3.2.2.2. State Machine Configuration

We have here a function ws2812_program_init which is provided to help the user to instantiate an instance of the LED driver program, based on a handful of parameters:

ws2812_program_init関数は、ユーザーがいくつかのパラメータに基づいてLEDドライバー・プログラムのインスタンスを生成するのを助けるために用意されている:

pio	Which of the PIO instances we are dealing withwith
sm	Which state machine on that PIO we want to configure to run the WS2812 programprogram
offset	Where the PIO program was loaded in PIO's 5-bit program address spacespace
pin	which GPIO pin our WS2812 LED chain is connected toto
freq	The frequency (or rather baud rate) we want to output data at.at.
rgbw	True if we are using 4-colour LEDs (red, green, blue, white) rather than the usual 3.3.
pio	どのPIOインスタンスを扱っているかどのPIOインスタンスを扱っているか
sm	WS2812プログラムを実行するために設定したいPIO上のステートマシンWS2812プログラムを実行するために設定したいPIO上のステートマシン
offset	PIOの5ビットプログラムアドレス空間でPIOプログラムがロードされた場所PIOの5ビットプログラムアドレス空間でPIOプログラムがロードされた場所
pin	WS2812 LEDチェーンがどのGPIOピンに接続されているかLEDチェーンがどのGPIOピンに接続されているか
freq	データを出力したい周波数(というかボーレート)。データを出力したい周波数(というかボーレート)。
rgbw	4色LED(赤、緑、青、白)を使う場合は真。4色LED(赤、緑、青、白)を使う場合は真。

Such that:

- pio_gpio_init(pio, pin); Configure a GPIO for use by PIO. (Set the GPIO function select.)

- pio_sm_set_consecutive_pindirs(pio, sm, pin, 1, true); Sets the PIO pin direction of 1 pin starting at pin number pin to out
- pio_sm_config c = ws2812_program_default_config(offset); Get the default configuration using the generated function for this program (this includes things like the .wrap and .side_set configurations from the program). We'll modify this configuration before loading it into the state machine.
- sm_config_set_sideset_pins(&c, pin); Sets the side-set to write to pins starting at pin pin (we say starting at because if you had .side_set 3, then it would be outputting values on numbers pin, pin+1, pin+2)
- sm_config_set_out_shift(&c, false, true, rgbw ? 32 : 24); False for shift_to_right (i.e. we want to shift out MSB first). True for autopull. 32 or 24 for the number of bits for the autopull threshold, i.e. the point at which the state machine triggers a refill of the OSR, depending on whether the LEDs are RGB or RGBW.
- int cycles_per_bit = ws2812_T1 + ws2812_T2 + ws2812_T3; This is the total number of execution cycles to output a single bit. Here we see the benefit of .define public; we can use the T1 - T3 values in our code.

このような

- pio_gpio_init(pio, pin); PIOが使用するGPIOを設定する。(GPIOのファンクション・セレクトを設定する。)
- pio_sm_set_consecutive_pindirs(pio, sm, pin, 1, true); ピン番号pinから始まる1ピンのPIOピンの方向をoutに設定する
- pio_sm_config c = ws2812_program_default_config(offset); このプログラム用に生成された関数を使用してデフォルト・コンフィギュレーションを取得する(これには、プログラムからの.wrapや.side_setコンフィギュレーションなどが含まれる)。ステートマシンにロードする前に、このコンフィギュレーションを修正する。
- sm_config_set_sideset_pins(&c, pin); ピンpinから始まるピンに書き込むサイドセットを設定する(ここから始まるというのは、.side_set 3を設定した場合、番号pin, pin+1, pin+2に値を出力することになるから)
- sm_config_set_out_shift(&c, false, true, rgbw ? 32 : 24); shift_to_rightの場合はfalse(つまり、MSBを最初にシフトアウトしたい)。オートプルの場合はtrue。オートプルしきい値のビット数は32または24(LEDがRGBかRGBWかによって、ステートマシンがOSRの再充填をトリガーするポイント)。
- int cycles_per_bit = ws2812_T1 + ws2812_T2 + ws2812_T3; これは、1ビットを出力するための総実行サイクル数です。ここで、.define publicの利点がわかる。T1〜T3の値をコードで使うことができる。