

19/May/2020

# TRABAJO PRÁCTICO N° 1

## Parpadeo de un LED

---

98934 Vazquez, Rodrigo rvazquez@fi.uba.ar

### Docentes:

Campiglio, Guillermo Carlos  
Stola, Gerardo Luis  
Cofman, Fernando  
Salaya Velazquez, Juan Guido

---

### Resumen

En el presente trabajo se desarrollará el diseño de programa de un parpadeo de LED sobre el microcontrolador *Atmel 328P* sobre la plataforma *Arduino*. Luego se armará el circuito de testeo apropiado utilizando un *Shield I/O* con los elementos pertinentes al TP.

---

## Índice

1. Objetivo	2
2. Descripción del proyecto	2
3. Diagrama de conexiones en bloques	2
4. Circuito esquemático	3
5. Listado de componentes y tabla de gastos	3
6. Software	3
7. Conclusiones	4
8. Anexo - Código	5
9. Anexo - Esquemático Shield I/O	5

## 1. Objetivo

Manejar los puertos de salida del microcontrolador *Atmel 328P* para hacer parpadear un LED.

## 2. Descripción del proyecto

Se realizará un programa en lenguaje *Assembly* para la familia de microcontroladores *AVR*, con la salvedad que en mi caso, conectaré el LED en el pin PB3. Esto se debe a que en conjunto con la placa **Arduino UNO**, utilizo un accesorio tipo *Shield*, con conexiones predeterminadas, una de ellas siendo un LED en el pin 3 del puerto B.

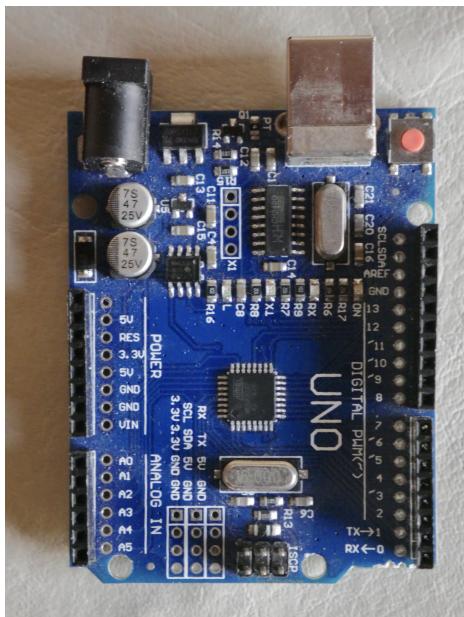


Figura 1: Placa Arduino UNO utilizada



Figura 2: Placa *Shield I/O* utilizada para la conexión LED

## 3. Diagrama de conexiones en bloques

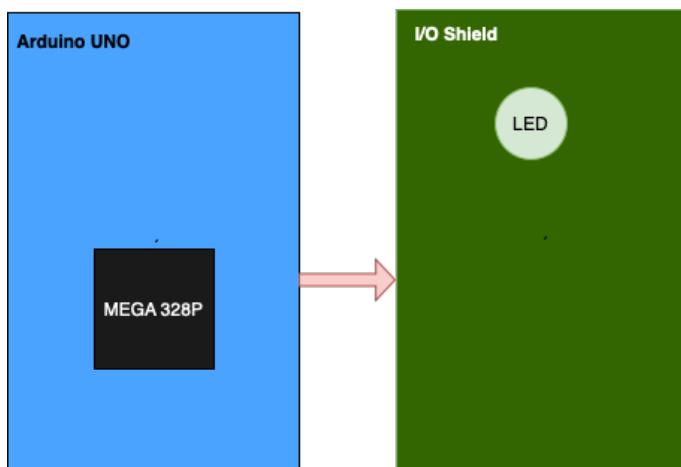


Figura 3: Diagrama en bloques del hardware

## 4. Circuito esquemático

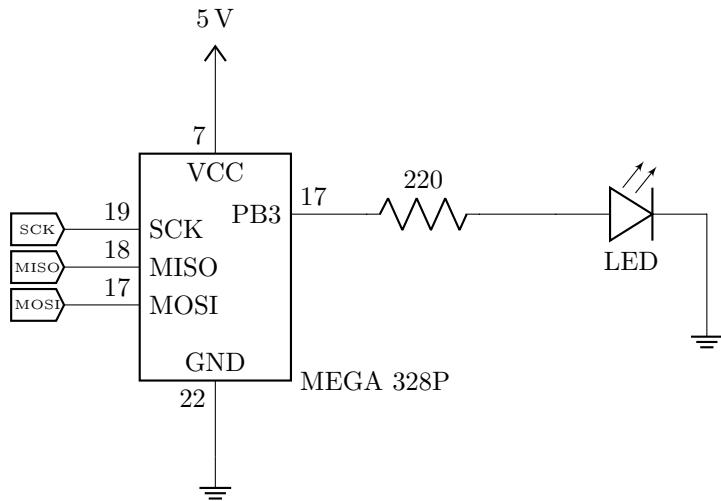


Figura 4: Circuito para parpadeo de LED

## 5. Listado de componentes y tabla de gastos

Componente	Precio
Arduino UNO	\$750
I/O Shield	\$950

Tabla 1: Tabla de componentes

## 6. Software

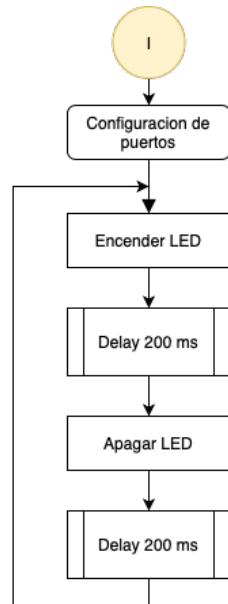


Figura 5: Diagrama de flujo del programa para parpadear un LED

El programa (anexado al final del documento) comienza definiendo el principio de la memoria de

programa via la directiva `.org`, y luego con la configuracion del PIN 3 del puerto B como salida, esto se realiza cargando un 1 en el tercer bit del registro DDRB. Para poder utilizar las denominaciones de los puertos en vez de sus posiciones en memoria, se debe agregar la biblioteca `m328Pdef.inc` la cual contiene todas las definiciones necesarias para este microcontrolador.

El software implementado tambien toma codificaciones hechas en ejercicios previos como la subrutina `DELAY_200`, esta actua como espera de tiempo 200 ms. Dicha subrutina se utilizó entre el encendido y apagado del LED, hecho con la instrucción `out`. Para que permanezca la luz encendida durante 200 ms.

## Resultados

Como el programa fue escrito en el editor de texto *Atom*, todos los pasos posteriores a la escritura del programa se hicieron por *Terminal*. Para el ensamble del programa se utilizó el programa por linea de comandos *avr*, este produce el archivo hexadecimal que luego se subirá al Arduino. Para hacer esto, se utilizó el programa *avrdude*, el comando completo es el siguiente

```
avrdude -p m328p -c stk500v1 -b 115200 -F -P /dev/cu.wchusbserial1410 -U  
flash:w:TP1_RodrigoVazquez.hex
```

La definicion del flag de puerto es tal que describa donde se encuentra un puerto serie en una computadora *Mac*. Luego de este proceso, se puede observar los resultados obtenidos en el siguiente video

## 7. Conclusiones

Las principales conclusiones de este trabajo recaen en el manejo apropiado de los puertos de entrada y salida del microcontrolador Atmel Mega 328P via el lenguaje de *assembly*, para ello siendo necesario un profundo conocimiento de la memoria del microcontrolador y de las instrucciones la administran.

## 8. Anexo - Código

```
.include "m328Pdef.inc"
.org $00
. equ LED_PIN = 0b00001000
; conecto led en PB3
sbi DDRB, 3 ; declaro a PB3 como salida
```

LOOP:

```
ldi r20, LED_PIN
out PORTB, r20 ; enciendo LED
call DELAY_200
ldi r20, 0b00000000
out PORTB, r20 ; apago LED
call DELAY_200
rjmp LOOP
```

DELAY\_200:

```
ldi r18, 17
ldi r19, 60
ldi r20, 204
```

L1:

```
dec r20
brne L1
dec r19
brne L1
dec r18
brne L1
ret
```

## 9. Anexo - Esquemático Shield I/O

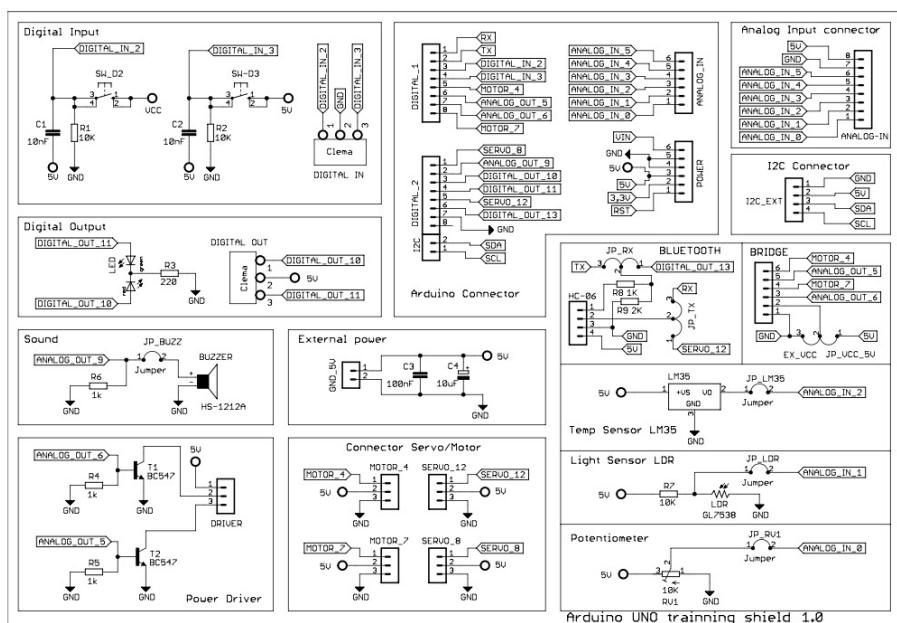


Figura 6: Circuito esquemático Shield I/O