their hours to.
- `#define MAX_MINUTES 59`
The largest value that the user can set their minutes to.
- `#define MAX_SECONDS 59`
The largest value that the user can set their seconds to.
- `#define MAX_YEAR 99`
The largest value that the user can set their year to.
- `#define MAX_MONTH 12`
The largest value that the user can set their month to.

- #define `MAX_DAY` 31
The largest value that the user can set their day to.
- #define `MIN_YEAR` 10
Minimum year that can be set in memory.
- #define `MIN_MONTH` 1
Minimum month that can be set in memory.
- #define `MIN_DAY` 1
Minimum day that can be set in memory.
- #define `EPOCH_YEAR` 1970
Unix Epoch year.
- #define `EPOCH_FIRST_DAY_OF_YEAR` 1
Unix Epoch first day of the year.
- #define `EPOCH_FIRST_MONTH_OF_YEAR` 1
Unix Epoch first month of the year.
- #define `EPOCH_FIRST_DAY_OF_WEEK_OF_YEAR` 5
Unix Epoch first day of the week in the year.
- #define `DAYS_IN_YEAR` 365
Number of days in a normal year.
- #define `DAYS_IN_LEAP_YEAR` 366
Number of days in a leap year.
- #define `MIN` 0
Minimum value that can be set for hours, minutes, and seconds.

Functions

- int `setdate` (char *date)
Sets the date of the system.
- int `setDateInMemory` (int month, int day, int year)
Sets the date in memory.
- int `getdate` (char *p)
Gets the date of the system.
- int `settime` (char *time)
Sets the time of the system.
- void `setTimeInMemory` (int hour, int minute, int second)
Sets the time into memory.
- int `gettime` (char *p)
Gets the system time.
- unsigned char `ltoBCD` (unsigned int value)
Converts 32-bit integer to 8-bit BCD.
- unsigned int `BCDtol` (unsigned char value)
Converts 8-bit BCD to 32-bit integer.
- char * `intToMonth` (int value)
Converts integer to a string month.
- char * `intToDayOfWeek` (int value)
Converts integer to string day of the week.
- int `daysInMonth` (int month, int year)
Calculates the number of days in a month.
- int `setAlarm` (char *args)
Set an alarm.
- int `showAlarms` (char *p)

Show all alarms.

- int `freeAlarm` (char *alarm)

Remove alarm from alarms.

- void `dispatchAlarm` ()

Alarm process.

- void `currentTime` ()

Current time.

6.61.1 Macro Definition Documentation

6.61.1.1 DAYS_IN_LEAP_YEAR

```
#define DAYS_IN_LEAP_YEAR 366
```

Number of days in a leap year.

6.61.1.2 DAYS_IN_YEAR

```
#define DAYS_IN_YEAR 365
```

Number of days in a normal year.

6.61.1.3 EPOCH_FIRST_DAY_OF_WEEK_OF_YEAR

```
#define EPOCH_FIRST_DAY_OF_WEEK_OF_YEAR 5
```

Unix Epoch first day of the week in the year.

6.61.1.4 EPOCH_FIRST_DAY_OF_YEAR

```
#define EPOCH_FIRST_DAY_OF_YEAR 1
```

Unix Epoch first day of the year.

6.61.1.5 EPOCH_FIRST_MONTH_OF_YEAR

```
#define EPOCH_FIRST_MONTH_OF_YEAR 1
```

Unix Epoch first month of the year.

6.61.1.6 EPOCH_YEAR

```
#define EPOCH_YEAR 1970
```

Unix Epoch year.

6.61.1.7 MAX_DAY

```
#define MAX_DAY 31
```

The largest value that the user can set their day to.

6.61.1.8 MAX_HOURS

```
#define MAX_HOURS 23
```

The largest value that the user can set their hours to.

6.61.1.9 MAX_MINUTES

```
#define MAX_MINUTES 59
```

The largest value that the user can set their minutes to.

6.61.1.10 MAX_MONTH

```
#define MAX_MONTH 12
```

The largest value that the user can set their month to.

6.61.1.11 MAX_SECONDS

```
#define MAX_SECONDS 59
```

The largest value that the user can set their seconds to.

6.61.1.12 MAX_YEAR

```
#define MAX_YEAR 99
```

The largest value that the user can set their year to.

6.61.1.13 MIN

```
#define MIN 0
```

Minimum value that can be set for hours, minutes, and seconds.

6.61.1.14 MIN_DAY

```
#define MIN_DAY 1
```

Minimum day that can be set in memory.

6.61.1.15 MIN_MONTH

```
#define MIN_MONTH 1
```

Minimum month that can be set in memory.

6.61.1.16 MIN_YEAR

```
#define MIN_YEAR 10
```

Minimum year that can be set in memory.

6.61.2 Function Documentation

6.61.2.1 BCDtoI()

```
unsigned int BCDtoI (
    unsigned char value )
```

Converts 8-bit BCD to 32-bit integer.

Converts an 8-bit BCD unsigned char to a 32-bit unsigned integer.

Parameters

<i>value</i>	8-bit BCD value that will be converted to 32-bit int
--------------	--

Returns

Returns 32-bit unsigned int

6.61.2.2 `currentTime()`

```
void currentTime ( )
```

Current time.

Gets the current time and stores it into string.

6.61.2.3 `daysInMonth()`

```
int daysInMonth (
    int month,
    int year )
```

Calculates the number of days in a month.

Calculates the number of days in the month based upon which month it is. If year is divisible by four then it is a leap year and will add 1 day for February for a total of 29 days. Otherwise, February will be 28 days.

Parameters

<i>month</i>	The month in the year (January = 1...December = 12)
<i>year</i>	The year that was being set

Returns

Returns the number of days in the month

6.61.2.4 `dispatchAlarm()`

```
void dispatchAlarm ( )
```

Alarm process.

The function that will be used during context switching. This will check all alarm times against the current time

6.61.2.5 freeAlarm()

```
int freeAlarm (
    char * alarm )
```

Remove alarm from alarms.

Removes the alarm from the alarm list and 'frees' the spot

Parameters

<i>time</i>	Alarm to remove from list
-------------	---------------------------

Returns

Returns 0 upon success, -1 upon error

6.61.2.6 getdate()

```
int getdate (
    char * p )
```

Gets the date of the system.

Returns a string that represents the current date of the system. This is in the format DayOfWeek, Month Day, Year
Ex: Wednesday, August 25, 2021

Parameters

<i>p</i>	Empty parameter that is required to call this method. Does not do anything.
----------	---

Returns

Returns 1 upon success, -1 upon error

6.61.2.7 gettime()

```
int gettime (
    char * p )
```

Gets the system time.

Gets the system time from memory by reading from the corresponding memory address. Time will be writtin to the interface in the syntax of Hour:Minute:Second Ex: 10:06:23

Parameters

<i>Empty</i>	parameter that does not do anything. Required in order to call from commhand
--------------	--

Returns

Returns 1 upon success, -1 upon error

6.61.2.8 intToDayOfWeek()

```
char * intToDayOfWeek (  
    int value )
```

Converts integer to string day of the week.

Converts a masked integer into an unmasked string day of the week. The days of the week are Sunday to Saturday and are 1 to 7 respectively. 1 = Sunday 2 = Monday ... 7 = Saturday

Parameters

<i>value</i>	The masked integer value of month
--------------	-----------------------------------

Returns

Returns the unmasked string value of month

6.61.2.9 intToMonth()

```
char * intToMonth (  
    int value )
```

Converts integer to a string month.

Converts masked int into string month.

Converts a masked integer into an unmasked string month. The months are January to December and are 1 to 12 respectively. 1 = January 2 = February ... 12 = December

Parameters

<i>value</i>	The masked month
--------------	------------------

Returns

Returns unmasked string month

Converts the masked integer value of month and converts it into an unmasked string of month

Parameters

<i>value</i>	Masked integer month
--------------	----------------------

Returns

Returns unmasked string of month

Converts integer to a string month.

Converts the masked integer value of month and converts it into an unmasked string of month

Parameters

<i>value</i>	Masked integer month
--------------	----------------------

Returns

Returns unmasked string of month

6.61.2.10 ItoBCD()

```
unsigned char ItoBCD (
    unsigned int value )
```

Converts 32-bit integer to 8-bit BCD.

Uses basic arithmetic and bit shifting to convert from 32-bit integer to 8-bit BCD.

Parameters

<i>value</i>	The 32-bit integer that is converted to BCD
--------------	---

Returns

8-bit BCD number as an unsigned char

6.61.2.11 setAlarm()

```
int setAlarm (
    char * args )
```

Set an alarm.

Sets an alarm which will print a user defined message. Alarm will go off at specified time.

Parameters

<i>args</i>	Time and (optional) message
-------------	-----------------------------

Returns

Returns 0 upon success and -1 upon error

6.61.2.12 setdate()

```
int setdate (
    char * date )
```

Sets the date of the system.

Parses the parameter to setdate, breaking the parameter into month, day and year before passing it to setDateInMemory. The basic syntax is month.day.year

Parameters

<i>date</i>	The parameter that is passed with setdate. This string is parsed and each segment is converted to a 32-bit int.
-------------	---

Returns

Returns 1 upon success, -1 upon error

6.61.2.13 setDateInMemory()

```
int setDateInMemory (
    int month,
    int day,
    int year )
```

Sets the date in memory.

Sets the date in memory by assigning the values to the appropriate places in memory. This method is called by the setdate method.

Parameters

<i>month</i>	The month (1 = January ... 12 = December)
<i>day</i>	The day in the month. Can be between 0 and 32
<i>year</i>	The current year. This is a 2-digit number

Returns

Returns 1 upon success, -1 upon error

6.61.2.14 settime()

```
int settime (
    char * time )
```

Sets the time of the system.

Takes the parameter which will be parsed into 32-bit int (later converted to BCD) and sets it into memory. The syntax is Hour.Minute.Second Ex: 10.23.00

Parameters

<i>The</i>	parameter passed with the settime call
------------	--

Returns

Returns 1 upon success, -1 upon error

6.61.2.15 setTimeInMemory()

```
void setTimeInMemory (
    int hour,
    int minute,
    int second )
```

Sets the time into memory.

This method is called by the settime method. Writes the data into memory. First converts all parameter from 32-bit int to 8-bit BCD and then writes to the appropriate address.

Parameters

<i>hour</i>	32-bit int hour
<i>minute</i>	32-bit int minute
<i>second</i>	32-bit int second

6.61.2.16 showAlarms()

```
int showAlarms (
    char * p )
```

Show all alarms.

Print all alarms currently in the alarm list

Parameters

<i>p</i>	Empty parameters
----------	------------------

Returns

Returns 0 upon success, -1 upon error

6.62 dnt.h

[Go to the documentation of this file.](#)

```
1
2 #define MAX_HOURS 23
4 #define MAX_MINUTES 59
6 #define MAX_SECONDS 59
7
9 #define MAX_YEAR 99
11 #define MAX_MONTH 12
13 #define MAX_DAY 31
14
16 #define MIN_YEAR 10
18 #define MIN_MONTH 1
20 #define MIN_DAY 1
21
23 #define EPOCH_YEAR 1970
25 #define EPOCH_FIRST_DAY_OF_YEAR 1
27 #define EPOCH_FIRST_MONTH_OF_YEAR 1
29 #define EPOCH_FIRST_DAY_OF_WEEK_OF_YEAR 5
31 #define DAYS_IN_YEAR 365
33 #define DAYS_IN_LEAP_YEAR 366
34
36 #define MIN 0
37
51 int setdate(char * date);
52
66 int setDateInMemory(int month,int day,int year);
67
80 int getdate(char * p);
81
94 int settime(char * time);
95
106 void setTimeInMemory(int hour, int minute, int second);
107
120 int gettime(char * p);
121
132 unsigned char ItoBCD(unsigned int value);
133
144 unsigned int BCDtoI(unsigned char value);
145
161 char * intToMonth(int value);
162
178 char * intToDayOfWeek(int value);
179
190 char * intToMonth(int value);
191
206 int daysInMonth(int month, int year);
207
```

```

217 int setAlarm(char * args);
218
228 int showAlarms(char * p);
229
240 int freeAlarm(char * alarm);
241
249 void dispatchAlarm();
250
258 void currentTime();

```

6.63 /home/maximillian/Desktop/MAMA/term/history.c File Reference

```

#include "commhand.h"
#include "visuals/cursor.c"
#include "visuals/syntax_highlight.h"
#include <lib/out.h>

```

Functions

- int `circular_next_index` (int)

Whether or not the most recent entry in the user's command history has been discarded by calling `hist_discard_↔last_frame`.
- int `circular_prev_index` (int i)

Returns the index immediately preceding the specified index in `cmd_hist`, an array-based circular queue containing entries in the user's command history.
- void `write_hist_to_buf` (char *buf, int *index, int *len)

Writes the history entry pointed to by `cmd_hist_current_index` to the specified buffer and prints the new buffer to the terminal.
- void `hist_rewind` (char *internal_buf, int *internal_index, int *internal_buf_len)

Moves backwards 1 entry in the user's command history.
- void `hist_forward` (char *internal_buf, int *internal_index, int *internal_buf_len)

Moves forwards 1 entry in the user's command history.
- void `hist_discard_last_frame` ()

Removes the most recent command input from the user from the user's command history.
- char * `hist_next_frame` ()

Requests a buffer to write user input to that will become the most recent entry in the user's command history.

6.63.1 Function Documentation

6.63.1.1 `circular_next_index()`

```

int circular_next_index (
    int i )

```

Whether or not the most recent entry in the user's command history has been discarded by calling `hist_discard_↔last_frame`.

Returns the index immediately following the specified index in `cmd_hist`, an array-based circular queue containing entries in the user's command history.

Parameters

<i>i</i>	An index in cmd_hist.
----------	-----------------------

Returns

The index of the slot immediately following the slot at index *i* in cmd_hist.

6.63.1.2 circular_prev_index()

```
int circular_prev_index (
    int i )
```

Returns the index immediately preceding the specified index in cmd_hist, an array-based circular queue containing entries in the user's command history.

Parameters

<i>i</i>	An index in cmd_hist.
----------	-----------------------

Returns

The index of the slot immediately preceding the slot at index *i* in cmd_hist.

6.63.1.3 hist_discard_last_frame()

```
void hist_discard_last_frame ( )
```

Removes the most recent command input from the user from the user's command history.

6.63.1.4 hist_forward()

```
void hist_forward (
    char * internal_buf,
    int * internal_index,
    int * internal_buf_len )
```

Moves forwards 1 entry in the user's command history.

Parameters

<i>internal_buf</i>	The buffer managed by low-level read operations containing user input from the terminal. Contents will be overwritten with the next entry in the user's command history.
<i>internal_index</i>	A pointer to the position of the cursor, managed by low-level read operations. The cursor position will be adjusted to point to the end of the line.
<i>internal_buf_len</i>	A pointer to the length of the buffer containing user input to the terminal. Will be adjusted to contain the length of the history entry being written to the buffer.

6.63.1.5 hist_next_frame()

```
char * hist_next_frame ( )
```

Requests a buffer to write user input to that will become the most recent entry in the user's command history.

Returns

A pointer to the first slot in a character buffer representing the next entry in the user's command history.

6.63.1.6 hist_rewind()

```
void hist_rewind (
    char * internal_buf,
    int * internal_index,
    int * internal_buf_len )
```

Moves backwards 1 entry in the user's command history.

Parameters

<i>internal_buf</i>	The buffer managed by low-level read operations containing user input from the terminal. Contents will be overwritten with the previous entry in the user's command history.
<i>internal_index</i>	A pointer to the position of the cursor, managed by low-level read operations. The cursor position will be adjusted to point to the end of the line.
<i>internal_buf_len</i>	A pointer to the length of the buffer containing user input to the terminal. Will be adjusted to contain the length of the history entry being written to the buffer.

6.63.1.7 write_hist_to_buf()

```
void write_hist_to_buf (
    char * buf,
```

```
int * index,
int * len )
```

Writes the history entry pointed to by `cmd_hist_current_index` to the specified buffer and prints the new buffer to the terminal.

Used internally by `hist_rewind` and `hist_forward`.

Parameters

<i>buf</i>	The buffer to write the current history entry to.
<i>index</i>	A pointer to the position of the cursor in the user's terminal.
<i>len</i>	A pointer to the length of the buffer.

6.64 /home/maximillian/Desktop/MAMA/term/history.h File Reference

Functions

- void [hist_rewind](#) (char *, int *, int *)
Moves backwards 1 entry in the user's command history.
- void [hist_forward](#) (char *, int *, int *)
Moves forwards 1 entry in the user's command history.
- char * [hist_next_frame](#) ()
Requests a buffer to write user input to that will become the most recent entry in the user's command history.

6.64.1 Function Documentation

6.64.1.1 hist_forward()

```
void hist_forward (
    char * internal_buf,
    int * internal_index,
    int * internal_buf_len )
```

Moves forwards 1 entry in the user's command history.

Parameters

<i>internal_buf</i>	The buffer managed by low-level read operations containing user input from the terminal. Contents will be overwritten with the next entry in the user's command history.
<i>internal_index</i>	A pointer to the position of the cursor, managed by low-level read operations. The cursor position will be adjusted to point to the end of the line.
<i>internal_buf_len</i>	A pointer to the length of the buffer containing user input to the terminal. Will be adjusted to contain the length of the history entry being written to the buffer.

6.64.1.2 hist_next_frame()

```
char * hist_next_frame ( )
```

Requests a buffer to write user input to that will become the most recent entry in the user's command history.

Returns

A pointer to the first slot in a character buffer representing the next entry in the user's command history.

6.64.1.3 hist_rewind()

```
void hist_rewind (
    char * internal_buf,
    int * internal_index,
    int * internal_buf_len )
```

Moves backwards 1 entry in the user's command history.

Parameters

<i>internal_buf</i>	The buffer managed by low-level read operations containing user input from the terminal. Contents will be overwritten with the previous entry in the user's command history.
<i>internal_index</i>	A pointer to the position of the cursor, managed by low-level read operations. The cursor position will be adjusted to point to the end of the line.
<i>internal_buf_len</i>	A pointer to the length of the buffer containing user input to the terminal. Will be adjusted to contain the length of the history entry being written to the buffer.

6.65 history.h

[Go to the documentation of this file.](#)

```
1 #ifndef HISTORY_H
2 #define HISTORY_H
3
4 void hist_rewind(char *, int *, int *);
5 void hist_forward(char *, int *, int *);
6 char *hist_next_frame();
7
8 #endif
```

6.66 /home/maximillian/Desktop/MAMA/term/memory_↵ management/mm.c File Reference

```
#include "mm.h"
#include <term/utils.h>
```

```
#include <include/core/serial.h>
```

Functions

- int `initHeap` (u32int size)
Allocate all memory available for the MPX.
- u32int `allocateMemory` (u32int size)
Allocate additional memory from the heap.
- void `removeFMCB` (cmcb_s *cmcb)
- void `removeAMCB` (cmcb_s *cmcb)
- void `insertAMCB` (cmcb_s *mcb)
- void `insertFMCB` (cmcb_s *mcb)
- int `showAllocated` (char *discard)
Shows addresses and block size of all blocks in allocated list.
- int `freeMemory` (void *addr)
Free a block of memory.
- int `isEmpty` (char *p)
- int `showFree` (char *p)
Shows the addresses and block size of all block in free list.

Variables

- u32int `start_addr`
Start address of the heap.
- mcb_queue_s `allocated`
- mcb_queue_s `free`
- mcb_queue_s * `amcb` = &`allocated`
Allocated Memory Control List.
- mcb_queue_s * `fmcb` = &`free`
Free Memory Control List.

6.66.1 Function Documentation

6.66.1.1 `allocateMemory()`

```
u32int allocateMemory (
    u32int size )
```

Allocate additional memory from the heap.

Allocates additional memory from the heap in a first-fit method.

@params size Amount of bytes to be allocated from the heap

Returns

Returns 0 upon success, -1 otherwise

6.66.1.2 freeMemory()

```
int freeMemory (
    void * addr )
```

Free a block of memory.

Frees a particular block of memory that was previously allocated. Searches for the block of memory, removes it from the allocated list and places it into the free list. If there are any adjacent blocks then merge.

@params *addr* Address of the block that will be free

Returns

Returns 0 upon success, -1 otherwise

6.66.1.3 initHeap()

```
int initHeap (
    u32int size )
```

Allocate all memory available for the MPX.

Allocates memory for both CMBC and LMCB. This will put a CMCB at the top of the heap and a LMCB at the bottom of the heap, both of type free. This method also initializes the free and allocated lists

Parameters

<i>size</i>	Size that will be allocated for the heap in bytes
-------------	---

Returns

Return 0 upon success, -1 otherwise

6.66.1.4 insertAMCB()

```
void insertAMCB (
    cmcb_s * mcb )
```

6.66.1.5 insertFMCB()

```
void insertFMCB (
    cmcb_s * mcb )
```


6.66.1.6 isEmpty()

```
int isEmpty (
    char * p )
```

6.66.1.7 removeAMCB()

```
void removeAMCB (
    cmcb_s * cmcb )
```

6.66.1.8 removeFMCB()

```
void removeFMCB (
    cmcb_s * cmcb )
```

6.66.1.9 showAllocated()

```
int showAllocated (
    char * discard )
```

Shows addresses and block size of all blocks in allocated list.

Traverses the allocated list and shows the addresses and the size of the block. Shown in the order of address.

6.66.1.10 showFree()

```
int showFree (
    char * p )
```

Shows the addresses and block size of all block in free list.

Traverses the free list and shows the addresses and the size of the block. Shown in the order of address.

6.66.2 Variable Documentation

6.66.2.1 allocated

`mcb_queue_s` allocated

6.66.2.2 amcb

```
mcb_queue_s* amcb = &allocated
```

Allocated Memory Control List.

6.66.2.3 fmcb

```
mcb_queue_s* fmcb = &free
```

Free Memory Control List.

6.66.2.4 free

```
mcb_queue_s free
```

6.66.2.5 start_addr

```
u32int start_addr
```

Start address of the heap.

6.67 /home/maximillian/Desktop/MAMA/term/memory_↵ management/mm.h File Reference

Classes

- struct `cmcb_s`
Complete Memory Control Block (CMBC)
- struct `mcb_queue_s`
"Master" controller of the MCB queue

Typedefs

- typedef struct `cmcb_s cmcb_s`
Complete Memory Control Block (CMBC)
- typedef struct `mcb_queue_s mcb_queue_s`
"Master" controller of the MCB queue

Enumerations

- enum `mcb_state_e` { `ALLOCATED` , `FREE` }

Indicates the type of CMCB.

Functions

- int `initHeap` (`u32int` size)
Allocate all memory available for the MPX.
- `u32int` `allocateMemory` (`u32int` size)
Allocate additional memory from the heap.
- int `freeMemory` (void *addr)
Free a block of memory.
- int `showAllocated` (char *)
Shows addresses and block size of all blocks in allocated list.
- int `showFree` (char *p)
Shows the addresses and block size of all block in free list.
- int `isEmpty` ()
Does the heap only contain free memory.
- void `removeFMCB` (`cmcb_s` *mcb)
- void `removeAMCB` (`cmcb_s` *mcb)
- void `insertAMCB` (`cmcb_s` *mcb)
- void `insertFMCB` (`cmcb_s` *mcb)

6.67.1 Typedef Documentation

6.67.1.1 `cmcb_s`

```
typedef struct cmcb_s cmcb_s
```

Complete Memory Control Block (CMBC)

6.67.1.2 `mcb_queue_s`

```
typedef struct mcb_queue_s mcb_queue_s
```

"Master" controller of the MCB queue

6.67.2 Enumeration Type Documentation

6.67.2.1 `mcb_state_e`

```
enum mcb_state_e
```

Indicates the type of CMCB.

Enumerator

ALLOCATED	Allocated CMCB.
FREE	Freed CMCB.

6.67.3 Function Documentation

6.67.3.1 allocateMemory()

```
u32int allocateMemory (
    u32int size )
```

Allocate additional memory from the heap.

Allocates additional memory from the heap in a first-fit method.

@params size Amount of bytes to be allocated from the heap

Returns

Returns 0 upon success, -1 otherwise

6.67.3.2 freeMemory()

```
int freeMemory (
    void * addr )
```

Free a block of memory.

Frees a particular block of memory that was previously allocated. Searches for the block of memory, removes it from the allocated list and places it into the free list. If there are any adjacent blocks then merge.

@params addr Address of the block that will be free

Returns

Returns 0 upon success, -1 otherwise

6.67.3.3 initHeap()

```
int initHeap (
    u32int size )
```

Allocate all memory available for the MPX.

Allocates memory for both CMCB and LMCB. This will put a CMCB at the top of the heap and a LMCB at the bottom of the heap, both of type free. This method also initializes the free and allocated lists

Parameters

<i>size</i>	Size that will be allocated for the heap in bytes
-------------	---

Returns

Return 0 upon success, -1 otherwise

6.67.3.4 insertAMCB()

```
void insertAMCB (
    cmcb_s * mcb )
```

6.67.3.5 insertFMCB()

```
void insertFMCB (
    cmcb_s * mcb )
```

6.67.3.6 isEmpty()

```
int isEmpty ( )
```

Does the heap only contain free memory.

Tells whether the heap contains only free memory (True) or not (false).

Returns

Returns 1 (True) if heap contains only free memory, Return 0 (False) if there is something within the heap.

6.67.3.7 removeAMCB()

```
void removeAMCB (
    cmcb_s * cmcb )
```

6.67.3.8 removeFMCB()

```
void removeFMCB (
    cmcb_s * mcb )
```

6.67.3.9 showAllocated()

```
int showAllocated (
    char * discard )
```

Shows addresses and block size of all blocks in allocated list.

Traverses the allocated list and shows the addresses and the size of the block. Shown in the order of address.

6.67.3.10 showFree()

```
int showFree (
    char * p )
```

Shows the addresses and block size of all block in free list.

Traverses the free list and shows the addresses and the size of the block. Shown in the order of address.

6.68 mm.h

[Go to the documentation of this file.](#)

```
1 #ifndef MM_H
2 #define MM_H
3
4 /*****
5 /***** Structures *****/
6 /*****
7
9 typedef enum {
11     ALLOCATED,
12
14     FREE
15 } mcb_state_e;
16
18 typedef struct cmcb_s { // This is 52 bytes long
20     mcb_state_e type;
21
23     u32int addr;
24
26     u32int size;
27
29     char name[32];
30
32     struct cmcb_s * next;
33
35     struct cmcb_s * prev;
36 } cmcb_s;
37
39 typedef struct mcb_queue_s {
41     cmcb_s * mcbq_head;
42
44     mcb_state_e mcb_queue_type;
45 } mcb_queue_s;
46
47 /*****
48 /***** Function Headers *****/
49 /*****/
```

```

50
63 int initHeap(u32int size);
64
75 u32int allocateMemory(u32int size);
76
89 int freeMemory(void * addr);
90
98 int showAllocated(char *);
99
107 int showFree(char * p);
108
118 int isEmpty();
119
120 void removeFMCB(cmcb_s * mcb);
121
122 void removeAMCB(cmcb_s * cmcb);
123
124 void insertAMCB(cmcb_s * mcb);
125
126 void insertFMCB(cmcb_s * mcb);
127
128 #endif

```

6.69 /home/maximillian/Desktop/MAMA/term/cmds/pcb.c File Reference

6.70 /home/maximillian/Desktop/MAMA/term/pcb/pcb.c File Reference

```

#include "pcb.h"
#include <modules/mpx_supt.h>
#include <include/string.h>
#include <lib/out.h>
#include <term/utills.h>
#include <term/args.h>
#include <term/dispatch/context.h>

```

Functions

- void `initPCB` ()
Initialize PCB Queue.
- `pcb_t` * `allocatePCB` ()
Allocate memory for a new PCB.
- int `freePCB` (`pcb_t` *pcb)
Free's memory associated with PCB.
- `pcb_t` * `setupPCB` (char *name, int process_class, int priority)
Creates a PCB.
- `pcb_t` * `findPCB` (char *name)
Searches for PCB.
- int `insertPCB` (`pcb_t` *pcb)
Insert PCB into queue.
- int `removePCB` (`pcb_t` *pcb)
Removes PCB from Queue.
- int `createPCB` (char *args)
Create a PCB.
- int `setPriority` (char *args)
Set a new priority to a PCB.
- int `showPCB` (char *args)

- *Show informatino of PCB.*
- int `showReady` (char *p)
Show PCBs in ready queue.
- int `showBlocked` (char *args)
Show PCBs in blocked queue.
- int `showAll` (char *args)
Show all PCBs.
- int `suspendPCB` (char *args)
Set PCB state to suspended.
- int `resumePCB` (char *args)
Set PCB state to resume.
- int `deletePCB` (char *args)
Delete PCB.
- int `blockPCB` (char *args)
Set PCB state to be blocked.
- int `unblockPCB` (char *name)
Set PCB state to unblocked.
- int `resumeAll` (char *p)
Resume all suspended processes.
- int `isSystemProcess` (char *name)
Checks whether specified process is a system process or not.

Variables

- `pcb_queue_t p_queue`
- `pcb_queue_t f_queue`
- `pcb_queue_t * priority_queue = &p_queue`
- `pcb_queue_t * fifo_queue = &f_queue`

6.70.1 Function Documentation

6.70.1.1 `allocatePCB()`

```
pcb_t * allocatePCB ( )
```

Allocate memory for a new PCB.

Allocates memory for a new PCB in the stack and performs actions to initialize PCB

Returns

Pointer to newly created PCB, NULL otherwise

6.70.1.2 `blockPCB()`

```
int blockPCB (
    char * name )
```

Set PCB state to be blocked.

Find the PCB name in queue and sets its state to blocked and reinserts it into the appropriate queue.

Parameters

<i>name</i>	Name of PCB to block
-------------	----------------------

6.70.1.3 createPCB()

```
int createPCB (
    char * user_input )
```

Create a PCB.

Creates a new, unique PCB in memory.

Parameters

<i>name</i>	Give name of the PCB
<i>process_class</i>	The type of process class that will be used
<i>priority</i>	Priority of the PCB

Returns

Returns 0 upon success, 1 upon error

Parse the user input

Error Handling

6.70.1.4 deletePCB()

```
int deletePCB (
    char * name )
```

Delete PCB.

Will remove a PCB from the appropriate queue and free all associated memory. Will find the PCB in the queue, unlink it and free it.

Parameters

<i>name</i>	Name of the PCB to delete
-------------	---------------------------

Returns

Return 0 upon success, 1 upon failure

6.70.1.5 findPCB()

```
pcb_t * findPCB (
    char * name )
```

Searches for PCB.

Given a PCB name, will search all queues for a process.

Parameters

<i>name</i>	Name of the PCB being searched
-------------	--------------------------------

Returns

Returns pointer to PCB upon success, NULL if PCB was not found

6.70.1.6 freePCB()

```
int freePCB (
    pcb_t * freed_pcb )
```

Free's memory associated with PCB.

Free's the memory associated with the PCB such as the stack and the PCB itself

Parameters

<i>freed_pcb</i>	Pointer to the PCB being freed
------------------	--------------------------------

Returns

Returns 1 upon success, 0 upon error

6.70.1.7 initPCB()

```
void initPCB ( )
```

Initialize PCB Queue.

Initialize the PCB queue's by assigning values for the two queues that exist. This method is called upon startup in the commhand

6.70.1.8 insertPCB()

```
int insertPCB (  
    pcb_t * pcb )
```

Insert PCB into queue.

Inserts a PCB into the appropriate queue

Parameters

<i>pcb</i>	Pointer to the PCB being inserted
------------	-----------------------------------

Returns

0 on success, 1 on error

6.70.1.9 isSystemProcess()

```
int isSystemProcess (
    char * name )
```

Checks whether specified process is a system process or not.

Checks if the user supplied process name is a system process or an application.

Parameters

<i>name</i>	Name of the process
-------------	---------------------

Returns

Returns 1 if the process is a system process, 0 if the process is an application

6.70.1.10 removePCB()

```
int removePCB (
    pcb_t * pcb )
```

Removes PCB from Queue.

Removes specified PCB from queue it is stored in.

Parameters

<i>pcb</i>	Pointer to the PCB being removed
------------	----------------------------------

Returns

Returns 1 upon success, 0 upon error

6.70.1.11 resumeAll()

```
int resumeAll (
    char * p )
```

Resume all suspended processes.

Iterates through READY queue and sets the state of the each PCB to READY

Parameters

<i>p</i>	Empty params
----------	--------------

Returns

Returns 0 upon success, -1 otherwise.

6.70.1.12 resumePCB()

```
int resumePCB (
    char * name )
```

Set PCB state to resume.

Places a PCB into a not suspended state and reinserts into the appropriate queue

Parameters

<i>name</i>	Name of PCB to resume
-------------	-----------------------

Returns

Returns 0 upon success, 1 upon error

6.70.1.13 setPriority()

```
int setPriority (
    char * args )
```

Set a new priority to a PCB.

Sets a PCB's priority and reinserts the process into the correct place in the correct queue

Parameters

<i>args</i>	Name of the PCB and new priority (PCB_NAME.PRIORITY)
-------------	--

Returns

Returns 0 upon success, 1 upon error

Parse the user input

Error Handling

6.70.1.14 setupPCB()

```
pcb_t * setupPCB (
    char * name,
    int process_class,
    int priority )
```

Creates a PCB.

Allocates and fill memory associated with the PCB being created. This is accomplished by calling [allocatePCB\(\)](#) to initialize the memory and the fills the data with the parameters.

Parameters

<i>name</i>	Name of the PCB
<i>process_class</i>	Type of process being created
<i>priority</i>	The priority of the PCB being created

Returns

Returns pointer to PCB upon success, NULL otherwise

6.70.1.15 showAll()

```
int showAll (
    char * args )
```

Show all PCBs.

Display information for each PCB in the ready and blocked queue. The information that is displayed is: Process Name, Class, State, Suspended Status, Priority.

@params args Empty params

Returns

Returns 0 upon success, 1 upon error

6.70.1.16 showBlocked()

```
int showBlocked (
    char * args )
```

Show PCBs in blocked queue.

Display information for each PCB in the blocked queue. The information that is displayed is: Process Name, Class, State, Suspended Status, Priority.

6.70.1.17 showPCB()

```
int showPCB (
    char * name )
```

Show informatino of PCB.

Display information of the PCB. The information that is displayed is: Process Name, Class, State, Suspended Status and Priority

Parameters

<i>name</i>	Name of PCB to have its information displayed
-------------	---

6.70.1.18 showReady()

```
int showReady (
    char * p )
```

Show PCBs in ready queue.

Display information for each PCB in the ready queue. The information that is displayed is: Process Name, Class, State, Suspended Status, Priority

Parameters

<i>p</i>	Empty parameters.
----------	-------------------

Returns

0 upon success, 1 upon failure

6.70.1.19 suspendPCB()

```
int suspendPCB (
    char * name )
```

Set PCB state to suspended.

Places a PCB state into suspended and reinserts into appropriate queue

Parameters

<i>name</i>	Name of PCB to suspend
-------------	------------------------

Returns

Returns 0 upon success, 1 upon error

6.70.1.20 unblockPCB()

```
int unblockPCB (  
    char * name )
```

Set PCB state to unblocked.

Sets PCB state into unblocked and reinserts it into the appropriate queue

Parameters

<i>name</i>	Name of the PCB to unblock
-------------	----------------------------

Returns

Returns 0 upon success, 1 upon error

6.70.2 Variable Documentation

6.70.2.1 f_queue

```
pcb_queue_t f_queue
```

6.70.2.2 fifo_queue

```
pcb_queue_t* fifo_queue = &f_queue
```


6.70.2.3 p_queue

`pcb_queue_t` p_queue

6.70.2.4 priority_queue

`pcb_queue_t*` priority_queue = `&p_queue`

6.71 /home/maximillian/Desktop/MAMA/term/pcb/pcb.h File Reference

Classes

- struct `pcb_t`
Process Control Block Structure.
- struct `pcb_node_t`
Individual PCB nodes. Each PCB is associated with one node.
- struct `pcb_queue`
"Master" controller of the PCB queue

Macros

- #define `MAX_STACK_SIZE` 1024
The maximum size the stack can be. May change.
- #define `MAX_PRIORITY` 9
Maximum priority a PCB can be given.
- #define `MIN_PRIORITY` 0
Minimum priority a PCB can be given.
- #define `MAX_NAME_SIZE` 32
Maximum name size that can be given to a pcb.

Typedefs

- typedef struct `pcb_node_t` `pcb_node_t`
Individual PCB nodes. Each PCB is associated with one node.
- typedef struct `pcb_queue` `pcb_queue_t`
"Master" controller of the PCB queue

Enumerations

- enum `pcb_queue_order_t` { `PRIORITY` , `FIFO` }
Type of Queue Ordering.
- enum `p_state_t` { `RUNNING` , `READY` , `BLOCKED` , `SUSPENDED_READY` , `SUSPENDED_BLOCKED` }
Types of process states.
- enum `p_protection_mode_t` { `DELETABLE` , `DELETABLE_WHEN_SUSPENDED` , `NOT_DELETABLE` }

Functions

- void `initPCB` ()
Initialize PCB Queue.
- `pcb_t * allocatePCB` ()
Allocate memory for a new PCB.
- int `freePCB` (`pcb_t *freed_pcb`)
Free's memory associated with PCB.
- `pcb_t * setupPCB` (char *name, int process_class, int priority)
Creates a PCB.
- `pcb_t * findPCB` (char *name)
Searches for PCB.
- int `insertPCB` (`pcb_t *pcb`)
Insert PCB into queue.
- int `removePCB` (`pcb_t *pcb`)
Removes PCB from Queue.
- int `createPCB` (char *user_input)
Create a PCB.
- int `deletePCB` (char *name)
Delete PCB.
- int `blockPCB` (char *name)
Set PCB state to be blocked.
- int `unblockPCB` (char *name)
Set PCB state to unblocked.
- int `suspendPCB` (char *name)
Set PCB state to suspended.
- int `resumePCB` (char *name)
Set PCB state to resume.
- int `setPriority` (char *args)
Set a new priority to a PCB.
- int `showPCB` (char *name)
Show informatino of PCB.
- int `showReady` (char *p)
Show PCBs in ready queue.
- int `showBlocked` (char *args)
Show PCBs in blocked queue.
- int `showAll` (char *args)
Show all PCBs.
- int `resumeAll` (char *p)
Resume all suspended processes.
- int `isSystemProcess` (char *name)
Checks whether specified process is a system process or not.

6.71.1 Macro Definition Documentation

6.71.1.1 MAX_NAME_SIZE

```
#define MAX_NAME_SIZE 32
```

Maximum name size that can be given to a pcb.

6.71.1.2 MAX_PRIORITY

```
#define MAX_PRIORITY 9
```

Maximum priority a PCB can be given.

6.71.1.3 MAX_STACK_SIZE

```
#define MAX_STACK_SIZE 1024
```

The maximum size the stack can be. May change.

6.71.1.4 MIN_PRIORITY

```
#define MIN_PRIORITY 0
```

Minimum priority a PCB can be given.

6.71.2 Typedef Documentation

6.71.2.1 pcb_node_t

```
typedef struct pcb_node_t pcb_node_t
```

Individual PCB nodes. Each PCB is associated with one node.

6.71.2.2 pcb_queue_t

```
typedef struct pcb_queue pcb_queue_t
```

"Master" controller of the PCB queue

6.71.3 Enumeration Type Documentation

6.71.3.1 p_protection_mode_t

```
enum p_protection_mode_t
```

Enumerator

DELETABLE	
DELETABLE_WHEN_SUSPENDED	
NOT_DELETABLE	

6.71.3.2 p_state_t

```
enum p_state_t
```

Types of process states.

Enumerator

RUNNING	Running State.
READY	Ready State.
BLOCKED	Blocked State.
SUSPENDED_READY	Suspended Ready State.
SUSPENDED_BLOCKED	Suspended Blocked State.

6.71.3.3 pcb_queue_order_t

```
enum pcb_queue_order_t
```

Type of Queue Ordering.

Enumerator

PRIORITY	Priority Queue (Ready)
FIFO	FIFO Queue (Blocked)

6.71.4 Function Documentation**6.71.4.1 allocatePCB()**

```
pcb_t * allocatePCB ( )
```

Allocate memory for a new PCB.

Allocates memory for a new PCB in the stack and performs actions to initialize PCB

Returns

Pointer to newly created PCB, NULL otherwise

6.71.4.2 blockPCB()

```
int blockPCB (
    char * name )
```

Set PCB state to be blocked.

Find the PCB name in queue and sets its state to blocked and reinserts it into the appropriate queue.

Parameters

<i>name</i>	Name of PCB to block
-------------	----------------------

6.71.4.3 createPCB()

```
int createPCB (
    char * user_input )
```

Create a PCB.

Creates a new, unique PCB in memory.

Parameters

<i>name</i>	Give name of the PCB
<i>process_class</i>	The type of process class that will be used
<i>priority</i>	Priority of the PCB

Returns

Returns 0 upon success, 1 upon error

Parse the user input

Error Handling

6.71.4.4 deletePCB()

```
int deletePCB (
    char * name )
```

Delete PCB.

Will remove a PCB from the appropriate queue and free all associated memory. Will find the PCB in the queue, unlink it and free it.

Parameters

<i>name</i>	Name of the PCB to delete
-------------	---------------------------

Returns

Return 0 upon success, 1 upon failure

6.71.4.5 findPCB()

```
pcb_t * findPCB (
    char * name )
```

Searches for PCB.

Given a PCB name, will search all queues for a process.

Parameters

<i>name</i>	Name of the PCB being searched
-------------	--------------------------------

Returns

Returns pointer to PCB upon success, NULL if PCB was not found

6.71.4.6 freePCB()

```
int freePCB (
    pcb_t * freed_pcb )
```

Free's memory associated with PCB.

Free's the memory associated with the PCB such as the stack and the PCB itself

Parameters

<i>freed_pcb</i>	Pointer to the PCB being freed
------------------	--------------------------------

Returns

Returns 1 upon success, 0 upon error

6.71.4.7 initPCB()

```
void initPCB ( )
```

Initialize PCB Queue.

Initialize the PCB queue's by assigning values for the two queues that exist. This method is called upon startup in the commhand

6.71.4.8 insertPCB()

```
int insertPCB (
    pcb_t * pcb )
```

Insert PCB into queue.

Inserts a PCB into the appropriate queue

Parameters

<i>pcb</i>	Pointer to the PCB being inserted
------------	-----------------------------------

Returns

0 on success, 1 on error

6.71.4.9 isSystemProcess()

```
int isSystemProcess (
    char * name )
```

Checks whether specified process is a system process or not.

Checks if the user supplied process name is a system process or an application.

Parameters

<i>name</i>	Name of the process
-------------	---------------------

Returns

Returns 1 if the process is a system process, 0 if the process is an application

6.71.4.10 removePCB()

```
int removePCB (
    pcb_t * pcb )
```

Removes PCB from Queue.

Removes specified PCB from queue it is stored in.

Parameters

<i>pcb</i>	Pointer to the PCB being removed
------------	----------------------------------

Returns

Returns 1 upon success, 0 upon error

6.71.4.11 resumeAll()

```
int resumeAll (
    char * p )
```

Resume all suspended processes.

Iterates through READY queue and sets the state of the each PCB to READY

Parameters

<i>p</i>	Empty params
----------	--------------

Returns

Returns 0 upon success, -1 otherwise.

6.71.4.12 resumePCB()

```
int resumePCB (
    char * name )
```

Set PCB state to resume.

Places a PCB into a not suspended state and reinserts into the appropriate queue

Parameters

<i>name</i>	Name of PCB to resume
-------------	-----------------------

Returns

Returns 0 upon success, 1 upon error

6.71.4.13 setPriority()

```
int setPriority (
    char * args )
```

Set a new priority to a PCB.

Sets a PCB's priority and reinserts the process into the correct place in the correct queue

Parameters

<i>args</i>	Name of the PCB and new priority (PCB_NAME.PRIORITY)
-------------	--

Returns

Returns 0 upon success, 1 upon error

Parse the user input

Error Handling

6.71.4.14 setupPCB()

```
pcb_t * setupPCB (
    char * name,
    int process_class,
    int priority )
```

Creates a PCB.

Allocates and fill memory associated with the PCB being created. This is accomplished by calling [allocatePCB\(\)](#) to initialize the memory and the fills the data with the parameters.

Parameters

<i>name</i>	Name of the PCB
<i>process_class</i>	Type of process being created
<i>priority</i>	The priority of the PCB being created

Returns

Returns pointer to PCB upon success, NULL otherwise

6.71.4.15 showAll()

```
int showAll (
    char * args )
```

Show all PCBs.

Display information for each PCB in the ready and blocked queue. The information that is displayed is: Process Name, Class, State, Suspended Status, Priority.

@params args Empty params

Returns

Returns 0 upon success, 1 upon error

6.71.4.16 showBlocked()

```
int showBlocked (
    char * args )
```

Show PCBs in blocked queue.

Display information for each PCB in the blocked queue. The information that is displayed is: Process Name, Class, State, Suspended Status, Priority.

6.71.4.17 showPCB()

```
int showPCB (
    char * name )
```

Show informatino of PCB.

Display information of the PCB. The information that is displayed is: Process Name, Class, State, Suspended Status and Priority

Parameters

<i>name</i>	Name of PCB to have its information displayed
-------------	---

6.71.4.18 showReady()

```
int showReady (
    char * p )
```

Show PCBs in ready queue.

Display information for each PCB in the ready queue. The information that is displayed is: Process Name, Class, State, Suspended Status, Priority

Parameters

<i>p</i>	Empty parameters.
----------	-------------------

Returns

0 upon success, 1 upon failure

6.71.4.19 suspendPCB()

```
int suspendPCB (
    char * name )
```

Set PCB state to suspended.

Places a PCB state into suspended and reinserts into appropriate queue

Parameters

<i>name</i>	Name of PCB to suspend
-------------	------------------------

Returns

Returns 0 upon success, 1 upon error

6.71.4.20 unblockPCB()

```
int unblockPCB (
    char * name )
```

Set PCB state to unblocked.

Sets PCB state into unblocked and reinserts it into the appropriate queue

Parameters

<i>name</i>	Name of the PCB to unblock
-------------	----------------------------

Returns

Returns 0 upon success, 1 upon error

6.72 pcb.h

[Go to the documentation of this file.](#)

```

1  #ifndef PCB_H
2  #define PCB_H
3
4  #define MAX_STACK_SIZE 1024
5
6  #define MAX_PRIORITY 9
7  #define MIN_PRIORITY 0
8
9  #define MAX_NAME_SIZE 32
10
11  /*
12   * ***** Structures *****
13   */
14
15  typedef enum {
16      PRIORITY,
17  } pcb_queue_order_t;
18
19  typedef enum {
20      RUNNING,
21
22      READY,
23
24      BLOCKED,
25
26      SUSPENDED_READY,
27
28      SUSPENDED_BLOCKED
29  } p_state_t;
30
31  typedef enum {
32      DELETABLE,
33      DELETABLE_WHEN_SUSPENDED,
34      NOT_DELETABLE
35  } p_protection_mode_t;
36
37  typedef struct {
38      char pcb_name[32];          // Can change size in the future
39
40      int pcb_process_class;      // I've decided that process class will be an int. SYS_PROCESS = 0,
41      APPLICATION = 1
42
43      int pcb_priority;
44
45      p_state_t pcb_process_state;
46
47      p_protection_mode_t pcb_protection_mode;
48
49      unsigned char * pcb_stack_top;
50
51      unsigned char * pcb_stack_bottom;
52  } pcb_t;
53
54  typedef struct pcb_node_t {
55      struct pcb_node_t *pcbn_next_pcb;
56
57      struct pcb_node_t *pcbn_prev_pcb;
58
59      pcb_t *pcb;
60  } pcb_node_t;
61
62  typedef struct pcb_queue {
63      int pcbq_count;
64
65      pcb_node_t *pcbq_head;
66
67      pcb_node_t *pcbq_tail;
68
69      pcb_queue_order_t queue_order;
70  } pcb_queue_t;

```

```

103
104 /*****
105 /***** Function Headers *****/
106 /*****
107
115 void initPCB();
116
126 pcb_t * allocatePCB();
127
137 int freePCB(pcb_t * freed_pcb);
138
153 pcb_t * setupPCB(char * name, int process_class, int priority);
154
165 pcb_t * findPCB(char * name);
166
176 int insertPCB(pcb_t * pcb);
177
188 int removePCB(pcb_t * pcb);
189
190
202 int createPCB(char * user_input);
203
216 int deletePCB(char * name);
217
227 int blockPCB(char * name);
228
239 int unblockPCB(char * name);
240
251 int suspendPCB(char * name);
252
263 int resumePCB(char * name);
264
275 int setPriority(char * args);
276
286 int showPCB(char * name);
287
299 int showReady(char * p);
300
309 int showBlocked(char * args);
310
322 int showAll(char * args);
323
324 /*****
325 /***** R4 Stuff Here *****/
326 /*****
327
339 int resumeAll(char * p);
340
341 /*****
342 /***** R6 Stuff Here *****/
343 /*****
344
356 int isSystemProcess(char * name);
357
358
359 #endif

```

6.73 /home/maximillian/Desktop/MAMA/term/syntax.c File Reference

```

#include "syntax.h"
#include "utils.h"

```

Functions

- int [changes_state](#) (char, enum [SyntaxState](#), enum [SyntaxState](#) *)
- enum [SyntaxState](#) [get_state](#) (char c, enum [SyntaxState](#) cur_state)

6.73.1 Function Documentation

6.73.1.1 changes_state()

```
int changes_state (
    char c,
    enum SyntaxState cur_state,
    enum SyntaxState * next_state )
```

6.73.1.2 get_state()

```
enum SyntaxState get_state (
    char c,
    enum SyntaxState cur_state )
```

6.74 /home/maximillian/Desktop/MAMA/term/syntax.h File Reference

Enumerations

- enum [SyntaxState](#) {
[CMD_NAME_OR_LEADING_WHITESPACE](#) , [CMD_NAME](#) , [PARAM_NAME](#) , [PARAM_VALUE](#) ,
[DOUBLE_QUOTE_STRING](#) , [DOUBLE_QUOTE_STRING_END_QUOTE](#) , [SINGLE_QUOTE_STRING](#) ,
[SINGLE_QUOTE_STRING_END_QUOTE](#) ,
[END_OF_INPUT](#) , [DEFAULT](#) }

Functions

- enum [SyntaxState](#) [get_state](#) (char, enum [SyntaxState](#))
- int [changes_state](#) (char, enum [SyntaxState](#), enum [SyntaxState](#) *)

6.74.1 Enumeration Type Documentation

6.74.1.1 SyntaxState

```
enum SyntaxState
```

Enumerator

CMD_NAME_OR_LEADING_WHITESPACE	
CMD_NAME	
PARAM_NAME	
PARAM_VALUE	
DOUBLE_QUOTE_STRING	
DOUBLE_QUOTE_STRING_END_QUOTE	
SINGLE_QUOTE_STRING	
SINGLE_QUOTE_STRING_END_QUOTE	
END_OF_INPUT	
DEFAULT	

6.74.2 Function Documentation

6.74.2.1 changes_state()

```
int changes_state (
    char c,
    enum SyntaxState,
    enum SyntaxState * next_state )
```

6.74.2.2 get_state()

```
enum SyntaxState get_state (
    char c,
    enum SyntaxState )
```

6.75 syntax.h

[Go to the documentation of this file.](#)

```
1 #ifndef SYNTAX_H
2 #define SYNTAX_H
3
4 enum SyntaxState {
5     CMD_NAME_OR_LEADING_WHITESPACE,
6     CMD_NAME,
7     PARAM_NAME,
8     PARAM_VALUE,
9     DOUBLE_QUOTE_STRING,
10    DOUBLE_QUOTE_STRING_END_QUOTE,
11    SINGLE_QUOTE_STRING,
12    SINGLE_QUOTE_STRING_END_QUOTE,
13    END_OF_INPUT,
14    DEFAULT
15 };
16
17 enum SyntaxState get_state(char, enum SyntaxState);
18 int changes_state(char, enum SyntaxState, enum SyntaxState *);
19
20 #endif
```

6.76 /home/maximillian/Desktop/MAMA/term/utils.c File Reference

```
#include <include/string.h>
```

Functions

- int [is_name_char](#) (char c)
Returns whether or not the specified character is a valid character in an identifier, such as a command or argument name.
- void [skip_ws](#) (char **c)
Moves the specified pointer to a character buffer forward until it points to the next non-whitespace character.

6.76.1 Function Documentation

6.76.1.1 is_name_char()

```
int is_name_char (
    char c )
```

Returns whether or not the specified character is a valid character in an identifier, such as a command or argument name.

Parameters

<i>c</i>	The character to test.
----------	------------------------

Returns

True if the specified character *c* is valid in an identifier, false otherwise.

6.76.1.2 skip_ws()

```
void skip_ws (
    char ** c )
```

Moves the specified pointer to a character buffer forward until it points to the next non-whitespace character.

Parameters

<i>c</i>	A pointer to a pointer to an entry in a character buffer. Will be modified to point to the next non-whitespace character in the buffer.
----------	---

6.77 /home/maximillian/Desktop/MAMA/term/utils.h File Reference

Functions

- int [is_name_char](#) (char)
Returns whether or not the specified character is a valid character in an identifier, such as a command or argument name.
- void [skip_ws](#) (char **)
Moves the specified pointer to a character buffer forward until it points to the next non-whitespace character.

6.77.1 Function Documentation

6.77.1.1 is_name_char()

```
int is_name_char (
    char c )
```

Returns whether or not the specified character is a valid character in an identifier, such as a command or argument name.

Parameters

<code>c</code>	The character to test.
----------------	------------------------

Returns

True if the specified character `c` is valid in an identifier, false otherwise.

6.77.1.2 skip_ws()

```
void skip_ws (
    char ** c )
```

Moves the specified pointer to a character buffer forward until it points to the next non-whitespace character.

Parameters

<code>c</code>	A pointer to a pointer to an entry in a character buffer. Will be modified to point to the next non-whitespace character in the buffer.
----------------	---

6.78 utils.h

[Go to the documentation of this file.](#)

```
1 #ifndef UTILS_H
2 #define UTILS_H
3
4 int is_name_char(char);
5 void skip_ws(char **);
6
7 #endif
```

6.79 /home/maximillian/Desktop/MAMA/term/cmds/clear.c File Reference

```
#include <term/visuals/cursor.h>
#include <term/visuals/clear.h>
```

Functions

- int `cmd_clear` (char *args)

6.79.1 Function Documentation

6.79.1.1 cmd_clear()

```
int cmd_clear (
    char * args )
```

6.80 /home/maximillian/Desktop/MAMA/term/visuals/clear.c File Reference

```
#include <lib/out.h>
```

Functions

- void [display_clear](#) ()

6.80.1 Function Documentation

6.80.1.1 display_clear()

```
void display_clear ( )
```

6.81 /home/maximillian/Desktop/MAMA/term/visuals/clear.h File Reference

Functions

- void [display_clear](#) ()

6.81.1 Function Documentation

6.81.1.1 display_clear()

```
void display_clear ( )
```

6.82 clear.h

[Go to the documentation of this file.](#)

```
1 void display_clear();
```

6.83 /home/maximillian/Desktop/MAMA/term/visuals/colorize.c File Reference

```
#include <lib/out.h>
```

Macros

- #define `START_SEQ` `"\e["`

Enumerations

- enum `Color` {
 `BLACK`, `RED`, `GREEN`, `YELLOW`,
 `BLUE`, `MAGENTA`, `CYAN`, `WHITE`,
 `BLACK`, `RED`, `GREEN`, `YELLOW`,
 `BLUE`, `MAGENTA`, `CYAN`, `WHITE` }

Functions

- void `print_color_code` (enum `Color` color)
Description: Prints part of the escape sequence needed to switch the foreground or background color to the specified color.
- void `display_fg_color` (enum `Color` color)
Switches the text color in the terminal so that subsequent text written to the screen will be the specified color.
- void `display_bg_color` (enum `Color` color)
Switches the background color in the terminal so that subsequent text written to the screen will appear on a backdrop of the specified color.
- void `display_reset` ()
Resets any formatting so that subsequent text written to the screen will use the default appearance.
- void `display_italicize` ()
Description: Causes subsequent text written to the screen to be displayed in italics.

6.83.1 Macro Definition Documentation

6.83.1.1 START_SEQ

```
#define START_SEQ "\e["
```

6.83.2 Enumeration Type Documentation

6.83.2.1 Color

enum `Color`

Enumerator

BLACK	
RED	
GREEN	
YELLOW	
BLUE	
MAGENTA	
CYAN	
WHITE	
BLACK	
RED	
GREEN	
YELLOW	
BLUE	
MAGENTA	
CYAN	
WHITE	

6.83.3 Function Documentation

6.83.3.1 display_bg_color()

```
void display_bg_color (
    enum Color color )
```

Switches the background color in the terminal so that subsequent text written to the screen will appear on a backdrop of the specified color.

Parameters

<i>color</i>	The color to switch to.
--------------	-------------------------

6.83.3.2 display_fg_color()

```
void display_fg_color (
    enum Color color )
```

Switches the text color in the terminal so that subsequent text written to the screen will be the specified color.

Parameters

<i>color</i>	The color to switch to.
--------------	-------------------------

6.83.3.3 display_italicize()

```
void display_italicize ( )
```

Description: Causes subsequent text written to the screen to be displayed in italics.

6.83.3.4 display_reset()

```
void display_reset ( )
```

Resets any formatting so that subsequent text written to the screen will use the default appearance.

6.83.3.5 print_color_code()

```
void print_color_code (
    enum Color color )
```

Description: Prints part of the escape sequence needed to switch the foreground or background color to the specified color.

Used internally by display_fg_color and display_bg_color.

Parameters

<i>color</i>	The color being switched to.
--------------	------------------------------

6.84 /home/maximillian/Desktop/MAMA/term/visuals/colorize.h File Reference

Enumerations

- enum Color {
BLACK, RED, GREEN, YELLOW,
BLUE, MAGENTA, CYAN, WHITE,
BLACK, RED, GREEN, YELLOW,
BLUE, MAGENTA, CYAN, WHITE }

Functions

- void display_fg_color (enum Color)

Switches the text color in the terminal so that subsequent text written to the screen will be the specified color.

- void `display_bg_color` (enum `Color`)

Switches the background color in the terminal so that subsequent text written to the screen will appear on a backdrop of the specified color.

- void `display_italicize` ()

Description: Causes subsequent text written to the screen to be displayed in italics.

- void `display_reset` ()

Resets any formatting so that subsequent text written to the screen will use the default appearance.

6.84.1 Enumeration Type Documentation

6.84.1.1 Color

enum `Color`

Enumerator

BLACK	
RED	
GREEN	
YELLOW	
BLUE	
MAGENTA	
CYAN	
WHITE	
BLACK	
RED	
GREEN	
YELLOW	
BLUE	
MAGENTA	
CYAN	
WHITE	

6.84.2 Function Documentation

6.84.2.1 `display_bg_color()`

```
void display_bg_color (
    enum Color color )
```

Switches the background color in the terminal so that subsequent text written to the screen will appear on a backdrop of the specified color.

Parameters

<code>color</code>	The color to switch to.
--------------------	-------------------------

6.84.2.2 `display_fg_color()`

```
void display_fg_color (
    enum Color color )
```

Switches the text color in the terminal so that subsequent text written to the screen will be the specified color.

Parameters

<code>color</code>	The color to switch to.
--------------------	-------------------------

6.84.2.3 `display_italicize()`

```
void display_italicize ( )
```

Description: Causes subsequent text written to the screen to be displayed in italics.

6.84.2.4 `display_reset()`

```
void display_reset ( )
```

Resets any formatting so that subsequent text written to the screen will use the default appearance.

6.85 `colorize.h`

[Go to the documentation of this file.](#)

```
1 #ifndef COLORIZE_H
2 #define COLORIZE_H
3
4 enum Color {
5     BLACK,
6     RED,
7     GREEN,
8     YELLOW,
9     BLUE,
10    MAGENTA,
11    CYAN,
12    WHITE
13 };
14
15 void display_fg_color(enum Color);
16 void display_bg_color(enum Color);
17 void display_italicize();
18 void display_reset();
19
20 #endif
```

6.86 /home/maximillian/Desktop/MAMA/term/visuals/cursor.c File Reference

```
#include <lib/out.h>
```

Functions

- void `cursor_left` (int steps)
Moves the visual cursor to the left a specified number of steps.
- void `cursor_right` (int steps)
Moves the visual cursor to the right a specified number of steps.
- void `cursor_down` (int steps)
Moves the visual cursor down a specified number of steps.
- void `cursor_up` (int steps)
Moves the visual cursor up a specified number of steps.
- void `cursor_return` ()
Moves the visual cursor to the beginning of the line.

6.86.1 Function Documentation

6.86.1.1 `cursor_down()`

```
void cursor_down (  
    int steps )
```

Moves the visual cursor down a specified number of steps.

Parameters

<i>steps</i>	The number of steps to move the cursor down.
--------------	--

6.86.1.2 `cursor_left()`

```
void cursor_left (  
    int steps )
```

Moves the visual cursor to the left a specified number of steps.

Parameters

<i>steps</i>	The number of steps to move the cursor to the left.
--------------	---

6.86.1.3 `cursor_return()`

```
void cursor_return ( )
```

Moves the visual cursor to the beginning of the line.

6.86.1.4 `cursor_right()`

```
void cursor_right (
    int steps )
```

Moves the visual cursor to the right a specified number of steps.

Parameters

<i>steps</i>	The number of steps to move the cursor to the right.
--------------	--

6.86.1.5 `cursor_up()`

```
void cursor_up (
    int steps )
```

Moves the visual cursor up a specified number of steps.

Parameters

<i>steps</i>	The number of steps to move the cursor up.
--------------	--

6.87 /home/maximillian/Desktop/MAMA/term/visuals/cursor.h File Reference

Functions

- void `cursor_left` (int)
Moves the visual cursor to the left a specified number of steps.
- void `cursor_right` (int)
Moves the visual cursor to the right a specified number of steps.
- void `cursor_up` (int)
Moves the visual cursor up a specified number of steps.

- void `cursor_down` (int)
Moves the visual cursor down a specified number of steps.
- void `cursor_return` ()
Moves the visual cursor to the beginning of the line.

6.87.1 Function Documentation

6.87.1.1 `cursor_down()`

```
void cursor_down (  
    int steps )
```

Moves the visual cursor down a specified number of steps.

Parameters

<i>steps</i>	The number of steps to move the cursor down.
--------------	--

6.87.1.2 `cursor_left()`

```
void cursor_left (  
    int steps )
```

Moves the visual cursor to the left a specified number of steps.

Parameters

<i>steps</i>	The number of steps to move the cursor to the left.
--------------	---

6.87.1.3 `cursor_return()`

```
void cursor_return ( )
```

Moves the visual cursor to the beginning of the line.

6.87.1.4 `cursor_right()`

```
void cursor_right (  
    int steps )
```

Moves the visual cursor to the right a specified number of steps.

Parameters

<i>steps</i>	The number of steps to move the cursor to the right.
--------------	--

6.87.1.5 cursor_up()

```
void cursor_up (
    int steps )
```

Moves the visual cursor up a specified number of steps.

Parameters

<i>steps</i>	The number of steps to move the cursor up.
--------------	--

6.88 cursor.h

[Go to the documentation of this file.](#)

```
1 #ifndef CURSOR_H
2 #define CURSOR_H
3
4 void cursor_left(int);
5 void cursor_right(int);
6 void cursor_up(int);
7 void cursor_down(int);
8 void cursor_return();
9
10 #endif
```

6.89 /home/maximillian/Desktop/MAMA/term/visuals/hints.c File Reference

```
#include <lib/out.h>
#include "cursor.h"
```

Functions

- void [hint_under_prompt](#) (char *str, int len, int ret_index)
Writes a line of text under the user's prompt in the terminal.

6.89.1 Function Documentation

6.89.1.1 `hint_under_prompt()`

```
void hint_under_prompt (
    char * str,
    int len,
    int ret_index )
```

Writes a line of text under the user's prompt in the terminal.

Recommended for providing hints or warnings to the user as they type.

Parameters

<i>str</i>	The text to write under the user's prompt.
<i>len</i>	The length of the text to write under the user's prompt.
<i>ret_index</i>	The position to return the user's cursor to after writing the text.

6.90 `/home/maximillian/Desktop/MAMA/term/visuals/hints.h` File Reference

Functions

- void [hint_under_prompt](#) (char *, int, int)
Writes a line of text under the user's prompt in the terminal.

6.90.1 Function Documentation

6.90.1.1 `hint_under_prompt()`

```
void hint_under_prompt (
    char * str,
    int len,
    int ret_index )
```

Writes a line of text under the user's prompt in the terminal.

Recommended for providing hints or warnings to the user as they type.

Parameters

<i>str</i>	The text to write under the user's prompt.
<i>len</i>	The length of the text to write under the user's prompt.
<i>ret_index</i>	The position to return the user's cursor to after writing the text.

6.91 hints.h

[Go to the documentation of this file.](#)

```
1 #ifndef HINTS_H
2 #define HINTS_H
3
4 void hint_under_prompt(char *, int, int);
5
6 #endif
```

6.92 /home/maximillian/Desktop/MAMA/term/visuals/syntax_highlight.c

File Reference

```
#include "../syntax.h"
#include "../syntax.c"
#include "syntax_highlight.h"
#include "../commhand.h"
#include "colorize.h"
#include "hints.c"
#include "../utils.c"
#include <include/string.h>
```

Functions

- void [switch_to](#) (enum [SyntaxState](#), int, int)
Whether or not syntax highlighting is enabled as the user types.
- void [color_for](#) (enum [SyntaxState](#) state)
Prints the ANSI color code for the specified syntax state.
- void [get_state_at](#) (int index, int *index_of_state_in_record)
Retrieves the index in the states and switch_indexes data structures corresponding to the specified cursor index.
- void [syntax_init](#) ()
Initializes internal data structures needed for syntax highlighting.
- void [syntax_enable_highlighting](#) ()
Enables syntax highlighting as the user types.
- void [syntax_disable_highlighting](#) ()
Disables syntax highlighting as the user types.
- void [syntax_handle_char](#) (char c, int index)
Adjusts the terminal color assuming the specified character will immediately be written to the screen at the specified index.

Variables

- enum [SyntaxState](#) states [[MAX_SYNTAX_SWITCHES](#)]
- int [switch_indexes](#) [[MAX_SYNTAX_SWITCHES](#)]
Array of all the states the cursor has been in as the user has typed. Entries correspond to entries in switch_indexes.
- int [newest_switch](#)
Array of indexes the cursor was at when the corresponding syntax state in states was switched to.
- int [enabled](#) = 0
The largest and most recent valid index in states and switch_indexes.

6.92.1 Function Documentation

6.92.1.1 color_for()

```
void color_for (
    enum SyntaxState state )
```

Prints the ANSI color code for the specified syntax state.

Used internally by syntax_handle_char.

Parameters

<i>state</i>	The syntax state for which to print the correct color code to the terminal for.
--------------	---

6.92.1.2 get_state_at()

```
void get_state_at (
    int index,
    int * index_of_state_in_record )
```

Retrieves the index in the states and switch_indexes data structures corresponding to the specified cursor index.

Used internally by syntax_handle_char.

Parameters

<i>index</i>	The index of the cursor.
<i>index_of_state_in_record</i>	A pointer to the index in the states and switch_indexes data structures corresponding to the specified cursor index. Will be updated to point to the correct index in the data structures.

6.92.1.3 switch_to()

```
void switch_to (
    enum SyntaxState state,
    int index,
    int record_index )
```

Whether or not syntax highlighting is enabled as the user types.

Switches to the specified syntax state.

Used internally by syntax_handle_char.

Parameters

<i>state</i>	The syntax state being switched to.
<i>index</i>	The index in the user's input at which this switch occurs.
<i>record_index</i>	The index in the internal data structures states and switch_indexes at which to write this switch to.

6.92.1.4 syntax_disable_highlighting()

```
void syntax_disable_highlighting ( )
```

Disables syntax highlighting as the user types.

6.92.1.5 syntax_enable_highlighting()

```
void syntax_enable_highlighting ( )
```

Enables syntax highlighting as the user types.

6.92.1.6 syntax_handle_char()

```
void syntax_handle_char (
    char c,
    int index )
```

Adjusts the terminal color assuming the specified character will immediately be written to the screen at the specified index.

Parameters

<i>c</i>	The next character that will be output to the screen.
<i>index</i>	The index of the cursor.

6.92.1.7 syntax_init()

```
void syntax_init ( )
```

Initializes internal data structures needed for syntax highlighting.

6.92.2 Variable Documentation

6.92.2.1 enabled

```
int enabled = 0
```

The largest and most recent valid index in states and switch_indexes.

6.92.2.2 newest_switch

```
int newest_switch
```

Array of indexes the cursor was at when the corresponding syntax state in states was switched to.

6.92.2.3 states

```
enum SyntaxState states[MAX\_SYNTAX\_SWITCHES]
```

6.92.2.4 switch_indexes

```
int switch_indexes[MAX\_SYNTAX\_SWITCHES]
```

Array of all the states the cursor has been in as the user has typed. Entries correspond to entries in switch_indexes.

6.93 /home/maximillian/Desktop/MAMA/term/visuals/syntax_highlight.h File Reference

Macros

- `#define MAX_SYNTAX_SWITCHES 40`
- `#define SYNTAX_COLOR_CMD_NAME CYAN`
- `#define SYNTAX_COLOR_PARAM_NAME MAGENTA`
- `#define SYNTAX_COLOR_PARAM_VALUE WHITE`
- `#define SYNTAX_COLOR_DOUBLE_QUOTE_STRING YELLOW`
- `#define SYNTAX_COLOR_SINGLE_QUOTE_STRING YELLOW`
- `#define SYNTAX_COLOR_DEFAULT WHITE`

Functions

- void `syntax_init` ()
Initializes internal data structures needed for syntax highlighting.
- void `syntax_enable_highlighting` ()
Enables syntax highlighting as the user types.
- void `syntax_disable_highlighting` ()
Disables syntax highlighting as the user types.
- void `syntax_handle_char` (char, int)
Adjusts the terminal color assuming the specified character will immediately be written to the screen at the specified index.

6.93.1 Macro Definition Documentation

6.93.1.1 MAX_SYNTAX_SWITCHES

```
#define MAX_SYNTAX_SWITCHES 40
```

6.93.1.2 SYNTAX_COLOR_CMD_NAME

```
#define SYNTAX_COLOR_CMD_NAME CYAN
```

6.93.1.3 SYNTAX_COLOR_DEFAULT

```
#define SYNTAX_COLOR_DEFAULT WHITE
```

6.93.1.4 SYNTAX_COLOR_DOUBLE_QUOTE_STRING

```
#define SYNTAX_COLOR_DOUBLE_QUOTE_STRING YELLOW
```

6.93.1.5 SYNTAX_COLOR_PARAM_NAME

```
#define SYNTAX_COLOR_PARAM_NAME MAGENTA
```

6.93.1.6 SYNTAX_COLOR_PARAM_VALUE

```
#define SYNTAX_COLOR_PARAM_VALUE WHITE
```

6.93.1.7 SYNTAX_COLOR_SINGLE_QUOTE_STRING

```
#define SYNTAX_COLOR_SINGLE_QUOTE_STRING YELLOW
```

6.93.2 Function Documentation

6.93.2.1 syntax_disable_highlighting()

```
void syntax_disable_highlighting ( )
```

Disables syntax highlighting as the user types.

6.93.2.2 syntax_enable_highlighting()

```
void syntax_enable_highlighting ( )
```

Enables syntax highlighting as the user types.

6.93.2.3 syntax_handle_char()

```
void syntax_handle_char (
    char c,
    int index )
```

Adjusts the terminal color assuming the specified character will immediately be written to the screen at the specified index.

Parameters

<i>c</i>	The next character that will be output to the screen.
<i>index</i>	The index of the cursor.

6.93.2.4 syntax_init()

```
void syntax_init ( )
```

Initializes internal data structures needed for syntax highlighting.

6.94 syntax_highlight.h

[Go to the documentation of this file.](#)

```
1 #ifndef SYNTAX_HIGHLIGHT_H
2 #define SYNTAX_HIGHLIGHT_H
3
4 #define MAX_SYNTAX_SWITCHES 40
5
6 #define SYNTAX_COLOR_CMD_NAME CYAN
7 #define SYNTAX_COLOR_PARAM_NAME MAGENTA
8 #define SYNTAX_COLOR_PARAM_VALUE WHITE
9 #define SYNTAX_COLOR_DOUBLE_QUOTE_STRING YELLOW
10 #define SYNTAX_COLOR_SINGLE_QUOTE_STRING YELLOW
11 #define SYNTAX_COLOR_DEFAULT WHITE
12
13 void syntax_init();
14 void syntax_enable_highlighting();
15 void syntax_disable_highlighting();
16 void syntax_handle_char(char, int);
17
18 #endif
```

6.95 /home/maximillian/Desktop/MAMA/WhoDidWhat.md File Reference

Index

/home/maximillian/Desktop/MAMA/README.md, [98](#)
/home/maximillian/Desktop/MAMA/WhoDidWhat.md, [211](#)
/home/maximillian/Desktop/MAMA/help.c, [108](#)
/home/maximillian/Desktop/MAMA/include/core/asm.h, [35](#)
/home/maximillian/Desktop/MAMA/include/core/comhand.h, [35, 36](#)
/home/maximillian/Desktop/MAMA/include/core/interrupts.h, [36](#)
/home/maximillian/Desktop/MAMA/include/core/io.h, [37](#)
/home/maximillian/Desktop/MAMA/include/core/serial.h, [38, 40](#)
/home/maximillian/Desktop/MAMA/include/core/tables.h, [41, 44](#)
/home/maximillian/Desktop/MAMA/include/mem/heap.h, [44, 47](#)
/home/maximillian/Desktop/MAMA/include/mem/paging.h, [48, 50](#)
/home/maximillian/Desktop/MAMA/include/string.h, [51, 53](#)
/home/maximillian/Desktop/MAMA/include/system.h, [54, 57](#)
/home/maximillian/Desktop/MAMA/kernel/core/interrupts.c, [57](#)
/home/maximillian/Desktop/MAMA/kernel/core/kmain.c, [65](#)
/home/maximillian/Desktop/MAMA/kernel/core/serial.c, [66](#)
/home/maximillian/Desktop/MAMA/kernel/core/system.c, [69](#)
/home/maximillian/Desktop/MAMA/kernel/core/tables.c, [71](#)
/home/maximillian/Desktop/MAMA/kernel/mem/heap.c, [73](#)
/home/maximillian/Desktop/MAMA/kernel/mem/paging.c, [75](#)
/home/maximillian/Desktop/MAMA/lib/out.c, [78](#)
/home/maximillian/Desktop/MAMA/lib/out.h, [79, 87](#)
/home/maximillian/Desktop/MAMA/lib/string.c, [88](#)
/home/maximillian/Desktop/MAMA/modules/mpx_supt.c, [90](#)
/home/maximillian/Desktop/MAMA/modules/mpx_supt.h, [92, 97](#)
/home/maximillian/Desktop/MAMA/serial_driver/driver.c, [98](#)
/home/maximillian/Desktop/MAMA/term/args.c, [102](#)
/home/maximillian/Desktop/MAMA/term/args.h, [105, 106](#)
/home/maximillian/Desktop/MAMA/term/ascii/mama.c, [106](#)
/home/maximillian/Desktop/MAMA/term/ascii/mama.h, [106, 107](#)
/home/maximillian/Desktop/MAMA/term/cmds/argtest.c, [107](#)
/home/maximillian/Desktop/MAMA/term/cmds/clear.c, [192](#)
/home/maximillian/Desktop/MAMA/term/cmds/echo.c, [107](#)
/home/maximillian/Desktop/MAMA/term/cmds/help.c, [110](#)
/home/maximillian/Desktop/MAMA/term/cmds/pcb.c, [165](#)
/home/maximillian/Desktop/MAMA/term/cmds/shutdown.c, [117](#)
/home/maximillian/Desktop/MAMA/term/cmds/version.c, [117](#)
/home/maximillian/Desktop/MAMA/term/commands.h, [118](#)
/home/maximillian/Desktop/MAMA/term/commhand.c, [119](#)
/home/maximillian/Desktop/MAMA/term/commhand.h, [121, 123](#)
/home/maximillian/Desktop/MAMA/term/dispatch/context.c, [123](#)
/home/maximillian/Desktop/MAMA/term/dispatch/context.h, [125, 127](#)
/home/maximillian/Desktop/MAMA/term/dispatch/procsr3.c, [128](#)
/home/maximillian/Desktop/MAMA/term/dispatch/procsr3.h, [132, 133](#)
/home/maximillian/Desktop/MAMA/term/dnt/dnt.c, [133](#)
/home/maximillian/Desktop/MAMA/term/dnt/dnt.h, [140, 151](#)
/home/maximillian/Desktop/MAMA/term/history.c, [152](#)
/home/maximillian/Desktop/MAMA/term/history.h, [155, 156](#)
/home/maximillian/Desktop/MAMA/term/memory_management/mm.c, [156](#)
/home/maximillian/Desktop/MAMA/term/memory_management/mm.h, [160, 164](#)
/home/maximillian/Desktop/MAMA/term/pcb/pcb.c, [165](#)
/home/maximillian/Desktop/MAMA/term/pcb/pcb.h, [175, 187](#)
/home/maximillian/Desktop/MAMA/term/syntax.c, [188](#)
/home/maximillian/Desktop/MAMA/term/syntax.h, [189, 190](#)
/home/maximillian/Desktop/MAMA/term/utils.c, [190](#)

- /home/maximillian/Desktop/MAMA/term/utils.h, 191, args.c
 - 192
- /home/maximillian/Desktop/MAMA/term/visuals/clear.c,
 - 193
- /home/maximillian/Desktop/MAMA/term/visuals/clear.h,
 - 193, 194
- /home/maximillian/Desktop/MAMA/term/visuals/colorize.c,
 - 194
- /home/maximillian/Desktop/MAMA/term/visuals/colorize.h,
 - 197, 199
- /home/maximillian/Desktop/MAMA/term/visuals/cursor.c,
 - 200
- /home/maximillian/Desktop/MAMA/term/visuals/cursor.h,
 - 201, 203
- /home/maximillian/Desktop/MAMA/term/visuals/hints.c,
 - 203
- /home/maximillian/Desktop/MAMA/term/visuals/hints.h,
 - 204, 205
- /home/maximillian/Desktop/MAMA/term/visuals/syntax_highlight.c,
 - 205
- /home/maximillian/Desktop/MAMA/term/visuals/syntax_highlight.h,
 - 208, 211
- __attribute__
 - tables.h, 41
- __end
 - heap.c, 74
- _end
 - heap.c, 74
- _kmalloc
 - heap.c, 73
 - heap.h, 45
- access
 - gdt_entry_struct, 19
 - tables.h, 42
- accessed
 - page_entry, 26
- addr
 - cmcb_s, 9
- alarms
 - dnt.c, 140
- aliasHelp
 - help.c, 111
 - out.h, 81
- alloc
 - heap.c, 74
 - heap.h, 45
- ALLOCATED
 - mm.h, 162
- allocated
 - mm.c, 159
- allocateMemory
 - mm.c, 157
 - mm.h, 162
- allocatePCB
 - pcb.c, 166
 - pcb.h, 178
- amcb
 - mm.c, 159
- args.c
 - cur_state, 104
 - flag, 103
 - get_token, 103
 - last_state, 104
 - MAX_PARSE_STACK_SIZE, 103
 - named_arg, 103
 - next_unnamed_arg, 103
 - parse_args, 103
 - parse_stack, 104
 - stack_empty, 104
 - stack_peek, 104
 - stack_pop, 104
 - stack_push, 104
 - stack_size, 105
- args.h
 - parse_args, 105
 - parsed_args, 105
- argtest.c
 - cmd_argtest, 107
- asinh.h,
 - system.h, 54
- atoi
 - string.c, 88
 - string.h, 51
- BASE
 - driver.c, 99
- base
 - gdt_descriptor_struct, 18
 - heap, 21
 - idt_struct, 23
 - tables.h, 42
- base_high
 - gdt_entry_struct, 19
 - idt_entry_struct, 22
 - tables.h, 42
- base_low
 - gdt_entry_struct, 19
 - idt_entry_struct, 22
 - tables.h, 43
- base_mid
 - gdt_entry_struct, 19
 - tables.h, 43
- BCDtol
 - dnt.c, 134
 - dnt.h, 144
- BLACK
 - colorize.c, 196
 - colorize.h, 198
- block
 - index_entry, 23
- BLOCKED
 - pcb.h, 178
- blockHelp
 - help.c, 111
 - out.h, 81
- blockPCB
 - pcb.c, 166

- pcb.h, 179
- BLUE
 - colorize.c, 196
 - colorize.h, 198
- bounds
 - interrupts.c, 59
- breakpoint
 - interrupts.c, 59
- buffer_ptr
 - param, 28
- cdir
 - paging.c, 77
- changes_state
 - syntax.c, 188
 - syntax.h, 190
- circular_next_index
 - history.c, 152
- circular_prev_index
 - history.c, 153
- clear.c
 - cmd_clear, 193
 - display_clear, 193
- clear.h
 - display_clear, 193
- clear_bit
 - paging.c, 76
 - paging.h, 48
- clearHelp
 - help.c, 112
 - out.h, 81
- cli
 - system.h, 54
- CLOSED
 - driver.c, 101
- cmcb_s, 9
 - addr, 9
 - mm.h, 161
 - name, 10
 - next, 10
 - prev, 10
 - size, 10
 - type, 10
- cmd_alias
 - commhand.c, 120
- cmd_argtest
 - argtest.c, 107
- cmd_clear
 - clear.c, 193
- cmd_echo
 - echo.c, 108
- cmd_func_t
 - commhand.c, 119
- cmd_handler
 - cmd_mapping, 11
- cmd_help
 - help.c, 108, 112
 - out.h, 81
- cmd_mapping, 11
 - cmd_handler, 11
 - cmd_name, 11
 - commhand.c, 120
 - default_args, 11
- cmd_mappings
 - commhand.c, 121
- CMD_NAME
 - syntax.h, 189
- cmd_name
 - cmd_mapping, 11
- CMD_NAME_OR_LEADING_WHITESPACE
 - syntax.h, 189
- cmd_shutdown
 - shutdown.c, 117
- cmd_version
 - version.c, 118
- Color
 - colorize.c, 195
 - colorize.h, 198
- color_for
 - syntax_highlight.c, 206
- colorize.c
 - BLACK, 196
 - BLUE, 196
 - Color, 195
 - CYAN, 196
 - display_bg_color, 196
 - display_fg_color, 196
 - display_italicize, 197
 - display_reset, 197
 - GREEN, 196
 - MAGENTA, 196
 - print_color_code, 197
 - RED, 196
 - START_SEQ, 194
 - WHITE, 196
 - YELLOW, 196
- colorize.h
 - BLACK, 198
 - BLUE, 198
 - Color, 198
 - CYAN, 198
 - display_bg_color, 198
 - display_fg_color, 199
 - display_italicize, 199
 - display_reset, 199
 - GREEN, 198
 - MAGENTA, 198
 - RED, 198
 - WHITE, 198
 - YELLOW, 198
- COM1
 - serial.h, 38
- COM1_control_block
 - driver.c, 102
- COM2
 - serial.h, 38
- COM3

- serial.h, 38
- COM4
 - serial.h, 38
- com_close
 - driver.c, 101
- com_open
 - driver.c, 101
- COM_PORT
 - mpx_supt.h, 93
- com_read
 - driver.c, 101
- com_write
 - driver.c, 101
- comhand
 - comhand.h, 35
- comhand.h
 - comhand, 35
- commhand
 - commhand.c, 120
 - commhand.h, 123
- commhand.c
 - cmd_alias, 120
 - cmd_func_t, 119
 - cmd_mapping, 120
 - cmd_mappings, 121
 - commhand, 120
 - extract_cmd_name, 120
 - fetch_cmd_mapping, 120
 - is_name_char, 120
 - priority_queue, 121
- commhand.h
 - commhand, 123
 - MAX_CMD_ARG_NAME_LEN, 121
 - MAX_CMD_ARG_VALUE_LEN, 122
 - MAX_CMD_COUNT, 122
 - MAX_CMD_FLAG_COUNT, 122
 - MAX_CMD_HIST_LEN, 122
 - MAX_CMD_NAME_LEN, 122
 - MAX_CMD_NAMED_ARG_COUNT, 122
 - MAX_CMD_STRING_LEN, 122
 - MAX_CMD_UNNAMED_ARG_COUNT, 122
- consume_special
 - serial.c, 67
- context, 11
 - context.h, 125
 - cs, 12
 - ds, 12
 - eax, 12
 - ebp, 12
 - ebx, 13
 - ecx, 13
 - edi, 13
 - edx, 13
 - eflags, 13
 - eip, 13
 - es, 13
 - esi, 14
 - esp, 14
 - fs, 14
 - gs, 14
- context.c
 - dispatcher, 124
 - loadr3, 124
 - yield, 124
- context.h
 - context, 125
 - dispatcher, 126
 - loadr3, 127
 - yield, 127
- cop
 - system.c, 70
- coprocessor
 - interrupts.c, 60
- coprocessor_segment
 - interrupts.c, 60
- count_ptr
 - param, 28
- createPCB
 - pcb.c, 167
 - pcb.h, 179
- createpcbHelp
 - help.c, 112
 - out.h, 81
- cs
 - context, 12
- cur_state
 - args.c, 104
- curr_heap
 - heap.c, 74
- current_module
 - mpx_supt.c, 91
- current_time
 - dnt.c, 140
- currentTime
 - dnt.c, 134
 - dnt.h, 145
- cursor.c
 - cursor_down, 200
 - cursor_left, 200
 - cursor_return, 201
 - cursor_right, 201
 - cursor_up, 201
- cursor.h
 - cursor_down, 202
 - cursor_left, 202
 - cursor_return, 202
 - cursor_right, 202
 - cursor_up, 203
- cursor_down
 - cursor.c, 200
 - cursor.h, 202
- cursor_left
 - cursor.c, 200
 - cursor.h, 202
- cursor_return
 - cursor.c, 201

- cursor.h, 202
- cursor_right
 - cursor.c, 201
 - cursor.h, 202
- cursor_up
 - cursor.c, 201
 - cursor.h, 203
- CYAN
 - colorize.c, 196
 - colorize.h, 198
- date_time, 14
 - day_m, 15
 - day_w, 15
 - day_y, 15
 - hour, 15
 - min, 15
 - mon, 15
 - sec, 15
 - year, 15
- day_m
 - date_time, 15
- day_w
 - date_time, 15
- day_y
 - date_time, 15
- DAYS_IN_LEAP_YEAR
 - dnt.h, 142
- DAYS_IN_YEAR
 - dnt.h, 142
- daysInMonth
 - dnt.c, 134
 - dnt.h, 145
- dcb_t, 16
 - driver.c, 100
 - eflag_p, 16
 - oper_status, 16
 - ready_state, 16
 - ring_buffer, 16
 - ring_buffer_head, 17
 - ring_buffer_tail, 17
 - user_read_buf, 17
 - user_read_count, 17
 - user_write_buf, 17
 - user_write_count, 17
- debug
 - interrupts.c, 60
- DEFAULT
 - syntax.h, 189
- default_args
 - cmd_mapping, 11
- DEFAULT_DEVICE
 - mpx_supt.h, 93
- DELETABLE
 - pcb.h, 178
- DELETABLE_WHEN_SUSPENDED
 - pcb.h, 178
- DELETE
 - serial.c, 66
- deletePCB
 - pcb.c, 167
 - pcb.h, 179
- deletepcbHelp
 - help.c, 112
 - out.h, 82
- device_id
 - param, 28
- device_not_available
 - interrupts.c, 60
- device_ready_state_t
 - driver.c, 100
- device_status_t
 - driver.c, 101
- dirty
 - page_entry, 26
- dispatchAlarm
 - dnt.c, 135
 - dnt.h, 145
- dispatcher
 - context.c, 124
 - context.h, 126
- display_bg_color
 - colorize.c, 196
 - colorize.h, 198
- display_clear
 - clear.c, 193
 - clear.h, 193
- display_fg_color
 - colorize.c, 196
 - colorize.h, 199
- display_italicize
 - colorize.c, 197
 - colorize.h, 199
- display_reset
 - colorize.c, 197
 - colorize.h, 199
- divide_error
 - interrupts.c, 60
- DIVISOR_LATCH_HIGH_BYTE_REGISTER
 - driver.c, 99
- DIVISOR_LATCH_LOW_BYTE_REGISTER
 - driver.c, 99
- dnt.c
 - alarms, 140
 - BCDtol, 134
 - current_time, 140
 - currentTime, 134
 - daysInMonth, 134
 - dispatchAlarm, 135
 - freeAlarm, 135
 - getdate, 135
 - gettime, 136
 - intToDayOfWeek, 136
 - intToMonth, 136
 - ltoBCD, 137
 - messages, 140
 - setAlarm, 137

- setdate, 138
- setDateInMemory, 138
- settime, 139
- setTimeInMemory, 139
- showAlarms, 139
- dnt.h
 - BCDtol, 144
 - currentTime, 145
 - DAYS_IN_LEAP_YEAR, 142
 - DAYS_IN_YEAR, 142
 - daysInMonth, 145
 - dispatchAlarm, 145
 - EPOCH_FIRST_DAY_OF_WEEK_OF_YEAR, 142
 - EPOCH_FIRST_DAY_OF_YEAR, 142
 - EPOCH_FIRST_MONTH_OF_YEAR, 142
 - EPOCH_YEAR, 143
 - freeAlarm, 145
 - getdate, 146
 - gettime, 146
 - intToDayOfWeek, 147
 - intToMonth, 147
 - ltoBCD, 148
 - MAX_DAY, 143
 - MAX_HOURS, 143
 - MAX_MINUTES, 143
 - MAX_MONTH, 143
 - MAX_SECONDS, 143
 - MAX_YEAR, 144
 - MIN, 144
 - MIN_DAY, 144
 - MIN_MONTH, 144
 - MIN_YEAR, 144
 - setAlarm, 148
 - setdate, 149
 - setDateInMemory, 149
 - settime, 150
 - setTimeInMemory, 150
 - showAlarms, 151
- do_bounds
 - interrupts.c, 60
- do_breakpoint
 - interrupts.c, 60
- do_coprocessor
 - interrupts.c, 60
- do_coprocessor_segment
 - interrupts.c, 61
- do_debug
 - interrupts.c, 61
- do_device_not_available
 - interrupts.c, 61
- do_divide_error
 - interrupts.c, 61
- do_double_fault
 - interrupts.c, 61
- do_general_protection
 - interrupts.c, 61
- do_invalid_op
 - interrupts.c, 61
- do_invalid_tss
 - interrupts.c, 61
- do_isr
 - interrupts.c, 62
- do_nmi
 - interrupts.c, 62
- do_overflow
 - interrupts.c, 62
- do_page_fault
 - interrupts.c, 62
- do_reserved
 - interrupts.c, 62
- do_segment_not_present
 - interrupts.c, 62
- do_stack_segment
 - interrupts.c, 62
- double_fault
 - interrupts.c, 62
- DOUBLE_QUOTE_STRING
 - syntax.h, 189
- DOUBLE_QUOTE_STRING_END_QUOTE
 - syntax.h, 189
- DOWN_ARROW
 - serial.c, 66
- driver.c
 - BASE, 99
 - CLOSED, 101
 - COM1_control_block, 102
 - com_close, 101
 - com_open, 101
 - com_read, 101
 - com_write, 101
 - dcb_t, 100
 - device_ready_state_t, 100
 - device_status_t, 101
 - DIVISOR_LATCH_HIGH_BYTE_REGISTER, 99
 - DIVISOR_LATCH_LOW_BYTE_REGISTER, 99
 - IDLE, 101
 - INTERRUPT_ENABLE_REGISTER, 99
 - INTERRUPT_IDENTIFICATION_REGISTER, 99
 - LINE_CONTROL_REGISTER, 99
 - LINE_STATUS_REGISTER, 99
 - MODEM_CONTROL_REGISTER, 100
 - MODEM_STATUS_REGISTER, 100
 - OPEN, 101
 - PIC_MASK, 100
 - READING, 101
 - RING_BUFFER_SIZE, 100
 - SCRATCH_REGISTER, 100
 - WRITING, 101
- ds
 - context, 12
- eax
 - context, 12
- ebp
 - context, 12
- ebx
 - context, 13

- echo.c
 - cmd_echo, 108
- ecx
 - context, 13
- edi
 - context, 13
- edx
 - context, 13
- eflag_p
 - dcb_t, 16
- eflags
 - context, 13
- eip
 - context, 13
- empty
 - index_entry, 24
- enabled
 - syntax_highlight.c, 208
- end
 - heap.c, 75
- END_OF_INPUT
 - syntax.h, 189
- EPOCH_FIRST_DAY_OF_WEEK_OF_YEAR
 - dnt.h, 142
- EPOCH_FIRST_DAY_OF_YEAR
 - dnt.h, 142
- EPOCH_FIRST_MONTH_OF_YEAR
 - dnt.h, 142
- EPOCH_YEAR
 - dnt.h, 143
- er1
 - procsr3.c, 130
- er2
 - procsr3.c, 130
- er3
 - procsr3.c, 130
- er4
 - procsr3.c, 130
- er5
 - procsr3.c, 130
- erSize
 - procsr3.c, 130
- es
 - context, 13
- esi
 - context, 14
- esp
 - context, 14
- EXIT
 - mpx_supt.h, 93
- extract_cmd_name
 - commhand.c, 120
- f_queue
 - pcb.c, 174
- FALSE
 - mpx_supt.h, 93
- fetch_cmd_mapping
 - commhand.c, 120
- FIFO
 - pcb.h, 178
- fifo_queue
 - pcb.c, 174
- find_free
 - paging.c, 76
- findPCB
 - pcb.c, 167
 - pcb.h, 181
- first_free
 - paging.h, 48
- flag
 - args.c, 103
- flag_count
 - parsed_args, 29
- flags
 - gdt_entry_struct, 20
 - idt_entry_struct, 22
 - parsed_args, 29
 - tables.h, 43
- fmcb
 - mm.c, 160
- footer, 18
 - head, 18
- frameaddr
 - page_entry, 27
- frames
 - paging.c, 77
- FREE
 - mm.h, 162
- free
 - mm.c, 160
- freeAlarm
 - dnt.c, 135
 - dnt.h, 145
- freealarmHelp
 - help.c, 112
 - out.h, 82
- freeMemory
 - mm.c, 157
 - mm.h, 162
- freePCB
 - pcb.c, 168
 - pcb.h, 181
- fs
 - context, 14
- GDT_CS_ID
 - system.h, 54
- gdt_descriptor_struct, 18
 - base, 18
 - limit, 18
- GDT_DS_ID
 - system.h, 55
- gdt_entries
 - tables.c, 72
- gdt_entry_struct, 19
 - access, 19
 - base_high, 19

- base_low, 19
- base_mid, 19
- flags, 20
- limit_low, 20
- gdt_init_entry
 - tables.c, 71
 - tables.h, 41
- gdt_ptr
 - tables.c, 72
- general_protection
 - interrupts.c, 63
- get_bit
 - paging.c, 76
 - paging.h, 49
- get_page
 - paging.c, 76
 - paging.h, 49
- get_state
 - syntax.c, 189
 - syntax.h, 190
- get_state_at
 - syntax_highlight.c, 206
- get_token
 - args.c, 103
- getdate
 - dnt.c, 135
 - dnt.h, 146
- getdateHelp
 - help.c, 109, 113
 - out.h, 82
- gettime
 - dnt.c, 136
 - dnt.h, 146
- gettimeHelp
 - help.c, 109, 113
 - out.h, 82
- global_context
 - system.c, 70
- GREEN
 - colorize.c, 196
 - colorize.h, 198
- gs
 - context, 14
- head
 - footer, 18
- header, 20
 - index_id, 20
 - size, 20
- heap, 21
 - base, 21
 - index, 21
 - max_size, 21
 - min_size, 21
- heap.c
 - __end, 74
 - _end, 74
 - _kmalloc, 73
 - alloc, 74
 - curr_heap, 74
 - end, 75
 - kdir, 75
 - kheap, 75
 - kmalloc, 74
 - make_heap, 74
 - phys_alloc_addr, 75
- heap.h
 - _kmalloc, 45
 - alloc, 45
 - init_kheap, 46
 - kfree, 46
 - KHEAP_BASE, 45
 - KHEAP_MIN, 45
 - KHEAP_SIZE, 45
 - kmalloc, 46
 - make_heap, 46
 - TABLE_SIZE, 45
- help.c
 - aliasHelp, 111
 - blockHelp, 111
 - clearHelp, 112
 - cmd_help, 108, 112
 - createpcbHelp, 112
 - deletepcbHelp, 112
 - freealarmHelp, 112
 - getdateHelp, 109, 113
 - gettimeHelp, 109, 113
 - helpHelp, 109, 113
 - helpList, 109, 113
 - isemptyHelp, 113
 - loadr3Help, 113
 - resumeallHelp, 114
 - resumeHelp, 114
 - setalarmHelp, 114
 - setdateHelp, 109, 114
 - setpriorityHelp, 114
 - settimeHelp, 109, 114
 - showalarmsHelp, 115
 - showallocHelp, 115
 - showallpcbHelp, 115
 - showblockedpcbHelp, 115
 - showfreeHelp, 115
 - showpcbHelp, 115
 - showreadypcbHelp, 116
 - shutdownHelp, 110, 116
 - suspendHelp, 116
 - unblockHelp, 116
 - versionHelp, 116
 - versionOs, 110
- helpHelp
 - help.c, 109, 113
 - out.h, 82
- helpList
 - help.c, 109, 113
 - out.h, 82
- hint_under_prompt
 - hints.c, 203

- hints.h, 204
- hints.c
 - hint_under_prompt, 203
- hints.h
 - hint_under_prompt, 204
- hist_discard_last_frame
 - history.c, 153
- hist_forward
 - history.c, 153
 - history.h, 155
- hist_next_frame
 - history.c, 154
 - history.h, 156
- hist_rewind
 - history.c, 154
 - history.h, 156
- history.c
 - circular_next_index, 152
 - circular_prev_index, 153
 - hist_discard_last_frame, 153
 - hist_forward, 153
 - hist_next_frame, 154
 - hist_rewind, 154
 - write_hist_to_buf, 154
- history.h
 - hist_forward, 155
 - hist_next_frame, 156
 - hist_rewind, 156
- hlt
 - system.h, 55
- hour
 - date_time, 15
- ICW1
 - interrupts.c, 59
- ICW4
 - interrupts.c, 59
- id
 - index_table, 24
- IDLE
 - driver.c, 101
 - mpx_supt.h, 93
- idle
 - mpx_supt.c, 90
 - mpx_supt.h, 95
- idt_entries
 - interrupts.c, 65
 - tables.c, 73
- idt_entry_struct, 21
 - base_high, 22
 - base_low, 22
 - flags, 22
 - sselect, 22
 - zero, 22
- idt_ptr
 - tables.c, 73
- idt_set_gate
 - tables.c, 71
 - tables.h, 42
- idt_struct, 23
 - base, 23
 - limit, 23
- inb
 - io.h, 37
- index
 - heap, 21
- index_entry, 23
 - block, 23
 - empty, 24
 - size, 24
- index_id
 - header, 20
- index_table, 24
 - id, 24
 - table, 24
- init_gdt
 - tables.c, 72
 - tables.h, 42
- init_idt
 - tables.c, 72
 - tables.h, 42
- init_irq
 - interrupts.c, 63
 - interrupts.h, 36
- init_kheap
 - heap.h, 46
- init_paging
 - paging.c, 76
 - paging.h, 49
- init_pic
 - interrupts.c, 63
 - interrupts.h, 36
- init_serial
 - serial.c, 67
 - serial.h, 39
- initHeap
 - mm.c, 158
 - mm.h, 162
- initPCB
 - pcb.c, 168
 - pcb.h, 181
- insertAMCB
 - mm.c, 158
 - mm.h, 163
- insertFMCB
 - mm.c, 158
 - mm.h, 163
- insertPCB
 - pcb.c, 168
 - pcb.h, 182
- INTERRUPT_ENABLE_REGISTER
 - driver.c, 99
- INTERRUPT_IDENTIFICATION_REGISTER
 - driver.c, 99
- interrupts.c
 - bounds, 59
 - breakpoint, 59

- coprocessor, [60](#)
- coprocessor_segment, [60](#)
- debug, [60](#)
- device_not_available, [60](#)
- divide_error, [60](#)
- do_bounds, [60](#)
- do_breakpoint, [60](#)
- do_coprocessor, [60](#)
- do_coprocessor_segment, [61](#)
- do_debug, [61](#)
- do_device_not_available, [61](#)
- do_divide_error, [61](#)
- do_double_fault, [61](#)
- do_general_protection, [61](#)
- do_invalid_op, [61](#)
- do_invalid_tss, [61](#)
- do_isr, [62](#)
- do_nmi, [62](#)
- do_overflow, [62](#)
- do_page_fault, [62](#)
- do_reserved, [62](#)
- do_segment_not_present, [62](#)
- do_stack_segment, [62](#)
- double_fault, [62](#)
- general_protection, [63](#)
- ICW1, [59](#)
- ICW4, [59](#)
- idt_entries, [65](#)
- init_irq, [63](#)
- init_pic, [63](#)
- invalid_op, [63](#)
- invalid_tss, [63](#)
- io_wait, [59](#)
- isr0, [63](#)
- nmi, [63](#)
- overflow, [64](#)
- page_fault, [64](#)
- PIC1, [59](#)
- PIC2, [59](#)
- reserved, [64](#)
- rtc_isr, [64](#)
- segment_not_present, [64](#)
- stack_segment, [64](#)
- sys_call_isr, [64](#)
- interrupts.h
 - init_irq, [36](#)
 - init_pic, [36](#)
- intToDayOfWeek
 - dnt.c, [136](#)
 - dnt.h, [147](#)
- intToMonth
 - dnt.c, [136](#)
 - dnt.h, [147](#)
- INVALID_BUFFER
 - mpx_supt.h, [93](#)
- INVALID_COUNT
 - mpx_supt.h, [94](#)
- invalid_op
 - interrupts.c, [63](#)
- INVALID_OPERATION
 - mpx_supt.h, [94](#)
- invalid_tss
 - interrupts.c, [63](#)
- io.h
 - inb, [37](#)
 - outb, [37](#)
- IO_MODULE
 - mpx_supt.h, [94](#)
- io_wait
 - interrupts.c, [59](#)
- iret
 - system.h, [55](#)
- is_name_char
 - commhand.c, [120](#)
 - utils.c, [191](#)
 - utils.h, [191](#)
- isEmpty
 - mm.c, [158](#)
 - mm.h, [163](#)
- isemptyHelp
 - help.c, [113](#)
 - out.h, [83](#)
- isr0
 - interrupts.c, [63](#)
- isspace
 - string.c, [88](#)
 - string.h, [51](#)
- isSystemProcess
 - pcb.c, [170](#)
 - pcb.h, [182](#)
- itoa
 - string.c, [88](#)
 - string.h, [51](#)
- ltoBCD
 - dnt.c, [137](#)
 - dnt.h, [148](#)
- kdir
 - heap.c, [75](#)
 - paging.c, [77](#)
- kfree
 - heap.h, [46](#)
- kheap
 - heap.c, [75](#)
 - paging.c, [77](#)
- KHEAP_BASE
 - heap.h, [45](#)
- KHEAP_MIN
 - heap.h, [45](#)
- KHEAP_SIZE
 - heap.h, [45](#)
- klogv
 - system.c, [69](#)
 - system.h, [56](#)
- kmain
 - kmain.c, [65](#)
- kmain.c

- kmain, 65
- kmalloc
 - heap.c, 74
 - heap.h, 46
- kpanic
 - system.c, 69
 - system.h, 56
- last_state
 - args.c, 104
- LEFT_ARROW
 - serial.c, 67
- limit
 - gdt_descriptor_struct, 18
 - idt_struct, 23
 - tables.h, 43
- limit_low
 - gdt_entry_struct, 20
 - tables.h, 43
- LINE_CONTROL_REGISTER
 - driver.c, 99
- LINE_STATUS_REGISTER
 - driver.c, 99
- load_page_dir
 - paging.c, 76
 - paging.h, 49
- loadr3
 - context.c, 124
 - context.h, 127
- loadr3Help
 - help.c, 113
 - out.h, 83
- MAGENTA
 - colorize.c, 196
 - colorize.h, 198
- make_heap
 - heap.c, 74
 - heap.h, 46
- mama
 - mama.c, 106
 - mama.h, 107
- mama.c
 - mama, 106
- mama.h
 - mama, 107
- MAX_CMD_ARG_NAME_LEN
 - commhand.h, 121
- MAX_CMD_ARG_VALUE_LEN
 - commhand.h, 122
- MAX_CMD_COUNT
 - commhand.h, 122
- MAX_CMD_FLAG_COUNT
 - commhand.h, 122
- MAX_CMD_HIST_LEN
 - commhand.h, 122
- MAX_CMD_NAME_LEN
 - commhand.h, 122
- MAX_CMD_NAMED_ARG_COUNT
 - commhand.h, 122
- MAX_CMD_STRING_LEN
 - commhand.h, 122
- MAX_CMD_UNNAMED_ARG_COUNT
 - commhand.h, 122
- MAX_DAY
 - dnt.h, 143
- MAX_HOURS
 - dnt.h, 143
- MAX_MINUTES
 - dnt.h, 143
- MAX_MONTH
 - dnt.h, 143
- MAX_NAME_SIZE
 - pcb.h, 176
- MAX_PARSE_STACK_SIZE
 - args.c, 103
- MAX_PRIORITY
 - pcb.h, 177
- MAX_SECONDS
 - dnt.h, 143
- max_size
 - heap, 21
- MAX_STACK_SIZE
 - pcb.h, 177
- MAX_SYNTAX_SWITCHES
 - syntax_highlight.h, 209
- MAX_YEAR
 - dnt.h, 144
- mcb_queue_s, 25
 - mcb_queue_type, 25
 - mcbq_head, 25
 - mm.h, 161
- mcb_queue_type
 - mcb_queue_s, 25
- mcb_state_e
 - mm.h, 161
- mcbq_head
 - mcb_queue_s, 25
- MEM_MODULE
 - mpx_supt.h, 94
- mem_size
 - paging.c, 77
- memset
 - string.c, 89
 - string.h, 52
- messages
 - dnt.c, 140
- MIN
 - dnt.h, 144
- min
 - date_time, 15
- MIN_DAY
 - dnt.h, 144
- MIN_MONTH
 - dnt.h, 144
- MIN_PRIORITY
 - pcb.h, 177

- min_size
 - heap, 21
- MIN_YEAR
 - dnt.h, 144
- mm.c
 - allocated, 159
 - allocateMemory, 157
 - amcb, 159
 - fmcb, 160
 - free, 160
 - freeMemory, 157
 - initHeap, 158
 - insertAMCB, 158
 - insertFMCB, 158
 - isEmpty, 158
 - removeAMCB, 159
 - removeFMCB, 159
 - showAllocated, 159
 - showFree, 159
 - start_addr, 160
- mm.h
 - ALLOCATED, 162
 - allocateMemory, 162
 - cmcb_s, 161
 - FREE, 162
 - freeMemory, 162
 - initHeap, 162
 - insertAMCB, 163
 - insertFMCB, 163
 - isEmpty, 163
 - mcb_queue_s, 161
 - mcb_state_e, 161
 - removeAMCB, 163
 - removeFMCB, 163
 - showAllocated, 164
 - showFree, 164
- MODEM_CONTROL_REGISTER
 - driver.c, 100
- MODEM_STATUS_REGISTER
 - driver.c, 100
- MODULE_F
 - mpx_supt.h, 94
- MODULE_R1
 - mpx_supt.h, 94
- MODULE_R2
 - mpx_supt.h, 94
- MODULE_R3
 - mpx_supt.h, 94
- MODULE_R4
 - mpx_supt.h, 95
- MODULE_R5
 - mpx_supt.h, 95
- mon
 - date_time, 15
- mpx_init
 - mpx_supt.c, 90
 - mpx_supt.h, 95
- mpx_supt.c
 - current_module, 91
 - idle, 90
 - mpx_init, 90
 - params, 92
 - student_free, 92
 - student_malloc, 92
 - sys_alloc_mem, 91
 - sys_free_mem, 91
 - sys_req, 91
 - sys_set_free, 91
 - sys_set_malloc, 91
- mpx_supt.h
 - COM_PORT, 93
 - DEFAULT_DEVICE, 93
 - EXIT, 93
 - FALSE, 93
 - IDLE, 93
 - idle, 95
 - INVALID_BUFFER, 93
 - INVALID_COUNT, 94
 - INVALID_OPERATION, 94
 - IO_MODULE, 94
 - MEM_MODULE, 94
 - MODULE_F, 94
 - MODULE_R1, 94
 - MODULE_R2, 94
 - MODULE_R3, 94
 - MODULE_R4, 95
 - MODULE_R5, 95
 - mpx_init, 95
 - READ, 95
 - sys_alloc_mem, 96
 - sys_free_mem, 96
 - sys_req, 96
 - sys_set_free, 96
 - sys_set_malloc, 96
 - TRUE, 95
 - WRITE, 95
- msg1
 - procsr3.c, 131
- msg2
 - procsr3.c, 131
- msg3
 - procsr3.c, 131
- msg4
 - procsr3.c, 131
- msg5
 - procsr3.c, 131
- msgSize
 - procsr3.c, 131
- name
 - cmcb_s, 10
- named_arg
 - args.c, 103
- named_arg_count
 - parsed_args, 29
- named_arg_names
 - parsed_args, 29

- named_arg_values
 - parsed_args, 30
- new_frame
 - paging.c, 77
 - paging.h, 49
- newest_switch
 - syntax_highlight.c, 208
- next
 - cmcb_s, 10
- next_unnamed_arg
 - args.c, 103
- nframes
 - paging.c, 78
- nmi
 - interrupts.c, 63
- NO_ERROR
 - serial.c, 67
- no_warn
 - system.h, 55
- nop
 - system.h, 55
- NOT_DELETABLE
 - pcb.h, 178
- NULL
 - system.h, 55
- op_code
 - param, 28
- OPEN
 - driver.c, 101
- oper_status
 - dcb_t, 16
- out.c
 - print, 78
 - putc, 78
 - printf, 79
 - println, 79
 - read, 79
- out.h
 - aliasHelp, 81
 - blockHelp, 81
 - clearHelp, 81
 - cmd_help, 81
 - createpcbHelp, 81
 - deletepcbHelp, 82
 - freealarmHelp, 82
 - getdateHelp, 82
 - gettimeHelp, 82
 - helpHelp, 82
 - helpList, 82
 - isemptyHelp, 83
 - loadr3Help, 83
 - print, 83
 - putc, 83
 - printf, 83
 - println, 83
 - read, 84
 - resumeallHelp, 84
 - resumeHelp, 84
 - setalarmHelp, 84
 - setdateHelp, 84
 - setpriorityHelp, 84
 - settimeHelp, 85
 - showalarmsHelp, 85
 - showallocHelp, 85
 - showallpcbHelp, 85
 - showblockedpcbHelp, 85
 - showfreeHelp, 85
 - showpcbHelp, 86
 - showreadypcbHelp, 86
 - shutdownHelp, 86
 - suspendHelp, 86
 - unblockHelp, 86
 - versionHelp, 86
- outb
 - io.h, 37
- overflow
 - interrupts.c, 64
- p_protection_mode_t
 - pcb.h, 177
- p_queue
 - pcb.c, 174
- p_state_t
 - pcb.h, 178
- page_dir, 25
 - tables, 26
 - tables_phys, 26
- page_entry, 26
 - accessed, 26
 - dirty, 26
 - frameaddr, 27
 - present, 27
 - reserved, 27
 - usermode, 27
 - writable, 27
- page_fault
 - interrupts.c, 64
- PAGE_SIZE
 - paging.h, 48
- page_size
 - paging.c, 78
- page_table, 27
 - pages, 28
- pages
 - page_table, 28
- paging.c
 - cdir, 77
 - clear_bit, 76
 - find_free, 76
 - frames, 77
 - get_bit, 76
 - get_page, 76
 - init_paging, 76
 - kdir, 77
 - kheap, 77
 - load_page_dir, 76
 - mem_size, 77

- new_frame, 77
- nframes, 78
- page_size, 78
- phys_alloc_addr, 78
- set_bit, 77
- paging.h
 - clear_bit, 48
 - first_free, 48
 - get_bit, 49
 - get_page, 49
 - init_paging, 49
 - load_page_dir, 49
 - new_frame, 49
 - PAGE_SIZE, 48
 - set_bit, 49
- param, 28
 - buffer_ptr, 28
 - count_ptr, 28
 - device_id, 28
 - op_code, 28
- PARAM_NAME
 - syntax.h, 189
- PARAM_VALUE
 - syntax.h, 189
- params
 - mpx_supt.c, 92
 - system.c, 70
- parse_args
 - args.c, 103
 - args.h, 105
- parse_stack
 - args.c, 104
- parsed_args, 29
 - args.h, 105
 - flag_count, 29
 - flags, 29
 - named_arg_count, 29
 - named_arg_names, 29
 - named_arg_values, 30
 - unnamed_arg_count, 30
 - unnamed_args, 30
 - unnamed_args_used_so_far, 30
- pcb
 - pcb_node_t, 31
- pcb.c
 - allocatePCB, 166
 - blockPCB, 166
 - createPCB, 167
 - deletePCB, 167
 - f_queue, 174
 - fifo_queue, 174
 - findPCB, 167
 - freePCB, 168
 - initPCB, 168
 - insertPCB, 168
 - isSystemProcess, 170
 - p_queue, 174
 - priority_queue, 175
 - removePCB, 170
 - resumeAll, 170
 - resumePCB, 171
 - setPriority, 171
 - setupPCB, 172
 - showAll, 172
 - showBlocked, 172
 - showPCB, 173
 - showReady, 173
 - suspendPCB, 173
 - unblockPCB, 174
- pcb.h
 - allocatePCB, 178
 - BLOCKED, 178
 - blockPCB, 179
 - createPCB, 179
 - DELETABLE, 178
 - DELETABLE_WHEN_SUSPENDED, 178
 - deletePCB, 179
 - FIFO, 178
 - findPCB, 181
 - freePCB, 181
 - initPCB, 181
 - insertPCB, 182
 - isSystemProcess, 182
 - MAX_NAME_SIZE, 176
 - MAX_PRIORITY, 177
 - MAX_STACK_SIZE, 177
 - MIN_PRIORITY, 177
 - NOT_DELETABLE, 178
 - p_protection_mode_t, 177
 - p_state_t, 178
 - pcb_node_t, 177
 - pcb_queue_order_t, 178
 - pcb_queue_t, 177
 - PRIORITY, 178
 - READY, 178
 - removePCB, 182
 - resumeAll, 183
 - resumePCB, 183
 - RUNNING, 178
 - setPriority, 184
 - setupPCB, 184
 - showAll, 184
 - showBlocked, 185
 - showPCB, 185
 - showReady, 185
 - SUSPENDED_BLOCKED, 178
 - SUSPENDED_READY, 178
 - suspendPCB, 186
 - unblockPCB, 186
- pcb_name
 - pcb_t, 33
- pcb_node_t, 30
 - pcb, 31
 - pcb.h, 177
 - pcbn_next_pcb, 31
 - pcbn_prev_pcb, 31

- pcb_priority
 - pcb_t, 33
- pcb_process_class
 - pcb_t, 34
- pcb_process_state
 - pcb_t, 34
- pcb_protection_mode
 - pcb_t, 34
- pcb_queue, 31
 - pcbq_count, 32
 - pcbq_head, 32
 - pcbq_tail, 32
 - queue_order, 32
- pcb_queue_order_t
 - pcb.h, 178
- pcb_queue_t
 - pcb.h, 177
- pcb_stack_bottom
 - pcb_t, 34
- pcb_stack_top
 - pcb_t, 34
- pcb_t, 33
 - pcb_name, 33
 - pcb_priority, 33
 - pcb_process_class, 34
 - pcb_process_state, 34
 - pcb_protection_mode, 34
 - pcb_stack_bottom, 34
 - pcb_stack_top, 34
- pcbn_next_pcb
 - pcb_node_t, 31
- pcbn_prev_pcb
 - pcb_node_t, 31
- pcbq_count
 - pcb_queue, 32
- pcbq_head
 - pcb_queue, 32
- pcbq_tail
 - pcb_queue, 32
- phys_alloc_addr
 - heap.c, 75
 - paging.c, 78
- PIC1
 - interrupts.c, 59
- PIC2
 - interrupts.c, 59
- PIC_MASK
 - driver.c, 100
- polling
 - serial.c, 67
 - serial.h, 39
- present
 - page_entry, 27
- prev
 - cmcb_s, 10
- print
 - out.c, 78
 - out.h, 83
- print_color_code
 - colorize.c, 197
- putc
 - out.c, 78
 - out.h, 83
- printf
 - out.c, 79
 - out.h, 83
- println
 - out.c, 79
 - out.h, 83
- PRIORITY
 - pcb.h, 178
- priority_queue
 - commhand.c, 121
 - pcb.c, 175
 - system.c, 70
- proc1
 - procsr3.c, 129
 - procsr3.h, 132
- proc2
 - procsr3.c, 129
 - procsr3.h, 132
- proc3
 - procsr3.c, 129
 - procsr3.h, 132
- proc4
 - procsr3.c, 129
 - procsr3.h, 132
- proc5
 - procsr3.c, 130
 - procsr3.h, 132
- procsr3.c
 - er1, 130
 - er2, 130
 - er3, 130
 - er4, 130
 - er5, 130
 - erSize, 130
 - msg1, 131
 - msg2, 131
 - msg3, 131
 - msg4, 131
 - msg5, 131
 - msgSize, 131
 - proc1, 129
 - proc2, 129
 - proc3, 129
 - proc4, 129
 - proc5, 130
 - RC_1, 128
 - RC_2, 128
 - RC_3, 129
 - RC_4, 129
 - RC_5, 129
- procsr3.h
 - proc1, 132
 - proc2, 132

- proc3, [132](#)
- proc4, [132](#)
- proc5, [132](#)
- queue_order
 - pcb_queue, [32](#)
- RC_1
 - procsr3.c, [128](#)
- RC_2
 - procsr3.c, [128](#)
- RC_3
 - procsr3.c, [129](#)
- RC_4
 - procsr3.c, [129](#)
- RC_5
 - procsr3.c, [129](#)
- READ
 - mpx_supt.h, [95](#)
- read
 - out.c, [79](#)
 - out.h, [84](#)
- READING
 - driver.c, [101](#)
- READY
 - pcb.h, [178](#)
- ready_state
 - dcb_t, [16](#)
- RED
 - colorize.c, [196](#)
 - colorize.h, [198](#)
- removeAMCB
 - mm.c, [159](#)
 - mm.h, [163](#)
- removeFMCB
 - mm.c, [159](#)
 - mm.h, [163](#)
- removePCB
 - pcb.c, [170](#)
 - pcb.h, [182](#)
- reserved
 - interrupts.c, [64](#)
 - page_entry, [27](#)
- resumeAll
 - pcb.c, [170](#)
 - pcb.h, [183](#)
- resumeallHelp
 - help.c, [114](#)
 - out.h, [84](#)
- resumeHelp
 - help.c, [114](#)
 - out.h, [84](#)
- resumePCB
 - pcb.c, [171](#)
 - pcb.h, [183](#)
- RIGHT_ARROW
 - serial.c, [67](#)
- ring_buffer
 - dcb_t, [16](#)
- ring_buffer_head
 - dcb_t, [17](#)
- RING_BUFFER_SIZE
 - driver.c, [100](#)
- ring_buffer_tail
 - dcb_t, [17](#)
- rtc_isr
 - interrupts.c, [64](#)
- RUNNING
 - pcb.h, [178](#)
- SCRATCH_REGISTER
 - driver.c, [100](#)
- sec
 - date_time, [15](#)
- segment_not_present
 - interrupts.c, [64](#)
- serial.c
 - consume_special, [67](#)
 - DELETE, [66](#)
 - DOWN_ARROW, [66](#)
 - init_serial, [67](#)
 - LEFT_ARROW, [67](#)
 - NO_ERROR, [67](#)
 - polling, [67](#)
 - RIGHT_ARROW, [67](#)
 - serial_port_in, [68](#)
 - serial_port_out, [68](#)
 - serial_print, [68](#)
 - serial_println, [68](#)
 - set_serial_in, [68](#)
 - set_serial_out, [68](#)
 - UP_ARROW, [67](#)
- serial.h
 - COM1, [38](#)
 - COM2, [38](#)
 - COM3, [38](#)
 - COM4, [38](#)
 - init_serial, [39](#)
 - polling, [39](#)
 - serial_print, [39](#)
 - serial_println, [39](#)
 - set_serial_in, [39](#)
 - set_serial_out, [40](#)
- serial_port_in
 - serial.c, [68](#)
- serial_port_out
 - serial.c, [68](#)
- serial_print
 - serial.c, [68](#)
 - serial.h, [39](#)
- serial_println
 - serial.c, [68](#)
 - serial.h, [39](#)
- set_bit
 - paging.c, [77](#)
 - paging.h, [49](#)
- set_serial_in
 - serial.c, [68](#)

- serial.h, 39
- set_serial_out
 - serial.c, 68
 - serial.h, 40
- setAlarm
 - dnt.c, 137
 - dnt.h, 148
- setalarmHelp
 - help.c, 114
 - out.h, 84
- setdate
 - dnt.c, 138
 - dnt.h, 149
- setdateHelp
 - help.c, 109, 114
 - out.h, 84
- setDateInMemory
 - dnt.c, 138
 - dnt.h, 149
- setPriority
 - pcb.c, 171
 - pcb.h, 184
- setpriorityHelp
 - help.c, 114
 - out.h, 84
- settime
 - dnt.c, 139
 - dnt.h, 150
- settimeHelp
 - help.c, 109, 114
 - out.h, 85
- setTimeInMemory
 - dnt.c, 139
 - dnt.h, 150
- setupPCB
 - pcb.c, 172
 - pcb.h, 184
- showAlarms
 - dnt.c, 139
 - dnt.h, 151
- showalarmsHelp
 - help.c, 115
 - out.h, 85
- showAll
 - pcb.c, 172
 - pcb.h, 184
- showAllocated
 - mm.c, 159
 - mm.h, 164
- showallocHelp
 - help.c, 115
 - out.h, 85
- showallpcbHelp
 - help.c, 115
 - out.h, 85
- showBlocked
 - pcb.c, 172
 - pcb.h, 185
- showblockedpcbHelp
 - help.c, 115
 - out.h, 85
- showFree
 - mm.c, 159
 - mm.h, 164
- showfreeHelp
 - help.c, 115
 - out.h, 85
- showPCB
 - pcb.c, 173
 - pcb.h, 185
- showpcbHelp
 - help.c, 115
 - out.h, 86
- showReady
 - pcb.c, 173
 - pcb.h, 185
- showreadypcbHelp
 - help.c, 116
 - out.h, 86
- shutdown.c
 - cmd_shutdown, 117
- shutdownHelp
 - help.c, 110, 116
 - out.h, 86
- SINGLE_QUOTE_STRING
 - syntax.h, 189
- SINGLE_QUOTE_STRING_END_QUOTE
 - syntax.h, 189
- size
 - cmcb_s, 10
 - header, 20
 - index_entry, 24
- size_t
 - system.h, 56
- skip_ws
 - utils.c, 191
 - utils.h, 192
- sselect
 - idt_entry_struct, 22
 - tables.h, 43
- stack_empty
 - args.c, 104
- stack_peek
 - args.c, 104
- stack_pop
 - args.c, 104
- stack_push
 - args.c, 104
- stack_segment
 - interrupts.c, 64
- stack_size
 - args.c, 105
- start_addr
 - mm.c, 160
- START_SEQ
 - colorize.c, 194

- states
 - syntax_highlight.c, 208
- sti
 - system.h, 55
- strcat
 - string.c, 89
 - string.h, 52
- strcmp
 - string.c, 89
 - string.h, 52
- strcpy
 - string.c, 89
 - string.h, 52
- string.c
 - atoi, 88
 - isspace, 88
 - itoa, 88
 - memset, 89
 - strcat, 89
 - strcmp, 89
 - strcpy, 89
 - strlen, 89
 - strtok, 89
- string.h
 - atoi, 51
 - isspace, 51
 - itoa, 51
 - memset, 52
 - strcat, 52
 - strcmp, 52
 - strcpy, 52
 - strlen, 52
 - strtok, 52
- strlen
 - string.c, 89
 - string.h, 52
- strtok
 - string.c, 89
 - string.h, 52
- student_free
 - mpx_supt.c, 92
- student_malloc
 - mpx_supt.c, 92
- SUSPENDED_BLOCKED
 - pcb.h, 178
- SUSPENDED_READY
 - pcb.h, 178
- suspendHelp
 - help.c, 116
 - out.h, 86
- suspendPCB
 - pcb.c, 173
 - pcb.h, 186
- switch_indexes
 - syntax_highlight.c, 208
- switch_to
 - syntax_highlight.c, 206
- syntax.c
 - changes_state, 188
 - get_state, 189
- syntax.h
 - changes_state, 190
 - CMD_NAME, 189
 - CMD_NAME_OR_LEADING_WHITESPACE, 189
 - DEFAULT, 189
 - DOUBLE_QUOTE_STRING, 189
 - DOUBLE_QUOTE_STRING_END_QUOTE, 189
 - END_OF_INPUT, 189
 - get_state, 190
 - PARAM_NAME, 189
 - PARAM_VALUE, 189
 - SINGLE_QUOTE_STRING, 189
 - SINGLE_QUOTE_STRING_END_QUOTE, 189
 - SyntaxState, 189
 - SYNTAX_COLOR_CMD_NAME
 - syntax_highlight.h, 209
 - SYNTAX_COLOR_DEFAULT
 - syntax_highlight.h, 209
 - SYNTAX_COLOR_DOUBLE_QUOTE_STRING
 - syntax_highlight.h, 209
 - SYNTAX_COLOR_PARAM_NAME
 - syntax_highlight.h, 209
 - SYNTAX_COLOR_PARAM_VALUE
 - syntax_highlight.h, 209
 - SYNTAX_COLOR_SINGLE_QUOTE_STRING
 - syntax_highlight.h, 210
 - syntax_disable_highlighting
 - syntax_highlight.c, 207
 - syntax_highlight.h, 210
 - syntax_enable_highlighting
 - syntax_highlight.c, 207
 - syntax_highlight.h, 210
 - syntax_handle_char
 - syntax_highlight.c, 207
 - syntax_highlight.h, 210
 - syntax_highlight.c
 - color_for, 206
 - enabled, 208
 - get_state_at, 206
 - newest_switch, 208
 - states, 208
 - switch_indexes, 208
 - switch_to, 206
 - syntax_disable_highlighting, 207
 - syntax_enable_highlighting, 207
 - syntax_handle_char, 207
 - syntax_init, 207
 - syntax_highlight.h
 - MAX_SYNTAX_SWITCHES, 209
 - SYNTAX_COLOR_CMD_NAME, 209
 - SYNTAX_COLOR_DEFAULT, 209
 - SYNTAX_COLOR_DOUBLE_QUOTE_STRING, 209
 - SYNTAX_COLOR_PARAM_NAME, 209
 - SYNTAX_COLOR_PARAM_VALUE, 209

- SYNTAX_COLOR_SINGLE_QUOTE_STRING, 210
- syntax_disable_highlighting, 210
- syntax_enable_highlighting, 210
- syntax_handle_char, 210
- syntax_init, 210
- sys_init
 - syntax_highlight.c, 207
 - syntax_highlight.h, 210
- SyntaxState
 - syntax.h, 189
- sys_alloc_mem
 - mpx_supt.c, 91
 - mpx_supt.h, 96
- sys_call
 - system.c, 70
- sys_call_isr
 - interrupts.c, 64
- sys_free_mem
 - mpx_supt.c, 91
 - mpx_supt.h, 96
- sys_req
 - mpx_supt.c, 91
 - mpx_supt.h, 96
- sys_set_free
 - mpx_supt.c, 91
 - mpx_supt.h, 96
- sys_set_malloc
 - mpx_supt.c, 91
 - mpx_supt.h, 96
- system.c
 - cop, 70
 - global_context, 70
 - klogv, 69
 - kpanic, 69
 - params, 70
 - priority_queue, 70
 - sys_call, 70
- system.h
 - asm, 54
 - cli, 54
 - GDT_CS_ID, 54
 - GDT_DS_ID, 55
 - hlt, 55
 - iret, 55
 - klogv, 56
 - kpanic, 56
 - no_warn, 55
 - nop, 55
 - NULL, 55
 - size_t, 56
 - sti, 55
 - u16int, 56
 - u32int, 56
 - u8int, 56
 - volatile, 56
- table
 - index_table, 24
- TABLE_SIZE
 - heap.h, 45
- tables
 - page_dir, 26
- tables.c
 - gdt_entries, 72
 - gdt_init_entry, 71
 - gdt_ptr, 72
 - idt_entries, 73
 - idt_ptr, 73
 - idt_set_gate, 71
 - init_gdt, 72
 - init_idt, 72
 - write_gdt_ptr, 72
 - write_idt_ptr, 72
- tables.h
 - __attribute__, 41
 - access, 42
 - base, 42
 - base_high, 42
 - base_low, 43
 - base_mid, 43
 - flags, 43
 - gdt_init_entry, 41
 - idt_set_gate, 42
 - init_gdt, 42
 - init_idt, 42
 - limit, 43
 - limit_low, 43
 - sselect, 43
 - zero, 43
- tables_phys
 - page_dir, 26
- TRUE
 - mpx_supt.h, 95
- type
 - cmcb_s, 10
- u16int
 - system.h, 56
- u32int
 - system.h, 56
- u8int
 - system.h, 56
- unblockHelp
 - help.c, 116
 - out.h, 86
- unblockPCB
 - pcb.c, 174
 - pcb.h, 186
- unnamed_arg_count
 - parsed_args, 30
- unnamed_args
 - parsed_args, 30
- unnamed_args_used_so_far
 - parsed_args, 30
- UP_ARROW
 - serial.c, 67
- user_read_buf

- dcb_t, [17](#)
- user_read_count
 - dcb_t, [17](#)
- user_write_buf
 - dcb_t, [17](#)
- user_write_count
 - dcb_t, [17](#)
- usermode
 - page_entry, [27](#)
- utils.c
 - is_name_char, [191](#)
 - skip_ws, [191](#)
- utils.h
 - is_name_char, [191](#)
 - skip_ws, [192](#)
- version.c
 - cmd_version, [118](#)
- versionHelp
 - help.c, [116](#)
 - out.h, [86](#)
- versionOs
 - help.c, [110](#)
- volatile
 - system.h, [56](#)
- WHITE
 - colorize.c, [196](#)
 - colorize.h, [198](#)
- WRITE
 - mpx_supt.h, [95](#)
- write_gdt_ptr
 - tables.c, [72](#)
- write_hist_to_buf
 - history.c, [154](#)
- write_idt_ptr
 - tables.c, [72](#)
- writable
 - page_entry, [27](#)
- WRITING
 - driver.c, [101](#)
- year
 - date_time, [15](#)
- YELLOW
 - colorize.c, [196](#)
 - colorize.h, [198](#)
- yield
 - context.c, [124](#)
 - context.h, [127](#)
- zero
 - idt_entry_struct, [22](#)
 - tables.h, [43](#)