

octochip8

Generated by Doxygen 1.8.4

Fri Jan 17 2014 02:44:31

Contents

1	Class Index	1
1.1	Class List	1
2	File Index	3
2.1	File List	3
3	Class Documentation	5
3.1	CPU Class Reference	5
3.1.1	Constructor & Destructor Documentation	6
3.1.1.1	CPU	6
3.1.1.2	~CPU	6
3.1.2	Member Function Documentation	6
3.1.2.1	emulateCycle	6
3.1.2.2	executeOpcode	6
3.1.2.3	getDrawFlag	6
3.1.2.4	getGFX	6
3.1.2.5	initalize	6
3.1.2.6	loadGame	6
3.1.2.7	setDrawFlag	7
3.1.2.8	setKeys	7
3.1.2.9	setOpcode	7
3.1.3	Member Data Documentation	7
3.1.3.1	drawFlag	7
3.1.3.2	gfx	7
3.1.3.3	l	7
3.1.3.4	key	7
3.1.3.5	memory	7
3.1.3.6	opcode	7
3.1.3.7	pc	7
3.1.3.8	running	8
3.1.3.9	SCREEN_HEIGHT	8
3.1.3.10	SCREEN_SIZE	8

3.1.3.11	SCREEN_WIDTH	8
3.1.3.12	sp	8
3.1.3.13	stack	8
3.1.3.14	V	8
3.2	Graphics Class Reference	8
3.2.1	Constructor & Destructor Documentation	8
3.2.1.1	Graphics	8
3.2.1.2	~Graphics	8
3.2.2	Member Function Documentation	8
3.2.2.1	draw	8
3.2.2.2	initialize	8
3.3	Input Class Reference	9
3.3.1	Constructor & Destructor Documentation	9
3.3.1.1	Input	9
3.3.1.2	Input	9
3.3.1.3	~Input	9
3.3.2	Member Function Documentation	9
3.3.2.1	initialize	9
4	File Documentation	11
4.1	src/CPU.cpp File Reference	11
4.2	src/CPU.h File Reference	11
4.3	src/Graphics.cpp File Reference	11
4.4	src/Graphics.h File Reference	11
4.5	src/Input.cpp File Reference	12
4.6	src/Input.h File Reference	12
4.7	src/octochip8.cpp File Reference	12
4.7.1	Function Documentation	12
4.7.1.1	main	12
4.7.2	Variable Documentation	12
4.7.2.1	cpu	12
4.7.2.2	gpu	12
4.7.2.3	input	12

Chapter 1

Class Index

1.1 Class List

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

CPU	5
Graphics	8
Input	9

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

src/ CPU.cpp	11
src/ CPU.h	11
src/ Graphics.cpp	11
src/ Graphics.h	11
src/ Input.cpp	12
src/ Input.h	12
src/ octochip8.cpp	12

Chapter 3

Class Documentation

3.1 CPU Class Reference

```
#include <CPU.h>
```

Public Member Functions

- [CPU](#) ()
- virtual [~CPU](#) ()
- void [inititalize](#) ()
- void [loadGame](#) (std::string filename)
- void [emulateCycle](#) ()
- bool [getDrawFlag](#) ()
- void [setDrawFlag](#) (bool flag)
- void [setKeys](#) ()
- vector< unsigned char > [getGFX](#) ()

Public Attributes

- bool [running](#)

Static Public Attributes

- static const int [SCREEN_SIZE](#) = 64 * 32
- static const int [SCREEN_WIDTH](#) = 64
- static const int [SCREEN_HEIGHT](#) = 32

Private Member Functions

- void [setOpcode](#) ()
- void [executeOpcode](#) ()

Private Attributes

- unsigned short [opcode](#)
- vector< unsigned char > [memory](#)
- vector< unsigned char > [V](#)

- unsigned short `l`
- unsigned short `pc`
- vector< bool > `gfx`
- vector< unsigned short > `stack`
- unsigned short `sp`
- vector< unsigned char > `key`
- bool `drawFlag`

3.1.1 Constructor & Destructor Documentation

3.1.1.1 CPU::CPU ()

The constructor for the class, initialise must be called after this to be used.

3.1.1.2 CPU::~~CPU () [virtual]

3.1.2 Member Function Documentation

3.1.2.1 void CPU::emulateCycle ()

Emulate a cycle of the `CPU`

3.1.2.2 void CPU::executeOpcode () [private]

Executes the opcode of the current program.

3.1.2.3 bool CPU::getDrawFlag ()

Gets the current draw flag determining wether or not to draw during this cpu cycle.

Returns

A bool of the current draw flag. True = Draw screen. False = Don't draw screen.

3.1.2.4 vector< unsigned char > CPU::getGFX ()

Gets the vector object of pixels for the current screen.

3.1.2.5 void CPU::inititalize ()

Called after construction, sets up all registers and memory.

3.1.2.6 void CPU::loadGame (std::string *filename*)

Load a game into the emulator.

Parameters

<i>filename</i>	The file to load. Type will probably change later.
-----------------	--

3.1.2.7 void CPU::setDrawFlag (bool *flag*)

Sets the draw flag. Should be called at end of every cpu loop.

Parameters

<i>flag</i>	The boolean value to set the draw flag.
-------------	---

3.1.2.8 void CPU::setKeys ()

Sets the keys for the current screen.

3.1.2.9 void CPU::setOpcode () [private]

Sets the opcode of the current program to the instruction at PC.

3.1.3 Member Data Documentation

3.1.3.1 bool CPU::drawFlag [private]

< The current state of the key

3.1.3.2 vector<bool> CPU::gfx [private]

A vector representing the current screen

3.1.3.3 unsigned short CPU::i [private]

< The CPU registers. CPU registers: The Chip 8 has 15 8-bit general purpose registers named V0,V1 up to VE. The 16th register is used for the 'carry flag'. The index register, counts down from value to 0 when in use.

3.1.3.4 vector<unsigned char> CPU::key [private]

The pointer to the current level in the stack

3.1.3.5 vector<unsigned char> CPU::memory [private]

The virtual memory - 8k memory

3.1.3.6 unsigned short CPU::opcode [private]

The current operation code.

3.1.3.7 unsigned short CPU::pc [private]

The program counter, counts down from value to 0 when in use.

3.1.3.8 bool CPU::running

The height of the screen in pixels. A boolean for the running state of the [CPU](#)

3.1.3.9 const int CPU::SCREEN_HEIGHT = 32 [static]

The width of the screen in pixels.

3.1.3.10 const int CPU::SCREEN_SIZE = 64 * 32 [static]

The amount of pixels for the screen

3.1.3.11 const int CPU::SCREEN_WIDTH = 64 [static]

3.1.3.12 unsigned short CPU::sp [private]

3.1.3.13 vector<unsigned short> CPU::stack [private]

The stack. Has 16 levels. ha pancakes

3.1.3.14 vector<unsigned char> CPU::V [private]

The documentation for this class was generated from the following files:

- [src/CPU.h](#)
- [src/CPU.cpp](#)

3.2 Graphics Class Reference

```
#include <Graphics.h>
```

Public Member Functions

- void [inititalize](#) ()
- void [draw](#) (std::vector< unsigned char > screen)
- [Graphics](#) ()
- virtual [~Graphics](#) ()

3.2.1 Constructor & Destructor Documentation

3.2.1.1 Graphics::Graphics ()

3.2.1.2 Graphics::~~Graphics () [virtual]

3.2.2 Member Function Documentation

3.2.2.1 void Graphics::draw (std::vector< unsigned char > screen)

3.2.2.2 void Graphics::inititalize ()

The documentation for this class was generated from the following files:

- [src/Graphics.h](#)
- [src/Graphics.cpp](#)

3.3 Input Class Reference

```
#include <Input.h>
```

Public Member Functions

- void [initialize](#) ()
- [Input](#) ()
- [Input](#) (const [Input](#) &orig)
- virtual [~Input](#) ()

3.3.1 Constructor & Destructor Documentation

3.3.1.1 [Input::Input](#) ()

3.3.1.2 [Input::Input](#) (const [Input](#) & *orig*)

3.3.1.3 [Input::~Input](#) () [[virtual](#)]

3.3.2 Member Function Documentation

3.3.2.1 void [Input::initialize](#) ()

The documentation for this class was generated from the following files:

- [src/Input.h](#)
- [src/Input.cpp](#)

Chapter 4

File Documentation

4.1 src/CPU.cpp File Reference

```
#include <algorithm>
#include "CPU.h"
```

4.2 src/CPU.h File Reference

```
#include <string>
#include <vector>
```

Classes

- class [CPU](#)

4.3 src/Graphics.cpp File Reference

```
#include "Graphics.h"
```

4.4 src/Graphics.h File Reference

```
#include "CPU.h"
#include <vector>
```

Classes

- class [Graphics](#)

4.5 src/Input.cpp File Reference

```
#include "Input.h"
```

4.6 src/Input.h File Reference

Classes

- class [Input](#)

4.7 src/octochip8.cpp File Reference

```
#include "CPU.h"  
#include "Graphics.h"  
#include "Input.h"
```

Functions

- int [main](#) (void)

Variables

- [CPU](#) `cpu`
- [Graphics](#) `gpu`
- [Input](#) `input`

4.7.1 Function Documentation

4.7.1.1 int main (void)

The input object to be used The main function to start it all off.

Returns

Exit code. 0 is normal.

4.7.2 Variable Documentation

4.7.2.1 CPU `cpu`

The [CPU](#) object to be used

4.7.2.2 Graphics `gpu`

4.7.2.3 Input `input`

The GPU object to be used

Index

- ~CPU
 - CPU, [6](#)
- ~Graphics
 - Graphics, [8](#)
- ~Input
 - Input, [9](#)
- CPU, [5](#)
 - ~CPU, [6](#)
 - CPU, [6](#)
 - CPU, [6](#)
 - drawFlag, [7](#)
 - emulateCycle, [6](#)
 - executeOpcode, [6](#)
 - getDrawFlag, [6](#)
 - getGFX, [6](#)
 - gfx, [7](#)
 - I, [7](#)
 - initalize, [6](#)
 - key, [7](#)
 - loadGame, [6](#)
 - memory, [7](#)
 - opcode, [7](#)
 - pc, [7](#)
 - running, [7](#)
 - SCREEN_HEIGHT, [8](#)
 - SCREEN_SIZE, [8](#)
 - SCREEN_WIDTH, [8](#)
 - setDrawFlag, [7](#)
 - setKeys, [7](#)
 - setOpcode, [7](#)
 - sp, [8](#)
 - stack, [8](#)
 - V, [8](#)
- cpu
 - octochip8.cpp, [12](#)
- draw
 - Graphics, [8](#)
- drawFlag
 - CPU, [7](#)
- emulateCycle
 - CPU, [6](#)
- executeOpcode
 - CPU, [6](#)
- getDrawFlag
 - CPU, [6](#)
- getGFX
 - CPU, [6](#)
- gfx
 - CPU, [6](#)
- gpu
 - CPU, [7](#)
- gpu
 - octochip8.cpp, [12](#)
- Graphics, [8](#)
 - ~Graphics, [8](#)
 - draw, [8](#)
 - Graphics, [8](#)
 - initalize, [8](#)
- I
 - CPU, [7](#)
- initalize
 - CPU, [6](#)
 - Graphics, [8](#)
 - Input, [9](#)
- Input, [9](#)
 - ~Input, [9](#)
 - initalize, [9](#)
 - Input, [9](#)
- input
 - octochip8.cpp, [12](#)
- key
 - CPU, [7](#)
- loadGame
 - CPU, [6](#)
- main
 - octochip8.cpp, [12](#)
- memory
 - CPU, [7](#)
- octochip8.cpp
 - cpu, [12](#)
 - gpu, [12](#)
 - input, [12](#)
 - main, [12](#)
- opcode
 - CPU, [7](#)
- pc
 - CPU, [7](#)
- running
 - CPU, [7](#)
- SCREEN_HEIGHT
 - CPU, [8](#)

- SCREEN_SIZE
 - CPU, [8](#)
- SCREEN_WIDTH
 - CPU, [8](#)
- setDrawFlag
 - CPU, [7](#)
- setKeys
 - CPU, [7](#)
- setOpcode
 - CPU, [7](#)
- sp
 - CPU, [8](#)
- src/CPU.cpp, [11](#)
- src/CPU.h, [11](#)
- src/Graphics.cpp, [11](#)
- src/Graphics.h, [11](#)
- src/Input.cpp, [12](#)
- src/Input.h, [12](#)
- src/octochip8.cpp, [12](#)
- stack
 - CPU, [8](#)
- V
 - CPU, [8](#)