|  |  |  |
| --- | --- | --- |
| Bochs Developers Guide | | |
| [Prev](http://docs.google.com/other.html) |  | [Next](http://docs.google.com/directory-structure.html) |

Chapter 2. About the code

2.1. Overview

The initial versions of some sections in this chapter are based on a document written by Peter "Firefly" Lund. It was added and updated in January 2006.

The Bochs virtual PC consists of many pieces of hardware. At a bare minimum there are always a CPU, a PIT (Programmable Interval Timer), a PIC (Programmable Interrupt Controller), a DMA controller, some memory (this includes both RAM and BIOS ROMs), a video card (usually VGA), a keyboard port (also handles the mouse), an RTC with battery backed NVRAM, and some extra motherboard circuitry.

There might also be an ethernet card, a PCI controller, a soundcard, an IDE controller (+ harddisks/CDROM), a SCSI controller (+ harddisks), a floppy controller, an APIC ...

There may also be more than one CPU.

Most of these pieces of hardware have their own C++ class - and if Bochs is configured to have more than one piece of a type of hardware, each will have its own object.

The pieces of hardware communicates over a couple of buses with each other - some of the things that the buses carry are reads and writes in memory space, reads and writes in I/O space, interrupt requests, interrupt acknowledges, DMA requests, DMA acknowledges, and NMI request/acknowledge. How that is simulated is explained later.

In addition to the simulator itself, some other components are required for the communication with the user. The most important parts are these:

* the window that simulates the monitor and receives keyboard / mouse events
* the configuration interface that allows to adjust simulation settings
* the simulator interface for the communication between the other componnents
* the parameter tree (for configuration settings and save/restore)
* the logfunctions class (handle and configure panic/error/info/debug)

These componnents of Bochs are optional:

* the plugin interface
* the builtin debugger
* the disassembler
* the instrumentation feature

The simulation window is handled by the GUI object (other terms used in the sources are "display library", "VGAW"). There are many different but compatible implementations of the GUI object, depending on whether you compile for X (Unix/Linux), Win32, Macintosh (two versions: one for Mac OS X and one for older OS's), Amiga, etc. The cross-platform libraries SDL and wxWidgets are also supported.

For the configuration interface there are also some different implementations: textconfig (text menus only), wxdialog (wxWidgets port), win32dialog/win32paramdlg (Windows port).

|  |  |  |
| --- | --- | --- |
| [Prev](http://docs.google.com/other.html) | [Home](http://docs.google.com/index.html) | [Next](http://docs.google.com/directory-structure.html) |
| Ideas for other sections |  | Directory Structure |