


 master  [pico-examples / pio / ws2812 / generated / ws2812.pio.h](#)

[Go to file](#) 

 [kilogramham](#) regenerated pio headers (#180) 

Latest commit c507d54 on Oct 28, 2021 [History](#)

 1 contributor

114 lines (95 sloc) 3.63 KB

[Raw](#) [Blame](#)    

```
1 // ----- //
2 // This file is autogenerated by pioasm; do not edit! //
3 // ----- //
4
5 #pragma once
6
7 #if !PICO_NO_HARDWARE
8 #include "hardware/pio.h"
9 #endif
10
11 // ----- //
12 // ws2812 //
13 // ----- //
14
15 #define ws2812_wrap_target 0
16 #define ws2812_wrap 3
17
18 #define ws2812_T1 2
19 #define ws2812_T2 5
20 #define ws2812_T3 3
21
22 static const uint16_t ws2812_program_instructions[] = {
23     // .wrap_target
24     0x6221, // 0: out x, 1 side 0 [2]
25     0x1123, // 1: jmp lx, 3 side 1 [1]
26     0x1400, // 2: jmp 0 side 1 [4]
27     0xa442, // 3: nop side 0 [4]
28     // .wrap
29 };
30
31 #if !PICO_NO_HARDWARE
32 static const struct pio_program ws2812_program = {
33     .instructions = ws2812_program_instructions,
34     .length = 4,
35     .origin = -1,
36 };
37
38 static inline pio_sm_config ws2812_program_get_default_config(uint offset) {
39     pio_sm_config c = pio_get_default_sm_config();
40     sm_config_set_wrap(&c, offset + ws2812_wrap_target, offset + ws2812_wrap);
41     sm_config_set_sideset(&c, 1, false, false);
42     return c;
43 }
44
45 #include "hardware/clocks.h"
46 static inline void ws2812_program_init(PIO pio, uint sm, uint offset, uint pin, float freq, bool rgbw) {
47     pio_gpio_init(pio, pin);
48     pio_sm_set_consecutive_pindirs(pio, sm, pin, 1, true);
49     pio_sm_config c = ws2812_program_get_default_config(offset);
50     sm_config_set_sideset_pins(&c, pin);
51     sm_config_set_out_shift(&c, false, true, rgbw ? 32 : 24); (public)
52     sm_config_set_fifo_join(&c, PIO_FIFO_JOIN_TX);
53     int cycles_per_bit = ws2812_T1 + ws2812_T2 + ws2812_T3;
54     float div = clock_get_hz(clk_sys) / (freq * cycles_per_bit); slow the state machine's execution down
55     sm_config_set_clkdiv(&c, div);
56     pio_sm_init(pio, sm, offset, &c);
57     pio_sm_set_enabled(pio, sm, true);
58 }
59
60 #endif
61
62 // ----- //
63 // ws2812_parallel //
64 // ----- //
65
66 #define ws2812_parallel_wrap_target 0
67 #define ws2812_parallel_wrap 3
68
69 #define ws2812_parallel_T1 2
70 #define ws2812_parallel_T2 5
71 #define ws2812_parallel_T3 3
72
73 static const uint16_t ws2812_parallel_program_instructions[] = {
74     // .wrap_target
75     0x6020, // 0: out x, 32
76     0xa10b, // 1: mov pins, !null [1]
77     0xa401, // 2: mov pins, x [4]
78     0xa103, // 3: mov pins, null [1]
79     // .wrap
80 };
81
82 #if !PICO_NO_HARDWARE
83 static const struct pio_program ws2812_parallel_program = {
84     .instructions = ws2812_parallel_program_instructions,
85     .length = 4,
86     .origin = -1,
87 };
88
89 static inline pio_sm_config ws2812_parallel_program_get_default_config(uint offset) {
90     pio_sm_config c = pio_get_default_sm_config();
91     sm_config_set_wrap(&c, offset + ws2812_parallel_wrap_target, offset + ws2812_parallel_wrap);
92     return c;
93 }
94
95 #include "hardware/clocks.h"
96 static inline void ws2812_parallel_program_init(PIO pio, uint sm, uint offset, uint pin_base, uint pin_count, float freq) {
97     for(uint i=pin_base; i<pin_base+pin_count; i++) {
98         pio_gpio_init(pio, i);
99     }
100     pio_sm_set_consecutive_pindirs(pio, sm, pin_base, pin_count, true);
101     pio_sm_config c = ws2812_parallel_program_get_default_config(offset);
102     sm_config_set_out_shift(&c, true, true, 32);
103     sm_config_set_out_pins(&c, pin_base, pin_count);
104     sm_config_set_set_pins(&c, pin_base, pin_count);
105     sm_config_set_fifo_join(&c, PIO_FIFO_JOIN_TX);
106     int cycles_per_bit = ws2812_parallel_T1 + ws2812_parallel_T2 + ws2812_parallel_T3;
107     float div = clock_get_hz(clk_sys) / (freq * cycles_per_bit);
108     sm_config_set_clkdiv(&c, div);
109     pio_sm_init(pio, sm, offset, &c);
110     pio_sm_set_enabled(pio, sm, true);
111 }
112
113 #endif
114
```

where the PIO program was located in PIO's 5-bit program address space



(public)

slow the state machine's execution down