FACULTAD
DE INGENIERIA
Universidad de Buenos Aires

Laboratorio de Microprocesadores - 86.07

Parpadeo de un LED

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	

Índice

1. Objetivo del Trabajo

El objetivo del siguiente trabajo es manejar los puertos de un Arduino (programado en Assembly), para encender y apagar un LED.

2. Descripción del Trabajo

Se realiza un programa en *Assembly*, que hace parpadear un LED alimentado y controlado por un Arduino UNO. Este programa hace parpadear al LED de dos formas distintas (a nivel programación, ya que en la practica no se van a notar mayores diferencias), mediante el *seteo* de todo el puerto a utilizar y el de un solo bit perteneciente al puerto (pin al que estara conectado el LED).

3. Diagrama en Bloques

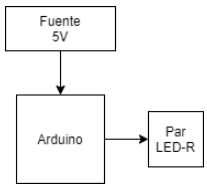

Figura 1: Diagrama en bloques

4. Circuito Esquemático

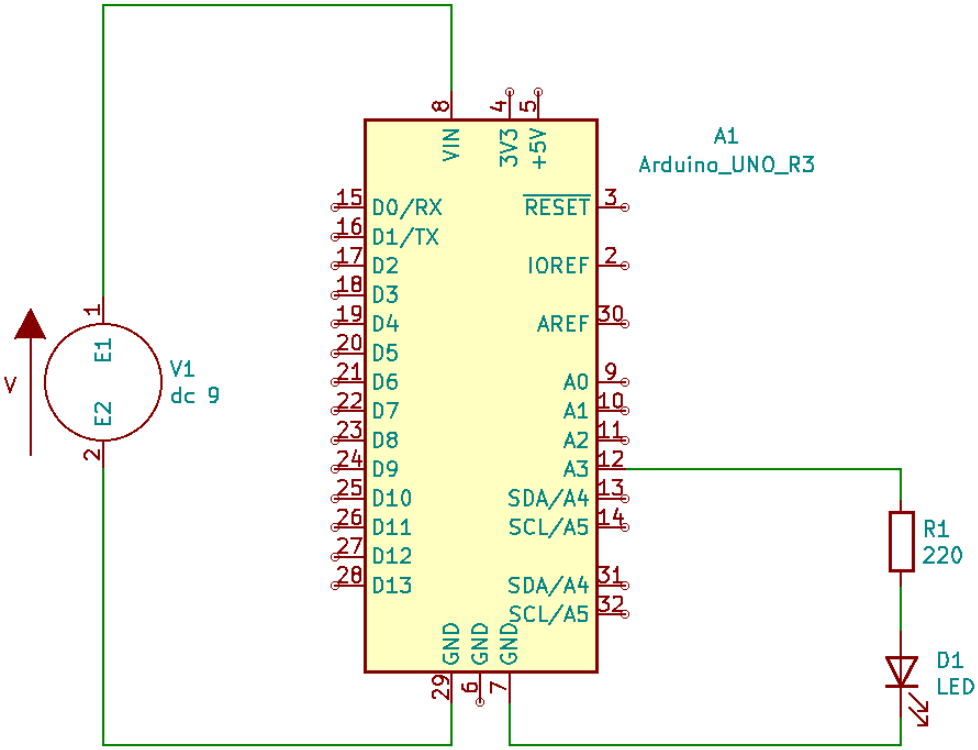

Figura 2: Esquemático del circuito

5. Listado de componentes

- Microcontrolador *ATmega328p* y programador USBasp (Arduino UNO)
- LED
- Resistencia (220 Ω)

6. Diagrama de flujo

A continuación un diagrama de flujo del programa.

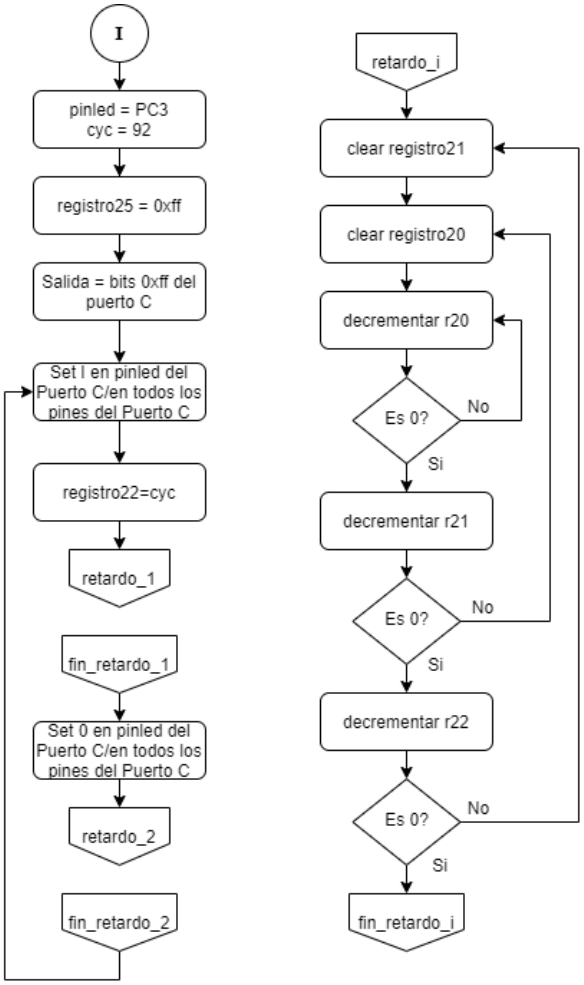

Figura 3: Diagrama de flujo del programa.

7. Código de programa

7.1. Código del primer método

El primer método consiste en prender solo un bit del puerto usado para el LED.

```
;
; tp1.asm
;
; Created: 5/19/2020 22:56:42
; Author : LeoMendU
;

.include "m328pdef.inc"

;defino variables
.equ pinled = PC3
.equ cyc = 92

.cseg

jmp main

main:
ldi r25,0xff
clr r24
out DDRC,r25 ;pongo el portc como salida

loopled:
sbi PORTC,pinled ;encendido del led
ldi r22,cyc ;uso este tiempo para el prox retardo
call retardador22
cbi PORTC,pinled ;cleareo el pin para apagar el led
ldi r22,cyc
call retardador22
rjmp loopled

retardador22: ; delay de aproximadamente 10.9 ms * R22
clr r21
loopretard1:
clr r20
loopretard2:
dec r20 ;cuento 256 veces con R20
brne loopretard2
dec r21 ;cuento 256 veces el conteo de R20 (hasta ahora 256*256)
brne loopretard1
dec r22 ;cuento "R22" veces el conteo de R21 (R22*256*256)
brne retardador22
ret
```

7.2. Código del segundo método

El segundo método consiste en prender todo el puerto usado para el LED.

```
;
; tp1.asm
;
; Created: 5/19/2020 22:56:42
; Author : LeoMendU
;

.include "m328pdef.inc"

;defino variables
.equ pinled = PC3
.equ cyc = 92

.cseg

jmp main

main:
ldi r25,0xff
clr r24
ser r23
out DDRC,r25 ;pongo el portc como salida

loopled:
out PORTC,r23 ;encendido del led (prendo todo el puerto)
ldi r22,cyc ;uso este tiempo para el prox retardo
call retardador22
out PORTC,r24 ;limpio el puerto para apagar el led
ldi r22,cyc
call retardador22
rjmp loopled

retardador22: ; delay de aproximadamente 10.9 ms * R22
clr r21
loopretard1:
clr r20
loopretard2:
dec r20 ;cuento 256 veces con R20
brne loopretard2
dec r21 ;cuento 256 veces el conteo de R20 (hasta ahora 256*256)
brne loopretard1
dec r22 ;cuento "R22" veces el conteo de R21 (R22*256*256)
brne retardador22
ret
```

8. Resultados

Se observo que el LED parpadeaba correctamente cada 1 seg (un segundo encendido y uno apagado), cuando se uso el primer método, solo funcionaba cuando se lo conectaba en el pin 3 del puerto C (o el A señalado en el Arduino de la figura 4); en cambio cuando se uso el segundo método, funcionaba en cualquier pin del puerto C.

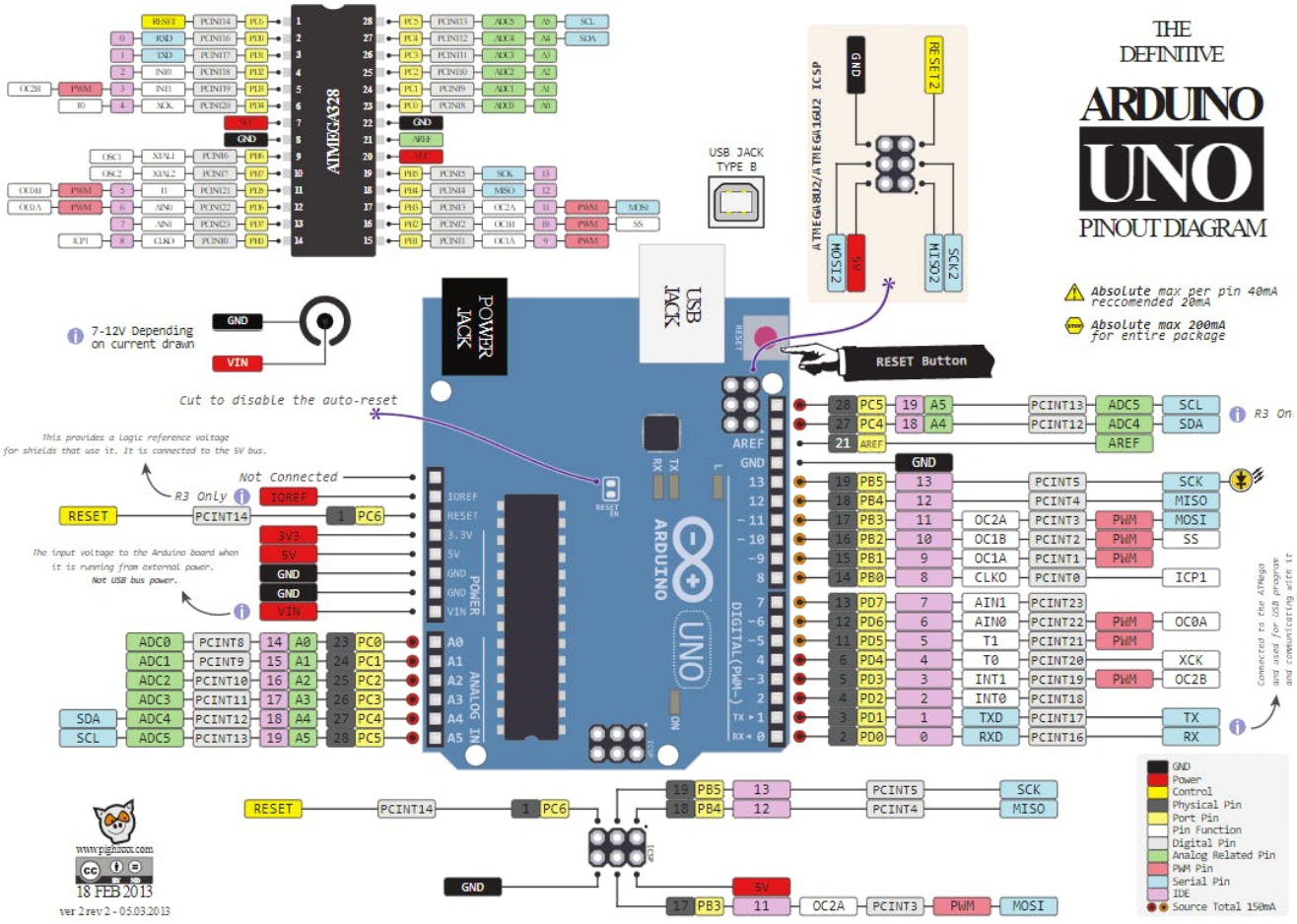

Figura 4: Pinout del arduino

9. Conclusiones

Para poder hacer que parpadee cada un segundo, se midió el tiempo del retardo (siendo de 10.9 ms aproximadamente), y con la frecuencia del clock del microcontrolador se dedujo la cantidad de ciclos que tarda la subrutina (173920) que es del orden del valor que da el debugger de AtmelStudio (197394), de esta forma se puede ver cuanto tiempo tarda en hacerse una operación (brne: 2 ciclos, dec: 1 ciclo, clr: 1 ciclo, etc.)

Al probar los dos métodos distintos, se explicita la diferencia entre sbi y out, y sobre que registros actúan (además de la diferencia entre DDRX, PINX y PORTX).

Por último, el parpadeo hace necesario que se emplee un delay de algún tipo, por lo que se uso una subrutina, haciendo el código mas ordenado y entendible, además de que sean intuitivas desde el comienzo del curso.