1. Verkennen

In deze opdracht ga je op onderzoek uit door de code van FRAMBOOS en voeg je enkele eenvoudige functies toe om te oefenen met C.

Structuur

De code van FRAMBOOS is opgedeeld in drie mappen:

- `kernel/` bevat alle kernel-mode functionaliteit, inclusief drivers voor en informatie over hardware in `kernel/hardware/`.
- `user/` gaat all user-mode functionaliteit bevatten, die moet nog geschreven worden;
- `shared/` bevat functionaliteit die zowel in kernel- als user-mode gebruikt kan worden, zoals een mini string formatting library in `shared/strfmt.{h,c}`.

Booting

Zoek code op die er voor zorgt dat de kernel geladen en gestart wordt en beantwoord de volgende vragen.

1. Welke bestanden en welke functies worden gebruikt om de kernel te starten?
2. Wat voor soort bronbestanden zijn dit? (C broncode, assembly, iets anders)
3. Hoe verwijzen deze bestanden naar elkaar?

Framebuffer

Bekijk `kernel/hardware/framebuffer.{h,c}` en beantwoord de volgende vragen.

4. Hoe is de data van de framebuffer opgeslagen? (globaal, lokaal, iets anders)
5. Hoe wordt deze data beschikbaar gemaakt aan andere C modules?

Tekenen

Bekijk nu `shared/graphics.{h,c}`. Deze bestanden bevat (de skeletten van) functies om pixels te vullen in de framebuffer. De functies `void draw_pixel(framebuffer_info_t*, point_t, color_t)` en `void draw_character(framebuffer_info_t*, point_t, color_t, char)` zijn al gegeven.

6. Schrijf de functie `void draw_rectangle(framebuffer_info_t *fb, area_t area, color_t color)` die een rechthoek van grootte `area` en kleur `color` tekent in framebuffer `fb`. Zorg er voor dat je iedere pixel binnen deze rechthoek vult.
7. Schrijf de functie `void draw_string(framebuffer_info_t *fb, point_t pos, color_t col, const char *str)` die elk karakter uit de string `str` tekent in kleur `color` beginnende bij positie `pos` in framebuffer `fb`. Maak hierbij gebruik van `draw_character`.

Panic

In `kernel/panic.{h,c}` is de code gedefinieerd voor een kernel panic. Op dit moment schrijft deze alleen een bericht naar het UART kanaal.

8. Pas de functie `void kernel_panic_implementation(const char*)` aan met de volgende functionaliteit: a. Teken een rechthoek, bij voorkeur in blauw, van 1000 bij 200 pixels in het midden van het scherm. b. Druk (grofweg) in het midden van deze rechthoek, bij voorkeur in wit, de meegegeven foutmelding af.
9. Pas `void kernel_main()` in `kernel.c` dusdanig aan, dat er direct na initialisatie een passende kernel panic wordt weergegeven.

Inleveren

Lever een zip *van je hele project* in via yOUlearn. Probeer dit te doen vóór de volgende bijeenkomst, dus voor dinsdag 6 december. Mocht dit niet lukken, neem dan contact op met de docent. Geef hieronder aan hoeveel uren je met deze opdracht bezig bent geweest.

Tijd gestoken in deze opdracht: … uur.

Het moeilijkste aan deze opdracht vond ik:

- …

Het leukste aan deze opdracht vond ik:

- …