The Basics of Valkyrie OS

Valkyrie OS — internal documentation

October 16, 2025

Abstract

This is the base guide to the Valkyrie Operating System. In this document would cover the code and features of Valkyrie OS and explains the code in the bootloader and kernel.

1 Introduction

The Valkyrie Operating System is made by Vincent4486 using C, other programming languages like Rust ot Go are expected to be used in the future.

- Target: 32-bit x86 real-mode boot to a small kernel.
- Current status: Stage 1 and Stage 2 bootloaders exist; FAT driver lives in ‘src/bootloader/stage2‘.
- Design: simple monolithic kernel written in C + assembly.

2 Design goals

List your goals here. Examples:

1. Minimal, understandable code.
2. Boot from FAT12/16 image.
3. Small, well-documented boot stages.
4. Incremental testing with tools in ‘tools/‘.

3 Repository layout

Explain the important folders and files so you don’t have to remember them later.

`src/` Source code for the bootloader and kernel.

`src/bootloader/stage1/` Minimal real-mode assembler stage 1.

`src/bootloader/stage2/` C + assembly for the second stage; FAT, disk access and helpers.

`src/kernel/` Kernel entrypoint and kernel-related assembly.

`tools/` Utilities (e.g. FAT tools) useful during development.

`docs/` This documentation.

4 Boot process

Describe how the system boots and which files are responsible for each step. Keep it chronological.

4.1 Stage 1

Explain the stage1 responsibilities (MBR/boot sector constraints) and point to the source file. Example file: ‘src/bootloader/stage1/boot.asm‘.

4.2 Stage 2

What stage2 does (switch to protected mode or load kernel), where the FAT driver is, and how you link it. Key files: ‘src/bootloader/stage2/main.asm‘, ‘src/bootloader/stage2/main.c‘, ‘src/bootloader/stage2/fat.c‘.

5 Important code snippets

You can include code directly or load files from the repo. When you build the PDF locally, listings will include the exact source.

Inline C example:

Listing 1: Minimal example from stdio.c

```c
// src/bootloader/stage2/stdio.c
#include "stdio.h"
int puts(const char *s) {
  while (*s) putchar(*s++);
  return 0;
}
```

Inline assembly example:

Listing 2: Boot sector entry (example)

```asm
; src/bootloader/stage1/boot.asm
org 0x7c00
cli
xor ax, ax
mov ds, ax
; ...
```

To include an entire file from the tree (recommended for accuracy):

```
\codefile{C}{../src/bootloader/stage2/main.c}
```

Replace the path above with the relative path from ‘docs/‘ to a source file.

6 Build and run

Keep simple reproducible commands here. From the project root (one level above ‘docs/‘):

- Build OS image: `make`
- Run (QEMU or emulator): `make run`

How to build this PDF (from ‘docs/‘):

```
# Recommended: use latexmk if available
latexmk -pdf -interaction=nonstopmode The_Basics_of_Valkyrie_OS.tex

# Or the simple sequence
pdflatex The_Basics_of_Valkyrie_OS.tex
bibtex The_Basics_of_Valkyrie_OS || true
pdflatex The_Basics_of_Valkyrie_OS.tex
pdflatex The_Basics_of_Valkyrie_OS.tex
```

Notes:

- Using ‘minted‘ gives better highlighting but requires ‘-shell-escape‘ and Pygments; ‘listings‘ works everywhere.
- If you include large files from ‘src/‘ the PDF will reflect the current source at compile time.

7 Development notes

Keep short actionable items here: known bugs, TODOs, testing checklist, and where to find test images.

8 Writing tips

Short recommendations to keep docs usable:

- Write short focused sections with a single purpose.
- Add relative paths when referencing source (so you can jump to the file from the PDF viewer in many editors).
- Use `\label{}` and `\ref{}` to cross-reference figures, sections, and listings.
- Prefer including source with `\lstinputlisting` so examples stay correct.

9 Appendix: Useful file list

Add short one-line descriptions for the most important files so you don't have to hunt later.

Last edited: October 16, 2025