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

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

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

Including header files needed to run the code

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

defining constants

```c
#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
```

Checking if default pin is assigned in one of the headers

won't return a value

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

defining function put_pixel, making it static and inline

P Writing a word of data, Blocking if FIFO full

static inline again

```c
static inline uint32_t urgb_u32(uint8_t r, uint8_t g, uint8_t b) {
    return
            ((uint32_t) (r) << 8) |
            ((uint32_t) (g) << 16) |
            (uint32_t) (b);
}
```

32 bit unsigned guaranteed

move r bits left 8 bits

move g bits left 16

reorganizing bits basically encoding as 24 binary bit #

28

```c
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
```

Creating a Pattern for led

Changing the LED value based on X to make pattern

Setting to certain color

Setting to certain color

Setting to certain color

```
                put_pixel(0);  Setting Led to 0
            }
        }

                     defining function
defining  void pattern_random(uint len, uint t) {
2nd           if (t % 8)      if  t/8 = 0  return
pattern           return;
              for (int i = 0; i < len; ++i)   defining till I is certain value
                  put_pixel(rand());  set Led random value
          }

                     defining function
defining  void pattern_sparkle(uint len, uint t) {
third         if (t % 8)
pattern           return;                     Same as above
              for (int i = 0; i < len; ++i)
                  put_pixel(rand() % 16 ? 0 : 0xffffffff);  Putting random value between 0, ffffff
          }
                         if rand/16 = 0 then

defining  void pattern_greys(uint len, uint t) {
4th           int max = 100; // let's not draw too much current!  setting max value of 100%
pattern       t %= max;
              for (int i = 0; i < len; ++i) {  defining for certain length
                  put_pixel(t * 0x10101);
                  if (++t >= max) t = 0;
              }
          }

          typedef void (*pattern)(uint len, uint t);
          const struct {                            Building
              pattern pat;                          a vector
              const char *name;                     With each
          } pattern_table[] = {                     Pattern
Array             {pattern_snakes,  "Snakes!"},
name              {pattern_random,  "Random data"},
                  {pattern_sparkle, "Sparkles"},    4 patterns
                  {pattern_greys,   "Greys"},       defined previously
          };

          int main() {
              //set_sys_clock_48();  commented out
initialize    stdio_init_all();①
all stdio     printf("WS2812 Smoke Test, using pin %d", WS2812_PIN);② Print which pin its using
types linked

              // todo get free sm
              PIO pio = pio0;③  setting the Pio Pin to pio 0
                                first Pio hardware instance
```

```
                                     defining an integel              Program name
    int sm = 0; 4                                          Pio0
    uint offset = pio_add_program(pio, &ws2812_program); 5
                   Trying to load Program
initalize
    ws2812_program_init(pio, sm, offset, WS2812_PIN, 800000, IS_RGBW); 6   Pulling from function
Program                      Pio0  0   uint  default Pin       True           other Printout

  22 int t = 0; new int
    while (1) {
While loop  23 int pat = rand() % count_of(pattern_table); randomly pick which pattern
to continuously  24 int dir = (rand() >> 30) & 1 ? 1 : -1;   Pick random number
run LED  25 puts(pattern_table[pat].name); Pick pattern in array
        26 puts(dir == 1 ? "(forward)" : "(backward)");  if dir=1 then forward
        for (int i = 0; i < 1000; ++i) { go for 1000 iterations
        27  pattern_table[pat].pat(NUM_PIXELS, t);
                                               150
            sleep_ms(10); Stal Momentarily
            t += dir; add dir to t
        }
    }
}
```

```
// -------------------------------------------------- //
// This file is autogenerated by pioasm; do not edit! //
// -------------------------------------------------- //

#pragma once

#if !PICO_NO_HARDWARE
#include "hardware/pio.h"
#endif
```

Checking if hardware already defined then defining it if not

```
// ------ //
// ws2812 //
// ------ //

#define ws2812_wrap_target 0
#define ws2812_wrap 3

#define ws2812_T1 2
#define ws2812_T2 5
#define ws2812_T3 3
```

defining Variables

```
static const uint16_t ws2812_program_instructions[] = {
            //     .wrap_target
    0x6221, //  0: out    x, 1            side 0 [2]
    0x1123, //  1: jmp    !x, 3           side 1 [1]
    0x1400, //  2: jmp    0               side 1 [4]
    0xa442, //  3: nop                    side 0 [4]
            //     .wrap
};
```

program instructions array

locations

Commented out

```
#if !PICO_NO_HARDWARE
static const struct pio_program ws2812_program = {
    .instructions = ws2812_program_instructions,
    .length = 4,
    .origin = -1,
};
```

if hardware not defined then using program instructions from above

Pio Configuration structure

Setting default config

```
static inline pio_sm_config ws2812_program_get_default_config(uint offset) {
    pio_sm_config c = pio_get_default_sm_config();
    sm_config_set_wrap(&c, offset + ws2812_wrap_target, offset + ws2812_wrap);
    sm_config_set_sideset(&c, 1, false, false);
    return c;
}
```

Pio config struct

Getting default config

offset from other file (10)

(11)

(12)

(13)

config Structure

Sets the wrap and sideset options in the state machine that it pulled from default config

```
#include "hardware/clocks.h"
```

Include header

no output

init Program based on Previously found Values

```
static inline void ws2812_program_init(PIO pio, uint sm, uint offset, uint pin,
float freq, bool rgbw) { 7
init Pin pio_gpio_init(pio, pin);
    pio_sm_set_consecutive_pindirs(pio, sm, pin, 1, true); Setting Consecutive Pin directions 8
    pio_sm_config c = ws2812_program_get_default_config(offset); Setting C to default config 9
 14 sm_config_set_sideset_pins(&c, pin); Setting Side set Pins [redacted]
 15 sm_config_set_out_shift(&c, false, true, rgbw ? 32 : 24); Setting up outshifting [redacted]
 16 sm_config_set_fifo_join(&c, PIO_FIFO_JOIN_TX); Setting up FiFo Joining in State machine [redacted]
 17 int cycles_per_bit = ws2812_T1 + ws2812_T2 + ws2812_T3;                 Clearing Value so You [redacted]
 18 float div = clock_get_hz(clk_sys) / (freq * cycles_per_bit);            can set the Clock Speed [redacted]
 19 sm_config_set_clkdiv(&c, div); Sets State Machine Clock divider [redacted]
 20 pio_sm_init(pio, sm, offset, &c); resets and configures to Constant State [redacted]
 21 pio_sm_set_enabled(pio, sm, true); enables State machine [redacted]
}

#endif
```

Referencing config structure C

```
// --------------- //
// ws2812_parallel //
// --------------- //

#define ws2812_parallel_wrap_target 0
#define ws2812_parallel_wrap 3

#define ws2812_parallel_T1 2
#define ws2812_parallel_T2 5
#define ws2812_parallel_T3 3

static const uint16_t ws2812_parallel_program_instructions[] = {
            //     .wrap_target
    0x6020, //  0: out    x, 32
    0xa10b, //  1: mov    pins, !null            [1]
    0xa401, //  2: mov    pins, x                [4]
    0xa103, //  3: mov    pins, null             [1]
            //     .wrap
};

#if !PICO_NO_HARDWARE
static const struct pio_program ws2812_parallel_program = {
    .instructions = ws2812_parallel_program_instructions,
    .length = 4,
    .origin = -1,
};
```

Same as above but for Parallel example Which we didn't do

```
static inline pio_sm_config ws2812_parallel_program_get_default_config(uint
offset) {
    pio_sm_config c = pio_get_default_sm_config();
    sm_config_set_wrap(&c, offset + ws2812_parallel_wrap_target, offset +
ws2812_parallel_wrap);
    return c;
}

#include "hardware/clocks.h"
static inline void ws2812_parallel_program_init(PIO pio, uint sm, uint offset,
uint pin_base, uint pin_count, float freq) {
    for(uint i=pin_base; i<pin_base+pin_count; i++) {
        pio_gpio_init(pio, i);
    }
    pio_sm_set_consecutive_pindirs(pio, sm, pin_base, pin_count, true);
    pio_sm_config c = ws2812_parallel_program_get_default_config(offset);
    sm_config_set_out_shift(&c, true, true, 32);
    sm_config_set_out_pins(&c, pin_base, pin_count);
    sm_config_set_set_pins(&c, pin_base, pin_count);
    sm_config_set_fifo_join(&c, PIO_FIFO_JOIN_TX);
    int cycles_per_bit = ws2812_parallel_T1 + ws2812_parallel_T2 +
ws2812_parallel_T3;
    float div = clock_get_hz(clk_sys) / (freq * cycles_per_bit);
    sm_config_set_clkdiv(&c, div);
    pio_sm_init(pio, sm, offset, &c);
    pio_sm_set_enabled(pio, sm, true);
}

#endif
```