FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

Laboratorio de Microprocesadores - 86.07

PWM (Pulse Width Modulation)

Profesor:			Ing. Guillermo Campiglio								
Cuatrimestre/Año:			1º/2020								
Turno de las clases prácticas			Miercoles 19 hs								
Jefe de trabajos prácticos:			Pedro Ignacio Martos								
Docente guía:			Pedro Martos, Fabricio Baglivo, Fernando Pucci								
Autores			Seguimiento del proyecto								
Nombre	Apellido	Padrón									
Leonel	Mendoza	101153									

Observaciones:

.....
.....
.....
.....
.....
.....
.....
.....
.....

Fecha de aprobación				Firma J.T.P

Coloquio	
Nota final	
Firma profesor	

1. Objetivo

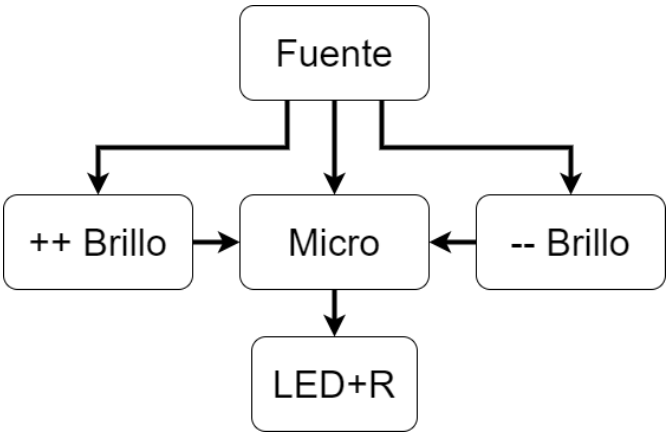
El objetivo de este trabajo es manejar los registros de timers, en particular el modo de PWM (Pulse Width Modulation) para manejar dispositivos que requieran de una tensión específica, pero que aun así se necesite entregar mas o menos potencia (caso de motores eléctricos, LEDs, etc.).

2. Descripción

Se pide hacer un programa que aumente y disminuya el brillo de un LED. Para eso se dispone de 2 pulsadores (UP, DOWN) y un LED como se indicará en el esquemático.

Se deberá usar el PWM, o modulación por ancho de pulso, la cual consiste en modificar el ciclo de trabajo de una señal (sin modificar la frecuencia). Con esta señal se alimentará el LED, de forma que el valor medio de la señal será proporcional al brillo del LED, a mayor ancho de pulso, más brillo.

3. Diagrama en bloques



4. Esquemático

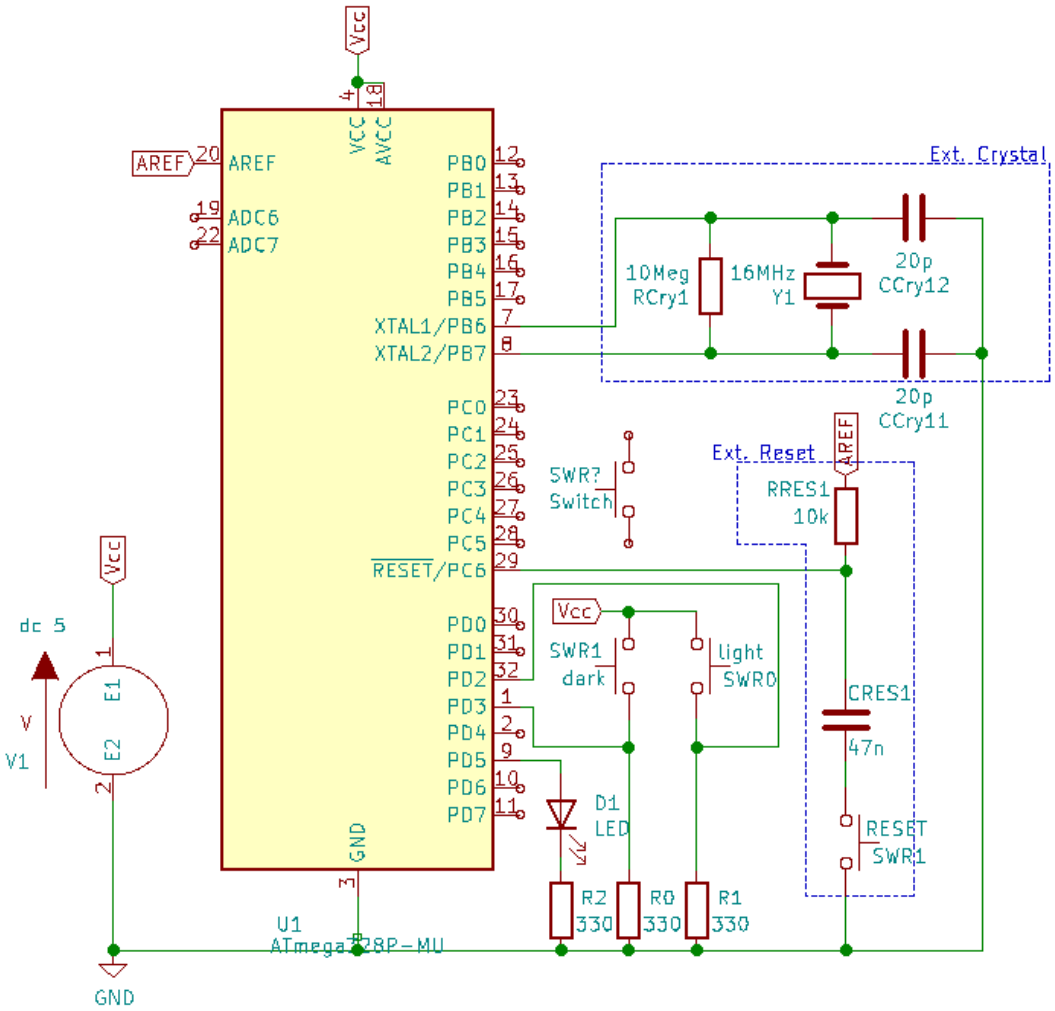
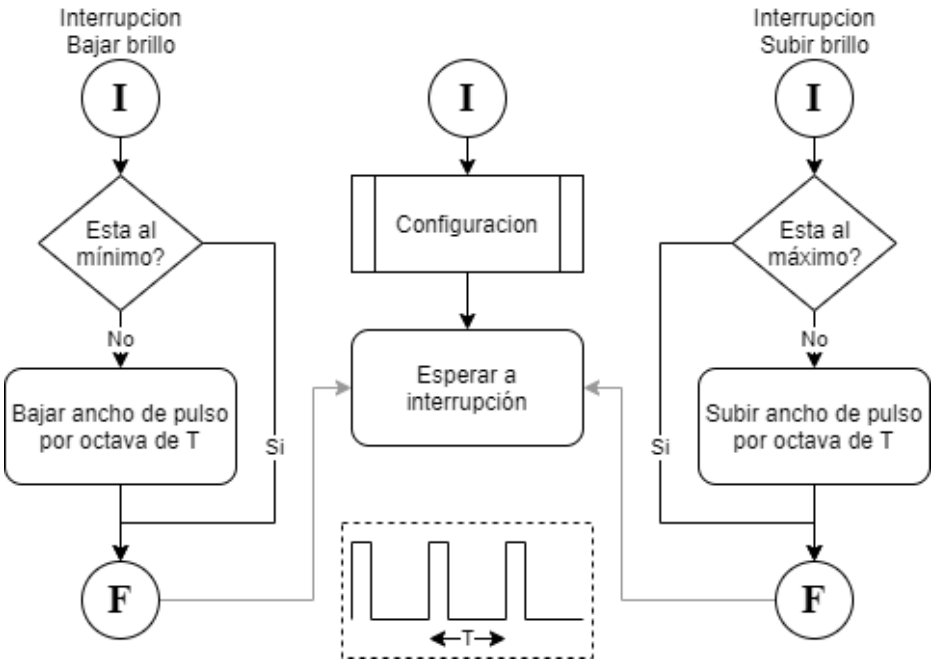


Figura 1: Esquemático del circuito

5. Listado de componentes

- Microcontrolador *ATmega328p* y programador USBasp (Arduino UNO) [AR$ 950]
- 1x LED [AR$ 10]
- 3x Resistencia (330 Ω) [AR$ 12]
- 1x Pulsador (10 $k\Omega$) [AR$ 15]

6. Diagrama de Flujo



7. Código de programa

```
.include "m328pdef.inc"

; * * * * * * * * * *
;    START MACROS    ;
; * * * * * * * * * *

.MACRO SET_SP ;[auxGPR]
    LDI @0, low(RAMEND)
    OUT SPL, @0
    LDI @0, high(RAMEND)
    OUT SPH, @0
.ENDM

.MACRO SET_X ;[LABEL to data memory]
    LDI XL, low(@0)
    LDI XH, high(@0)
.ENDM

.MACRO SET_Y ;[LABEL to data memory]
    LDI YL, low(@0)
    LDI YH, high(@0)
.ENDM

.MACRO SET_Z ;[LABEL to prog memory]
    LDI ZL, low(@0 << 1)
    LDI ZH, high(@0 << 1)
.ENDM
```

```
; * * * * * * * * * *
;    END MACROS     ;
; * * * * * * * * * *



.DEF aux = R16
.DEF counter = R17

.CSEG

    .ORG 0X0000 ; En esta direccion escribo la instruccion JMP conf
    JMP conf

    .ORG INT0addr
    JMP isr_lower       ; interrupcion para disminuir brillo

    .ORG INT1addr
    JMP isr_higher      ; interrupcion para aumentar brillo

    .ORG INT_VECTORS_SIZE   ; Direccion donde escribir el codigo

conf:
    SET_SP  aux

    LDI     aux, 0b01000000     ; SALIDA al LED por PD6 (OC0A)
    OUT     DDRD, aux   ; ENTRADA de los dos pulsadores por PD2 y PD3 (interrupciones)

; compare A enabled (noninverting), B disabled, WGM11/10 Fast PWM 0xFF
    LDI     aux, 0b10000011
    OUT     TCCR0A, aux
; Input Capture Noise Cancel 0, Input Capture Edge 0, WGM12 Fast PWM 0xFF
    LDI     aux, 0b00000001
    OUT     TCCR0B, aux     ; CS clock normal

    LDI     aux, 0b00001111     ; Configuro interrupciones (ambas por ascendente)
    STS     EICRA, aux
    LDI     aux, 0b00000011     ; Habilito dos INT
    OUT     EIMSK, aux

    LDI     aux, 127    ; pongo el brillo en la mitad
    OUT     OCR0A, aux

    SEI     ; Habilito interrupciones

main:
    NOP
    RJMP main

isr_lower:
    IN      aux, OCR0A
    CPI     aux, 0
    BREQ    endl    ; si es 0 no puedo bajar mas el brillo, dejo como esta
    LDI     counter, 32
```

```
loopl:          ; el loop ademas de incrementar sirve de delay,
    DEC     aux     ; para no triggerear la isr mas de una vez
    BREQ    endl
    DEC     counter
    BRNE    loopl
endl:
    OUT     OCR0A, aux
    RETI

isr_higher:
    IN      aux, OCR0A
    CPI     aux, 255
    BREQ    endh    ; si es 255 no puedo subir mas el brillo, dejo como esta
    LDI     counter, 32
looph:          ; el loop ademas de incrementar sirve de delay,
    INC     aux     ; para no triggerear la isr mas de una vez
    BREQ    ovf
    DEC     counter
    BRNE    looph
endh:
    OUT     OCR0A, aux
    RETI
ovf:    ; si llega a 0 despues de incrementar, se paso
    DEC aux     ; decremento para volver a 255
    RJMP endh
```

8. Resultados

Se logro controlar el brillo del LED mediante uso de PWM (en particular Fast PWM), el uso de pulsadores es ruidoso y dificulta la prueba del software. No obstante, se probó una cantidad considerable de veces para confirmar que se estaban ejecutando correctamente las rutinas de interrupción.

9. Conclusiones

Mediante la configuración de los registros de timers, en particular el modo de operación, se pudo variar el brillo de un LED. Usando la salida del timer en el modo de PWM directamente al LED. De esta forma se entendió el funcionamiento de PWM y la aplicación a dispositivos que necesiten una tensión dada para su correcto funcionamiento (en el caso del led $\approx$2.7 V) y aun asi poder entregar menos potencia para reducir la intensidad con la que opera (en este caso luminosidad, en otro caso e.g. motores, el torque)