

C8E

0.1

Generated by Doxygen 1.7.2

Mon Dec 13 2010 20:41:44

Contents

1	File Index	1
1.1	File List	1
2	File Documentation	3
2.1	src/C8E.c File Reference	3
2.1.1	Function Documentation	3
2.1.1.1	main	3
2.2	src/CPU.h File Reference	3
2.2.1	Detailed Description	3
2.3	src/Logs.c File Reference	4
2.3.1	Function Documentation	4
2.3.1.1	addEntry	4
2.3.1.2	closeLogs	4
2.3.1.3	setupLogs	4
2.4	src/Logs.h File Reference	5
2.4.1	Define Documentation	5
2.4.1.1	DEFAULT_DEBUG_LEVEL	5
2.4.1.2	DEFAULT_OUTPUT_FILENAME	5
2.4.2	Enumeration Type Documentation	6
2.4.2.1	DEBUG_LEVELS	6
2.4.3	Function Documentation	6
2.4.3.1	addEntry	6
2.4.3.2	closeLogs	6
2.4.3.3	setupLogs	6
2.5	src/Memory.c File Reference	6
2.5.1	Detailed Description	7
2.5.2	Function Documentation	7
2.5.2.1	cleanupMemory	7
2.5.2.2	read	8
2.5.2.3	setupMemory	8
2.5.2.4	write	8
2.6	src/Memory.h File Reference	8
2.6.1	Define Documentation	9
2.6.1.1	DATA_SPACE_START	9
2.6.1.2	DATA_SPACE_STOP	9
2.6.1.3	MAX_REGISTERS	10
2.6.1.4	RESERVED_MEMORY_START	10
2.6.1.5	RESERVED_MEMORY_STOP	10
2.6.2	Function Documentation	10

2.6.2.1	cleanupMemory	10
2.6.2.2	read	10
2.6.2.3	setupMemory	10
2.6.2.4	write	11

Chapter 1

File Index

1.1 File List

Here is a list of all files with brief descriptions:

src/ C8E.c	3
src/ CPU.h	3
src/ Logs.c	4
src/ Logs.h	5
src/ Memory.c (Define all functions, variables and defines for memory management)	6
src/ Memory.h	8

Chapter 2

File Documentation

2.1 `src/C8E.c` File Reference

```
#include "Logs.h"
```

Functions

- int `main` ()

2.1.1 Function Documentation

2.1.1.1 int `main` ()

Definition at line 3 of file `C8E.c`.

2.2 `src/CPU.h` File Reference

2.2.1 Detailed Description

Version

0.1

Date

December 12, 2010

Author

Maxime Gaudin

Definition in file `CPU.h`.

2.3 src/Logs.c File Reference

```
#include "Logs.h"
```

Functions

- int [setupLogs](#) (unsigned char debugLevel, char *const outputFilename)
Setup output log file and debug level to values passed in paramaters. Moreover, a file descriptor is created and initialized.
- int [closeLogs](#) ()
Close output log file descriptor and flush file buffer.
- void [addEntry](#) (unsigned char level, const char *const message)
Add new entry in output log file if [level] is below or equal to debug level.

2.3.1 Function Documentation

2.3.1.1 void addEntry (unsigned char *level*, const char *const *message*)

Add new entry in output log file if [level] is below or equal to debug level.

Definition at line 49 of file Logs.c.

2.3.1.2 int closeLogs ()

Close output log file descriptor and flush file buffer.

Returns

0 if success, 0 otherwise.

Definition at line 44 of file Logs.c.

2.3.1.3 int setupLogs (unsigned char *debugLevel*, char *const *outputFilename*)

Setup output log file and debug level to values passed in paramaters. Moreover, a file descriptor is created and initialized.

Returns

0 if success, 0 otherwise.

Definition at line 33 of file Logs.c.

2.4 src/Logs.h File Reference

```
#include <stdio.h>
```

Defines

- #define `DEFAULT_DEBUG_LEVEL` 1
Specifies the default debug level : Warning.
- #define `DEFAULT_OUTPUT_FILENAME` "DEBUG_LOGS"
Specifies the default output filename, i.e. the file where log will be written.

Enumerations

- enum `DEBUG_LEVELS` { `ERROR` = 0, `WARNING` = 1, `DRAWING` = 2, `DISASSEMBLING` = 3 }

Functions

- int `setupLogs` (unsigned char debugLevel, char *const outputFilename)
Setup output log file and debug level to values passed in parameters. Moreover, a file descriptor is created and initialized.
- int `closeLogs` ()
Close output log file descriptor and flush file buffer.
- void `addEntry` (unsigned char level, const char *const message)
Add new entry in output log file if [level] is below or equal to debug level.

2.4.1 Define Documentation

2.4.1.1 #define `DEFAULT_DEBUG_LEVEL` 1

Specifies the default debug level : Warning.

Definition at line 23 of file Logs.h.

2.4.1.2 #define `DEFAULT_OUTPUT_FILENAME` "DEBUG_LOGS"

Specifies the default output filename, i.e. the file where log will be written.

Definition at line 26 of file Logs.h.

2.4.2 Enumeration Type Documentation

2.4.2.1 enum DEBUG_LEVELS

Enumerator:

ERROR

WARNING

DRAWING

DISASSEMBLING

Definition at line 20 of file Logs.h.

2.4.3 Function Documentation

2.4.3.1 void addEntry (unsigned char *level*, const char *const *message*)

Add new entry in output log file if [level] is below or equal to debug level.

Definition at line 49 of file Logs.c.

2.4.3.2 int closeLogs ()

Close output log file descriptor and flush file buffer.

Returns

0 if success, 0 otherwise.

Definition at line 44 of file Logs.c.

2.4.3.3 int setupLogs (unsigned char *debugLevel*, char *const *outputFilename*)

Setup output log file and debug level to values passed in paramaters. Moreover, a file descriptor is created and initialized.

Returns

0 if success, 0 otherwise.

Definition at line 33 of file Logs.c.

2.5 src/Memory.c File Reference

Define all functions, variables and defines for memory management.

```
#include "Memory.h"
#include <stdlib.h>
#include <string.h>
```

Functions

- int [setupMemory](#) ()
Initialize memory to 0.
- void [cleanupMemory](#) ()
Cleanup all memory.
- int [write](#) (unsigned short addr, char *const data, unsigned int len)
write [len] bytes from [data] into memory at adress [addr]
- int [read](#) (short addr, unsigned short len, char *const buffer)
Read [len] bytes of data from address [addr] to buffer.

2.5.1 Detailed Description

Define all functions, variables and defines for memory management.

Version

0.1

Date

December 12, 2010

Author

Maxime Gaudin

Definition in file [Memory.c](#).

2.5.2 Function Documentation

2.5.2.1 void cleanupMemory ()

Cleanup all memory.

Definition at line 32 of file Memory.c.

2.5.2.2 int read (short *addr*, unsigned short *len*, char *const *buffer*)

Read [*len*] bytes of data from address [*addr*] to buffer.

Parameters

in	<i>addr</i>	Address where read begins
in	<i>len</i>	Number of bytes read
out	<i>buffer</i>	Pointer to the data buffer

Returns

0 if success, 1 otherwise.

Definition at line 45 of file Memory.c.

2.5.2.3 int setupMemory ()

Initialize memory to 0.

Returns

0 if success, 1 otherwise.

Definition at line 24 of file Memory.c.

2.5.2.4 int write (unsigned short *addr*, char *const *data*, unsigned int *len*)

write [*len*] bytes from [*data*] into memory at adress [*addr*]

Parameters

in	<i>addr</i>	Address where data will be written
in	<i>data</i>	Pointer to data buffer
in	<i>len</i>	Number of byte written

Returns

0 if success, 1 otherwise.

Definition at line 36 of file Memory.c.

2.6 src/Memory.h File Reference

Defines

- #define [RESERVED_MEMORY_START](#) 0x0
Specifies where memory starts (0x0, what a surprise isn't it ??).

- #define `RESERVED_MEMORY_STOP` 0x200
Specifies where the memory stops.
- #define `DATA_SPACE_START` 0x200
Specifies the beginning of the data space.
- #define `DATA_SPACE_STOP` 0xFFFF
Specifies the end of the data space.
- #define `MAX_REGISTERS` 0xF
Specifies the maximum number of registers..

Functions

- int `setupMemory` ()
Initialize memory to 0.
- void `cleanupMemory` ()
Cleanup all memory.
- int `write` (unsigned short addr, char *const data, unsigned int len)
write [len] bytes from [data] into memory at adress [addr]
- int `read` (short addr, unsigned short len, char *const buffer)
Read [len] bytes of data from address [addr] to buffer.

2.6.1 Define Documentation

2.6.1.1 #define `DATA_SPACE_START` 0x200

Specifies the beginning of the data space.

Definition at line 36 of file Memory.h.

2.6.1.2 #define `DATA_SPACE_STOP` 0xFFFF

Specifies the end of the data space.

Definition at line 38 of file Memory.h.

2.6.1.3 `#define MAX_REGISTERS 0xF`

Specifies the maximum number of registers..

Definition at line 41 of file Memory.h.

2.6.1.4 `#define RESERVED_MEMORY_START 0x0`

Specifies where memory starts (0x0, what a surprise isn't it ??).

Definition at line 31 of file Memory.h.

2.6.1.5 `#define RESERVED_MEMORY_STOP 0x200`

Specifies where the memory stops.

Definition at line 33 of file Memory.h.

2.6.2 Function Documentation

2.6.2.1 `void cleanupMemory ()`

Cleanup all memory.

Definition at line 32 of file Memory.c.

2.6.2.2 `int read (short addr, unsigned short len, char *const buffer)`

Read [*len*] bytes of data from address [*addr*] to buffer.

Parameters

<i>in</i>	<i>addr</i>	Address where read begins
<i>in</i>	<i>len</i>	Number of bytes read
<i>out</i>	<i>buffer</i>	Pointer to the data buffer

Returns

0 if success, 1 otherwise.

Definition at line 45 of file Memory.c.

2.6.2.3 `int setupMemory ()`

Initialize memory to 0.

Returns

0 if success, 1 otherwise.

Definition at line 24 of file Memory.c.

2.6.2.4 `int write (unsigned short addr, char *const data, unsigned int len)`

write [len] bytes from [data] into memory at adress [addr]

Parameters

in	<i>addr</i>	Address where data will be written
in	<i>data</i>	Pointer to data buffer
in	<i>len</i>	Number of byte written

Returns

0 if success, 1 otherwise.

Definition at line 36 of file Memory.c.

Index

addEntry
 Logs.c, [4](#)
 Logs.h, [6](#)

C8E.c
 main, [3](#)

cleanupMemory
 Memory.c, [7](#)
 Memory.h, [10](#)

closeLogs
 Logs.c, [4](#)
 Logs.h, [6](#)

DATA_SPACE_START
 Memory.h, [9](#)

DATA_SPACE_STOP
 Memory.h, [9](#)

DEBUG_LEVELS
 Logs.h, [6](#)

DEFAULT_DEBUG_LEVEL
 Logs.h, [5](#)

DEFAULT_OUTPUT_FILENAME
 Logs.h, [5](#)

DISASSEMBLING
 Logs.h, [6](#)

DRAWING
 Logs.h, [6](#)

ERROR
 Logs.h, [6](#)

Logs.c
 addEntry, [4](#)
 closeLogs, [4](#)
 setupLogs, [4](#)

Logs.h
 addEntry, [6](#)
 closeLogs, [6](#)
 DEBUG_LEVELS, [6](#)
 DEFAULT_DEBUG_LEVEL, [5](#)
 DEFAULT_OUTPUT_FILENAME, [5](#)
 DISASSEMBLING, [6](#)
 DRAWING, [6](#)
 ERROR, [6](#)
 setupLogs, [6](#)
 WARNING, [6](#)

main
 C8E.c, [3](#)

MAX_REGISTERS
 Memory.h, [9](#)

Memory.c
 cleanupMemory, [7](#)
 read, [7](#)
 setupMemory, [8](#)
 write, [8](#)

Memory.h
 cleanupMemory, [10](#)
 DATA_SPACE_START, [9](#)
 DATA_SPACE_STOP, [9](#)
 MAX_REGISTERS, [9](#)
 read, [10](#)
 RESERVED_MEMORY_START, [10](#)
 RESERVED_MEMORY_STOP, [10](#)
 setupMemory, [10](#)
 write, [11](#)

read
 Memory.c, [7](#)
 Memory.h, [10](#)

RESERVED_MEMORY_START
 Memory.h, [10](#)

RESERVED_MEMORY_STOP
 Memory.h, [10](#)

setupLogs
 Logs.c, [4](#)
 Logs.h, [6](#)

setupMemory
 Memory.c, [8](#)
 Memory.h, [10](#)

src/C8E.c, [3](#)

src/CPU.h, [3](#)

src/Logs.c, [4](#)
src/Logs.h, [5](#)
src/Memory.c, [6](#)
src/Memory.h, [8](#)

WARNING

Logs.h, [6](#)

write

Memory.c, [8](#)
Memory.h, [11](#)