

Ape

DONTRELEASEME-129

Generated by Doxygen 1.8.14

Contents

1	Main Page	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Class Documentation	9
5.1	Core::CPU::CPU Class Reference	9
5.1.1	Detailed Description	11
5.2	Core::HW::FloppyDrive Class Reference	11
5.2.1	Detailed Description	11
5.3	Core::CPU::Instruction Class Reference	12
5.3.1	Detailed Description	12
5.3.2	Member Enumeration Documentation	13
5.3.2.1	SegmentPrefix	13
5.3.3	Constructor & Destructor Documentation	13
5.3.3.1	Instruction()	13
5.3.4	Member Function Documentation	13
5.3.4.1	GetLength()	13
5.3.4.2	Resolve()	14

5.4	Core::CPU::InvalidInstructionException Class Reference	14
5.4.1	Detailed Description	14
5.5	Core::Machine Class Reference	14
5.5.1	Detailed Description	15
5.6	MainWindow Class Reference	15
5.7	Core::Memory Class Reference	15
5.7.1	Detailed Description	16
5.8	Core::CPU::Instruction::Parameter Class Reference	16
5.8.1	Detailed Description	17
5.9	Core::CPU::ParameterLengthMismatchException Class Reference	17
5.9.1	Detailed Description	17
5.10	TTYBackend Class Reference	17
5.11	TTYWidget Class Reference	18
5.12	Core::CPU::UnhandledInstructionException Class Reference	19
5.12.1	Detailed Description	19
5.13	Core::CPU::UnhandledInterruptException Class Reference	19
5.13.1	Detailed Description	19
5.14	Core::CPU::UnhandledParameterException Class Reference	20
5.14.1	Detailed Description	20
5.15	Core::CPU::UnsupportedParameterException Class Reference	20
5.15.1	Detailed Description	20
6	File Documentation	21
6.1	/home/max/sources/ape/Docs/Terms.dox File Reference	21
6.1.1	Detailed Description	21
6.2	/home/max/sources/ape/Source/Common/Logger.h File Reference	23
6.3	/home/max/sources/ape/Source/Common/Types.h File Reference	24
6.4	/home/max/sources/ape/Source/Core/CPU/CPU.h File Reference	24
6.5	/home/max/sources/ape/Source/Core/CPU/Exception.h File Reference	25
6.6	/home/max/sources/ape/Source/Core/CPU/Instruction.h File Reference	25
6.7	/home/max/sources/ape/Source/Core/HW/FloppyDrive.h File Reference	26
6.8	/home/max/sources/ape/Source/Core/Machine.h File Reference	26
6.9	/home/max/sources/ape/Source/Core/Memory.h File Reference	26
	Index	27

Chapter 1

Main Page

Introduction

Ape (Another PC Emulator) is an experimental IBM PC compatible emulator written in C++17.

Important Pages

[Core::CPU::Instruction](#)

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Core::CPU::CPU	9
exception	
Core::CPU::InvalidInstructionException	14
Core::CPU::ParameterLengthMismatchException	17
Core::CPU::UnhandledInstructionException	19
Core::CPU::UnhandledInterruptException	19
Core::CPU::UnhandledParameterException	20
Core::CPU::UnsupportedParameterException	20
Core::HW::FloppyDrive	11
Core::CPU::Instruction	12
Core::Machine	14
Core::Memory	15
Core::CPU::Instruction::Parameter	16
QMainWindow	
MainWindow	15
QTextBrowser	
TTYWidget	18
TTYBackend	17
TTYWidget	18

Chapter 3

Class Index

3.1 Class List

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

Core::CPU::CPU	
Representation of the Central Processing Unit	9
Core::HW::FloppyDrive	
Representation of a floppy drive	11
Core::CPU::Instruction	
High-Level representation of a instruction	12
Core::CPU::InvalidInstructionException	14
Core::Machine	
Representation of a PC	14
MainWindow	15
Core::Memory	
Wrapper around emulated RAM	15
Core::CPU::Instruction::Parameter	
High-Level representation of a instruction parameter	16
Core::CPU::ParameterLengthMismatchException	17
TTYBackend	17
TTYWidget	18
Core::CPU::UnhandledInstructionException	
Thrown when the CPU hits an unimplemented Instruction::Type	19
Core::CPU::UnhandledInterruptException	
Thrown when a interrupt is neither handled by software nor hardware	19
Core::CPU::UnhandledParameterException	
Thrown when the CPU hits an unimplemented Instruction::Parameter::Type	20
Core::CPU::UnsupportedParameterException	
Thrown when a instruction doesn't support the parameter type provided	20

Chapter 4

File Index

4.1 File List

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

/home/max/sources/ape/Source/ApeQt/ MainWindow.h	??
/home/max/sources/ape/Source/ApeQt/ QueueOnObject.h	??
/home/max/sources/ape/Source/ApeQt/ TTYWidget.h	??
/home/max/sources/ape/Source/Common/ File.h	??
/home/max/sources/ape/Source/Common/ Logger.h	23
/home/max/sources/ape/Source/Common/ String.h	??
/home/max/sources/ape/Source/Common/ Types.h	24
/home/max/sources/ape/Source/Core/ Machine.h	26
/home/max/sources/ape/Source/Core/ Memory.h	26
/home/max/sources/ape/Source/Core/ TTY.h	??
/home/max/sources/ape/Source/Core/ TTYBackend.h	??
/home/max/sources/ape/Source/Core/CPU/ CPU.h	24
/home/max/sources/ape/Source/Core/CPU/ Exception.h	25
/home/max/sources/ape/Source/Core/CPU/ Flags.h	??
/home/max/sources/ape/Source/Core/CPU/ Instruction.h	25
/home/max/sources/ape/Source/Core/HW/ FloppyDrive.h	26
/home/max/sources/ape/Source/Core/MSDOS/ File.h	??

Chapter 5

Class Documentation

5.1 Core::CPU::CPU Class Reference

Representation of the Central Processing Unit.

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

Public Member Functions

- **CPU** ([Machine](#) *machine)
- void [Tick](#) ()
Execute one [CPU](#) cycle.
- void [Start](#) ()
Execute instructions until shutdown is requested.

Public Attributes

- [u16](#) & [AX](#) = AX_struct.AX
AX (Accumulator)
- [u8](#) & [AH](#) = AX_struct.b8.AH
AH (High)
- [u8](#) & [AL](#) = AX_struct.b8.AL
AL (Low)
- [u16](#) & [BX](#) = BX_struct.BX
BX.
- [u8](#) & [BH](#) = BX_struct.b8.BH
BH (High)
- [u8](#) & [BL](#) = BX_struct.b8.BL
BL (Low)
- [u16](#) & [CX](#) = CX_struct.CX
CX.
- [u8](#) & [CH](#) = CX_struct.b8.CH
CH (High)
- [u8](#) & [CL](#) = CX_struct.b8.CL

- CL (Low)*
 - `u16 & DX = DX_struct.DX`
 - DX.*
 - `u8 & DH = DX_struct.b8.DH`
 - DH (High)*
 - `u8 & DL = DX_struct.b8.DL`
 - DL (Low)*
 - `u16 CS = 0`
 - Code Segment.*
 - `u16 DS = 0`
 - Data Segment.*
 - `u16 ES = 0`
 - Extra(?) Segment.*
 - `u16 SS = 0`
 - Stack Segment.*
 - `u16 IP = 0`
 - Instruction Pointer.*
 - `u16 BP = 0`
 - Base Pointer.*
 - `u16 SP = 0`
 - Stack Pointer.*
 - `u16 SI = 0`
 - Source Index.*
 - `u16 DI = 0`
 - Destination Index.*
 - `bool AF = false`
 - Adjust Flag.*
 - `bool CF = false`
 - Carry Flag.*
 - `bool IF = true`
 - Interrupt Flag.*
 - `bool DF = false`
 - Direction Flag.*
 - `bool OF = false`
 - Overflow Flag.*
 - `bool PF = false`
 - Parity Flag.*
 - `bool SF = false`
 - Sign Flag.*
 - `bool ZF = false`
 - Zero Flag.*
 - `bool simulate_msdos = false`
 - Simulate MS-DOS (Handle its interrupts)*
 - `std::atomic< bool > running = false`
 - Set whether the CPU is running.*

5.1.1 Detailed Description

Representation of the Central Processing Unit.

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

- [/home/max/sources/ape/Source/Core/CPU/CPU.h](#)
- [/home/max/sources/ape/Source/Core/BIOS/Interrupt.cpp](#)
- [/home/max/sources/ape/Source/Core/CPU/CPU.cpp](#)
- [/home/max/sources/ape/Source/Core/CPU/Flags.cpp](#)
- [/home/max/sources/ape/Source/Core/CPU/Flags.h](#)

5.2 Core::HW::FloppyDrive Class Reference

Representation of a floppy drive.

```
#include <FloppyDrive.h>
```

Public Member Functions

- bool [Insert](#) (const std::string &path)
Insert an image into the drive.
- bool [HasDisc](#) () const
Check if a disc is present.
- [u32 GetSize](#) () const
Get the size of the inserted disc.
- bool [IsBootable](#) ()
Check if the provided image is bootable.
- [u32 GetSectorSize](#) () const
Get size of a floppy.
- [u32 GetSectorsPerTrack](#) () const
Get sectors per track.
- [u32 GetHeadCount](#) () const
Get head count.
- void [Eject](#) ()
Eject the image.
- bool [Read](#) ([u32](#) offset, [u32](#) size, [u8](#) *buffer)
Read data from the image.
- bool [Read](#) ([u8](#) cylinder, [u8](#) head, [u8](#) sector, [u8](#) count, [u8](#) *buffer)

5.2.1 Detailed Description

Representation of a floppy drive.

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

- [/home/max/sources/ape/Source/Core/HW/FloppyDrive.h](#)
- [/home/max/sources/ape/Source/Core/HW/FloppyDrive.cpp](#)

5.3 Core::CPU::Instruction Class Reference

High-Level representation of a instruction.

```
#include <Instruction.h>
```

Classes

- class [Parameter](#)
High-Level representation of a instruction parameter.

Public Types

- enum [SegmentPrefix](#) : u8
Enum of all possible segment prefixes.
- enum [Type](#) : u8 { **PRIVATE** }
Determines the type of the instruction.

Public Member Functions

- [Instruction](#) (u8 opcode, u32 offset=0)
Turns the provided opcode into an [Instruction](#).
- [Instruction](#) (const [Instruction](#) &ins, u8 opcode, u32 offset=0)
One but with prefixes.
- [Type](#) [GetType](#) () const
Get the Type associated with this instruction.
- [SegmentPrefix](#) [GetPrefix](#) () const
Get the SegmentPrefix associated with this instruction.
- bool [IsResolved](#) ()
Checks whether this [Instruction](#) needs further resolving.
- bool [IsPrefix](#) () const
Checks if this instruction is actually a prefix.
- bool [Resolve](#) (u8 mod, std::vector< u8 > data)
Resolves the [Instruction](#).
- u8 [GetLength](#) (u8 mod)
Get the length of the instruction provided.
- std::string [ToString](#) () const
Get a disassembly for the [Instruction](#) provided.
- const std::vector< [Parameter](#) > & [GetParameters](#) () const
Get a vector of parameters.
- [Parameter](#) & [GetParameter](#) (size_t index)
Get a specific vector.
- void [AddParameter](#) ([Parameter](#) parameter)
Add a new parameter.

5.3.1 Detailed Description

High-Level representation of a instruction.

5.3.2 Member Enumeration Documentation

5.3.2.1 SegmentPrefix

```
enum Core::CPU::Instruction::SegmentPrefix : u8 [strong]
```

Enum of all possible segment prefixes.

These prefixes override the default segment of an instruction.

5.3.3 Constructor & Destructor Documentation

5.3.3.1 Instruction()

```
Core::CPU::Instruction::Instruction (
    u8 opcode,
    u32 offset = 0 ) [explicit]
```

Turns the provided opcode into an [Instruction](#).

Parameters

<i>opcode</i>	Opcode to be decoded
---------------	----------------------

5.3.4 Member Function Documentation

5.3.4.1 GetLength()

```
u8 Instruction::GetLength (
    u8 mod )
```

Get the length of the instruction provided.

Returns

Length of the instruction in bytes

5.3.4.2 Resolve()

```
bool Instruction::Resolve (
    u8 mod,
    std::vector< u8 > data )
```

Resolves the [Instruction](#).

Parameters

<i>mod</i>	Modifier byte. The byte after the opcode regardless if it is used as such
<i>data</i>	needs to have as many bytes as specified in GetInstructionLength()

Returns

Returns `false` if there was an error during resolving.

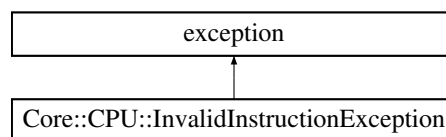
The documentation for this class was generated from the following files:

- [/home/max/sources/ape/Source/Core/CPU/Instruction.h](#)
- [/home/max/sources/ape/Source/Core/CPU/Decoder.cpp](#)
- [/home/max/sources/ape/Source/Core/CPU/Instruction.cpp](#)

5.4 Core::CPU::InvalidInstructionException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for Core::CPU::InvalidInstructionException:



5.4.1 Detailed Description

Thrown when the [CPU](#) hits an instruction that couldn't be resolved / is invalid

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

- [/home/max/sources/ape/Source/Core/CPU/Exception.h](#)

5.5 Core::Machine Class Reference

Representation of a PC.

```
#include <Machine.h>
```

Public Member Functions

- [HW::FloppyDrive](#) & [GetFloppyDrive](#) ()
Get this machines [HW::FloppyDrive](#).
- [Memory](#) & [GetMemory](#) ()
Get this machines [Memory](#).
- bool [BootFloppy](#) ()
Boot the machine from the floppy drive.
- void [Shutdown](#) ()
Shutdown the machine.
- bool [BootCOM](#) (const std::string &file)
Directly execute a COM file.

5.5.1 Detailed Description

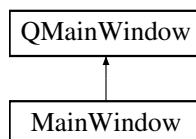
Representation of a PC.

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

- </home/max/sources/ape/Source/Core/Machine.h>
- </home/max/sources/ape/Source/Core/Machine.cpp>

5.6 MainWindow Class Reference

Inheritance diagram for MainWindow:



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

- </home/max/sources/ape/Source/ApeQt/MainWindow.h>
- </home/max/sources/ape/Source/ApeQt/MainWindow.cpp>

5.7 Core::Memory Class Reference

Wrapper around emulated RAM.

```
#include <Memory.h>
```

Public Member Functions

- **Memory** (u32 size)
*Create **Memory** of the specified size (in bytes)*
- std::vector< u8 > & **Get** ()
Get the contents of RAM.
- template<typename T >
T & **Get** (u16 segment, u16 offset)
- template<typename T >
T * **GetPtr** (u16 segment, u16 offset)

Static Public Member Functions

- static u32 **VirtToPhys** (u16 segment, u16 offset)
Converts a virtual address to an absolute one.

5.7.1 Detailed Description

Wrapper around emulated RAM.

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

- /home/max/sources/ape/Source/Core/**Memory.h**
- /home/max/sources/ape/Source/Core/**Memory.cpp**

5.8 Core::CPU::Instruction::Parameter Class Reference

High-Level representation of a instruction parameter.

```
#include <Instruction.h>
```

Public Types

- enum **Type** : u8 { **PRIVATE** }
Enum to determine the types of parameters.

Public Member Functions

- **Parameter** (**Parameter::Type** type)
*Create a parameter with the **Instruction::Parameter::Type** provided.*
- void **Resolve** (u32 data)
Provide the parameter with its missing data.
- void **Resolve** (**Parameter::Type** type, u32 data=0)
Change the type of the parameter and provide missing data (if any)
- **Type** **GetType** () const
Get the parameters type.
- template<typename T >
T **GetData** () const
Get the data associated with this parameter.
- bool **IsResolved** () const
Returns `true` if this parameter is resolved.
- bool **IsWord** () const
Returns `true` if this parameter points to or is a word.
- std::string **ToString** (**SegmentPrefix** prefix=**SegmentPrefix::None**, u32 offset=0) const
Get a human readable form of this parameter.

5.8.1 Detailed Description

High-Level representation of a instruction parameter.

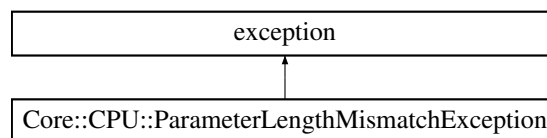
The documentation for this class was generated from the following files:

- [/home/max/sources/ape/Source/Core/CPU/Instruction.h](#)
- [/home/max/sources/ape/Source/Core/CPU/Instruction.cpp](#)

5.9 Core::CPU::ParameterLengthMismatchException Class Reference

```
#include <Exception.h>
```

Inheritance diagram for Core::CPU::ParameterLengthMismatchException:



5.9.1 Detailed Description

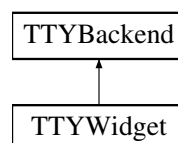
Thrown when a `word` and `byte` parameter are used together when they shouldn't be

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

- [/home/max/sources/ape/Source/Core/CPU/Exception.h](#)

5.10 TTYBackend Class Reference

Inheritance diagram for TTYBackend:



Public Member Functions

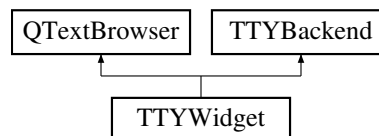
- virtual void **Write** (const std::string &string)=0
- virtual void **Write** (const char c)=0
- virtual void **Scroll** (const [u8](#) lines, const [u8](#) colors)=0
- virtual void **MoveCursor** (const [u32](#), const [u32](#))=0
- virtual [u8](#) **GetCursorRow** () const =0
- virtual void **SetCursorRow** ([u8](#) row)=0
- virtual [u8](#) **GetCursorColumn** () const =0
- virtual void **SetCursorColumn** ([u8](#) column)=0
- virtual void **Clear** ()=0
- virtual char **Read** ()=0

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

- /home/max/sources/ape/Source/Core/TTYBackend.h

5.11 TTYWidget Class Reference

Inheritance diagram for TTYWidget:



Public Member Functions

- void **Write** (const std::string &string) override
- void **Write** (const char c) override
- void **Scroll** (const [u8](#) lines, const [u8](#) colors) override
- void **MoveCursor** (const [u32](#) x, const [u32](#) y) override
- [u8](#) **GetCursorRow** () const override
- void **SetCursorRow** ([u8](#) row) override
- [u8](#) **GetCursorColumn** () const override
- void **SetCursorColumn** ([u8](#) column) override
- void **Clear** () override
- char **Read** () override

Public Attributes

- [u8](#) **m_row** = 0
- [u8](#) **m_column** = 0

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

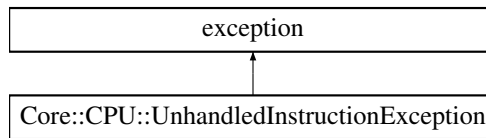
- /home/max/sources/ape/Source/ApeQt/TTYWidget.h
- /home/max/sources/ape/Source/ApeQt/TTYWidget.cpp

5.12 Core::CPU::UnhandledInstructionException Class Reference

Thrown when the [CPU](#) hits an unimplemented [Instruction::Type](#).

```
#include <Exception.h>
```

Inheritance diagram for Core::CPU::UnhandledInstructionException:



5.12.1 Detailed Description

Thrown when the [CPU](#) hits an unimplemented [Instruction::Type](#).

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

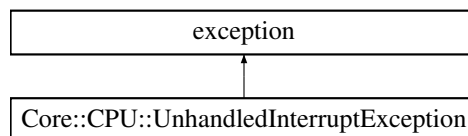
- [/home/max/sources/ape/Source/Core/CPU/Exception.h](#)

5.13 Core::CPU::UnhandledInterruptException Class Reference

Thrown when a interrupt is neither handled by software nor hardware.

```
#include <Exception.h>
```

Inheritance diagram for Core::CPU::UnhandledInterruptException:



5.13.1 Detailed Description

Thrown when a interrupt is neither handled by software nor hardware.

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

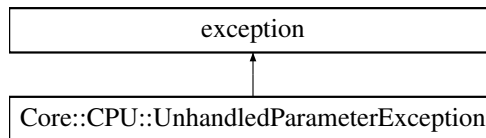
- [/home/max/sources/ape/Source/Core/CPU/Exception.h](#)

5.14 Core::CPU::UnhandledParameterException Class Reference

Thrown when the [CPU](#) hits an unimplemented [Instruction::Parameter::Type](#).

```
#include <Exception.h>
```

Inheritance diagram for Core::CPU::UnhandledParameterException:



5.14.1 Detailed Description

Thrown when the [CPU](#) hits an unimplemented [Instruction::Parameter::Type](#).

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

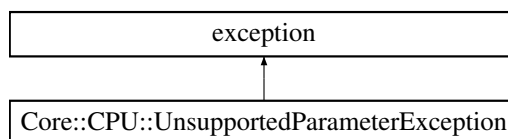
- </home/max/sources/ape/Source/Core/CPU/Exception.h>

5.15 Core::CPU::UnsupportedParameterException Class Reference

Thrown when a instruction doesn't support the parameter type provided.

```
#include <Exception.h>
```

Inheritance diagram for Core::CPU::UnsupportedParameterException:



5.15.1 Detailed Description

Thrown when a instruction doesn't support the parameter type provided.

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

- </home/max/sources/ape/Source/Core/CPU/Exception.h>

Chapter 6

File Documentation

6.1 /home/max/sources/ape/Docs/Terms.dox File Reference

6.1.1 Detailed Description

Terminology

Here are some common terms used in this documentation and their meaning

PC

Personal Computer. Means IBM PC in this context

RAM

Random Access Memory. Means non-sequential, volatile memory in this context

CPU

Central Processing Unit. Most important component of a PC that controls the other component.

Opcode

Operation code. A byte (or byte + reg modifier) that tells the CPU what type of instruction to execute.

Instruction

The smallest logical unit the CPU processes. Usually only performs one task and updates flags accordingly.

Flags

Indicators of the results of previous instructions. Often used in conjunction if conditional jumps.

AF

Adjust Flag. Used when adjusting values (e.g. DAA)

CF

Carry Flag. Used when an arithmetic instruction exceeds the lower bounds of the data type.

OF

Overflow Flag. Used when an instruction exceeds the upper bounds of the data type.

PF

Parity Flag. Indicates whether the result of an instruction has bit-parity (The same amount of 1s and 0s)

SF

Sign Flag. Indicates whether the signed result of an instruction is negative. (If set, negative else positive).

ZF

Zero Flag. Indicates whether the result of an instruction is zero.

IF

Interrupt Flag. Enables / Disables interrupts.

DF

Direction Flag. Used by string instructions to decide whether to increment or decrement after an operation (If set, decrement else increment).

Registers**General purpose****AX/AH/AL**

Often used as an accumulator

BX/BH/BL

Often used to point at data in the data segment.

CX/CH/CL

Often used as a counter (e.g. by REP).

DX/DH/DI

Often used as an I/O pointer.

Segment registers**CS**

Code Segment. Used in conjunction with IP for fetching opcodes

DS

Data Segment. Used by default in most instructions dealing with loading values from memory.

ES

Extra Segment. Used as a destination segment in string instructions

SS

Stack Segment. Used for storing the stack segment.

Pointers**IP**

Instruction Pointer. Used in conjunction with CS:IP to represent the current offset.

BP

Base Pointer. Pointer to data on the stack

SP

Stack Pointer. Pointer to the current top of stack

SI

Source Index. Used as a source pointer by string instruction.

DI

Destination Index. Used as a destination pointer by string instruction.

6.2 /home/max/sources/ape/Source/Common/Logger.h File Reference

```
#include <string>
```

Macros

- `#define LOG(msg) __MSG("LOG", __FILE__, __LINE__, msg)`
Log a message.
- `#define WARN(msg) __MSG("WARNING", __FILE__, __LINE__, msg)`
Log a warning.
- `#define ERROR(msg) __MSG("ERROR", __FILE__, __LINE__, msg)`
Log an error.

6.3 /home/max/sources/apc/Source/Common/Types.h File Reference

```
#include <stdint>
```

Typedefs

- using `u8` = `uint8_t`
A 8-bit unsigned integer (unsigned byte)
- using `i8` = `int8_t`
A 8-bit signed integer (signed byte)
- using `u16` = `uint16_t`
A 16-bit unsigned integer (unsigned word)
- using `i16` = `int16_t`
A 16-bit signed integer (signed word)
- using `u32` = `uint32_t`
A 32-bit unsigned integer.
- using `i32` = `int32_t`
A 32-bit signed integer.
- using `u64` = `uint64_t`
A 64-bit unsigned integer.
- using `i64` = `int64_t`
A 64-bit signed integer.

6.4 /home/max/sources/apc/Source/Core/CPU/CPU.h File Reference

```
#include <atomic>
#include "Common/Types.h"
#include "Core/CPU/Instruction.h"
#include "Core/Memory.h"
```

Classes

- class `Core::CPU::CPU`
Representation of the Central Processing Unit.

6.5 /home/max/sources/ape/Source/Core/CPU/Exception.h File Reference

```
#include <exception>
```

Classes

- class [Core::CPU::InvalidInstructionException](#)
- class [Core::CPU::UnhandledInstructionException](#)
Thrown when the CPU hits an unimplemented [Instruction::Type](#).
- class [Core::CPU::UnhandledParameterException](#)
Thrown when the CPU hits an unimplemented [Instruction::Parameter::Type](#).
- class [Core::CPU::ParameterLengthMismatchException](#)
- class [Core::CPU::UnsupportedParameterException](#)
Thrown when a instruction doesn't support the parameter type provided.
- class [Core::CPU::UnhandledInterruptException](#)
Thrown when a interrupt is neither handled by software nor hardware.

6.6 /home/max/sources/ape/Source/Core/CPU/Instruction.h File Reference

```
#include <map>
#include <string>
#include <vector>
#include "Common/Types.h"
```

Classes

- class [Core::CPU::Instruction](#)
High-Level representation of a instruction.
- class [Core::CPU::Instruction::Parameter](#)
High-Level representation of a instruction parameter.

Functions

- `std::string Core::CPU::TypeToString (const Instruction::Type &type)`
Get the corresponding nmoroic for the Type provided.
- `std::string Core::CPU::ParameterTypeToString (const Instruction::Parameter::Type &type, Instruction::↔ SegmentPrefix prefix=Instruction::SegmentPrefix::None)`
Get a human readable form of the [Parameter::Type](#) provided.
- `bool Core::CPU::ParameterNeedsResolving (const Instruction::Parameter::Type ¶meter)`
Checks whether this [Parameter::Type](#) needs resolving.

6.7 /home/max/sources/ape/Source/Core/HW/FloppyDrive.h File Reference

```
#include <fstream>
#include <memory>
#include <string>
#include <vector>
#include "Common/Types.h"
```

Classes

- class [Core::HW::FloppyDrive](#)
Representation of a floppy drive.

6.8 /home/max/sources/ape/Source/Core/Machine.h File Reference

```
#include "Core/CPU/CPU.h"
#include "Core/HW/FloppyDrive.h"
#include "Core/Memory.h"
#include <vector>
```

Classes

- class [Core::Machine](#)
Representation of a PC.

6.9 /home/max/sources/ape/Source/Core/Memory.h File Reference

```
#include <vector>
#include "Common/Types.h"
```

Classes

- class [Core::Memory](#)
Wrapper around emulated RAM.

Index

[/home/max/sources/ape/Docs/Terms.dox](#), 21
[/home/max/sources/ape/Source/Common/Logger.h](#), 23
[/home/max/sources/ape/Source/Common/Types.h](#), 24
[/home/max/sources/ape/Source/Core/CPU/CPU.h](#), 24
[/home/max/sources/ape/Source/Core/CPU/Exception.h](#), 25
[/home/max/sources/ape/Source/Core/CPU/Instruction.h](#), 25
[/home/max/sources/ape/Source/Core/HW/FloppyDrive.h](#), 26
[/home/max/sources/ape/Source/Core/Machine.h](#), 26
[/home/max/sources/ape/Source/Core/Memory.h](#), 26

Core::CPU::CPU, 9
Core::CPU::Instruction, 12
 [GetLength](#), 13
 [Instruction](#), 13
 [Resolve](#), 13
 [SegmentPrefix](#), 13
Core::CPU::Instruction::Parameter, 16
Core::CPU::InvalidInstructionException, 14
Core::CPU::ParameterLengthMismatchException, 17
Core::CPU::UnhandledInstructionException, 19
Core::CPU::UnhandledInterruptException, 19
Core::CPU::UnhandledParameterException, 20
Core::CPU::UnsupportedParameterException, 20
Core::HW::FloppyDrive, 11
Core::Machine, 14
Core::Memory, 15

[GetLength](#)
 Core::CPU::Instruction, 13

[Instruction](#)
 Core::CPU::Instruction, 13

[MainWindow](#), 15

[Resolve](#)
 Core::CPU::Instruction, 13

[SegmentPrefix](#)
 Core::CPU::Instruction, 13

[TTYBackend](#), 17
[TTYWidget](#), 18