```
/**
 * Copyright (c) 2020 Raspberry Pi (Trading) Ltd.
 *
 * SPDX-License-Identifier: BSD-3-Clause
 */
```

These are all the header files

```
#include <stdio.h>
#include <stdlib.h>

#include "pico/stdlib.h"
#include "hardware/pio.h"
#include "hardware/clocks.h"
#include "ws2812.pio.h"
```

Here we are defining IS_RGBW as true & NUM_PIXELS as 150

```
#define IS_RGBW true
#define NUM_PIXELS 150
```

Check if WS2812 pin already has a connected pin, then WS2812_PIN will get that default PIN. If not then it will be assigned 2.

```
#ifdef PICO_DEFAULT_WS2812_PIN
#define WS2812_PIN PICO_DEFAULT_WS2812_PIN
#else
// default to pin 2 if the board doesn't have a default WS2812 pin defined
#define WS2812_PIN 2
#endif
```

To write word in TXFIFO of state machine, if it is already full then it will block.

```
static inline void put_pixel(uint32_t pixel_grb) {
    pio_sm_put_blocking(pio0, 0, pixel_grb << 8u);
}
```

It returns 32 bit RGB pixel value

```
static inline uint32_t urgb_u32(uint8_t r, uint8_t g, uint8_t b) {
```

Here we will Left shift 'r' & 'g' value by 8 & 16 bits respectively. OR value and it will then return 32-bits with 'r', 'g' & 'b' 8 bit each.

```
    return
        ((uint32_t) (r) << 8) |
        ((uint32_t) (g) << 16) |
        (uint32_t) (b);
}
```

Here we are taking remainder & then in if & else the Red value is set highest while Green & blue are Lowest, after that G is highest & then B is highest rest are Lowest in each case. In the End we will clean the LEDs by put_pixel(0)

```
void pattern_snakes(uint len, uint t) {
    for (uint i = 0; i < len; ++i) {
        uint x = (i + (t >> 1)) % 64;
        if (x < 10)
            put_pixel(urgb_u32(0xff, 0, 0));
        else if (x >= 15 && x < 25)
            put_pixel(urgb_u32(0, 0xff, 0));
        else if (x >= 30 && x < 40)
            put_pixel(urgb_u32(0, 0, 0xff));
        else
            put_pixel(0);
    }
}
```

Here it is checking the remainder of t. i.e, it is divisible by 8 then get out otherwise add the values of random pixels

```
void pattern_random(uint len, uint t) {
    if (t % 8)
        return;
    for (int i = 0; i < len; ++i)
        put_pixel(rand());
}
```

Here we can see if 't' is divisible by 8, then we will exit the function.

```
void pattern_sparkle(uint len, uint t) {
    if (t % 8)
```

```
    return;
  for (int i = 0; i < len; ++i)
    put_pixel(rand() % 16 ? 0 : 0xffffffff);
}
```

After putting the pixel value, it will make t = 0, if its value exceeds max

```
void pattern_greys(uint len, uint t) {
  int max = 100; // let's not draw too much current!
  t %= max;
  for (int i = 0; i < len; ++i) {
    put_pixel(t * 0x10101);
    if (++t >= max) t = 0;
  }
}
```

The table

```
typedef void (*pattern)(uint len, uint t);
const struct {
  pattern pat;
  const char *name;
} pattern_table[] = {
    {pattern_snakes,  "Snakes!"},
    {pattern_random,  "Random data"},
    {pattern_sparkle, "Sparkles"},
    {pattern_greys,   "Greys"},
};

int main() {
  //set_sys_clock_48();
  stdio_init_all();
  printf("WS2812 Smoke Test, using pin %d", WS2812_PIN);
```

Here we will initialize all present standard stdio types (which are binary linked)

Here we will display the default pin

```
// todo get free sm

PIO pio = pio0;
int sm = 0;
uint offset = pio_add_program(pio, &ws2812_program);
ws2812_program_init(pio, sm, offset, WS2812_PIN, 800000, IS_RGBW);

int t = 0;

while (1) {

    int pat = rand() % count_of(pattern_table);
    int dir = (rand() >> 30) & 1 ? 1 : -1;
    puts(pattern_table[pat].name);
    puts(dir == 1 ? "(forward)" : "(backward)");
    for (int i = 0; i < 1000; ++i) {

        pattern_table[pat].pat(NUM_PIXELS, t);
        sleep_ms(10);
        t += dir;
    }

  }

}
```

From the two available pio ports 1 and 2 we will choose which instance to use

Here we are setting the state machine to 0.

For the program, find allocation where it has enough space for the program.

An instant of the desired program is instantiated

Here we will assign random variables/number in range b/w 0 to size of pattern_table

Check if the logical & is sig-bit shifting 1 yes then random no. 2 one 1 otherwise -1.

Here we are writting string to stdout

dir is equal to 1 then write forward otherwise backward to stdout

Here we are calling the pattern function

Here it is waiting for 10 ms

Add the direction value to t