For example-
This is the verilog module corresponding to the peripheral:

```
// ****** MY PWM BLOCK ****** //
wire [15:0] gpio_15_0;
assign gpio_15_0 = {
                    gpio_15_out,
                    gpio_14_out,
                    io20_cell_out,
                    io19_cell_out,
                    io18_cell_out,
                    io17_cell_out,
                    io16_cell_out,
                    gpio_8_out,
                    gpio_7_out,
                    io13_cell_out,
                    io12_cell_out,
                    gpio_4_out,
                    io10_cell_out,
                    io9_cell_out,
                    io8_cell_out,
                    io7_cell_out
                    };

wire [15:0] gpio_31_16;
assign gpio_31_16 = {
                   gpio_31_out,
                   gpio_30_out,
                   gpio_29_out,
                   gpio_28_out,
                   gpio_27_out,
                   gpio_26_out,
                   gpio_25_out,
                   gpio_24_out,
                   gpio_23_out,
                   gpio_22_out,
                   gpio_21_out,
                   gpio_20_out,
                   gpio_19_out,
                   gpio_18_out,
                   gpio_17_out,
                   gpio_16_out
                   };

  mypwm pwm_gen(
```

```
                .clk(core_clk),
                .max_count(gpio_15_0),
                .cmp_val(gpio_31_16),
                .vout(VOUT)
               );
```

C Code:

```
 #include "platform.h"
#include "utils.h"
#include "gpio.h"

void togglegpio()
{
//Assumption 1 ---> output, 0 ---> input
write_word(GPIO_DIRECTION_CNTRL_REG, 0xFFFFEFFF); //GPIO0 to 23-> output

while (1) {

write_word(GPIO_DATA_REG, 0x08001000);
delay_loop(1000, 5000);
write_word(GPIO_DATA_REG, 0xFFFFFFFF);
delay_loop(1000, 5000);
}
}

void main()
{
togglegpio();
return 0;
}
```

platform.h:

```
/*!General Purpose Input / Output */
#define GPIO_START 0x00040100 //GPIO Start Address */
#define GPIO_OFFSET 0x08 /*!Generic offset used to access GPIO registers*/
#define PLIC_GPIO_OFFSET 6
```

gpio.h:

```
#define GPIO_DIRECTION_CNTRL_REG (uint32_t*) (GPIO_START  + (0 * GPIO_OFFSET ))
#define GPIO_DATA_REG (uint32_t*) (GPIO_START + (1 * GPIO_OFFSET ))
```