Fiuba
Facultad de Ingeniería
Universidad de Buenos Aires

Laboratorio de Microprocesadores - 86.07

Trabajo Práctico Nº7:
PWM

Profesor:			Ing. Guillermo Campiglio									
Cuatrimestre/Año:			1º/2020									
Turno de las clases prácticas			Miércoles									
Jefe de trabajos prácticos:			Ing. Pedro Ignacio Martos									
Docentes guía:			Ing. Fabricio Baglivo, Ing. Fernando Pucci									
Fecha de presentación:			12/08/2020									
Alumno			Seguimiento del proyecto									
Nombre	Apellido	Padrón										
Maximiliano Daniel	Reigada	100565										

Observaciones:

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

Fecha de aprobación			Firma J.T.P

Índice

1. Objetivo

El presente trabajo práctico tiene como objetivo progresar con el manejo de registros de timers de los que dispone el microcontrolador ATmega328P, utilizando la modalidad de PWM o modulación de ancho de pulso para controlar el brillo en el encendido de un LED.

2. Descripción

Mediante la implementación de una placa de microcontrolador de código abierto basado en el microchip ATmega328P, denominada Arduino Uno, se busca desarrollar un programa para aumentar o disminuir el brillo de un LED conectado al pin `PB1` del puerto B configurado como salida, a partir de dos pulsadores (UP, DOWN) conectados a los pines `PD0` y `PD1` del puerto D configurado como entrada.

Para esta tarea, se utiliza el modo PWM rápido de 8 bits del que dispone el timer1, seteando al clock sin prescaler. Se verifica mediante polling el estado en que se encuentra cada pulsador, en caso de que UP o DOWN se encuentre presionado, se incrementa o disminuye respectivamente, el valor del registro `OCR1AL`. Este valor es proporcional al ancho de pulso de la señal que se emite por el pin `OC1A` al que se encuentra conectado el LED, de manera que se modifica el ciclo de trabajo de la señal (sin alterar la frecuencia), obteniendo así que el valor medio de ésta es proporcional al brillo del LED.

3. Diagrama de conexiones en bloques

A continuación, puede verse el diagrama de conexiones en bloques correspondiente a este trabajo práctico:

Figura 1: Diagrama de conexiones en bloques.

4. Circuito esquemático

Se procede a realizar el esquemático del circuito a implementar. Cabe destacar que, si bien en la realidad los componentes se encuentran conectado al Arduino Uno y este a su vez a una fuente de tensión externa, resulta más útil representar en el esquema solo los componentes electrónicos esenciales del sistema, conectados a referencias de tensión de un determinado valor:

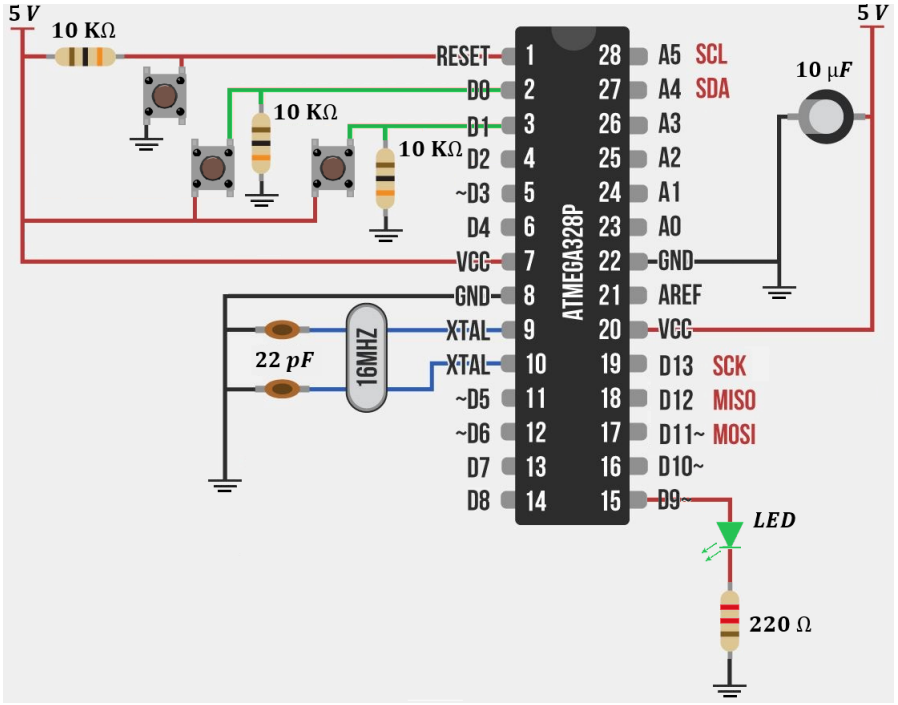

Figura 2: Circuito esquemático.

5. Listado de componentes

En el desarrollo de este trabajo práctico se utilizan los siguientes componentes:

- 1 LED verde de 5mm
- 1 Resistor de 220 Ω
- 2 Resistores de 10 kΩ
- 2 Pulsadores Dip Tact Switch 6x6x5mm
- 6 Cables de conexión
- 1 Protoboard
- 1 Arduino Uno (con microcontrolador ATmega328P incluido)

6. Diagrama de flujo del Software

A continuación, se explicitan los diagramas de flujo correspondientes al código desarrollado:

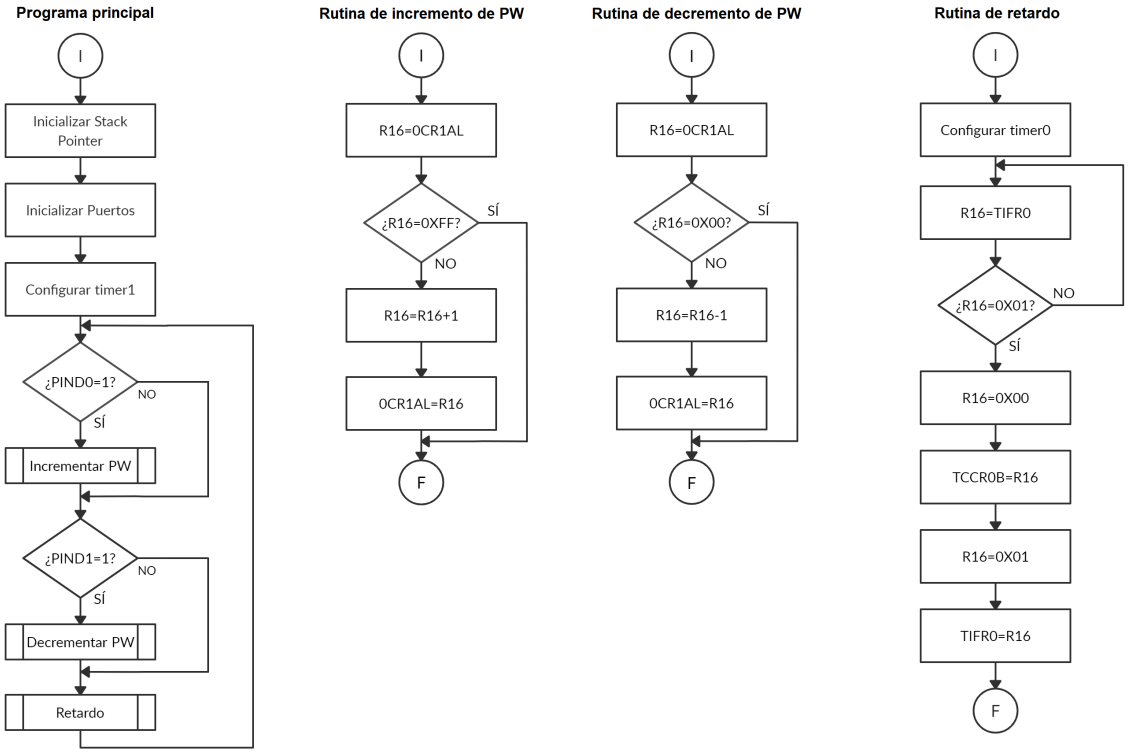

Figura 3: Diagramas de flujo.

7. Código de programa

A continuación, puede verse el código desarrollado para este proyecto:

```
;***********************************************************************
; Código correspondiente al ejercicio planteado en el Trabajo Práctico 7.
;
; Alumno: Reigada Maximiliano Daniel
; Padrón: 100565
;***********************************************************************

.INCLUDE "m328pdef.inc"

.DEF AUX=R16

.CSEG
.ORG 0X0000
      RJMP  config


.ORG INT_VECTORS_SIZE

config:
      LDI    AUX, HIGH(RAMEND)              ;Inicializo el SP al final de la RAM.
      OUT    SPH, AUX
      LDI    AUX, LOW(RAMEND)
      OUT    SPL, AUX

      LDI    AUX, 0X00                      ;Declaro al puerto D como entrada.
      OUT    DDRD, AUX

      LDI    AUX, 0XFF                      ;Declaro al puerto B como salida.
      OUT    DDRB, AUX

      LDI    AUX, (1 << COM1A1 | 1 << WGM10) ;Configuro el timer1 en modo PWM rápido de
      STS    TCCR1A, AUX                    ;8 bits y seteo al clock sin prescaler.

      LDI    AUX, (1 << WGM12 | 1 << CS10)
      STS    TCCR1B, AUX

main:
      SBIC   PIND, 0               ;Si PIND0=1, incremento el ancho de pulso.
      RCALL  incrementar_PW

      SBIC   PIND, 1               ;Si PIND1=1, decremento el ancho de pulso.
      RCALL  decrementar_PW

      RCALL  retardo               ;Ejecuto un retardo de aproximadamente 16ms para que
                                   ;la transición de los estados del LED sea visible.
      RJMP   main

incrementar_PW:
      LDS    AUX, OCR1AL
      CPI    AUX, 0XFF             ;Si OCR1AL=255 retorno porque el ancho de pulso llegó
      BREQ   retorno_inc           ;a su máximo y no es posible seguir incrementándolo.

      INC    AUX
      STS    OCR1AL, AUX

retorno_inc:
      RET

decrementar_PW:
      LDS    AUX, OCR1AL           ;Si OCR1AL=0 retorno porque el ancho de pulso llegó
      CPI    AUX, 0X00             ;a su mínimo y no es posible seguir decrementándolo.
      BREQ   retorno_dec

      DEC    AUX
      STS    OCR1AL, AUX

retorno_dec:
      RET
```

```
69
70  retardo:
71        LDI     AUX, 0X00                     ;Me aseguro de iniciar el registro TCNT0 en 0.
72        OUT     TCNT0, AUX
73
74        LDI     AUX, (1 << CS02 | 1 << CS00) ;Configuro el timer0 en modo normal
75        OUT     TCCR0B, AUX                   ;y seteo al clock con prescaler 1024.
76
77  loop_retardo:
78        IN      AUX, TIFR0                    ;En caso de que se active el flag de overflow
79        SBRS    AUX, TOV0                     ;del timer0, esquivo la próxima instrucción.
80        RJMP    loop_retardo
81
82        LDI     AUX, 0X00
83        OUT     TCCR0B, AUX                   ;Desactivo el timer0.
84        LDI     AUX, (1<<TOV0)
85        OUT     TIFR0, AUX                    ;Limpio el flag de overflow del timer0.
86
87        RET
```

8. Resultados

Luego de desarrollar el código descripto en las secciones anteriores y armar el circuito especificado, se logra controlar el brillo de un LED conectado al pin `PB1` a partir de pulsadores dispuestos en los pines `PD0` y `PD1`, utilizando el modo PWM rápido de 8 bits del que dispone el timer1.

Cabe destacar, que entre cada lectura de los puertos de entrada se debe ejecutar un retardo para que la transición del punto más brilloso al apagado del LED, y viceversa, sea perceptible y no suceda demasiado rápido. Este retardo se decide por implementarlo utilizando el timer0 del que dispone el microcontrolador.

Finalmente, se procede a realizar un listado de los componentes utilizados y sus costos:

Componentes	Costos
Led verde de 5mm	$7,00
Pulsadores Dip Tact Switch	$44,00
Resistor de 220 Ω	$5,50
Resistores de 10kΩ	$11,00
Cables de conexión	$12,00
Protoboard	$232,00
Arduino Uno	$720,00
TOTAL	$1031,50

9. Conclusiones

Tras haber realizado todos los pasos pedidos en el enunciado de este trabajo práctico, resta destacar las conclusiones que la experiencia ha aportado.

A partir de este trabajo se logra afianzar el manejo de timers del que puede disponer un microcontrolador, poniendo en práctica la modulación del ancho de pulso de una señal para controlar el brillo aparente en un LED.

Es necesario hacer énfasis en todos los parámetros configurables del timer1 de los que se tuvo que tomar nota y especificar para esta práctica en particular a fin de hacerlo funcionar en modo PWM rápido de 8 bits. Así mismo, se logra indagar en algunas de las opciones de uso que en la práctica anterior no se tuvieron en cuenta al no haber sido implementadas.