## RUNTIME TERROR OS R5

Generated by Doxygen 1.9.1

| 1 | Class Index  | 1  |
|---|--|----|
|   | 1.1 Class List   | 1  |
| 2 | File Index   | 3  |
|   | 2.1 File List  | 3  |
| 3 | Class Documentation  | 5  |
|   | 3.1 Alarm Struct Reference   | 5  |
|   | 3.1.1 Detailed Description   | 5  |
|   | 3.2 CMCB Struct Reference  | 5  |
|   | 3.2.1 Detailed Description   | 6  |
|   | 3.3 context Struct Reference   | 6  |
|   | 3.3.1 Detailed Description   | 6  |
|   | 3.4 date_time Struct Reference   | 6  |
|   | 3.4.1 Detailed Description   | 7  |
|   | 3.5 footer Struct Reference  | 7  |
|   | 3.5.1 Detailed Description   | 7  |
|   | 3.6 gdt_descriptor_struct Struct Reference   | 7  |
|   | 3.6.1 Detailed Description   | 7  |
|   | 3.7 gdt_entry_struct Struct Reference  | 7  |
|   | 3.7.1 Detailed Description   | 8  |
|   | 3.8 header Struct Reference  | 8  |
|   | 3.8.1 Detailed Description   | 8  |
|   | 3.9 heap Struct Reference  | 8  |
|   | 3.9.1 Detailed Description   | 8  |
|   | 3.10 idt_entry_struct Struct Reference   | 9  |
|   | 3.10.1 Detailed Description  | 9  |
|   | 3.11 idt struct Struct Reference   | 9  |
|   | 3.11.1 Detailed Description  | 9  |
|   | 3.12 index_entry Struct Reference  | 9  |
|   | 3.12.1 Detailed Description  | 10 |
|   | 3.13 index_table Struct Reference  | 10 |
|   | 3.13.1 Detailed Description  | 10 |
|   | 3.14 List Struct Reference   | 10 |
|   | 3.14.1 Detailed Description  | 10 |
|   | 3.15 MemList Struct Reference  | 10 |
|   | 3.15.1 Detailed Description  | 11 |
|   | 3.16 page_dir Struct Reference   | 11 |
|   | 3.16.1 Detailed Description  | 11 |
|   | 3.17 page_entry Struct Reference   | 11 |
|   | 3.17.1 Detailed Description  | 11 |
|   | 3.18 page_table Struct Reference   | 12 |
|   | 3.18.1 Detailed Description  | 12 |
|   | Control Dotalina Dotalina Control Cont | 14 |

|     | 3.19 param Struct Reference   | 12 |
|-----|---|----|
|     | 3.19.1 Detailed Description   | 12 |
|     | 3.20 PCB Struct Reference   | 12 |
|     | 3.20.1 Detailed Description   | 13 |
|     | 3.21 Queue Struct Reference   | 13 |
|     | 3.21.1 Detailed Description   | 13 |
| 4 F | File Documentation  | 15 |
|     | 4.1 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/asm.h File Reference        | 15 |
|     | 4.2 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/interrupts.h File Reference | 15 |
|     | 4.3 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/io.h File Reference         | 15 |
|     | 4.3.1 Macro Definition Documentation  | 15 |
|     | 4.3.1.1 inb   | 16 |
|     | 4.4 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/serial.h File Reference     | 16 |
|     | 4.5 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/tables.h File Reference     | 16 |
|     | 4.6 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/mem/heap.h File Reference        | 17 |
|     | 4.7 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/mem/paging.h File Reference      | 18 |
|     | 4.8 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/string.h File Reference          | 18 |
|     | 4.8.1 Function Documentation  | 18 |
|     | 4.8.1.1 atoi()  | 18 |
|     | 4.8.1.2 isspace()   | 19 |
|     | 4.8.1.3 memset()  | 19 |
|     | 4.8.1.4 strcat()  | 21 |
|     | 4.8.1.5 strcmp()  | 21 |
|     | 4.8.1.6 strcpy()  | 22 |
|     | 4.8.1.7 strlen()  | 22 |
|     | 4.8.1.8 strtok()  | 23 |
|     | 4.9 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/system.h File Reference          | 23 |
|     | 4.10 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/interrupts.c File Reference | 24 |
|     | 4.11 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/kmain.c File Reference      | 26 |
|     | 4.12 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/serial.c File Reference     | 26 |
|     | 4.13 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/system.c File Reference     | 27 |
|     | 4.14 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/tables.c File Reference     | 27 |
|     | 4.15 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/mem/heap.c File Reference        | 27 |
|     | 4.16 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/mem/paging.c File Reference      | 28 |
|     | 4.17 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/lib/string.c File Reference             | 29 |
|     | 4.17.1 Function Documentation   | 29 |
|     | 4.17.1.1 atoi()   | 29 |
|     | 4.17.1.2 isspace()  | 30 |
|     | 4.17.1.3 memset()   | 30 |
|     | 4.17.1.4 strcat()   | 31 |
|     | 4.17.1.5 strcmp()   | 31 |

| 4.17.1.6 strcpy()   | 31 |
|---|----|
| 4.17.1.7 strlen()   | 33 |
| 4.17.1.8 strtok()   | 33 |
| 4.18 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/mpx_supt.c File Reference                               | 34 |
| 4.19 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/mpx_supt.h File Reference                               | 35 |
| 4.20 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R1/comHand.h File Reference                             | 35 |
| 4.20.1 Function Documentation   | 36 |
| 4.20.1.1 comHand()  | 36 |
| $4.21\ D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R1/userFunctions.c\ File\ Reference\ .\ .\ .\ .$  | 40 |
| 4.21.1 Function Documentation   | 41 |
| 4.21.1.1 BCDtoDec()   | 41 |
| 4.21.1.2 Block()  | 41 |
| 4.21.1.3 Create_PCB()   | 42 |
| 4.21.1.4 DectoBCD()   | 43 |
| 4.21.1.5 Delete_PCB()   | 43 |
| 4.21.1.6 EdgeCase()   | 43 |
| 4.21.1.7 GetDate()  | 44 |
| 4.21.1.8 GetTime()  | 45 |
| 4.21.1.9 Help()   | 45 |
| 4.21.1.10 itoa()  | 47 |
| 4.21.1.11 Resume()  | 48 |
| 4.21.1.12 Set_Priority()  | 48 |
| 4.21.1.13 SetDate()   | 49 |
| 4.21.1.14 SetTime()   | 50 |
| 4.21.1.15 Show_All()  | 51 |
| 4.21.1.16 Show_Blocked()  | 51 |
| 4.21.1.17 Show_PCB()  | 53 |
| 4.21.1.18 Show_Ready()  | 54 |
| 4.21.1.19 Suspend()   | 56 |
| 4.21.1.20 toLowercase()   | 56 |
| 4.21.1.21 Unblock()   | 56 |
| 4.21.1.22 Version()   | 57 |
| 4.21.2 Variable Documentation   | 57 |
| 4.21.2.1 AlarmList  | 57 |
| $4.22\ D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R1/userFunctions.h\ File\ Reference \\ \ .\ .\ .$ | 58 |
| 4.22.1 Function Documentation   | 59 |
| 4.22.1.1 BCDtoDec()   | 59 |
| 4.22.1.2 Block()  | 59 |
| 4.22.1.3 Create_PCB()   | 60 |
| 4.22.1.4 DectoBCD()   | 60 |
| 4.22.1.5 Delete_PCB()   | 61 |
| 4.22.1.6 EdgeCase()   | 61 |

| 4.22.1.7 GetDate()   | 62 |
|--|----|
| 4.22.1.8 GetTime()   | 62 |
| 4.22.1.9 Help()  | 63 |
| 4.22.1.10 itoa()   | 65 |
| 4.22.1.11 Resume()   | 66 |
| 4.22.1.12 Set_Priority()   | 66 |
| 4.22.1.13 SetDate()  | 67 |
| 4.22.1.14 SetTime()  | 68 |
| 4.22.1.15 Show_All()   | 68 |
| 4.22.1.16 Show_Blocked()   | 69 |
| 4.22.1.17 Show_PCB()   | 70 |
| 4.22.1.18 Show_Ready()   | 72 |
| 4.22.1.19 Suspend()  | 73 |
| 4.22.1.20 toLowercase()  | 74 |
| 4.22.1.21 Unblock()  | 74 |
| 4.22.1.22 Version()  | 75 |
| 4.23 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/sys_proc_loader.c File Reference | 75 |
| 4.24 D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/sys_proc_loader.h File Reference | 75 |
| Index  | 77 |

# **Chapter 1**

## **Class Index**

### 1.1 Class List

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

| Alarm                 | . 5  |
|-----------------------|------|
| CMCB                  | . 5  |
| context               | . 6  |
| date_time             | . 6  |
| footer                | . 7  |
| gdt_descriptor_struct | . 7  |
| gdt_entry_struct      | . 7  |
| header                | . 8  |
| heap                  | . 8  |
| idt_entry_struct      | . 9  |
| idt_struct            | . 9  |
| index_entry           | . 9  |
| index_table           | . 10 |
| List                  | . 10 |
| MemList               | . 10 |
| page_dir              | . 11 |
| page_entry            | . 11 |
| page_table            | . 12 |
| param                 | . 12 |
| PCB                   | . 12 |
| Qualita               | 12   |

2 Class Index

# **Chapter 2**

# File Index

## 2.1 File List

Here is a list of all documented files with brief descriptions:

| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/string.h                | 18 |
|--|----|
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/system.h                | 23 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/asm.h              | 15 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/interrupts.h       | 15 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/io.h               | 15 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/serial.h           | 16 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/tables.h           | 16 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/mem/heap.h              | 17 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/mem/paging.h            | 18 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/interrupts.c        | 24 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/kmain.c             | 26 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/serial.c            | 26 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/system.c            | 27 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/tables.c            | 27 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/mem/heap.c               | 27 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/mem/paging.c             | 28 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/lib/string.c                    | 29 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/mpx_supt.c              | 34 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/mpx_supt.h              | 35 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/ <b>procsr3.c</b>       | ?? |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/ <b>procsr3.h</b>       | ?? |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/sys_proc_loader.c       | 75 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/sys_proc_loader.h       | 75 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R1/comHand.c            | ?? |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R1/comHand.h            | 35 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R1/userFunctions.c      | 40 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R1/userFunctions.h      | 58 |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R2/ <b>PCB.c</b>        | ?? |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R2/ <b>PCB.h</b>        | ?? |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R5/ <b>MCB.c</b>        | ?? |
|  | ?? |
|  | ?? |
| D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R5/ <b>R5commands.h</b> | ?? |

File Index

## **Chapter 3**

## **Class Documentation**

### 3.1 Alarm Struct Reference

### **Public Attributes**

- int hour
- int minute
- · int second
- char message [85]
- struct Alarm \* next
- struct Alarm \* prev

### 3.1.1 Detailed Description

Definition at line 15 of file userFunctions.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R1/userFunctions.h

### 3.2 CMCB Struct Reference

- u32int size
- struct CMCB \* prev
- struct CMCB \* next
- char Process\_name [10]
- u32int address
- int MEMState

6 Class Documentation

### 3.2.1 Detailed Description

Definition at line 4 of file MCB.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R5/MCB.h

### 3.3 context Struct Reference

### **Public Attributes**

- u32int gs
- u32int fs
- u32int es
- u32int ds
- u32int edi
- u32int **esi**
- u32int ebp
- u32int esp
- u32int **ebx**
- u32int edx
- u32int ecx
- u32int **eax**
- u32int eip
- u32int cs
- · u32int eflags

### 3.3.1 Detailed Description

Definition at line 34 of file PCB.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R2/PCB.h

### 3.4 date time Struct Reference

- int sec
- int min
- int hour
- int day\_w
- int day\_m
- int day\_y
- int mon
- int year

3.5 footer Struct Reference 7

### 3.4.1 Detailed Description

Definition at line 32 of file system.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/system.h

### 3.5 footer Struct Reference

### **Public Attributes**

· header head

### 3.5.1 Detailed Description

Definition at line 18 of file heap.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/mem/heap.h

### 3.6 gdt\_descriptor\_struct Struct Reference

### **Public Attributes**

- u16int limit
- u32int base

### 3.6.1 Detailed Description

Definition at line 25 of file tables.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/core/tables.h

### 3.7 gdt\_entry\_struct Struct Reference

- u16int limit\_low
- u16int base low
- u8int base mid
- u8int access
- · u8int flags
- u8int base\_high

8 Class Documentation

### 3.7.1 Detailed Description

Definition at line 32 of file tables.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/core/tables.h

### 3.8 header Struct Reference

### **Public Attributes**

- int size
- int index\_id

### 3.8.1 Detailed Description

Definition at line 13 of file heap.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/mem/heap.h

### 3.9 heap Struct Reference

### **Public Attributes**

- index table index
- u32int base
- u32int max\_size
- u32int min\_size

### 3.9.1 Detailed Description

Definition at line 35 of file heap.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/mem/heap.h

### 3.10 idt\_entry\_struct Struct Reference

### **Public Attributes**

- u16int base\_low
- u16int sselect
- u8int zero
- u8int flags
- u16int base\_high

### 3.10.1 Detailed Description

Definition at line 8 of file tables.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/core/tables.h

### 3.11 idt\_struct Struct Reference

### **Public Attributes**

- u16int limit
- u32int base

### 3.11.1 Detailed Description

Definition at line 18 of file tables.h.

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

 $\bullet \ \ D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/core/tables.h$ 

### 3.12 index\_entry Struct Reference

- int size
- · int empty
- · u32int block

10 Class Documentation

### 3.12.1 Detailed Description

Definition at line 22 of file heap.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/mem/heap.h

### 3.13 index\_table Struct Reference

### **Public Attributes**

- index entry table [TABLE SIZE]
- int id

### 3.13.1 Detailed Description

Definition at line 29 of file heap.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/mem/heap.h

### 3.14 List Struct Reference

### **Public Attributes**

- Alarm \* head
- Alarm \* tail

### 3.14.1 Detailed Description

Definition at line 24 of file userFunctions.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R1/userFunctions.h

### 3.15 MemList Struct Reference

### **Public Attributes**

CMCB \* head

### 3.15.1 Detailed Description

Definition at line 18 of file MCB.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R5/MCB.h

### 3.16 page dir Struct Reference

### **Public Attributes**

- page\_table \* tables [1024]
- u32int tables\_phys [1024]

### 3.16.1 Detailed Description

Definition at line 36 of file paging.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/mem/paging.h

### 3.17 page entry Struct Reference

### **Public Attributes**

u32int present: 1
u32int writeable: 1
u32int usermode: 1
u32int accessed: 1
u32int dirty: 1

u32int reserved: 7u32int frameaddr: 20

### 3.17.1 Detailed Description

Definition at line 14 of file paging.h.

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

• D:/GITHUB/CS 450 RunTime Terror/mpx core/include/mem/paging.h

12 Class Documentation

### 3.18 page table Struct Reference

### **Public Attributes**

• page\_entry pages [1024]

### 3.18.1 Detailed Description

Definition at line 28 of file paging.h.

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

• D:/GITHUB/CS 450 RunTime Terror/mpx core/include/mem/paging.h

### 3.19 param Struct Reference

### **Public Attributes**

- int op code
- int device\_id
- char \* buffer\_ptr
- int \* count\_ptr

### 3.19.1 Detailed Description

Definition at line 34 of file mpx\_supt.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/mpx\_supt.h

### 3.20 PCB Struct Reference

- unsigned char **stack** [MEM1K]
- unsigned char \* stackTop
- struct PCB \* prev
- struct PCB \* next
- char Process\_Name [10]
- int Process\_Class
- int **Priority**
- · int ReadyState
- · int SuspendedState

### 3.20.1 Detailed Description

Definition at line 15 of file PCB.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R2/PCB.h

### 3.21 Queue Struct Reference

### **Public Attributes**

- int count
- PCB \* head
- PCB \* tail

### 3.21.1 Detailed Description

Definition at line 27 of file PCB.h.

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

• D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R2/PCB.h

14 Class Documentation

## **Chapter 4**

## **File Documentation**

4.1 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/core/asm.h File Reference

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

4.2 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/core/interrupts.h File Reference

### **Functions**

- void init\_irq (void)
- void init\_pic (void)
- 4.3 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/include/core/io.h File Reference

### **Macros**

- #define **outb**(port, data) asm volatile ("outb %%al,%%dx" : : "a" (data), "d" (port))
- #define inb(port)
- 4.3.1 Macro Definition Documentation

### 4.3.1.1 inb

Definition at line 17 of file io.h.

# 4.4 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/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)

# 4.5 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/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

# 4.6 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/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)

# 4.7 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_← core/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)

# 4.8 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/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)

### 4.8.1 Function Documentation

### 4.8.1.1 atoi()

```
int atoi ( {\rm const\ char\ *\ s\ )}
```

Description: Convert an ASCII string to an integer

### **Parameters**

```
s String
```

Definition at line 50 of file string.c.

```
int charVal=0;
char sign = ' ';
char c = *s;
53
54
55
56
57
      while(isspace(&c)){ ++s; c = *s;} // advance past whitespace
60
      if (*s == '-' \mid | *s == '+') sign = *(s++); // save the sign
61
62
63
    while(*s != '\0'){
    charVal = *s - 48;
res = res * 10 + charVal;
64
66
67
       s++;
68
69
      }
70
71
72
     if ( sign == '-') res=res * -1;
73
    return res; // return integer
74
75 }
```

### 4.8.1.2 isspace()

```
int isspace ( {\tt const\ char\ *\ c}\ )
```

Description: Determine if a character is whitespace.

### **Parameters**

c character to check

Definition at line 121 of file string.c.

### 4.8.1.3 memset()

```
void* memset ( \label{eq:void*} \mbox{void} * s \mbox{,}
```

```
int c, size_t n)
```

Description: Set a region of memory.

### **Parameters**

| s | destination   |
|---|---------------|
| С | byte to write |
| n | count         |

Definition at line 139 of file string.c.

```
140 {
wnile(n--) {
    *p++ = (unsigned char) c;
    144    }
    return s;
    146 }
         unsigned char *p = (unsigned char *) s;
```

### 4.8.1.4 strcat()

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

Description: Concatenate the contents of one string onto another.

#### **Parameters**

| s1 | destination |
|----|-------------|
| s2 | source      |

Definition at line 108 of file string.c.

```
110 char *rc = s1;

111 if (*s1) while(*++s1);

112 while( (*s1++ = *s2++) );

113 return rc;

114 }
```

### 4.8.1.5 strcmp()

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

Description: String comparison

### **Parameters**

| s1 | string 1 |
|----|----------|
| s2 | string 2 |

Definition at line 81 of file string.c.

### 4.8.1.6 strcpy()

Description: Copy one string to another.

#### **Parameters**

| s1 | destination |
|----|-------------|
| s2 | source      |

Definition at line 38 of file string.c.

```
39 {
40    char *rc = s1;
41    while( (*s1++ = *s2++) );
42    return rc; // return pointer to destination string
43 }
```

### 4.8.1.7 strlen()

```
int strlen ( {\rm const\ char\ *\ s\ )}
```

Description: Returns the length of a string.

### **Parameters**

```
s input string
```

Definition at line 26 of file string.c.

```
27 {
28   int r1 = 0;
29   if (*s) while(*s++) r1++;
30   return r1;//return length of string
31 }
```

### 4.8.1.8 strtok()

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

Description: Split string into tokens

#### **Parameters**

| s1 | String    |
|----|-----------|
| s2 | delimiter |

#### Definition at line 153 of file string.c.

```
155
      static char *tok_tmp = NULL;
      const char *p = s2;
156
157
158
      //new string
      if (s1!=NULL) {
160
       tok\_tmp = s1;
161
162
      //old string cont'd
      else {
  if (tok_tmp==NULL) {
163
164
165
          return NULL;
166
167
        s1 = tok\_tmp;
168 }
169
      //skip leading s2 characters
while ( *p && *s1 ) {
170
      if (*s1==*p) {
172
        ++s1;
p = s2;
173
174
175
          continue;
176
177
179
      //no more to parse
180
181
      if (!*s1){
        return (tok_tmp = NULL);
182
183
184
185
      //skip non-s2 characters
     tok_tmp = s1;
while (*tok_tmp) {
   p = s2;
186
187
188
       while (*p) {
   if (*tok_tmp==*p++) {
189
191
       *tok\_tmp++ = ' \setminus 0';
       return s1;
192
193
          }
194
195
        ++tok_tmp;
196
197
      //end of string
tok_tmp = NULL;
198
199
200 return s1;
201 }
```

## D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/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)

# 4.10 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_← core/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]

# 4.11 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/kernel/core/kmain.c File Reference

```
#include <stdint.h>
#include <string.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 <mem/paging.h>
#include "modules/R1/comHand.h"
#include "modules/R1/comHand.h"
#include "modules/R5/R5commands.h"
#include "modules/R5/R5commands.h"
#include "modules/R5/MCB.h"
```

### **Functions**

· void kmain (void)

# 4.12 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/kernel/core/serial.c File Reference

```
#include <stdint.h>
#include <string.h>
#include <core/io.h>
#include <core/serial.h>
```

### **Macros**

• #define NO\_ERROR 0

### **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 \*cmdBuffer, int \*count)

### **Variables**

- int serial\_port\_out = 0
- int serial\_port\_in = 0

# 4.13 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_← core/kernel/core/system.c File Reference

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

#### **Functions**

- void klogv (const char \*msg)
- void kpanic (const char \*msg)

# 4.14 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/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]

# 4.15 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/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
```

# 4.16 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_← core/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

# 4.17 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/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)
- 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)

### 4.17.1 Function Documentation

### 4.17.1.1 atoi()

```
int atoi ( {\rm const\ char\ *\ s\ )}
```

Description: Convert an ASCII string to an integer

### **Parameters**

s String

Definition at line 50 of file string.c.

```
int res=0;
53
     int charVal=0;
      char sign = '
55
56
57
      while(isspace(&c)){ ++s; c = *s;} // advance past whitespace
58
59
      if (*s == '-' \mid | *s == '+') sign = *(s++); // save the sign
63
      while(*s != '\0'){
   charVal = *s - 48;
64
65
       res = res \star 10 + charVal;
66
68
69
      }
70
72
      if ( sign == '-') res=res * -1;
```

```
74 return res; // return integer
75 }
```

### 4.17.1.2 isspace()

```
int isspace ( {\rm const\ char\ *\ }c\ )
```

Description: Determine if a character is whitespace.

### **Parameters**

```
c character to check
```

Definition at line 121 of file string.c.

### 4.17.1.3 memset()

```
void* memset ( \label{eq:void*} \mbox{void} * s, \\ \mbox{int } c, \\ \mbox{size\_t } n \mbox{)}
```

Description: Set a region of memory.

### **Parameters**

| s | destination   |
|---|---------------|
| С | byte to write |
| n | count         |

Definition at line 139 of file string.c.

```
140 {
141    unsigned char *p = (unsigned char *) s;
142    while (n--) {
143         *p++ = (unsigned char) c;
144    }
145    return s;
146 }
```

#### 4.17.1.4 strcat()

```
char* strcat ( \label{eq:char} \mbox{char} \ * \ s1, \mbox{const char} \ * \ s2 \ )
```

Description: Concatenate the contents of one string onto another.

#### **Parameters**

| s1 | destination |
|----|-------------|
| s2 | source      |

Definition at line 108 of file string.c.

```
109 {
110    char *rc = s1;
111    if (*s1) while(*++s1);
112    while( (*s1++ = *s2++) );
113    return rc;
114 }
```

#### 4.17.1.5 strcmp()

```
int strcmp (  {\rm const~char} \ * \ s1, \\ {\rm const~char} \ * \ s2 \ )
```

Description: String comparison

#### **Parameters**

| s1 | string 1 |
|----|----------|
| s2 | string 2 |

Definition at line 81 of file string.c.

# 4.17.1.6 strcpy()

Description: Copy one string to another.

#### **Parameters**

| s1 | destination |
|----|-------------|
| s2 | source      |

Definition at line 38 of file string.c.

```
Jennal.

39 {
40    char *rc = s1;
41    while( (*s1++ = *s2++) );
42    return rc; // return pointer to destination string
```

# 4.17.1.7 strlen()

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

Description: Returns the length of a string.

#### **Parameters**

```
input string
```

Definition at line 26 of file string.c.

```
27 {
28   int r1 = 0;
29   if (*s) while(*s++) r1++;
30   return r1;//return length of string
31 }
```

# 4.17.1.8 strtok()

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

Description: Split string into tokens

#### **Parameters**

| s1 | String    |
|----|-----------|
| s2 | delimiter |

Definition at line 153 of file string.c.

```
155 static char *tok_tmp = NULL;
     const char *p = s2;
157
     //new string
if (s1!=NULL) {
158
159
160
      tok\_tmp = s1;
161
```

```
//old string cont'd
162
163
      else {
       if (tok_tmp==NULL) {
164
165
         return NULL;
166
        s1 = tok_tmp;
167
168
169
170
      //skip leading s2 characters
      while ( *p && *s1 ) {
  if (*s1==*p) {
171
172
173
         ++s1;
        p = s2;
continue;
174
175
176
177
178
179
180
      //no more to parse
      if (!*s1){
182
       return (tok_tmp = NULL);
183
184
      //skip non-s2 characters
185
186
      tok_tmp = s1;
while (*tok_tmp) {
187
       p = s2;
while (*p){
188
189
          if (*tok_tmp==*p++) {
190
        *tok_tmp++ = ' \setminus 0';
191
192
        return s1;
        }
}
193
194
195
        ++tok_tmp;
196 }
197
      //end of string
198
199 tok_tmp = NULL;
200 return s1;
201 }
```

# 4.18 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/mpx\_ supt.c File Reference

```
#include "mpx_supt.h"
#include <mem/heap.h>
#include <string.h>
#include <core/serial.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 ()
- u32int \* sys\_call (context \*registers)

# **Variables**

- param params
- int current\_module = -1
- u32int(\* student\_malloc )(u32int)
- int(\* student\_free )(void \*)
- PCB \* cop
- context \* initial

# 4.19 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/mpx\_ supt.h File Reference

```
#include <system.h>
#include "R2/PCB.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 ()
- u32int \* sys\_call (context \*registers)

# 4.20 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R1/com Hand.h File Reference

# **Functions**

• int comHand ()

#### 4.20.1 Function Documentation

#### 4.20.1.1 comHand()

```
int comHand ( )
```

Description: Interprets user input to call the appropriate user functions.

Definition at line 23 of file comHand.c.

```
24
            Help("\0");
2.5
26
            char cmdBuffer[100];
            int bufferSize = 99;
29
            int quit = 0;
30
            int shutdown = 0;
31
            while(quit != 1)
32
33
                memset (cmdBuffer, '\0', 100);
              memset(cmdBuller, *(0*, 100);
sys_req(READ, DEFAULT_DEVICE, cmdBuffer, &bufferSize);
char* FirstToken = strtok(cmdBuffer, "-");
char* SecondToken = strtok(NULL, "-");
char* ThirdToken = strtok(NULL, "-");
35
36
37
                char* FourthToken = strtok(NULL, "-");
38
                char* FifthToken = strtok(NULL, "-");
if(shutdown == 0) {
39
42
                     R1 comHand
43
                          if (strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, NULL) == 0)
44
45
                              Help("\0");
46
                          //R1 Commands
48
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "version") == 0 &&
        strcmp(ThirdToken,NULL) == 0) {
49
                              Help("Version");
50
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "getDate") == 0 &&
51
        strcmp(ThirdToken, NULL) == 0) {
52
                              Help("GetDate");
53
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "setDate") == 0 &&
54
        strcmp(ThirdToken, NULL) == 0) {
                              Help("SetDate");
55
57
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "getTime") == 0 &&
        strcmp(ThirdToken, NULL) == 0) {
                              Help("GetTime");
58
59
60
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "setTime") == 0 &&
        strcmp(ThirdToken, NULL) == 0) {
61
                              Help("SetTime");
62
63
                          // R2 Commands
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "suspend") == 0 &&
64
       strcmp(ThirdToken, NULL) == 0) {
65
                              Help("suspend");
66
67
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "resume") == 0 &&
        strcmp(ThirdToken, NULL) == 0) {
                              Help("resume");
68
69
70
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "setPriority") == 0 &&
        strcmp(ThirdToken, NULL) == 0)
71
                              Help("setPriority");
72
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "showPCB") == 0 &&
73
        strcmp(ThirdToken, NULL) == 0) {
74
                              Help("showPCB");
75
76
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "showAll") == 0 &&
        strcmp(ThirdToken, NULL) == 0) {
                              Help("showAll");
78
79
                          else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "showReady") == 0 &&
        strcmp(ThirdToken, NULL) == 0)
```

```
80
                            Help("showReady");
81
82
                        else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "showBlocked") == 0 &&
       strcmp(ThirdToken, NULL) == 0) {
                            Help("showBlocked");
83
84
85
                        // Temporary R2 commands
86
                        // else if(strcmp(FirstToken,"help") == 0 && strcmp(SecondToken,"createPCB") == 0 &&
       strcmp(ThirdToken, NULL) == 0)
87
                           Help("createPCB");
                        // }
88
                        else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "deletePCB") == 0 &&
89
       strcmp(ThirdToken, NULL) == 0)
90
                            Help("deletePCB");
91
92
                        // else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "block") == 0 &&
       strcmp(ThirdToken, NULL) == 0)
93
                           Help("block");
                        //
94
95
                        // else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "unblock") == 0 &&
       strcmp(ThirdToken, NULL) == 0)
96
                           Help("unblock");
                        // }
97
                        else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "shutdown") == 0 &&
98
       strcmp(ThirdToken, NULL) == 0)
99
                           Help("shutdown");
100
101
                         else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "infinite") == 0 &&
       strcmp(ThirdToken, NULL) == 0)
                             Help("infinte");
102
103
104
                         // R4 Commands
105
                         else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "loadr3") == 0 &&
       strcmp(ThirdToken,NULL) == 0)
106
                             Help("loadr3");
107
                         else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "alarm") == 0 &&
108
       strcmp(ThirdToken, NULL) == 0)
109
                             Help("alarm");
110
111
                         // Bonus Command
                         else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "clear") == 0 &&
112
       strcmp(ThirdToken,NULL) == 0)
113
                             Help("clear");
114
       // else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "clear") == 0 && strcmp(ThirdToken, NULL) == 0) {
115
116
117
                            Help("heap");
                         //
118
                         // }
119
                         // else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "clear") == 0 &&
       strcmp(ThirdToken, NULL) == 0) {
120
                            Help("alloc");
121
                         // else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "clear") == 0 &&
122
       strcmp(ThirdToken, NULL) == 0)
123
                         // Help("free");
124
125
                         // else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "clear") == 0 &&
       strcmp(ThirdToken,NULL) == 0) {
                             Help("empty");
126
                         11
127
128
                         // R5 Commands
129
                         else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "clear") == 0 &&
       strcmp(ThirdToken, NULL) == 0)
130
                             Help("showFree");
131
                         else if(strcmp(FirstToken, "help") == 0 && strcmp(SecondToken, "clear") == 0 &&
132
       strcmp(ThirdToken, NULL) == 0)
133
                             Help("showAlloc");
134
135
136
137
138
139
140
141
                         else if(strcmp(FirstToken, "version") == 0 && strcmp(SecondToken, NULL) == 0)
142
                             Version();
                         else if(strcmp(FirstToken, "clear") == 0 && strcmp(SecondToken, NULL) == 0)
143
144
                             clear();
145
146
                         else if(strcmp(FirstToken, "getDate") == 0 && strcmp(SecondToken, NULL) == 0)
147
                             GetDate();
148
                         else if(strcmp(FirstToken, "setDate") == 0){
149
                             if (EdgeCase (SecondToken) == 1 && EdgeCase (ThirdToken) == 1 &&
150
```

```
EdgeCase(FourthToken) == 1 && EdgeCase(FifthToken) == 1)
                               SetDate(atoi(SecondToken), atoi(ThirdToken), atoi(FourthToken),
151
       atoi(FifthToken));
152
153
                           else
                               printf("\x1b[31m""\nERROR: Invalid parameters for setDate \n""\x1b[0m");
154
155
156
                        else if(strcmp(FirstToken, "getTime") == 0 && strcmp(SecondToken, NULL) == 0) //Return
       the current time held by the registers.
157
                           GetTime();
                        else if(strcmp(FirstToken, "setTime") == 0 && strcmp(FifthToken, NULL) == 0){
158
                           if (EdgeCase(SecondToken) == 1 && EdgeCase(ThirdToken) == 1 &&
159
       EdgeCase(FourthToken) == 1)
160
                                   SetTime(atoi(SecondToken), atoi(ThirdToken), atoi(FourthToken));
       //input as Hour-Minute-Seconds
161
162
                           else
                               printf("\x1b[31m""\nERROR: Invalid parameters for setTime \n""\x1b[0m");
163
164
165
166
167
168
169
170
171
        172
                        R2 comHand
173
        else if(strcmp(FirstToken, "suspend") == 0 && strcmp(ThirdToken, NULL) == 0 &&
174
       strcmp (FourthToken, NULL) == 0 && strcmp (FifthToken, NULL) == 0) {
175
                           Suspend (SecondToken):
176
177
                        else if(strcmp(FirstToken, "resume") == 0 && strcmp(ThirdToken, NULL) == 0 &&
       strcmp(FourthToken,NULL) == 0 && strcmp(FifthToken,NULL) == 0) {
178
                           Resume (SecondToken);
179
                       else if(strcmp(FirstToken, "setPriority") == 0 && strcmp(FourthToken, NULL) == 0 &&
180
       strcmp(FifthToken, NULL) == 0) {
181
                           if (EdgeCase (ThirdToken) == 1)
                               Set_Priority(SecondToken, atoi(ThirdToken)); //input as
182
       setPriority-Process_Name-Priority
183
                           }
184
                           else
                              printf("\x1b[31m""\nERROR: Invalid parameters for setPriority, priority must
185
       be entered as a integer. \n""\x1b[0m");
186
187
                        else if(strcmp(FirstToken, "showPCB") == 0 && strcmp(ThirdToken, NULL) == 0 &&
       strcmp (FourthToken, NULL) == 0 && strcmp (FifthToken, NULL) == 0) {
188
                           Show_PCB (SecondToken);
189
                           printf("\n");
190
                        else if(strcmp(FirstToken, "showAll") == 0 && strcmp(SecondToken, NULL) == 0 &&
191
       strcmp(ThirdToken,NULL) == 0 && strcmp(FourthToken,NULL) == 0 && strcmp(FifthToken,NULL) == 0) {
                           Show_All();
192
                           printf("\n");
193
194
                        else if(strcmp(FirstToken, "showReady") == 0 && strcmp(SecondToken, NULL) == 0 &&
195
       strcmp (ThirdToken, NULL) == 0 && strcmp (FourthToken, NULL) == 0 && strcmp (FifthToken, NULL) == 0) {
196
                           Show_Ready();
197
                           printf("\n");
198
                        else if(strcmp(FirstToken, "showBlocked") == 0 && strcmp(SecondToken, NULL) == 0 &&
199
       strcmp(ThirdToken, NULL) == 0 && strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
200
                           Show_Blocked();
201
                           printf("\n");
202
                        }
203
204
205
206
                        /****** R2 Temp Commands *******/
207
                        //Removed from active for R3/R4
208
209
                        else if(strcmp(FirstToken, "createPCB") == 0) {
                           if( strlen(SecondToken) < 11)
210
                                   Create_PCB(SecondToken, atoi(ThirdToken), atoi(FourthToken));
211
       //input as Process_Name-Priority-Class
212
213
        printf("\x1b[31m""\nERROR: Invalid parameters for createPCB, Process\_name must only contain 10 or fewer characters. <math display="block"> n""\x1b[0m"]; 
214
215
216
                        else if(strcmp(FirstToken, "deletePCB") == 0 && strcmp(ThirdToken, NULL) == 0 &&
217
       strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
218
                           Delete_PCB(SecondToken);
219
220
```

```
221
222
223
                        //Removed from active for R3/R4
224
                       else if(strcmp(FirstToken, "block") == 0 && strcmp(ThirdToken, NULL) == 0 &&
225
       strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
226
                           Block (SecondToken);
227
228
                       else if(strcmp(FirstToken, "unblock") == 0 && strcmp(ThirdToken, NULL) == 0 &&
       strcmp(FourthToken,NULL) == 0 && strcmp(FifthToken,NULL) == 0) {
229
                           Unblock (SecondToken);
230
231
                       */
232
233
                       R3 comHand
234
            *******************************
235
                       //Removed for R4
236
                       else if(strcmp(FirstToken, "yield") == 0 && strcmp(SecondToken, NULL) == 0 &&
237
       strcmp(ThirdToken, NULL) == 0 && strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
238
                           yield();
                           printf("\n");
239
2.40
                       else if(strcmp(FirstToken, "loadr3") == 0 && strcmp(SecondToken, NULL) == 0 &&
2.41
       strcmp (ThirdToken, NULL) == 0 && strcmp (FourthToken, NULL) == 0) {
242
                           loader();
243
                           printf("\n");
244
245
                       */
246
            *****
247
                       R4 comHand
248
249
                       else if(strcmp(FirstToken, "alarm") == 0) {
250
                           if (EdgeCase(ThirdToken) == 1 && EdgeCase(FourthToken) == 1 &&
       EdgeCase(FifthToken) == 1)
                               if (atoi(ThirdToken) < 24 && atoi(FourthToken) < 60 && atoi(FifthToken) <</pre>
251
       60)
           {
252
                                   loaderalarm(SecondToken, atoi(ThirdToken), atoi(FourthToken),
       atoi(FifthToken));
253
                                   printf("\n"); //input as Message-Hour-Minute-Seconds
254
255
                               else
                               printf("\x1b[31m""\nERROR: Invalid parameters for alarm, must be a valid
2.56
       time \n""\x1b[0m");
257
258
259
                               printf("\x1b[31m""\nERROR: Invalid parameters for alarm \n""\x1b[0m");
2.60
261
                       else if(strcmp(FirstToken, "loadr3") == 0 && strcmp(SecondToken, NULL) == 0 &&
262
       strcmp (ThirdToken, NULL) == 0 && strcmp (FourthToken, NULL) == 0 && strcmp (FifthToken, NULL) == 0) {
263
                           loader();
264
                           printf("\n");
265
                       else if(strcmp(FirstToken, "infinite") == 0 && strcmp(SecondToken, NULL) == 0 &&
266
       strcmp(ThirdToken, NULL) == 0 && strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
267
                           loaderinfinite();
268
                           printf("\n");
269
                       }
270
271
272
                       R5 comHand
273
                        // else if(strcmp(FirstToken, "heap") == 0 && strcmp(ThirdToken, NULL) == 0 &&
274
       strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
275
                       // Init_Heap(atoi(SecondToken));
276
                       // else if(strcmp(FirstToken, "alloc") == 0 && strcmp(ThirdToken, NULL) == 0 &&
277
       strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
278
                           Alloc_Mem(atoi(SecondToken));
279
280
                       // else if(strcmp(FirstToken, "free") == 0 && strcmp(ThirdToken, NULL) == 0 &&
       strcmp(FourthToken,NULL) == 0 && strcmp(FifthToken,NULL) == 0) {
281
                       // Free_Mem(atoi(SecondToken));
282
                       // else if(strcmp(FirstToken, "empty") == 0 && strcmp(SecondToken, NULL) == 0 &&
283
       strcmp(ThirdToken, NULL) == 0 && strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
                       // IsEmpty();
// }
284
285
                       else if (strcmp(FirstToken, "showFree") == 0 && strcmp(SecondToken, NULL) == 0 &&
286
       strcmp(ThirdToken, NULL) == 0 && strcmp(FourthToken, NULL) == 0 && strcmp(FifthToken, NULL) == 0) {
287
                           ShowFree();
288
                       else if(strcmp(FirstToken, "showAlloc") == 0 && strcmp(SecondToken, NULL) == 0 &&
289
       strcmp(ThirdToken,NULL) == 0 && strcmp(FourthToken,NULL) == 0 && strcmp(FifthToken,NULL) == 0) {
290
                           ShowAlloc();
291
```

```
293
294
                       shutdown comHand
295
           else if(strcmp(FirstToken, "shutdown") == 0 && strcmp(SecondToken, NULL) == 0){
296
                          printf("\x1b[33m""\nAre you sure you want to shutdown? [yes/no]\n""\x1b[0m");
297
299
300
                           printf("\x1b[31m""\nERROR: Not a valid command \n""\x1b[0m");
301
302
303
304
                   else{
305
                       if(strcmp(FirstToken, "yes") == 0 && shutdown == 1)
306
                           quit = 1;
307
                       else if(strcmp(FirstToken, "no") == 0){
308
                           printf("\x1b[33m""\nShutdown Cancelled\\x1b[0m \n");
309
310
                           shutdown = 0;
312
                           \label{lem:lem:printf("x1b[31m""\nERROR: Please enter \"yes\" or \"no\" \n""\x1b[0m");
313
314
                   sys_req(IDLE, DEFAULT_DEVICE, NULL, NULL);
315
316
           getReady() -> head = NULL;
sys_req(EXIT, DEFAULT_DEVICE, NULL, NULL);
317
318
319
           return 0; //shutdown procedure
320
```

# 4.21 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R1/user Functions.c File Reference

```
#include <stdint.h>
#include <string.h>
#include <system.h>
#include <core/serial.h>
#include <core/io.h>
#include "../mpx_supt.h"
#include "userFunctions.h"
#include "../procsr3.h"
#include "../sys_proc_loader.h"
```

#### **Functions**

- · void clear ()
- char \* itoa (int num)
- int BCDtoDec (int BCD)
- int DectoBCD (int Decimal)
- void **printf** (char msg[])
- int EdgeCase (char \*pointer)
- void SetTime (int hours, int minutes, int seconds)
- void GetTime ()
- void SetDate (int day, int month, int millennium, int year)
- void GetDate ()
- void Version ()
- char toLowercase (char c)
- void Help (char \*request)
- void Suspend (char \*ProcessName)
- void Resume (char \*ProcessName)
- void Set\_Priority (char \*ProcessName, int Priority)

- void Show\_PCB (char \*ProcessName)
- void Show\_All ()
- void Show Ready ()
- void Show\_Blocked ()
- void Create\_PCB (char \*ProcessName, int Priority, int Class)
- void Delete\_PCB (char \*ProcessName)
- void Block (char \*ProcessName)
- void Unblock (char \*ProcessName)
- void loader ()
- void loadr3 (char \*name, u32int func)
- · void yield ()
- void loaderinfinite ()
- List \* getList ()
- void loaderalarm (char text[], int hours, int minutes, int seconds)

## **Variables**

List AlarmList

## 4.21.1 Function Documentation

# 4.21.1.1 BCDtoDec()

```
int BCDtoDec (
          int BCD )
```

Description: Changes binary number to decimal numbers.

## **Parameters**

value Binary number to be changed to decimal

## Definition at line 82 of file userFunctions.c.

```
82 {
83    return (((BCD»4)*10) + (BCD & 0xF));
84 }
```

## 4.21.1.2 Block()

Brief Description: Places a PCD in the blocked state and reinserts it into the correct queue.

Description: Can except a string as a pointer that is the Process Name. The specified PCB will be places in a blocked state and reinserted into the appropriate queue. An error check for a valid name occurs.

## **Parameters**

| Process_Name | Character pointer that matches the name of process. |  |
|--------------|---|--|
|--------------|---|--|

Definition at line 958 of file userFunctions.c.

```
959
      PCB* pcb = FindPCB(ProcessName);
960
      if (pcb == NULL)
       printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
961
962
963
     else {
       if (pcb->ReadyState == BLOCKED) {
964
           printf("\x1b[32m""\nThis Process is already BLOCKED \n""\x1b[0m");
965
966
967
       else
         RemovePCB (pcb);
968
         pcb->ReadyState = BLOCKED;
969
970
          InsertPCB(pcb);
971
972
     }
973 }
```

#### 4.21.1.3 Create PCB()

Brief Description: Calls SetupPCB() and inserts PCB into appropriate queue.

Description: Can except a string as a pointer that is the Process Name. Can accept two integers, Priority and Class. SetupPCB() will be called and the PCB will be inserted into the appropriate queue. An error check for unique and valid Process Name, an error check for valid process class, and an error check for process priority.

# Parameters

| Process_Name | Character pointer that matches the name of process. |
|--------------|---|
| Priority     | integer that matches the priority number.           |
| Class        | integer that matches the class number.              |

# Definition at line 901 of file userFunctions.c.

```
902
       if (FindPCB(ProcessName) == NULL)
         if(Priority >= 0 && Priority < 10){
  if(Class == 0 || Class == 1){
    PCB* pcb = SetupPCB(ProcessName, Class, Priority);</pre>
903
904
905
              InsertPCB (pcb);
906
907
           } else{
             printf("\x1b[31m""\nERROR: Not a valid Class \n""\x1b[0m");
909
910
         } else{
           printf("\x1b[31m""\\nERROR: Not a valid Priority \n""\\x1b[0m");
911
912
913
      } else{
         printf("\x1b[31m""\nERROR: This Process Name already exists \n""\x1b[0m");
914
915 }
916 }
```

#### 4.21.1.4 DectoBCD()

Description: Changes decimal numbers to binary numbers.

#### **Parameters**

Decimal Decimal number to be changed to binary

Definition at line 89 of file userFunctions.c.

#### 4.21.1.5 Delete PCB()

Brief Description: Removes PCB from appropriate queue and frees all associated memory.

Description: Can except a string as a pointer that is the Process Name. Removes PCB from the appropriate queue and then frees all associated memory. An error check to make sure process name is valid.

#### **Parameters**

*Process\_Name* Character pointer that matches the name of process.

Definition at line 927 of file userFunctions.c.

```
928
     PCB* pcb = FindPCB(ProcessName);
     printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
}
929
930
931
    else if(strcmp(pcb->Process_Name, "InfProc") == 0) {
932
     if (pcb->SuspendedState == YES) {
933
934
               RemovePCB (pcb);
935
               FreePCB(pcb);
936
937
           else
               printf("\x1b[3lm""\nERROR:This process cannot be deleted unless it is in the suspended
938
      state\n""\x1b[0m");
939
940
     else if(pcb -> Process_Class == SYSTEM) {
       printf("\x1b[31m""\nERROR: System Processes cannot be deleted from the system. <math>\n""\x1b[0m");
941
942
     else {
943
         RemovePCB (pcb);
944
945
           FreePCB(pcb);
946
947 }
```

## 4.21.1.6 EdgeCase()

Description: Compares pointer char to validate if it is a number or not.

#### **Parameters**

Compares pointer char to validate if it is a number or not.

Definition at line 110 of file userFunctions.c.

```
110
111
        int valid = 0;
        if (strcmp(pointer, "00") == 0) {
  valid = 1;
112
113
114
          return valid;
115
       else if (strcmp(pointer, "0") == 0) {
  valid = 1;
  return valid;
116
117
118
119
120
       else {
121
          int j;
          find j,
valid = 0;
for(j = 0; j <= 99; j++) {
   if(strcmp(pointer,itoa(j)) == 0)
   valid = 1;</pre>
122
123
124
125
126
         if (valid == 0)
127
128
             return valid;
         }
129
130
       }
131
       return valid:
132 }
```

#### 4.21.1.7 GetDate()

```
void GetDate ( )
```

Description: Returns the full date back to the user in decimal form.

No parameters.

Definition at line 272 of file userFunctions.c.

```
272
273
      outb (0x70, 0x07);
275
        unsigned char day = BCDtoDec(inb(0x71));
276
         outb(0x70,0x08);
277
         unsigned char month = BCDtoDec(inb(0x71));
278
         outb (0x70, 0x32);
        unsigned char millennium = BCDtoDec(inb(0x71)); char msg[2] = "-";
279
280
        char msg3[10] = "Date: ";
281
282
        printf(msg3);
283
        printf(itoa(day));
//sys_req(WRITE, COM1, itoa(day), &check);
284
285
286
        printf(msg);
        printf(itoa(month));
287
288
         //sys_req(WRITE, COM1, itoa(month), &check);
289
         printf(msg);
      printf(itoa(millennium));
//sys_req(WRITE, COM1, itoa(millennium), &check);
outb(0x70,0x09);
if(BCDtoDec(inb(0x71)) == 0){
290
291
292
293
294
          printf("00");
295
         //sys_req(WRITE, COM1, "00", &check);
296
297
      else {
298
             unsigned char year = BCDtoDec(inb(0x71));
299
              printf(itoa(year));
300
              //sys_req(WRITE, COM1, itoa(year), &check);
301
302
         printf("\n");
303 }
```

## 4.21.1.8 GetTime()

```
void GetTime ( )
```

Description: retrieve and return the time values for hours, minutes, and seconds form the clock register using inb(Port,address).

No parameters.

Definition at line 191 of file userFunctions.c.

```
192
193
        int hour;
194
        int minute;
195
        int second;
196
        outb (0x70, 0x04);
197
        unsigned char hours = inb(0x71);
198
        outb(0x70,0x02);
199
        unsigned char minutes = inb(0x71);
200
        outb(0x70,0x00);
        unsigned char seconds = inb(0x71);
char msg1[2] = ":";
2.01
202
        char msg2[10] = "Time: ";
203
204
        printf(msg2);
205
        hour = BCDtoDec(hours);
206
        printf(itoa(hour));
        //sys_req(WRITE, COM1, itoa(hour), &check);
printf(msg1);
minute = BCDtoDec(minutes);
207
208
209
210
        printf(itoa(minute));
211
        //sys_req(WRITE, COM1, itoa(minute), &check);
212
        printf(msg1);
        second = BCDtoDec(seconds);
213
       printf(itoa(second));
214
        //sys_req(WRITE, COM1, itoa(second), &check);
215
     printf("\n");
```

## 4.21.1.9 Help()

Brief Description: Gives helpful information for one of the functions

Description: Can except a string as a pointer, if the pointer is null then the function will print a complete list of avaliable commands to the console. If the pointer is a avaliable commands then instructions on how to use the command will be printed. If the command does not exist then a message explaining that it is not a valid command will be displayed.

#### **Parameters**

request Character pointer that matches the name of the function that you need help with.

Definition at line 332 of file userFunctions.c.

```
332
          if (request[0] == '\0') {
333
                //removed for R3/R4 from active command list
334
                //n createPCB \n block \n unblock
//\n heap alloc \n free
335
                //\n heap
336
                                                                       empty
               printf("\n to chain commands and parameters, please use \"-\" between keywords \n");
337
         printf("\n getDate setDate \n getTime setTime \n version su
setPriority \n showPCB showAll \n showReady showBlocked \n deletePCB
clear \n loadr3 infinte \n showFree showAlloc \n\n");
                                                                                                                  suspend \n resume
338
                                                                                                                        shutdown \n alarm
```

```
339
        else if (strcmp(request, "GetDate") == 0) {
340
            printf("\n getDate returns the current date that is loaded onto the operating system.\n");
341
342
        else if (strcmp(request, "SetDate") == 0) {
343
       printf("\n setDate allows the user to reset the correct date into the system, as follows setDate-"BLU"day"RESET"-"BLU"month"RESET"-"BLU"year"RESET".\n Time must be inputed as a two digit
344
       number, Example 02 or 00");
345
346
        else if (strcmp(request, "GetTime") == 0) {
            printf("\n getTime returns the current time as hours, minutes, seconds that is loaded onto the
347
       operating system.\n");
348
349
        else if (strcmp(request, "SetTime") == 0) {
350
            \texttt{printf("} \\ \texttt{\sc n} \  \, \texttt{setTime} \  \, \texttt{allows} \  \, \texttt{the user to reset the correct time into the system, as follows}
       setTime-"BLU"hour"RESET"-"BLU"minute"RESET"-"BLU"second"RESET".\n Time must be inputed as a two digit
       number, Example 02 or 00");
351
352
        else if (strcmp(request, "Version") == 0) {
353
            printf("\n version returns the current operating software version that the system is
       running.\n");
354
        else if (strcmp(request, "infinte") == 0) {
355
            printf("\n infinite Loads the infinite process into the ready queue.\n");
356
357
358
        else if (strcmp(request, "loadr3") == 0)
            printf("\n loadr3 Loads in all five of the R3 test processes in a suspended state into the
359
       queue.\n");
360
361
        else if (strcmp(request, "alarm") == 0)
           printf("\n alarm creates a user specified alarm with a user set message and time
362
       alarm-MSG-hour-minute-second.\n");
363
364
        else if (strcmp(request, "clear") == 0)
365
           \verb|printf("\n clear erases the console of all typed commands and refreshes it with just the command
       list.\n");
366
        }
367
368
      else if(strcmp(request, "shutdown") == 0)
       printf("\n shutdown shuts down the system.\n");
369
370
371
372
373
371
                R2 Commands
375
       else if(strcmp(request, "suspend") == 0) {
376
            printf("\n Suspend takes in the name of a PCB (suspend-NAME) then places it into the suspended
377
       state and reinserts it into the correct queue. \n");
378
379
      else if(strcmp(request, "resume") == 0) {
380
            printf("\ \ \ Resume\ takes\ in\ the\ name\ of\ a\ PCB\ (resume-NAME)\ then\ removes\ it\ from\ the\ suspended
       state and adds it to the correct queue. \n");
381
382
      else if(strcmp(request, "setPriority") == 0) {
            printf("\n SetPriority takes in the name of a PCB and the priority (setPrioriry-NAME-PRIORITY)
383
       it needs to be set to then reinstates the specified PCB into a new location by priority.\n^n);
384
      else if(strcmp(request, "showPCB") == 0) {
385
            printf("\n ShowPCB takes in the name of a PCB and returns all the associated attributes to the 
386
       user.\n");
387
388
      else if(strcmp(request, "showAll") == 0) {
389
            \verb|printf("\n ShowAll takes no parameters but returns all PCB's that are currently in any of the
       queues.\n");
390
      else if(strcmp(request, "showReady") == 0) {
391
            printf("\n ShowReady takes in no parameters but returns all PCB's and there attributes that
392
       currently are in the ready state.\n");
393
      else if(strcmp(request, "showBlocked") == 0) {
    printf("\n ShowBlocked takes in no parameters but returns all PCB's and there attributes that
394
395
       currently are in the blocked state.\n");
396
397
398 /******** R2 Temp Commands
       ************************
      else if(strcmp(request, "deletePCB") == 0) {
399
            printf("\n DeletePCB takes in the process_name (deletePCB-NAME) then deletes it from the queue
400
       and free's all the memory that was previously allocated to the specified PCB.\n");
401
402
      //removed for R3/R4 from active command list
403
      else if(strcmp(request, "createPCB") == 0) {
404
            printf("\n CreatePCB takes in the process_name, process_class, and
405
```

```
process_priority.(createPCB-NAME-PRIORITY-CLASS) Then assigns this new process into the correct
406
             else if(strcmp(request,"block") == 0) { printf("\n Block takes in the process_name (block-NAME) then sets it's state to blocked and
407
408
              reinserts it back into the correct queue.\n");
409
410
             else if(strcmp(request, "unblock") == 0) {
411
                          \verb|printf("\n Unblock takes in the process_name (unblock-NAME)| then sets it's state to ready and |
              reinserts it back into the correct queue. \n");
412
             }
413
414
415
416 /******* R5 Temp Commands
            417
                // printf("\n heap initializes the memory heap for the entire system.\n");
418
419
            // else if(strcmp(request, "alloc") == 0)
                         printf("\n alloc allocates the specified amount of memory to the specific process
              (alloc-process_name-size).\n");
422
             // else if(strcmp(request, "free") == 0) {
423
                 // printf("\n free frees the specified memory at the address given (free-address).\n");
424
425
426
            // else if(strcmp(request, "empty") == 0) {
           // printf("\n isempty returns true or false depending on if the heap has allocated memory.\n");
// }
427
428
429 /****** R5 Commands
              **********************
            else if(strcmp(request,"showFree") == 0) {
    printf("\n showfree shows all the free blocks avaliable within the heap list.\n");
430
431
432
            else if(strcmp(request, "showAlloc") == 0) { printf("\n showAlloc shows all the allocated blocks within the heap list.\n");
433
434
            }
435
436
437
438
                          \dot{\text{printf("}} \\ \text{xlb[31m""} \\ \text{nThe requested command does not exist please refer to the Help function for a likely of the state 
439
              full list of commands.\n""\x1b[0m");
440
441 }
```

## 4.21.1.10 itoa()

```
char* itoa (
                int num )
```

Description: An integer is taken and seperated into individual chars and then all placed into a character array. Adapted from geeksforgeeks.org.

#### **Parameters**

```
num integer to be put into array Title: itoa Author: Neha Mahajan Date: 29 May, 2017 Availability: https://www.geeksforgeeks.org/implement-itoa/
```

Definition at line 51 of file userFunctions.c.

```
53
      int i, j, k, count;
54
      i = num;
      j = 0;
55
      count = 0;
      while(i){ // count number of digits
      count++;
59
          i /= 10;
60
     }
61
     char* arr1;
62
      char arr2[count];
```

```
64
       arr1 = (char*)sys_alloc_mem(count); //memory allocation
66
       while(num){ // seperate last digit from number and add ASCII
           arr2[++j] = num%10 + '0';
67
           num /= 10;
68
69
70
71
       for (k = 0; k < j; k++) \{ // \text{ reverse array results} \}
72
           arr1[k] = arr2[j-k];
73
       arr1[k] = ' \setminus 0';
74
       sys_free_mem(arr1);
75
76
       return(char*)arr1;
```

# 4.21.1.11 Resume()

Brief Description: Places a PCD in the not suspended state and reinserts it into the appropriate queue.

Description: Can except a string as a pointer that is the Process Name. Places a PCB in the not suspended state and reinserts it into the appropriate queue. An error check for valid Process Name.

#### **Parameters**

Process Name

Character pointer that matches the name of process.

Definition at line 484 of file userFunctions.c.

```
PCB* pcb = FindPCB(ProcessName);
485
        if (pcb == NULL)
486
487
         printf(RED"\nERROR: Not a valid process name \n"RESET);
488
489
490
            if(pcb->SuspendedState == NO)
                \label{lem:printf} \verb|printf(GRN"\nThis Process is already in the NONSUSPENDED state \n"RESET)|;
491
492
            else if(pcb -> Process_Class == APPLICATION)
493
               pcb->SuspendedState = NO;
495
496
            else
                 printf("\x1b[31m""\nERROR: Cannot Alter System Process \n""\x1b[0m");
497
498
499 }
```

## 4.21.1.12 Set\_Priority()

Brief Description: Sets PCB priority and reinserts the process into the correct place in the correct queue.

Description: Can except a string as a pointer that is the Process Name. Can accept and integer than is the Priority. Sets a PCB's priority and reinserts the process into the correct place in the correct queue. An error check for valid Process Name and an error check for a valid priority 1 - 9.

#### **Parameters**

| Process_Name | Character pointer that matches the name of process. |
|--------------|---|
| Priority     | integer that matches the priority number.           |

#### Definition at line 511 of file userFunctions.c.

```
511
       PCB* pcb = FindPCB(ProcessName);
if (pcb == NULL) {
512
513
           printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
514
515
516
        else if(Priority >= 10){
           printf("\x1b[31m""\nERROR: Not a valid Priority \n""\x1b[0m");
517
518
519
        else if(pcb -> Process_Class == APPLICATION) {
520
           RemovePCB(pcb);
521
            pcb->Priority = Priority;
522
            InsertPCB(pcb);
523
524
        else
            printf("\x1b[31m""\nERROR: Cannot Alter System Process \n""\x1b[0m");
526 }
```

#### 4.21.1.13 SetDate()

```
void SetDate (
    int day,
    int month,
    int millennium,
    int year )
```

Description: Sets the date register to the new values that the user inputed, all values must be inputed as Set 

Dime(day, month, millenial, year).

{

#### Parameters

| day       | Integer to be set in the Day position       |
|-----------|---|
| month     | Integer to be set in the Month position     |
| millenial | Integer to be set in the Millenial position |
| year      | Integer to be set in the Year position      |

#### Definition at line 225 of file userFunctions.c.

```
225
226
     outb (0x70, 0x07);
227
     int tempDay = BCDtoDec(inb(0x71));
     outb(0x70,0x08);
228
229
     int tempMonth = BCDtoDec(inb(0x71));
230
     outb (0x70, 0x32);
     int tempMillennium = BCDtoDec(inb(0x71));
231
     outb(0x70,0x09);
int tempYear = BCDtoDec(inb(0x71));
232
233
234
     cli();
      outb(0x70,0x07);
235
236
       outb(0x71,DectoBCD (day));
237
      outb(0x70,0x08);
       outb(0x71,DectoBCD (month));
238
239
       outb(0x70,0x32);
240
       outb(0x71, DectoBCD (millennium));
241
       outb (0x70, 0x09);
242
       outb(0x71,DectoBCD (year));
```

```
246
      outb (0x70, 0x08);
247
      unsigned char newMonth = BCDtoDec(inb(0x71));
248
      outb (0x70, 0x32);
249
      unsigned char newMillennium = BCDtoDec(inb(0x71));
      outb (0x70, 0x09);
2.50
251
      unsigned char newYear = BCDtoDec(inb(0x71));
      if (newDay != day || newMonth != month || newMillennium != millennium || newYear != year) {
252
253
        printf("Your input was invalid\n");
254
        cli();
255
            outb (0x70, 0x07);
            outb(0x71,DectoBCD (tempDay));
256
            outb(0x70,0x08);
257
            outb(0x71, DectoBCD (tempMonth));
258
259
            outb(0x70,0x32);
260
            outb(0x71,DectoBCD (tempMillennium));
261
            outb (0x70, 0x09);
            outb(0x71,DectoBCD (tempYear));
262
263
            sti();
264
     }
265
     else
266
        printf("Date Set\n");
267 }
```

#### 4.21.1.14 SetTime()

```
void SetTime (
                 int hours,
                 int minutes,
                 int seconds )
```

Description: sets the time register to the new values that the user inputed, all values must be inputed as SetTime(← Hours, Minutes, Seconds).

#### **Parameters**

| hours   | Integer to be set in the Hour position    |
|---------|---|
| minutes | Integer to be set in the Minutes position |
| seconds | Integer to be set in the Seconds position |

#### Definition at line 152 of file userFunctions.c.

```
152
153
      outb (0x70.0x04):
      unsigned char tempHours = BCDtoDec(inb(0x71));
154
155
      outb(0x70,0x02);
      unsigned char tempMinutes = BCDtoDec(inb(0x71));
156
      outb (0x70, 0x00);
158
      unsigned char tempSeconds = BCDtoDec(inb(0x71));
159
        cli(); //outb(device + 1, 0x00); //disable interrupts
        outb (0x70, 0x04);
160
161
        outb(0x71, DectoBCD(hours));// change to bcd
        outb(0x70,0x02);
162
163
        outb(0x71, DectoBCD(minutes));
164
        outb(0x70,0x00);
        outb(0x71, DectoBCD(seconds));
sti(); //outb(device + 4, 0x0B); //enable interrupts, rts/dsr set
165
166
      outb (0x70, 0x04);
167
168
      unsigned char newHours = BCDtoDec(inb(0x71));
169
      outb(0x70,0x02);
170
      unsigned char newMinutes = BCDtoDec(inb(0x71));
171
      outb (0x70, 0x00);
      unsigned char newSeconds = BCDtoDec(inb(0x71));
if(newHours != hours || newMinutes != minutes || newSeconds != seconds){
172
173
174
        printf("Your input was invalid\n");
175
        cli(); //outb(device + 1, 0x00); //disable interrupts
176
             outb(0x70,0x04);
177
             outb(0x71, DectoBCD(tempHours));// change to bcd
             outb(0x70,0x02);
178
             outb(0x71, DectoBCD(tempMinutes));
outb(0x70,0x00);
179
180
181
             outb(0x71, DectoBCD(tempSeconds));
```

#### 4.21.1.15 Show All()

```
void Show_All ( )
```

Brief Description: Displays the process name, class, state, suspended status, and priority of all PCB in the ready and blocked queues.

Description: The process name, claas, state, suspend status, and priority of each of he PCB's in the ready and blocked queues.

Definition at line 611 of file userFunctions.c.

```
611 {
612 Show_Ready();
613 Show_Blocked();
614 }
```

#### 4.21.1.16 Show Blocked()

```
void Show_Blocked ( )
```

Brief Description: Displays the process name, class, state, suspended status, and priority of all PCB in the blocked queue.

Description: The process name, claas, state, suspend status, and priority of each of he PCB's in the blocked queue.

Definition at line 760 of file userFunctions.c.

```
760
761
         if (getBlocked() ->head == NULL)
             printf("x1b[32m""\n The Blocked Queue is empty n""\x1b[0m");
762
763
764
        else
765
               int class, state, prior, status;
766
               char name[20];
               char block[] = "\x1B[34m""Blocked Queue: \n""\x1b[0m"; char cname[] = "Name: ";
768
              char cclass[] = "Class: ";
char cstate[] = "State: ";
769
770
               char cstatus[] = "Status: ";
771
772
              char cprior[] = "Priority: ";
773
               char line[] = "\n";
774
775
              printf(block);
776
               //sys_req(WRITE, COM1, block, &check);
777
778
               PCB* pcb = getBlocked()->head;
780
               if(pcb->next == NULL) {
781
                class = pcb->Process_Class;
782
                     strcpy(name,pcb->Process_Name);
                     state = pcb->ReadyState;
status = pcb->SuspendedState;
783
784
                     prior = pcb->Priority;
785
786
787
                     printf(cname);
788
                     printf(name);
789
                     printf(line);
790
791
                     printf(cclass);
                      if(pcb->Process_Class == 0) {
```

```
printf("0");
793
794
795
                      else {
                        printf(itoa(class));
796
797
                        //sys_req(WRITE, COM1, itoa(class), &check);
798
                     printf(line);
800
801
                     printf(cstate);
                      if(pcb->ReadyState == 0) {
  printf("0");
802
803
804
805
                     else
806
                       printf(itoa(state));
807
                        //sys_req(WRITE, COM1, itoa(state), &check);
808
                     printf(line);
809
810
811
                     printf(cstatus);
                      if (pcb->SuspendedState == 0) {
813
                       printf("0");
814
                     else {
815
                        printf(itoa(status));
816
                        //sys_req(WRITE, COM1, itoa(status), &check);
817
818
                     printf(line);
819
820
821
                     printf(cprior);
822
                      if (pcb->Priority == 0) {
                        printf("0");
823
824
                        printf("\n\n");
825
826
                      else {
827
                       printf(itoa(prior));
                       //sys_req(WRITE, COM1, itoa(prior), &check);
printf("\n\n");
828
829
830
831
832
               else {
833
                 while (pcb != NULL) {
                     class = pcb->Process_Class;
    strcpy(name, pcb->Process_Name);
834
835
                          state = pcb->ReadyState;
836
837
                          status = pcb->SuspendedState;
838
                          prior = pcb->Priority;
839
840
                          printf(cname);
                          printf(name);
841
                          printf(line);
842
843
844
                          printf(cclass);
845
                          if (pcb->Process_Class == 0) {
846
                            printf("0");
847
848
                          else
849
                             printf(itoa(class));
850
                            //sys_req(WRITE, COM1, itoa(class), &check);
851
852
                          printf(line);
853
                          printf(cstate);
854
855
                          if (pcb->ReadyState == 0) {
856
                            printf("0");
857
858
                                printf(itoa(state));
859
                            //sys_req(WRITE, COM1, itoa(state), &check);
860
861
862
                          printf(line);
863
864
                          printf(cstatus);
                          if(pcb->SuspendedState == 0) {
  printf("0");
865
866
867
868
869
                            printf(itoa(status));
870
                            //sys_req(WRITE, COM1, itoa(status), &check);
871
                          printf(line);
872
873
                          printf(cprior);
875
                          if(pcb->Priority == 0) {
876
                            printf("0");
                            printf("\n\n");
877
878
879
                          else {
```

## 4.21.1.17 Show PCB()

Brief Description: Displays the process name, class, state, suspended status, and priority of a PCB.

Description: Can except a string as a pointer that is the Process Name. The process name, claas, state, suspend status, and priority of a PCB are displayed. An error check for a valid name occurs.

#### **Parameters**

Process\_Name

Character pointer that matches the name of process

#### Definition at line 536 of file userFunctions.c.

```
537
         if (FindPCB(ProcessName) == NULL)
538
              printf("\x1b[31m""\nERROR: PCB does not exist \n""\x1b[0m");
539
540
         else
541
542
             char name[10];
543
             char cname[] = "Name: ";
             char cclass[] = "Class: ";
char cstate[] = "State: ";
char cstatus[] = "Status: ";
544
545
546
             char cordates[] = "Priority: ";
char line[] = "\n";
547
548
549
             PCB* pcb = FindPCB(ProcessName);
550
             strcpy(name,pcb->Process_Name);
             int class = pcb->Process_Class;
int state = pcb->ReadyState;
int status = pcb->SuspendedState;
551
552
553
             int prior = pcb->Priority;
554
556
              if (name == NULL) {
                  printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
557
558
559
              else
                 printf(cname);
560
561
                  printf(ProcessName);
562
                  printf(line);
563
                  printf(cclass);
                  if(pcb->Process_Class == 0) {
564
                       printf("0");
565
566
567
                  else
568
                       printf(itoa(class));
569
                        //sys_req(WRITE, COM1, itoa(class), &check);
570
571
                  printf(line);
572
                  printf(cstate);
573
                  if (pcb->ReadyState == 0) {
574
                       printf("0");
575
576
                  else {
                       printf(itoa(state));
//sys_req(WRITE, COM1, itoa(state), &check);
577
578
580
                  printf(line);
581
                  printf(cstatus);
```

```
if(pcb->SuspendedState == 0) {
583
                     printf("0");
584
585
                 else
                     printf(itoa(status));
586
                      //sys_req(WRITE, COM1, itoa(status), &check);
587
588
589
                 printf(line);
590
                 printf(cprior);
                  if (pcb->Priority == 0) {
591
                      printf("0");
printf("\n\n");
592
593
594
595
596
                      printf(itoa(prior));
                      //sys_req(WRITE, COM1, itoa(prior), &check); printf("\n\);
597
598
599
                 }
600
             }
601
        }
602 }
```

## 4.21.1.18 Show\_Ready()

```
void Show_Ready ( )
```

Brief Description: Displays the process name, class, state, suspended status, and priority of all PCB in the ready queue.

Description: The process name, claas, state, suspend status, and priority of each of he PCB's in the ready queue.

Definition at line 623 of file userFunctions.c.

```
624
625
626
627
         else
             int class, state, prior, status;
628
           char name[10];

char ready[] = "\x1B[34m""\nReady Queue:\n""\x1B[0m";

char cname[] = "Name: ";

char cclass[] = "Class: ";
629
630
631
632
633
           char cstate[] = "State: ";
           char cstatus[] = "Status: ";
char cprior[] = "Priority: ";
char line[] = "\n";
634
635
636
637
638
           printf(ready);
           //sys_req(WRITE, COM1, ready, &check);
639
640
641
           PCB* pcb = getReady()->head;
642
643
             if(pcb->next == NULL)
644
                  class = pcb->Process_Class;
                  strcpy(name,pcb->Process_Name);
state = pcb->ReadyState;
status = pcb->SuspendedState;
645
646
648
                 prior = pcb->Priority;
649
650
                  printf(cname);
                  printf(name);
651
                  printf(line);
652
653
654
                  printf(cclass);
655
                  if(pcb->Process_Class == 0) {
656
                    printf("0");
657
                  else {
658
659
                    printf(itoa(class));
660
                     //sys_req(WRITE, COM1, itoa(class), &check);
661
662
                  printf(line);
663
664
                  printf(cstate);
                  if(pcb->ReadyState == 0) {
665
                    printf("0");
```

```
667
                 else {
668
669
                   printf(itoa(state));
                   //sys_req(WRITE, COM1, itoa(state), &check);
670
671
                 printf(line);
672
673
674
                 printf(cstatus);
675
                 if(pcb->SuspendedState == 0) {
676
                  printf("0");
677
678
                 else
679
                  printf(itoa(status));
680
                   //sys_req(WRITE, COM1, itoa(status), &check);
681
682
                 printf(line);
683
684
                 printf(cprior);
                 if (pcb->Priority == 0) {
685
                   printf("0");
686
687
                   printf("\n\n");
688
                 else {
689
                  printf(itoa(prior));
//sys_req(WRITE, COM1, itoa(prior), &check);
690
691
                   printf("\n\n");
692
693
694
695
          else {
            while (pcb != NULL) {
696
                    class = pcb->Process_Class;
697
698
                     strcpy(name, pcb->Process_Name);
699
                     state = pcb->ReadyState;
700
                     status = pcb->SuspendedState;
701
                     prior = pcb->Priority;
702
703
                     printf(cname);
                     printf(name);
704
705
                     printf(line);
706
707
                     printf(cclass);
708
                     if(pcb->Process_Class == 0) {
                       printf("0");
709
710
711
                     else
                       printf(itoa(class));
712
713
                       //sys_req(WRITE, COM1, itoa(class), &check);
714
                     printf(line);
715
716
717
                     printf(cstate);
718
                     if (pcb->ReadyState == 0) {
                       printf("0");
719
720
721
                     else {
722
                       printf(itoa(state));
723
                       //sys_req(WRITE, COM1, itoa(state), &check);
724
                     printf(line);
725
726
                     printf(cstatus):
727
728
                     if (pcb->SuspendedState == 0) {
729
                       printf("0");
730
731
                     else {
732
                       printf(itoa(status));
733
                       //sys_req(WRITE, COM1, itoa(status), &check);
734
735
                     printf(line);
736
737
                     printf(cprior);
738
                     if (pcb->Priority == 0) {
                       printf("0");
printf("\n\n");
739
740
741
742
                     else
743
                       printf(itoa(prior));
                       //sys_req(WRITE, COM1, itoa(prior), &check);
printf("\n\n");
744
745
746
                     pcb = pcb->next;
747
748
                }
749
750
        }
751 }
```

## 4.21.1.19 Suspend()

Brief Description: Places a PCD in the suspended state and reinserts it into the appropriate queue.

Description: Can except a string as a pointer that is the Process Name. Places a PCB in the suspended state and reinserts it into the appropriate queue. An error check for valid Process Name.

#### **Parameters**

Process\_Name Character pointer that matches the name of process.

Definition at line 458 of file userFunctions.c.

```
PCB* pcb = FindPCB(ProcessName);
        if (pcb == NULL) {
   printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
460
461
462
463
        else {
             if(pcb->SuspendedState == YES) {
    printf("\x1b[32m""\nThis Process is already SUSPENDED \n""\x1b[0m");
464
466
467
             else if(pcb -> Process_Class == APPLICATION)
468
               pcb->SuspendedState = YES;
            }
469
470
             else
471
                 printf("\x1b[31m""\nERROR: Cannot Alter System Process \n""\x1b[0m");
473 }
```

## 4.21.1.20 toLowercase()

```
char toLowercase ( char c )
```

Description: If a letter is uppercase, it changes it to lowercase. (char)

#### **Parameters**

c Character that is to be changed to its lowercase equivalent

Definition at line 315 of file userFunctions.c.

# 4.21.1.21 Unblock()

Brief Description: Places a PCD in the unblocked state and reinserts it into the correct queue.

Description: Can except a string as a pointer that is the Process Name. The specified PCB will be places in an unblocked state and reinserted into the appropriate queue. An error check for a valid name occurs.

#### **Parameters**

Process\_Name Character pointer that matches the name of process.

Definition at line 984 of file userFunctions.c.

```
PCB* pcb = FindPCB(ProcessName);
if (pcb == NULL) {
985
986
987
        printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
988
      else {
        if(pcb->ReadyState == READY) {
    printf("\xlb[32m""\nThis Process is already in the READY state \n""\xlb[0m");
990
991
992
        else {
   RemovePCB(pcb);
   pcb->ReadyState = READY;
993
994
995
996
            InsertPCB(pcb);
997
998 }
999 }
```

#### 4.21.1.22 Version()

```
void Version ( )
```

Description: Simply returns a char containing "Version: R(module).(the iteration that module is currently on).

No parameters.

Definition at line 308 of file userFunctions.c.

```
308 {
309    printf("Version: R5.2 \n");
310 }
```

#### 4.21.2 Variable Documentation

## 4.21.2.1 AlarmList

```
List AlarmList
```

## Initial value:

```
-{
    .head = NULL,
    .tail = NULL
```

Definition at line 1046 of file userFunctions.c.

# 4.22 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/R1/user Functions.h File Reference

## **Classes**

- struct Alarm
- struct List

#### **Macros**

- #define RED "\x1B[31m"
- #define GRN "\x1B[32m"
- #define YEL "\x1B[33m"
- #define BLU "\x1B[34m"
- #define MAG "\x1B[35m"
- #define CYN "\x1B[36m"
- #define WHT "\x1B[37m"
- #define RESET "\x1B[0m"

# **Typedefs**

- · typedef struct Alarm Alarm
- typedef struct List List

#### **Functions**

- void SetTime (int hours, int minutes, int seconds)
- void GetTime ()
- int DectoBCD (int Decimal)
- · void clear ()
- char \* itoa (int num)
- void SetDate (int day, int month, int millennium, int year)
- int BCDtoDec (int BCD)
- · void GetDate ()
- void Version ()
- void Help (char \*request)
- void printf (char msg[])
- int EdgeCase (char \*pointer)
- char toLowercase (char c)
- void Suspend (char \*ProcessName)
- void Resume (char \*ProcessName)
- void Set\_Priority (char \*ProcessName, int Priority)
- void Show PCB (char \*ProcessName)
- void Show All ()
- void Show\_Ready ()
- void Show\_Blocked ()
- void Create\_PCB (char \*ProcessName, int Priority, int Class)
- void Delete\_PCB (char \*ProcessName)
- void Block (char \*ProcessName)
- void Unblock (char \*ProcessName)
- void loader ()
- void loadr3 (char \*name, u32int func)
- void yield ()
- void loaderinfinite ()
- List \* getList ()
- void loaderalarm ()

# 4.22.1 Function Documentation

## 4.22.1.1 BCDtoDec()

```
int BCDtoDec (
          int BCD )
```

Description: Changes binary number to decimal numbers.

#### **Parameters**

value Binary number to be changed to decimal

Definition at line 82 of file userFunctions.c.

#### 4.22.1.2 Block()

Brief Description: Places a PCD in the blocked state and reinserts it into the correct queue.

Description: Can except a string as a pointer that is the Process Name. The specified PCB will be places in a blocked state and reinserted into the appropriate queue. An error check for a valid name occurs.

#### **Parameters**

Process\_Name | Character pointer that matches the name of process.

Definition at line 958 of file userFunctions.c.

```
958
959
      PCB* pcb = FindPCB(ProcessName);
      if (pcb == NULL) {
960
       printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
961
962
963
      if(pcb->ReadyState == BLOCKED) {
    printf("\x1b[32m""\nThis Process is already BLOCKED \n""\x1b[0m");
964
965
966
967
       else
968
        RemovePCB(pcb);
969
          pcb->ReadyState = BLOCKED;
970
          InsertPCB(pcb);
971
972
     }
973 }
```

## 4.22.1.3 Create\_PCB()

Brief Description: Calls SetupPCB() and inserts PCB into appropriate queue.

Description: Can except a string as a pointer that is the Process Name. Can accept two integers, Priority and Class. SetupPCB() will be called and the PCB will be inserted into the appropriate queue. An error check for unique and valid Process Name, an error check for valid process class, and an error check for process priority.

#### **Parameters**

| Process_Name | Character pointer that matches the name of process. |
|--------------|---|
| Priority     | integer that matches the priority number.           |
| Class        | integer that matches the class number.              |

Definition at line 901 of file userFunctions.c.

```
if (FindPCB(ProcessName) == NULL) {
903
       if(Priority >= 0 && Priority < 10){</pre>
904
         if(Class == 0 || Class == 1){
           PCB* pcb = SetupPCB(ProcessName, Class, Priority);
905
906
           InsertPCB(pcb);
907
           printf("\x1b[31m""\nERROR: Not a valid Class \n""\x1b[0m");
908
910
       } else{
         printf("\x1b[31m""\nERROR: Not a valid Priority \n""\x1b[0m");
911
912
913
     } else{
       printf("\x1b[31m""\nERROR: This Process Name already exists \n""\x1b[0m");
914
915
916 }
```

## 4.22.1.4 DectoBCD()

Description: Changes decimal numbers to binary numbers.

## **Parameters**

Decimal number to be changed to binary

Definition at line 89 of file userFunctions.c.

```
89 {
90     return (((Decimal/10) « 4) | (Decimal % 10));
91 }
```

#### 4.22.1.5 Delete\_PCB()

Brief Description: Removes PCB from appropriate queue and frees all associated memory.

Description: Can except a string as a pointer that is the Process Name. Removes PCB from the appropriate queue and then frees all associated memory. An error check to make sure process name is valid.

#### **Parameters**

Process\_Name | Character pointer that matches the name of process.

Definition at line 927 of file userFunctions.c.

```
PCB* pcb = FindPCB(ProcessName);
      if (pcb == NULL) {
   printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
929
930
931
      else if(strcmp(pcb->Process_Name, "InfProc") == 0) {
   if(pcb->SuspendedState == YES) {
932
933
934
                   RemovePCB (pcb);
935
                  FreePCB (pcb);
936
937
                  printf("\x1b[31m""\nERROR:This process cannot be deleted unless it is in the suspended
938
       state\n""\x1b[0m");
939
      else if (pcb -> Process_Class == SYSTEM) {
    printf("\x1b[31m""\nERROR: System Processes cannot be deleted from the system. \n""\x1b[0m");
940
941
942
      }
      else {
943
944
            RemovePCB (pcb);
945
             FreePCB (pcb);
947 }
```

# 4.22.1.6 EdgeCase()

Description: Compares pointer char to validate if it is a number or not.

## **Parameters**

Compares pointer char to validate if it is a number or not.

Definition at line 110 of file userFunctions.c.

```
111
      int valid = 0;
      if (strcmp(pointer, "00") == 0) {
112
113
       valid = 1;
return valid;
114
115
116
      else if (strcmp(pointer, "0") == 0) {
      valid = 1;
return valid;
117
118
119
     else
120
       int j;
121
```

```
122
       valid = 0;
123
       for(j = 0; j <= 99; j++)
124
         if(strcmp(pointer,itoa(j)) == 0)
125
           valid = 1;
126
127
       if (valid == 0)
128
         return valid;
129
130
131
      return valid;
132 }
```

## 4.22.1.7 GetDate()

```
void GetDate ( )
```

Description: Returns the full date back to the user in decimal form.

No parameters.

Definition at line 272 of file userFunctions.c.

```
272
273
      outb(0x70,0x07);
unsigned char day = BCDtoDec(inb(0x71));
274
275
276
        outb(0x70,0x08);
        unsigned char month = BCDtoDec(inb(0x71));
278
        outb(0x70,0x32);
        unsigned char millennium = BCDtoDec(inb(0x71));
char msg[2] = "-";
char msg3[10] = "Date: ";
279
280
281
282
        printf(msg3);
283
284
         printf(itoa(day));
        //sys_req(WRITE, COM1, itoa(day), &check);
285
286
        printf(msg);
287
        printf(itoa(month));
        //sys_req(WRITE, COM1, itoa(month), &check);
288
289
        printf(msg);
290
        printf(itoa(millennium));
         //sys_req(WRITE, COM1, itoa(millennium), &check);
291
292
      outb (0x70, 0x09);
293
      if(BCDtoDec(inb(0x71)) == 0){
294
         printf("00");
295
        //sys_req(WRITE, COM1, "00", &check);
296
297
298
             unsigned char year = BCDtoDec(inb(0x71));
             printf(itoa(year));
//sys_req(WRITE, COM1, itoa(year), &check);
299
300
301
      }
302
        printf("\n");
303 }
```

## 4.22.1.8 GetTime()

```
void GetTime ( )
```

Description: retrieve and return the time values for hours, minutes, and seconds form the clock register using inb(Port,address).

No parameters.

Definition at line 191 of file userFunctions.c.

191 192

```
193
        int hour;
194
        int minute;
195
        int second;
196
        outb (0x70, 0x04);
197
        unsigned char hours = inb(0x71);
198
        outb (0x70,0x02);
199
        unsigned char minutes = inb(0x71);
200
        outb (0x70, 0x00);
201
        unsigned char seconds = inb(0x71);
202
        char msg1[2] = ":";
        char msg2[10] = "Time: ";
203
204
        printf(msq2);
205
        hour = BCDtoDec(hours);
206
        printf(itoa(hour));
207
        //sys_req(WRITE, COM1, itoa(hour), &check);
208
        printf(msg1);
        minute = BCDtoDec(minutes);
209
        printf(itoa(minute));
210
211
        //sys_req(WRITE, COM1, itoa(minute), &check);
        printf(msg1);
213
        second = BCDtoDec(seconds);
214
        printf(itoa(second));
      //sys_req(WRITE, COM1, itoa(second), &check);
printf("\n");
215
216
```

## 4.22.1.9 Help()

Brief Description: Gives helpful information for one of the functions

Description: Can except a string as a pointer, if the pointer is null then the function will print a complete list of avaliable commands to the console. If the pointer is a avaliable commands then instructions on how to use the command will be printed. If the command does not exist then a message explaining that it is not a valid command will be displayed.

# **Parameters**

request Character pointer that matches the name of the function that you need help with.

Definition at line 332 of file userFunctions.c.

```
333
            if (request[0] == ' \setminus 0')
334
                  //removed for R3/R4 from active command list
                 //\n createPCB \n block \n unblock 
//\n heap alloc \n free
335
                  //\n heap
336
                                                                             empty
                //(n neap alloc \n free empty
printf("\n to chain commands and parameters, please use \"-\" between keywords \n");
printf("\n getDate setDate \n getTime setTime \n version suspend \n resume
riority \n showPCB showAll \n showReady showBlocked \n deletePCB shutdown \n alarm
clear \n loadr3 infinte \n showFree showAlloc \n\n");
337
         339
           else if (strcmp(request, "GetDate") == 0) {
340
                printf("\n getDate returns the current date that is loaded onto the operating system.\n");
341
342
          else if (strcmp(request, "SetDate") == 0) {
    printf("\n setDate allows the user to reset the correct date into the system, as follows
setDate-"BLU"day"RESET"-"BLU"month"RESET"-"BLU"year"RESET".\n Time must be inputed as a two digit
343
344
          number, Example 02 or 00");
345
346
           else if (strcmp(request, "GetTime") == 0) {
                 printf("\n") getTime returns the current time as hours, minutes, seconds that is loaded onto the
347
          operating system.\n");
348
349
           else if (strcmp(request, "SetTime") == 0) {
          printf("\n setTime allows the user to reset the correct time into the system, as follows setTime-"BLU"hour"RESET"-"BLU"minute"RESET"-"BLU"second"RESET".\n Time must be inputed as a two digit
350
          number, Example 02 or 00");
351
```

```
else if (strcmp(request, "Version") == 0) {
            printf("\n version returns the current operating software version that the system is
       running.\n");
354
        else if (strcmp(request, "infinte") == 0) {
355
           printf("\n infinite Loads the infinite process into the ready queue.\n");
356
357
        else if (strcmp(request, "loadr3") == 0)
358
359
            printf("\n loadr3 Loads in all five of the R3 test processes in a suspended state into the
       queue.\n");
360
        else if (strcmp(request, "alarm") == 0)
361
           printf("\n alarm creates a user specified alarm with a user set message and time
362
       alarm-MSG-hour-minute-second.\n");
363
       else if (strcmp(request, "clear") == 0) {
    printf("\n clear erases the console of all typed commands and refreshes it with just the command
list.\n");
364
365
366
367
368
      else if(strcmp(request, "shutdown") == 0)
369
       printf("\n shutdown shuts down the system.\n");
370
371
372
373
371
                R2 Commands
375
       else if(strcmp(request, "suspend") == 0) {
376
377
            printf("\n Suspend takes in the name of a PCB (suspend-NAME) then places it into the suspended
       state and reinserts it into the correct queue.\n");
378
      else if(strcmp(request, "resume") == 0) { printf("\n Resume takes in the name of a PCB (resume-NAME) then removes it from the suspended
379
380
      state and adds it to the correct queue. \n");
381
382
      else if(strcmp(request, "setPriority") == 0) {
            printf("\n SetPriority takes in the name of a PCB and the priority (setPrioriry-NAME-PRIORITY)
383
      it needs to be set to then reinstates the specified PCB into a new location by priority.\n^n);
384
      else if(strcmp(request, "showPCB") == 0) {
385
           printf("\n ShowPCB takes in the name of a PCB and returns all the associated attributes to the
386
       user.\n");
387
      else if(strcmp(request, "showAll") == 0) { printf("\n ShowAll takes no parameters but returns all PCB's that are currently in any of the
388
389
       queues.\n");
390
      else if(strcmp(request, "showReady") == 0) {
    printf("\n ShowReady takes in no parameters but returns all PCB's and there attributes that
391
392
       currently are in the ready state.\n");
393
      else if(strcmp(request, "showBlocked") == 0) {
    printf("\n ShowBlocked takes in no parameters but returns all PCB's and there attributes that
394
395
       currently are in the blocked state.\n");
396
397
398 /******* R2 Temp Commands
       ************************
      else if(strcmp(request, "deletePCB") == 0) {
          printf("\n DeletePCB takes in the process_name (deletePCB-NAME) then deletes it from the queue
399
400
       and free's all the memory that was previously allocated to the specified PCB.\n");
401
402
      //removed for R3/R4 from active command list
403
      else if(strcmp(request, "createPCB") == 0) {
404
           printf("\n CreatePCB takes in the process_name, process_class, and
405
       process_priority.(createPCB-NAME-PRIORITY-CLASS) Then assigns this new process into the correct
406
      else if(strcmp(request, "block") == 0) { printf("\n Block takes in the process_name (block-NAME) then sets it's state to blocked and
407
408
       reinserts it back into the correct queue. \n");
409
410
      else if(strcmp(request, "unblock") == 0) {
411
            \texttt{printf("} \\ \texttt{N} \  \, \texttt{Unblock takes in the process\_name (unblock-NAME)} \  \, \texttt{then sets it's state to ready and} \\
       reinserts it back into the correct queue.\n");
412
      1
      */
413
414
416 /****** R5 Temp Commands
                 **************
      // else if(strcmp(request, "heap") == 0) {
417
       // printf("\n heap initializes the memory heap for the entire system.\n");
418
```

```
419
     // else if(strcmp(request, "alloc") == 0) {
420
421
       // printf("\n alloc allocates the specified amount of memory to the specific process
      (alloc-process_name-size).\n");
422
     // else if(strcmp(request, "free") == 0) {
423
       // printf("\n free frees the specified memory at the address given (free-address).\n");
424
425
426
     // else if(strcmp(request,"empty") == 0) {
427
       // printf("\n isempty returns true or false depending on if the heap has allocated memory.\n");
     // }
428
429 /****** R5 Commands
430
     else if(strcmp(request, "showFree") == 0) {
431
          printf("\n showfree shows all the free blocks available within the heap list.\n");
432
     else if(strcmp(request, "showAlloc") == 0) {
433
           printf("\n showAlloc shows all the allocated blocks within the heap list.\n");
434
     }
435
436
437
438
           printf("\x1b[31m""\\nThe requested command does not exist please refer to the Help function for a
439
      full list of commands.\n""\x1b[0m");
440
     }
441 }
```

#### 4.22.1.10 itoa()

```
char* itoa (
          int num )
```

Description: An integer is taken and seperated into individual chars and then all placed into a character array. Adapted from geeksforgeeks.org.

#### **Parameters**

```
num integer to be put into array Title: itoa Author: Neha Mahajan Date: 29 May, 2017 Availability: https://www.geeksforgeeks.org/implement-itoa/
```

Definition at line 51 of file userFunctions.c.

```
52
       int i, j, k, count;
53
       i = num;
       j = 0;
       count = 0;
       while(i) { // count number of digits
57
          count++;
58
           i /= 10;
59
60
       char* arr1;
63
       char arr2[count];
64
       arr1 = (char*)sys_alloc_mem(count); //memory allocation
65
66
       while(num){ // seperate last digit from number and add ASCII
        arr2[++j] = num%10 + '0';
68
           num /= 10;
69
      }
70
       for (k = 0; k < j; k++) \{ // \text{ reverse array results} \}
71
          arr1[k] = arr2[j-k];
72
74
       arr1[k] = ' \setminus 0';
75
       sys_free_mem(arr1);
76
       return(char*)arr1;
77 }
```

#### 4.22.1.11 Resume()

Brief Description: Places a PCD in the not suspended state and reinserts it into the appropriate queue.

Description: Can except a string as a pointer that is the Process Name. Places a PCB in the not suspended state and reinserts it into the appropriate queue. An error check for valid Process Name.

#### **Parameters**

| Process_Name | Character pointer that matches the name of process. |
|--------------|---|
|--------------|---|

Definition at line 484 of file userFunctions.c.

```
PCB* pcb = FindPCB(ProcessName);
486
        if (pcb == NULL)
         printf(RED"\nERROR: Not a valid process name \n"RESET);
487
488
489
       else {
            if (pcb->SuspendedState == NO)
490
491
                printf(GRN"\nThis Process is already in the NONSUSPENDED state \n"RESET);
492
493
            else if(pcb -> Process_Class == APPLICATION)
494
               pcb->SuspendedState = NO;
            }
495
496
            else
497
               printf("\x1b[31m""\nERROR: Cannot Alter System Process \n""\x1b[0m");
498
499 }
```

## 4.22.1.12 Set\_Priority()

Brief Description: Sets PCB priority and reinserts the process into the correct place in the correct queue.

Description: Can except a string as a pointer that is the Process Name. Can accept and integer than is the Priority. Sets a PCB's priority and reinserts the process into the correct place in the correct queue. An error check for valid Process Name and an error check for a valid priority 1 - 9.

#### **Parameters**

| Process_Name | Character pointer that matches the name of process. |
|--------------|---|
| Priority     | integer that matches the priority number.           |

#### Definition at line 511 of file userFunctions.c.

```
511
512 PCB* pcb = FindPCB(ProcessName);
513 if (pcb == NULL) {
    printf("\xlb[3lm""\nERROR: Not a valid process name \n""\xlb[0m");
515 }
516 else if (Priority >= 10) {
    printf("\xlb[3lm""\nERROR: Not a valid Priority \n""\xlb[0m");
518 }
519 else if (pcb -> Process_Class == APPLICATION) {
```

```
520 RemovePCB(pcb);
521 pcb->Priority = Priority;
522 InsertPCB(pcb);
523 }
524 else
525 printf("\xlb[3lm""\nERROR: Cannot Alter System Process \n""\xlb[0m");
526 }
```

# 4.22.1.13 SetDate()

```
void SetDate (
    int day,
    int month,
    int millennium,
    int year )
```

Description: Sets the date register to the new values that the user inputed, all values must be inputed as Set 

Dime(day, month, millenial, year).

#### **Parameters**

| day       | Integer to be set in the Day position       |
|-----------|---|
| month     | Integer to be set in the Month position     |
| millenial | Integer to be set in the Millenial position |
| year      | Integer to be set in the Year position      |

### Definition at line 225 of file userFunctions.c.

```
225
226
       outb (0x70, 0x07);
       int tempDay = BCDtoDec(inb(0x71));
outb(0x70,0x08);
int tempMonth = BCDtoDec(inb(0x71));
227
228
229
230
       outb (0x70, 0x32);
231
       int tempMillennium = BCDtoDec(inb(0x71));
       outb(0x70,0x09);
int tempYear = BCDtoDec(inb(0x71));
232
233
234
      cli();
235
        outb(0x70,0x07);
236
         outb(0x71,DectoBCD (day));
237
         outb (0x70, 0x08);
         outb(0x71,DectoBCD (month));
238
        outb(0x70,0x32);
outb(0x71,DectoBCD (millennium));
239
240
        outb(0x70,0x09);
241
         outb(0x71,DectoBCD (year));
242
243
         sti();
2.44
       outb (0x70, 0x07);
       unsigned char newDay = BCDtoDec(inb(0x71));
245
246
       outb (0x70, 0x08);
247
       unsigned char newMonth = BCDtoDec(inb(0x71));
248
       outb(0x70,0x32);
249
       unsigned char newMillennium = BCDtoDec(inb(0x71));
250
       outb (0x70, 0x09);
       unsigned char newYear = BCDtoDec(inb(0x71));
if(newDay != day || newMonth != month || newMillennium != millennium || newYear != year){
  printf("Your input was invalid\n");
251
252
253
254
         cli();
255
             outb(0x70,0x07);
              outb(0x71,DectoBCD (tempDay));
256
              outb(0x70,0x08);
2.57
             outb(0x70,0x00),
outb(0x71,DectoBCD (tempMonth));
outb(0x70,0x32);
258
259
260
             outb(0x71,DectoBCD (tempMillennium));
              outb (0x70, 0x09);
261
262
              outb(0x71,DectoBCD (tempYear));
263
              sti();
       }
264
265
       else
```

```
266 printf("Date Setn");
267 }
```

# 4.22.1.14 SetTime()

```
void SetTime (
                int hours,
                int minutes,
                int seconds )
```

Description: sets the time register to the new values that the user inputed, all values must be inputed as SetTime(← Hours, Minutes, Seconds).

#### **Parameters**

| hours   | Integer to be set in the Hour position    |
|---------|---|
| minutes | Integer to be set in the Minutes position |
| seconds | Integer to be set in the Seconds position |

Definition at line 152 of file userFunctions.c.

```
152
153
      outb (0x70, 0x04);
154
      unsigned char tempHours = BCDtoDec(inb(0x71));
155
      outb(0x70,0x02);
156
      unsigned char tempMinutes = BCDtoDec(inb(0x71));
157
      outb (0x70, 0x00);
      unsigned char tempSeconds = BCDtoDec(inb(0x71));
cli(); //outb(device + 1, 0x00); //disable interrupts
158
159
        outb(0x70,0x04);
160
161
        outb(0x71, DectoBCD(hours));// change to bcd
162
        outb (0x70, 0x02);
163
        outb(0x71, DectoBCD(minutes));
164
        outb (0x70, 0x00);
        outb(0x71, DectoBCD(seconds));
sti(); //outb(device + 4, 0x0B); //enable interrupts, rts/dsr set
165
166
      outb (0x70, 0x04);
167
      unsigned char newHours = BCDtoDec(inb(0x71));
169
      outb (0x70, 0x02);
170
      unsigned char newMinutes = BCDtoDec(inb(0x71));
171
      outb (0x70, 0x00);
      unsigned char newSeconds = BCDtoDec(inb(0x71));
if(newHours != hours || newMinutes != minutes || newSeconds != seconds){
172
173
        printf("Your input was invalid\n");
174
175
        cli(); //outb(device + 1, 0x00); //disable interrupts
176
177
             outb(0x70,0x04);
             outb(0x71, DectoBCD(tempHours));// change to bcd
             outb(0x70,0x02);
178
179
             outb(0x71, DectoBCD(tempMinutes));
180
             outb(0x70,0x00);
181
             outb(0x71, DectoBCD(tempSeconds));
182
             sti(); //outb(device + 4, 0x0B); //enable interrupts, rts/dsr set
183
184
      else
        printf("Time Set\n");
185
```

# 4.22.1.15 Show\_All()

```
void Show All (
```

Brief Description: Displays the process name, class, state, suspended status, and priority of all PCB in the ready and blocked queues.

Description: The process name, claas, state, suspend status, and priority of each of he PCB's in the ready and blocked queues.

Definition at line 611 of file userFunctions.c.

```
611 {
612 Show_Ready();
613 Show_Blocked();
614 }
```

#### 4.22.1.16 Show Blocked()

```
void Show_Blocked ( )
```

Brief Description: Displays the process name, class, state, suspended status, and priority of all PCB in the blocked queue.

Description: The process name, claas, state, suspend status, and priority of each of he PCB's in the blocked queue.

Definition at line 760 of file userFunctions.c.

```
761
762
763
764
        else
765
              int class, state, prior, status;
              char name[20]; char block[] = "\x1B[34m""Blocked Queue: \n""\x1b[0m";
766
767
              char cname[] = "Name: ";
char cclass[] = "Class: ";
768
769
              char cstate[] = "State: ";
770
              char cstatus[] = "Status: ";
char cprior[] = "Priority: ";
char line[] = "\n";
771
772
773
774
775
              printf(block);
776
              //sys_req(WRITE, COM1, block, &check);
778
              PCB* pcb = getBlocked()->head;
779
780
              if(pcb->next == NULL) {
                class = pcb->Process_Class;
781
                    strcpy(name,pcb->Process_Name);
state = pcb->ReadyState;
782
783
784
                     status = pcb->SuspendedState;
785
                    prior = pcb->Priority;
786
787
                     printf(cname);
788
                     printf(name);
789
                    printf(line);
790
791
                     printf(cclass);
792
                     if(pcb->Process_Class == 0) {
793
                       printf("0");
794
795
                     else
796
                       printf(itoa(class));
797
                       //sys_req(WRITE, COM1, itoa(class), &check);
798
799
                     printf(line);
800
801
                     printf(cstate);
                     if (pcb->ReadyState == 0) {
802
803
                       printf("0");
804
805
                     else {
                       printf(itoa(state));
806
                       //sys_req(WRITE, COM1, itoa(state), &check);
807
808
809
                     printf(line);
810
811
                     printf(cstatus);
                     if(pcb->SuspendedState == 0) {
812
                      printf("0");
813
814
815
                     else {
```

```
printf(itoa(status));
817
                        //sys_req(WRITE, COM1, itoa(status), &check);
818
                      printf(line);
819
820
821
                      printf(cprior);
                      if (pcb->Priority == 0) {
822
823
                        printf("0");
                        printf("\n\n");
824
825
                      else {
826
                       printf(itoa(prior));
//sys_req(WRITE, COM1, itoa(prior), &check);
827
828
829
                        printf("\n\n");
830
831
               else (
832
                 while (pcb != NULL) {
833
                     class = pcb->Process_Class;
834
835
                        strcpy(name,pcb->Process_Name);
                          state = pcb->ReadyState;
status = pcb->SuspendedState;
836
837
                          prior = pcb->Priority;
838
839
840
                          printf(cname);
841
                          printf(name);
842
                          printf(line);
843
844
                          printf(cclass);
                          if(pcb->Process_Class == 0) {
845
                            printf("0");
846
847
848
849
                             printf(itoa(class));
850
                            //sys_req(WRITE, COM1, itoa(class), &check);
851
                          printf(line);
852
853
854
                          printf(cstate);
855
                          if(pcb->ReadyState == 0) {
856
                            printf("0");
857
858
                          else {
                                printf(itoa(state));
859
860
                            //sys_req(WRITE, COM1, itoa(state), &check);
861
862
                          printf(line);
863
                          printf(cstatus);
864
                          if(pcb->SuspendedState == 0) {
865
                           printf("0");
866
867
868
                          else {
869
                            printf(itoa(status));
                            //sys_req(WRITE, COM1, itoa(status), &check);
870
871
872
                          printf(line);
873
874
                          printf(cprior);
                          if(pcb->Priority == 0) {
  printf("0");
875
876
                            printf("\n\n");
877
879
                          else {
                            printf(itoa(prior));
880
                            //sys_req(WRITE, COM1, itoa(prior), &check); printf("\n\");
881
882
883
884
                     pcb = pcb->next;
885
                 }
886
             }
887
888 }
```

# 4.22.1.17 Show\_PCB()

Brief Description: Displays the process name, class, state, suspended status, and priority of a PCB.

Description: Can except a string as a pointer that is the Process Name. The process name, claas, state, suspend status, and priority of a PCB are displayed. An error check for a valid name occurs.

#### **Parameters**

*Process\_Name* Character pointer that matches the name of process

### Definition at line 536 of file userFunctions.c.

```
if (FindPCB(ProcessName) == NULL) {
   printf("\x1b[31m""\nERROR: PCB does not exist \n""\x1b[0m");
537
538
539
540
        else
542
             char name[10];
             char cname[] = "Name: ";
543
             char cclass[] = "Class: ";
char cstate[] = "State: ";
544
545
546
             char cstatus[] = "Status: ";
             char cprior[] = "Priority: ";
char line[] = "\n";
547
548
             PCB* pcb = FindPCB(ProcessName);
549
550
             strcpy(name,pcb->Process_Name);
             int class = pcb->Process_Class;
int state = pcb->ReadyState;
int status = pcb->SuspendedState;
551
552
553
554
             int prior = pcb->Priority;
555
556
             if (name == NULL) {
                  printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
557
558
559
560
                 printf(cname);
561
                  printf(ProcessName);
562
                  printf(line);
                  printf(cclass);
563
                  if(pcb->Process_Class == 0) {
564
565
                      printf("0");
566
567
                  else
568
                      printf(itoa(class));
                       //sys_req(WRITE, COM1, itoa(class), &check);
569
570
571
                  printf(line);
572
                  printf(cstate);
573
                  if(pcb->ReadyState == 0) {
574
                      printf("0");
575
576
                  else {
                      printf(itoa(state));
578
                       //sys_req(WRITE, COM1, itoa(state), &check);
579
                  printf(line);
580
                  printf(cstatus);
581
                  if(pcb->SuspendedState == 0) {
   printf("0");
582
583
585
586
                      printf(itoa(status));
587
                       //sys_req(WRITE, COM1, itoa(status), &check);
588
                  printf(line);
589
590
                  printf(cprior);
591
                  if (pcb->Priority == 0) {
592
                      printf("0");
                      printf("\n\n");
593
594
                  else {
595
                      printf(itoa(prior));
597
                       //sys_req(WRITE, COM1, itoa(prior), &check);
598
                      printf("\n\n");
599
                 }
600
             }
        }
601
602 }
```

# 4.22.1.18 Show\_Ready()

```
void Show_Ready ( )
```

Brief Description: Displays the process name, class, state, suspended status, and priority of all PCB in the ready queue.

Description: The process name, claas, state, suspend status, and priority of each of he PCB's in the ready queue.

Definition at line 623 of file userFunctions.c.

```
623
624
         if (getReady()->head == NULL)
625
               printf("\x1b[32m""\n The Ready Queue is empty \n""\x1b[0m");
626
627
         else
628
             int class, state, prior, status;
           char name[10];
char ready[] = "\x1B[34m""\nReady Queue:\n""\x1B[0m";
char cname[] = "Name: ";
629
630
631
           char chame() = "Name.";
char cclass[] = "Class: ";
char cstate[] = "State: ";
char cstatus[] = "Status: ";
char cprior[] = "Priority: ";
char line[] = "\n";
633
634
635
636
637
638
           printf(ready);
639
           //sys_req(WRITE, COM1, ready, &check);
640
           PCB* pcb = getReady()->head;
641
642
             if(pcb->next == NULL)
643
644
                  class = pcb->Process_Class;
645
                  strcpy(name,pcb->Process_Name);
646
                  state = pcb->ReadyState;
                  status = pcb->SuspendedState;
prior = pcb->Priority;
647
648
649
650
                  printf(cname);
651
                  printf(name);
652
                  printf(line);
653
                  printf(cclass);
654
655
                  if(pcb->Process_Class == 0) {
656
                    printf("0");
657
658
                  else
659
                    printf(itoa(class));
                     //sys_req(WRITE, COM1, itoa(class), &check);
660
661
                  printf(line);
662
663
664
                  printf(cstate);
665
                   if (pcb->ReadyState == 0) {
                    printf("0");
666
667
668
                  else
669
                    printf(itoa(state));
670
                     //sys_req(WRITE, COM1, itoa(state), &check);
671
                  printf(line);
672
673
                  printf(cstatus);
674
                  if (pcb->SuspendedState == 0) {
676
                    printf("0");
677
678
                  else
                     printf(itoa(status));
679
                     //sys_req(WRITE, COM1, itoa(status), &check);
680
681
682
                  printf(line);
683
                  printf(cprior);
684
685
                   if (pcb->Priority == 0) {
                    printf("0");
686
                     printf("\n\n");
687
688
689
                  else
690
                    printf(itoa(prior));
                     //sys_req(WRITE, COM1, itoa(prior), &check); printf("\n\");
691
692
693
           }
```

```
695
          else
                {
696
            while (pcb != NULL) {
697
                    class = pcb->Process_Class;
                    strcpy(name,pcb->Process_Name);
698
699
                    state = pcb->ReadyState;
status = pcb->SuspendedState;
700
                    prior = pcb->Priority;
701
702
703
                     printf(cname);
                     printf(name);
704
705
                    printf(line);
706
707
                     printf(cclass);
708
                     if (pcb->Process_Class == 0) {
709
                      printf("0");
710
711
                     else {
                      printf(itoa(class));
712
                       //sys_req(WRITE, COM1, itoa(class), &check);
713
714
                     printf(line);
715
716
717
                     printf(cstate);
718
                     if(pcb->ReadyState == 0) {
719
                      printf("0");
720
721
                      printf(itoa(state));
722
723
                       //sys_req(WRITE, COM1, itoa(state), &check);
724
725
                     printf(line);
726
727
                     printf(cstatus);
728
                     if(pcb->SuspendedState == 0) {
                      printf("0");
729
730
731
                     else {
732
                      printf(itoa(status));
733
                       //sys_req(WRITE, COM1, itoa(status), &check);
734
735
                     printf(line);
736
                     printf(cprior);
737
                     if (pcb->Priority == 0) {
738
                      printf("0");
740
                      printf("\n\n");
741
742
                     else {
                      printf(itoa(prior));
743
744
                       //sys_req(WRITE, COM1, itoa(prior), &check);
                      printf("\n\n");
745
746
747
                    pcb = pcb->next;
748
               }
            }
749
        }
750
```

#### 4.22.1.19 Suspend()

Brief Description: Places a PCD in the suspended state and reinserts it into the appropriate queue.

Description: Can except a string as a pointer that is the Process Name. Places a PCB in the suspended state and reinserts it into the appropriate queue. An error check for valid Process Name.

#### **Parameters**

Process Name | Character pointer that matches the name of process.

Definition at line 458 of file userFunctions.c.

```
PCB* pcb = FindPCB(ProcessName);
459
        if (pcb == NULL) {
  printf("\xlb[3lm""\nERROR: Not a valid process name \n""\xlb[0m");
460
461
462
463
464
            if(pcb->SuspendedState == YES) {
465
               printf("\x1b[32m""\nThis Process is already SUSPENDED \n""\x1b[0m");
466
            else if(pcb -> Process_Class == APPLICATION)
467
               pcb->SuspendedState = YES;
468
469
470
                printf("\x1b[31m""\nERROR: Cannot Alter System Process \n""\x1b[0m");
471
472
473 }
```

#### 4.22.1.20 toLowercase()

```
char toLowercase ( char c )
```

Description: If a letter is uppercase, it changes it to lowercase. (char)

#### **Parameters**

c Character that is to be changed to its lowercase equivalent

Definition at line 315 of file userFunctions.c.

```
315 {
316    if((c >= 65) && (c <= 90)) {
317         c = c + 32;
318    }
319    return c;
320 }
```

# 4.22.1.21 Unblock()

Brief Description: Places a PCD in the unblocked state and reinserts it into the correct queue.

Description: Can except a string as a pointer that is the Process Name. The specified PCB will be places in an unblocked state and reinserted into the appropriate queue. An error check for a valid name occurs.

#### **Parameters**

*Process Name* Character pointer that matches the name of process.

Definition at line 984 of file userFunctions.c.

```
984 {
985    PCB* pcb = FindPCB(ProcessName);
986    if (pcb == NULL) {
987        printf("\x1b[31m""\nERROR: Not a valid process name \n""\x1b[0m");
```

```
988  }
989  else {
990    if(pcb->ReadyState == READY) {
991         printf("\xlb[32m""\nThis Process is already in the READY state \n""\xlb[0m");
992    }
993    else    {
994         RemovePCB(pcb);
995         pcb->ReadyState = READY;
996         InsertPCB(pcb);
997    }
998  }
999 }
```

#### 4.22.1.22 Version()

```
void Version ( )
```

Description: Simply returns a char containing "Version: R(module).(the iteration that module is currently on).

No parameters.

Definition at line 308 of file userFunctions.c.

```
309 printf("Version: R5.2 \n");
310 }
```

# 4.23 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/sys\_proc \_loader.c File Reference

```
#include <stdint.h>
#include <string.h>
#include <system.h>
#include <core/serial.h>
#include <core/io.h>
#include "mpx_supt.h"
#include "R1/userFunctions.h"
#include "procsr3.h"
#include "R1/comHand.h"
#include "sys_proc_loader.h"
```

# **Functions**

- · void sysLoader ()
- · void loadSysProc (char \*name, u32int func, int priority)
- void InfiniteProc ()
- void AlarmProc ()

# 4.24 D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/modules/sys\_proc \_loader.h File Reference

# **Functions**

- void sysLoader ()
- void loadSysProc (char \*name, u32int func, int priority)
- void InfiniteProc ()
- void AlarmProc ()

# Index

```
Alarm. 5
                                                      D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/tables.c,
AlarmList
    userFunctions.c, 57
                                                      D:/GITHUB/CS 450 RunTime Terror/mpx core/kernel/mem/heap.c,
atoi
                                                               27
                                                      D:/GITHUB/CS\_450\_RunTime\_Terror/mpx\_core/kernel/mem/paging.c,
    string.c, 29
    string.h, 18
                                                      D:/GITHUB/CS 450 RunTime Terror/mpx core/lib/string.c,
BCDtoDec
    userFunctions.c, 41
                                                      D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/mpx_supt.c,
    userFunctions.h, 59
Block
                                                      D:/GITHUB/CS 450 RunTime Terror/mpx core/modules/mpx supt.h,
    userFunctions.c, 41
                                                               35
    userFunctions.h, 59
                                                      D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R1/comHand.h,
CMCB, 5
                                                      D:/GITHUB/CS 450 RunTime Terror/mpx core/modules/R1/userFunctio
comHand
    comHand.h, 36
                                                      D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/R1/userFunctio
comHand.h
    comHand, 36
                                                      D:/GITHUB/CS_450_RunTime_Terror/mpx_core/modules/sys_proc_loade
context, 6
                                                               75
Create PCB
                                                      D:/GITHUB/CS 450 RunTime Terror/mpx core/modules/sys proc loade
    userFunctions.c, 42
    userFunctions.h, 59
                                                      date time, 6
DectoBCD
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/asm.h
userFunctions.c, 42
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/interrupts.h, 60 Delete_PCB
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/io.h.____userFunctions.c, 43
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/serial b,
                                                          userFunctions.c, 43
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/core/tables-hunctions.h, 61
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/mpm/epap.h,
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/medit/potessiongiphor_struct, 7
                                                      gdt entry struct, 7
D:/GITHUB/CS 450 RunTime Terror/mpx core/include/st@edDate
                                                          userFunctions.c, 44
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/include/systemuserFunctions.h, 62
                                                      GetTime
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/intersepfs.roctions.c, 44
                                                          userFunctions.h, 62
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/kmain.c,
                                                      header, 8
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/colesserial.c,
D:/GITHUB/CS_450_RunTime_Terror/mpx_core/kernel/core/system.Eunctions.c, 45
                                                          userFunctions.h, 63
         27
```

78 INDEX

| idt_entry_struct, 9  | string.h, 21   |
|--|--|
| idt_struct, 9  | strcpy   |
| inb  | string.c, 31   |
| io.h, 15   | string.h, 22   |
| index_entry, 9   | string.c   |
| index_table, 10  | atoi, 29   |
| io.h   | isspace, 30  |
| inb, 15  | memset, 30   |
| isspace  | strcat, 30   |
| string.c, 30   | strcmp, 31   |
| string.h, 19   | strcpy, 31   |
| itoa   | strlen, 33   |
| userFunctions.c, 47  | strtok, 33   |
| userFunctions.h, 65  | string.h   |
| addit ditations.ii, od   | atoi, 18   |
| List, 10   | isspace, 19  |
| ,  |  |
| MemList, 10  | memset, 19   |
| memset   | strcat, 21   |
| string.c, 30   | strcmp, 21   |
| string.h, 19   | strcpy, 22   |
| 3 , 1  | strlen, 22   |
| page_dir, 11   | strtok, 22   |
| page_entry, 11   | strlen   |
| page_table, 12   | string.c, 33   |
| param, 12  | string.h, 22   |
| PCB, 12  | strtok   |
| 1 65, 12   | string.c, 33   |
| Queue, 13  | string.h, 22   |
|  | Suspend  |
| Resume   | userFunctions.c, 55  |
| userFunctions.c, 48  | userFunctions.h, 73  |
| ugar Functions h. CF   |  |
| user-unctions.n, 65  |  |
| userFunctions.h, 65  | toLowercase  |
| Set_Priority   |  |
|  | toLowercase<br>userFunctions.c, 56<br>userFunctions.h, 74  |
| Set_Priority   | userFunctions.c, 56  |
| Set_Priority userFunctions.c, 48   | userFunctions.c, 56  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66   | userFunctions.c, 56<br>userFunctions.h, 74   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49   | userFunctions.c, 56<br>userFunctions.h, 74<br>Unblock  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate   | userFunctions.c, 56<br>userFunctions.h, 74<br>Unblock<br>userFunctions.c, 56   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime   | userFunctions.c, 56<br>userFunctions.h, 74<br>Unblock<br>userFunctions.c, 56<br>userFunctions.h, 74<br>userFunctions.c   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50   | userFunctions.c, 56<br>userFunctions.h, 74<br>Unblock<br>userFunctions.c, 56<br>userFunctions.h, 74<br>userFunctions.c<br>AlarmList, 57  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74 userFunctions.c AlarmList, 57 BCDtoDec, 41  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All  | userFunctions.c, 56<br>userFunctions.h, 74  Unblock userFunctions.c, 56<br>userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51  | userFunctions.c, 56<br>userFunctions.h, 74  Unblock userFunctions.c, 56<br>userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68  | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked userFunctions.c, 51   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked userFunctions.c, 51 userFunctions.c, 51 userFunctions.h, 69   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked userFunctions.c, 51 userFunctions.c, 51 userFunctions.h, 69 Show_PCB  | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked userFunctions.c, 51 userFunctions.c, 51 userFunctions.c, 51 userFunctions.c, 51 userFunctions.c, 51 userFunctions.c, 53   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66  SetDate userFunctions.c, 49 userFunctions.h, 67  SetTime userFunctions.c, 50 userFunctions.h, 68  Show_All userFunctions.c, 51 userFunctions.h, 68  Show_Blocked userFunctions.c, 51 userFunctions.c, 51 userFunctions.c, 51 userFunctions.c, 51 userFunctions.c, 53 userFunctions.h, 69   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked userFunctions.c, 51 userFunctions.c, 51 userFunctions.h, 69 Show_PCB userFunctions.c, 53 userFunctions.h, 70 Show_Ready   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47 Resume, 48  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked userFunctions.c, 51 userFunctions.c, 51 userFunctions.h, 69 Show_PCB userFunctions.c, 53 userFunctions.h, 70 Show_Ready userFunctions.c, 54   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47 Resume, 48 Set_Priority, 48   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66  SetDate userFunctions.c, 49 userFunctions.h, 67  SetTime userFunctions.c, 50 userFunctions.h, 68  Show_All userFunctions.c, 51 userFunctions.h, 68  Show_Blocked userFunctions.c, 51 userFunctions.h, 69  Show_PCB userFunctions.c, 53 userFunctions.h, 70  Show_Ready userFunctions.c, 54 userFunctions.h, 71                                   | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47 Resume, 48 Set_Priority, 48 SetDate, 49   |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked userFunctions.c, 51 userFunctions.h, 69 Show_PCB userFunctions.c, 53 userFunctions.h, 70 Show_Ready userFunctions.c, 54 userFunctions.h, 71 strcat                                  | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47 Resume, 48 Set_Priority, 48 SetDate, 49 SetTime, 50                               |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66  SetDate userFunctions.c, 49 userFunctions.h, 67  SetTime userFunctions.c, 50 userFunctions.h, 68  Show_All userFunctions.c, 51 userFunctions.h, 68  Show_Blocked userFunctions.c, 51 userFunctions.h, 69  Show_PCB userFunctions.c, 53 userFunctions.h, 70  Show_Ready userFunctions.c, 54 userFunctions.h, 71  strcat string.c, 30              | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47 Resume, 48 Set_Priority, 48 SetDate, 49 SetTime, 50 Show_All, 51                  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66 SetDate userFunctions.c, 49 userFunctions.h, 67 SetTime userFunctions.c, 50 userFunctions.h, 68 Show_All userFunctions.c, 51 userFunctions.h, 68 Show_Blocked userFunctions.c, 51 userFunctions.h, 69 Show_PCB userFunctions.c, 53 userFunctions.h, 70 Show_Ready userFunctions.c, 54 userFunctions.h, 71 strcat                                  | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47 Resume, 48 Set_Priority, 48 SetDate, 49 SetTime, 50 Show_All, 51 Show_Blocked, 51 |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66  SetDate userFunctions.c, 49 userFunctions.h, 67  SetTime userFunctions.c, 50 userFunctions.h, 68  Show_All userFunctions.c, 51 userFunctions.h, 68  Show_Blocked userFunctions.c, 51 userFunctions.h, 69  Show_PCB userFunctions.c, 53 userFunctions.h, 70  Show_Ready userFunctions.c, 54 userFunctions.h, 71  strcat string.c, 30              | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47 Resume, 48 Set_Priority, 48 SetDate, 49 SetTime, 50 Show_All, 51                  |
| Set_Priority userFunctions.c, 48 userFunctions.h, 66  SetDate userFunctions.c, 49 userFunctions.h, 67  SetTime userFunctions.c, 50 userFunctions.h, 68  Show_All userFunctions.c, 51 userFunctions.h, 68  Show_Blocked userFunctions.c, 51 userFunctions.h, 69  Show_PCB userFunctions.c, 53 userFunctions.h, 70  Show_Ready userFunctions.c, 54 userFunctions.h, 71  strcat string.c, 30 string.h, 21 | userFunctions.c, 56 userFunctions.h, 74  Unblock userFunctions.c, 56 userFunctions.h, 74  userFunctions.c AlarmList, 57 BCDtoDec, 41 Block, 41 Create_PCB, 42 DectoBCD, 42 Delete_PCB, 43 EdgeCase, 43 GetDate, 44 GetTime, 44 Help, 45 itoa, 47 Resume, 48 Set_Priority, 48 SetDate, 49 SetTime, 50 Show_All, 51 Show_Blocked, 51 |

INDEX 79

```
Suspend, 55
    toLowercase, 56
    Unblock, 56
    Version, 57
userFunctions.h
    BCDtoDec, 59
    Block, 59
    Create_PCB, 59
    DectoBCD, 60
    Delete_PCB, 60
    EdgeCase, 61
    GetDate, 62
    GetTime, 62
    Help, 63
    itoa, 65
    Resume, 65
    Set_Priority, 66
    SetDate, 67
    SetTime, 68
    Show_All, 68
    Show_Blocked, 69
    Show_PCB, 70
    Show_Ready, 71
    Suspend, 73
    toLowercase, 74
    Unblock, 74
    Version, 75
Version
    userFunctions.c, 57
    userFunctions.h, 75
```