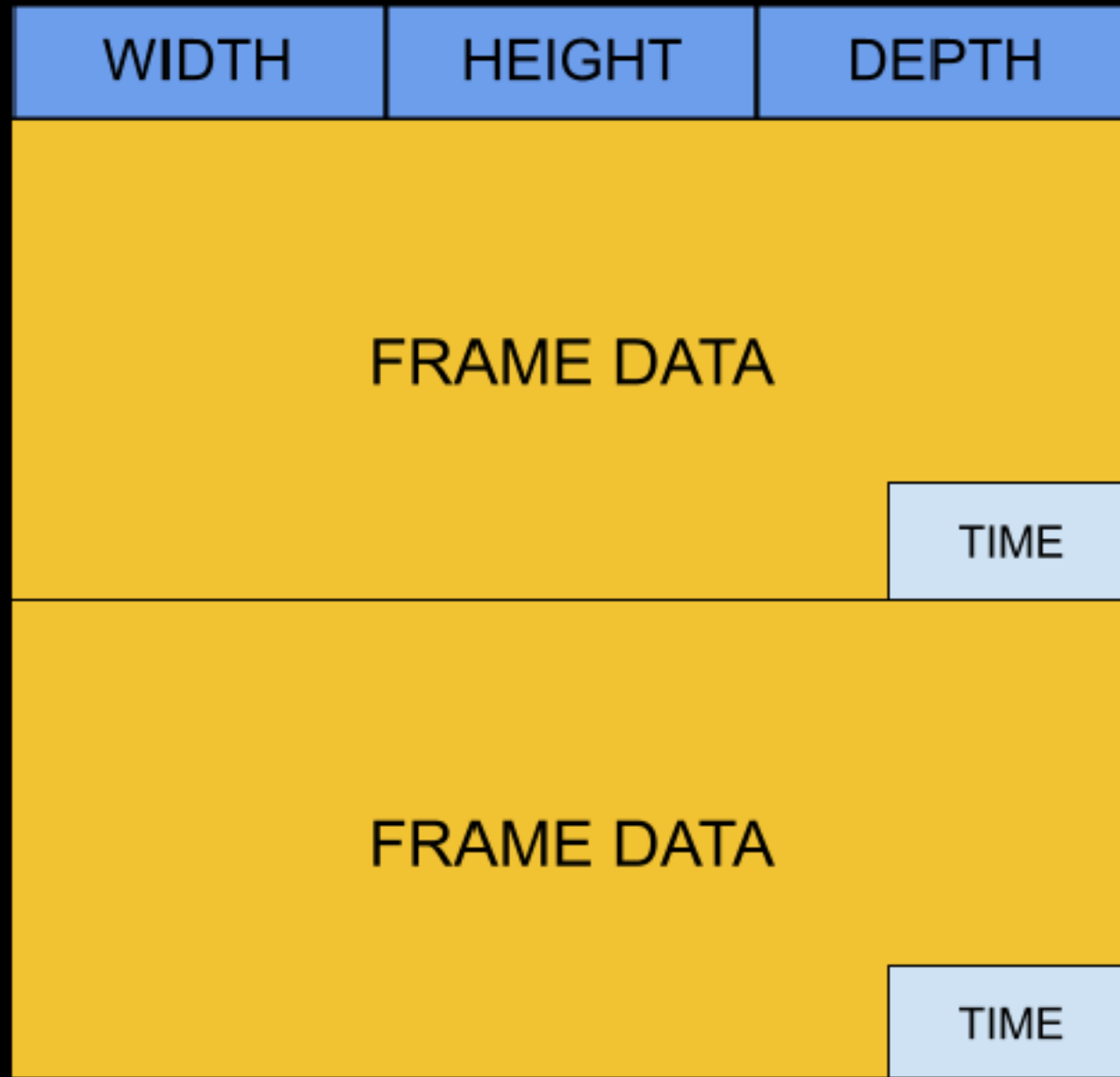cfb

Features:
- Record framebuffer data
- Compile to desired format

```
Usage:
$ cfb (record|compile) FILE <ARGS>
```

Recording

```
$ cfb record FILE (--device /dev/fb0)
```

```
       /\            blakely@tau
      /  \
     /\   \          os      Arch Linux
    /      \         host    TOSHIBA NB305 PLL3AC-02P014
   /   ,,   \        kernel  6.3.1-arch1-1
  /   |  |  -\       uptime  1d 32m
 /_-''    ''-_\      pkgs    1133
                     memory  126M / 955M
```

```
[blakely@tau] ~/school/math
$
```

```
  0[||||||||||||||||||87.7%] Tasks: 36, 3 thr, 80 kthr; 2 r
  1[|||||||||||||||||||99.4%] Load average: 5.13 4.86 4.33
  Mem[||||||||||||151M/956M] Uptime: 1 day, 00:32:44
  Swp[||          256K/1.50G]

[Main] [I/O]
  PID USER      PRI  NI  VIRT   RES   SHR S  CPU% MEM%▽   TIME
    1 root       20   0  101M 14700  9776 S   0.0  1.5  0:20.
 1006 root       20   0  542M 25868 18432 S   0.0  2.6  0:05.
 1202 root       20   0  542M 25868 18432 S   0.0  2.6  0:00.
 1210 root       20   0  542M 25868 18432 S   0.0  2.6  0:00.
 1211 root       20   0  542M 25868 18432 S   0.0  2.6  0:01.
  162 root       20   0 48920 15928 14768 S   0.0  1.6  0:01.
 2461 root       20   0 17560 11588 10016 S   0.0  1.2  0:00.
 1793 blakely    20   0 18888 11012  9216 S   0.0  1.1  0:03.
 1798 blakely    20   0  103M  5548  1992 S   0.0  0.6  0:00.
  182 root       20   0 33404  9924  7592 S   0.0  1.0  0:04.
 1234 root       20   0 24180  9248  7456 S   2.4  0.9  2:56.
 1496 root       20   0  7800  5092  4364 S   0.0  0.5  0:00.
 1981 blakely    20   0  7940  4452  3552 S   0.0  0.5  0:00.
F1Help  F2Setup F3Search F4Filter F5List  F6SortBy F7Nice -F8Nice +
```

```rust
22 mod record;
21 mod compile;
20 mod cli;
19
18 use compile::OutputFormat;
17 use cli::{Cli, Commands};
16 use clap::Parser;
15
14 /// Uses [clap](`clap`) as implemented in module [cli](`cra
13 /// to parse arguments to pass them to the relevant functio
12 fn main() -> Result<(), std::io::Error> {
11     let args = Cli::parse();
10     match &args.command {
 9         Commands::Record(arg) => {
 8             record::capture(arg.device.as_str(), arg.file.a
 7         },
 6         Commands::Compile(arg) => {
 5             compile::compile(arg.file.as_str(), &arg.format
 4         }
 3     };
 2
 1     Ok(())
62 }
```

```
src/main.rs                                    62,1            Bot
```

Compiling

```
$ cfb compile FILE -o OUTPUT (--format mp4)
```

```
<OUTPUT>1      <OUTPUT>1.png      <OUTPUT>.mp4

<OUTPUT>2      <OUTPUT>2.png

<OUTPUT>N      <OUTPUT>N.png
```

But I'm not done yet..

Planned Features

- `--append`
- `--run-on-start`
- `check`

Thank you.
Questions?