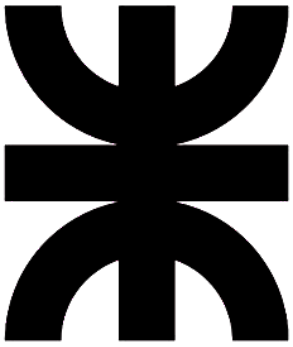
UNIVERSIDAD TECNOLÓGICA NACIONAL



FACULTAD REGIONAL PARANÁ

CARRERA: Ingeniería Electrónica
CÁTEDRA: Técnicas Digitales II

Trabajo Práctico N°10
Control de intensidad de LED con PWM

ALUMNOS:
Battaglia Carlo
Escobar Gabriel

Paraná, 8 de noviembre de 2022

Para implementar el sistema requerido, elegimos el *Timer/Counter1* de 16 bits que posee el *ATmega328p* para hacer uso de su función *Fast PWM*.

Configuramos este modo de manera que el ciclo de trabajo de la señal PWM esté dada por el valor almacenado en el registro *OCR1A* y el valor máximo del contador está dado por el registro *ICR1*. Esto nos permite elegir la resolución requerida en el enunciado (16 valores) fijando el máximo en 15 a través de *ICR1*.

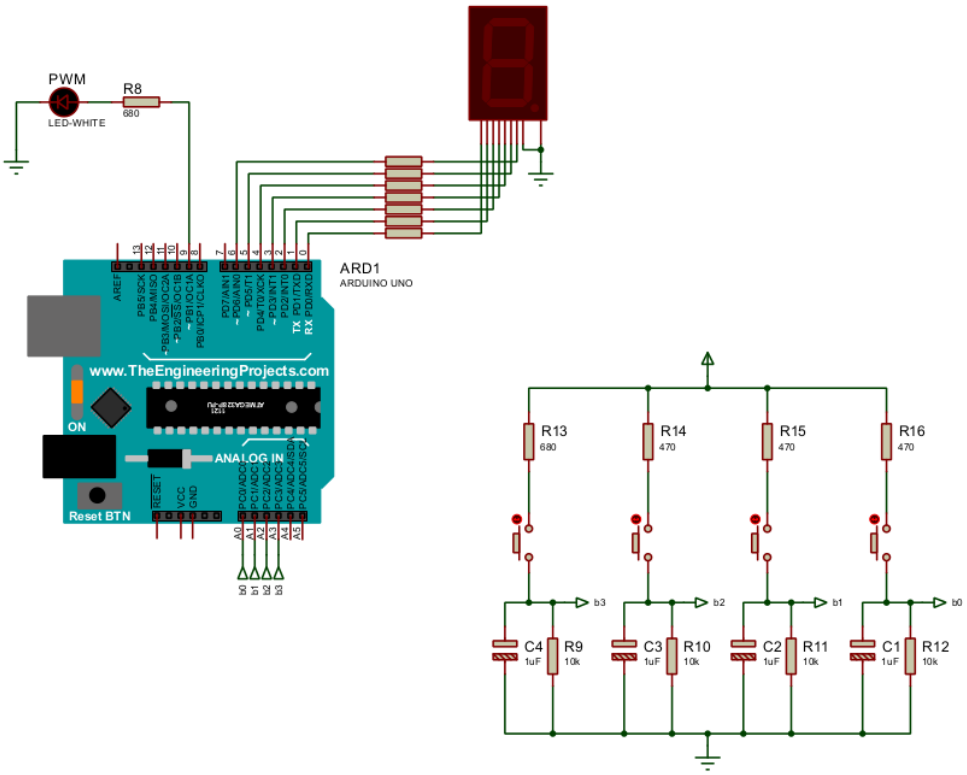
Además, para cumplir con la condición de obtener una señal completamente nula al ser el ciclo de trabajo igual a 0, configuramos el modo *Fast PWM* para hacer uso de la señal en modo invertido, tal como se ve en la siguiente tabla:

COM1A1/COM1B1	COM1A0/COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13:0 = 14 or 15: Toggle OC1A on compare match, OC1B disconnected (normal port operation). For all other WGM1 settings, normal port operation, OC1A/OC1B disconnected.
1	0	Clear OC1A/OC1B on compare match, set OC1A/OC1B at BOTTOM (non-inverting mode)
1	1	Set OC1A/OC1B on compare match, clear OC1A/OC1B at BOTTOM (inverting mode)

Esto es debido a que en el caso particular en que *OCR1A* es seteado al mismo valor que *BOTTOM (0x0000)* la salida no será una señal nula como habría de esperarse, sino que será un pico angosto con un ancho de un ciclo de timer cada vez que se alcance *TOP+1*.

Al tomar esta decisión de diseño, debimos tener en cuenta que los valores tomados directamente de los pulsadores leyendo el registro *PINC* estarán invertidos a fines prácticos, por lo que han de ser tratados mediante una operación *XOR* con un valor de *0x0F* antes de poder ser cargados en el registro *OCR1A*.

1. Circuito implementado



2. Código

```
;
; TP N°10.asm
; Authors : Battaglia Carlo y Escobar Gabriel
;

.cseg
.org 0x00
rjmp start

start:
clr r16
out SREG, r16      ;Reset status del sistema

ldi r16, LOW(RAMEND)     ;Inicio del stack pointer
out SPL, r16
ldi r16, HIGH(RAMEND)
```

```
17 out SPH, r16
18
19 clr r16
20 ldi r16, ( 1 << PB1 )
21 out DDRB, r16        ;Configuramos el PB1 como salida.
22
23 clr r16
24 sts TCNT1L, r16
25 sts TCNT1H, r16        ;Reseteamos el contador 1
26
27 clr r16
28 sts ICR1H, r16
29 ldi r16,15
30 sts ICR1L, r16         ;Seteamos el TOP del PWM (Resolución
      =16)
31
32 clr r16
33 ldi r16, (1 << COM1A1) | (1 << COM1A0) | (1 << WGM11) | (0 <<
       WGM10) ;Ponemos a BOTTOM el OCR1A cuando coincida el
      Compare Match y el TOP de Fast PWM dado por ICR1
34 sts TCCR1A, r16
35
36 clr r16
37 ldi r16, (1 << WGM13) | (1 << WGM12) | (0 << CS12) | (0 <<
      CS11) | ( 1 << CS10 ) ;Fast PWM: TOP: ICR1
38 sts TCCR1B, r16
39
40 clr r16
41 sts OCR1AH,r16
42 ldi r16, 15                  ;Inicio PWM en 0
43 sts OCR1AL,r16
44
45 ldi r16, (1 << DDD7) | (1 << DDD6) | (1 << DDD5) | (1 << DDD4
      ) | (1 << DDD3) | (1 << DDD2) | (1 << DDD1) | (1 << DDD0)
46 out DDRD, r16        ;PortD como salida
47 clr r16            ;Borro el PortD
48 out PORTD, r16
49
50 mainloop:
51 in r16, PINC
52 andi r16, 0x0F
53 ldi r17, 0x0F
54 eor r17, r16
55 sts OCR1AL, r17
56 call HEXTo7Segment
57 out PORTD, r16
58 rjmp mainloop        ;loop infinito
59
60 ;
```

```
61 ; HEXTo7Segment
62 ;
63 ; Convierte el valor, pasado en el registro r16, a una
       representación en display de 7 segmentos
64 ;
65
66 HEXTo7Segment:
67 push ZH
68 push ZL
69 ldi ZH,HIGH(2*BCDTo7Seg) ; Carga la tabla
70 ldi ZL,LOW(2*BCDTo7Seg)
71 add ZL,r16
72 lpm
73 mov r16,R0
74 pop ZL
75 pop ZH
76 ret
77
78
79 ; Tabla de conversión decimal a 7 segmentos
80
81 HEXTo7Seg:
82 .db 0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F,0x77,0
       x7C,0x39,0x5E,0x79,0x71
```