Pattern Table Documentation

1. Overview

This document defines the binary file (.dat) format used by the Pattern Table Pipeline.
The format is designed for efficient playback of time-based LED animations with optical fibers (OF) and LED strips.
The system consists of two binary files:

File	Purpose
control.dat	Global and hardware configuration
frame.dat	Sequential frame-based animation data

2. Data Format

Item	Specification
File type	Binary (.dat)
Byte order	Little endian
Numeric types	Unsigned integers only (uint8, uint32)
Color format	RGB888 (GRB order)

- Channel : we define channel as a single controllable lighting unit, which can be either an Optical Fiber or an LED strip
  - OF (Optical Fiber) Channel : one optical fiber corresponds to exactly one channel and contains a single color value defined by one GRB triplet
  - LED Strip Channel : one strip corresponds to one channel, but contains multiple color values, where each LED on the strip has its own GRB triplet

- Due to hardware data storage requirements, our color values are stored in GRB order instead of RGB

3. “control.dat” Specification

3.1 Purpose

The “control.dat” defines the global configuration parameters at the beginning. All frames in “frame.dat” must follow this configuration.

3.2 File Layout

All fields are 1 byte/uint8 unless otherwise specified

Offset	Element	Size	Description
0,1	Version	2 byte (uint8 * 2) little-endian byte order	Curent version is 1.2
2	OF_channel[0]	1 byte (uint8)	Whether OF 0 is enable (True=1 / False=0)
3	OF_channel[1]	1 byte (uint8)	Whether OF 1 is enable (True=1 / False=0)
…	…	…	…
41	OF_channel[39]	1 byte (uint8)	Whether OF 39 is enable (True=1 / False=0)
42	Strip_channel[0]	1 byte (uint8)	Number of LEDs on Strip 0
43	Strip_channel[1]	1 byte (uint8)	Number of LEDs on Strip 1
…	…	…	…
49	Strip_channel[7]	1 byte (uint8)	Number of LEDs on Strip 7
50	Frame_num	4 byte (uint32) little-endian byte order	Number of total frame in this version
54	time_stamp[0]	4 byte (uint32) little-endian byte order	Start time of frame 0
58	time_stamp[1]	4 byte (uint32) little-endian byte order	Start time of frame 1
…	…	…	…
50 + 4 * Frame_num	time_stamp [Frame_num]	4 byte (uint32) little-endian byte order	Start time of last frame
54 + 4 * Frame_num	checksum	uint32 little-endian byte order	Checkpoint of total “control.dat”

Version : The version is stored as two separate bytes (uint8) in little–endian order

- Byte 0 = major version (main version number)
- Byte 1 = minor version (sub–version number)
- For example : version 1.2 is save as Byte 0 = 1, Byte 1 = 2

Term definition:

- OF_num: total number of enabled OF. Range 0~40
- Strip_num: total number of enabled Strip. Range 0~8
- LED_num: The total number of LEDs on all Strips . Range 0~800

Checksum definition : we use 4 bytes to save a checkpoint for total “control.dat”, define Checksum = (Σall bytes in “control.dat”) mod 2^32, which include version, OF_channel[], LED_channel[], frame_num, time_stamp[].

OF order follows the I2C address sequence on the ESP32 as table below:

OF order	OF 0~4	OF 5~9	OF 10~14	OF 15~19
I2C address	0x1F	0x20	0x22	0x23
OF order	OF 20~24	OF 25~29	OF 30~34	OF 35~39
I2C address	0x5b	0x5c	0x5e	0x5f

Strip order follows the GPIO pin sequence on the ESP32 as table below:

Strip order	Strip 0	Strip 1	Strip 2	Strip 3
GPIO	GPIO_NUM_32	GPIO_NUM_25	GPIO_NUM_26	GPIO_NUM_27
Strip order	Strip 4	Strip 5	Strip 6	Strip 7
GPIO	GPIO_NUM_19	GPIO_NUM_18	GPIO_NUM_5	GPIO_NUM_17

LED order within a strip is determined by distance from the connection point — the closer an LED is to the wiring connection, the lower its index (starting from 0)

4. “frame.dat” Specification

4.1 File Layout

“frame.dat” is a frame sequence with no global header

Offset	Element	Size	Description
0,1	Version	2 byte (uint8 * 2) little-endian byte order	Curent version is 1.2
2 ~ 2 + Frame_size	Frame 1	Frame_size byte (uint8)	Data of first frame
2 + Frame_size ~ 2 + 2*Frame_size	Frame 2	Frame_size byte (uint8)	Data of second frame
…	…	…	…
2 + (n-1)*Frame_size ~ 2 + n*Frame_size	Frame n	Frame_size byte (uint8)	Data of frame n

Frame count is determined by end-of-file, no padding or delimiter exists between frames.

A single frame consists data below (in order) :

Field	Size	Description
start_time	uint32 little-endian byte order	Frame start timestamp
fade	uint8	Fade enable flag (True=1 / False=0)
OF GRB data	uint8 total OF_num * 3 byte	GRB color data of all OF in current frame
LED GRB data	uint8 total LED_num * 3 byte	GRB color data of all Strip in current frame
checksum	uint32 little-endian byte order	Checkpoint of a total frame

where Frame_size stands for all bytes in a single frame.

Fade definition : we use 4 bytes to determine whether a color transition is applied between the current frame and the next frame

- If fade is set to true in frame i, the lighting output shall smoothly transition from the colors defined in frame i to the colors defined in frame i+1.
- If fade is set to false, the colors of frame i+1 shall be applied without transition.

Checksum definition : we use 1 bytes to save a checkpoint for a frame, define Checksum = (Σall bytes in frame) mod 2^32, which include start_time, fade, OF GRB data, LED GRB data

4.2 GRB Data Layout

OF channel GRB data

OF[0].G	OF[0].R	OF[0].B	OF[1].G	OF[1].R	OF[1].B	…	OF[n].G	OF[n].R	OF[n].B

- 1 byte (uint8, 0~255) for each R/G/B
- Total size = OF_num × 3 byte
- Stored sequentially for i = 0 … n = OF_num − 1

Strip channel GRB Data

First ordered by strip index, then by LED index

LED[0][0].G	LED[0][0].R	LED[0][0].B	LED[0][1].G	LED[0][1].R	LED[0][1].B	…	LED[0][n_1].G	LED[0][n_1].R	LED[0][n_1].B
LED[1][0].G	LED[1][0].R	LED[1][0].B	LED[1][1].G	LED[1][1].R	LED[1][1].B	…	LED[1][n_2].G	LED[1][n_2].R	LED[1][n_2].B
…	…	…	…	…	…		…	…	…
LED[m][0].G	LED[m][0].R	LED[m][0].B	LED[m][1].G	LED[m][1].R	LED[m][1].B	…	LED[m][n_m].G	LED[m][n_m].R	LED[m][n_m].B

- 1 byte (uint8, 0~255) for each G, R, B
- Total size = LED_num * 3 byte
- LED[i][j] Stored sequentially for i = 0 … m = Strip_num − 1, j = 0 … n_i = Strip_channel[I]
  - For LED[i][j], the index j starts from 0 at the LED order by the distance from the connection point (see **3.2 File Layout**)
  - For example : LED[i][j] stand for the j−th LED on i−th strip

5. Consistency Rules

- “frame.dat” must be parsed using the corresponding “control.dat”
- OF_num, Strip_num, and must match in each frame of “frame.dat” and “control.dat”
- Frames are expected to be ordered by non−decreasing start_time
- Behavior is undefined if frames are not sorted by time

6.Revision History

Revision	Date	Descripition
0.1	1/19	testing version of the documentation
1.0	1/21	Same as v0.1. pass the API test on ESP32
1.0.1	1/22	Added explicit definition of strip and LED ordering logic at part: • Strip order follows GPIO pin sequence (Strip 0—7 mapping to GPIO_NUM_xx) in **3.2 File Layout** • LED order within a strip is defined by physical distance from ESP32 (closest LED = index 0) in **3.2 File Layout**
1.1.0	1/31	• Fix typo • Update format of Control.dat in **3.2 File Layout** • Rename Strip_channel in **3.2 File Layout** • Add order of OF in **3.2 File Layout** • Delete **3.3 Field Constraints** • Update LED order in **4.2 GRB Data Layout**
1.2.0	2/6	• Fix offset, clear description, update version and add checksum in **3.2 File Layout** • Fix typo, clear description in **4.1 File Layout**